UM10503

LPC43xx ARM Cortex-M4/M0 dual-core microcontroller

Rev. 1 — 12 December 2011 **Preliminary user manual**

Document information

Info	Content
Keywords	LPC43xx, LPC4350, LPC4330, LPC4320, LPC4310, LPC4357, LPC4353, LPC4337, LPC4333, LPC4327, LPC4325, LPC4323, LPC4322, LPC4317, LPC4315, LPC4313, LPC4312, LPC4310, ARM Cortex-M4, ARM Cortex-M0, SPIFI, SCT, USB, Ethernet
Abstract	LPC4300 preliminary user manual

Revision history

Rev	Date	Description
1	20111212	Preliminary LPC43xx User manual.

Contact information

For more information, please visit: **http://www.nxp.com**

For sales office addresses, please send an email to: **salesaddresses@nxp.com**

1.1 Introduction

The LPC43xx are ARM Cortex-M4 based microcontrollers for embedded applications which include an ARM Cortex-M0 coprocessor, up to 264 kB of SRAM, advanced configurable peripherals such as the State Configurable Timer (SCT) and the Serial General Purpose I/O (SGPIO) interface, two High-speed USB controllers, Ethernet, LCD, an external memory controller, and multiple digital and analog peripherals. The LPC43xx operate at CPU frequencies of up to 204 MHz.

The ARM Cortex-M4 is a next generation 32-bit core that offers system enhancements such as low power consumption, enhanced debug features, and a high level of support block integration. The ARM Cortex-M4 CPU incorporates a 3-stage pipeline, uses a Harvard architecture with separate local instruction and data buses as well as a third bus for peripherals, and includes an internal prefetch unit that supports speculative branching. The ARM Cortex-M4 supports single-cycle digital signal processing and SIMD instructions. A hardware floating-point processor is integrated in the core.

The ARM Cortex-M0 coprocessor is an energy-efficient and easy-to-use 32-bit core which is code- and tool-compatible with the Cortex-M4 core. The Cortex-M0 coprocessor, designed as a replacement for existing 8/16-bit microcontrollers, offers up to 204 MHz performance with a simple instruction set and reduced code size.

1.2 Features

- Cortex-M4 Processor core
  - ARM Cortex-M4 processor, running at frequencies of up to 204 MHz.
  - ARM Cortex-M4 built-in Memory Protection Unit (MPU) supporting eight regions.
  - ARM Cortex-M4 built-in Nested Vectored Interrupt Controller (NVIC).
  - Hardware floating-point unit.
  - Non-maskable Interrupt (NMI) input.
  - JTAG and Serial Wire Debug (SWD), serial trace, eight breakpoints, and four watch points.
  - Enhanced Trace Module (ETM) and Enhanced Trace Buffer (ETB) support.
  - System tick timer.
- Cortex-M0 Processor core
  - ARM Cortex-M0 co-processor capable of off-loading the main ARM Cortex-M4 application processor.
  - Running at frequencies of up to 204 MHz.
  - JTAG, Serial Wire Debug, and built-in NVIC.
- On-chip memory
  - Up to 264 kB SRAM for code and data use.

- Multiple SRAM blocks with separate bus access. Two SRAM blocks can be powered down individually.
- 64 kB ROM containing boot code and on-chip software drivers.
- 128 bit general-purpose One-Time Programmable (OTP) memory.

- Configurable digital peripherals
  - Serial GPIO (SGPIO) interface.
  - State Configurable Timer (SCT) subsystem on AHB.
  - Global Input Multiplexer Array (GIMA) allows to cross-connect multiple inputs and outputs to event driven peripherals like the timers, SCT, and ADC0/1.

- Serial interfaces
  - Quad SPI Flash Interface (SPIFI) with 1-, 2-, or 4-bit data at rates of up to 40 MB per second.
  - 10/100T Ethernet MAC with RMII and MII interfaces and DMA support for high throughput at low CPU load. Support for IEEE 1588 time stamping/advanced time stamping (IEEE 1588-2008 v2).
  - One High-speed USB 2.0 Host/Device/OTG interface with DMA support and on-chip high-speed PHY.
  - One High-speed USB 2.0 Host/Device interface with DMA support, on-chip full-speed PHY and ULPI interface to external high-speed PHY.
  - USB interface electrical test software included in ROM USB stack.
  - One 550 UART with DMA support and full modem interface.
  - Three 550 USARTs with DMA and synchronous mode support and a smart card interface conforming to ISO7816 specification. One USART with IrDA interface.
  - Two C_CAN 2.0B controllers with one channel each.
  - Two SSP controllers with FIFO and multi-protocol support. Both SSPs with DMA support.
  - One SPI controller.
  - One Fast-mode Plus I^2C-bus interface with monitor mode and with open-drain I/O pins conforming to the full I^2C-bus specification. Supports data rates of up to 1 Mbit/s.
  - One standard I^2C-bus interface with monitor mode and with standard I/O pins.
  - Two I^2S interfaces, each with DMA support and with one input and one output.

- Digital peripherals
  - External Memory Controller (EMC) supporting external SRAM, ROM, NOR flash, and SDRAM devices.
  - LCD controller with DMA support and a programmable display resolution of up to 1024H $\times$ 768V. Supports monochrome and color STN panels and TFT color panels; supports 1/2/4/8 bpp Color Look-Up Table (CLUT) and 16/24-bit direct pixel mapping.
  - Secure Digital Input Output (SD/MMC) card interface.
  - Eight-channel General-Purpose DMA (GPDMA) controller can access all memories on the AHB and all DMA-capable AHB slaves.

- Up to 164 General-Purpose Input/Output (GPIO) pins with configurable pull-up/pull-down resistors and open-drain mode.
- GPIO registers are located on the AHB for fast access. GPIO ports have DMA support.
- Up to eight GPIO pins can be selected from all GPIO pins as edge and level sensitive interrupt sources.
- Two GPIO group interrupt modules enable an interrupt based on a programmable pattern of input states of a group of GPIO pins.
- Four general-purpose timer/counters with capture and match capabilities.
- One motor control Pulse Width Modulator (PWM) for three-phase motor control.
- One Quadrature Encoder Interface (QEI).
- Repetitive Interrupt timer (RI timer).
- Windowed watchdog timer (WWDT).
- Ultra-low power Real-Time Clock (RTC) on separate power domain with 256 bytes of battery powered backup registers.
- Alarm timer; can be battery powered.

- Analog peripherals
  - One 10-bit DAC with DMA support and a data conversion rate of 400 kSamples/s.
  - Two 10-bit ADCs with DMA support and a data conversion rate of 400 kSamples/s.
- Security
  - AES decryption programmable through an on-chip API.
  - Two 128-bit secure OTP memories for AES key storage and customer use.
  - Random number generator (RNG) accessible through AES API.
  - Unique ID for each device.
- Clock generation unit
  - Crystal oscillator with an operating range of 1 MHz to 25 MHz.
  - 12 MHz Internal RC (IRC) oscillator trimmed to 1 % accuracy over temperature and voltage.
  - Ultra-low power Real-Time Clock (RTC) crystal oscillator.
  - Three PLLs allow CPU operation up to the maximum CPU rate without the need for a high-frequency crystal. The second PLL is dedicated to the High-speed USB, the third PLL can be used as audio PLL.
  - Clock output.
- Power
  - Single 3.3 V (2.2 V to 3.6 V) power supply with on-chip DC-to-DC converter for the core supply and the RTC power domain.
  - RTC power domain can be powered separately by a 3 V battery supply.
  - Four reduced power modes: Sleep, Deep-sleep, Power-down, and Deep power-down.

- Processor wake-up from Sleep mode via wake-up interrupts from various peripherals.
- Wake-up from Deep-sleep, Power-down, and Deep power-down modes via external interrupts and interrupts generated by battery powered blocks in the RTC power domain.
- Brownout detect with four separate thresholds for interrupt and forced reset.
- Power-On Reset (POR).
- Available as 256-pin, 180-pin, and 100-pin LBGA package and as 208-pin, 144-pin. and 100-pin LQFP packages.

1.3 Ordering information (flashless parts LPC4350/30/20/10)

Table 1. Ordering information

Type number	Package		
	Name	Description	Version
LPC4350FET256	LBGA256	Plastic low profile ball grid array package; 256 balls; body 17 × 17 × 1 mm	SOT740-2
LPC4350FET180	TFBGA180	Thin fine-pitch ball grid array package; 180 balls	SOT570-3
LPC4350FBD208	LQFP208	Plastic low profile quad flat package; 208 leads; body 28 × 28 × 1.4 mm	SOT459-1
LPC4330FET256	LBGA256	Plastic low profile ball grid array package; 256 balls; body 17 × 17 × 1 mm	SOT740-2
LPC4330FET180	TFBGA180	Thin fine-pitch ball grid array package; 180 balls	SOT570-3
LPC4330FET100	TFBGA100	Plastic thin fine-pitch ball grid array package; 100 balls; body 9 × 9 × 0.7 mm	SOT926-1
LPC4330FBD144	LQFP144	Plastic low profile quad flat package; 144 leads; body 20 × 20 × 1.4 mm	SOT486-1
LPC4320FET100	TFBGA100	Plastic thin fine-pitch ball grid array package; 100 balls; body 9 × 9 × 0.7 mm	SOT926-1
LPC4320FBD144	LQFP144	Plastic low profile quad flat package; 144 leads; body 20 × 20 × 1.4 mm	SOT486-1
LPC4320FBD100	LQFP100	Plastic low profile quad flat package; 100 leads; body 14 × 14 × 1.4 mm	SOT407-1
LPC4310FET100	TFBGA100	Plastic thin fine-pitch ball grid array package; 100 balls; body 9 × 9 × 0.7 mm	SOT926-1
LPC4310FBD144	LQFP144	Plastic low profile quad flat package; 144 leads; body 20 × 20 × 1.4 mm	SOT486-1

Table 2. Ordering options

Type number	Total SRAM	LCD	Ethernet	USB0 (Host, Device, OTG)	USB1 (Host, Device)/ ULPI interface	ADC channels	PWM	QEI	GPIO	Package
LPC4350FET256	264 kB	yes	yes	yes	yes/yes	8	yes	yes	164	LBGA256
LPC4350FET180	264 kB	yes	yes	yes	yes/yes	8	yes	yes	118	TFBGA180
LPC4350FBD208	264 kB	yes	yes	yes	yes/yes	8	yes	yes	142	LQFP208
LPC4330FET256	264 kB	no	yes	yes	yes/yes	8	yes	yes	164	LBGA256
LPC4330FET180	264 kB	no	yes	yes	yes/yes	8	yes	yes	118	TFBGA180
LPC4330FET100	264 kB	no	yes	yes	yes/no	4	no	no	49	TFBGA100
LPC4330FBD144	264 kB	no	yes	yes	yes/no	8	yes	no	83	LQFP144
LPC4320FET100	200 kB	no	no	yes	no	4	no	no	49	TFBGA100
LPC4320FBD144	200 kB	no	no	yes	no	8	yes	no	83	LQFP144

Table 2. Ordering options

Type number	Total SRAM	LCD	Ethernet	USB0 (Host, Device, OTG)	USB1 (Host, Device)/ ULPI interface	ADC channels	PWM	QEI	GPIO	Package
LPC4320FBD100	200 kB	no	no	yes	no	5	no	no	49	LQFP100
LPC4310FET100	168 kB	no	no	no	no	4	no	no	49	TFBGA100
LPC4310FBD144	168 kB	no	no	no	no	8	yes	no	83	LQFP144

1.4 Block diagram

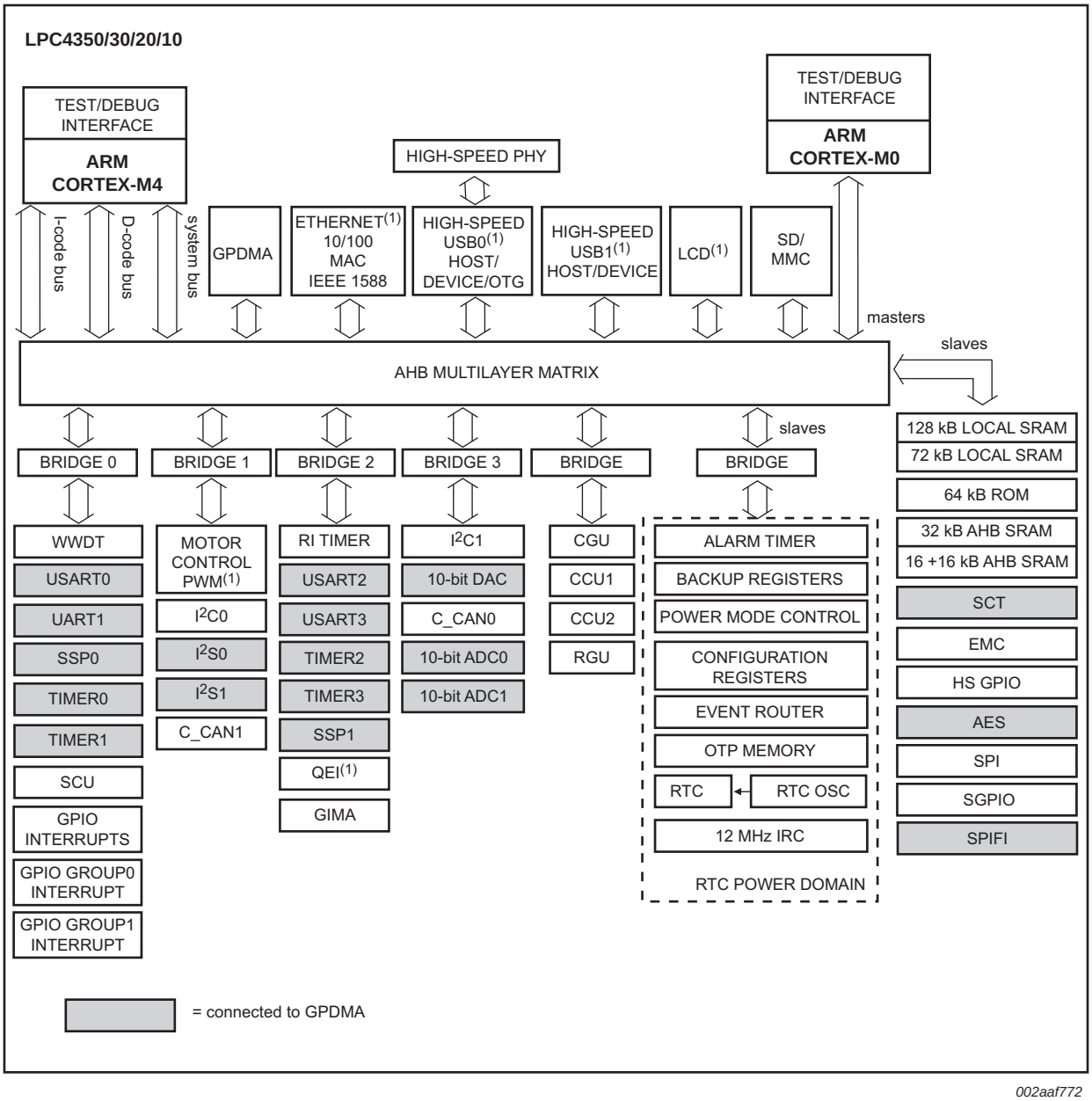


Fig 1. LPC43xx Block diagram

2.1 How to read this chapter

The ARM Cortex-M0 co-processor is available on all LPC43xx parts.

2.2 Basic configuration

The ARM Cortex-M0 co-processor is configured as follows:

- See Table 3 for clocking and power control.
- The ARM Cortex-M0 is reset by the M0APP_RST (reset # 56) or by a general Reset.
- After power-up, the ARM Cortex-M0 reset must be released by clearing the corresponding RESET_CTRL1 bit (see Table 106).
- The ARM Cortex-M0 interrupt is connected to interrupt slot # 1 in the ARM Cortex-M4 NVIC. See Table 22 for peripheral interrupts connected to the ARM Cortex-M0.
- To clear the ARM-Cortex-M0 interrupt, use the M0TXEVENT register (Table 46). See Section 2.4.2.

Table 3. ARM Cortex-M0 clocking and power control

	Base clock	Branch clock	Operating frequency
ARM Cortex-M0 clock	BASE_M4_CLK	CLK_M4_M0	up to 204 MHz

2.3 Introduction

In microcontrollers such as the LPC43xx, the ARM Cortex-M4 is used as application processor. A second processor, the ARM Cortex-M0, is available and can be used as co-processor to off-load the ARM Cortex-M4 and to perform serial I/O tasks. A communication protocol between the two processors is needed. This chapter describes the Inter Process Communication (IPC) protocol for the LPC43xx.

2.4 General description

On the LPC43xx, the ARM Cortex-M4 host CPU is used as the top-level system controller. The LPC43xx also includes a second CPU, an ARM Cortex-M0. The ARM Cortex-M0 CPU is controlled by the host CPU. The communication between both CPUs makes use of shared memory space and interrupts.

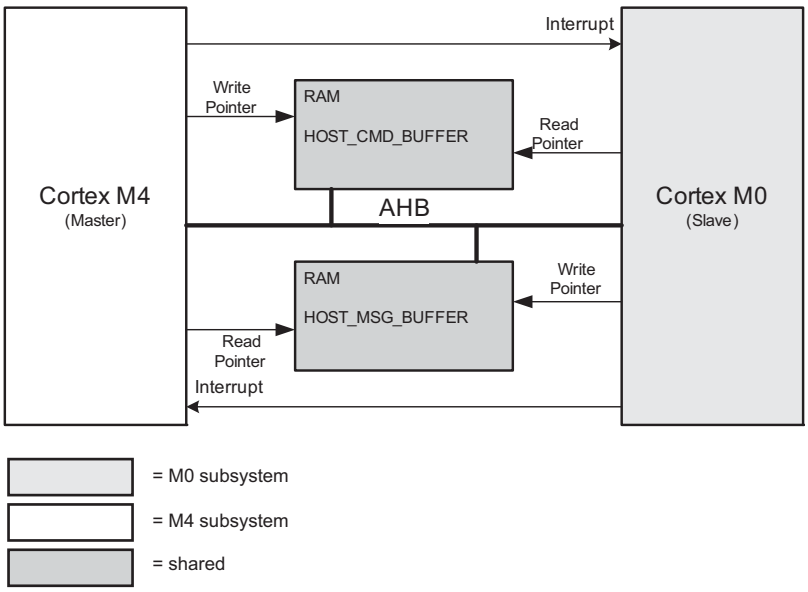


Fig 2. Dual-core block diagram

2.4.1 Hardware

Instead of using dedicated hardware, the IPC uses existing hardware components. The buffers in shared memory can use any of the available SRAM. The buffer pointers are maintained in software. The interrupts are captured in the processor's NVIC and cleared in the CREG block (see Table 44 and Table 46).

2.4.2 Interrupt handling

The ARM Cortex-M4 and ARM Cortex-M0 trigger interrupts to each other via CREG registers M4TXEVENT and M0TXEVENT (see Table 44 and Table 46). The M4-to-M0 and M0-to-M4 interrupts use the SendEvent instruction (SEV) to raise the signal TXEV. This signal is captured by CREG. It should be cleared by the interrupt handler of the receiving core.

2.5 IPC Protocol description

The IPC supports low-level interfaces, e.g. a register level interface, but also higher levels like an API.

The ARM Cortex-M4 host CPU is always master. It initiates commands to the ARM Cortex-M0 that mimic a hardware register level interface. The commands can be issued either synchronously (wait for the reply message) or asynchronously (not wait for the reply message) depending on the host application.

The ARM Cortex-M0 responds to commands given by the ARM Cortex-M4 by issuing messages.

Since the ARM Cortex-M4 and ARM Cortex-M0 cannot at the same time write to the same location, there is no need for a synchronization object (e.g. a semaphore) in this IPC.

The basic IPC features are:

- The ARM Cortex-M4 initializes the ARM Cortex-M0 system.
- The ARM Cortex-M4 communicates with the ARM Cortex-M0 system via a command queue.
- The message queues are located in the ARM Cortex-M4 address space because the ARM Cortex-M4 can be blocked from access to the ARM Cortex-M0 hardware subsystem.

2.5.1 IPC queues

The ARM Cortex-M4 has an output command queue and an input message queue. A queue is defined by four registers:

1. queue start address
2. queue end address
3. write pointer
4. read pointer

The ARM Cortex-M4 initializes these four registers. These registers reside in the same shared SRAM as the queues to ensure that data and registers changes are synchronous. Their location is static and known up front by the ARM Cortex-M0.

Messages are passed through queues using cyclic buffers. A queue is filled with commands or messages from start to end address. When a buffer pointer points beyond the end address it wraps around to the start address. When the read pointer is equal to the write pointer, the queue can be either empty or completely full. To avoid this ambiguity the queue shall never be filled completely. The minimum queue size is thus 3 words (the longest command/message +1 word). An equal write and read pointer will indicate an empty queue.

The command queue is filled by the ARM Cortex-M4 and emptied by the ARM Cortex-M0; the write pointer is advanced by the ARM Cortex-M4 every time it adds a new command to the queue. The read pointer is advanced by the ARM Cortex-M0 every time it removes a command from the queue.

The message queue is filled by the ARM Cortex-M0 and emptied by the ARM Cortex-M4; the write pointer is advanced by the ARM Cortex-M0 every time it adds a new message to the queue. The read pointer is advanced by the ARM Cortex-M4 every time it removes a message from the queue.

When a new command or message has been added to the queue and the write pointer had been updated, an interrupt is raised to the other processor. The commands are acknowledged by a return message (accept or fail).

The ARM Cortex-M4 and ARM Cortex-M0 only have one IPC write and one IPC read task. If multiple instances exist then a local arbiter shall ensure that all write and read operations are atomic; after data has been written (read) the write (read) pointer is updated before another write (read) operation can start.

It is the responsibility of the process writing to a queue making sure that the queue is not filled completely; before loading a new item the process should confirm that the write pointer will not be equal to, or overtake the read pointer and will leave at least one free space. On the other hand the receiving side shall promptly process and remove items from the queue.

No explicit error handling is performed. It is assumed that the ARM Cortex-M0 will always respond to a ARM Cortex-M4 command.

2.5.2 Protocol

The ARM Cortex-M0 is used a co-processor to off-load the ARM Cortex-M4 and to perform serial IO tasks. The ARM Cortex-M4 should be able to initialize tasks executed on the ARM Cortex-M0. The ARM Cortex-M0 should be able to signal to the ARM Cortex-M4 when these tasks have completed or failed. This is done by issuing commands from ARM Cortex-M4 to ARM Cortex-M0, where the ARM Cortex-M0 responds with messages. This command and message interface resembles a hardware register level interface with command and status registers.

The ARM Cortex-M4 issues 32-bit commands to the ARM Cortex-M0. Each command starts with a 16-bit ID that defines which task is referred to. The LSBit indicates the command type. A Write command is followed by a 32-bit operand. When a new command is available, the ARM Cortex-M4 signals this to the ARM Cortex-M0 by raising an interrupt.

The ARM Cortex-M0 return 32-bit messages to the ARM Cortex-M4. A messages starts with a 16-bit ID that indicates which tasks the message refers to. The LSByte indicates the message type. A Read response message is followed by the 32-bit read operand. When a new message is available, the ARM Cortex-M0 signals this to the ARM Cortex-M4 by raising an interrupt.

Small data transfers can be performed by the single 32-bit data read, CMD_RD_ID, and write, CMD_WR_ID, commands. These commands use a 3-Byte addressing scheme to support an argument space of 212 = 4096 32-bit words. Large data transfers can be more efficiently handled using pointers.

Also higher level interfaces using API calls will typically use indirect, pointer based, reads and writes.

When multiple tasks are running concurrently the ID is used to distinguish commands and messages belonging to a certain task. A global command parser should be used to the kick off commands to the tasks running on the ARM Cortex-M0.

The same holds true for the ARM Cortex-M4 side, a global message parser channels back messages to the task dispatchers running on the ARM Cortex-M4 side.

Table 4. Command list

Command	Bit mask	Description
CMD_RD_ID	0xTTTT.PPP0	read 32 bit WORD with argument ID=0xPPP from the task with ID = 0xTTTT
CMD_WR_ID	0xTTTT.PPP1, WORD	write 32 bit WORD with argument ID=0xPPP to the task with ID = 0xTTTT

Table 5. Message list

Message	Bit mask	Description
MSG_SRV_ID	0xTTTT.SS00	ARM Cortex-M0 request servicing for the task with ID = 0xTTTT. The service type is coded in bytes SS. The meaning of SS is proprietary per task. SS=0x00 means the task has finished.
MSG_RD_ID	0xTTTT.PPP1, VALUE	ARM Cortex-M0 responds with VALUE to a read of WORD with argument ID=0xPPP* from the task with ID = 0xTTTT.
MSG_RD_STS_ID	0xTTTT.PPPR	ARM Cortex-M0 response to a read of WORD with argument ID=0xPPP* from the task with ID = 0xTTTT fails. Cause of the failure is coded in R; R = 2...4 2 = invalid argument 3 = reserved 4 = reserved
MSG_WR_STS_ID	0xTTTT.PPPW	ARM Cortex-M0 response to a write with argument ID=0xPPP* from the task with ID = 0xTTTT. Response is coded in W; W = 5...7 5 = write was successful 6 = write failed 7 = reserved

Table 6. Command responses

Command	Possible responses	Description
CMD_RD_ID	MSG_RD_ID, VALUE	read acknowledged
	MSG_RD_STS_ID	read failed
CMD_WR, WORD	MSG_WR_STS_ID	write is acknowledged as a success or failure

2.5.3 Example

Assume a certain task with ID 0x1234 should be executed by the ARM Cortex-M0. For example, read data from a register level interface controlled by the ARM Cortex-M0. Text in bold indicates a register.

The registers can either be located in the ARM Cortex-M0 SRAM, for more deterministic access times, or in shared SRAM. If the ARM Cortex-M0 SRAM is used, then the register data needs to be copied at initialization time. This copying takes time. The ARM Cortex-M4 can poll a status register to determine when the transfer has finished.

The ARM Cortex-M4 initializes the command and message queues by loading the start- and end addresses and write and read pointers.

Then the ARM Cortex-M4 loads the register values in a reserved area in common SRAM memory. An alternative approach is that the ARM Cortex-M4 writes register per register, however this requires more communication overhead than loading all data in one go. Once all data has been set up, the ARM Cortex-M0 task can be started.

Table 7. IPC example

	Command	Message	Byte values	Description
1	CMD_WR		0x1234<rst>1, pointer	Command to initialise the task, the pointer informs the ARM Cortex-M0 the location of the register values.
The ARM Cortex-M0 processes the registers				
2		MSG_WR_STS	0x1234<rst>2	ARM Cortex-M0 signals write data has been processed
3		MSR_SRV	0x1234<serve>00	ARM Cortex-M0 requests service, e.g. because data has been captured and is available
4	CMD_RD		0x1234<status>1	ARM Cortex-M4 read status
5		MSG_RD,VALUE	0x1234<status>1, VALUE	ARM Cortex-M0 responds with status
Depending on the status the ARM Cortex-M4 may decide to read more data				
6	CMD_RD		0x1234<result>,1	ARM Cortex-M4 reads result
7		MSG_RD,VALUE	0x1234<result>1, pointer	ARM Cortex-M0 responds with pointer to results.
:	:	:	:	:
n	CMD_WR		0x1234<stop>1, value	Command to stop the task
n+1		MSG_WR_STS	0x1234<stop>2	ARM Cortex-M0 signals stop has been processed

3.1 How to read this chapter

The available peripherals and their memories vary for different parts.

- Ethernet: available only on LPC435x/3x.
- USB0: available only on LPC435x/3x/2x.
- USB1: available only on LPC435x/3x.
- SRAM: see Table 8.

The registers and memory regions corresponding to unavailable peripheral and memory blocks are reserved.

3.2 Basic configuration

In the CREG block (see Table 42), select the interface to access the 16 kB block of RAM located at address 0x2000 C000. This RAM memory block can be accessed either by the Embedded Trace Buffer (ETB) (this is the default) or be used as normal SRAM on the AHB bus.

Remark: When the ETB is used , the 16 kB memory space at 0x2000 C000 must not be used by any other process.

3.3 Memory configuration

3.3.1 On-chip static RAM

The LPC43xx support up to 264 kB SRAM with separate bus master access for higher throughput and individual power control for low power operation.

Table 8. LPC4350/30/20/10 SRAM configuration

Part	Local SRAM 0x1000 0000	Local SRAM 0x1008 0000	AHB SRAM 0x2000 0000	AHB SRAM 0x2000 8000	AHB SRAM/ ETB SRAM[1] 0x2000 C000	
LPC4350	128 kB	72 kB	32 kB	16 kB	16 kB	Figure 3
LPC4330	128 kB	72 kB	32 kB	16 kB	16 kB	Figure 3
LPC4320	96 kB	40 kB	32 kB	16 kB	16 kB	Figure 3
LPC4310	96 kB	40 kB	16 kB	-	16 kB	Figure 3

[1] To configure SRAM memory use for AHB or ETB, see Table 42.

3.3.2 Bit banding

Bit-banding offers efficient bit accesses. Bits in the bit-band region (0x2000 0000 to 0x2010 0000 and 0x4000 0000 to 0x40100000) can be accessed in the so-called alias region at 0x22000 0000 and 0x42000 0000. Reads return the respective bit from the bit-band region. Writes perform an atomic read-modify-write on the respective bit of the bit-band region. For details, see the *ARM Cortex-M4 technical reference manual*.

Remark: Bit banding can not be used with the MAC_RWAKE_FRFLT register (see Section 26.6.10).

Remark: Although the EEPROM is mapped in a bit-banding capable region, attempts to write access the EEPROM in the bit-banding aliased memory space will not result in a bit write

3.3.3 Memory retention in the Power-down modes

In Deep-sleep mode, all SRAM content is retained. At wake-up the system can restart immediately.

In Power-down mode, only the top 8 kB of the SRAM block starting at 0x1008 0000 is retained - that is 8 kB of SRAM located at 0x1009 0000. All other SRAM content is lost. Common practice is to store the stack and other variables that need to be retained in this 8 kB memory space as well as code to restart the rest of the system.

In Deep power-down mode, no SRAM content is retained. Variables that need to be retained in deep power down can be stored in the 256-byte register file located in the RTC domain at 0x4004 1000.

3.3.4 Memory Protection Unit (MPU)

The MPU is a integral part of the ARM Cortex-M4 for memory protection and supported by all LPC43xx parts. The processor supports the standard ARMv7 Protected Memory System Architecture model. The MPU provides full support for:

- protection regions
- overlapping protection regions, with ascending region priority (7 = highest priority, 0 = lowest priority)
- access permissions
- exporting memory attributes to the system

MPU mismatches and permission violations invoke the programmable-priority MemManage fault handler. See the *ARMv7-M Architecture Reference Manual* for more information.

The access permission bits, TEX, C, B, AP, and XN, of the Region Access Control Register control access to the corresponding memory region. If an access is made to an area of memory without the required permissions, a permission fault is raised. For more information, see the *ARMv7-M Architecture Reference Manual*.

The MPU is used to enforce privilege rules, to separate processes, and to enforce access rules. For details on how to use the MPU and for the register description refer to the A*RM Cortex-M4 Technical Reference Manual*.

3.4 Memory map

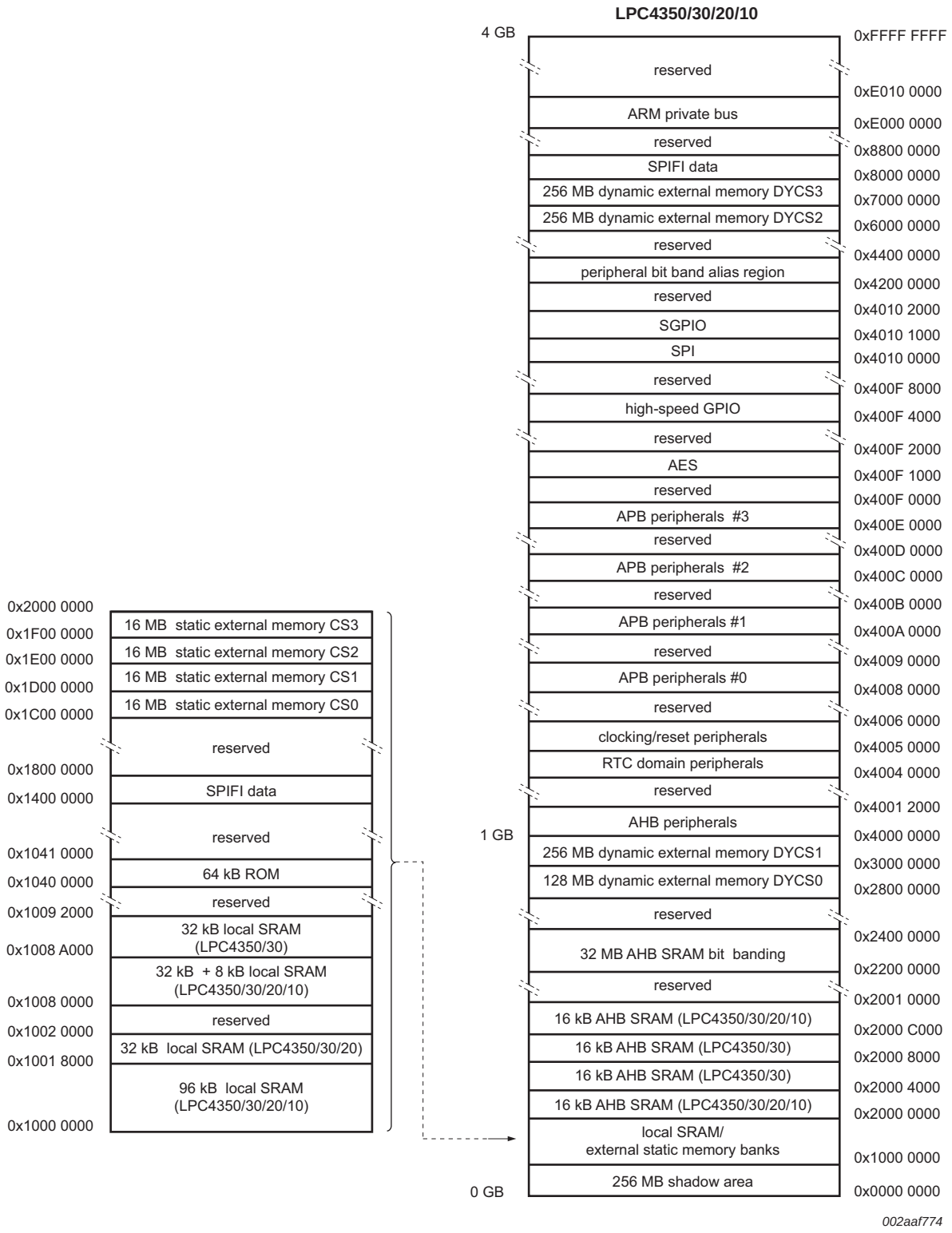


Fig 3. System memory map (see Figure 4 for detailed addresses of all peripherals)

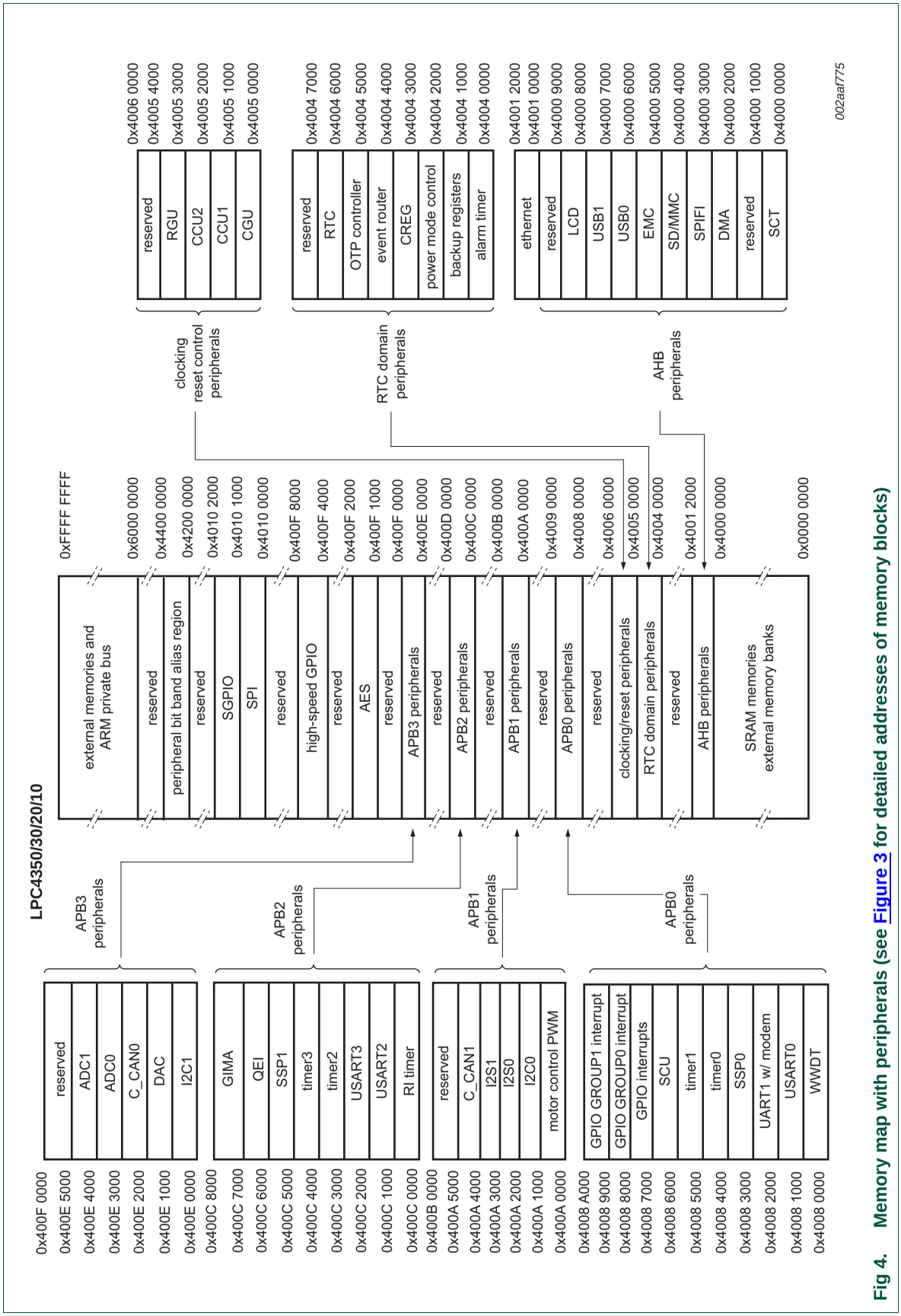


Fig 4. Memory map with peripherals (see Figure 3 for detailed addresses of memory blocks)

3.5 AHB Multilayer matrix configuration

The multilayer AHB matrix enables all bus masters to access any embedded memory as well as external SPI flash memory connected to the SPIFI interface. When two or more bus masters try to access the same slave, a round robin arbitration scheme is used; each master takes turns accessing the slave in circular order. The access length is determined by the burst access length of the master. For the CPU, the burst size is 1, for GP-DMA, the burst size can be up to 8. To optimize CPU performance, low-latency code should be stored in a memory that is not accessed by other bus masters, especially masters that use a long burst size.

To optimize the CPU performance, the ARM Cortex-M4 has three buses for Instruction (code) (I) access, Data (D) access, and System (S) access. The I- and D-bus access memory space is located below 0x2000 0000, the S-bus accesses the memory space staring from 0x2000 0000. When instructions and data are kept in separate memories, then code and data accesses can be done in parallel in one cycle. When code and data are kept in the same memory, then instructions that load or store data may take two cycles.

When the Embedded Trace Buffer is used (see ETBCFG register, Table 42), the 16 kB memory space at 0x2000 C000 must not be used by any other process.

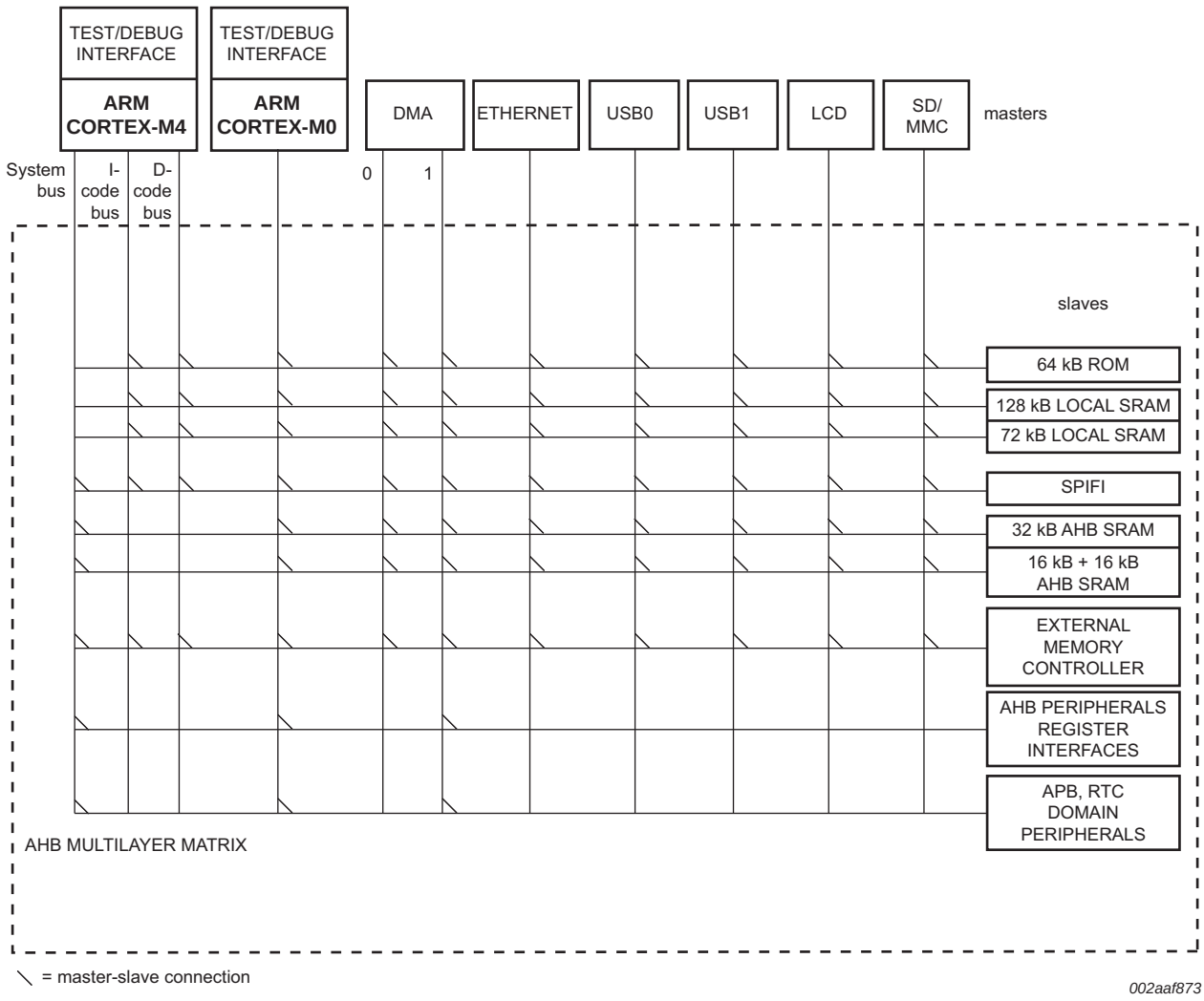


Fig 5. LPC43xx AHB multilayer matrix connections

4.1 How to read this chapter

This chapter applies to all LPC43xx parts.

4.2 Features

- The OTP memory stores the following information:
  - Part ID
  - Unique ID contains a unique ID for each microcontroller.
  - User programmable are the boot source, the USB vendor and product ID, and the AES keys.
  - Unused fields can be used to store other data.
- API support for programming the OTP

4.3 General description

The OTP contains pre-programmed device specific information part ID and Unique ID using two OTP banks. All other 384 OTP bits must be programmed by the user.

The virgin OTP state is all zeros. This implies that a zero value can be overwritten by a one value, but a one value cannot be changed.

Programming the OTP requires a higher voltage than reading. The read voltage is generated internally. The programming voltage is supplied via pin VPP. The OTP controller automatically selects the correct voltage. If the VPP pin is not connected, then the OTP cannot be programmed.

The AES keys in the OTP memory are not readable by software.

4.4 Register description

Table 9. OTP memory description (OTP base address 0x4004 5000)

OTP bank	Word	Access	Address offset	Size	Description	Reference
0	0	Pre-programmed; cannot be changed by the user.	0x000	32 bit	Part ID	-
0	1	Pre-programmed; cannot be changed by the user.	0x004	32 bit	Reserved	-
0	2	Pre-programmed; cannot be changed by the user.	0x008	32 bit	Unique ID	-
0	3	Pre-programmed; cannot be changed by the user.	0x00C	32 bit	Reserved	-

Table 9. OTP memory description (OTP base address 0x4004 5000)

OTP bank	Word	Access	Address offset	Size	Description	Reference
1	0	User programmable; initial state = 0	0x010	32 bit	General purpose OTP memory 0, word 0, or AES key 0, word 0	-
1	1	User programmable; initial state = 0	0x014	32 bit	General purpose OTP memory 0, word 1, or AES0 key 0, word 1	-
1	2	User programmable; initial state = 0	0x018	32 bit	General purpose OTP memory 0, word 2, or AES0 key 0, word 2	-
1	3	User programmable; initial state = 0	0x01C	32 bit	General purpose OTP memory 0, word 3, or AES0 key 0, word 3	-
2	0	User programmable; initial state = 0	0x020	32 bit	General purpose OTP memory 1, word 0, or AES key 1, word 0	-
2	1	User programmable; initial state = 0	0x024	32 bit	General purpose OTP memory 1, word 1, or AES key 1, word 1	-
2	2	User programmable; initial state = 0	0x028	32 bit	General purpose OTP memory 1, word 2, or AES key 1, word 2	-
2	3	User programmable; initial state = 0	0x02C	32 bit	General purpose OTP memory 1, word 3, or AES key 1, word 3	-
3	0	User programmable; initial state = 0	0x030	32 bit	Customer control data	Table 10
3	1	User programmable; initial state = 0	0x034	32 bit	General purpose OTP memory 2, word 0, or USB ID	Table 11
3	2	User programmable; initial state = 0	0x038	32 bit	General purpose OTP memory 2, word 1	Table 12
3	3	User programmable; initial state = 0	0x03C	32 bit	General purpose OTP memory 2, word 2	-

Table 10. OTP memory bank 3, word 0 - Customer control data (address offset 0x030)

Bit	Symbol	Value	Description
22:0	-	-	Reserved
23	USB_ID_ENABLE		Setting this bit allows to enable OTP defined USB vendor and product IDs. When enabled, the USB driver uses the USB_VENDOR_ID and USB_PRODUCT_ID values. If disabled, the NXP vendor ID (0x1FC9) and product ID (0x000C) is used.
		0	Disabled
		1	Enabled
24	-	-	Reserved

Table 10. OTP memory bank 3, word 0 - Customer control data (address offset 0x030)

Bit	Symbol	Value	Description
28:25	BOOT_SRC		Boot source selection in OTP. For details, see Table 15.
		0000	External pins
		0001	UART0
		0010	Reserved
		0011	EMC 8-bit
		0100	EMC 16-bit
		0101	EMC 32-bit
		0110	USB0
		0111	USB1
		1000	SPI (via SSP)
		1001	UART3
29	-		Reserved. Do not write to this bit.
30	-		Reserved. Do not write to this bit.
31	JTAG_DISABLE		If this bit set, JTAG cannot be enabeld by software and remains disabled.

Table 11. OTP memory bank 3, word 1 - General purpose OTG memory 2, word 0, or USB ID (address offset 0x034)

Bit	Symbol	Description
15:0	USB_VENDOR_ID	If USB_ID_ENABLE bit not set, it is used as general purpose OTG memory 2, word 0, GP2_0.
31:16	USB_PRODUCT_ID	

Table 12. OTP memory bank 3, word 2 - General purpose OTG memory 2, word 1, or Ethernet MAC (address 0x038)

Bit	Symbol	Description
31:0	ETH_MAC	If ETH_MAC_ENABLE bit not set, it is used as general purpose OTG memory 2, word 1, GP2_1.

4.5 OTP API

The OTP memory is controlled through a set of simple API calls located in the LPC43xx ROM.

The API calls to the ROM are performed by executing functions which are pointed to by pointer within the ROM driver table.

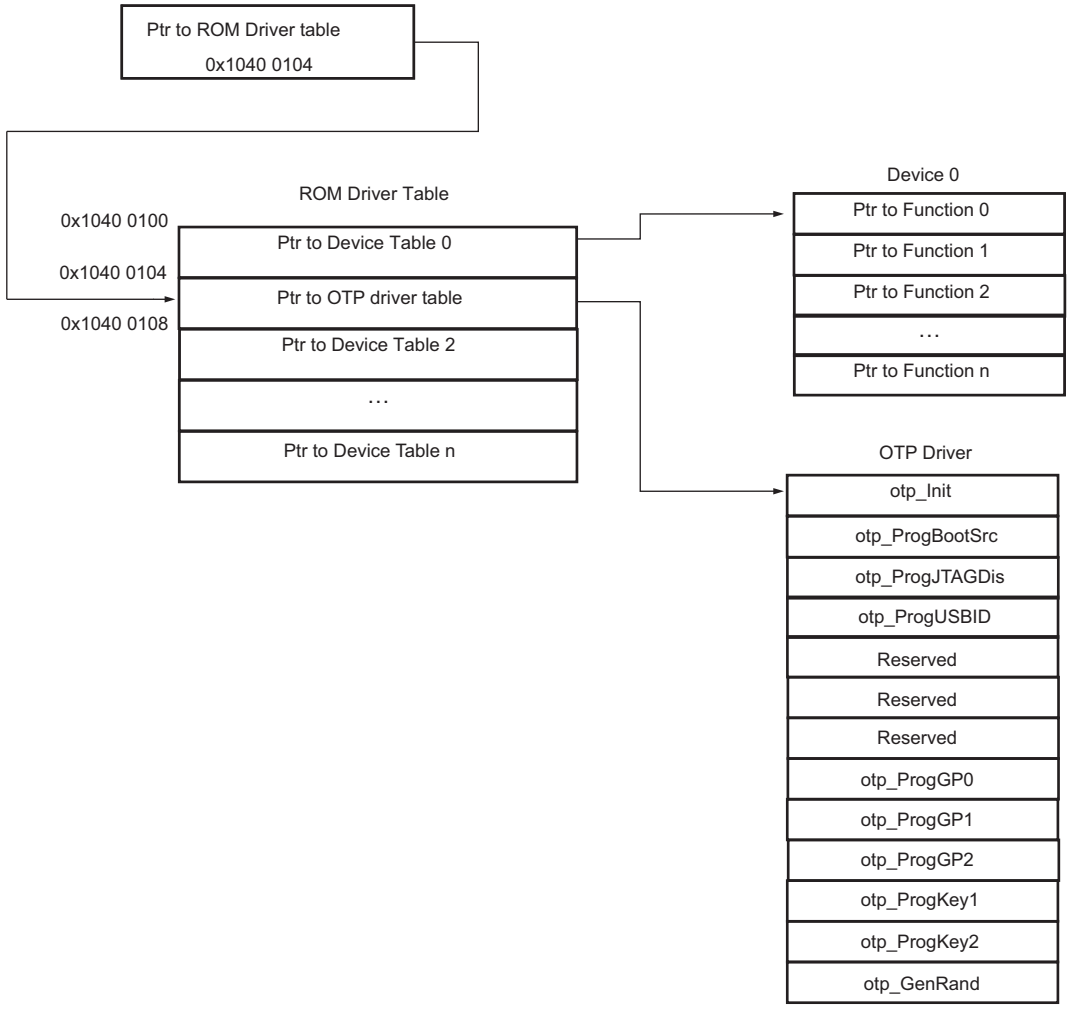


Fig 6. OTP driver pointer structure

Table 13. ROM driver pointers (main API entry point 0x1040 0100)

API	Address	Description
OTP	0x04	Pointer to the OTP driver base

4.5.1 OTP function allocation

Table 14. OTP function allocation

Function	Offset	Description
otp_Init	0x00	Initializes OTP controller. Parameter - void Return- unsigned: see the general error codes <tbd>.
otp_ProgBootSrc	0x04	Programs boot source. Parameter - unsigned <src> Return- unsigned: see the general error codes <tbd>.
otp_ProgJTAGDis	0x08	Set JTAG disable. This command disables JTAG only when the device is AES capable. Parameter - void Return- unsigned: see the general error codes <tbd>.
otp_ProgUSBID	0x0C	Programs USB_ID. Parameter - unsigned <prod_id>, unsigned <vend_id> Return- unsigned: see the general error codes <tbd>.
-	0x10	Reserved
-	0x14	Reserved
-	0x18	Reserved
otp_ProgGP0	0x1C	Programs the general purpose OTP memory GP0. Use only if the device is not AES capable. Parameter - unsigned <data>, unsigned <mask> Return- unsigned: see the general error codes <tbd>.
otp_ProgGP1	0x20	Programs the general pupose OTP memory GP1. Use only if the device is not AES capable. Parameter - unsigned <data>, unsigned <mask> Return- unsigned: see the general error codes <tbd>.
otp_ProgGP2	0x24	Programs the general purpose OTP memory GP2. Use for customer specific data. Parameter - unsigned <data>, unsigned <mask> Return- unsigned: see the general error codes <tbd>.
otp_ProgKey1	0x28	Program AES key1. Parameter - unsigned char *key (16 bytes expected) Return- unsigned: see the general error codes <tbd>.
otp_ProgKey2	0x2C	Program AES key2 Parameter - unsigned char *key (16 bytes expected) Return- unsigned: see the general error codes <tbd>.
otp_GenRand	0x30	Generate new random number using the hardware Random Number Generator (RNG). Parameter - void Return- unsigned: see the general error codes <tbd>.

4.5.2 OTP API calls

<tbd>

5.1 How to read this chapter

This chapter applies to all LPC4350/30/20/10 parts.

5.2 Features

The boot ROM memory includes the following features:

- ROM memory size is 64 kB.
- Supports booting from UART interfaces and external static memory such as NOR flash, SPI flash, quad SPI flash.
- Includes APIs for OTP programming.
- Includes USB drivers.
- ISP mode for loading data to on-chip SRAM and execute code from on-chip SRAM.

AES capable parts also support:

- CMAC authentication on the boot image.
- Secure booting from an encrypted image.
- Supports development mode for booting from a plain text image. Development mode is terminated by programming the AES key.
- API for AES programming.

5.3 Functional description

The internal ROM memory is used to store the boot code. After a reset, the ARM processor will start its code execution from this memory.

The ARM core is configured to start executing code, upon reset, with the program counter being set to the value 0x0000 0000. The LPC43xx contains a shadow pointer that allows areas of memory to be mapped to address 0x0000 0000. The default value of the shadow pointer is 0x1040 0000, ensuring that the code contained in the boot ROM is executed at reset.

Several boot modes are available depending on the values of the OTP bits BOOT_SRC (see Section 4.4). If the OTP memory is not programmed or the BOOT_SRC bits are all zero, the boot mode is determined by the states of the boot pins P2_9, P2_8, P1_2, and P1_1.

Table 15. Boot mode when OTP BOOT_SRC bits are programmed

Boot mode	BOOT_SRC bit 3	BOOT_SRC bit 2	BOOT_SRC bit 1	BOOT_SRC bit 0	Description
Boot pins	0	0	0	0	Boot source is defined by the reset state of P1_1, P1_2, P2_9, and P2_8 pins. See Table 16.
USART0	0	0	0	1	Boot from device connected to USART0 using pins P2_0 and P2_1.
SPIFI	0	0	1	0	Boot from Quad SPI flash connected to the SPIFI interface using pins P3_3 to P3_8.
EMC 8-bit	0	0	1	1	Boot from external static memory (such as NOR flash) using CS0 and an 8-bit data bus.
EMC 16-bit	0	1	0	0	Boot from external static memory (such as NOR flash) using CS0 and a 16-bit data bus.
EMC 32-bit	0	1	0	1	Boot from external static memory (such as NOR flash) using CS0 and a 32-bit data bus.
USB0	0	1	1	0	Boot from USB0.
USB1	0	1	1	1	Boot from USB1.
SPI (SSP)	1	0	0	0	Boot from SPI flash connected to the SSP0 interface on P3_3 (function SSP0_SCK), P3_6 (function SSP0_MISO), P3_7 (function SSP0_MOSI), and P3_8 (function SSP0_SSEL)[1].
USART3	1	0	0	1	Boot from device connected to USART3 using pins P2_3 and P2_4.

[1] The boot loader programs the appropriate pin function at reset to boot using SSP0.

Table 16. Boot mode when OTP BOOT_SRC bits are zero

Boot mode	P2_9	P2_8	P1_2	P1_1	Description
USART0	LOW	LOW	LOW	LOW	Boot from device connected to USART0 using pins P2_0 and P2_1.
SPIFI	LOW	LOW	LOW	HIGH	Boot from Quad SPI flash connected to the SPIFI interface on P3_3 to P3_8[1].
EMC 8-bit	LOW	LOW	HIGH	LOW	Boot from external static memory (such as NOR flash) using CS0 and an 8-bit data bus.
EMC 16-bit	LOW	LOW	HIGH	HIGH	Boot from external static memory (such as NOR flash) using CS0 and a 16-bit data bus.
EMC 32-bit	LOW	HIGH	LOW	LOW	Boot from external static memory (such as NOR flash) using CS0 and a 32-bit data bus.
USB0	LOW	HIGH	LOW	HIGH	Boot from USB0.
USB1	LOW	HIGH	HIGH	LOW	Boot from USB1.
SPI (SSP)	LOW	HIGH	HIGH	HIGH	Boot from SPI flash connected to the SSP0 interface on P3_3 (function SSP0_SCK), P3_6 (function SSP0_MISO), P3_7 (function SSP0_MOSI), and P3_8 (function SSP0_SSEL)[1].
USART3	HIGH	LOW	LOW	LOW	Boot from device connected to USART3 using pins P2_3 and P2_4.

[1] The boot loader programs the appropriate pin function at reset to boot from SPIFI or SSP0.

5.3.1 AES capable devices

AES capable products will normally always boot from a secure (encrypted) image and use CMAC authentication. However a special development mode allows booting from a plain text image. This development mode is active when the AES key has not been programmed. In this case the AES key consists of all zeros. Once the key is programmed (to a non-zero value), the development mode is terminated.

5.3.2 Boot process

The top level boot process is illustrated in Figure 7. The boot starts after Reset is released. The IRC is selected as CPU clock and the Cortex-M4 starts by executing boot ROM. By default the JTAG access to the chip is disabled at reset. When the part is non-AES capable or it is AES capable but the AES key has not been programmed then JTAG access is enabled.

As shown in Figure 7, the boot ROM determines the boot mode based on the OTP BOOT_SRC value or reset state of the pins P1_1, P1_2, P2_8, and P2_9. The boot ROM copies the image to internal SRAM at location 0x1000 0000 and jumps to that location (it sets ARM's shadow pointer to 0x1000 0000) after image verification. Hence the images for LPC43xx should be compiled with entry point at 0x0000 0000. On AES capable LPC43xx with a programmed AES key the image and header are authenticated using the CMAC algorithm. If authentication fails the device is reset.

On AES capable LPC43xx in development mode and non-AES capable LPC43xx, the image and header are not authenticated. If the image is not preceded by a header then the image is not copied to SRAM but assumed to be executable as-is. In that case the shadow pointer is set to the first address location of the external boot memory. The header-less images for LPC43xx should be compiled with entry point at 0x0000 0000, the same as for an image with header.

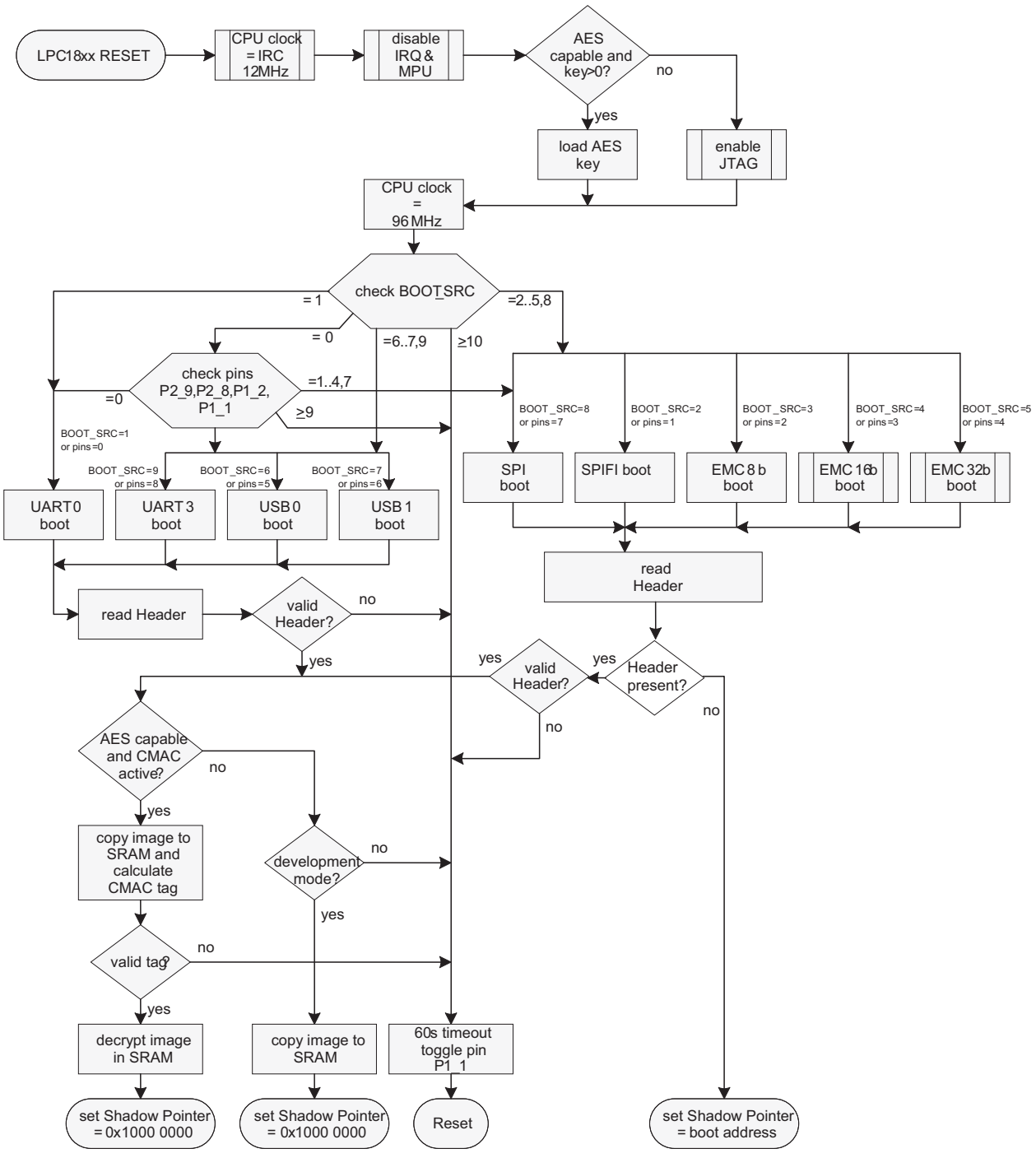


Fig 7. Boot process

5.3.3 Boot image format

AES capable products with a programmed AES key will always boot from a secure image and use CMAC authentication. A secure image should always include a header.

Non-AES capable products may boot from an image with header or execute directly from the boot source (when the boot source is memory mapped; SPIFI or EMC). When no valid header is found then the CPU will try to execute code from the first location of the memory mapped boot source. The user should take care that this location contains executable code, otherwise a hard fault exception will occur. This exception jumps to a while(1) loop.

The image must be preceded by a header that has the layout described in Table 17. Non-encrypted images may omit the header.

Table 17. Image header

Address	Name	Description	size [bits]
5:0	AES_ACTIVE[1]	AES encryption active 0x25 (100101): AES encryption active 0x1A (011010): AES encryption not active all other values: invalid image	6
7:6	HASH_ACTIVE[1]	Indicates whether a hash is used: 00: CMAC hash is used, value is HASH_VALUE 01: reserved 10: reserved 11: no hash is used	2
13:8	reserved	11...11 (binary)	6
15:14	AES_CONTROL	These 2 bits can be set to a value such that when AES encryption is active, that the AES_ACTIVE field, after AES encryption, is not equal to the value 0x1A (AES encryption not active)	2
31:16	HASH_SIZE[3]	Size of the part of the image over which the hash value is calculated in number of 512 Byte frames. Also size of image copied to internal SRAM at boot time. Hash size = 16[2] + HASH_SIZE x 512 Byte.	16
95:32	HASH_VALUE	CMAC hash value calculated over the first bytes of the image (starting right from the header) as indicated by HASH_SIZE. The value is truncated to the 64 MSB.	64
127:96	reserved	11...11 (binary)	32

[1] Can only be active if device is AES capable, else is considered an invalid image.

[2] 16 extra bytes are required for the header bytes.

[3] The image size should be set to no more than the size of the SRAM located at 0x1000 0000.

5.3.4 Boot image creation

5.3.4.1 CMAC

The CMAC algorithm is used to calculate a tag which is used for image authentication. The tag is stored in the header field HASH_VALUE.

The authentication process works as follows:

1. Use the CMAC algorithm to generate the 128-bit tag. Truncate the tag to 64 MSB and insert this truncated tag in the header.
2. At boot time the tag is recalculated. Authentication passes when the calculated tag is equal to the received tag in the image header.

To generate an l-bit CMAC tag T of message M using a 128-bit block cipher AES and secret key K, the CMAC tag generation process works as follows:

1. Generate sub key K_1:
   - Calculate a temporary value $K_0 = AES_K(0)$.
   - If $msb(K_0) = 0$ then $K_1 = (K_0 << 1)$ else $K_1 = (K_0 << 1) \oplus 0x87$
2. Divide message into 128-bit blocks $M = M_1 || ... || M_{n-1} || M_n^*$, where $M_1 ... M_{n-1}$ are complete blocks.
3. The last block, M_n^*, should be padded to be a complete block and then $M_n = K_1 \oplus M_n^*$.
4. Let $c_0 = 00...0$.
5. For i = 1, ..., n, calculate $c_i = AES_K(c_{i-1} \oplus M_i)$.
6. Output $T = msb_l(c_n)$.

The first message block is the header. Since the CMAC tag is stored in the header field HASH_VALUE, and this tag is not yet known until after CMAC calculation, a temporary header with a dummy tag value of 0x3456789A is used during CMAC calculation. This dummy value should be replaced by the calculated tag value in the final header field HASH_VALUE.

For LPC43xx the chosen CMAC parameters are: encryption key K = User Key (same as used for decryption) and tag length l = 64. Data is processed in little endian mode. This means that the first byte read from the image is integrated into the AES codeword as least significant byte. The 16th byte read from the image is the most significant byte of the first AES codeword.

CMAC is calculated over the header and encrypted image.

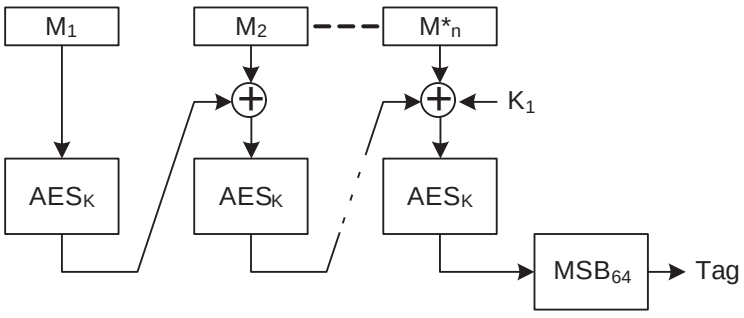


Fig 8. CMAC generation

5.3.4.2 UART boot mode

Figure 9 details the boot-flow steps of the UART boot mode. The execution of this mode occurs only if the boot mode is set accordingly (see boot modes Table 15 and Table 16).

As illustrated in Figure 9, configure the UART with the following settings:

- Baudrate = 115200, 57600, 38400, 19200, or 9600
- Data bits = 8
- Parity = None
- Stop bits = 1

Auto baud is active; boot waits until 0x3F is received and responds with “OK”. This should be followed by the header and image. The boot ROM doesn't implement any flow control or any handshake mechanisms during file transfer.

After the boot image is downloaded, it is checked (based on header information) to be a valid or invalid image and OK respectively FAILED is sent to a host followed by CR and LF. Finally, the full boot image is copied, processed to address 0x10000000 and executed from there.

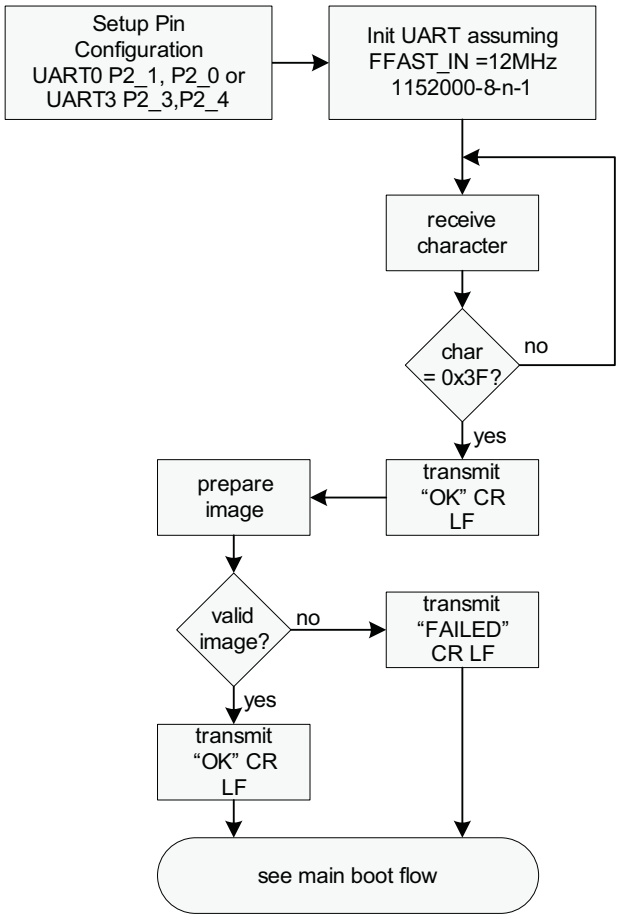


Fig 9. UART boot process

5.3.4.3 EMC boot modes

The EMC boot process follows the main flow shown in Figure 10. The CPU clock is set to 96 MHz, and a non-AES capable part will boot directly from EMC when the image does not contain a header. EMC uses 0xE wait states providing approximately 156 ns delay before capturing data.

Note that the number of address bits selected in pin configuration is initially EMC_A[13:0]. All higher address bit pins are configured as pull down but not actively driven. After reading the header, the address bits are extended to be in line with the image size as defined by HASH_SIZE, e.g. if HASH_SIZE is 100 kB then pins EMC_A[16:14] are configured since 2^{17} > 100 kB. When booting without header, then the image should configure extra address pins if more are needed beyond the initially configured EMC_A[13:0]. This configuration should happen in the initial 16 kB area of the image.

If no header is present it is assumed that the image is located at address 0x1C00 0000 and is executed from there.

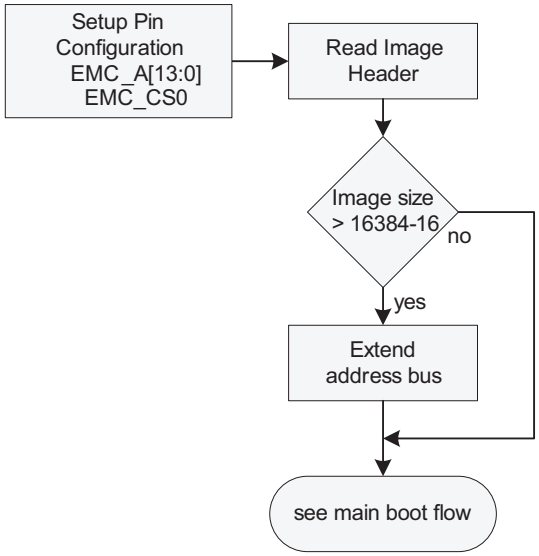


Fig 10. EMC boot process

5.3.4.4 SPI boot mode

The boot uses SSP0 in SPI mode. The SPI clock is 18 MHz.

Figure 11 details the boot-flow steps of the SPI flash boot mode. The execution of this mode occurs only if the boot mode is set accordingly (see boot modes Table 15 and Table 16).

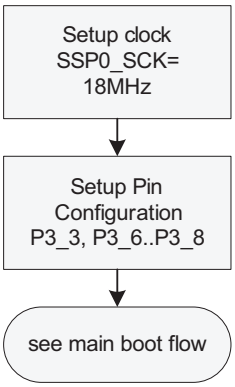


Fig 11. SPI boot process

5.3.4.5 SPIFI boot mode

Figure 12 details the boot-flow steps of the Quad SPI flash boot mode. The execution of

this mode occurs only if the boot mode is set accordingly (see boot modes in Table 15 andTable 16). The SPIFI clock is 32 MHz. If the detected device is unknown, the SPIFI clock is reduced to 18 MHz.

If no header is present, it is assumed that the image is located on address 0x80000000 and is executed from there.

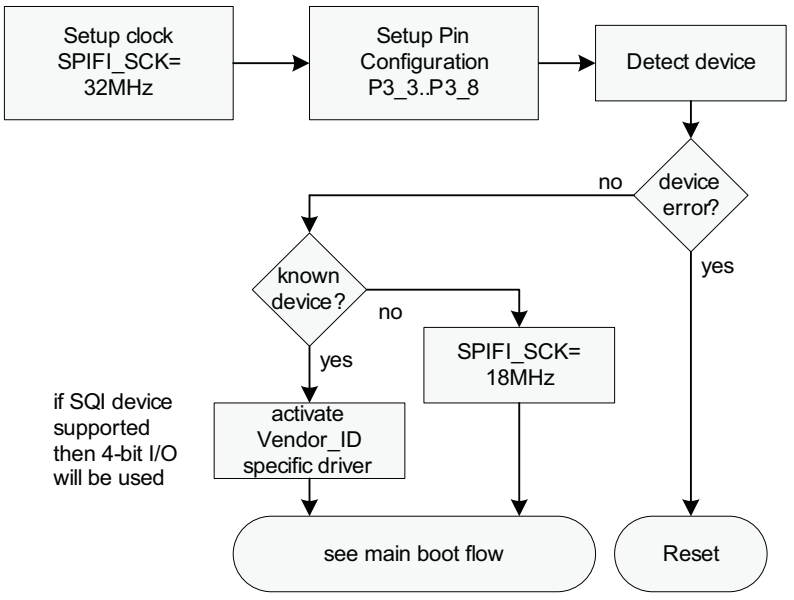


Fig 12. SPIFI boot process

5.3.4.6 USB boot mode

For booting from USB, two USB interfaces are available. USB0 supports high-speed and full-speed while USB1 supports only full-speed. The USB clock is respectively set to 480 MHz or 60 MHz. USB1 requires VBUS pin to be set correctly. Initially, the USB0 PHY

is disabled to save some power. After it is enabled, enumeration can start. The DFU class is used to download a boot image. After receiving a boot image from a host, the image is validated based on a set of rules mentioned earlier. If valid, the image is processed accordingly to address 0x10000000 and executed from there.

USB product and vendor ID are defined by the OTP (see Table 10 and Table 11).

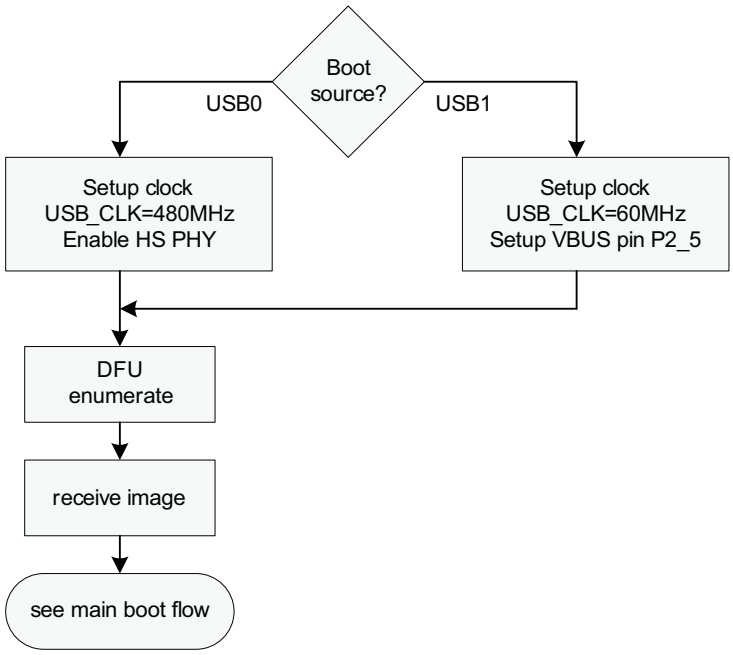


Fig 13. USB boot process

5.3.5 Boot process timing

The following parameters describe the timing of the boot process:

Table 18. Boot process timing parameters

Parameter	Description	Value
t_a	Check boot selection pins	< 1.25 μs <tbd>
t_b	Initialize device	250 μs <tbd>
t_c	Copy image to embedded SRAM	< 0.3 μs <tbd>
	If part is executing from external flash with no copy	
	If the image is encrypted or must be copied	< 1 μs to 10000 μs <tbd> depending on the size of the image and the speed of the boot memory

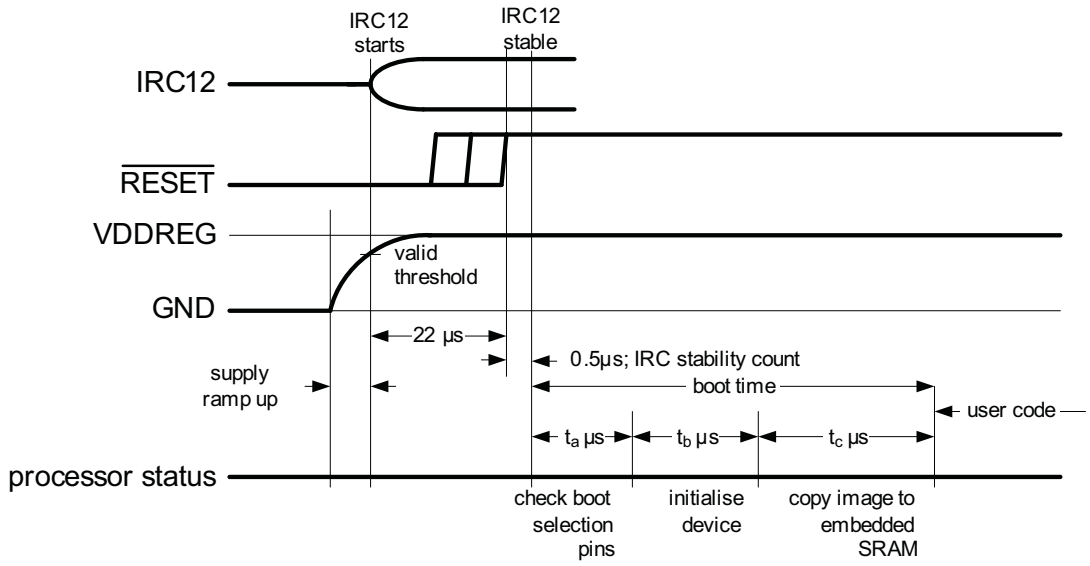


Fig 14. Boot process timing

5.3.6 ISP

In-System programming (ISP) is programming or re-programming the on-chip SRAM memory using the boot loader software and the USART0 serial port. ISP can be performed when the part resides in the end user board. ISP allows to load data into on-chip SRAM and execute code from on-chip SRAM. For details, see Chapter 45.

6.1 How to read this chapter

All LPC43xx parts support AES decryption. See also Chapter 5.

6.2 Features

- Decryption of external image data.
- Secure storage of decryption keys.
- Support for CMAC hash calculation to authenticate data.
- Support for one secret hardware key that cannot be read.
- AES engine performance of 1 byte/clock cycle.
- AES engine supports:
  - ECB decode mode with 128-bit key.
  - CBC decode mode with 128-bit key.
  - CMAC hash calculation (see Section 5.3.4.1).

Details of the AES decryption pertaining to the boot process are described in Chapter 5.

6.3 General description

The LPC43xx uses an external image to store instruction code and data. The LPC43xx offers hardware to protect the external image content and to accelerate processing for data decryption, data integrity, and proof of origin.

The hardware consists of:

- One-time programmable (OTP) non-volatile memories to store the AES key. Two instances (OTP1/2) are offered to store two keys using the OTP API (Table 14).
- An AES engine to perform the AES decryption. This engine supports an external GPDMA module to read and write data. The engine uses a 128-bit key and processes blocks of 128 bit. Using the AES API, the keys can be stored in a dedicated hardware interface that is not visible to software.

The AES engine can be loaded with four different keys:

1. Key 1- stored in the OTP
2. Key 2 - stored in the OTP
3. A randomly generated key
4. A software defined key

Remark: The randomly generated and software defined keys are not retained during Deep power-down and reset and must be reloaded.

Remark: To update the Random Number Generator (RNG) and load a new random number, first use the otp_GenRand() API call, and then the aes_LoadKeyRNG() call (see Section 6.5.5).

6.4 AES API

The AES is controlled through a set of simple API calls located in the LPC43xx ROM.

The API calls to the ROM are performed by executing functions which are pointed to by pointer within the ROM driver table.

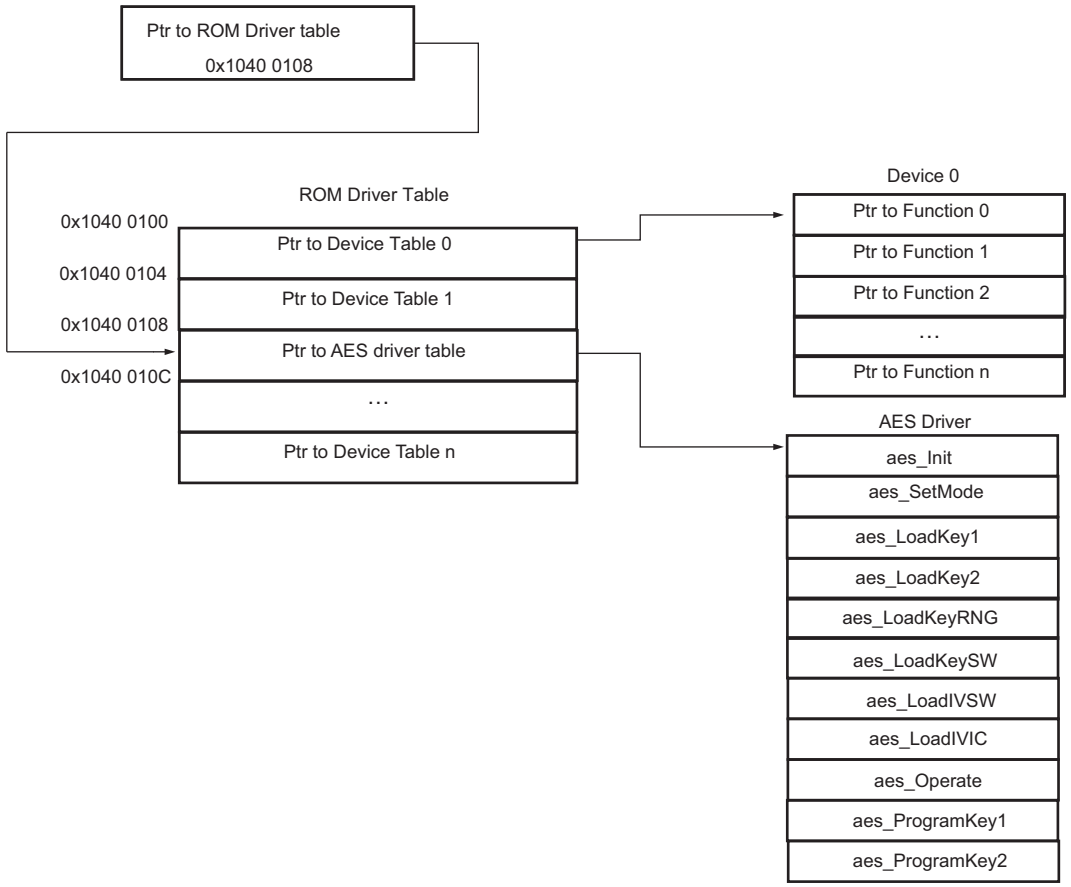


Fig 15. AES driver pointer structure

6.4.1 AES function allocation

The security API controls the AES block.

Table 19. Security API calls

Function	Offset relative to the API entry point	Description
aes_Init	0x00	Initialize AES engine Parameter - void Return - void
aes_SetMode	0x04	Defines AES engine operation mode Parameter: unsigned cmd with values: 0 - Reserved. Do not use. 1 - AES_API_CMD_DECODE_ECB 2 - Reserved. Do not use. 3 - AES_API_CMD_DECODE_CBC Return - unsigned: see general error codes.
aes_LoadKey1	0x08	Loads 128-bit AES user key 1 Parameter - void Return - void
aes_LoadKey2	0x0C	Loads 128-bit AES user key 2 Parameter - void Return - void
aes_LoadKeyRNG	0x10	Loads randomly generated key in AES engine. To update the RNG and load a new random number the API call otp_GenRand should be used before aes_LoadKeyRNG. Parameter - void Return - void
aes_LoadKeySW	0x14	Loads 128-bit AES software defined user key Parameter - unsigned char *key(16 bytes) Return - void
aes_LoadIV_SW	0x18	Loads 128-bit AES initialization vector Parameter - unsigned char *iv(16 bytes) Return - void
aes_LoadIV_IC	0x1C	Loads 128-bit AES IC specific initialization vector, which is used to decrypt a boot image. Parameter - void Return - void

Table 19. Security API calls

Function	Offset relative to the API entry point	Description
aes_Operate	0x1C	Performs the AES decryption after the AES mode has been set using aes_Set_Mode and the appropriate keys and init vectors have been loaded. Parameter1 - unsigned char *data_out Parameter2 - unsigned char *data_in Parameter3 - unsigned size (128-bit word - 16 byte) Return - unsigned: see general error codes.
aes_ProgramKey1	0x20	Programs 128-bit AES key in OTP. Parameter: unsigned char *key (16 byte) Return - unsigned: see general error codes.
aes_ProgramKey2	0x24	Programs 128-bit AES key in OTP. Parameter: unsigned char *key (16 byte) Return - unsigned: see general error codes.

6.5 Functional description

6.5.1 AES Decryption

The data is decrypted using the following steps (see Figure 16):

1. Decrypt the Header using AES with the user key (AES user key1 stored in OTP (see Table 9)) as AES key and iv=0. The Header provides the HASH_VALUE and the HASH_SIZE over which the CMAC is calculated.
2. In the Header, replace HASH_VALUE by the constant 0x3456789A.
3. Encrypt the Header using AES with the user Key as AES key and iv=0.
4. Calculate the CMAC tag as defined in Section 6.5.2 and confirm that the 64 MSB are equal to HASH_VALUE, if not then reset the device.
5. Decrypt the Cipher Text frames using CBC AES with the user Key as AES key and the iv as defined in Section 6.5.3. The number of frames to decrypt is given by the value HASH_SIZE. If more frames need to be decrypted then this needs to be done by the application.

It is possible to decrypt a frame of Cipher Text independent of other Cipher Text frames. This is useful when a random frame needs to be accessed.

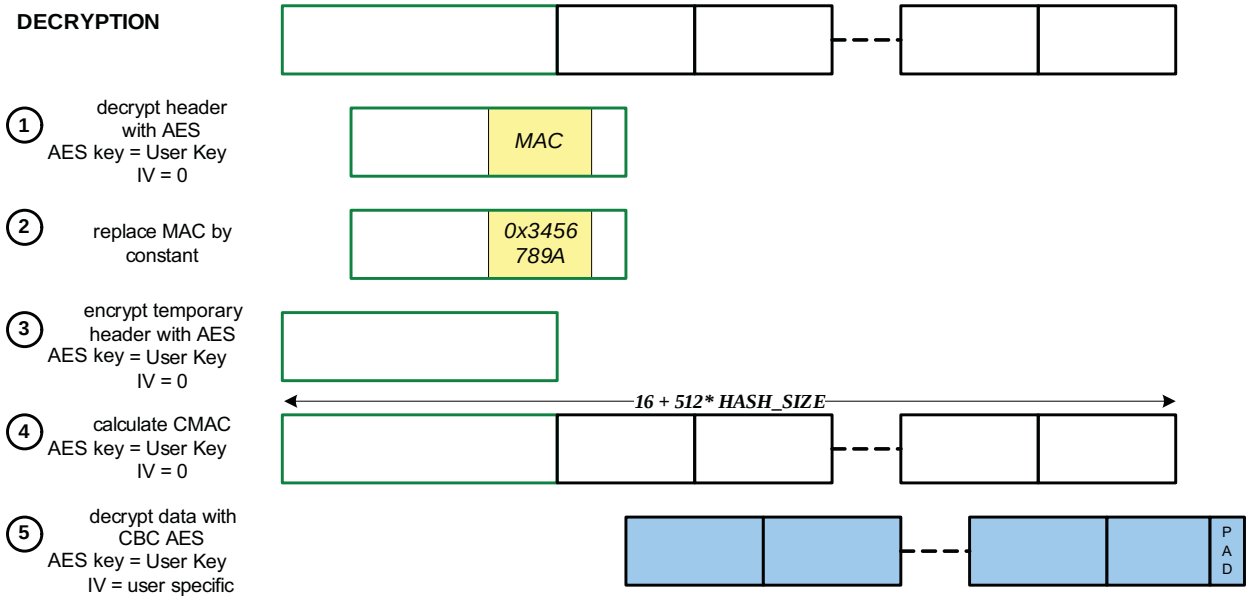


Fig 16. AES decryption flow

6.5.2 CMAC using AES hardware acceleration

CMAC is an authentication algorithm that uses the AES engine.

CMAC is calculated over the cipher text. This is better than calculating CMAC over the plain text because the cipher text will be more random (due to using CBC), even if the plain text is not random.

Generate sub-keys

To generate an l-bit CMAC tag T of message m using a b-bit block cipher E and secret key k, first generate two b-bit sub-keys k1 and k2 using the following algorithm (this is equivalent to multiplication by x and x2 in a finite field GF(2b)). Let << signify a standard left-shift operator:

1. Calculate a temporary value $k_0 = E_k(0)$.
2. If $msb(k_0) = 0$ then k1 = k0 << 1 else k1 = (k0 << 1) $\oplus$ C, where C is a certain constant that depends only on b. Specifically, C is the non-leading coefficients of the lexicographically first irreducible degree-b binary polynomial with the minimal number of ones. For b =128, C = 00...010000111.
3. If $msb(k_1) = 0$ then k2 = k1 << 1 else k2 = (k1 << 1) $\oplus$ C.

Remark: Images should consist of complete blocks; Mn* is 128-bit hence K2 is never used.

Generate the CMAC tag

To generate a CMAC tag, follow these steps:

1. Divide message into b-bit blocks $M = M_1 \| \ldots \| M_{n-1} \| M_n^*$ where $M_1, \ldots, M_{n-1}$ are complete blocks.
2. $M_n = K_1 \oplus M_n^*$
3. Set $c_0 = 00...0$ (binary).
4. For i = 1,…, n, calculate $c_i = E_k(c_{i-1} \oplus M_i)$.
5. Output $T = MSB_l(c_n)$.

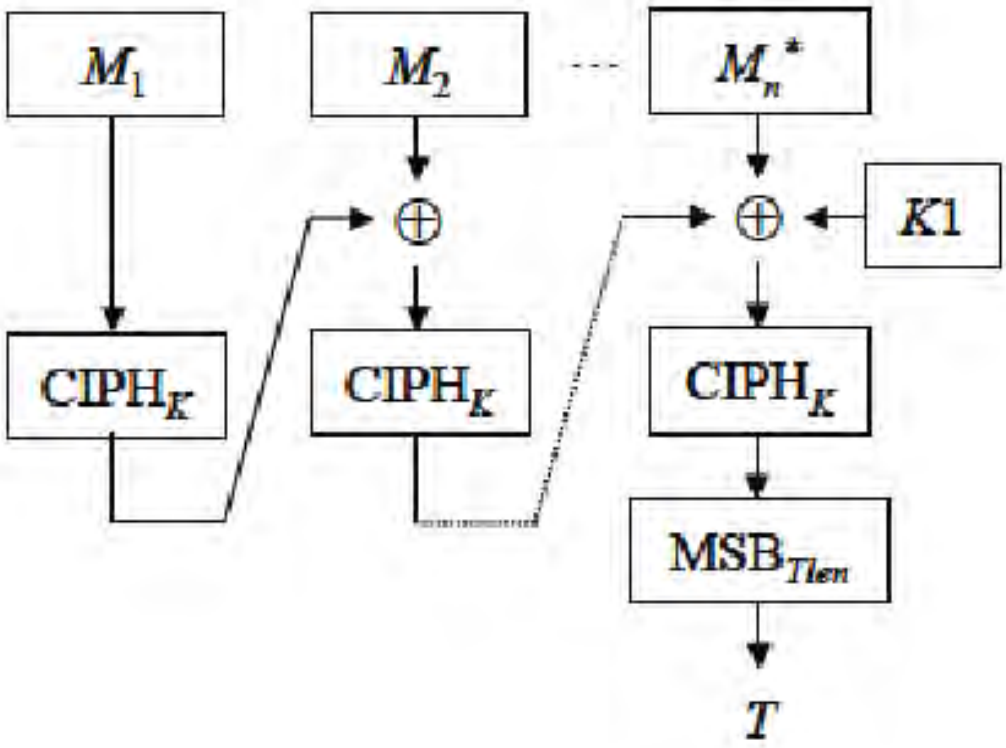


Fig 17. CMAC generation

Verify the CMAC tag

An encrypted image is authenticated by the boot code.:

6.5.3 Use of AES keys

The software key is a software defined AES key. Since this key is visible to software, it is less secure than the hardware defined keys in OTP. However, the OTP can only store two keys whereas multiple keys can be stored in software.

The 128-bit AES init vector iv is used to randomize the encryption when the same data is encrypted multiple times, The init vector does not have to be secret. and is also used to decrypt the data. For the CMAC calculation, an AES initialization vector of iv = 0 is used.

For the LPC43xx, a user specific iv is used:

$iv = AES^{-1}(User\ Key, 1)$

6.5.4 Endianess

The AES engine is capable of processing 128-bit (16-byte) blocks per operation. To load/store an AES block, the 32-bit infrastructure is fully used. For convenience, the API interface uses byte order rather than word order. The API passes/obtains a pointer to an array of bytes, and the AES low-level driver type-casts the pointer to an unsigned 32-bit array. shows 16-byte data AES encryption with a 16-byte key. For simplicity, data and key bytes are chosen in incrementing order starting from 00.

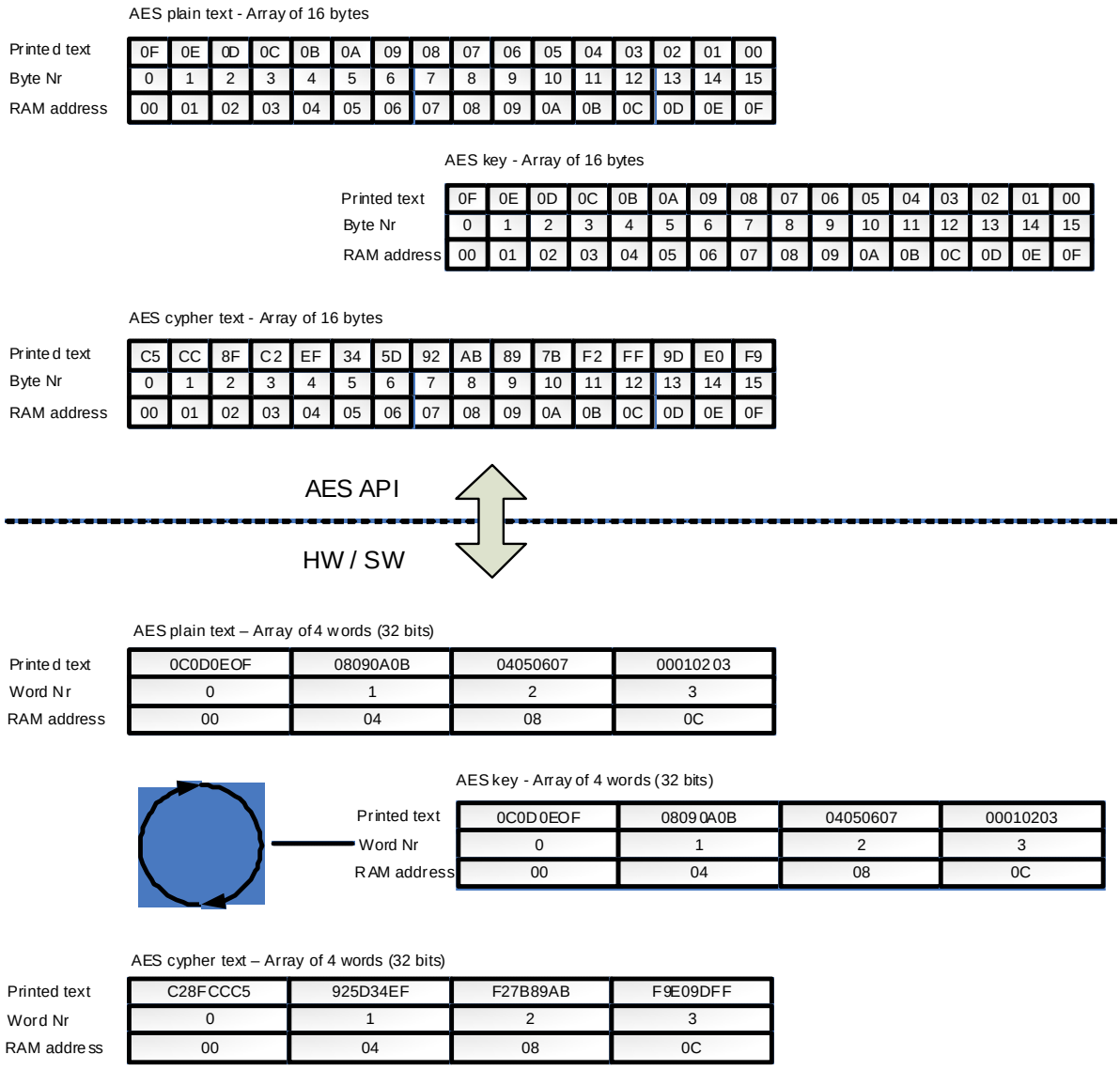


Fig 18. AES endianess

6.5.5 Storing AES keys in Deep power-down mode

In Deep power-down mode, all AES information is lost. After wake-up, the AES keys need to be reloaded. If the same RNG key is to be used as before going to Deep power-down, then this RNG key may be stored in the backup registers at 0x4004 1000. The following processing order should be kept:

1. Generate a random number by calling the otp_GenRand() API.
2. Store this number in the RTC REGFILE registers.
3. Load this number in the AES engine using aes_LoadKeySW.

After every wake-up, perform the following operations:

1. Load the stored random number from the backup register.
2. Load this number in the AES engine using aes_LoadKeySW.

7.1 How to read this chapter

The NVIC interrupt sources vary for different parts.

- Ethernet interrupt: available only on LPC4350/30.
- USB0 interrupt: available only on LPC4350/30/20.
- USB1 interrupt: available only on LPC4350/30.

7.2 Basic configuration

On the LPC43xx, each core is connected to its own NVIC, the ARM Cortex-M4 NVIC and the ARM Cortex-M0 NVIC.

7.3 Features

- Nested Vectored Interrupt Controllers are integral parts of the ARM Cortex-M4 and M0 processors.
- Tightly coupled interrupt controllers provides low interrupt latency.
- NVICs control system exceptions and peripheral interrupts.
- Software interrupt generation is supported.
- Cortex-M4 core:
  - Up to 52 interrupts
  - Relocatable vector table
  - Non-Maskable Interrupt
  - 8 programmable interrupt priority levels with hardware priority level masking
- Cortex-M0 core:
  - Up to 32 interrupts
  - 4 <tbd> programmable interrupt priority levels with hardware priority level masking

7.4 General description

The Nested Vectored Interrupt Controller (NVIC) is an integral part of the Cortex-M4 and M0. The tight coupling to the CPU allows for low interrupt latency and efficient processing of late arriving interrupts.

Refer to the Cortex-M4 User Guide for details of NVIC operation.

7.5 Pin description

Table 20. NVIC pin description

Function	Direction	Description
NMI	I	External Non-Maskable Interrupt (NMI) input

7.6 Interrupt sources

Table 21 lists the interrupt sources for each peripheral function. Each peripheral device may have one or more interrupt lines to the Vectored Interrupt Controller. Each line may represent more than one interrupt source, as noted.

Exception numbers relate to where entries are stored in the exception vector table. Interrupt numbers are used in some other contexts, such as software interrupts.

In addition to the signals listed in Table 21 and Table 22, the NVIC handles the Non-Maskable Interrupt (NMI). In order for NMI to operate from an external signal, the NMI function must be connected to the related device pin (P4_0 or PE_4). When connected, a logic one on the pin will cause the NMI to be processed. For details, refer to the Cortex-M4 or Cortex-M0 User Guide.

7.6.1 Interrupt sources for the Cortex-M4

Table 21. Connection of interrupt sources to the Cortex-M4 NVIC

Interrupt ID	Exception Number	Vector Offset	Function	Flag(s)
0	16	0x40	DAC	
1	17	0x44	M0CORE	Cortex-M0; Latched TXEV; for M4-M0 communication
2	18	0x48	DMA	
3	19	0x4C	-	Reserved
4	20	0x50	-	Reserved
5	21	0x54	ETHERNET	Ethernet interrupt
6	22	0x58	SDIO	SD/MMC interrupt
7	23	0x5C	LCD	
8	24	0x60	USB0	OTG interrupt
9	25	0x64	USB1	<tbd>
10	26	0x68	SCT	SCT combined interrupt
11	27	0x6C	RITIMER	
12	28	0x70	TIMER0	
13	29	0x74	TIMER1	
14	30	0x78	TIMER2	
15	31	0x7C	TIMER3	
16	32	0x80	MCPWM	Motor control PWM
17	33	0x84	ADC0	
18	34	0x88	I2C0	

Table 21. Connection of interrupt sources to the Cortex-M4 NVIC

Interrupt ID	Exception Number	Vector Offset	Function	Flag(s)
19	35	0x8C	I2C1	
20	36	0x90	SPI	
21	37	0x94	ADC1	
22	38	0x98	SSP0	
23	39	0x9C	SSP1	
24	40	0xA0	USART0	
25	41	0xA4	UART1	Combined UART interrupt with Modem interrupt
26	42	0xA8	USART2	
27	43	0xAC	USART3	Combined USART interrupt with IrDA interrupt
28	44	0xB0	I2S0	
29	45	0xB4	I2S1	
30	46	0xB8	SPIFI	
31	47	0xBC	SGPIO	
32	48	0xC0	PIN_INT0	GPIO pin interrupt 0
33	49	0xC4	PIN_INT1	GPIO pin interrupt 1
34	50	0xC8	PIN_INT2	GPIO pin interrupt 2
35	51	0xCC	PIN_INT3	GPIO pin interrupt 3
36	52	0xD0	PIN_INT4	GPIO pin interrupt 4
37	53	0xD4	PIN_INT5	GPIO pin interrupt 5
38	54	0xD8	PIN_INT6	GPIO pin interrupt 6
39	55	0xDC	PIN_INT7	GPIO pin interrupt 7
40	56	0xE0	GINT0	GPIO global interrupt 0
41	57	0xE4	GINT1	GPIO global interrupt 1
42	58	0xE8	EVENTROUTER	Event router interrupt
43	59	0xEC	C_CAN1	
44	60	0xF0	Reserved	
45	61	0xF4	Reserved	
46	62	0xF8	ATIMER	Alarm timer interrupt
47	63	0xFC	RTC	
48	64	0x100	Reserved	
49	65	0x104	WWDT	
50	66	0x108	Reserved	
51	67	0x10C	C_CAN0	
52	68	0x110	QEI	

7.6.2 Interrupt sources for the Cortex-M0

Table 22. Connection of interrupt sources to the Cortex-M0 NVIC

Interrupt ID	Exception Number	Vector Offset	Function	Flag(s)
0	16	0x40	M0_RTC	
1	17	0x44	M0_M4CORE	Interrupt from the M4 core
2	18	0x48	M0_DMA	
3	19	0x4C	-	Reserved
4	20	0x50	-	Reserved
5	21	0x54	M0_ETHERNET	Ethernet interrupt
6	22	0x58	M0_SDIO	
7	23	0x5C	M0_LCD	
8	24	0x60	M0_USB0	OTG interrupt
9	25	0x64	M0_USB1	
10	26	0x68	M0_SCT	SCT combined interrupt
11	27	0x6C	M0_RITIMER_OR_ WWDT	RI timer interrupt ORed with WWDT interrupt
12	28	0x70	M0_TIMER0	
13	29	0x74	M0_GINT1	GPIO global interrupt 1
14	30	0x78	PIN_INT4	GPIO pin interrupt 4<tbd>
15	31	0x7C	M0_TIMER3	
16	32	0x80	M0_MCPWM	Motor control PWM
17	33	0x84	M0_ADC0	
18	34	0x88	M0_I2C0_OR_I2C1	
19	35	0x8C	M0_SGPIO	
20	36	0x90	M0_SPI_OR_DAC	SPI interrupt ORed with DAC interrupt
21	37	0x94	M0_ADC1	
22	38	0x98	M0_SSP0_OR_SSP1	SSP0 interrupt ORed with SSP1 interrupt
23	39	0x9C	M0_EVENTROUTER	Event router
24	40	0xA0	M0_USART0	
25	41	0xA4	M0_UART1	Modem interrupt
26	42	0xA8	M0_USART_OR_ C_CAN1	USART2 interrupt ORed with C_CAN1 interrupt
27	43	0xAC	M0_USART3	
28	44	0xB0	M0_I2S0_OR_I2S1_QEI	I2S0 OR I2S1 OR QEI interrupt
29	45	0xB4	M0_C_CAN0	
30	46	0xB8	Reserved	
31	47	0xBC	Reserved	

8.1 How to read this chapter

Remark: The event router controls the wake-up process and various event inputs to the NVIC.

The event router sources vary for different parts.

- Ethernet: available only on LPC4350/30.
- USB0: available only on LPC4350/30/20.
- USB1: available only on LPC4350/30.

8.2 Basic configuration

- See Table 23 for clocking.
- The event router is connected to interrupt #42 in the NVIC (see Table 21).
- The CREG0 register configures the WAKEUP0/1 pins as inputs to the event router or outputs, which monitor the output of the event router (see Table 38).
- In order to detect an event connected to any of the peripheral interrupts, set the corresponding bit to HIGH in the HILO register (default setting of this register is LOW).
- For events #4 and #5, activate the 32 kHz oscillator in the CREG0 register (Table 38).

Table 23. Event router clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to event router	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz

8.3 General description

The event router is used to process wake-up events such as certain interrupts and external or internal inputs for wake-up from any of the Power-down modes (Sleep, Deep-sleep, Power-down, and Deep power-down modes).

In Deep-sleep, Power-down, or Deep power-down mode, only events on one of the four WAKEUP pins and the RTC and alarm timer events, if the 32 kHz oscillator is running, are active (see Table 24).

All events can wake up the part from Sleep mode. However, the RTC and alarm timer events require that the 32 kHz oscillator is running.

The event router has multiple event inputs from various peripherals. When the proper edge detection is set in the EDGE configuration register, the event router can wake up the part or can raise an interrupt in the NVIC.

Each event input to the event router can be configured to trigger an output signal on rising or falling edges or on HIGH or LOW levels. The event router combines all events to an output signal which is used as follows:

- Create an interrupt if the event router interrupt is enabled in the NVIC.
- Send a wake-up signal to the power management unit to wake up from Deep-sleep, Power-down, and Deep power-down modes.
- Send a wake-up signal to CCU1 and CCU2 for waking up from Sleep mode (see Section 12.5.3).

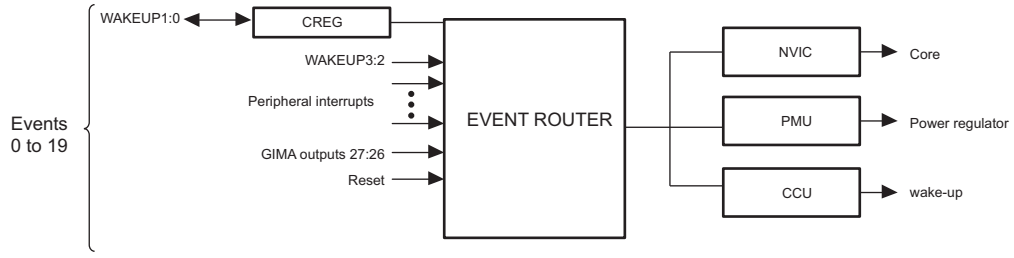


Fig 19. Event router block diagram

8.4 Event router inputs

Table 24. Event router inputs

Event #	Source	Notes
0	WAKEUP0 pin	WAKEUP0 pin. Always active. Use for wake-up from Deep power-down and all other Power-down modes.
1	WAKEUP1 pin	WAKEUP1 pin. Always active. Use for wake-up from Deep power-down and all other Power-down modes.
2	WAKEUP2 pin	WAKEUP2 pin. Always active. Use for wake-up from Deep power-down and all other Power-down modes.
3	WAKEUP3 pin	WAKEUP3 pin. Always active. Use for wake-up from Deep power-down and all other Power-down modes.
4	Alarm timer peripheral	Alarm timer interrupt. Active whenever the 32 kHz oscillator is running.
5	RTC peripheral	RTC interrupt. Active whenever the 32 kHz oscillator is running.
6	BOD trip level 1	BOD interrupt; wake-up from low power mode. Not active in Deep power-down mode. Use for wake-up from Sleep, Deep-sleep, and Power-down mode.
7	WWDT peripheral	WWDT interrupt. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
8	Ethernet peripheral	Wake-up packet indicator. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
9	USB0 peripheral	Wake-up request signal. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
10	USB1 peripheral	ahb_needclk signal. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.

Table 24. Event router inputs

Event #	Source	Notes
11	SD/MMC peripheral	SD/MMC interrupt. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
12	C_CAN0/1 peripherals	ORed C_CAN0 and C_CAN1 interrupt. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
13	GIMA output 25	Output 2 of the combined timer (ORed output of SCT output 2 and the match channel 2 of timer 0). See Table 139. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
14	GIMA output 26	Output 6 of the combined timer (ORed output of SCT output 6 and the match channel 2 of timer 1). See Table 139. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
15	QEI peripheral	QEI interrupt
16	GIMA output 27	Output 14 of the combined timer (ORed output of SCT output 14 and the match channel 2 of timer 3). See Table 139. Not active in Deep-sleep, Power-down, and Deep power-down mode. Use for wake-up from Sleep mode.
17	WAKEUP0 pin	Reserved
18	-	Reserved
19	Reset	<tbd>

8.5 Pin description

Table 25. Event router pin description

Pin	Direction	Description
WAKEUP0/1/	I/O	External wake-up input; can raise an event router interrupt and can cause wake-up from any of the low power modes. These pins can be configured to monitor the event router output through the CREG0 register (Table 38).
WAKEUP2/3	I	External wake-up input; can raise an event router interrupt and can cause wake-up from any of the low power modes.

8.6 Register description

Table 26. Register overview: Event router (base address 0x4004 4000)

Name	Access	Address offset	Description	Reset Value	Reference
HILO	R/W	0x000	Level configuration register	0x000	Table 27
EDGE	R/W	0x004	Edge configuration	0x000	Table 29
-	-	0x008 - 0xFD4	Reserved	-	-
CLR_EN	W	0xFD8	Clear event enable register	0x0	Table 30

Table 26. Register overview: Event router (base address 0x4004 4000)

Name	Access	Address offset	Description	Reset Value	Reference
SET_EN	W	0xFDC	Set event enable register	0x0	Table 31
STATUS	R	0xFE0	Event Status register	0x0	Table 32
ENABLE	R	0xFE4	Event Enable register	0x0	Table 33
CLR_STAT	W	0xFE8	Clear event status register	0x0	Table 34
SET_STAT	W	0xFEC	Set event status register	0x0	Table 35

8.6.1 Level configuration register

This register works in combination with the edge configuration register EDGE (see Table 29) to configure the level and edge detection for each input to the event router.

Table 27. Level configuration register (HILO - address 0x4004 4000) bit description

Bit	Symbol	Value	Description	Reset value
0	WAKEUP0_L		Level detect mode for WAKEUP0 event.	0
		0	Detect LOW level on the WAKEUP0 pin if bit 0 in the EDGE register is 0. Detect falling edge if bit 0 in the EDGE register is 1.	
		1	Detect HIGH level on the WAKEUP0 pin if bit 0 in the EDGE register is 0. Detect rising edge if bit 0 in the EDGE register is 1.	
1	WAKEUP1_L		Level detect mode for WAKEUP1 event. The corresponding bit in the EDGE register must be 0.	0
		0	Detect LOW level on the WAKEUP1 pin if bit 1 in the EDGE register is 0.	
		1	Detect HIGH level on the WAKEUP1 pin if bit 1 in the EDGE register is 0. Detect rising edge if bit 1 in the EDGE register is 1.	
2	WAKEUP2_L		Level detect mode for WAKEUP2 event.	0
		0	Detect LOW level on the WAKEUP2 pin if bit 2 in the EDGE register is 0. Detect falling edge if bit 2 in the EDGE register is 1.	
		1	Detect HIGH level on the WAKEUP2 pin if bit 2 in the EDGE register is 0. Detect rising edge if bit 2 in the EDGE register is 1.	
3	WAKEUP3_L		Level detect mode for WAKEUP3 event.	0
		0	Detect LOW level on the WAKEUP3 pin if bit 3 in the EDGE register is 0. Detect falling edge if bit 3 in the EDGE register is 1.	
		1	Detect HIGH level on the WAKEUP3 pin if bit 3 in the EDGE register is 0. Detect rising edge if bit 3 in the EDGE register is 1.	

Table 27. Level configuration register (HILO - address 0x4004 4000) bit description

Bit	Symbol	Value	Description	Reset value
4	ATIMER_L		Level detect mode for alarm timer event.	0
		0	Detect LOW level of the alarm timer interrupt if bit 4 in the EDGE register is 0. Detect falling edge if bit 4 in the EDGE register is 1.	
		1	Detect HIGH level of the alarm timer interrupt if bit 4 in the EDGE register is 0. Detect rising edge if bit 4 in the EDGE register is 1.	
5	RTC_L		Level detect mode for RTC event.	0
		0	Detect LOW level of the RTC interrupt if bit 5 in the EDGE register is 0. Detect falling edge if bit 5 in the EDGE register is 1.	
		1	Detect HIGH level of the RTC interrupt if bit 5 in the EDGE register is 0. Detect rising edge if bit 5 in the EDGE register is 1.	
6	BOD_L		Level detect mode for BOD event.	0
		0	Detect LOW level of the BOD interrupt if bit 6 in the EDGE register is 0. Detect falling edge if bit 6 in the EDGE register is 1.	
		1	Detect HIGH level of the BOD interrupt if bit 6 in the EDGE register is 0. Detect rising edge if bit 6 in the EDGE register is 1.	
7	WWDT_L		Level detect mode for WWDT event.	0
		0	Detect LOW level of the WWDT interrupt if bit 7 in the EDGE register is 0. Detect falling edge if bit 7 in the EDGE register is 1.	
		1	Detect HIGH level of the WWDT interrupt if bit 7 in the EDGE register is 0. Detect rising edge if bit 7 in the EDGE register is 1.	
8	ETH_L		Level detect mode for Ethernet event	0
		0	Detect LOW level of the Ethernet interrupt if bit 8 in the EDGE register is 0. Detect falling edge if bit 8 in the EDGE register is 1.	
		1	Detect HIGH level of the Ethernet interrupt if bit 8 in the EDGE register is 0. Detect rising edge if bit 8 in the EDGE register is 1.	
9	USB0_L		Level detect mode for USB0 event	0
		0	Detect LOW level of the USB0 interrupt if bit 9 in the EDGE register is 0. Detect falling edge if bit 9 in the EDGE register is 1.	
		1	Detect HIGH level of the USB0 interrupt if bit 9 in the EDGE register is 0. Detect rising edge if bit 9 in the EDGE register is 1.	

Table 27. Level configuration register (HILO - address 0x4004 4000) bit description

Bit	Symbol	Value	Description	Reset value
10	USB1_L		Level detect mode for USB1 event	0
		0	Detect LOW level of the USB1 interrupt if bit 10 in the EDGE register is 0. Detect falling edge if bit 10 in the EDGE register is 1.	
		1	Detect HIGH level of the USB1 interrupt if bit 10 in the EDGE register is 0. Detect rising edge if bit 10 in the EDGE register is 1.	
11	SDMMC_L		Level detect mode for SD/MMC event	0
		0	Detect LOW level of the SD/MMC interrupt if bit 11 in the EDGE register is 0. Detect falling edge if bit 11 in the EDGE register is 1.	
		1	Detect HIGH level of the SD/MMC interrupt if bit 11 in the EDGE register is 0. Detect rising edge if bit 11 in the EDGE register is 1.	
12	CAN_L		Level detect mode for C_CAN event.	0
		0	Detect LOW level of the combined C_CAN interrupt if bit 12 in the EDGE register is 0. Detect falling edge if bit 12 in the EDGE register is 1.	
		1	Detect HIGH level of the combined C_CAN interrupt if bit 12 in the EDGE register is 0. Detect rising edge if bit 12 in the EDGE register is 1.	
13	TIM2_L		Level detect mode for combined timer output 2 event.	0
		0	Detect LOW level GIMA output 25 if bit 13 in the EDGE register is 0. Detect falling edge if bit 13 in the EDGE register is 1.	
		1	Detect HIGH level GIMA output 25 if bit 13 in the EDGE register is 0. Detect rising edge if bit 13 in the EDGE register is 1.	
14	TIM6_L		Level detect mode for combined timer output 6 event.	0
		0	Detect LOW level of GIMA output 26 if bit 14 in the EDGE register is 0. Detect falling edge if bit 14 in the EDGE register is 1.	
		1	Detect HIGH level of GIMA output 26 if bit 14 in the EDGE register is 0. Detect rising edge if bit 14 in the EDGE register is 1.	
15	QEI_L		Level detect mode for QEI event.	0
		0	Detect LOW level of the QEI interrupt if bit 15 in the EDGE register is 0. Detect falling edge if bit 15 in the EDGE register is 1.	
		1	Detect HIGH level of the QEI interrupt if bit 15 in the EDGE register is 0. Detect rising edge if bit 15 in the EDGE register is 1.	

Table 27. Level configuration register (HILO - address 0x4004 4000) bit description

Bit	Symbol	Value	Description	Reset value
16	TIM14_L		Level detect mode for combined timer output 14 event.	0
		0	Detect LOW level of GIMA output 27 if bit 16 in the EDGE register is 0. Detect falling edge if bit 16 in the EDGE register is 1.	
		1	Detect HIGH level of GIMA output 27 if bit 16 in the EDGE register is 0. Detect rising edge if bit 16 in the EDGE register is 1.	
18:17	-	-	Reserved.	
19	RESET_L		Level detect mode for Reset	0
		0	Detect LOW level if bit 17 in the EDGE register is 0. Detect falling edge if bit 17 in the EDGE register is 1.	
		1	Detect HIGH level if bit 17 in the EDGE register is 0. Detect rising edge if bit 17 in the EDGE register is 1.	
31:20	-	-	Reserved.	

8.6.2 Edge configuration register

This register works in combination with the level configuration register HILO (see Table 27) to configure the level or edge detection for each input to the event router.

The EDGE configuration register determines whether the event router responds to a level change (EDGEn=1), or a constant level (EDGEn=0). The HILOn bit determines a response to a rising edge (HILOn=1) or a falling edge (HILOn=0).

Table 28. EDGE and HILO combined register settings

HILO bit n	EDGE bit n	Description
0	0	Detect LOW level
0	1	Detect falling edge
1	0	Detect HIGH level
1	1	Detect rising edge

When a HIGH level detect is active, the event router status bits cannot be cleared until the signal is LOW. When a rising edge detect is active, the event router status bit can be cleared right after the event has occurred.

Table 29. Edge configuration register (EDGE - address 0x4004 4004) bit description

Bit	Symbol	Value	Description	Reset value
0	WAKEUP0_E		Edge detect mode for WAKEUP0 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of WAKEUP0 pin. Detect falling edge if bit 0 in the HILO register is 0. Detect rising edge if bit 0 in the HILO register is 1.	

Table 29. Edge configuration register (EDGE - address 0x4004 4004) bit description

Bit	Symbol	Value	Description	Reset value
1	WAKEUP1_E		Edge/level detect mode for WAKEUP1 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of WAKEUP1 pin. Detect falling edge if bit 1 in the HILO register is 0. Detect rising edge if bit 1 in the HILO register is 1.	
2	WAKEUP2_E		Edge/level detect mode for WAKEUP2 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of WAKEUP2 pin. Detect falling edge if bit 2 in the HILO register is 0. Detect rising edge if bit 2 in the HILO register is 1.	
3	WAKEUP3_E		Edge/level detect mode for WAKEUP3 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of WAKEUP3 pin. Detect falling edge if bit 30 in the HILO register is 0. Detect rising edge if bit 3 in the HILO register is 1.	
4	ATIMER_E		Edge/level detect mode for alarm timer event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the alarm timer interrupt. Detect falling edge if bit 4 in the HILO register is 0. Detect rising edge if bit 4 in the HILO register is 1.	
5	RTC_E		Edge/level detect mode for RTC event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the RTC interrupt. Detect falling edge if bit 5 in the HILO register is 0. Detect rising edge if bit 5 in the HILO register is 1.	
6	BOD_E		Edge/level detect mode for BOD event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the BOD interrupt. Detect falling edge if bit 6 in the HILO register is 0. Detect rising edge if bit 6 in the HILO register is 1.	
7	WWDT_E		Edge/level detect mode for WWDTD event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the WWDT interrupt. Detect falling edge if bit 7 in the HILO register is 0. Detect rising edge if bit 7 in the HILO register is 1.	

Table 29. Edge configuration register (EDGE - address 0x4004 4004) bit description

Bit	Symbol	Value	Description	Reset value
8	ETH_E		The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the Ethernet interrupt. Detect falling edge if bit 8 in the HILO register is 0. Detect rising edge if bit 8 in the HILO register is 1.	
9	USB0_E		The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the USB0 interrupt. Detect falling edge if bit 9 in the HILO register is 0. Detect rising edge if bit 9 in the HILO register is 1.	
10	USB1_E		The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the USB1 interrupt. Detect falling edge if bit 10 in the HILO register is 0. Detect rising edge if bit 10 in the HILO register is 1.	
11	SDMMC_E		Edge/level detect mode for SD/MMC event.The corresponding bit in the EDGE register must be 0.	
		0	Level detect.	
		1	Edge detect of the SD/MMC interrupt. Detect falling edge if bit 10 in the HILO register is 0. Detect rising edge if bit 10 in the HILO register is 1.	
12	CAN_E		Edge/level detect mode for C_CAN event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the combined C_CAN interrupt. Detect falling edge if bit 12 in the HILO register is 0. Detect rising edge if bit 12 in the HILO register is 1.	
13	TIM2_E		Edge/level detect mode for combined timer output 2 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of GIMA output 25. Detect falling edge if bit 13 in the HILO register is 0. Detect rising edge if bit 13 in the HILO register is 1.	
14	TIM6_E		Edge/level detect mode for combined timer output 6 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of GIMA output 26. Detect falling edge if bit 14 in the HILO register is 0. Detect rising edge if bit 14 in the HILO register is 1.	

Table 29. Edge configuration register (EDGE - address 0x4004 4004) bit description

Bit	Symbol	Value	Description	Reset value
15	QEI_E		Edge/level detect mode for QEI interrupt signal. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of QEI interrupt. Detect falling edge if bit 15 in the HILO register is 0. Detect rising edge if bit 15 in the HILO register is 1.	
16	TIM14_E		Edge/level detect mode for combined timer output 14 event. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of GIMA output 27. Detect falling edge if bit 16 in the HILO register is 0. Detect rising edge if bit 16 in the HILO register is 1.	
18:17	-	-	Reserved.	
19	RESET_E		Edge/level detect mode for Reset. The corresponding bit in the EDGE register must be 0.	0
		0	Level detect.	
		1	Edge detect of the reset signal. Detect falling edge if bit 19 in the HILO register is 0. Detect rising edge if bit 19 in the HILO register is 1.	
31:20	-	-	Reserved.	

8.6.3 Clear event enable register

Table 30. Clear event enable register (CLR_EN - address 0x4004 4FD8) bit description

Bit	Symbol	Description	Reset value
0	WAKEUP0_CLREN	Writing a 1 to this bit clears the event enable bit 0 in the ENABLE register.	-
1	WAKEUP1_CLREN	Writing a 1 to this bit clears the event enable bit 1 in the ENABLE register.	-
2	WAKEUP2_CLREN	Writing a 1 to this bit clears the event enable bit 2 in the ENABLE register.	-
3	WAKEUP3_CLREN	Writing a 1 to this bit clears the event enable bit 3 in the ENABLE register.	-
4	ATIMER_CLREN	Writing a 1 to this bit clears the event enable bit 4 in the ENABLE register.	-
5	RTC_CLREN	Writing a 1 to this bit clears the event enable bit 5 in the ENABLE register.	-
6	BOD_CLREN	Writing a 1 to this bit clears the event enable bit 6 in the ENABLE register.	-
7	WWDT_CLREN	Writing a 1 to this bit clears the event enable bit 7 in the ENABLE register.	-
8	ETH_CLREN	Writing a 1 to this bit clears the event enable bit 8 in the ENABLE register.	-

Table 30. Clear event enable register (CLR_EN - address 0x4004 4FD8) bit description

Bit	Symbol	Description	Reset value
9	USB0_CLREN	Writing a 1 to this bit clears the event enable bit 9 in the ENABLE register.	-
10	USB1_CLREN	Writing a 1 to this bit clears the event enable bit 10 in the ENABLE register.	-
11	SDMMC_CLREN	Writing a 1 to this bit clears the event enable bit 11 in the ENABLE register.	-
12	CAN_CLREN	Writing a 1 to this bit clears the event enable bit 12 in the ENABLE register.	-
13	TIM2_CLREN	Writing a 1 to this bit clears the event enable bit 13 in the ENABLE register.	-
14	TIM6_CLREN	Writing a 1 to this bit clears the event enable bit 14 in the ENABLE register.	-
15	QEI_CLREN	Writing a 1 to this bit clears the event enable bit 15 in the ENABLE register.	-
16	TIM14_CLREN	Writing a 1 to this bit clears the event enable bit 16 in the ENABLE register.	-
18:17	-	Reserved.	-
19	RESET_CLREN	Writing a 1 to this bit clears the event enable bit 19 in the ENABLE register.	-
31:20	-	Reserved.	-

8.6.4 Set event enable register

Table 31. Event set enable register (SET_EN - address 0x4004 4FDC) bit description

Bit	Symbol	Description	Reset value
0	WAKEUP0_SETEN	Writing a 1 to this bit sets the event enable bit 0 in the ENABLE register.	-
1	WAKEUP1_SETEN	Writing a 1 to this bit sets the event enable bit 1 in the ENABLE register.	-
2	WAKEUP2_SETEN	Writing a 1 to this bit sets the event enable bit 2 in the ENABLE register.	-
3	WAKEUP3_SETEN	Writing a 1 to this bit sets the event enable bit 3 in the ENABLE register.	-
4	ATIMER_SETEN	Writing a 1 to this bit sets the event enable bit 4 in the ENABLE register.	-
5	RTC_SETEN	Writing a 1 to this bit sets the event enable bit 5 in the ENABLE register.	-
6	BOD_SETEN	Writing a 1 to this bit sets the event enable bit 6 in the ENABLE register.	-
7	WWDT_SETEN	Writing a 1 to this bit sets the event enable bit 7 in the ENABLE register.	-
8	ETH_SETEN	Writing a 1 to this bit sets the event enable bit 8 in the ENABLE register.	-
9	USB0_SETEN	Writing a 1 to this bit sets the event enable bit 9 in the ENABLE register.	-

Table 31. Event set enable register (SET_EN - address 0x4004 4FDC) bit description

Bit	Symbol	Description	Reset value
10	USB1_SETEN	Writing a 1 to this bit sets the event enable bit 10 in the ENABLE register.	-
11	SDMMC_SETEN	Writing a 1 to this bit sets the event enable bit 11 in the ENABLE register.	-
12	CAN_SETEN	Writing a 1 to this bit sets the event enable bit 12 in the ENABLE register.	-
13	TIM2_SETEN	Writing a 1 to this bit sets the event enable bit 13 in the ENABLE register.	-
14	TIM6_SETEN	Writing a 1 to this bit sets the event enable bit 14 in the ENABLE register.	-
15	QEI_SETEN	Writing a 1 to this bit sets the event enable bit 15 in the ENABLE register.	-
16	TIM14_SETEN	Writing a 1 to this bit sets the event enable bit 16 in the ENABLE register.	-
18:17	-	Reserved.	-
19	RESET_SETEN	Writing a 1 to this bit sets the event enable bit 19 in the ENABLE register.	-
31:20	-	Reserved.	-

8.6.5 Event status register

Table 32. Event status register (STATUS - address 0x4004 4FE0) bit description

Bit	Symbol	Description	Reset value
0	WAKEUP0_ST	A 1 in this bit shows that the WAKEUP0 event has been raised.	-
1	WAKEUP1_ST	A 1 in this bit shows that the WAKEUP1 event has been raised.	-
2	WAKEUP2_ST	A 1 in this bit shows that the WAKEUP2 event has been raised.	-
3	WAKEUP3_ST	A 1 in this bit shows that the WAKEUP3 event has been raised.	-
4	ATIMER_ST	A 1 in this bit shows that the ATIMER event has been raised.	-
5	RTC_ST	A 1 in this bit shows that the RTC event has been raised.	-
6	BOD_ST	A 1 in this bit shows that the BOD event has been raised.	-
7	WWDT_ST	A 1 in this bit shows that the WWDT event has been raised.	-
8	ETH_ST	A 1 in this bit shows that the ETHERNET event has been raised.	-
9	USB0_ST	A 1 in this bit shows that the USB0 event has been raised.	-
10	USB1_ST	A 1 in this bit shows that the USB1 event has been raised.	-
11	SDMMC_ST	A 1 in this bit indicates that the SDMMC event has been raised.	-
12	CAN_ST	A 1 in this bit shows that the C_CAN event has been raised.	-
13	TIM2_ST	A 1 in this bit shows that the combined timer 2 output event has been raised.	-
14	TIM6_ST	A 1 in this bit shows that the combined timer 6 output event has been raised.	-
15	QEI_ST	A 1 in this bit shows that the QEI event has been raised.	-
16	TIM14_ST	A 1 in this bit shows that the combined timer 14 output event has been raised.	-

Table 32. Event status register (STATUS - address 0x4004 4FE0) bit description

Bit	Symbol	Description	Reset value
18:17	-	Reserved.	-
19	RESET_ST	A 1 in this bit shows that the <tbd> event has been raised.	-
31:20	-	Reserved.	-

8.6.6 Event enable register

Table 33. Event enable register (ENABLE - address 0x4004 4FE4) bit description

Bit	Symbol	Description	Reset value
0	WAKEUP0_EN	A 1 in this bit shows that the WAKEUP0 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
1	WAKEUP1_EN	A 1 in this bit shows that the WAKEUP1 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
2	WAKEUP2_EN	A 1 in this bit shows that the WAKEUP2 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
3	WAKEUP3_EN	A 1 in this bit shows that the WAKEUP3 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
4	ATIMER_EN	A 1 in this bit shows that the ATIMER event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
5	RTC_EN	A 1 in this bit shows that the RTC event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
6	BOD_EN	A 1 in this bit shows that the BOD event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
7	WWDT_EN	A 1 in this bit shows that the WWDT event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
8	ETH_EN	A 1 in this bit shows that the ETHERNET event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
9	USB0_EN	A 1 in this bit shows that the USB0 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
10	USB1_EN	A 1 in this bit shows that the USB1 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
11	SDMMC_EN	A 1 in this bit indicates that the SDMMC event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
12	CAN_EN	A 1 in this bit shows that the CAN event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0

Table 33. Event enable register (ENABLE - address 0x4004 4FE4) bit description

Bit	Symbol	Description	Reset value
13	TIM2_EN	A 1 in this bit shows that the TIM2 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
14	TIM6_EN	A 1 in this bit shows that the TIM6 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
15	QEI_EN	A 1 in this bit shows that the QEI event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
16	TIM14_EN	A 1 in this bit shows that the TIM14 event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
18:17	-	Reserved	-
19	RESET_EN	A 1 in this bit shows that the RESET event has been enabled. This event wakes up the chip and contributes to the event router interrupt when bit 0 = 1 in the STATUS register.	0
31:20	-	Reserved.	-

8.6.7 Clear event status register

Table 34. Clear event status register (CLR_STAT - address 0x4004 4FE8) bit description

Bit	Symbol	Description	Reset value
0	WAKEUP0_CLRST	Writing a 1 to this bit clears the STATUS event bit 0 in the STATUS register.	-
1	WAKEUP1_CLRST	Writing a 1 to this bit clears the STATUS event bit 1 in the STATUS register.	-
2	WAKEUP2_CLRST	Writing a 1 to this bit clears the STATUS event bit 2 in the STATUS register.	-
3	WAKEUP3_CLRST	Writing a 1 to this bit clears the STATUS event bit 3 in the STATUS register.	-
4	ATIMER_CLRST	Writing a 1 to this bit clears the STATUS event bit 4 in the STATUS register.	-
5	RTC_CLRST	Writing a 1 to this bit clears the STATUS event bit 5 in the STATUS register.	-
6	BOD_CLRST	Writing a 1 to this bit clears the STATUS event bit 6 in the STATUS register.	-
7	WWDT_CLRST	Writing a 1 to this bit clears the STATUS event bit 7 in the STATUS register.	-
8	ETH_CLRST	Writing a 1 to this bit clears the STATUS event bit 8 in the STATUS register.	-
9	USB0_CLRST	Writing a 1 to this bit clears the STATUS event bit 9 in the STATUS register.	-
10	USB1_CLRST	Writing a 1 to this bit clears the STATUS event bit 10 in the STATUS register.	-
11	SDMMC_CLRST	Writing a 1 to this bit clears the STATUS event bit 11 in the STATUS register.	-

Table 34. Clear event status register (CLR_STAT - address 0x4004 4FE8) bit description

Bit	Symbol	Description	Reset value
12	CAN_CLRST	Writing a 1 to this bit clears the STATUS event bit 12 in the STATUS register.	-
13	TIM2_CLRST	Writing a 1 to this bit clears the STATUS event bit 13 in the STATUS register.	-
14	TIM6_CLRST	Writing a 1 to this bit clears the STATUS event bit 14 in the STATUS register.	-
15	QEI_CLRST	Writing a 1 to this bit clears the STATUS event bit 15 in the STATUS register.	-
16	TIM14_CLRST	Writing a 1 to this bit clears the STATUS event bit 16 in the STATUS register.	-
18:17	-		-
19	RESET_CLRST	Writing a 1 to this bit clears the STATUS event bit 19 in the STATUS register.	-
31:20	-	Reserved.	-

8.6.8 Set event status register

Table 35. Set event status register (SET_STAT - address 0x4004 4FEC) bit description

Bit	Symbol	Description	Reset value
0	WAKEUP0_SETST	Writing a 1 to this bit sets the STATUS event bit 0 in the STATUS register.	-
1	WAKEUP1_SETST	Writing a 1 to this bit sets the STATUS event bit 1 in the STATUS register.	-
2	WAKEUP2_SETST	Writing a 1 to this bit sets the STATUS event bit 2 in the STATUS register.	-
3	WAKEUP3_SETST	Writing a 1 to this bit sets the STATUS event bit 3 in the STATUS register.	-
4	ATIMER_SETST	Writing a 1 to this bit sets the STATUS event bit 4 in the STATUS register.	-
5	RTC_SETST	Writing a 1 to this bit sets the STATUS event bit 5 in the STATUS register.	-
6	BOD_SETST	Writing a 1 to this bit sets the STATUS event bit 6 in the STATUS register.	-
7	WWDT_SETST	Writing a 1 to this bit sets the STATUS event bit 7 in the STATUS register.	-
8	ETH_SETST	Writing a 1 to this bit sets the STATUS event bit 8 in the STATUS register.	-
9	USB0_SETST	Writing a 1 to this bit sets the STATUS event bit 9 in the STATUS register.	-
10	USB1_SETST	Writing a 1 to this bit sets the STATUS event bit 10 in the STATUS register.	-
11	SDMMC_SETST	Writing a 1 to this bit sets the STATUS event bit 11 in the STATUS register.	-
12	CAN_SETST	Writing a 1 to this bit sets the STATUS event bit 12 in the STATUS register.	-

Table 35. Set event status register (SET_STAT - address 0x4004 4FEC) bit description

Bit	Symbol	Description	Reset value
13	TIM2_SETST	Writing a 1 to this bit sets the STATUS event bit 13 in the STATUS register.	-
14	TIM6_SETST	Writing a 1 to this bit sets the STATUS event bit 14 in the STATUS register.	-
15	QEI_SETST	Writing a 1 to this bit sets the STATUS event bit 15 in the STATUS register.	-
16	TIM14_SETST	Writing a 1 to this bit sets the STATUS event bit 16 in the STATUS register.	-
18:17	-	Reserved.	-
19	RESET_SETST	Writing a 1 to this bit sets the STATUS event bit 19 in the STATUS register.	-
31:20	-	Reserved.	-

9.1 How to read this chapter

The available peripherals vary for different parts.

- Ethernet: available only on LPC4350/30.
- USB0: available only on LPC4350/30/20.
- USB1: available only on LPC4350/30.

If a peripheral is not available, the corresponding bits in the CREG registers are reserved.

9.2 Basic configuration

The CREG block is configured as follows:

- See Table 36 for clocking and power control.
- The CREG block cannot be reset by software.

Table 36. CREG clocking and power control

	Base clock	Branch clock	Operating frequency
CREG	BASE_M4_CLK	CLK_M4_CREG	up to 204 MHz

9.3 Features

The following settings are controlled in the configuration register block:

- ETB SRAM configuration
- BOD trip settings
- Oscillator output
- DMA-to-peripheral muxing
- Ethernet mode
- Memory mapping
- Timer/UART inputs
- USB1 PHY control
- RTC_ALARM and WAKEUP0/1 pin functions

In addition, the Creg block contains the part id and the part configuration information.

9.4 Register description

Table 37. Register overview: Configuration registers (base address 0x4004 3000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC, UART0/3 boot	Reset value after USB0/1 boot	Reference
-	-	0x000	Reserved	-	-	-	-
CREG0	R/W	0x004	Chip configuration register 32 kHz oscillator output and BOD control register.	<tbd>	0xF3C	0xF1C	Table 38
-	-	0x008	Reserved	-	-	-	-
-	-	0x008 - 0x0FC	Reserved	-	-	-	-
M4MEMMAP	R/W	0x100	ARM Cortex-M4 memory mapping	0x10400	0x1000 0000	0x1000 0000	Table 39
-	-	0x104	Reserved	<tbd>	<tbd>	<tbd>	<tbd>
CREG1	RO	0x108	Chip configuration register 1	<tbd>	<tbd>	<tbd>	<tbd>
CREG2	RO	0x10C	Chip configuration register 2	<tbd>	<tbd>	<tbd>	<tbd>
CREG3	RO	0x110	Chip configuration register 3	<tbd>	<tbd>	<tbd>	<tbd>
CREG4	RO	0x114	Chip configuration register 4	<tbd>	<tbd>	<tbd>	<tbd>
CREG5	R/W	0x118	Chip configuration register 5. Controls JTAG access.	<tbd>	<tbd>	<tbd>	Table 40
DMAMUX	R/W	0x11C	DMA muxing control	<tbd>	<tbd>	<tbd>	Table 41
-	-	0x120 - 0x124	Reserved	-	-	-	-
ETBCFG	R/W	0x128	ETB RAM configuration	0	0x1	<tbd>	Table 42
CREG6	R/W	0x12C	Chip configuration register 6. Controls multiple functions : Ethernet interface, SCT output, I2S0/1 inputs, EMC clock.	0	<tbd>	<tbd>	Table 43
M4TXEVENT	R/W	0x130	Cortex-M4 TXEV event clear	0	<tbd>	<tbd>	Table 44
-	-	0x134 - 0x1FC	Reserved	-	-	-	-
CHIPID	RO	0x200	Part ID	<tbd>	<tbd>	<tbd>	Table 45
-	-	0x204 - 0x2FC	Reserved	-	-	-	-
-	-	0x300	Reserved	-	-	-	-
-	-	0x304	Reserved	-	-	-	-
-	-	0x308	Reserved	-	-	-	-
-	-	0x30C - 0x3FC	Reserved	-	-	-	-
M0TXEVENT	R/W	0x400	Cortex-M0 TXEV event clear	0	<tbd>	<tbd>	Table 46
M0APPMEMMAP	R/W	0x404	ARM Cortex-M0 memory mapping	0x20000	<tbd>	<tbd>	Table 47

9.4.1 CREG0 control register

Table 38. CREG0 register (CREG0, address 0x4004 3004) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	EN1KHZ		Enable 1 kHz output.	0	R/W
		0	1 kHz output disabled.		
		1	1 kHz output enabled.		
1	EN32KHZ		Enable 32 kHz output	0	R/W
		0	32 kHz output disabled.		
		1	32 kHz output enabled.		
2	RESET32KHZ		32 kHz oscillator reset	1	R/W
		0	Clear reset.		
		1	Reset active.		
3	PD32KHZ		32 kHz power control.	1	R/W
		0	Powered.		
		1	Powered-down.		
4	-		Reserved	-	-
5	USB0PHY		USB0 PHY power control.	1	R/W
		0	Enable USB0 PHY power.		
		1	Disable USB0 PHY. PHY powered down.		
7:6	ALARMCTRL		RTC_ALARM pin output control	0	R/W
		0x0	RTC alarm.		
		0x1	Event router event.		
		0x2	Reserved.		
		0x3	Inactive.		
9:8	BODLVL1		BOD trip level to generate an interrupt.	0x3	R/W
		0x0	2.8 V		
		0x1	2.9 V		
		0x2	3.0 V		
		0x3	3.1 V		
11:10	BODLVL2		BOD trip level to generate a reset.	0x3	R/W
		0x0	1.75 V		
		0x1	1.85 V		
		0x2	1.95 V		
		0x3	2.05 V		
13:12	-		Reserved	-	-
15:14	WAKEUP0CTRL		WAKEUP0 pin input/output control	0	R/W
		0x0	Input to the event router.		
		0x1	Output from the event router.		
		0x2	Reserved.		
		0x3	Input to the event router.		

Table 38. CREG0 register (CREG0, address 0x4004 3004) bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
17:16	WAKEUP1CTRL		WAKEUP1 pin input/output control	0	R/W
		0x0	Input to event router.		
		0x1	Output from the event router.		
		0x2	Reserved		
		0x3	Input to event router.		
31:18	-		Reserved	-	-

9.4.2 ARM Cortex-M4 memory mapping register

Table 39. Memory mapping register (M4MEMMAP, address 0x4004 3100) bit description

Bit	Symbol	Description	Reset value	Access
11:0		Reserved	0x000	-
31:12	M4MAP	Shadow address when accessing memory at address 0x0000 0000	0x10400	R/W

9.4.3 CREG5 control register

Table 40. CREG5 control register (CREG5, address 0x4004 3118) bit description

Bit	Symbol	Value	Description	Reset value	Access
4:0	-		Reserved.	-	-
5	-		Reserved.	0	-
6	M4TAPSEL		JTAG debug select <tbd>	<tbd>	R/W
		0	Bypass JTAG debug.		
		1	Enable JTAG debug.		
7	-		Reserved.	0	-
8	-		Reserved.	0	-
31:9	-		Reserved.	-	-

9.4.4 DMA muxing register

This register controls which set of peripherals is connected to the DMA controller (see Table 260).

Table 41. DMA muxing register (DMAMUX, address 0x4004 311C) bit description

Bit	Symbol	Value	Description	Reset value	Access
1:0	DMAMUXCH0		Select DMA to peripheral connection for DMA peripheral 0.	0	R/W
		0x0	SPIFI		
		0x1	SCT match 2		
		0x2	Reserved		
		0x3	T3 match 1		

Table 41. DMA muxing register (DMAMUX, address 0x4004 311C) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
5:4	DMAMUXCH2		Select DMA to peripheral connection for DMA peripheral 2.	0	R/W
		0x0	Timer 0 match 0		
		0x1	USART0 transmit		
		0x2	Reserved		
		0x3	Reserved		
7:6	DMAMUXCH3		Select DMA to peripheral connection for DMA peripheral 3.	0	R/W
		0x0	Timer 1 match 0		
		0x1	UART1 transmit		
		0x2	I2S1 channel 0		
		0x3	SSP1 transmit		
9:8	DMAMUXCH4		Select DMA to peripheral connection for DMA peripheral 4.	0	R/W
		0x0	Timer 1 match 1		
		0x1	UART1 receive		
		0x2	I2S1 channel 1		
		0x3	SSP1 receive		
11:10	DMAMUXCH5		Select DMA to peripheral connection for DMA peripheral 5.	0	R/W
		0x0	Timer 2 match 0		
		0x1	USART2 transmit		
		0x2	SSP1 transmit		
		0x3	Reserved		
13:12	DMAMUXCH6		Selects DMA to peripheral connection for DMA peripheral 6.	0	R/W
		0x0	Timer 2 match 1		
		0x1	USART2 receive		
		0x2	SSP1 receive		
		0x3	Reserved		
15:14	DMAMUXCH7		Selects DMA to peripheral connection for DMA peripheral 7.	0	R/W
		0x0	Timer 3 match l 0		
		0x1	USART3 transmit		
		0x2	SCT match output 0		
		0x3	Reserved		

Table 41. DMA muxing register (DMAMUX, address 0x4004 311C) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
5:4	DMAMUXCH2		Select DMA to peripheral connection for DMA peripheral 2.	0	R/W
		0x0	Timer 0 match 1		
		0x1	USART0 receive		
		0x2	Reserved		
		0x3	Reserved		
7:6	DMAMUXCH3		Select DMA to peripheral connection for DMA peripheral 3.	0	R/W
		0x0	Timer 1 match 0		
		0x1	UART1 transmit		
		0x2	I2S1 channel 0		
		0x3	SSP1 transmit		
9:8	DMAMUXCH4		Select DMA to peripheral connection for DMA peripheral 4.	0	R/W
		0x0	Timer 1 match 1		
		0x1	UART1 receive		
		0x2	I2S1 channel 1		
		0x3	SSP1 receive		
11:10	DMAMUXCH5		Select DMA to peripheral connection for DMA peripheral 5.	0	R/W
		0x0	Timer 2 match 0		
		0x1	USART2 transmit		
		0x2	SSP1 transmit		
		0x3	Reserved		
13:12	DMAMUXCH6		Selects DMA to peripheral connection for DMA peripheral 6.	0	R/W
		0x0	Timer 2 match 1		
		0x1	USART2 receive		
		0x2	SSP1 receive		
		0x3	Reserved		
15:14	DMAMUXCH7		Selects DMA to peripheral connection for DMA peripheral 7.	0	R/W
		0x0	Timer 3 match l 0		
		0x1	USART3 transmit		
		0x2	SCT match output 0		
		0x3	Reserved		

Table 41. DMA muxing register (DMAMUX, address 0x4004 311C) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
19:18	DMAMUXCH9		Select DMA to peripheral connection for DMA peripheral 9.	0	R/W
		0x0	SSP0 receive		
		0x1	I2S0 channel 0		
		0x2	SCT match output 1		
		0x3	Reserved		
21:20	DMAMUXCH10		Select DMA to peripheral connection for DMA peripheral 10.	0	R/W
		0x0	SSP0 transmit		
		0x1	I2S0 channel 1		
		0x2	SCT match output 0		
		0x3	Reserved		
23:22	DMAMUXCH11		Selects DMA to peripheral connection for DMA peripheral 11.	0	R/W
		0x0	SSP1 receive		
		0x1	Reserved		
		0x2	USART0 transmit		
		0x3	Reserved		
25:24	DMAMUXCH12		Select DMA to peripheral connection for DMA peripheral 12.	0	R/W
		0x0	SSP1 transmit		
		0x1	Reserved		
		0x2	USART0 receive		
		0x3	Reserved		
27:26	DMAMUXCH13		Select DMA to peripheral connection for DMA peripheral 13.	0	R/W
		0x0	ADC0		
		0x1	Reserved		
		0x2	SSP1 receive		
		0x3	USART3 receive		
29:28	DMAMUXCH14		Select DMA to peripheral connection for DMA peripheral 14.	0	R/W
		0x0	ADC1		
		0x1	Reserved		
		0x2	SSP1 transmit		
		0x3	USART3 transmit		
31:30	DMAMUXCH15		Select DMA to peripheral connection for DMA peripheral 15.	0	R/W
		0x0	DAC		
		0x1	SCT match output 3		
		0x2	Reserved		
		0x3	Timer 3 match 0		

9.4.5 ETB SRAM configuration register

This register selects the interface that is used to the 16 kB block of RAM located at address 0x2000 C000. This RAM memory block can be accessed either by the ETB or be used as normal SRAM on the AHB bus.

Note that by default, this memory area will be accessed by the ETB.

Table 42. ETB SRAM configuration register (ETBCFG, address 0x4004 3128) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ETB		Select SRAM interface	0	R/W
		0	ETB accesses SRAM at address 0x2000 C000.		
		1	AHB accesses SRAM at address 0x2000 C000.		
31:1	-		Reserved.	-	-

9.4.6 CREG6 control register

This register controls various aspects of the LLPC43xx:

- Bits 2:0 control the Ethernet PHY interface. The ethernet block reads this register during set-up. Therefore the ethernet must be reset after changing the PHY interface.
- Bit 4 selects the functionality of the SCT outputs connected to the CTOUT_n pins and selected GIMA inputs:
  - SCT output ORed with timer match output (default). See <tbd>
  - SCT output only.
- Bits 12:15 control the I2S clock connections.
- Bit 16 controls the external memory controller clocking.

Table 43. CREG6 control register (CREG6, address 0x4004 312C) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	ETHMODE		Selects the Ethernet mode. Reset the ethernet after changing the PHY interface. All other settings are reserved.		R/W
		0x0	MII		
		0x4	RMII		
3	-		Reserved.		R/W
4	CTOUTCTRL		Selects the functionality of the SCT outputs.	0	R/W
		0	Combine SCT and timer match outputs. SCT outputs are ORed with timer outputs.		
		1	SCT outputs only. SCT outputs are used without timer match outputs.		
11:5	-		Reserved.	-	-

Table 43. CREG6 control register (CREG6, address 0x4004 312C) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
12	I2S0_TX_SCK_IN_SEL		I2S0_TX_SCK input select	0	R/W
		0	I2 S clock selected as defined by the I2S transmit mode register Table 895.		
		1	Audio PLL for I2S transmit clock MCLK input and MCLK output. The I2S must be configured in slave mode.		
13	I2S0_RX_SCK_IN_SEL		I2S0_RX_SCK input select	0	R/W
		0	I2 S clock selected as defined by the I2S receive mode register Table 896.		
		1	Audio PLL for I2S receive clock MCLK input and MCLK output. The I2S must be configured in slave mode.		
14	I2S1_TX_SCK_IN_SEL		I2S1_TX_SCK input select	0	R/W
		0	I2 S clock selected as defined by the I2S transmit mode register Table 895.		
		1	Audio PLL (PLL0AUDIO) for I2S transmit clock MCLK input and MCLK output. The I2S must be configured in slave mode.		
15	I2S1_RX_SCK_IN_SEL		I2S1_RX_SCK input select	0	R/W
		0	I2 S clock selected as defined by the I2S receive mode register Table 896.		
		1	Audio PLL (PLL0AUDIO) for I2S receive clock MCLK input and MCLK output. The I2S must be configured in slave mode.		
16	EMC_CLK_SEL		EMC_CLK divided clock select (see Section 21.1).	0	R/W
		0	EMC_CLK_DIV not divided.		
		1	EMC_CLK_DIV divided by 2.		
31:17	-		Reserved.	-	-

9.4.7 Cortex-M4 TXEV event clear register

This register captures the signal TXEV from the ARM Cortex-M4 processor (see Section 2.4.2).

Table 44. M4 TXEV clear register (M4TXEVENT, address 0x4004 3130) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	TXEVCLR		Cortex-M4 TXEV event.	0	R/W
		0	Clear the TXEV event.		
		1	No effect.		
31:1	-		Reserved.	-	-

9.4.8 Part ID register

Table 45. Part ID register (CHIPID, address 0x4004 3200) bit description

Bit	Symbol	Description	Reset value	Access
31:0	ID	<tbd>	-	RO

9.4.9 Cortex-M0 TXEV event clear register

This register captures the signal TXEV from the ARM Cortex-M0 processor (see Section 2.4.2).

Table 46. Cortex-M0 TXEV clear register (M0TXEVENT, address 0x4004 3400) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	TXEVCLR		Cortex-M0 TXEV event.	0	R/W
		0	Clear the TXEV event.		
		1	No effect.		
31:1	-		Reserved.	-	-

9.4.10 ARM Cortex-M0 memory mapping register

Table 47. Memory mapping register (M0APPMEMMAP, address 0x4004 3404) bit description

Bit	Symbol	Description	Reset value	Access
11:0		Reserved	0	-
31:12	M0APPMAP	Shadow address when accessing memory at address 0x0000 0000	0x20000	R/W

10.1 How to read this chapter

The power management controller is identical on all LPC43xx parts.

10.2 Basic configuration

The PMC controls the power-down modes (Deep-sleep, Power-down, and Deep power-down).

10.3 General description

The PMC implements the control sequences to enable transitioning between different power modes and controls the power state of each peripheral. In addition, wake-up from any of the power-down modes based on hardware events is supported.

Power-down modes can be reached from Active mode only, and transitions between Power-down modes are not allowed. Power-down modes are entered by a WFI or WFE instruction. Wake-up is caused by an interrupt or event. The interrupt cause is captured in the NVIC (Chapter 7), and events are captured in the Event router (Chapter 8).

The PMC supports the following Power-down modes: Deep-sleep, Power-down, and Deep power-down. The wake-up from any of the Power-down modes will always result in the Active mode.

The LPC43xx supports five power modes in order from highest to lowest power consumption:

1. Active mode
2. Sleep mode (controlled by the ARM Cortex-M4 core)
3. Power-down modes:
   a. Deep-sleep mode (controlled by the ARM Cortex-M4 core)
   b. Power-down mode (controlled by the ARM Cortex-M4 core)
   c. Deep power-down mode

Remark: Before the Deep-sleep mode or Power-down mode can be selected, the IRC must be selected as the clock source for all output clocks through the CGU registers (see Section 11.8.1), and all PLLs must be powered down.

10.3.1 Active mode

By default, the LPC43xx is in Active mode, which means that every peripheral can perform a functional operation at nominal operating conditions. The other low-power modes are standby modes in which the peripheral clocks are disabled and operating conditions are adapted to achieve further power savings. The peripheral clocks are enabled again at wake-up.

In Active (or Sleep mode), the following operating modes are supported. The operating modes are programmable through an API.

- Reduced power mode: The CPU and core logic are optimized for power efficiency rather than for highest performance.
- Normal power mode: The CPU operates at the nominal performance level. This is the default after reset.
- High-performance mode: The CPU operates at high performance level.

10.3.2 Sleep mode

In Sleep mode, the CPU clock is shut down to save power; the peripherals can still remain active and fully functional. The Sleep mode is entered by a WFI or WFE instruction if the SLEEPDEEP bit in the ARM Cortex-M4 system control register is set to 0.

The same operating modes (reduced power, normal, and high-performance) are supported as in Active mode.

10.3.3 Deep-sleep mode

In Deep-sleep mode the CPU clock and peripheral clocks are shut down to save power; logic states and SRAM memory are maintained. All analog blocks and the BOD control circuit are powered down. The Deep-sleep mode is entered by a WFI or WFE instruction if the SLEEPDEEP bit in the ARM Cortex-M4 system control register is set to 1 and the PD0_SLEEP0_MODE register (see Table 51) is programmed with the Deep-sleep mode value.

When the LPC43xx wakes up from Deep-sleep mode, the 12 MHz IRC is used as the clock source for all base clocks.

Remark: Before entering Deep-sleep mode, program the CGU as follows:

- Switch the clock source of all base clocks to the IRC.
- Put the PLLs in power-down mode.

Reprogramming the CGU avoids any undefined or unlocked PLL clocks at wake-up and minimizes power consumption during Deep-sleep mode.

10.3.4 Power-down mode

In Power-down mode the CPU clock and peripheral clocks are shut down but logic states are maintained. All SRAM memory except for the upper 8 kB of the local SRAM located at 0x1008 0000, all analog blocks, and the BOD control circuit are powered down.The Power-down mode is entered by a WFI or WFE instruction if the SLEEPDEEP bit in the ARM Cortex-M4 system control register is set to 1 and the PD0_SLEEP0_MODE register (see Table 51) is programmed with the Power-down mode value.

When the LPC43xx wakes up from Power-down mode, the 12 MHz IRC is used as the clock source for all base clocks.

Remark: Before entering Power-down mode, program the CGU as follows:

- Switch the clock source of all base clocks to the IRC.

- Put the PLLs in power-down mode.

Reprogramming the CGU avoids any undefined or unlocked PLL clocks at wake-up and minimizes power consumption during Power-down mode.

10.3.5 Deep power-down

In Deep power-down mode the entire core logic is powered down and the logic state of the entire system including the I/O pads is lost. Only the logic in the RTC power domain remains active. The Deep power-down mode is entered by a WFI or WFE instruction if the SLEEPDEEP bit in the ARM Cortex-M4 system control register is set to 1 and the PD0_SLEEP0_MODE register (see Table 51) is programmed with the Deep power-down value.

When the LPC43xx wakes up from Deep power-down mode, the boot loader configures the PLL1 as the clock source running at 96 MHz.

10.3.6 Memory retention in Power-down modes

Table 48 shows which parts of the SRAM memory are preserved in Sleep mode and the various power-down modes.

In addition, all FIFO memory contained in the peripheral blocks (USB0/1, LCD, CAN, Ethernet, USART0/2/3, UART) is retained in Sleep mode and Deep-sleep mode but not in Power-down mode and Deep-power-down mode.

Table 48. Memory retention

Mode	128 kB local SRAM starting at 0x1000 0000	64 kB local SRAM starting at 0x1008 0000	8 kB local SRAM starting at 0x1009 0000	64 kB AHB SRAM starting at 0x2000 0000	256 byte backup registers at 0x4004 1000 (RTC power domain)
Sleep mode	yes	yes	yes	yes	yes
Deep-sleep mode	yes	yes	yes	yes	yes
Power-down mode	no	no	yes	no	yes
Deep power-down mode	no	no	no	no	yes

10.4 Register description

Table 49. Register overview: Power Mode Controller (PMC) (base address 0x4004 2000)

Name	Access	Address offset	Description	Reset value	Reference
PD0_SLEEP0_HW_ENA	R/W	0x000	Hardware sleep event enable register	0x0000 0001	Table 50
-	-	0x004 - 0x018	Reserved	-	-
PD0_SLEEP0_MODE	R/W	0x01C	Power-down mode control register	0x003F FF7F	Table 51

10.4.1 Hardware sleep event enable register PD0_SLEEP0_HW_ENA

Table 50. Hardware sleep event enable register (PD0_SLEEP0_HW_ENA - address 0x4004 2000) bit description

Bit	Symbol	Description	Reset value	Access
0	ENA_EVENT0	Writing a 1 enables any Power-down modes for Cortex-M4.	1	R/W
31:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-	-

10.4.2 Power-down modes register PD0_SLEEP0_MODE

The PD0_SLEEP0_MODE register controls which of the three reduced power modes, Deep-sleep, Power-down, or Deep power-down is entered when an ARM WFE/WFI instruction is issued and the SLEEPDEEP bit is set to 1.

Table 51. Power-down modes register (PD0_SLEEP0_MODE - address 0x4004 201C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	PWR_STATE	Selects between Deep-sleep, Power-down, and Deep power-down modes. Only one of the following three values can be programmed in this register: 0x0030 00AA = Deep-sleep mode 0x0030 FCBA = Power-down mode 0x003F FF7F = Deep power-down mode	0x003F FF7F	R/W

10.5 Functional description

10.5.1 Run-time programming

The PD0_SLEEP0_MODE register can be programmed at run-time to change the default power state of the LPC43xx after the next transition to a reduced-power state.

Table 52. Typical settings for PMC power modes

Power-down mode	Description	PD0_SLEEP0_MODE register bit settings
Deep-sleep	CPU, peripherals, analog, USB PHY in retention mode; all SRAM supplies in active mode; BOD in power-down mode.	0x0030 00AA
Power-down	CPU, peripherals, analog supplies in retention mode; USB PHY in power-down mode; SRAM1 in active mode; all other SRAMs in power-down mode; BOD in power-down mode.	0x0030 FCBA
Deep power-down	CPU, peripherals, analog, USB PHY in power-down mode; all SRAMs in power-down mode; BOD in power-down mode.	0x003F FF7F

11.1 How to read this chapter

Ethernet, USB0, USB1, and LCD related clocks are not available on all packages. See Section 1.3. The corresponding clock control registers are reserved.

The VADC peripheral is available on <tbd> parts only.

11.2 Basic configuration

The CGU is configured as follows:

- See Table 53 for clocking and power control.
- Do not reset the CGU during normal operation.

Table 53. CGU clocking and power control

	Base clock	Branch clock	Operating frequency
CGU	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz

11.3 Features

- PLL control
- Supports three PLLs:
  - the PLL0USB for creating the 480 MHz clock for the high-speed USB0
  - the PLL0AUDIO with fractional divider for creating a wide variety of frequencies for audio applications with high accuracy
  - the PLL1 for creating the core and peripheral clocks.
- Oscillator control
- Clock generation and clock source multiplexing
- Integer dividers for clock output stages

11.4 General description

The CGU generates multiple independent clocks for the core and the peripheral blocks of the LPC43x. Each independent clock is called a base clock and itself is one of the inputs to the two Clock Control Units (CCUs) which control the branch clocks to the individual peripherals (see Chapter 12).

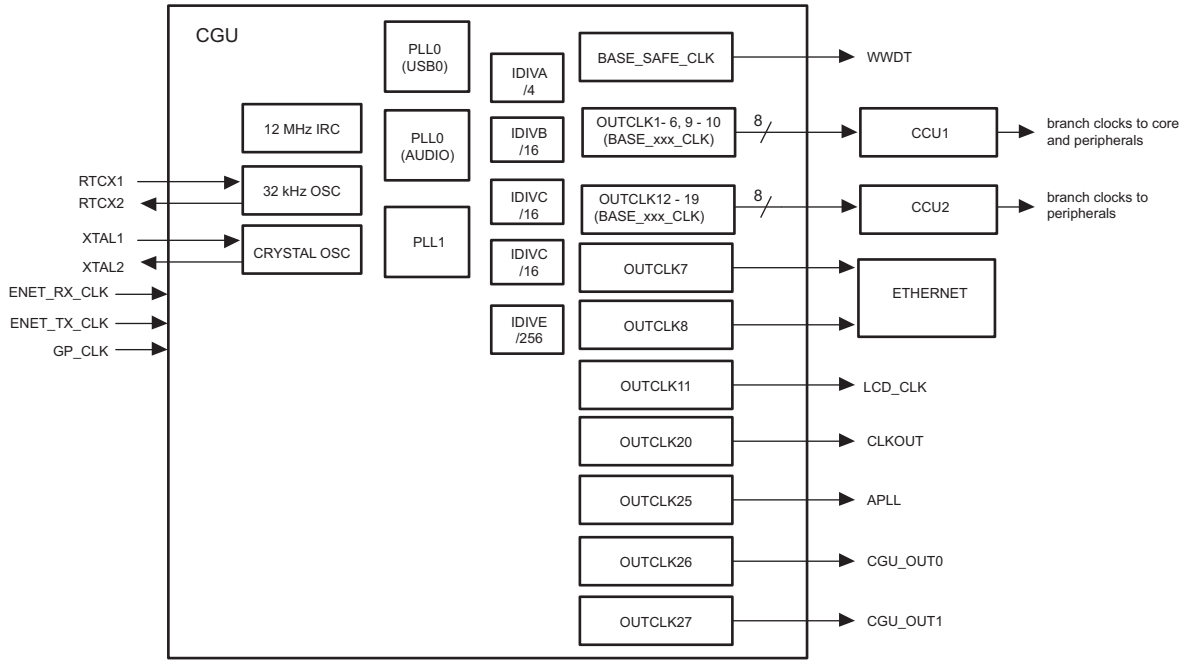


Fig 20. CGU and CCU0/1 block diagram

The CGU selects the inputs to the clock generators from multiple clock sources, controls the clock generation, and routes the outputs of the clock generators through the clock source bus to the output stages. Each output stage provides an independent clock source and corresponds to one of the base clocks for the LPC43xx. See Table 54 for a description of each base clock and Table 56 for the possible clock sources for each base clock.

The CGU contains four types of clock generators:

1. External clock inputs and internal clocks: The external clock inputs are the Ethernet PHY clocks and the general purpose input clock GP_CLKIN. The clocks from the internal oscillators are the IRC and the 32 kHz oscillator output clocks. These clock generators have no selectable inputs from the clock source bus and provide one clock output each to the clock source bus.
2. Crystal oscillator: The crystal oscillator is controlled by the CGU. The input to the crystal oscillator are the XTAL pins. The crystal oscillator creates one output to the clock source bus.
3. PLLs: PLL0USB0, PLL0AUDIO, and PLL1 are controlled by the CGU. Each PLL can select one input from the clock source bus and provides one output to the clock source bus. The input to the PLLs can be selected from all external and internal clocks and oscillators, from the other PLLs, and from the outputs of any of the integer dividers (see Table 55). One PLL0 cannot select the other PLL0 as input.
4. Integer dividers: Each of the five integer dividers can select one input from the clock source bus and creates one divided output clock to the clock source bus. The input to all integer dividers can be selected from all external and internal clocks and oscillators, and from all three PLLs. In addition, the output of the first integer divider can be selected as an input to all other integer dividers (see Table 55).

- Integer divider A: maximum division factor = 4 (see Table 72).
- Integer dividers B, C, D: maximum division factor = 16 (see Table 73).
- Integer divider E: maximum division factor = 256 (see Table 74).

The output stages select a clock source from the clock source bus for each base clock (see Table 56). Except for the base clocks to the WWDT (BASE_SAFE_CLK) and USB0 (BASE_USB0_CLK), the clock source for each output stage can be any of the external and internal clocks and oscillators directly or one of the PLL outputs or any of the outputs of the integer dividers.

Table 54. CGU0 base clocks

Number	Name	Frequency [1]	Description
0	BASE_SAFE_CLK	12 MHz	Base safe clock (always on) for WDT
1	BASE_USB0_CLK	480 MHz	Base clock for USB0
2	BASE_PERIPH_CLK	204 MHz	Base clock for SGPIO peripheral
3	BASE_USB1_CLK	204 MHz	Base clock for USB1
4	BASE_M4_CLK	204 MHz	System base clock for ARM Cortex-M4 core and APB peripheral blocks #0 and #2
5	BASE_SPIFI_CLK	204 MHz	Base clock for SPIFI
6	BASE_SPI_CLK	204 MHz	Base clock for SPI
7	BASE_PHY_RX_CLK	75 MHz	Base clock for Ethernet PHY Receive clock
8	BASE_PHY_TX_CLK	75 MHz	Base clock for Ethernet PHY Transmit clock
9	BASE_APB1_CLK	204 MHz	Base clock for APB peripheral block # 1
10	BASE_APB3_CLK	204 MHz	Base clock for APB peripheral block # 3
11	BASE_LCD_CLK	204 MHz	Base clock for LCD
12	BASE_VADC_CLK	204 MHz	Base clock for VADC
13	BASE_SDIO_CLK	204 MHz	Base clock for SD/MMC
14	BASE_SSP0_CLK	204 MHz	Base clock for SSP0
15	BASE_SSP1_CLK	204 MHz	Base clock for SSP1
16	BASE_UART0_CLK	204 MHz	Base clock for UART0
17	BASE_UART1_CLK	204 MHz	Base clock for UART1
18	BASE_UART2_CLK	204 MHz	Base clock for UART2
19	BASE_UART3_CLK	204 MHz	Base clock for UART3
20	BASE_OUT_CLK	204 MHz	Base clock for CLKOUT pin
24-21	-	-	Reserved
25	BASE_APLL_CLK	204 MHz	Base clock for audio PLL
26	BASE_CGU_OUT0_CLK	204 MHz	Base clock for CGU_OUT0 clock output
27	BASE_CGU_OUT1_CLK	204 MHz	Base clock for CGU_OUT1 clock output

[1] Maximum frequency that guarantees stable operation of the LPC43xx.

Table 55 shows all available input clock sources for each clock generator.

Table 55. Clock sources for clock generators with selectable inputs

Clock sources	Clock generators							
	PLL0 USB	PLL0 AUDIO	PLL1	IDIVA /4	IDIVB /16	IDIVC /16	IDIVD /16	IDIVE /256
32 kHz oscillator	yes	yes	yes	yes	yes	yes	yes	yes
IRC 12 MHz	yes	yes	yes	yes	yes	yes	yes	yes
ENET_RX_CLK	yes	yes	yes	yes	yes	yes	yes	yes
ENET_TX_CLK	yes	yes	yes	yes	yes	yes	yes	yes
GP_CLKIN	yes	yes	yes	yes	yes	yes	yes	yes
Crystal oscillator	yes	yes	yes	yes	yes	yes	yes	yes
PLL0USB	no	no	yes	yes	no	no	no	no
PLL0AUDIO	no	no	yes	yes	yes	yes	yes	yes
PLL1	yes	yes	no	yes	yes	yes	yes	yes
IDIVA	yes	yes	yes	no	yes	yes	yes	yes
IDIVB	yes	yes	yes	no	no	no	no	no
IDIVC	yes	yes	yes	no	no	no	no	no
IDIVD	yes	yes	yes	no	no	no	no	no
IDIVE	yes	yes	yes	no	no	no	no	no

Table 56. Clock sources for output stages

Clock sources	Output stages (d = default clock source, y = yes (clock source available), n = no (clock source not available))																							
	BASE_SAFE_CLK	BASE_USB0_CLK	BASE_PERIPH_CLK	BASE_USB1_CLK	BASE_M4_CLK	BASE_SPIFI_CLK	BASE_SPI_CLK	BASE_PHY_RX_CLK	BASE_PHY_TX_CLK	BASE_APB1_CLK	BASE_APB3_CLK	BASE_LCD_CLK	BASE_VADC_CLK	BASE_SDIO_CLK	BASE_SSP0_CLK	BASE_SSP1_CLK	BASE_UART0_CLK	BASE_UART1_CLK	BASE_UART2_CLK	BASE_UART3_CLK	BASE_OUT_CLK	BASE_APLL_CLK	BASE_CGU_OUT0_CLK	BASE_CGU_OUT1_CLK
32 kHz oscillator	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
IRC 12 MHz	d	n	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d	d
ENET_RX_CLK	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
ENET_TX_CLK	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
GP_CLKIN	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
Crystal oscillator	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
PLL0 USB	n	d	n	y	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	n	y	n	y	y
PLL0 AUDIO	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y

Table 56. Clock sources for output stages

Clock sources	Output stages (d = default clock source, y = yes (clock source available), n = no (clock source not available))																						
	BASE_SAFE_CLK	BASE_USB0_CLK	BASE_PERIPH_CLK	BASE_USB1_CLK	BASE_M4_CLK	BASE_SPIFI_CLK	BASE_SPI_CLK	BASE_PHY_RX_CLK	BASE_PHY_TX_CLK	BASE_APB1_CLK	BASE_APB3_CLK	BASE_LCD_CLK	BASE_VADC_CLK	BASE_SDIO_CLK	BASE_SSP0_CLK	BASE_SSP1_CLK	BASE_UART0_CLK	BASE_UART1_CLK	BASE_UART2_CLK	BASE_UART3_CLK	BASE_OUT_CLK	BASE_APLL_CLK	BASE_CGU_OUT0_CLK	BASE_CGU_OUT1_CLK
PLL1	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
IDIVA	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
IDIVB	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
IDIVC	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
IDIVD	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y
IDIVE	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y

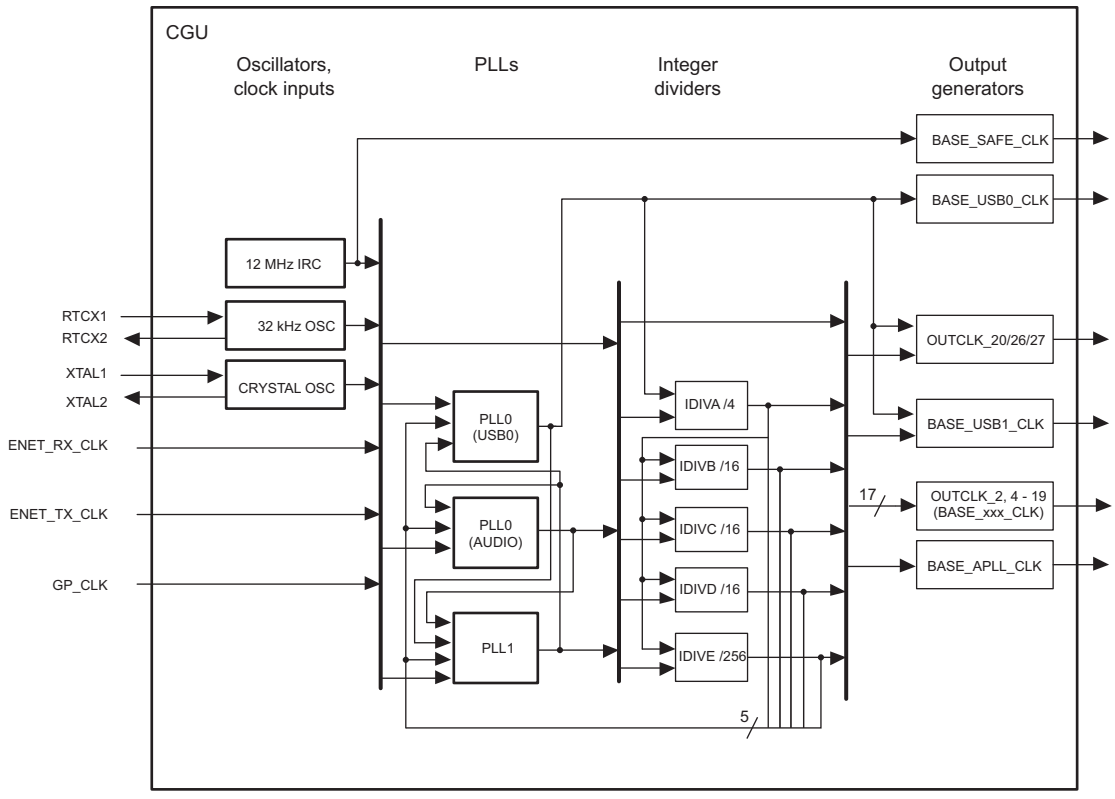


Fig 21. CGU block diagram

11.5 Pin description

Table 57. CGU pin description

Pin name/ function name	Direction	Description
XTAL1	I	Crystal oscillator input
XTAL2	O	Crystal oscillator output
RTCX1	I	RTC 32 kHz oscillator input
RTCX2	O	RTC 32 kHz oscillator output
GP_CLKIN	I	General purpose input clock
ENET_TX_CLK	I	Ethernet PHY transmit clock
ENET_RX_CLK	I	Ethernet PHY receive clock
CLKOUT	O	Clock output pin
CGU_OUT0	O	CGU spare output 0
CGU_OUT1	O	CGU spare output 1

11.6 Register description

The register addresses of the CGU are shown in Table 58.

Remark: The CGU is configured by the boot loader at reset and when waking up from Deep power-down to produce a 72 MHz clock using PLL1. Note that this configuration is not reflected in the reset values given in Table 58.

Table 58. Register overview: CGU (base address 0x4005 0000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC, UART0/ 3 boot	Reset value after USB0/1 boot	Reference
-	R	0x000	Reserved	-	-	-	-
-	R	0x004	Reserved	-	-	-	-
-	R	0x008	Reserved	-	-	-	-
-	R	0x00C	Reserved	-	-	-	-
-	-	0x010	Reserved	-			-
FREQ_MON	R/W	0x014	Frequency monitor register	0	0	0	Table 59
XTAL_OSC_CTRL	R/W	0x018	Crystal oscillator control register	0x0000 0005	0x0100 0000	0 (USB0)	Table 60
PLL0USB_STAT	R	0x01C	PLL0USB status register	0x0100 0000	0x0100 0000	0x1 (USB0)	Table 61
PLL0USB_CTRL	R/W	0x020	PLL0USB control register	0x0100 0003	0x0100 0000	0x0600 0818 (USB0)	Table 62
PLL0USB_MDIV	R/W	0x024	PLL0USB M-divider register	0x05F8 5B6A	0x0100 0000	0x0196 7FFA (USB0)	Table 63

Table 58. Register overview: CGU (base address 0x4005 0000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC, UART0/3 boot	Reset value after USB0/1 boot	Reference
PLL0USB_NP_DIV	R/W	0x028	PLL0USB N/P-divider register	0x000B 1002	0x0100 0000	0x0030 2062 (USB0)	Table 64
PLL0AUDIO_STAT	R	0x02C	PLL0AUDIO status register	0x0100 0000	0x0100 0000	0x0100 0000	Table 65
PLL0AUDIO_CTRL	R/W	0x030	PLL0AUDIO control register	0x0100 4003	0x0100 0000	0x0100 0000	Table 66
PLL0AUDIO_MDIV	R/W	0x034	PLL0AUDIO M-divider register	0x05F8 5B6A	0x0100 0000	0x0100 0000	Table 67
PLL0AUDIO_NP_DIV	R/W	0x038	PLL0AUDIO N/P-divider register	0x000B 1002	0x0100 0000	0x0100 0000	Table 68
PLLAUDIO_FRAC	R/W	0x03C	PLL0AUDIO fractional divider register	0x0020 0000	0x0100 0000	0x0100 0000	Table 69
PLL1_STAT	R	0x040	PLL1 status register	0x0100 0000	0x1	0x1 (USB1)	Table 70
PLL1_CTRL	R/W	0x044	PLL1 control register	0x0100 0003	0x0117 0880	0x0117 0880 (USB1)	Table 71
IDIVA_CTRL	R/W	0x048	Integer divider A control register	0x0100 0000	0x0100 0000	0x0100 0000	Table 72
IDIVB_CTRL	R/W	0x04C	Integer divider B control register	0x0100 0000	0x0100 0000	0x0100 0000	Table 73
IDIVC_CTRL	R/W	0x050	Integer divider C control register	0x0100 0000	0x0900 0808	0x0900 0808	Table 73
IDIVD_CTRL	R/W	0x054	Integer divider D control register	0x0100 0000	0x0100 0000	0x0100 0000	Table 73
IDIVE_CTRL	R/W	0x058	Integer divider E control register	0x0100 0000	0x0100 0000	0x0100 0000	Table 74
BASE_SAFE_CLK	R/W	0x05C	Output stage 0 control register for base clock BASE_SAFE_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 75
BASE_USB0_CLK	R/W	0x060	Output stage 1 control register for base clock BASE_USB0_CLK	0x0700 0000	0x0100 0000	0x0700 0800 (USB0)	Table 76
BASE_PERIPH_CLK	R/W	0x064	Output stage 2 control register for base clock BASE_PERIPH_CLK	0x0700 0000	<tbd>	<tbd>	Table 78
BASE_USB1_CLK	R/W	0x068	Output stage 3 control register for base clock BASE_USB1_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 78
BASE_M4_CLK	R/W	0x06C	Output stage 4 control register for base clock BASE_M4_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79

Table 58. Register overview: CGU (base address 0x4005 0000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC, UART0/3 boot	Reset value after USB0/1 boot	Reference
BASE_SPIFI_CLK	R/W	0x070	Output stage 5 control register for base clock BASE_SPIFI_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_SPI_CLK	R/W	0x074	Output stage 6 control register for base clock BASE_SPI_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_PHY_RX_CLK	R/W	0x078	Output stage 7 control register for base clock BASE_PHY_RX_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_PHY_TX_CLK	R/W	0x07C	Output stage 8 control register for base clock BASE_PHY_TX_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_APB1_CLK	R/W	0x080	Output stage 9 control register for base clock BASE_APB1_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_APB3_CLK	R/W	0x084	Output stage 10 control register for base clock BASE_APB3_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_LCD_CLK	R/W	0x088	Output stage 11 control register for base clock BASE_LCD_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_VADC_CLK	R/W	0x08C	Output stage 12 control register for base clock BASE_VADC_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_SDIO_CLK	R/W	0x090	Output stage 13 control register for base clock BASE_SDIO_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_SSP0_CLK	R/W	0x094	Output stage 14 control register for base clock BASE_SSP0_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_SSP1_CLK	R/W	0x098	Output stage 15 control register for base clock BASE_SSP1_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_UART0_CLK	R/W	0x09C	Output stage 16 control register for base clock BASE_UART0_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_UART1_CLK	R/W	0x0A0	Output stage 17 control register for base clock BASE_UART1_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_UART2_CLK	R/W	0x0A4	Output stage 18 control register for base clock BASE_UART2_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79
BASE_UART3_CLK	R/W	0x0A8	Output stage 19 control register for base clock BASE_UART3_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 79

Table 58. Register overview: CGU (base address 0x4005 0000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC, UART0/3 boot	Reset value after USB0/1 boot	Reference
BASE_OUT_CL	R/W	0x0AC	Output stage 20 control register for base clock BASE_OUT_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 80
OUTCLK_21_CTRL to OUTCLK_24_CTRL	R/W	0x0B0 to 0x0BC	Reserved output stages	-	-	-	-
BASE_APLL_CLK	R/W	0x0C0	Output stage 25 control register for base clock BASE_APLL_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 81
BASE_CGU_OUT0_CLK	R/W	0x0C4	Output stage 26 control register for base clock BASE_CGU_OUT0_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 82
BASE_CGU_OUT1_CLK	R/W	0x0C8	Output stage 27 control register for base clock BASE_CGU_OUT1_CLK	0x0100 0000	0x0100 0000	0x0100 0000	Table 82

11.6.1 Frequency monitor register

The CGU can report the relative frequency of any operating clock. The clock to be measured must be selected by software, while the fixed-frequency IRC clock fref is used as the reference frequency. A 14-bit counter then counts the number of cycles of the measured clock that occur during a user-defined number of reference-clock cycles. When the MEAS bit is set, the measured-clock counter is reset to 0 and counts up, while the 9-bit reference-clock counter is loaded with the value in RCNT and then counts down towards 0. When either counter reaches its terminal value both counters are disabled and the MEAS bit is reset to 0. The current values of the counters can then be read out and the selected frequency obtained by the following equation:

$$fselected = \frac{FCNT}{RCNT} \times fref$$

Note that the accuracy of this measurement can be affected by several factors:

1. Quantization error is noticeable if the ratio between the two clocks is large (e.g. 100 kHz vs. 1 kHz), because one counter saturates while the other still has only a small count value.
2. Due to synchronization, the counters are not started and stopped at exactly the same time.
3. The measured frequency can only be to the same level of precision as the reference frequency.

Table 59. FREQ_MON register (FREQ_MON, address 0x4005 0014) bit description

Bit	Symbol	Value	Description	Reset value	Access
8:0	RCNT		9-bit reference clock-counter value	0	R/W
22:9	FCNT		14-bit selected clock-counter value	0	R
23	MEAS		Measure frequency	0	R/W
		0	RCNT and FCNT disabled		
		1	Frequency counters started		
28:24	CLK_SEL		Clock-source selection for the clock to be measured. All other values are reserved.	0	R/W
		0x00	32 kHz oscillator (default)		
		0x01	IRC		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x05	Reserved		
		0x06	Crystal oscillator		
		0x07	PLL0USB		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0A	Reserved		
		0x0B	Reserved		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.2 Crystal oscillator control register

The register XTAL_OSC_CONTROL contains the control bits for the crystal oscillator.

Table 60. XTAL_OSC_CTRL register (XTAL_OSC_CTRL, address 0x4005 0018) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ENABLE		Oscillator-pad enable. Do not change the BYPASS and ENABLE bits in one write-action: this will result in unstable device operation!	1	R/W
		0	Enable		
		1	Power-down (default)		

Table 60. XTAL_OSC_CTRL register (XTAL_OSC_CTRL, address 0x4005 0018) bit description

Bit	Symbol	Value	Description	Reset value	Access
1	BYPASS		Configure crystal operation or external-clock input pin XTAL1. Do not change the BYPASS and ENABLE bits in one write-action: this will result in unstable device operation!	0	R/W
		0	Operation with crystal connected (default).		
		1	Bypass mode. Use this mode when an external clock source is used instead of a crystal.		
2	HF		Select frequency range	1	R/W
		0	Oscillator low-frequency mode (crystal or external clock source 1 to 20 MHz). Between 15 MHz and 20 MHz, the state of the HF bit is don't care.		
		1	Oscillator high-frequency mode; crystal or external clock source 15 to 25 MHz. Between 15 MHz and 20 MHz, the state of the HF bit is don't care.		
31:3	-		Reserved	-	-

11.6.3 PLL0USB registers

The PLL0 provides a dedicated clock to the High-speed USB0 interface.

See Section 11.7.4.5 for instructions on how to set up the PLL0.

11.6.3.1 PLL0USB status register

Table 61. PLL0USB status register (PLL0USB_STAT, address 0x4005 001C) bit description

Bit	Symbol	Description	Reset value	Access
0	LOCK	PLL0 lock indicator	0	R
1	FR	PLL0 free running indicator	0	R
31:2	-	Reserved		-

11.6.3.2 PLL0USB control register

Table 62. PLL0USB control register (PLL0USB_CTRL, address 0x4005 0020) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		PLL0 power down	1	R/W
		0	PLL0 enabled		
		1	PLL0 powered down		
1	BYPASS		Input clock bypass control	1	R/W
		0	CCO clock sent to post-dividers. Use this in normal operation.		
		1	PLL0 input clock sent to post-dividers (default).		
2	DIRECTI		PLL0 direct input	0	R/W

Table 62. PLL0USB control register (PLL0USB_CTRL, address 0x4005 0020) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
3	DIRECTO		PLL0 direct output	0	R/W
4	CLKEN		PLL0 clock enable	0	R/W
5	-		Reserved	-	-
6	FRM		Free running mode	0	R/W
7	-		Reserved	0	R/W
8	-		Reserved. Reads as zero. Do not write one to this register.	0	R/W
9	-		Reserved. Reads as zero. Do not write one to this register.	0	R/W
10	-		Reserved. Reads as zero. Do not write one to this register.	0	R/W
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.3.3 PLL0USB M-divider register

Remark: The PLL M-divider register does not use the direct binary representations of M directly. Instead, it uses an encoded version MDEC of M. The valid range for M is 1 to 2^15. This value is encoded into a 17-bit MDEC value.

The relationship between M and MDEC is expressed via the following pseudo-code. For specific examples see Section 11.8.3 and Section 11.8.4.

```
M_max=0x00008000, x=0x00004000;
    switch (M) {
        case 0: x = 0xFFFFFFFF;
```

```
        case 1: x = 0x00018003;
        case 2: x = 0x00010003;

        default: for (i = M; i <= M_max; i++)
                    x = (((x ^ (x>>1)) & 1) << 14) | ((x>>1) & 0x3FFF); }
    MDEC[16:0] = x;
```

The values for SELP, SELI, and SELR depend on the value for M as expressed by the following pseudo-code:

```
if (M < 60) then
      SELP = (M>>1) + 1
   else
      SELP = 31;
if (M > 16384) then
      SELI = 1
   else if (M > 8192) then
      SELI = 2
   else if (M > 2048) then
      SELI = 4
   else if (M >= 501) then
      SELI = 8
   else if (M >=60) then
      SELI = 4*(1024/(M+9))
   else
      SELI = (M & 0x3C) + 4; /* & denotes bitwise AND */
   SELR = 0;
```

Table 63. PLL0USB M-divider register (PLL0USB_MDIV, address 0x4005 0024) bit description

Bit	Symbol	Description	Reset value	Access
16:0	MDEC	Decoded M-divider coefficient value. Select values for the M-divider between 1 and 131071.	0x05F8 5B6A	R/W
21:17	SELP	Bandwidth select P value	11100	R/W
27:22	SELI	Bandwidth select I value	010111	R/W
31:28	SELR	Bandwidth select R value	0000	R/W

11.6.3.4 PLL0USB NP-divider register

Remark: The PLL NP-divider register does not use the direct binary representations of N and P directly. Instead, it uses encoded versions NDEC and PDEC of N and P respectively.

- The valid range for N is from 1 to 2^8. This value is encoded into a 10-bit NDEC value. The relationship can be expressed through the following pseudo-code:

```
N_max=0x00000100, x=0x00000080;
   switch (N) {
      case 0: x = 0xFFFFFFFF;
      case 1: x = 0x00000302;
      case 2: x = 0x00000202;
```

```
        default: for (i = N; i <= N_max; i++)
                    x = (((x ^ (x>>2) ^ (x>>3) ^ (x>>4)) & 1) << 7) | ((x>>1) & 0x7F); }
    NENC[9:0] = x;
```

- The valid range for P is from 1 to 2^5. This value is encoded into a 7-bit PDEC value. The relationship can be expressed through the following pseudo-code:

```
P_max=0x00000200, x=0x00000010;
    switch (P) {
        case 0: x = 0xFFFFFFFF;
        case 1: x = 0x00000062;
        case 2: x = 0x00000042;

        default: for (i = P; i <= P_max; i++)
                    x = (((x ^ (x>>2)) & 1) << 4) | ((x>>1) & 0xF); }
    PDEC[6:0] = x;
```

For specific examples see Section 11.8.3 and Section 11.8.4.

Table 64. PLL0USB NP-divider register (PLL0USB_NP_DIV, address 0x4005 0028) bit description

Bit	Symbol	Description	Reset value	Access
6:0	PDEC	Decoded P-divider coefficient value	000 0010	R/W
11:7	-	Reserved	-	-
21:12	NDEC	Decoded N-divider coefficient value	1011 0001	R/W
31:22	-	Reserved	-	-

11.6.4 PLL0AUDIO registers

The PLL0AUDIO provides a wide range of frequencies for audio applications and can be connected to multiple base clocks. The PLL0AUDIO can be used with or without a fractional divider.

See Section 11.7.4.5 for instructions on how to set up the PLL0.

11.6.4.1 PLL0AUDIO status register

Table 65. PLL0AUDIO status register (PLL0AUDIO_STAT, address 0x4005 002C) bit description

Bit	Symbol	Description	Reset value	Access
0	LOCK	PLL0 lock indicator	0	R
1	FR	PLL0 free running indicator	0	R
31:2	-	Reserved		-

11.6.4.2 PLL0AUDIO control register

Table 66. PLL0AUDIO control register (PLL0AUDIO_CTRL, address 0x4005 0030) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		PLL0 power down	1	R/W
		0	PLL0 enabled		
		1	PLL0 powered down		
1	BYPASS		Input clock bypass control	1	R/W
		0	CCO clock sent to post-dividers. Use this in normal operation.		
		1	PLL0 input clock sent to post-dividers (default).		
2	DIRECTI		PLL0 direct input	0	R/W
3	DIRECTO		PLL0 direct output	0	R/W
4	CLKEN		PLL0 clock enable	0	R/W
5	-		Reserved	-	-
6	FRM		Free running mode	0	R/W
7	-		Reserved	0	R/W
8	-		Reserved. Reads as zero. Do not write one to this register.	0	R/W
9	-		Reserved. Reads as zero. Do not write one to this register.	0	R/W
10	-		Reserved. Reads as zero. Do not write one to this register.	0	R/W
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
12	PLLFRAQ_REQ		Fractional PLL word write request. Set this bit to 1 if the fractional divider is enabled in the SEL_EXT bit.	0	R/W
13	SEL_EXT		Select fractional divider.	0	R/W
		0	Enable fractional divider.		
		1	MDEC enabled. Fractional divider not used.		
14	MOD_PD		Sigma-Delta modulator power-down	1	R/W
		0	Sigma-Delta modulator enabled		
		1	Sigma-Delta modulator powered down		
23:15	-		Reserved	-	-

Table 66. PLL0AUDIO control register (PLL0AUDIO_CTRL, address 0x4005 0030) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.4.3 PLL0AUDIO M-divider register

The PLL0AUDIO M-divider register can be set directly if the PLL0AUDIO is not used in fractional mode. If the fractional divider is enabled (see Table 69), then the fractional divider sets the MDEC value.

Remark: The PLL M-divider register does not use the direct binary representations of M directly. Instead, it uses an encoded version MDEC of M. The valid range for M is 1 to 2^15. This value is encoded into a 17-bit MDEC value.

The relationship between M and MDEC is expressed via the following pseudo-code. For specific examples see Section 11.8.3 and Section 11.8.4.

```
M_max=0x00008000, x=0x00004000;
    switch (M) {
        case 0: x = 0xFFFFFFFF;
        case 1: x = 0x00018003;
        case 2: x = 0x00010003;

        default: for (i = M; i <= M_max; i++)
                    x = (((x ^ (x>>1)) & 1) << 14) | ((x>>1) & 0x3FFF); }
    MDEC[16:0] = x;
```

Remark: The values for SELP, SELI, and SELR are generated by the encoding block and need not be programmed explicitly.

Table 67. PLL0AUDIO M-divider register (PLL0AUDIO_MDIV, address 0x4005 0034) bit description

Bit	Symbol	Description	Reset value	Access
16:0	MDEC	Decoded M-divider coefficient value. Select values for the M-divider between 1 and 131071.	0x05F8 5B6A	R/W
31:17	-	Reserved	-	-

11.6.4.4 PLL0AUDIO NP-divider register

Remark: The PLL NP-divider register does not use the direct binary representations of N and P directly. Instead, it uses encoded versions NDEC and PDEC of N and P respectively.

- The valid range for N is 1 to 2^8. This value is encoded into a 10-bit NDEC value. The relationship can be expressed through the following pseudo-code:

```
N_max=0x00000100, x=0x00000080;
    switch (N) {
        case 0: x = 0xFFFFFFFF;
        case 1: x = 0x00000302;
        case 2: x = 0x00000202;

        default: for (i = N; i <= N_max; i++)
                    x = (((x ^ (x>>2) ^ (x>>3) ^ (x>>4)) & 1) << 7) | ((x>>1) & 0x7F); }
     NENC[9:0] = x;
```

- The valid range for P is from 1 to 2^5. This value is encoded into a 7-bit PDEC value. The relationship can be expressed through the following pseudo-code:

```
P_max=0x00000200, x=0x00000010;
    switch (P) {
        case 0: x = 0xFFFFFFFF;
        case 1: x = 0x00000062;
        case 2: x = 0x00000042;

        default: for (i = P; i <= P_max; i++)
                    x = (((x ^ (x>>2)) & 1) << 4) | ((x>>1) & 0xF); }
    PDEC[6:0] = x;
```

Table 68. PLL0 AUDIO NP-divider register (PLL0AUDIO_NP_DIV, address 0x4005 0038) bit description

Bit	Symbol	Description	Reset value	Access
6:0	PDEC	Decoded P-divider coefficient value	000 0010	R/W
11:7	-	Reserved	-	-
21:12	NDEC	Decoded N-divider coefficient value	1011 0001	R/W
31:22	-	Reserved	-	-

11.6.4.5 PLL0AUDIO fractional divider register

When the fractional divider is active, the sigma-delta modulator block generates divider values M and M+1 in the correct proportion so that an average division ratio of M+K/L is realized where 0<=K<=L and M, K, and L are integer values. M Is determined by the integer part of the PLLFRACT_CTRL register (PLLFRACT[21:15]) and K is determined by the fractional part of the PLLFRACT_CTRL register (PLLFRACT[14:0]). Consecutive M and M+1 values are then further encoded into appropriate MDEC values before being presented as input to the M-divider.

Table 69. PLL0AUDIO fractional divider register (PLL0AUDIO_FRAC, address 0x4005 003C) bit description

Bit	Symbol	Description	Reset value	Access
21:0	PLLFRACT_CTRL	PLL fractional divider control word	000 0000	R/W
31:22	-	Reserved	-	-

11.6.5 PLL1 registers

The PLL1 is used for the core and all peripheral blocks.

11.6.5.1 PLL1 status register

Table 70. PLL1 status register (PLL1_STAT, address 0x4005 0040) bit description

Bit	Symbol	Description	Reset value	Access
0	LOCK	PLL1 lock indicator	0	R
31:1	-	Reserved	-	-

11.6.5.2 PLL1 control register

Table 71. PLL1_CTRL register (PLL1_CTRL, address 0x4005 0044) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		PLL1 power down	1	R/W
		0	PLL1 enabled		
		1	PLL1 powered down		
1	BYPASS		Input clock bypass control	1	R/W
		0	CCO clock sent to post-dividers. Use for normal operation.		
		1	PLL1 input clock sent to post-dividers (default).		
2	-		Reserved. Do not write one to this bit.	0	R/W
5:3	-		Reserved. Do not write one to these bits.	-	-

Table 71. PLL1_CTRL register (PLL1_CTRL, address 0x4005 0044) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
6	FBSEL		PLL feedback select (see Figure 24 “PLL1 block diagram”).	0	R/W
		0	CCO output is used as feedback divider input clock.		
		1	PLL output clock (clkout) is used as feedback divider input clock. Use for normal operation.		
7	DIRECT		PLL direct CCO output	0	R/W
		0	Disabled		
		1	Enabled		
9:8	PSEL[		Post-divider division ratio P. The value applied is 2xP.	01	R/W
		0x0	1		
		0x1	2 (default)		
		0x2	4		
		0x3	8		
10	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
13:12	NSEL		Pre-divider division ratio	10	R/W
		0x0	1		
		0x1	2		
		0x2	3 (default)		
		0x3	4		
15:14	-		Reserved	-	-
23:16	MSEL		Feedback-divider division ratio (M) 00000000 = 1 00000001 = 2 ... 11111111 = 256	11000	R/W

Table 71. PLL1_CTRL register (PLL1_CTRL, address 0x4005 0044) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock-source selection.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x05	Reserved		
		0x06	crystal oscillator		
		0x07	PLL0USB		
		0x08	PLL0AUDIO		
		0x09	Reserved		
		0x0A	Reserved		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.6 Integer divider register A

Table 72. IDIVA control register (IDIVA_CTRL, address 0x4005 0048) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Integer divider A power down	0	R/W
		0	IDIVA enabled (default)		
		1	power-down		
1	-		Reserved	-	-
3:2	IDIV		Integer divider A divider values (1/(IDIV + 1))	00	R/W
		0x0	1 (default)		
		0x1	2		
		0x2	3		
		0x3	4		
10:4	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 72. IDIVA control register (IDIVA_CTRL, address 0x4005 0048) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x07	PLL0USB		
		0x08	PLL0AUDIO		
		0x09	PLL1		
31:29	-		Reserved	-	-

11.6.7 Integer divider register B, C, D

Table 73. IDIVB/C/D control registers (IDIVB_CTRL, address 0x4005 004C; IDIVC_CTRL, address 0x4005 0050; IDIVC_CTRL, address 0x4005 0054) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Integer divider power down	0	R/W
		0	IDIV enabled (default)		
		1	power-down		
1	-		Reserved	-	-
5:2	IDIV		Integer divider B, C, D divider values (1/(IDIV + 1)) 0000 = 1 (default) 0001 = 2 ... 1111 = 16	0000	R/W
10:6	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 73. IDIVB/C/D control registers (IDIVB_CTRL, address 0x4005 004C; IDIVC_CTRL, address 0x4005 0050; IDIVC_CTRL, address 0x4005 0054) bit description

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock-source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
31:29	-		Reserved	-	-

11.6.8 Integer divider register E

Table 74. IDIVE control register (IDIVE_CTRL, address 0x4005 0058) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Integer divider power down	0	R/W
		0	IDIV enabled (default)		
		1	power-down		
1	-		Reserved	-	-
9:2	IDIV		Integer divider E divider values (1/(IDIV + 1)) 00000000 = 1 (default) 00000001 = 2 ... 11111111 = 256	00000000	R/W
10	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 74. IDIVE control register (IDIVE_CTRL, address 0x4005 0058) bit description

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock-source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
31:29	-		Reserved	-	-

11.6.9 BASE_SAFE_CLK control register

This register controls the BASE_SAFE_CLK to the watchdog oscillator. The only possible clock source for this base clock is the IRC.

Table 75. BASE_SAFE_CLK control register (BASE_SAFE_CLK, address 0x4005 005C) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x01	IRC (default)		
31:29	-		Reserved	-	-

11.6.10 BASE_USB0_CLK control register

This register controls the BASE_USB0_CLK to the High-speed USB0. The only possible clock source for this base clock is the PLL0USB output.

Table 76. BASE_USB0_CLK control register (BASE_USB0_CLK, address 0x4005 0060) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-
28:24	CLK_SEL		Clock-source selection.	0x07	R/W
		0x07	PLL0USB (default)		
31:29	-		Reserved	-	-

11.6.11 BASE_PERIPH_CLK control register

This register controls base clock 2 to the SGPIO block.

Table 77. BASE_PERIPH_CLK control register (BASE_PERIPH_CLK, address 0x4005 0064) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 77. BASE_PERIPH_CLK control register (BASE_PERIPH_CLK, address 0x4005 0064) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.12 BASE_USB1_CLK control register

These registers control base clocks 3 (USB1).

Table 78. BASE_USB1_CLK control register BASE_USB1_CLK, address 0x4005 0068) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 78. BASE_USB1_CLK control register BASE_USB1_CLK, address 0x4005 0068) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x07	PLL0USB		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.13 BASE_M4_CLK to BASE_UART3_CLKcontrol registers

These registers control base clocks 4 to 19.

Table 79. BASE_M4_CLK to BASE_UART3_CLK control registers (BASE_M4_CLK to BASE_UART3_CLK, address 0x4005 006C to 0x4005 00A8) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 79. BASE_M4_CLK to BASE_UART3_CLK control registers (BASE_M4_CLK to BASE_UART3_CLK, address 0x4005 006C to 0x4005 00A8) bit description

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock source selection. All other values are reserved.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x06	Crystal oscillator		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.14 BASE_OUT_CLK register

This register controls the clock output to the CLKOUT pin. All clock generator outputs can be monitored through this pin.

Table 80. BASE_OUT_CLK control register BASE_OUT_CLK, addresses 0x4005 00AC) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 80. BASE_OUT_CLK control register BASE_OUT_CLK, addresses 0x4005 00AC) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock-source selection.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x05	Reserved		
		0x06	Crystal oscillator		
		0x07	PLL0USB		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.15 BASE_APLL_CLK register

This register controls the clock output to the <tbd>.

Table 81. BASE_APLL_CLK control register (BASE_APLL_CLK, addresses 0x4005 00C0) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 81. BASE_APLL_CLK control register (BASE_APLL_CLK, addresses 0x4005 00C0) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock-source selection.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x05	Reserved		
		0x06	Crystal oscillator		
		0x07	Reserved		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.6.16 BASE_CGU_OUT0_CLK to BASE_CGU_OUT1_CLK register

This register controls the clock output to the spare CGU outputs pins CGU_OUT0 and CGU_OUT1. All clock generator outputs can be monitored through this pin.

Table 82. BASE_CGU_OUT0_CLK to BASE_CGU_OUT1_CLK control register (BASE_CGU_OUT0_CLK to BASE_CGU_OUT1_CLK , addresses 0x4005 00C4 to 0x4005 00C8) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Output stage power down	0	R/W
		0	Output stage enabled (default)		
		1	power-down		
10:1	-		Reserved	-	-
11	AUTOBLOCK		Block clock automatically during frequency change	0	R/W
		0	Autoblocking disabled		
		1	Autoblocking enabled		
23:12	-		Reserved	-	-

Table 82. BASE_CGU_OUT0_CLK to BASE_CGU_OUT1_CLK control register (BASE_CGU_OUT0_CLK to BASE_CGU_OUT1_CLK , addresses 0x4005 00C4 to 0x4005 00C8) bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
28:24	CLK_SEL		Clock-source selection.	0x01	R/W
		0x00	32 kHz oscillator		
		0x01	IRC (default)		
		0x02	ENET_RX_CLK		
		0x03	ENET_TX_CLK		
		0x04	GP_CLKIN		
		0x05	Reserved		
		0x06	Crystal oscillator		
		0x07	PLL0USB		
		0x08	PLL0AUDIO		
		0x09	PLL1		
		0x0C	IDIVA		
		0x0D	IDIVB		
		0x0E	IDIVC		
		0x0F	IDIVD		
		0x10	IDIVE		
31:29	-		Reserved	-	-

11.7 Functional description

11.7.1 32 kHz oscillator

The 32 kHz oscillator output is controlled by the CREG block (see Table 38). The RTC and the Alarm timer are connected directly to the 32 kHz oscillator.

11.7.2 IRC

The IRC is a trimmed 12 MHz internal oscillator. Although it's part of the CGU, the CGU has no control over this clock source. The IRC is put into power down depending on the power saving mode.

11.7.3 Crystal oscillator

The crystal oscillator is controlled by the XTAL_OSC_CTRL register in the CGU (see Table 60).

11.7.4 PLL0 (PLL0USB and PLL0AUDIO)

The PLL blocks of the PLL0USB and PLL0AUDIO are identical. The PLL0AUDIO supports an additional fractional divider to obtain more PLL frequencies with higher accuracy for audio applications.

11.7.4.1 Features

- Input frequency: 14 kHz to 150 MHz. The input from an external crystal is limited to 25 MHz.
- CCO frequency: 275 MHz to 550 MHz.
- Output clock range: 4.3 MHz to 550 MHz.
- Programmable dividers:
  - Pre-divider N (N, 1 to 2^8)
  - Feedback-divider 2 x M (M, 1 to 2^{15})
  - Post-divider P x 2 (P, 1 to 2^5).
- Programmable bandwidth (integrating action, proportional action, high frequency pole).
- On-the-fly adjustment of the clock possible (dividers with handshake control).
- Positive edge clocking.
- Frequency limiter to avoid hang-up of the PLL.
- Lock detector.
- Power-down mode.
- Free running mode

Remark: Both PLL0 blocks are functionally identical. The PLL0 for audio applications (PLL0 for audio) supports an additional fractional divider stage (see Section 11.7.5).

11.7.4.2 PLL0 description

The block diagram of the PLL is shown in Figure 22. The clock input has to be fed to pin clkin. Pin clkout is the PLL clock output. The analog part of the PLL consists of a Phase Frequency Detector (PFD), filter and a Current Controlled Oscillator (CCO). The PFD has two inputs, a reference input from the (divided) external clock and one input from the divided CCO output clock. The PFD compares the phase/frequency of these input signals and generates a control signal if they don't match. This control signal is fed to a filter which drives the CCO.

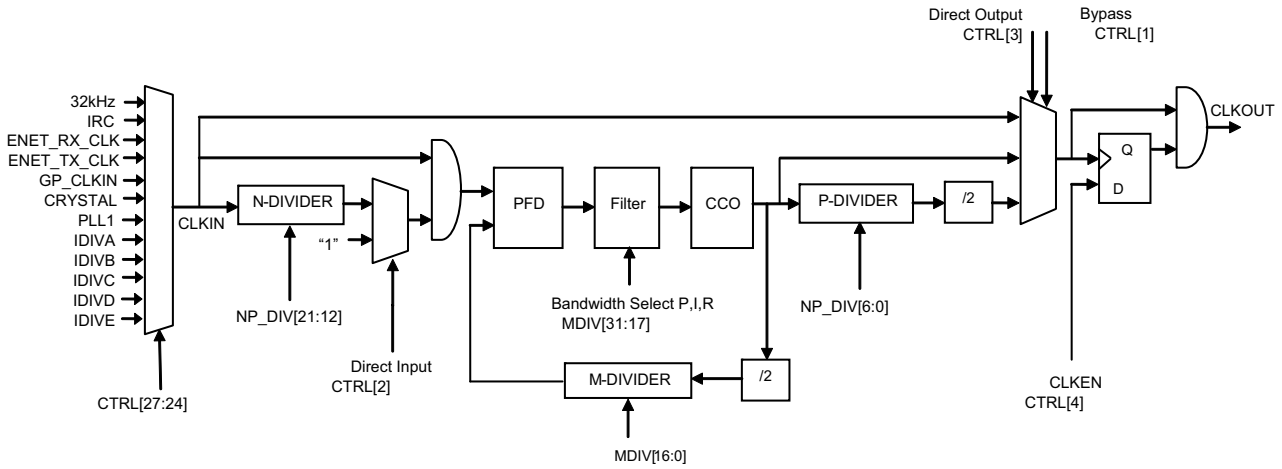


Fig 22. PLL0 block diagram

The PLL contains three programmable dividers: pre-divider (N), feedback-divider (M) and post-divider (P). The PLL contains a lock detector which measures the phase difference between the rising edges of the input and feedback clocks. Only when this difference is smaller than the so called "lock criterion" for more than seven consecutive input clock periods, the lock output switches from low to high. A single too large phase difference immediately resets the counter and causes the lock signal to drop (if it was high). Requiring seven phase measurements in a row to be below a certain figure ensures that the lock detector will not indicate lock until both the phase and frequency of the input and feedback clocks are very well aligned. This effectively prevents false lock indications, and thus ensures a glitch free lock signal.

To avoid frequency hang-up the PLL contains a frequency limiter. This feature is built in to prevent the CCO from running too fast, this can occur if e.g. a wrong feedback-divider (M) ratio is applied to the PLL.

11.7.4.3 Use of PLL0 operating modes

Table 83. PLL operating modes

	PLL0_Mode bit settings:					
Mode	**PD**	**CLKEN**	**BYPASS**	**DIRECTI**	**DIRECTO**	**FRM**
1: Normal	0	1	0	1/0	1/0	0
3: Power Down	1	x	x	x	x	x

11.7.4.3.1 Normal Mode

Mode 1 is the normal operating mode.

The pre- and post-divider can be selected to give:

- mode 1a: Normal operating mode without post-divider and without pre-divider
- mode 1b: Normal operating mode with post-divider and without pre-divider

- mode 1c: Normal operating mode without post-divider and with pre-divider
- mode 1d: Normal operating mode with post-divider and with pre-divider

To get at the output of the PLL (clkout) the best phase-noise and jitter performance, the highest possible reference clock (clkref) at the PFD has to be used. Therefore mode 1a and 1b are recommended, when it is possible to make the right output frequency without pre-divider.

By using the post-divider the clock at the output of the PLL (clkout) the divider ratio is always even because the divide-by-2 divider after the post-divider.

Table 84. DIRECTL and DIRECTO bit settings in HP0/1_Mode register

Mode	DIRECTI	DIRECTO
1a	1	1
1b	1	0
1c	0	1
1d	0	0

11.7.4.3.2 Mode 1a: Normal operating mode without post-divider and without pre-divider

In normal operating mode 1a the post-divider and pre-divider are bypassed. The operating frequencies are:

Fout = Fcco = 2 x M x Fin Λ (275 MHz $\leq$Fcco $\leq$550 MHz, 4 kHz $\leq$ Fin $\leq$150 MHz)

The feedback divider ratio is programmable:

- Feedback-divider M (M, 1 to 2^{15})

11.7.4.3.3 Mode 1b: Normal operating mode with post-divider and without pre-divider

In normal operating mode 1b the pre-divider is bypassed. The operating frequencies are:

Fout = Fcco /(2 x P) = (M / P) x Fin Λ (275 MHz $\leq$Fcco $\leq$550 MHz, 4 kHz $\leq$Fin $\leq$ 150 MHz)

The divider ratios are programmable:

- Feedback-divider M (M, 1 to 2^{15})
- Post-divider P (P, 1 to 32)

11.7.4.3.4 Mode 1c: Normal operating mode without post-divider and with pre-divider

In normal operating mode 1c the post-divider with divide-by-2 divider is bypassed. The operating frequencies are:

Fout = Fcco = 2 x M x Fin / N Λ (275 MHz $\leq$Fcco $\leq$550 MHz, 4 kHz $\leq$Fin/N $\leq$150 MHz)

The divider ratios are programmable:

- Pre-divider N (N, 1 to 256)
- Feedback-divider M (M, 1 to 2^{15})

11.7.4.3.5 Mode 1d: Normal operating mode with post-divider and with pre-divider

In normal operating mode 1d none of the dividers are bypassed. The operating frequencies are:

Fout = Fcco /(2 x P) = M x Fin /(N x P) Λ (275 MHz ≤Fcco ≤550 MHz, 4 kHz ≤Fin/N ≤150 MHz)

The divider ratios are programmable:

- Pre-divider N (N, 1 to 256)
- Feedback-divider M (M, 1 to 2^{15})
- Post-divider P (P, 1 to 32)

11.7.4.3.6 Mode 3: Power down mode (pd)

In this mode (pd = '1'), the oscillator will be stopped, the lock output will be made low, and the internal current reference will be turned off. During pd it is also possible to load new divider ratios at the input buses (msel, psel, nsel). Power-down mode is ended by making pd low, causing the PLL to start up. The lock signal will be made high once the PLL has regained lock on the input clock.

11.7.4.4 Settings for USB0

Table 85 shows the divider settings used for configuring an output frequency F_{out} of 480 MHz for USB0.

11.7.4.5 Usage notes

In order to set up the PLL0, follow these steps:

1. Power down the PLL0 by setting bit 1 in the PLL0_CTRL register to 1. This step is only needed if the PLL0 is currently enabled.
2. Configure the PLL0 m, n, and p divider values in the PLL0_M and PLL0_NP registers.
3. Power up the PLL0 by setting bit 1 in the PLL0_CTRL register to 0.
4. Wait for the PLL0 to lock by monitoring the LOCK bit in the PLL0_STAT register.
5. Enable the PLL0 clock output in the PLL0_CTRL register.

11.7.5 Fractional divider for PLL0AUDIO

The PLL0 for audio applications (PLL0AUDIO) includes an additional fractional divider. The SEL_EXT bit in the PLL0AUDIO control register determines whether the fractional divider is used (SEL_EXT=0) or bypassed (SEL_EXT=1). In the latter case, PLL0AUDIO operates exactly as PLL0USB and the MDEC value is used directly to control the feedback divider.

When the fractional divider is active, the sigma-delta modulator block generates divider values M and M+1 in the correct proportion so that an average division ratio of M+K/L is realized where 0<=K<=L and M, K, and L are integer values. M Is determined by the integer part of the PLLFRACT_CTRL register (PLLFRACT[21:15]) and K is determined by the fractional part of the PLLFRACT_CTRL register (PLLFRACT[14:0]). Consecutive M and M+1 values are then further encoded into appropriate MENC values before being presented as input to the M-divider.

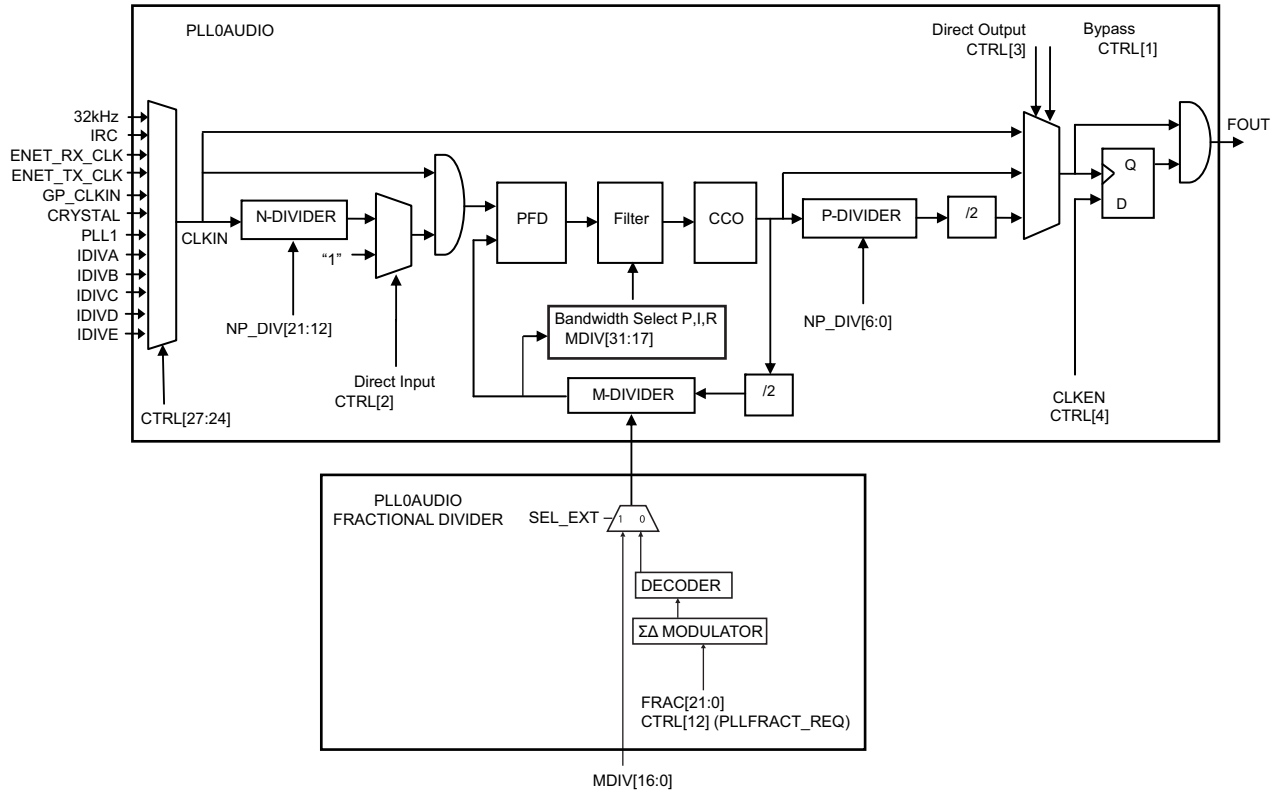


Fig 23. PLL0 with fractional divider

11.7.6 PLL1

11.7.6.1 Features

- 1 MHz to 50 MHz input frequency. The input from an external crystal is limited to 25 MHz.
- 9.75 MHz to 320 MHz selectable output frequency with 50% duty cycle.
- 156 MHz to 320 MHz Current Controlled Oscillator (CCO) frequency.
- Power-down mode.
- Lock detector.

11.7.6.2 PLL1 description

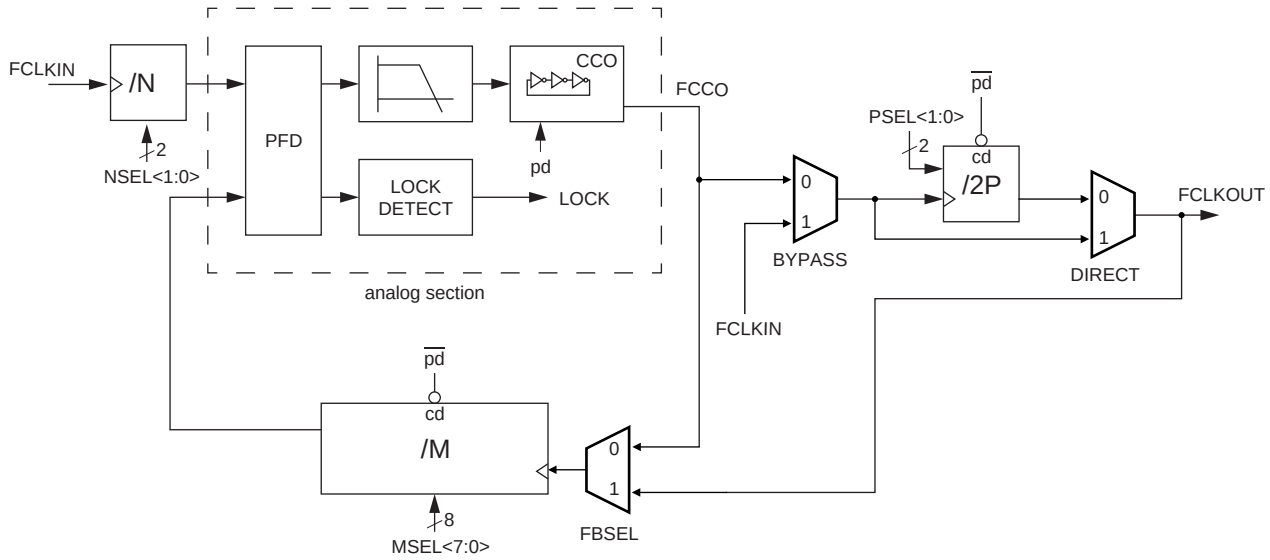


Fig 24. PLL1 block diagram

The block diagram of this PLL is shown in Figure 24. The input frequency range is 10 MHz to 25 MHz. The input clock is fed directly to the Phase-Frequency Detector (PFD). This block compares the phase and frequency of its inputs, and generates a control signal when phase and/ or frequency do not match. The loop filter filters these control signals and drives the current controlled oscillator (CCO), which generates the main clock. The CCO frequency range is 156 MHz to 320 MHz.These clocks are either divided by 2xP by the programmable post divider to create the output clocks, or are sent directly to the outputs. The main output clock is then divided by M by the programmable feedback divider to generate the feedback clock. The output signal of the phase-frequency detector is also monitored by the lock detector, to signal when the PLL has locked on to the input clock.

11.7.6.3 Lock detector

The lock detector measures the phase difference between the rising edges of the input and feedback clocks. Only when this difference is smaller than the so called “lock criterion” for more than eight consecutive input clock periods, the lock output switches from low to high. A single too large phase difference immediately resets the counter and causes the lock signal to drop (if it was high). Requiring eight phase measurements in a row to be below a certain figure ensures that the lock detector will not indicate lock until both the phase and frequency of the input and feedback clocks are very well aligned. This effectively prevents false lock indications, and thus ensures a glitch free lock signal.

11.7.6.4 Power-down control

To reduce the power consumption when the PLL clock is not needed, a Power-down mode has been incorporated. In this mode, the internal current reference will be turned off, the oscillator and the phase-frequency detector will be stopped and the dividers will enter a reset state. While in Power-down mode, the lock output will be low to indicate that

the PLL is not in lock. When the Power-down mode is terminated, the PLL will resume its normal operation and will make the lock signal high once it has regained lock on the input clock.

11.7.6.5 Selectable feedback divider clock

To allow a trade-off to be made between functionality and power consumption, the feedback divider can be connected to either the CCO clock by setting FBSEL to 0 or to the output clock by setting FBSEL to 1. If the post-divider is used to divide down the CCO clock the current consumption of the feedback divider can be reduced by making it run on the lower output clock instead of the CCO clock, but doing so will limit the relation between output and phase detector clock frequencies to integer values.

11.7.6.6 Direct output mode

In normal operating mode (with DIRECT set to 0) the CCO clock is divided by 2, 4, 8 or 16 depending on the value of PSEL[1:0], automatically giving an output clock with a 50% duty cycle. If a higher output frequency is needed, the CCO clock can be sent directly to the output by setting DIRECT to 1. Since the CCO was designed to directly generate a clock with a 50% duty cycle, the output clock duty cycle will also be 50% in direct mode.

11.7.6.7 Divider ratio programming

Pre-divider

The pre-divider's division ratio is controlled by the NSEL[1:0] input. The division ratio between PLL's input clock and the phase detector clock is the decimal value on NSEL[1:0] plus one.

Post-divider

The division ratio of the post divider is controlled by the PSEL bits. The division ratio is two times the value of P selected by PSEL bits. This guarantees an output clock with a 50% duty cycle.

Feedback divider

The feedback divider's division ratio is controlled by the MSEL bits. The division ratio between the PLL's output clock and the input clock is the decimal value on MSEL bits plus one.

Changing the divider values

Changing the divider ratio while the PLL is running is not recommended. As there is no way to synchronize the change of the NSEL, MSEL, and PSEL values with the dividers, the risk exists that the counter will read in an undefined value, which could lead to unwanted spikes or drops in the frequency of the output clock. The recommended way of changing between divider settings is to power down the PLL, adjust the divider settings and then let the PLL start up again.

11.7.6.8 Frequency selection

The PLL frequency equations use the following parameters (also see Figure 24):

Integer mode

In this mode the post divider is enabled and the feedback divider is set to run on the PLL output clock, giving the following frequency relations:

(1)

$$FCLKOUT = M \times \frac{FCLKIN}{N}$$

(2)

$$FCCO = 2 \times P \times FCLKOUT = 2 \times P \times M \times \frac{FCLKIN}{N}$$

Non-integer mode

In this mode the post-divider is enabled and the feedback divider is set to run directly on the CCO clock, which gives the following frequency dividers:

(3)

$$FCLKOUT = \frac{FCCO}{2 \times P} = \frac{M}{2 \times P} \times \frac{FCLKIN}{N}$$

(4)

$$FCCO = M \times \frac{FCLKIN}{N}$$

Direct mode

In this mode, the post-divider is disabled and the CCO clock is sent directly to the output, leading to the following frequency equation:

(5)

$$FCLKOUT = FCCO = M \times \frac{FCLKIN}{N}$$

Power-down mode

In this mode, the internal current reference will be turned off, the oscillator and the phase-frequency detector will be stopped and the dividers will enter a reset state. While in Power-down mode, the lock output will be low, to indicate that the PLL is not in lock. When the Power-down mode is terminated, the PLL will resume its normal operation and will make the lock signal high once it has regained lock on the input clock.

11.8 Example CGU configurations

11.8.1 Programming the CGU for Deep-sleep and Power-down modes

Before the LPC43xx enters Deep-sleep or Power-down mode, the IRC must be programmed as the clock source in the control registers for all output stages (OUTCLK_0 to OUTCLK_27). In addition, the PLLs must be in Power-down mode.

When the LPC43xx wakes up from Deep-sleep or Power-down mode, the IRC is used as the clock sources for all output stages. Also see Chapter 10.

11.8.2 Programming the CGU for using I2S at peripheral clock rate of 30 MHz

In this example the peripheral clock of the I2S interface is set to 30 MHz. The peripheral I2S clock is a branch of the BASE_APB1_CLK. Using a crystal of 12 MHz as clock source, a PLL1 multiplier of 10, and an integer divider of 4 provide the desired clock rate.

For this example, program the CGU as follows:

1. Enable the crystal oscillator in the XTAL_OSC_CTRL register (Table 60).
2. Wait for the crystal to stabilize.
3. Select the crystal oscillator as input to the PLL1 and set up the divider in the PLL1_CTRL register (see Table 71):
   - Set bits CLK_SEL in the PLL1_CTRL register to 0x6.
   - Set MSEL = 9.
   - Set NSEL = 0.
   - Set PSEL = 1.
   - Set FBSEL = 1.
   - Set BYPASS = 0, DIRECT = 0.
4. Wait for the PLL1 to lock.
5. Select the PLL1 as clock source of the integer divider A (IDIVA) in the IDIVA register and set AUTOBLOCK = 1 (see Table 71).
6. Select IDIVA as clock source of the base clock BASE_APB1_CLK and set AUTOBLOCK = 1 (see Table 72).
7. Ensure that the I2S branch clock CLK_APB1_I2S is enabled in the CCU (see Table 91).

11.8.3 PLL0USB settings for USB applications

Table 85 shows examples of the M-divider and NP-divider register settings that produce a 480 MHz output clock for the USB0.

Table 85. PLL0 (for USB) settings for 480 MHz output clock

Fclkin [MHz]	PLL0USB_MDIV Table 63	PLL0USB_NP_DIV Table 64
1	0x073E 56C9	0x0030 2062
2	0x073E 2DAD	0x0030 2062
3	0x0B3E 34B1	0x0030 2062
4	0x0E3E 7777	0x0030 2062
5	0x0D32 6667	0x0030 2062
6	0x0B2A 2A66	0x0030 2062
8	0x0820 6AAA	0x0030 2062
10	0x071A 7FAA	0x0030 2062
12	0x0616 7FFA	0x0030 2062
15	0x0512 3FFF	0x0030 2062
16	0x0410 1FFF	0x0030 2062
20	0x040E 03FF	0x0030 2062
24	0x030C 00FF	0x0030 2062

11.8.4 PLL0AUDIO settings for audio applications

11.8.4.1 Using the fractional divider

Table 86 shows typical divider settings for the audio PLL0 with the fractional divider active.

To use the fractional divider, follow these steps:

1. Set bit SEL_EXT = 0 and PLLFRACT_REQ = 1 in the PLL0AUDIO_CTRL register (Table 66).
2. Calculate NDEC, PDEC, and PLLFRACT_CTRL for the output frequency Fout.
3. Write the calculated NDEC and PDEC values to the PLL0AUDIO_NP_DIV register.
4. Write the calculated PLLFRACT_CTRL value to the PLL0AUDIOFRAC register.

Table 86. PLL0AUDIO divider settings for 12 MHz input

Fs [kHz]	Fout [MHz]	Fcco [MHz]	Error [Hz]	NDEC	PDEC	PLL0AUDIO_NP_DIV Table 68	PLLF0RACT_CTRL Table 69
128Fs							
192	24.576	540.672	1	514	29	0x0000201d	0x16872b
96	12.288	417.792	1	1	3	0x00001003	0x1a1cac
88.2	11.2896	338.688	1	0	24	0x00000018	0x070e56
64	8.192	344.064	1	0	30	0x0000001e	0x072b02
48	6.144	307.2	1	1	6	0x00001006	0x133333
44.1	5.6448	282.24	1	1	6	0x00001006	0x11a3d7
256Fs							
192	49.152	491.52	11	514	5	0x00002005	0x147ae1
96	24.576	540.672	1	514	29	0x0000201d	0x16872b
88.2	22.5792	451.584	1	1	14	0x0000100e	0x1c3958
64	16.384	360.448	1	1	29	0x0000101d	0x16872b

Table 86. PLL0AUDIO divider settings for 12 MHz input

Fs [kHz]	Fout [MHz]	Fcco [MHz]	Error [Hz]	NDEC	PDEC	PLL0AUDIO_NP_DIV Table 68	PLLF0RACT_CTRL Table 69
48	12.288	417.792	1	1	3	0x00001003	0x1a1cac
44.1	11.2896	338.688	1	0	24	0x00000018	0x070e56
32	8.192	344.064	1	0	30	0x0000001e	0x072b02
24	6.144	307.2	1	1	6	0x00001006	0x133333
22.05	5.6448	282.24	1	1	6	0x00001006	0x11a3d7
384Fs							
192	73.728	442.368	4	2		0x00002001	0x24dd2f
96	36.864	442.368	2	2		0x0000200a	0x24dd2f
88.2	33.8688	338.688	2	0		0x00000005	0x070e56
64	24.576	540.672	1	514		0x0000201d	0x16872b
48	18.432	442.368	1	2		0x0000201b	0x24dd2f
44.1	16.9344	338.688	1	0		0x0000000e	0x070e56
32	12.288	417.792	1	1		0x00001003	0x1a1cac
24	9.216	276.48	1	1		0x00001018	0x1147ae
22.05	8.4672	338.688	1	0		0x0000001f	0x070e56
16	6.144	307.2	1	1		0x00001006	0x133333
12	4.608	276.48	1	1		0x00001012	0x1147ae
512Fs							
192	98.304	393.216	30	2	66	0x00002042	0x20c49b
96	49.152	491.52	11	514	5	0x00002005	0x147ae1
88.2	45.1584	451.584	2	1	5	0x00001005	0x1c3958
64	32.768	458.752	4	1	21	0x00001015	0x1cac08
48	24.576	540.672	1	514	29	0x0000201d	0x16872b
44.1	22.5792	451.584	1	1	14	0x0000100e	0x1c3958
32	16.384	360.448	1	1	29	0x0000101d	0x16872b
24	12.288	417.792	1	1	3	0x00001003	0x1a1cac
22.05	11.2896	338.688	1	0	24	0x00000018	0x070e56
16	8.192	344.064	1	0	30	0x0000001e	0x072b02
12	6.144	307.2	1	1	6	0x00001006	0x133333
11.025	5.6448	282.24	1	1	6	0x00001006	0x11a3d7
1024Fs							
192	196.608	393.216	60	2	98	0x00002062	0x20c49b
96	98.304	393.216	30	2	66	0x00002042	0x20c49b
88.2	90.3168	541.9008	28	0	1	0x00000001	0x0b4a23
64	65.536	524.288	13	2	2	0x00002002	0x2bb0cf
48	49.152	491.52	11	514	5	0x00002005	0x147ae1
44.1	45.1584	451.584	2	1	5	0x00001005	0x1c3958
32	32.768	458.752	4	1	21	0x00001015	0x1cac08
24	24.576	540.672	1	514	29	0x0000201d	0x16872b
22.05	22.5792	451.584	1	1	14	0x0000100e	0x1c3958

Table 86. PLL0AUDIO divider settings for 12 MHz input

Fs [kHz]	Fout [MHz]	Fcco [MHz]	Error [Hz]	NDEC	PDEC	PLL0AUDIO_NP_DIV Table 68	PLLF0RACT_CTRL Table 69
16	16.384	360.448	1	1	29	0x0000101d	0x16872b
12	12.288	417.792	1	1	3	0x00001003	0x1a1cac
11.025	11.2896	338.688	1	0	24	0x00000018	0x070e56
8	8.192	344.064	1	0	30	0x0000001e	0x072b02

11.8.4.2 Bypassing the fractional divider

To bypass the fractional divider and use the MDEC value directly, follow these steps:

1. Set bit SEL_EXT = 1 in the PLL0AUDIO_CTRL register (Table 66).
2. Calculate NDEC, PDEC, and MDEC for the output frequency Fout.
3. Write the calculated NDEC and PDEC values to the PLL0AUDIO_NP_DIV register.
4. Write the calculated MDEC value to the PLL0AUDIO_MDIV register.

Table 87. PLL0AUDIO divider setting for 12 MHz with fractional divider bypassed

Fs [KHz]	Fout [MHz]	Fcco [MHz]	Error [Hz]	NDEC	MDEC	PDEC	PLL0AUDIO_MDIV Table 67	PLL0AUDIO_NP_DIV Table 68
128 Fs								
192	24.576	491.52	0	63	13523	14	0x000034d3	0x0003f00e
96	12.288	368.64	0	63	2665	24	0x00000a69	0x0003f018
88.2	11.2896	338.688	0	45	18810	24	0x0000497a	0x0002d018
64	8.192	409.6	0	61	18724	6	0x00004924	0x0003d006
48	6.144	307.2	0	5	30580	6	0x00007774	0x00005006
44.1	5.6448	282.24	0	63	31356	6	0x00007a7c	0x0003f006
256 Fs								
192	49.152	491.52	0	63	13523	5	0x000034d3	0x0003f005
96	24.576	491.52	0	63	13523	14	0x000034d3	0x0003f00e
88.2	22.5792	451.584	0	45	29122	14	0x000071c2	0x0002d00e
64	16.384	491.52	0	63	13523	24	0x000034d3	0x0003f018
48	12.288	368.64	0	63	2665	24	0x00000a69	0x0003f018
44.1	11.2896	338.688	0	45	18810	24	0x0000497a	0x0002d018
32	8.192	409.6	0	61	18724	6	0x00004924	0x0003d006
24	6.144	307.2	0	5	30580	6	0x00007774	0x00005006
22.05	5.6448	282.24	0	63	31356	6	0x00007a7c	0x0003f006
384 Fs								
192	73.728	294.912	0	45	4770	66	0x000012a2	0x0002d042
96	36.864	368.64	0	63	2665	5	0x00000a69	0x0003f005
88.2	33.8688	338.688	0	45	18810	5	0x0000497a	0x0002d005
64	24.576	491.52	0	63	13523	14	0x000034d3	0x0003f00e

Table 87. PLL0AUDIO divider setting for 12 MHz with fractional divider bypassed

Fs [KHz]	Fout [MHz]	Fcco [MHz]	Error [Hz]	NDEC	MDEC	PDEC	PLL0AUDIO_MDIV Table 67	PLL0AUDIO_NP_DIV Table 68
48	18.432	368.64	0	63	2665	14	0x00000a69	0x0003f00e
44.1	16.9344	338.688	0	45	18810	14	0x0000497a	0x0002d00e
32	12.288	368.64	0	63	2665	24	0x00000a69	0x0003f018
24	9.216	460.8	0	5	12733	6	0x000031bd	0x00005006
22.05	8.4672	423.36	0	63	17692	6	0x0000451c	0x0003f006
16	6.144	307.2	0	5	30580	6	0x00007774	0x00005006
12	4.608	276.48	0	63	1513	18	0x000005e9	0x0003f012
512 Fs								
192	98.304	393.216	0	45	10784	66	0x00002a20	0x0002d042
96	49.152	491.52	0	63	13523	5	0x000034d3	0x0003f005
88.2	45.1584	451.584	0	45	29122	5	0x000071c2	0x0002d005
64	32.768	327.68	0	102	7482	5	0x00001d3a	0x00066005
48	24.576	491.52	0	63	13523	14	0x000034d3	0x0003f00e
44.1	22.5792	451.584	0	45	29122	14	0x000071c2	0x0002d00e
32	16.384	491.52	0	63	13523	24	0x000034d3	0x0003f018
24	12.288	368.64	0	63	2665	24	0x00000a69	0x0003f018
22.05	11.2896	338.688	0	45	18810	24	0x0000497a	0x0002d018
16	8.192	409.6	0	61	18724	6	0x00004924	0x0003d006
12	6.144	307.2	0	5	30580	6	0x00007774	0x00005006
11.025	5.6448	282.24	0	63	31356	6	0x00007a7c	0x0003f006
1024 Fs								
192	196.608	393.216	0	45	10784	98	0x00002a20	0x0002d062
96	98.304	393.216	0	45	10784	66	0x00002a20	0x0002d042
88.2	90.3168	541.901	31	181	3535	1	0x00000dcf	0x000b5001
64	65.536	393.216	0	45	10784	1	0x00002a20	0x0002d001
48	49.152	491.52	0	63	13523	5	0x000034d3	0x0003f005
44.1	45.1584	451.584	0	45	29122	5	0x000071c2	0x0002d005
32	32.768	327.68	0	102	7482	5	0x00001d3a	0x00066005
24	24.576	491.52	0	63	13523	14	0x000034d3	0x0003f00e
22.05	22.5792	451.584	0	45	29122	14	0x000071c2	0x0002d00e
16	16.384	491.52	0	63	13523	24	0x000034d3	0x0003f018
12	12.288	368.64	0	63	2665	24	0x00000a69	0x0003f018
11.025	11.2896	338.688	0	45	18810	24	0x0000497a	0x0002d018
8	8.192	409.6	0	61	18724	6	0x00004924	0x0003d006

12.1 How to read this chapter

Remark: The VADC is not available on parts LPC4350/30/20/10.

Ethernet, USB0, USB1, and LCD-related clocks are not available on all packages. See Table 2.

12.2 Basic configuration

The CCU1/2 are configured as follows:

- See Table 88 for clocking and power control.
- All branch clocks are enabled by default.
- Do not reset the CCUs during normal operation.
- Configure the output clock for the EMC clock divider (Table 97) together with bit 16 in the CREG6 register (Table 43).

Table 88. CCU clocking and power control

	Base clock	Branch clock	Operating frequency
CCU1	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz
CCU2	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz

12.3 Features

The CCUs switch the clocks to individual peripherals on or off.

- Auto mode activates the AHB disable protocol before switching off the branch clock.
- In Wake-up mode, clocks can be selected to run automatically after a wake-up event.

12.4 General description

Each CGU base clock has several clock branches which can be turned on or off independently by the Clock Control Units CCU1 or CCU2. The branch clocks are distributed between CCU1 and CCU2.

Table 89. CCU1 branch clocks

Base clock	Branch clock	Description
BASE_APB3_CLK	CLK_APB3_BUS	APB3 bus clock.
	CLK_APB3_I2C1	Clock to the I2C1 register interface and I2C1 peripheral clock.
	CLK_APB3_DAC	Clock to the DAC register interface.
	CLK_APB3_ADC0	Clock to the ADC0 register interface and ADC0 peripheral clock.
	CLK_APB3_ADC1	Clock to the ADC1 register interface and ADC1 peripheral clock.
	CLK_APB3_CAN0	Clock to the C_CAN0 register interface and C_CAN0 peripheral clock.
BASE_APB1_CLK	CLK_APB1_BUS	APB1 bus clock.
	CLK_APB1_MOTOCON	Clock to the PWM Motor control block and PWM Motor control peripheral clock.
	CLK_APB1_I2C0	Clock to the I2C0 register interface and I2C0 peripheral clock.
	CLK_APB1_I2S	Clock to the I2S0 and I2S1 register interfaces and I2S0 and I2S1 peripheral clock.
	CLK_APB1_CAN1	Clock to the C_CAN1 register interface and C_CAN1 peripheral clock.
BASE_SPIFI_CLK	CLK_SPIFI	clock for the SPIFI SCKI clock input.
BASE_M4_CLK	CLK_M4_BUS	M4 bus clock.
	CLK_M4_SPIFI	Clock to the SPIFI register interface.
	CLK_M4_GPIO	Clock to the GPIO register interface
	CLK_M4_LCD	Clock to the LCD register interface.
	CLK_M4_ETHERNET	Clock to the Ethernet register interface.
	CLK_M4_USB0	Clock to the USB0 register interface.
	CLK_M4_EMC	Clock to the External memory controller.
	CLK_M4_SDIO	Clock to the SD/MMC register interface.
	CLK_M4_DMA	Clock to the DMA register interface.
	CLK_M4_M4CORE	Clock to the Cortex-M4 core
	CLK_M4_SCT	Clock to the SCT register interface.
	CLK_M4_USB1	Clock to the USB1 register interface.
	CLK_M4_EMC_DIV	Clock to the EMC with clock divider.
	CLK_M4_M0APP	Clock to the M0APP coprocessor.
	CLK_M4_VADC	Clock to the VADC.
	CLK_M4_WWDT	Clock to the WWDT register interface.
	CLK_M4_UART0	Clock to the USART0 register interface.
	CLK_M4_UART1	Clock to the UART1 register interface.
	CLK_M4_SSP0	Clock to the SSP0 register interface.
	CLK_M4_TIMER0	Clock to the timer0 register interface and timer0 peripheral clock.
	CLK_M4_TIMER1	Clock to the timer1 register interface and timer1 peripheral clock.

Table 89. CCU1 branch clocks

Base clock	Branch clock	Description
BASE_M4_CLK	CLK_M4_SCU	Clock to the System control unit register interface.
	CLK_M4_CREG	Clock to the CREG register interface.
	CLK_M4_RITIMER	Clock to the RI timer register interface and RI timer peripheral clock.
	CLK_M4_UART2	Clock to the UART2 register interface.
	CLK_M4_UART3	Clock to the UART3 register interface.
	CLK_M4_TIMER2	Clock to the timer2 register interface and timer2 peripheral clock.
	CLK_M4_TIMER3	Clock to the timer3 register interface and timer3 peripheral clock.
	CLK_M4_SSP1	Clock to the SSP1 register interface.
	CLK_M4_QEI	Clock to the QEI register interface and QEI peripheral clock.
BASE_PERIPH_CLK	CLK_PERIPH_BUS	Clock to the peripheral bus.
	CLK_PERIPH_CORE	Clock to the peripheral core.
	CLK_PERIPH_SGPIO	Clock to the SGPIO interface.
BASE_USB0_CLK	CLK_USB0	USB0 peripheral clock.
BASE_USB1_CLK	CLK_USB1	USB1 peripheral clock.
BASE_SPI_CLK	CLK_SPI	Clock to the SPI interface.
BASE_VADC_CLK	CLK_VADC	VADC clock.

Table 90. CCU2 branch clocks

Base clock	Branch clock	Description
BASE_APLL_CLK	CLK_APLL	Audio PLL clock
BASE_UART3_CLK	CLK_APB2_UART3	USART3 peripheral clock.
BASE_UART2_CLK	CLK_APB2_UART2	USART2 peripheral clock.
BASE_UART1_CLK	CLK_APB0_UART1	UART1 peripheral clock.
BASE_UART0_CLK	CLK_APB0_UART0	USART0 peripheral clock.
BASE_SSP1_CLK	CLK_APB2_SSP1	SSP1 peripheral clock.
BASE_SSP0_CLK	CLK_APB0_SSP0	SSP0 peripheral clock.
BASE_SDIO_CLK	CLK_SDIO	SD/MMC peripheral clock.

12.5 Register description

Table 91. Register overview: CCU1 (base address 0x4005 1000)

Name	Access	Address offset	Description	Reset value	Reference
PM	R/W	0x000	CCU1 power mode register	0x0000 0000	Table 93
BASE_STAT	R	0x004	CCU1 base clock status register	0x0000 0FFF	Table 94
-	-	0x008 to 0x0FC	Reserved	-	
CLK_APB3_BUS_CFG	R/W	0x100	CLK_APB3_BUS clock configuration register	0x0000 0001	Table 96
CLK_APB3_BUS_STAT	R	0x104	CLK_APB3_BUS clock status register	0x0000 0001	Table 99
CLK_APB3_I2C1_CFG	R/W	0x108	CLK_APB3_I2C1 configuration register	0x0000 0001	Table 96
CLK_APB3_I2C1_STAT	R	0x10C	CLK_APB3_I2C1 status register	0x0000 0001	Table 99
CLK_APB3_DAC_CFG	R/W	0x110	CLK_APB3_DAC configuration register	0x0000 0001	Table 96
CLK_APB3_DAC_STAT	R	0x114	CLK_APB3_DAC status register	0x0000 0001	Table 99
CLK_APB3_ADC0_CFG	R/W	0x118	CLK_APB3_ADC0 configuration register	0x0000 0001	Table 96
CLK_APB3_ADC0_STAT	R	0x11C	CLK_APB3_ADC0 status register	0x0000 0001	Table 99
CLK_APB3_ADC1_CFG	R/W	0x120	CLK_APB3_ADC1 configuration register	0x0000 0001	Table 96
CLK_APB3_ADC1_STAT	R	0x124	CLK_APB3_ADC1 status register	0x0000 0001	Table 99
CLK_APB3_CAN0_CFG	R/W	0x128	CLK_APB3_CAN0 configuration register	0x0000 0001	Table 96
CLK_APB3_CAN0_STAT	R	0x12C	CLK_APB3_CAN0 status register	0x0000 0001	Table 99
-	-	0x130 to 0x1FC	Reserved	-	-
CLK_APB1_BUS_CFG	R/W	0x200	CLK_APB1_BUS configuration register	0x0000 0001	Table 96
CLK_APB1_BUS_STAT	R	0x204	CLK_APB1_BUS status register	0x0000 0001	Table 99
CLK_APB1_MOTOCON PWM_CFG	R/W	0x208	CLK_APB1_MOTOCON configuration register	0x0000 0001	Table 96
CLK_APB1_MOTOCON PWM_STAT	R	0x20C	CLK_APB1_MOTOCON status register	0x0000 0001	Table 99
CLK_APB1_I2C0_CFG	R/W	0x210	CLK_APB1_I2C0 configuration register	0x0000 0001	Table 96
CLK_APB1_I2C0_STAT	R	0x214	CLK_APB1_I2C0 status register	0x0000 0001	Table 99
CLK_APB1_I2S_CFG	R/W	0x218	CLK_APB1_I2S configuration register	0x0000 0001	Table 96
CLK_APB1_I2S_STAT	R	0x21C	CLK_APB1_I2S status register	0x0000 0001	Table 99
CLK_APB1_CAN1_CFG	R/W	0x220	CLK_APB3_CAN1 configuration register	0x0000 0001	Table 96
CLK_APB1_CAN1_STAT	R	0x224	CLK_APB3_CAN1 status register	0x0000 0001	Table 99
-	-	0x220 to 0x2FC	Reserved	-	-
CLK_SPIFI_CFG	R/W	0x300	CLK_SPIFI configuration register	0x0000 0001	Table 96
CLK_SPIFI_STAT	R	0x304	CLK_SPIFI status register	0x0000 0001	Table 99
-	-	0x308 to 0x3FC	Reserved	-	-
CLK_M4_BUS_CFG	R/W	0x400	CLK_M4_BUS configuration register	0x0000 0001	Table 96
CLK_M4_BUS_STAT	R	0x404	CLK_M4_BUS status register	0x0000 0001	Table 99
CLK_M4_SPIFI_CFG	R/W	0x408	CLK_M4_SPIFI configuration register	0x0000 0001	Table 96

Table 91. Register overview: CCU1 (base address 0x4005 1000)

Name	Access	Address offset	Description	Reset value	Reference
CLK_M4_SPIFI_STAT	R	0x40C	CLK_M4_SPIFI status register	0x0000 0001	Table 99
CLK_M4_GPIO_CFG	R/W	0x410	CLK_M4_GPIO configuration register	0x0000 0001	Table 96
CLK_M4_GPIO_STAT	R	0x414	CLK_M4_GPIO status register	0x0000 0001	Table 99
CLK_M4_LCD_CFG	R/W	0x418	CLK_M4_LCD configuration register	0x0000 0001	Table 96
CLK_M4_LCD_STAT	R	0x41C	CLK_M4_LCD status register	0x0000 0001	Table 99
CLK_M4_ETHERNET_CFG	R/W	0x420	CLK_M4_ETHERNET configuration register	0x0000 0001	Table 96
CLK_M4_ETHERNET_STAT	R	0x424	CLK_M4_ETHERNET status register	0x0000 0001	Table 99
CLK_M4_USB0_CFG	R/W	0x428	CLK_M4_USB0 configuration register	0x0000 0001	Table 96
CLK_M4_USB0_STAT	R	0x42C	CLK_M4_USB0 status register	0x0000 0001	Table 99
CLK_M4_EMC_CFG	R/W	0x430	CLK_M4_EMC configuration register	0x0000 0001	Table 96
CLK_M4_EMC_STAT	R	0x434	CLK_M4_EMC status register	0x0000 0001	Table 99
CLK_M4_SDIO_CFG	R/W	0x438	CLK_M4_SDIO configuration register	0x0000 0001	Table 96
CLK_M4_SDIO_STAT	R	0x43C	CLK_M4_SDIO status register	0x0000 0001	Table 99
CLK_M4_DMA_CFG	R/W	0x440	CLK_M4_DMA configuration register	0x0000 0001	Table 96
CLK_M4_DMA_STAT	R	0x444	CLK_M4_DMA status register	0x0000 0001	Table 99
CLK_M4_M4CORE_CFG	R/W	0x448	CLK_M4_M4CORE configuration register	0x0000 0001	Table 96
CLK_M4_M4CORE_STAT	R	0x44C	CLK_M4_M4CORE status register	0x0000 0001	Table 99
-	-	0x450 to 0x45C	Reserved	-	-
-	-	0x460 to 0x464	Reserved	-	-
CLK_M4_SCT_CFG	R/W	0x468	CLK_M4_SCT configuration register	0x0000 0001	Table 96
CLK_M4_SCT_STAT	R	0x46C	CLK_M4_SCT status register	0x0000 0001	Table 99
CLK_M4_USB1_CFG	R/W	0x470	CLK_M4_USB1 configuration register	0x0000 0001	Table 96
CLK_M4_USB1_STAT	R	0x474	CLK_M4_USB1 status register	0x0000 0001	Table 99
CLK_M4_EMCDIV_CFG	R/W	0x478	CLK_M4_EMCDIV configuration register	0x0000 0001	Table 97
CLK_M4_EMCDIV_STAT	R	0x47C	CLK_M4_EMCDIV status register	0x0000 0001	Table 99
-	-	0x480	-	-	-
-	-	0x484	-	-	-
-	-	0x488	-	-	-
-	-	0x48C	-	-	-
CLK_M4_M0APP_CFG	-	0x490	CLK_M4_M0_CFG configuration register	0x0000 0001	Table 97
CLK_M4_M0APP_STAT	-	0x494	CLK_M4_M0_STAT status register	0x0000 0001	Table 99
CLK_M4_VADC_CFG	-	0x498	CLK_M4_VADC_CFG configuration register	0x0000 0001	Table 97
CLK_M4_VADC_STAT		0x49C	CLK_M4_VADC_STAT configuration register	0x0000 0001	Table 99
-	-	0x480 to 0x4FC	Reserved	-	
CLK_M4_WWDT_CFG	R/W	0x500	CLK_M4_WWDT configuration register	0x0000 0001	Table 96
CLK_M4_WWDT_STAT	R	0x504	CLK_M4_WWDT status register	0x0000 0001	Table 99

Table 91. Register overview: CCU1 (base address 0x4005 1000)

Name	Access	Address offset	Description	Reset value	Reference
CLK_M4_USART0_CFG	R/W	0x508	CLK_M4_UART0 configuration register	0x0000 0001	Table 96
CLK_M4_USART0_STAT	R	0x50C	CLK_M4_UART0 status register	0x0000 0001	Table 99
CLK_M4_UART1_CFG	R/W	0x510	CLK_M4_UART1 configuration register	0x0000 0001	Table 96
CLK_M4_UART1_STAT	R	0x514	CLK_M4_UART1 status register	0x0000 0001	Table 99
CLK_M4_SSP0_CFG	R/W	0x518	CLK_M4_SSP0 configuration register	0x0000 0001	Table 96
CLK_M4_SSP0_STAT	R	0x51C	CLK_M4_SSP0 status register	0x0000 0001	Table 99
CLK_M4_TIMER0_CFG	R/W	0x520	CLK_M4_TIMER0 configuration register	0x0000 0001	Table 96
CLK_M4_TIMER0_STAT	R	0x524	CLK_M4_TIMER0 status register	0x0000 0001	Table 99
CLK_M4_TIMER1_CFG	R/W	0x528	CLK_M4_TIMER1 configuration register	0x0000 0001	Table 96
CLK_M4_TIMER1_STAT	R	0x52C	CLK_M4_TIMER1 status register	0x0000 0001	Table 99
CLK_M4_SCU_CFG	R/W	0x530	CLK_M4_SCU configuration register	0x0000 0001	Table 96
CLK_M4_SCU_STAT	R	0x534	CLK_M4_SCU status register	0x0000 0001	Table 99
CLK_M4_CREG_CFG	R/W	0x538	CLK_M4_CREG configuration register	0x0000 0001	Table 96
CLK_M4_CREG_STAT	R	0x53C	CLK_M4_CREG status register	0x0000 0001	Table 99
-	-	0x540 to 0x5FC	Reserved	-	-
CLK_M4_RITIMER_CFG	R/W	0x600	CLK_M4_RITIMER configuration register	0x0000 0001	Table 96
CLK_M4_RITIMER_STAT	R	0x604	CLK_M4_RITIMER status register	0x0000 0001	Table 99
CLK_M4_USART2_CFG	R/W	0x608	CLK_M4_UART2 configuration register	0x0000 0001	Table 96
CLK_M4_USART2_STAT	R	0x60C	CLK_M4_UART2 status register	0x0000 0001	Table 99
CLK_M4_USART3_CFG	R/W	0x610	CLK_M4_UART3 configuration register	0x0000 0001	Table 96
CLK_M4_USART3_STAT	R	0x614	CLK_M4_UART3 status register	0x0000 0001	Table 99
CLK_M4_TIMER2_CFG	R/W	0x618	CLK_M4_TIMER2 configuration register	0x0000 0001	Table 96
CLK_M4_TIMER2_STAT	R	0x61C	CLK_M4_TIMER2 status register	0x0000 0001	Table 99
CLK_M4_TIMER3_CFG	R/W	0x620	CLK_M4_TIMER3 configuration register	0x0000 0001	Table 96
CLK_M4_TIMER3_STAT	R	0x624	CLK_M4_TIMER3 status register	0x0000 0001	Table 99
CLK_M4_SSP1_CFG	R/W	0x628	CLK_M4_SSP1 configuration register	0x0000 0001	Table 96
CLK_M4_SSP1_STAT	R	0x62C	CLK_M4_SSP1 status register	0x0000 0001	Table 99
CLK_M4_QEI_CFG	R/W	0x630	CLK_M4_QEI configuration register	0x0000 0001	Table 96
CLK_M4_QEI_STAT	R	0x634	CLK_M4_QEI status register	0x0000 0001	Table 99
-	R/W	0x638 to 0x6FC	Reserved	-	-
CLK_PERIPH_BUS_CFG	R/W	0x700	CLK_PERIPH_BUS configuration register	0x0000 0001	Table 96
CLK_PERIPH_BUS_STAT	R	0x704	CLK_PERIPH_BUS status register	0x0000 0001	Table 99
-	R/W	0x708 to 0x70C	Reserved	-	-
CLK_PERIPH_CORE_CFG	R/W	0x710	CLK_PERIPH_CORE configuration register	0x0000 0001	Table 96
CLK_PERIPH_CORE_STAT	R	0x714	CLK_PERIPH_CORE status register	0x0000 0001	Table 99
CLK_PERIPH_SGPIO_CFG	R/W	0x718	CLK_PERIPH_SGPIO configuration register	0x0000 0001	Table 96

Table 91. Register overview: CCU1 (base address 0x4005 1000)

Name	Access	Address offset	Description	Reset value	Reference
CLK_PERIPH_SGPIO_STAT	R	0x71C	CLK_PERIPH_SGPIO status register	0x0000 0001	Table 99
-	R/W	0x720 to 0x7FC	Reserved	-	-
CLK_USB0_CFG	R/W	0x800	CLK_USB0 configuration register	0x0000 0001	Table 96
CLK_USB0_STAT	R	0x804	CLK_USB0 status register	0x0000 0001	Table 99
-	-	0x808 to 0x8FC	Reserved	-	-
CLK_USB1_CFG	R/W	0x900	CLK_USB1 configuration register	0x0000 0001	Table 96
CLK_USB1_STAT	R	0x904	CLK_USB1 status register	0x0000 0001	Table 99
-	-	0x908 to 0x9FC	Reserved	-	-
CLK_SPI_CFG	R/W	0xA00	CLK_SPI configuration register	0x0000 0001	Table 96
CLK_SPI_STAT	R	0xA04	CLK_SPI status register	0x0000 0001	Table 99
CLK_VADC_CFG	R/W	0xB00	CLK_VADC configuration register	0x0000 0001	Table 96
CLK_VADC_STAT	R	0xB04	CLK_VADC status register	0x0000 0001	Table 99

Table 92. Register overview: CCU2 (base address 0x4005 2000)

Name	Access	Address offset	Description	Reset value	Reference
PM	R/W	0x000	CCU2 power mode register	0x0000 0000	Table 93
BASE_STAT	R	0x004	CCU2 base clocks status register	0x0000 0FFF	Table 95
-	-	0x008 to 0x0FC	Reserved	-	
CLK_APLL_CFG	R/W	0x100	CLK_APLL configuration register	0x0000 0001	Table 98
CLK_APLL_STAT	R	0x104	CLK_APLL status register	0x0000 0001	Table 100
-	-	0x108 to 0x1FC	Reserved	-	
CLK_APB2_USART3_CFG	R/W	0x200	CLK_APB2_UART3 configuration register	0x0000 0001	Table 98
CLK_APB2_USART3_STAT	R	0x204	CLK_APB2_UART3 status register	0x0000 0001	Table 100
-	-	0x208 to 0x2FC	Reserved	-	
CLK_APB2_USART2_CFG	R/W	0x300	CLK_APB2_UART2 configuration register	0x0000 0001	Table 98
CLK_APB2_USART2_STAT	R	0x304	CLK_APB2_UART2 status register	0x0000 0001	Table 100
-	-	0x308 to 0x3FC	Reserved	-	
CLK_APB0_UART1_CFG	R/W	0x400	CLK_APB0_UART1 configuration register	0x0000 0001	Table 98
CLK_APB0_UART1_STAT	R	0x404	CLK_APB0_UART1 status register	0x0000 0001	Table 100
-	-	0x408 to 0x4FC	Reserved	-	
CLK_APB0_USART0_CFG	R/W	0x500	CLK_APB0_UART0 configuration register	0x0000 0001	Table 98
CLK_APB0_USART0_STAT	R	0x504	CLK_APB0_UART0 status register	0x0000 0001	Table 100
-	-	0x508 to 0x5FC	Reserved	-	

Table 92. Register overview: CCU2 (base address 0x4005 2000)

Name	Access	Address offset	Description	Reset value	Reference
CLK_APB2_SSP1_CFG	R/W	0x600	CLK_APB2_SSP1 configuration register	0x0000 0001	Table 98
CLK_APB2_SSP1_STAT	R	0x604	CLK_APB2_SSP1 status register	0x0000 0001	Table 100
-	-	0x608 to 0x6FC	Reserved	-	
CLK_APB0_SSP0_CFG	R/W	0x700	CLK_APB0_SSP0 configuration register	0x0000 0001	Table 98
CLK_APB0_SSP0_STAT	R	0x704	CLK_APB0_SSP0 status register	0x0000 0001	Table 100
-	-	0x708 to 0x7FC	Reserved	-	
CLK_SDIO_CFG	R/W	0x800	CLK_SDIO configuration register (for SD/MMC)	0x0000 0001	Table 98
CLK_SDIO_STAT	R	0x804	CLK_SDIO status register (for SD/MMC)	0x0000 0001	Table 100

12.5.1 Power mode register

This register contains a single bit, PD, that disables all output clocks with Wake-up enabled. (W = 1 in the CCU branch clock configuration registers, Section 12.5.3). Clocks disabled by writing to this register are reactivated when a wake-up interrupt is detected or when a 0 is written into the PD bit.

Table 93. CCU1/2 power mode register (CCU1_PM, address 0x4005 1000 and CCU2_PM, address 0x4005 2000) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	PD		Initiate power-down mode	0	R/W
		0	Normal operation.		
		1	Clocks with wake-up mode enabled (W = 1) are disabled.		
31:1	-		Reserved.	-	-

12.5.2 Base clock status register

Each bit in this register indicates whether the specified base clock can be safely switched off. A logic zero indicates that all branch clocks generated from this base clock are disabled. Hence, the base clock can also be switched off. A logic one value indicates that there is still at least one branch clock running.

Remark: Reactivate the base clock before writing to the configuration register of the branch clock.

Table 94. CCU1 base clock status register (CCU1_BASE_STAT, address 0x4005 1004) bit description

Bit	Symbol	Description	Reset value	Access
0	BASE_APB3_CLK_IND	Base clock indicator for BASE_APB3_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
1	BASE_APB1_CLK_IND	Base clock indicator for BASE_APB1_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
2	BASE_SPIFI_CLK_IND	Base clock indicator for BASE_SPIFI_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
3	BASE_M4_CLK_IND	Base clock indicator for BASE_M4_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
5:4	-	Reserved	-	-
6	BASE_PERIPH_CLK_IND	Base clock indicator for BASE_PERIPH_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
7	BASE_USB0_CLK_IND	Base clock indicator for BASE_USB0_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
8	BASE_USB1_CLK_IND	Base clock indicator for BASE_USB1_CLK 0 = All branch clocks switched off. 1 = at least one branch clock running.	1	R
9	BASE_SPI_CLK_IND	Base clock indicator for BASE_SPI_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
31:10	-	Reserved	-	-

Table 95. CCU2 base clock status register (CCU2_BASE_STAT, address 0x4005 2004) bit description

Bit	Symbol	Description	Reset value	Access
0	-	Reserved.	-	-
1	BASE_UART3_CLK	Base clock indicator for BASE_UART3_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
2	BASE_UART2_CLK	Base clock indicator for BASE_UART2_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
3	BASE_UART1_CLK	Base clock indicator for BASE_UART1_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R

Table 95. CCU2 base clock status register (CCU2_BASE_STAT, address 0x4005 2004) bit description *...continued*

Bit	Symbol	Description	Reset value	Access
4	BASE_UART0_CLK	Base clock indicator for BASE_UART0_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
5	BASE_SSP1_CLK	Base clock indicator for BASE_SSP1_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
6	BASE_SSP0_CLK	Base clock indicator for BASE_SSP0_CLK 0 = All branch clocks switched off. 1 = At least one branch clock running.	1	R
7	-	Reserved.	-	-
31:8	-	Reserved.	-	-

12.5.3 CCU1/2 branch clock configuration registers

Each generated output clock from the CCU has a configuration register. They all follow the format as described in Table 96 and Table 98.

On the LPC43xx, all branch clocks are in Run mode after reset. Auto and wake-up features are disabled.

The clock can be configured to run in the following modes described by the bits RUN, AUTO, and WAKEUP in the CLK_XXX_CFG registers:

RUN — The WAKEUP, PD, and AUTO control bits determine the activation of the branch clock. If register bit AUTO is set, the AHB disable protocol must complete before the clock is switched off. The PD bit is set in Table 93.

AUTO — Enable auto (AHB disable mechanism). The PMU initiates the AHB disable protocol before switching the clock off. This protocol ensures that all AHB transactions have been completed before turning the clock off.

WAKEUP — The branch clock is wake-up enabled under the following conditions:

- The PD bit in the Power Mode register (seeTable 93) is set.
- Wake-up enabled clocks are switched off.

Wake-up enabled clocks are switched on when a wake-up event is detected or when the PD bit is cleared. If register bit AUTO is set, the AHB disable protocol must complete before the clock is switched off.

Remark: To disable any of the branch clocks safely, use two separate writes to the CLK_XXX_CFG register: first set the AUTO bit, and then on the next write, disable the clock by setting the RUN bit to zero.

Table 96. CCU1 branch clock configuration register (CLK_XXX_CFG, addresses 0x4005 1100, 0x4005 1104,..., 0x4005 1A00) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RUN		Run enable	1	R/W
		0	Clock is disabled.		
		1	Clock is enabled.		
1	AUTO		Auto (AHB disable mechanism) enable	0	R/W
		0	Auto is disabled.		
		1	Auto is enabled.		
2	WAKEUP		Wake-up mechanism enable	0	R/W
		0	Wake-up is disabled.		
		1	Wake-up is enabled.		
31:3	-		Reserved	-	-

Remark: Configure the output clock for the EMC clock divider (Table 97) together with bit 16 in the CREG6 register (Table 43).

Table 97. CCU1 branch clock configuration register (CLK_EMCDIV_CFG, addresses 0x4005 1478) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RUN		Run enable	1	R/W
		0	Clock is disabled.		
		1	Clock is enabled.		
1	AUTO		Auto (AHB disable mechanism) enable	0	R/W
		0	Auto is disabled.		
		1	Auto is enabled.		
2	WAKEUP		Wake-up mechanism enable	0	R/W
		0	Wake-up is disabled.		
		1	Wake-up is enabled.		
3	-		Reserved	-	-
4	-		Reserved	-	-
7:5	DIV		Clock divider value	0	R/W
		0x0	No division (divide by 1).		
		0x1	Divide by 2.		
		0x2	Reserved		
		0x3	Reserved		
		0x4	Reserved		
31:8	-		Reserved	-	-

Table 98. CCU2 branch clock configuration register (CLK_XXX_CFG, addresses 0x4005 2100, 0x4005 2200,..., 0x4005 2800) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RUN		Run enable	1	R/W
		0	Clock is disabled.		
		1	Clock is enabled.		
1	AUTO		Auto (AHB disable mechanism) enable	0	R/W
		0	Auto is disabled.		
		1	Auto is enabled.		
2	WAKEUP		Wake-up mechanism enable	0	R/W
		0	Wake-up is disabled.		
		1	Wake-up is enabled.		
31:3	-		Reserved	-	-

12.5.4 CCU1/2 branch clock status registers

Each generated output clock from the CCU has a status register. When writing to the configuration register of an output clock, the Auto or Wake-up mechanism can delay the update of the actual hardware signals. The Status Register shows the current value of these signals. All output clock Status Registers follow the format as described in Table 99 and Table 100.

Table 99. CCU1 branch clock status register (CLK_XXX_STAT, addresses 0x4005 1104, 0x4005 110C,..., 0x4005 1A04) bit description

Bit	Symbol	Description	Reset value	Access
0	RUN	Run enable status 0 = clock is disabled. 1 = clock is enabled.	1	R
1	AUTO	Auto (AHB disable mechanism) enable status 0 = Auto is disabled. 1 = Auto is enabled.	0	R
2	WAKEUP	Wake-up mechanism enable status 0 = Wake-up is disabled. 1 = Wake-up is enabled.	0	R
31:3	-	Reserved	-	-

Table 100. CCU2 branch clock status register (CLK_XXX_STAT, addresses 0x4005 2104, 0x4005 2204,..., 0x4005 2804) bit description

Bit	Symbol	Description	Reset value	Access
0	RUN	Run enable status 0 = clock is disabled 1 = clock is enabled	1	R
1	AUTO	Auto (AHB disable mechanism) enable status 0 = Auto is disabled 1 = Auto is enabled	0	R
2	WAKEUP	Wake-up mechanism enable status 0 = Wake-up is disabled 1 = Wake-up is enabled	0	R
31:3	-	Reserved	-	-

13.1 How to read this chapter

The RGU is identical for all LPC43xx parts. Ethernet, USB0, USB1, and LCD related clocks are not available on all packages. See Section 1.3. The corresponding reset registers are reserved.

13.2 Basic configuration

Table 101. RGU clocking and power control

	Base clock	Branch clock	Operating frequency
RGU	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz
RGU delay clocks	BASE_SAFE_CLK	-	12 MHz

The RGU is reset by a BUS_RST (reset #8).

13.3 General description

The RGU allows generation of independent reset signals for various blocks and peripherals on the LPC43xx. Each reset signal is asserted by a reset generator with one output (the reset signal) and one or more inputs, which link the reset generators together and create a reset hierarchy.

Remark: The ARM Cortex-M4 SYSRESETREQ triggers a peripheral reset PERIPH_RST.

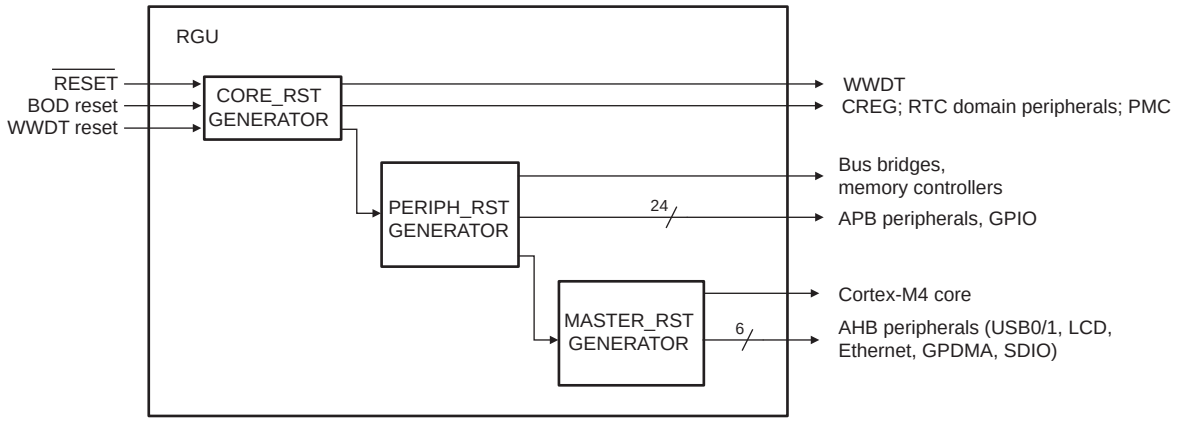


Fig 25. RGU Block diagram

Table 102. Reset output configuration

Reset output generator	Reset output #	Reset source	Parts of the device reset when activated
CORE_RST	0	external reset, BOD reset, WWDT time-out reset	Entire chip including peripherals in the battery-powered domain: CGU, power management controller, general purpose registers, alarm timer, parts of the CREG block, and RTC.
PERIPH_RST	1	CORE_RST	All peripherals with reset source PERIPH_RST and MASTER_RST
MASTER_RST	2	PERIPH_RST	All peripherals with reset source MASTER_RST
Reserved	3	-	-
WWDT_RST	4	CORE_RST	WWDT. No software reset.
CREG_RST	5	CORE_RST	Configuration register block, Event router, backup registers, RTC, alarm timer. No software reset.
Reserved	6 - 7	-	-
BUS_RST	8	PERIPH_RST	Buses; RGU, CCU, and CGU registers; memory controllers; bus bridges. Do not use during normal operation.
SCU_RST	9	PERIPH_RST	System control unit
Reserved	10 - 12	-	-
M4_RST	13	MASTER_RST	Cortex-M4 system reset
Reserved	14	-	-
Reserved	15	-	-
LCD_RST	16	MASTER_RST	LCD controller reset
USB0_RST	17	MASTER_RST	USB0 reset
USB1_RST	18	MASTER_RST	USB1 reset
DMA_RST	19	MASTER_RST	DMA reset
SDIO_RST	20	MASTER_RST	SDIO reset
EMC_RST	21	MASTER_RST	External memory controller reset
ETHERNET_RST	22	MASTER_RST	Ethernet reset
Reserved	23	-	-
Reserved	24 - 27	-	-
GPIO_RST	28	PERIPH_RST	GPIO reset
Reserved	29 - 31	-	-
TIMER0_RST	32	PERIPH_RST	Timer0 reset
TIMER1_RST	33	PERIPH_RST	Timer1 reset
TIMER2_RST	34	PERIPH_RST	Timer2 reset
TIMER3_RST	35	PERIPH_RST	Timer3 reset
RITIMER_RST	36	PERIPH_RST	Repetitive Interrupt timer reset
SCT_RST	37	PERIPH_RST	State Configurable Timer reset
MOTOCONPWM_RST	38	PERIPH_RST	Motor control PWM reset
QEI_RST	39	PERIPH_RST	QEI reset

Table 102. Reset output configuration *...continued*

Reset output generator	Reset output #	Reset source	Parts of the device reset when activated
ADC0_RST	40	PERIPH_RST	ADC0 reset (ADC register interface and analog block)
ADC1_RST	41	PERIPH_RST	ADC1 reset (ADC register interface and analog block)
DAC_RST	42	PERIPH_RST	DAC reset (DAC register interface and analog block)
Reserved	43	-	-
UART0_RST	44	PERIPH_RST	USART0 reset
UART1_RST	45	PERIPH_RST	UART1 reset
UART2_RST	46	PERIPH_RST	USART2 reset
UART3_RST	47	PERIPH_RST	USART3 reset
I2C0_RST	48	PERIPH_RST	I2C0 reset
I2C1_RST	49	PERIPH_RST	I2C1 reset
SSP0_RST	50	PERIPH_RST	SSP0 reset
SSP1_RST	51	PERIPH_RST	SSP1 reset
I2S_RST	52	PERIPH_RST	I2S0 and I2S1 reset
SPIFI_RST	53	PERIPH_RST	SPIFI reset
CAN1_RST	54	PERIPH_RST	C_CAN1 reset
CAN0_RST	55	PERIPH_RST	C_CAN0 reset
M0APP_RST	56	MASTER_RST	ARM Cortex-M0 co-processor reset. **Remark:** Software must clear the M0 co-processor reset by writing to the RESET_CTRL1 register.
SGPIO_RST	57	PERIPH_RST	SGPIO reset
SPI_RST	58	PERIPH_RST	SPI reset
Reserved	59 - 63	-	-

The RGU also monitors the reset cause for each reset output. The reset cause can be retrieved with two levels of granularity.

The first level is monitored by the RESET_STATUS0 to 3 registers and indicates one of the following reset causes (see Table 107 to Table 110):

- No reset has taken place.
- Reset generated by software (using the registers RESET_CTRL0 and RESET_CTRL1).
- Reset generated by any of the reset sources.

The second level of granularity is monitored by one individual register for each reset output (RESET_EXT_STATUSn) in which the detailed reset cause is indicated, that is whether or not any of the possible inputs to each reset generator are activated. The following lists all inputs, but note that only a subset of inputs are connected to each reset generator:

- External reset (from external reset pin)

- CORE_RST output
- PERIPH_RST output
- MASTER_RST output
- BOD reset signal
- WWDT time-out signal

13.3.1 Reset hierarchy

The hierarchy is as follows (see Table 103):

1. External reset, BOD reset signal, WWDT time-out, and reset signal from the PMU
2. CORE_RST (inputs are the external reset pin, BOD reset, and the WWDT time-out reset); resets the whole chip including the WWDT and the configuration register block CREG.
3. PERIPH_RST (input is the CORE_RST); resets all APB peripherals and the ARM core, but not the WWDT and the CREG block.
4. MASTER_RST (input is the PERIPH_RST); resets the ARM Cortex-M4 core and the AHB peripherals (DMA, USB0/1, LCD, SDIO, EMC).

Table 103. Reset priority

Priority	Reset input	WWDT	CREG/ RTC/ Event router	ABP peripherals	Cortex-M4 Core	AHB peripherals	RGU	EMC	GPIO	SRAM controllers
1	External reset pin, BOD, WWDT	yes	yes	yes	yes	yes	yes	yes	yes	yes
2	CORE_RST	yes	yes	yes	yes	yes	yes	yes	yes	yes
3	PERIPH_RST	no	no	yes	yes	yes	yes	yes	yes	yes
4	MASTER_RST	no	no	no	yes	yes	yes	yes	yes	yes

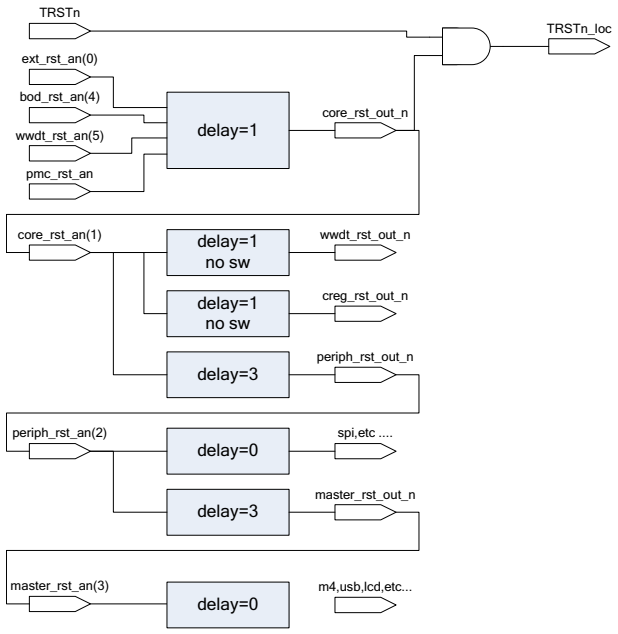


Fig 26. RGU Reset structure

13.4 Register overview

Table 104. Register overview: RGU (base address: 0x4005 3000)

Name	Access	Address offset	Description	Reset value	Reference
RESET_CTRL0	W	0x100	Reset control register 0	-	see Table 105
RESET_CTRL1	W	0x104	Reset control register 1	-	see Table 106
RESET_STATUS0	R/W	0x110	Reset status register 0	0x5555 0050	see Table 107
RESET_STATUS1	R/W	0x114	Reset status register 1	0x5555 5555	see Table 108
RESET_STATUS2	R/W	0x118	Reset status register 2	0x5555 5555	see Table 109
RESET_STATUS3	R/W	0x11C	Reset status register 3	0x5555 5555	see Table 110
RESET_ACTIVE_STATUS0	R	0x150	Reset active status register 0	0x0	see Table 111
RESET_ACTIVE_STATUS1	R	0x154	Reset active status register 1	0x0	see Table 112
RESET_EXT_STAT0	R/W	0x400	Reset external status register 0 for CORE_RST	0x0	see Table 113
RESET_EXT_STAT1	R/W	0x404	Reset external status register 1 for PERIPH_RST	0x0	see Table 114
RESET_EXT_STAT2	R/W	0x408	Reset external status register 2 for MASTER_RST	0x0	see Table 115
RESET_EXT_STAT3	-	0x40C	Reserved	-	
RESET_EXT_STAT4	R/W	0x410	Reset external status register 4 for WWDT_RST	0x0	see Table 116
RESET_EXT_STAT5	R/W	0x414	Reset external status register 5 for CREG_RST	0x0	see Table 117
RESET_EXT_STAT6	-	0x418	Reserved	-	-

Table 104. Register overview: RGU (base address: 0x4005 3000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
RESET_EXT_STAT7	-	0x41C	Reserved	-	-
RESET_EXT_STAT8	R/W	0x420	Reset external status register 8 for BUS_RST	0x0	see Table 118
RESET_EXT_STAT9	R/W	0x424	Reset external status register 9 for SCU_RST	0x0	see Table 118
RESET_EXT_STAT10	-	0x428	Reserved	-	-
RESET_EXT_STAT11	-	0x42C	Reserved	-	-
RESET_EXT_STAT12	-	0x430	Reserved	-	-
RESET_EXT_STAT13	R/W	0x434	Reset external status register 13 for M4_RST	0x0	see Table 119
RESET_EXT_STAT14	-	0x438	Reserved	-	-
RESET_EXT_STAT15	-	0x43C	Reserved	-	-
RESET_EXT_STAT16	R/W	0x440	Reset external status register 16 for LCD_RST	0x0	see Table 119
RESET_EXT_STAT17	R/W	0x444	Reset external status register 17 for USB0_RST	0x0	see Table 119
RESET_EXT_STAT18	R/W	0x448	Reset external status register 18 for USB1_RST	0x0	see Table 119
RESET_EXT_STAT19	R/W	0x44C	Reset external status register 19 for DMA_RST	0x0	see Table 119
RESET_EXT_STAT20	R/W	0x450	Reset external status register 20 for SDIO_RST	0x0	see Table 119
RESET_EXT_STAT21	R/W	0x454	Reset external status register 21 for EMC_RST	0x0	see Table 119
RESET_EXT_STAT22	R/W	0x458	Reset external status register 22 for ETHERNET_RST	0x0	see Table 119
RESET_EXT_STAT23	-	0x45C	Reserved	-	-
RESET_EXT_STAT24	-	0x460	Reserved	-	-
RESET_EXT_STAT25	-	0x464	Reserved	-	-
RESET_EXT_STAT26	-	0x468	Reserved	-	-
RESET_EXT_STAT27	-	0x46C	Reserved	-	-
RESET_EXT_STAT28	R/W	0x470	Reset external status register 28 for GPIO_RST	0x0	see Table 118
RESET_EXT_STAT29	-	0x474	Reserved	-	-
RESET_EXT_STAT30	-	0x478	Reserved	-	-
RESET_EXT_STAT31	-	0x47C	Reserved	-	-
RESET_EXT_STAT32	R/W	0x480	Reset external status register 32 for TIMER0_RST	0x0	see Table 118
RESET_EXT_STAT33	R/W	0x484	Reset external status register 33 for TIMER1_RST	0x0	see Table 118
RESET_EXT_STAT34	R/W	0x488	Reset external status register 34 for TIMER2_RST	0x0	see Table 118
RESET_EXT_STAT35	R/W	0x48C	Reset external status register 35 for TIMER3_RST	0x0	see Table 118

Table 104. Register overview: RGU (base address: 0x4005 3000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
RESET_EXT_STAT36	R/W	0x490	Reset external status register 36 for RITIMER_RST	0x0	see Table 118
RESET_EXT_STAT37	R/W	0x494	Reset external status register 37 for SCT_RST	0x0	see Table 118
RESET_EXT_STAT38	R/W	0x498	Reset external status register 38 for MOTOCONPWM_RST	0x0	see Table 118
RESET_EXT_STAT39	R/W	0x49C	Reset external status register 39 for QEI_RST	0x0	see Table 118
RESET_EXT_STAT40	R/W	0x4A0	Reset external status register 40 for ADC0_RST	0x0	see Table 118
RESET_EXT_STAT41	R/W	0x4A4	Reset external status register 41 for ADC1_RST	0x0	see Table 118
RESET_EXT_STAT42	R/W	0x4A8	Reset external status register 42 for DAC_RST	0x0	see Table 118
RESET_EXT_STAT43	-	0x4AC	Reserved	0x0	-
RESET_EXT_STAT44	R/W	0x4B0	Reset external status register 44 for UART0_RST	0x0	see Table 118
RESET_EXT_STAT45	R/W	0x4B4	Reset external status register 45 for UART1_RST	0x0	see Table 118
RESET_EXT_STAT46	R/W	0x4B8	Reset external status register 46 for UART2_RST	0x0	see Table 118
RESET_EXT_STAT47	R/W	0x4BC	Reset external status register 47 for UART3_RST	0x0	see Table 118
RESET_EXT_STAT48	R/W	0x4C0	Reset external status register 48 for I2C0_RST	0x0	see Table 118
RESET_EXT_STAT49	R/W	0x4C4	Reset external status register 49 for I2C1_RST	0x0	see Table 118
RESET_EXT_STAT50	R/W	0x4C8	Reset external status register 50 for SSP0_RST	0x0	see Table 118
RESET_EXT_STAT51	R/W	0x4CC	Reset external status register 51 for SSP1_RST	0x0	see Table 118
RESET_EXT_STAT52	R/W	0x4D0	Reset external status register 52 for I2S_RST	0x0	see Table 118
RESET_EXT_STAT53	R/W	0x4D4	Reset external status register 53 for SPIFI_RST	0x0	see Table 118
RESET_EXT_STAT54	R/W	0x4D8	Reset external status register 54 for CAN1_RST	0x0	see Table 118
RESET_EXT_STAT55	R/W	0x4DC	Reset external status register 55 for CAN0_RST	0x0	see Table 118
RESET_EXT_STAT56	R/W	0x4E0	Reset external status register 56 for M0APP_RST	<tbd>	see Table 118
RESET_EXT_STAT57	R/W	0x4E4	Reset external status register 57 for SGPIO_RST	0x0	see Table 118
RESET_EXT_STAT58	R/W	0x4E8	Reset external status register 58 for SPI_RST	0x0	see Table 118
RESET_EXT_STAT59	-	0x4EC	Reserved	-	-

Table 104. Register overview: RGU (base address: 0x4005 3000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
RESET_EXT_STAT60	-	0x4F0	Reserved	-	-
RESET_EXT_STAT61	-	0x4F4	Reserved	-	-
RESET_EXT_STAT62	-	0x4F8	Reserved	-	-
RESET_EXT_STAT63	-	0x4FC	Reserved	-	-

13.4.1 RGU reset control register

The RGU reset control register allows software to activate and clear individual reset outputs. Each bit corresponds to an individual reset output, and writing a one activates that output. The reset output is automatically de-activated after a fixed delay period with the exception of the M0APP_RST. If the reset output has a manual release, it stays activated once pulled low until a 0 is written to the appropriate bit in this register. This applies whether the reset activation came from the Reset Control Register or any other source

Table 105. Reset control register 0 (RESET_CTRL0, address 0x4005 3100) bit description

Bit	Symbol	Description	Reset value	Access
0	CORE_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
1	PERIPH_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after three clock cycles.	0	W
2	MASTER_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after three clock cycles.	0	W
3	-	Reserved	0	-
4	WWDT_RST	Writing a one to this bit has no effect.	0	-
5	CREG_RST	Writing a one to this bit has no effect.	0	-
6	-	Reserved	0	-
7	-	Reserved	0	-
8	BUS_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle. Do not use during normal operation	0	W
9	SCU_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
10	PINMUX_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
11	-	Reserved	0	-
12	-	Reserved	0	-
13	M4_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
14	-	Reserved	0	-
15	-	Reserved	0	-
16	LCD_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
17	USB0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W

Table 105. Reset control register 0 (RESET_CTRL0, address 0x4005 3100) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
18	USB1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
19	DMA_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
20	SDIO_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
21	EMC_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
22	ETHERNET_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
23	-	Reserved	-	-
24	-	Reserved	-	-
25	-	Reserved	-	-
26	-	Reserved	-	-
27	-	Reserved	-	-
28	GPIO_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
29	-	Reserved	-	-
30	-	Reserved	-	-
31	-	Reserved	-	-

Table 106. Reset control register 1 (RESET_CTRL1, address 0x4005 3104) bit description

Bit	Symbol	Description	Reset value	Access
0	TIMER0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
1	TIMER1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
2	TIMER2_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
3	TIMER3_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
4	RITIMER_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
5	SCT_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
6	MOTOCONPWM_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
7	QEI_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
8	ADC0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
9	ADC1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W

Table 106. Reset control register 1 (RESET_CTRL1, address 0x4005 3104) bit description

…continued

Bit	Symbol	Description	Reset value	Access
10	DAC_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
11	-	Reserved	-	-
12	UART0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
13	UART1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
14	UART2_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
15	UART3_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
16	I2C0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
17	I2C1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
18	SSP0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
19	SSP1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
20	I2S_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
21	SPIFI_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
22	CAN1_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
23	CAN0_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
24	M0APP_RST	Writing a one activates the reset. Writing a 0 clears the reset. **This bit must be cleared by software.**	1	W
25	SGPIO_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
26	SPI_RST	Writing a one activates the reset. This bit is automatically cleared to 0 after one clock cycle.	0	W
27	-	Reserved	-	-
28	-	Reserved	-	-
29	-	Reserved	-	-
30	-	Reserved	-	-
31	-	Reserved	-	-

13.4.2 RGU reset status register

The reset status register shows which source (if any) caused the last reset activation per individual reset output of the RGU. When one (or more) inputs of the RGU caused the Reset Output to go active (indicated by value 01), the corresponding RESET_EXT_STATUS register can be read, see Section 13.4.4.

The RESET_STATUS registers are cleared by writing 0 to each of the status bits.

Table 107. Reset status register 0 (RESET_STATUS0, address 0x4005 3110) bit description

Bit	Symbol	Description	Reset value	Access
1:0	CORE_RST	Status of the CORE_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	00	R/W
3:2	PERIPH_RST	Status of the PERIPH_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator - this reset is self-clearing 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	00	R/W
5:4	MASTER_RST	Status of the MASTER_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator - this reset is self-clearing 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
7:6	-	Reserved	01	
9:8	WWDT_RST	Status of the WWDT_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reserved	00	R/W
11:10	CREG_RST	Status of the CREG_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reserved	00	R/W
13:12	-	Reserved	01	-
15:14	-	Reserved	01	-
17:16	BUS_RST	Status of the BUS_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
19:18	SCU_RST	Status of the SCU_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
21:20	-	Reserved	01	-

Table 107. Reset status register 0 (RESET_STATUS0, address 0x4005 3110) bit description

Bit	Symbol	Description	Reset value	Access
23:22	-	Reserved	01	-
25:24	-	Reserved	01	-
27:26	M4_RST	Status of the M4_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
29:28	-	Reserved	01	-
31:30	-	Reserved	01	-

Table 108. Reset status register 1 (RESET_STATUS1, address 0x4005 3114) bit description

Bit	Symbol	Description	Reset value	Access
1:0	LCD_RST	Status of the LCD_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
3:2	USB0_RST	Status of the USB0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
5:4	USB1_RST	Status of the USB1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
7:6	DMA_RST	Status of the DMA_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
9:8	SDIO_RST	Status of the SDIO_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W

Table 108. Reset status register 1 (RESET_STATUS1, address 0x4005 3114) bit description

…continued

Bit	Symbol	Description	Reset value	Access
11:10	EMC_RST	Status of the EMC_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
13:12	ETHERNET_RST	Status of the ETHERNET_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
15:14	-	Reserved	01	-
17:16	-	Reserved	01	-
19:18	-	Reserved	01	-
21:20	-	Reserved	01	-
23:22	-	Reserved	01	-
25:24	GPIO_RST	Status of the GPIO_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
27:26	-	Reserved	01	-
29:28	-	Reserved	01	-
31:30	-	Reserved	01	-

Table 109. Reset status register 2 (RESET_STATUS2, address 0x4005 3118) bit description

Bit	Symbol	Description	Reset value	Access
1:0	TIMER0_RST	Status of the TIMER0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
3:2	TIMER1_RST	Status of the TIMER1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
5:4	TIMER2_RST	Status of the TIMER2_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
7:6	TIMER3_RST	Status of the TIMER3_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
9:8	RITIMER_RST	Status of the RITIMER_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
11:10	SCT_RST	Status of the SCT_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
13:12	MOTOCONPWM_RST	Status of the MOTOCONPWM_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
15:14	QEI_RST	Status of the QEI_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W

Table 109. Reset status register 2 (RESET_STATUS2, address 0x4005 3118) bit description *…continued*

…continued

Bit	Symbol	Description	Reset value	Access
17:16	ADC0_RST	Status of the ADC0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
19:18	ADC1_RST	Status of the ADC1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
21:20	DAC_RST	Status of the DAC_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
23:22	-	Reserved	01	R/W
25:24	UART0_RST	Status of the UART0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
27:26	UART1_RST	Status of the UART1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
29:28	UART2_RST	Status of the UART2_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
31:30	UART3_RST	Status of the UART3_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W

Table 110. Reset status register 3 (RESET_STATUS3, address 0x4005 311C) bit description

Bit	Symbol	Description	Reset value	Access
1:0	I2C0_RST	Status of the I2C0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
3:2	I2C1_RST	Status of the I2C1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
5:4	SSP0_RST	Status of the SSP0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
7:6	SSP1_RST	Status of the SSP1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
9:8	I2S_RST	Status of the I2S_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
11:10	SPIFI_RST	Status of the SPIFI_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
13:12	CAN1_RST	Status of the CAN1_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
15:14	CAN0_RST	Status of the CAN0_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W

Table 110. Reset status register 3 (RESET_STATUS3, address 0x4005 311C) bit description
…continued

Bit	Symbol	Description	Reset value	Access
17:16	M0APP_RST	Status of the M0APP_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	11	R/W
19:18	SGPIO_RST	Status of the SGPIO_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
21:20	SPI_RST	Status of the SPI_RST reset generator output 00 = No reset activated 01 = Reset output activated by input to the reset generator 10 = Reserved 11 = Reset output activated by software write to RESET_CTRL register	01	R/W
23:22	-	Reserved	01	-
25:24	-	Reserved	01	-
27:26	-	Reserved	01	-
29:28	-	Reserved	01	-
31:30	-	Reserved	01	-

13.4.3 RGU reset active status register

The reset active status register shows the current value of the reset outputs of the RGU. Note that the resets are active LOW.

Table 111. Reset active status register 0 (RESET_ACTIVE_STATUS0, address 0x4005 3150) bit description

Bit	Symbol	Description	Reset value	Access
0	CORE_RST	Current status of the CORE_RST 0 = Reset asserted 1 = No reset	0	R
1	PERIPH_RST	Current status of the PERIPH_RST 0 = Reset asserted 1 = No reset	0	R
2	MASTER_RST	Current status of the MASTER_RST 0 = Reset asserted 1 = No reset	0	R
3	-	Reserved	0	
4	WWDT_RST	Current status of the WWDT_RS 0 = Reset asserted 1 = No reset	0	R

Table 111. Reset active status register 0 (RESET_ACTIVE_STATUS0, address 0x4005 3150) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
5	CREG_RST	Current status of the CREG_RST 0 = Reset asserted 1 = No reset	0	R
6	-	Reserved	0	
7	-	Reserved	0	
8	BUS_RST	Current status of the BUS_RST 0 = Reset asserted 1 = No reset	0	R
9	SCU_RST	Current status of the SCU_RST 0 = Reset asserted 1 = No reset	0	R
10	PINMUX_RST	Current status of the PINMUX_RST 0 = Reset asserted 1 = No reset	0	R
11	-	Reserved	0	-
12	-	Reserved	0	-
13	M4_RST	Current status of the M4_RST 0 = Reset asserted 1 = No reset	0	R
14	-	Reserved	0	
15	-	Reserved	0	
16	LCD_RST	Current status of the LCD_RST 0 = Reset asserted 1 = No reset	0	R
17	USB0_RST	Current status of the USB0_RST 0 = Reset asserted 1 = No reset	0	R
18	USB1_RST	Current status of the USB1_RST 0 = Reset asserted 1 = No reset	0	R
19	DMA_RST	Current status of the DMA_RST 0 = Reset asserted 1 = No reset	0	R
20	SDIO_RST	Current status of the SDIO_RST 0 = Reset asserted 1 = No reset	0	R
21	EMC_RST	Current status of the EMC_RST 0 = Reset asserted 1 = No reset	0	R

Table 111. Reset active status register 0 (RESET_ACTIVE_STATUS0, address 0x4005 3150) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
22	ETHERNET_RST	Current status of the ETHERNET_RST 0 = Reset asserted 1 = No reset	0	R
23	-	Reserved	-	-
24	-	Reserved	-	-
25	-	Reserved	-	-
26	-	Reserved	-	-
27	-	Reserved	-	-
28	GPIO_RST	Current status of the GPIO_RST 0 = Reset asserted 1 = No reset	0	R
29	-	Reserved	-	-
30	-	Reserved	-	-
31	-	Reserved	-	-

Table 112. Reset active status register 1 (RESET_ACTIVE_STATUS1, address 0x4005 3154) bit description

Bit	Symbol	Description	Reset value	Access
0	TIMER0_RST	Current status of the TIMER0_RST 0 = Reset asserted 1 = No reset	0	R
1	TIMER1_RST	Current status of the TIMER1_RST 0 = Reset asserted 1 = No reset	0	R
2	TIMER2_RST	Current status of the TIMER2_RST 0 = Reset asserted 1 = No reset	0	R
3	TIMER3_RST	Current status of the TIMER3_RST 0 = Reset asserted 1 = No reset	0	R
4	RITIMER_RST	Current status of the RITIMER_RST 0 = Reset asserted 1 = No reset	0	R
5	SCT_RST	Current status of the SCT_RST 0 = Reset asserted 1 = No reset	0	R
6	MOTOCONPWM_RST	Current status of the MOTOCONPWM_RST 0 = Reset asserted 1 = No reset	0	R

Table 112. Reset active status register 1 (RESET_ACTIVE_STATUS1, address 0x4005 3154) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
7	QEI_RST	Current status of the QEI_RST 0 = Reset asserted 1 = No reset	0	R
8	ADC0_RST	Current status of the ADC0_RST 0 = Reset asserted 1 = No reset	0	R
9	ADC1_RST	Current status of the ADC1_RST 0 = Reset asserted 1 = No reset	0	R
10	DAC_RST	Current status of the DAC_RST 0 = Reset asserted 1 = No reset	0	R
11	-		-	-
12	UART0_RST	Current status of the UART0_RST 0 = Reset asserted 1 = No reset	0	R
13	UART1_RST	Current status of the UART1_RST 0 = Reset asserted 1 = No reset	0	R
14	UART2_RST	Current status of the UART2_RST 0 = Reset asserted 1 = No reset	0	R
15	UART3_RST	Current status of the UART3_RST 0 = Reset asserted 1 = No reset	0	R
16	I2C0_RST	Current status of the I2C0_RST 0 = Reset asserted 1 = No reset	0	R
17	I2C1_RST	Current status of the I2C1_RST 0 = Reset asserted 1 = No reset	0	R
18	SSP0_RST	Current status of the SSP0_RST 0 = Reset asserted 1 = No reset	0	R
19	SSP1_RST	Current status of the SSP1_RST 0 = Reset asserted 1 = No reset	0	R
20	I2S_RST	Current status of the I2S_RST 0 = Reset asserted 1 = No reset	0	R

Table 112. Reset active status register 1 (RESET_ACTIVE_STATUS1, address 0x4005 3154) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
21	SPIFI_RST	Current status of the SPIFI_RST 0 = Reset asserted 1 = No reset	0	R
22	CAN1_RST	Current status of the CAN1_RST 0 = Reset asserted 1 = No reset	0	R
23	CAN0_RST	Current status of the CAN0_RST 0 = Reset asserted 1 = No reset	0	R
24	M0APP_RST	Current status of the M0APP_RST 0 = Reset asserted 1 = No reset	0	R
25	SGPIO_RST	Current status of the SGPIO_RST 0 = Reset asserted 1 = No reset	0	R
26	SPI_RST	Current status of the SPI_RST 0 = Reset asserted 1 = No reset	0	R
27	-	Reserved.	-	-
28	-	Reserved.	-	-
29	-	Reserved.	-	-
30	-	Reserved.	-	-
31	-	Reserved.	-	-

13.4.4 Reset external status registers

The external status registers indicate which input to the reset generator caused the reset output to go active. Any bit set to 1 in the Reset external status register should be cleared to 0 after a read operation to allow the detection of the next reset.

All reset generators except the WWDT time-out reset, the BOD reset, the reset signal from the PMU, and the software reset, which have no inputs, have an associated external status register. The CORE_RST reset generator has three possible inputs (the WWDT time-out reset, the BOD reset, and the PMU), and which input caused the reset is indicated in the external status register. All other reset generators have only one input which, depending on the hierarchy, can be either the CORE_RST, the PERIPHERAL_RST, or the MASTER_RST.

Note that the external status register does not show whether or not the reset was activated by a software reset. The software reset is indicated in the reset status registers 0 to 3 (see Table 107 to Table 110).

13.4.4.1 Reset external status register 0 for CORE_RST

This register shows whether or not any of the inputs to the CORE_RST reset generator has activated the CORE_RST. The CORE_RST can be activated by the external reset pin, a WWDT time-out, a BOD reset or by writing to bit 0 of the RESET_CTRL0 register.

Table 113. Reset external status register 0 (RESET_EXT_STAT0, address 0x4005 3400) bit description

Bit	Symbol	Description	Reset value	Access
0	EXT_RESET	Reset activated by external reset from reset pin. Write 0 to clear. 0 = Reset not activated by reset pin 1 = Reset activated	0	R/W
1	-	Reserved. Do not modify; read as logic 0.	0	-
2	-	Reserved. Do not modify; read as logic 0.	0	-
3	-	Reserved. Do not modify; read as logic 0.	0	-
4	BOD_RESET	Reset activated by BOD reset. Write 0 to clear. 0 = Reset not activated by BOD 1 = Reset activated	0	R/W
5	WWDT_RESET	Reset activated by WWDT time-out. Write 0 to clear. 0 = Reset not activated by WWDT 1 = Reset activated	0	R/W
31:6	-	Reserved. Do not modify; read as logic 0.	0	-

13.4.4.2 Reset external status register 1 for PERIPH_RST

This register shows whether or not the CORE_RST output has activated the PERIPH_RST. A reset generated from the CORE_RST is the only possible reset source for the PERIPH_RST aside from a software reset by writing to the RESET_CTRL register.

Table 114. Reset external status register 1 (RESET_EXT_STAT1, address 0x4005 3404) bit description

Bit	Symbol	Description	Reset value	Access
0	-	Reserved. Do not modify; read as logic 0.	0	-
1	CORE_RESET	Reset activated by CORE_RST output. Write 0 to clear. 0 = Reset not activated 1 = Reset activated	0	R/W
31:2	-	Reserved. Do not modify; read as logic 0.	0	-

13.4.4.3 Reset external status register 2 for MASTER_RST

Table 115. Reset external status register 2 (RESET_EXT_STAT2, address 0x4005 3408) bit description

Bit	Symbol	Description	Reset value	Access
1:0	-	Reserved. Do not modify; read as logic 0.	0	-
2	PERIPHERAL_RESET	Reset activated by PERIPHERAL_RST output. Write 0 to clear. 0 = Reset not activated 1 = Reset activated	0	R/W
31:3	-	Reserved. Do not modify; read as logic 0.	0	-

13.4.4.4 Reset external status register 4 for WWDT_RST

Table 116. Reset external status register 4 (RESET_EXT_STAT4, address 0x4005 3410) bit description

Bit	Symbol	Description	Reset value	Access
0	-	Reserved. Do not modify; read as logic 0.	0	-
1	CORE_RESET	Reset activated by CORE_RST output. Write 0 to clear. 0 = Reset not activated 1 = Reset activated	0	R/W
31:2	-	Reserved. Do not modify; read as logic 0.	0	-

13.4.4.5 Reset external status register 5 for CREG_RST

Table 117. Reset external status register 5 (RESET_EXT_STAT5, address 0x4005 3414) bit description

Bit	Symbol	Description	Reset value	Access
0	-	Reserved. Do not modify; read as logic 0.	0	-
1	CORE_RESET	Reset activated by CORE_RST output. Write 0 to clear. 0 = Reset not activated 1 = Reset activated	0	R/W
31:2	-	Reserved. Do not modify; read as logic 0.	0	-

13.4.4.6 Reset external status registers for PERIPHERAL_RESET

Refer to Table 104 for reset generators which have the PERIPH_RST output as reset source.

Table 118. Reset external status registers x (RESET_EXT_STATx, address 0x4005 34xx) bit description

Bit	Symbol	Description	Reset value	Access
1:0	-	Reserved. Do not modify; read as logic 0.	0	-
2	PERIPHERAL_RESET	Reset activated by PERIPHERAL_RST output. Write 0 to clear. 0 = Reset not activated 1 = Reset activated	0	R/W
31:3	-	Reserved. Do not modify; read as logic 0.	0	-

13.4.4.7 Reset external status registers for MASTER_RESET

Refer to Table 104 for reset generators which have the MASTER_RST output as reset source. These are the ARM Cortex-M4 core, the LCD controller, the USB0, the GPDMA, the SDIO controller, the external memory controller, and the Ethernet controller.

The reset value is dependent on the peripheral, see Table 104.

Table 119. Reset external status registers y (RESET_EXT_STATy, address 0x4005 34yy) bit description

Bit	Symbol	Description	Reset value	Access
2:0	-	Reserved. Do not modify; read as logic 0.	0	-
3	MASTER_RESET	Reset activated by MASTER_RST output. Write 0 to clear. 0 = Reset not activated 1 = Reset activated	0	R/W
31:4	-	Reserved. Do not modify; read as logic 0.	0	-

14.1 How to read this chapter

This chapter applies to all parts.

14.2 Pin description

On the LPC43xx, digital pins are grouped into 16 pin groups, named P0 to P9 and PA to PF, with up to 20 pins used per group. Each digital pin may support up to eight different digital pin functions, including General Purpose I/O (GPIO), selectable through the SCU pin configuration registers. Some digital pins support an additional analog function selectable through the ENAIO registers in the SCU.

Remark: Note that the pin name is not indicative of the GPIO port assigned to it.

Table 120. Pin description

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
Multiplexed digital pins										
P0_0	L3	x	G2	47	32	22	[3]	I; PU	I/O	**GPIO0[0] —** General purpose digital input/output pin.
									I/O	**SSP1_MISO —** Master In Slave Out for SSP1.
									I	**ENET_RXD1 —** Ethernet receive data 1 (RMII/MII interface).
									I/O	**SGPIO0 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**I2S0_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I²S-bus specification*.
									I/O	**I2S1_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I²S-bus specification*.
P0_1	M2	x	G1	50	34	23	[3]	I; PU	I/O	**GPIO0[1] —** General purpose digital input/output pin.
									I/O	**SSP1_MOSI —** Master Out Slave in for SSP1.
									I	**ENET_COL —** Ethernet Collision detect (MII interface).
									I/O	**SGPIO1 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
										ENET_TX_EN — Ethernet transmit enable (RMII/MII interface).
									I/O	**I2S1_TX_SDA —** I2S1 transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I²S-bus specification*.
P1_0	P2	x	H1	54	38	25	[3]	I; PU	I/O	**GPIO0[4] —** General purpose digital input/output pin.
									I	**CTIN_3 —** SCT input 3. Capture input 1 of timer 1.
									I/O	**EMC_A5 —** External memory address line 5.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SSP0_SSEL —** Slave Select for SSP0.
									I/O	**SGPIO7 —** General purpose digital input/output pin.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state [2]	Type	Description
P1_1	R2	x	K2	58	42	28	[3]	I; PU	I/O	**GPIO0[8] —** General purpose digital input/output pin. Boot pin (see Table 16).
									O	**CTOUT_7 —** SCT output 7. Match output 3 of timer 1.
									I/O	**EMC_A6 —** External memory address line 6.
									I/O	**SGPIO8 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SSP0_MISO —** Master In Slave Out for SSP0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P1_2	R3	x	K1	60	43	29	[3]	I; PU	I/O	**GPIO0[9] —** General purpose digital input/output pin. Boot pin (see Table 16).
									O	**CTOUT_6 —** SCT output 6. Match output 2 of timer 1.
									I/O	**EMC_A7 —** External memory address line 7.
									I/O	**SGPIO9 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SSP0_MOSI —** Master Out Slave in for SSP0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P1_3	P5	x	J1	61	44	30	[3]	I; PU	I/O	**GPIO0[10] —** General purpose digital input/output pin.
									O	**CTOUT_8 —** SCT output 8. Match output 0 of timer 2.
									I/O	**SGPIO10 —** General purpose digital input/output pin.
									O	**$\overline{\text{EMC\_OE}}$ —** LOW active Output Enable signal.
									O	**USB0_IND1 —** USB0 port indicator LED control output 1.
									I/O	**SSP1_MISO —** Master In Slave Out for SSP1.
									-	**R —** Function reserved.
									O	**SD_RST —** SD/MMC reset signal for MMC4.4 card.
P1_4	T3	x	J2	64	47	32	[3]	I; PU	I/O	**GPIO0[11] —** General purpose digital input/output pin.
									O	**CTOUT_9 —** SCT output 9. Match output 1 of timer 2.
									I/O	**SGPIO11 —** General purpose digital input/output pin.
									O	**$\overline{\text{EMC\_BLS0}}$ —** LOW active Byte Lane select signal 0.
									O	**USB0_IND0 —** USB0 port indicator LED control output 0.
									I/O	**SSP1_MOSI —** Master Out Slave in for SSP1.
									-	**R —** Function reserved.
									O	**SD_VOLT1 —** SD/MMC bus voltage select output 1.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P1_5	R5	x	J4	65	48	33	[3]	I; PU	I/O	**GPIO1[8] —** General purpose digital input/output pin.
									O	**CTOUT_10 —** SCT output 10. Match output 2 of timer 2.
									-	**R —** Function reserved.
									O	**EMC_CS0 —** LOW active Chip Select 0 signal.
									O	**USB0_PWR_FAULT —** Port power fault signal indicating overcurrent condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
									I/O	**SSP1_SSEL —** Slave Select for SSP1.
									I/O	**SGPIO15 —** General purpose digital input/output pin.
									O	**SD_POW —** SD/MMC power monitor output.
P1_6	T4	x	K4	67	49	34	[3]	I; PU	I/O	**GPIO1[9] —** General purpose digital input/output pin.
									I	**CTIN_5 —** SCT input 5. Capture input 2 of timer 2.
									-	**R —** Function reserved.
									O	**EMC_WE —** LOW active Write Enable signal.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO14 —** General purpose digital input/output pin.
									I/O	**SD_CMD —** SD/MMC command signal.
P1_7	T5	x	G4	69	50	35	[3]	I; PU	I/O	**GPIO1[0] —** General purpose digital input/output pin.
									I	**U1_DSR —** Data Set Ready input for UART1.
									O	**CTOUT_13 —** SCT output 13. Match output 1 of timer 3.
									I/O	**EMC_D0 —** External memory data line 0.
									O	**USB0_PPWR —** VBUS drive signal (towards external charge pump or power management unit); indicates that VBUS must be driven (active HIGH). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the USB_PPWR used on other NXP LPC parts.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P1_8	R7	x	H5	71	51	36	[3]	I; PU	I/O	**GPIO1[1] —** General purpose digital input/output pin.
									O	**U1_DTR —** Data Terminal Ready output for UART1.
									O	**CTOUT_12 —** SCT output 12. Match output 0 of timer 3.
									I/O	**EMC_D1 —** External memory data line 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**SD_VOLT0 —** SD/MMC bus voltage select output 0.
P1_9	T7	x	J5	73	52	37	[3]	I; PU	I/O	**GPIO1[2] —** General purpose digital input/output pin.
									O	**U1_RTS —** Request to Send output for UART1.
									O	**CTOUT_11 —** SCT output 11. Match output 3 of timer 2.
									I/O	**EMC_D2 —** External memory data line 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_DAT0 —** SD/MMC data bus line 0.
P1_10	R8	x	H6	75	53	38	[3]	I; PU	I/O	**GPIO1[3] —** General purpose digital input/output pin.
									I	**U1_RI —** Ring Indicator input for UART1.
									O	**CTOUT_14 —** SCT output 14. Match output 2 of timer 3.
									I/O	**EMC_D3 —** External memory data line 3.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_DAT1 —** SD/MMC data bus line 1.
P1_11	T9	x	J7	77	55	39	[3]	I; PU	I/O	**GPIO1[4] —** General purpose digital input/output pin.
									I	**U1_CTS —** Clear to Send input for UART1.
									O	**CTOUT_15 —** SCT output 15. Match output 3 of timer 3.
									I/O	**EMC_D4 —** External memory data line 4.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_DAT2 —** SD/MMC data bus line 2.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P1_12	R9	x	K7	78	56	40	[3]	I; PU	I/O	**GPIO1[5] —** General purpose digital input/output pin.
									I	**U1_DCD —** Data Carrier Detect input for UART1.
									-	**R —** Function reserved.
									I/O	**EMC_D5 —** External memory data line 5.
									I	**T0_CAP1 —** Capture input 1 of timer 0.
									-	**R —** Function reserved.
									I/O	**SGPIO8 —** General purpose digital input/output pin.
									I/O	**SD_DAT3 —** SD/MMC data bus line 3.
P1_13	R10	x	H8	83	60	41	[3]	I; PU	I/O	**GPIO1[6] —** General purpose digital input/output pin.
									O	**U1_TXD —** Transmitter output for UART1.
									-	**R —** Function reserved.
									I/O	**EMC_D6 —** External memory data line 6.
									I	**T0_CAP0 —** Capture input 0 of timer 0.
									-	**R —** Function reserved.
									I/O	**SGPIO9 —** General purpose digital input/output pin.
									I	**SD_CD —** SD/MMC card detect input.
P1_14	R11	x	J8	85	61	42	[3]	I; PU	I/O	**GPIO1[7] —** General purpose digital input/output pin.
									I	**U1_RXD —** Receiver input for UART1.
									-	**R —** Function reserved.
									I/O	**EMC_D7 —** External memory data line 7.
									O	**T0_MAT2 —** Match output 2 of timer 0.
									-	**R —** Function reserved.
									I/O	**SGPIO10 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
P1_15	T12	x	K8	87	62	43	[3]	I; PU	I/O	**GPIO0[2] —** General purpose digital input/output pin.
									O	**U2_TXD —** Transmitter output for USART2.
									I/O	**SGPIO2 —** General purpose digital input/output pin.
									I	**ENET_RXD0 —** Ethernet receive data 0 (RMII/MII interface).
									O	**T0_MAT1 —** Match output 1 of timer 0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P1_16	M7	x	H9	90	64	44	[3]	I; PU	I/O	**GPIO0[3] —** General purpose digital input/output pin.
									I	**U2_RXD —** Receiver input for USART2.
									I/O	**SGPIO3 —** General purpose digital input/output pin.
									I	**ENET_CRS —** Ethernet Carrier Sense (MII interface).
									O	**T0_MAT0 —** Match output 0 of timer 0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**ENET_RX_DV —** Ethernet Receive Data Valid (RMII/MII interface).
P1_17	M8	x	H10	93	66	45	[4]	I; PU	I/O	**GPIO0[12] —** General purpose digital input/output pin.
									I/O	**U2_UCLK —** Serial clock input/output for USART2 in synchronous mode.
									-	**R —** Function reserved.
									I/O	**ENET_MDIO —** Ethernet MIIM data input and output.
									I	**T0_CAP3 —** Capture input 3 of timer 0.
									O	**CAN1_TD —** CAN1 transmitter output.
									I/O	**SGPIO11 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
P1_18	N12	x	J10	95	67	46	[3]	I; PU	I/O	**GPIO0[13] —** General purpose digital input/output pin.
									I/O	**U2_DIR —** RS-485/EIA-485 output enable/direction control for USART2.
									-	**R —** Function reserved.
									O	**ENET_TXD0 —** Ethernet transmit data 0 (RMII/MII interface).
									O	**T0_MAT3 —** Match output 3 of timer 0.
									I	**CAN1_RD —** CAN1 receiver input.
									I/O	**SGPIO12 —** General purpose digital input/output pin.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P1_19	M11	x	K9	96	68	47	[3]	I; PU	I	**ENET_TX_CLK (ENET_REF_CLK) —** Ethernet Transmit Clock (MII interface) or Ethernet Reference Clock (RMII interface).
									I/O	**SSP1_SCK —** Serial clock for SSP1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**CLKOUT —** Clock output pin.
									-	**R —** Function reserved.
									O	**I2S0_RX_MCLK —** I2S receive master clock.
									I/O	**I2S1_TX_SCK —** Transmit Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I^2S-bus specification.
P1_20	M10	x	K10	100	70	48	[3]	I; PU	I/O	**GPIO0[15] —** General purpose digital input/output pin.
									I/O	**SSP1_SSEL —** Slave Select for SSP1.
									-	**R —** Function reserved.
									O	**ENET_TXD1 —** Ethernet transmit data 1 (RMII/MII interface).
									I	**T0_CAP2 —** Capture input 2 of timer 0.
									-	**R —** Function reserved.
									I/O	**SGPIO13 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
P2_0	T16	x	G10	108	75	50	[3]	I; PU	I/O	**SGPIO4 —** General purpose digital input/output pin.
									O	**U0_TXD —** Transmitter output for USART0.
									I/O	**EMC_A13 —** External memory address line 13.
									O	**USB0_PPWR —** VBUS drive signal (towards external charge pump or power management unit); indicates that VBUS must be driven (active HIGH). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the $\overline{\text{USB\_PPWR}}$ used on other NXP LPC parts.
									I/O	**GPIO5[0] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**T3_CAP0 —** Capture input 0 of timer 3.
									O	**ENET_MDC —** Ethernet MIIM clock.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state [2]	Type	Description
P2_1	N15	x	G7	116	81	54	[3]	I; PU	I/O	**SGPIO5 —** General purpose digital input/output pin.
									I	**U0_RXD —** Receiver input for USART0.
									I/O	**EMC_A12 —** External memory address line 12.
									O	**USB0_PWR_FAULT —** Port power fault signal indicating overcurrent condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
									I/O	**GPIO5[1] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**T3_CAP1 —** Capture input 1 of timer 3.
									-	**R —** Function reserved.
P2_2	M15	x	F5	121	84	56	[3]	I; PU	I/O	**SGPIO6 —** General purpose digital input/output pin.
									I/O	**U0_UCLK —** Serial clock input/output for USART0 in synchronous mode.
									I/O	**EMC_A11 —** External memory address line 11.
									O	**USB0_IND1 —** USB0 port indicator LED control output 1.
									I/O	**GPIO5[2] —** General purpose digital input/output pin.
									I	**CTIN_6 —** SCT input 6. Capture input 1 of timer 3.
									I	**T3_CAP2 —** Capture input 2 of timer 3.
									-	**R —** Function reserved.
P2_3	J12	x	D8	127	87	57	[4]	I; PU	I/O	**SGPIO12 —** General purpose digital input/output pin.
									I/O	**I2C1_SDA —** I^2C1 data input/output (this pin does not use a specialized I^2C pad).
									O	**U3_TXD —** Transmitter output for USART3.
									I	**CTIN_1 —** SCT input 1. Capture input 1 of timer 0. Capture input 1 of timer 2.
									I/O	**GPIO5[3] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT0 —** Match output 0 of timer 3.
									O	**USB0_PPWR —** VBUS drive signal (towards external charge pump or power management unit); indicates that VBUS must be driven (active HIGH). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the USB_PPWR used on other NXP LPC parts.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P2_4	K11	x	D9	128	88	58	[4]	I; PU	I/O	**SGPIO13 —** General purpose digital input/output pin.
									I/O	**I2C1_SCL —** I^2C1 clock input/output (this pin does not use a specialized I^2C pad).
									I	**U3_RXD —** Receiver input for USART3.
									I	**CTIN_0 —** SCT input 0. Capture input 0 of timer 0, 1, 2, 3.
									I/O	**GPIO5[4] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT1 —** Match output 1 of timer 3.
									O	**USB0_PWR_FAULT —** Port power fault signal indicating overcurrent condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
P2_5	K14	x	D10	131	91	61	[4]	I; PU	I/O	**SGPIO14 —** General purpose digital input/output pin.
									I	**CTIN_2 —** SCT input 2. Capture input 2 of timer 0.
									I	**USB1_VBUS —** Monitors the presence of USB1 bus power. **Note:** This signal must be HIGH for USB reset to occur.
									I	**ADCTRIG1 —** ADC trigger input 1.
									I/O	**GPIO5[5] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT2 —** Match output 2 of timer 3.
									O	**USB0_IND0 —** USB0 port indicator LED control output 0.
P2_6	K16	x	G9	137	95	64	[3]	I; PU	I/O	**SGPIO7 —** General purpose digital input/output pin.
									I/O	**U0_DIR —** RS-485/EIA-485 output enable/direction control for USART0.
									I/O	**EMC_A10 —** External memory address line 10.
									O	**USB0_IND0 —** USB0 port indicator LED control output 0.
									I/O	**GPIO5[6] —** General purpose digital input/output pin.
									I	**CTIN_7 —** SCT input 7.
									I	**T3_CAP3 —** Capture input 3 of timer 3.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P2_7	H14	x	C10	138	96	65	[3]	I; PU	I/O	**GPIO0[7] —** General purpose digital input/output pin. If this pin is pulled LOW at reset, the part enters ISP mode using USART0.
									O	**CTOUT_1 —** SCT output 1. Match output 1 of timer 0.
									I/O	**U3_UCLK —** Serial clock input/output for USART3 in synchronous mode.
									I/O	**EMC_A9 —** External memory address line 9.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**T3_MAT3 —** Match output 3 of timer 3.
									-	**R —** Function reserved.
P2_8	J16	x	C6	140	98	67	[3]	I; PU	I/O	**SGPIO15 —** General purpose digital input/output pin. Boot pin (see Table 16).
									O	**CTOUT_0 —** SCT output 0. Match output 0 of timer 0.
									I/O	**U3_DIR —** RS-485/EIA-485 output enable/direction control for USART3.
									I/O	**EMC_A8 —** External memory address line 8.
									I/O	**GPIO5[7] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P2_9	H16	x	B10	144	102	70	[3]	I; PU	I/O	**GPIO1[10] —** General purpose digital input/output pin. Boot pin (see Table 16).
									O	**CTOUT_3 —** SCT output 3. Match output 3 of timer 0.
									I/O	**U3_BAUD —** Baud pin for USART3.
									I/O	**EMC_A0 —** External memory address line 0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P2_10	G16	x	E8	146	104	71	[3]	I; PU	I/O	**GPIO0[14] —** General purpose digital input/output pin.
									O	**CTOUT_2 —** SCT output 2. Match output 2 of timer 0.
									O	**U2_TXD —** Transmitter output for USART2.
									I/O	**EMC_A1 —** External memory address line 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state [2]	Type	Description
P2_11	F16	x	A9	148	105	72	[3]	I; PU	I/O	**GPIO1[11] —** General purpose digital input/output pin.
									O	**CTOUT_5 —** SCT output 5. Match output 1 of timer 1.
									I	**U2_RXD —** Receiver input for USART2.
									I/O	**EMC_A2 —** External memory address line 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P2_12	E15	x	B9	153	106	73	[3]	I; PU	I/O	**GPIO1[12] —** General purpose digital input/output pin.
									O	**CTOUT_4 —** SCT output 4. Match output 0 of timer 1.
									-	**R —** Function reserved.
									I/O	**EMC_A3 —** External memory address line 3.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**U2_UCLK —** Serial clock input/output for USART2 in synchronous mode.
P2_13	C16	x	A10	156	108	75	[3]	I; PU	I/O	**GPIO1[13] —** General purpose digital input/output pin.
									I	**CTIN_4 —** SCT input 4. Capture input 2 of timer 1.
									-	**R —** Function reserved.
									I/O	**EMC_A4 —** External memory address line 4.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**U2_DIR —** RS-485/EIA-485 output enable/direction control for USART2.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P3_0	F13	x	A8	161	112	78	[3]	I; PU	I/O	**I2S0_RX_SCK —** I2S receive clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the *I²S-bus specification*.
									O	**I2S0_RX_MCLK —** I2S receive master clock.
									I/O	**I2S0_TX_SCK —** Transmit Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I²S-bus specification.
									O	**I2S0_TX_MCLK —** I2S transmit master clock.
									I/O	**SSP0_SCK —** Serial clock for SSP0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P3_1	G11	x	F7	163	114	79	[3]	I; PU	I/O	**I2S0_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I²S-bus specification*.
									I/O	**I2S0_RX_WS —** Receive Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I²S-bus specification*.
									I	**CAN0_RD —** CAN receiver input.
									O	**USB1_IND1 —** USB1 Port indicator LED control output 1.
									I/O	**GPIO5[8] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**LCD_VD15 —** LCD data.
									-	**R —** Function reserved.
P3_2	F11	x	G6	166	116	80	[3]	I; PU	I/O	**I2S0_TX_SDA —** I2S transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I²S-bus specification*.
									I/O	**I2S0_RX_SDA —** I2S Receive data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I²S-bus specification*.
									O	**CAN0_TD —** CAN transmitter output.
									O	**USB1_IND0 —** USB1 Port indicator LED control output 0.
									I/O	**GPIO5[9] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**LCD_VD14 —** LCD data.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P3_3	B14	x	A7	169	118	81	[5]	I; PU	-	**R —** Function reserved.
									I/O	**SPI_SCK —** Serial clock for SPI.
									I/O	**SSP0_SCK —** Serial clock for SSP0.
									O	**SPIFI_SCK —** Serial clock for SPIFI.
									O	**CGU_OUT1 —** CGU spare clock output 1.
									-	**R —** Function reserved.
									O	**I2S0_TX_MCLK —** I2S transmit master clock.
									I/O	**I2S1_TX_SCK —** Transmit Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I^2S-bus specification.
P3_4	A15	x	B8	171	119	82	[3]	I; PU	I/O	**GPIO1[14] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SPIFI_SIO3 —** I/O lane 3 for SPIFI.
									O	**U1_TXD —** Transmitter output for UART 1.
									I/O	**I2S0_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									I/O	**I2S1_RX_SDA —** I2S1 Receive data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									O	**LCD_VD13 —** LCD data.
P3_5	C12	x	B7	173	121	84	[3]	I; PU	I/O	**GPIO1[15] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SPIFI_SIO2 —** I/O lane 2 for SPIFI.
									I	**U1_RXD —** Receiver input for UART 1.
									I/O	**I2S0_TX_SDA —** I2S transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									I/O	**I2S1_RX_WS —** Receive Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									O	**LCD_VD12 —** LCD data.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P3_6	B13	x	C7	174	122	85	[3]	I; PU	I/O	**GPIO0[6] —** General purpose digital input/output pin.
									I/O	**SPI_MISO —** Master In Slave Out for SPI.
									I/O	**SSP0_SSEL —** Slave Select for SSP0.
									I/O	**SPIFI_MISO —** Input 1 in SPIFI quad mode; SPIFI output IO1.
									-	**R —** Function reserved.
									I/O	**SSP0_MISO —** Master In Slave Out for SSP0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P3_7	C11	x	D7	176	123	86	[3]	I; PU	-	**R —** Function reserved.
									I/O	**SPI_MOSI —** Master Out Slave In for SPI.
									I/O	**SSP0_MISO —** Master In Slave Out for SSP0.
									I/O	**SPIFI_MOSI —** Input I0 in SPIFI quad mode; SPIFI output IO0.
									I/O	**GPIO5[10] —** General purpose digital input/output pin.
									I/O	**SSP0_MOSI —** Master Out Slave in for SSP0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P3_8	C10	x	E7	179	124	87	[3]	I; PU	-	**R —** Function reserved.
									I	**SPI_SSEL —** Slave Select for SPI. Note that this pin in an input pin only. The SPI in master mode cannot drive the CS input on the slave. Any GPIO pin can be used for SPI chip select in master mode.
									I/O	**SSP0_MOSI —** Master Out Slave in for SSP0.
									I/O	**SPIFI_CS —** SPIFI serial flash chip select.
									I/O	**GPIO5[11] —** General purpose digital input/output pin.
									I/O	**SSP0_SSEL —** Slave Select for SSP0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P4_0	D5	x	-	1	1	-	[3]	I; PU	I/O	**GPIO2[0] —** General purpose digital input/output pin.
									O	**MCOA0 —** Motor control PWM channel 0, output A.
									I	**NMI —** External interrupt input to NMI.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**LCD_VD13 —** LCD data.
									I/O	**U3_UCLK —** Serial clock input/output for USART3 in synchronous mode.
									-	**R —** Function reserved.
P4_1	A1	x	-	3	3	-	[6]	I; PU	I/O	**GPIO2[1] —** General purpose digital input/output pin.
									O	**CTOUT_1 —** SCT output 1. Match output 1 of timer 0.
									O	**LCD_VD0 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**LCD_VD19 —** LCD data.
									O	**U3_TXD —** Transmitter output for USART3.
									I	**ENET_COL —** Ethernet Collision detect (MII interface).
									I	**ADC0_1 —** ADC0, input channel 1.
P4_2	D3	x	-	12	8	-	[3]	I; PU	I/O	**GPIO2[2] —** General purpose digital input/output pin.
									O	**CTOUT_0 —** SCT output 0. Match output 0 of timer 0.
									O	**LCD_VD3 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**LCD_VD12 —** LCD data.
									I	**U3_RXD —** Receiver input for USART3.
									I/O	**SGPIO8 —** General purpose digital input/output pin.
P4_3	C2	x	-	10	7	-	[6]	I; PU	I/O	**GPIO2[3] —** General purpose digital input/output pin.
									O	**CTOUT_3 —** SCT output 3. Match output 3 of timer 0.
									O	**LCD_VD2 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**LCD_VD21 —** LCD data.
									I/O	**U3_BAUD —** Baud pin for USART3.
									I/O	**SGPIO9 —** General purpose digital input/output pin.
									I	**ADC0_0 —** ADC0, input channel 0.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P4_4	B1	x	-	14	9	-	[6]	I; PU	I/O	**GPIO2[4] —** General purpose digital input/output pin.
									O	**CTOUT_2 —** SCT output 2. Match output 2 of timer 0.
									O	**LCD_VD1 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**LCD_VD20 —** LCD data.
									I/O	**U3_DIR —** RS-485/EIA-485 output enable/direction control for USART3.
									I/O	**SGPIO10 —** General purpose digital input/output pin.
									O	**DAC —** DAC output.
P4_5	D2	x	-	15	10	-	[3]	I; PU	I/O	**GPIO2[5] —** General purpose digital input/output pin.
									O	**CTOUT_5 —** SCT output 5. Match output 1 of timer 1.
									O	**LCD_FP —** Frame pulse (STN). Vertical synchronization pulse (TFT).
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO11 —** General purpose digital input/output pin.
P4_6	C1	x	-	17	11	-	[3]	I; PU	I/O	**GPIO2[6] —** General purpose digital input/output pin.
									O	**CTOUT_4 —** SCT output 4. Match output 0 of timer 1.
									O	**LCD_ENAB/LCDM —** STN AC bias drive or TFT data enable input.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO12 —** General purpose digital input/output pin.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P4_7	H4	x	-	21	14	-	[3]	O; PU	O	**LCD_DCLK —** LCD panel clock.
									I	**GP_CLKIN —** General purpose clock input to the CGU.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**I2S1_TX_SCK —** Transmit Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I^2S-bus specification.
									I/O	**I2S0_TX_SCK —** Transmit Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I^2S-bus specification.
P4_8	E2	x	-	23	15	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_5 —** SCT input 5. Capture input 2 of timer 2.
									O	**LCD_VD9 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[12] —** General purpose digital input/output pin.
									O	**LCD_VD22 —** LCD data.
									O	**CAN1_TD —** CAN1 transmitter output.
									I/O	**SGPIO13 —** General purpose digital input/output pin.
P4_9	L2	x	-	48	33	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_6 —** SCT input 6. Capture input 1 of timer 3.
									O	**LCD_VD11 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[13] —** General purpose digital input/output pin.
									O	**LCD_VD15 —** LCD data.
									I	**CAN1_RD —** CAN1 receiver input.
									I/O	**SGPIO14 —** General purpose digital input/output pin.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P4_10	M3	x	-	51	35	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_2 —** SCT input 2. Capture input 2 of timer 0.
									O	**LCD_VD10 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[14] —** General purpose digital input/output pin.
									O	**LCD_VD14 —** LCD data.
									-	**R —** Function reserved.
									I/O	**SGPIO15 —** General purpose digital input/output pin.
P5_0	N3	x	-	53	37	-	[3]	I; PU	I/O	**GPIO2[9] —** General purpose digital input/output pin.
									O	**MCOB2 —** Motor control PWM channel 2, output B.
									I/O	**EMC_D12 —** External memory data line 12.
									-	**R —** Function reserved.
									I	**U1_DSR —** Data Set Ready input for UART 1.
									I	**T1_CAP0 —** Capture input 0 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P5_1	P3	x	-	55	39	-	[3]	I; PU	I/O	**GPIO2[10] —** General purpose digital input/output pin.
									I	**MCI2 —** Motor control PWM channel 2, input.
									I/O	**EMC_D13 —** External memory data line 13.
									-	**R —** Function reserved.
									O	**U1_DTR —** Data Terminal Ready output for UART 1. Can also be configured to be an RS-485/EIA-485 output enable signal for UART 1.
									I	**T1_CAP1 —** Capture input 1 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P5_2	R4	x	-	63	46	-	[3]	I; PU	I/O	**GPIO2[11] —** General purpose digital input/output pin.
									I	**MCI1 —** Motor control PWM channel 1, input.
									I/O	**EMC_D14 —** External memory data line 14.
									-	**R —** Function reserved.
									O	**U1_RTS —** Request to Send output for UART 1. Can also be configured to be an RS-485/EIA-485 output enable signal for UART 1.
									I	**T1_CAP2 —** Capture input 2 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P5_3	T8	x	-	76	54	-	[3]	I; PU	I/O	**GPIO2[12] —** General purpose digital input/output pin.
									I	**MCI0 —** Motor control PWM channel 0, input.
									I/O	**EMC_D15 —** External memory data line 15.
									-	**R —** Function reserved.
									I	**U1_RI —** Ring Indicator input for UART 1.
									I	**T1_CAP3 —** Capture input 3 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P5_4	P9	x	-	80	57	-	[3]	I; PU	I/O	**GPIO2[13] —** General purpose digital input/output pin.
									O	**MCOB0 —** Motor control PWM channel 0, output B.
									I/O	**EMC_D8 —** External memory data line 8.
									-	**R —** Function reserved.
									I	**U1_CTS —** Clear to Send input for UART 1.
									O	**T1_MAT0 —** Match output 0 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P5_5	P10	x	-	81	58	-	[3]	I; PU	I/O	**GPIO2[14] —** General purpose digital input/output pin.
									O	**MCOA1 —** Motor control PWM channel 1, output A.
									I/O	**EMC_D9 —** External memory data line 9.
									-	**R —** Function reserved.
									I	**U1_DCD —** Data Carrier Detect input for UART 1.
									O	**T1_MAT1 —** Match output 1 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P5_6	T13	x	-	89	63	-	[3]	I; PU	I/O	**GPIO2[15] —** General purpose digital input/output pin.
									O	**MCOB1 —** Motor control PWM channel 1, output B.
									I/O	**EMC_D10 —** External memory data line 10.
									-	**R —** Function reserved.
									O	**U1_TXD —** Transmitter output for UART 1.
									O	**T1_MAT2 —** Match output 2 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P5_7	R12	x	-	91	65	-	[3]	I; PU	I/O	**GPIO2[7] —** General purpose digital input/output pin.
									O	**MCOA2 —** Motor control PWM channel 2, output A.
									I/O	**EMC_D11 —** External memory data line 11.
									-	**R —** Function reserved.
									I	**U1_RXD —** Receiver input for UART 1.
									O	**T1_MAT3 —** Match output 3 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_0	M12	x	H7	105	73	-	[3]	I; PU	-	**R —** Function reserved.
									O	**I2S0_RX_MCLK —** I2S receive master clock.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**I2S0_RX_SCK —** Receive Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the *I^2S-bus specification*.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_1	R15	x	G5	107	74	-	[3]	I; PU	I/O	**GPIO3[0] —** General purpose digital input/output pin.
									O	**EMC_DYCS1 —** SDRAM chip select 1.
									I/O	**U0_UCLK —** Serial clock input/output for USART0 in synchronous mode.
									I/O	**I2S0_RX_WS —** Receive Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									-	**R —** Function reserved.
									I	**T2_CAP0 —** Capture input 2 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P6_2	L13	x	J9	111	78	-	[3]	I; PU	I/O	**GPIO3[1] —** General purpose digital input/output pin.
									O	**EMC_CKEOUT1 —** SDRAM clock enable 1.
									I/O	**U0_DIR —** RS-485/EIA-485 output enable/direction control for USART0.
									I/O	**I2S0_RX_SDA —** I2S Receive data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									-	**R —** Function reserved.
									I	**T2_CAP1 —** Capture input 1 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_3	P15	x	-	113	79	-	[3]	I; PU	I/O	**GPIO3[2] —** General purpose digital input/output pin.
									O	**USB0_PPWR —** VBUS drive signal (towards external charge pump or power management unit); indicates that the VBUS signal must be driven (active HIGH). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the USB_PPWR used on other NXP LPC parts.
									I/O	**SGPIO4 —** General purpose digital input/output pin.
									O	**EMC_CS1 —** LOW active Chip Select 1 signal.
									-	**R —** Function reserved.
									I	**T2_CAP2 —** Capture input 2 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_4	R16	x	F6	114	80	53	[3]	I; PU	I/O	**GPIO3[3] —** General purpose digital input/output pin.
									I	**CTIN_6 —** SCT input 6. Capture input 1 of timer 3.
									O	**U0_TXD —** Transmitter output for USART0.
									O	**EMC_CAS —** LOW active SDRAM Column Address Strobe.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state [2]	Type	Description
P6_5	P16	x	F9	117	82	55	[3]	I; PU	I/O	**GPIO3[4] —** General purpose digital input/output pin.
									O	**CTOUT_6 —** SCT output 6. Match output 2 of timer 1.
									I	**U0_RXD —** Receiver input for USART0.
									O	**$\overline{\text{EMC\_RAS}}$ —** LOW active SDRAM Row Address Strobe.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_6	L14	x	-	119	83	-	[3]	I; PU	I/O	**GPIO0[5] —** General purpose digital input/output pin.
									O	**$\overline{\text{EMC\_BLS1}}$ —** LOW active Byte Lane select signal 1.
									I/O	**SGPIO5 —** General purpose digital input/output pin.
									O	**USB0_PWR_FAULT —** Port power fault signal indicating overcurrent condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
									-	**R —** Function reserved.
									I	**T2_CAP3 —** Capture input 3 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_7	J13	x	-	123	85	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**EMC_A15 —** External memory address line 15.
									I/O	**SGPIO6 —** General purpose digital input/output pin.
									O	**USB0_IND1 —** USB0 port indicator LED control output 1.
									I/O	**GPIO5[15] —** General purpose digital input/output pin.
									O	**T2_MAT0 —** Match output 0 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P6_8	H13	x	-	125	86	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**EMC_A14 —** External memory address line 14.
									I/O	**SGPIO7 —** General purpose digital input/output pin.
									O	**USB0_IND0 —** USB0 port indicator LED control output 0.
									I/O	**GPIO5[16] —** General purpose digital input/output pin.
									O	**T2_MAT1 —** Match output 1 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_9	J15	x	F8	139	97	66	[3]	I; PU	I/O	**GPIO3[5] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**$\overline{\text{EMC\_DYCS0}}$ —** SDRAM chip select 0.
									-	**R —** Function reserved.
									O	**T2_MAT2 —** Match output 2 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_10	H15	x	-	142	100	-	[3]	I; PU	I/O	**$\overline{\text{GPIO3[6]}}$ —** General purpose digital input/output pin.
									O	**MCABORT —** Motor control PWM, LOW-active fast abort.
									-	**R —** Function reserved.
									O	**EMC_DQMOUT1 —** Data mask 1 used with SDRAM and static devices.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P6_11	H12	x	C9	143	101	69	[3]	I; PU	I/O	**GPIO3[7] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**EMC_CKEOUT0 —** SDRAM clock enable 0.
									-	**R —** Function reserved.
									O	**T2_MAT3 —** Match output 3 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state [2]	Type	Description
P6_12	G15	x	-	145	103	-	[3]	I; PU	I/O	**GPIO2[8] —** General purpose digital input/output pin.
									O	**CTOUT_7 —** SCT output 7. Match output 3 of timer 1.
									-	**R —** Function reserved.
									O	**EMC_DQMOUT0 —** Data mask 0 used with SDRAM and static devices.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P7_0	B16	x	-	158	110	-	[3]	I; PU	I/O	**GPIO3[8] —** General purpose digital input/output pin.
									O	**CTOUT_14 —** SCT output 14. Match output 2 of timer 3.
									-	**R —** Function reserved.
									O	**LCD_LE —** Line end signal.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO4 —** General purpose digital input/output pin.
P7_1	C14	x	-	162	113	-	[3]	I; PU	I/O	**GPIO3[9] —** General purpose digital input/output pin.
									O	**CTOUT_15 —** SCT output 15. Match output 3 of timer 3.
									I/O	**I2S0_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									O	**LCD_VD19 —** LCD data.
									O	**LCD_VD7 —** LCD data.
									-	**R —** Function reserved.
									O	**U2_TXD —** Transmitter output for USART2.
									I/O	**SGPIO5 —** General purpose digital input/output pin.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P7_2	A16	x	-	165	115	-	[3]	I; PU	I/O	**GPIO3[10] —** General purpose digital input/output pin.
									I	**CTIN_4 —** SCT input 4. Capture input 2 of timer 1.
									I/O	**I2S0_TX_SDA —** I2S transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									O	**LCD_VD18 —** LCD data.
									O	**LCD_VD6 —** LCD data.
									-	**R —** Function reserved.
									I	**U2_RXD —** Receiver input for USART2.
									I/O	**SGPIO6 —** General purpose digital input/output pin.
P7_3	C13	x	-	167	117	-	[3]	I; PU	I/O	**GPIO3[11] —** General purpose digital input/output pin.
									I	**CTIN_3 —** SCT input 3. Capture input 1 of timer 1.
									-	**R —** Function reserved.
									O	**LCD_VD17 —** LCD data.
									O	**LCD_VD5 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P7_4	C8	x	-	189	132	-	[6]	I; PU	I/O	**GPIO3[12] —** General purpose digital input/output pin.
									O	**CTOUT_13 —** SCT output 13. Match output 1 of timer 3.
									-	**R —** Function reserved.
									O	**LCD_VD16 —** LCD data.
									O	**LCD_VD4 —** LCD data.
									O	**TRACEDATA[0] —** Trace data, bit 0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**ADC0_4 —** ADC0, input channel 4.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P7_5	A7	x	-	191	133	-	[6]	I; PU	I/O	**GPIO3[13] —** General purpose digital input/output pin.
									O	**CTOUT_12 —** SCT output 12. Match output 0 of timer 3.
									-	**R —** Function reserved.
									O	**LCD_VD8 —** LCD data.
									O	**LCD_VD23 —** LCD data.
									O	**TRACEDATA[1] —** Trace data, bit 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**ADC0_3 —** ADC0, input channel 3.
P7_6	C7	x	-	194	134	-	[3]	I; PU	I/O	**GPIO3[14] —** General purpose digital input/output pin.
									O	**CTOUT_11 —** SCT output 1. Match output 3 of timer 2.
									-	**R —** Function reserved.
									O	**LCD_LP —** Line synchronization pulse (STN). Horizontal synchronization pulse (TFT).
									-	**R —** Function reserved.
									O	**TRACEDATA[2] —** Trace data, bit 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
P7_7	B6	x	-	201	140	-	[6]	I; PU	I/O	**GPIO3[15] —** General purpose digital input/output pin.
									O	**CTOUT_8 —** SCT output 8. Match output 0 of timer 2.
									-	**R —** Function reserved.
									O	**LCD_PWR —** LCD panel power enable.
									-	**R —** Function reserved.
									O	**TRACEDATA[3] —** Trace data, bit 3.
									O	**ENET_MDC —** Ethernet MIIM clock.
									I/O	**SGPIO7 —** General purpose digital input/output pin.
									I	**ADC1_6 —** ADC1, input channel 6.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P8_0	E5	x	-	2	-	-	[4]	I; PU	I/O	**GPIO4[0] —** General purpose digital input/output pin.
									O	**USB0_PWR_FAULT —** Port power fault signal indicating overcurrent condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
									-	**R —** Function reserved.
									I	**MCI2 —** Motor control PWM channel 2, input.
									I/O	**SGPIO8 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**T0_MAT0 —** Match output 0 of timer 0.
P8_1	H5	x	-	34	-	-	[4]	I; PU	I/O	**GPIO4[1] —** General purpose digital input/output pin.
									O	**USB0_IND1 —** USB0 port indicator LED control output 1.
									-	**R —** Function reserved.
									I	**MCI1 —** Motor control PWM channel 1, input.
									I/O	**SGPIO9 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**T0_MAT1 —** Match output 1 of timer 0.
P8_2	K4	x	-	36	-	-	[4]	I; PU	I/O	**GPIO4[2] —** General purpose digital input/output pin.
									O	**USB0_IND0 —** USB0 port indicator LED control output 0.
									-	**R —** Function reserved.
									I	**MCI0 —** Motor control PWM channel 0, input.
									I/O	**SGPIO10 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**T0_MAT2 —** Match output 2 of timer 0.
P8_3	J3	x	-	37	-	-	[3]	I; PU	I/O	**GPIO4[3] —** General purpose digital input/output pin.
									I/O	**USB1_ULPI_D2 —** ULPI link bidirectional data line 2.
									-	**R —** Function reserved.
									O	**LCD_VD12 —** LCD data.
									O	**LCD_VD19 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**T0_MAT3 —** Match output 3 of timer 0.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P8_4	J2	x	-	39	-	-	[3]	I; PU	I/O	**GPIO4[4] —** General purpose digital input/output pin.
									I/O	**USB1_ULPI_D1 —** ULPI link bidirectional data line 1.
									-	**R —** Function reserved.
									O	**LCD_VD7 —** LCD data.
									O	**LCD_VD16 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**T0_CAP0 —** Capture input 0 of timer 0.
P8_5	J1	x	-	40	-	-	[3]	I; PU	I/O	**GPIO4[5] —** General purpose digital input/output pin.
									I/O	**USB1_ULPI_D0 —** ULPI link bidirectional data line 0.
									-	**R —** Function reserved.
									O	**LCD_VD6 —** LCD data.
									O	**LCD_VD8 —** LCD data.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**T0_CAP1 —** Capture input 1 of timer 0.
P8_6	K3	x	-	43	-	-	[3]	I; PU	I/O	**GPIO4[6] —** General purpose digital input/output pin.
									I	**USB1_ULPI_NXT —** ULPI link NXT signal. Data flow control signal from the PHY.
									-	**R —** Function reserved.
									O	**LCD_VD5 —** LCD data.
									O	**LCD_LP —** Line synchronization pulse (STN). Horizontal synchronization pulse (TFT).
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**T0_CAP2 —** Capture input 2 of timer 0.
P8_7	K1	x	-	45	-	-	[3]	I; PU	I/O	**GPIO4[7] —** General purpose digital input/output pin.
									O	**USB1_ULPI_STP —** ULPI link STP signal. Asserted to end or interrupt transfers to the PHY.
									-	**R —** Function reserved.
									O	**LCD_VD4 —** LCD data.
									O	**LCD_PWR —** LCD panel power enable.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**T0_CAP3 —** Capture input 3 of timer 0.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P8_8	L1	x	-	49	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**USB1_ULPI_CLK —** ULPI link CLK signal. 60 MHz clock generated by the PHY.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**CGU_OUT0 —** CGU spare clock output 0.
									O	**I2S1_TX_MCLK —** I2S1 transmit master clock.
P9_0	T1	x	-	59	-	-	[3]	I; PU	I/O	**GPIO4[12] —** General purpose digital input/output pin.
									O	**$\overline{\text{MCABORT}}$ —** Motor control PWM, LOW-active fast abort.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**ENET_CRS —** Ethernet Carrier Sense (MII interface).
									I/O	**SGPIO0 —** General purpose digital input/output pin.
									I/O	**SSP0_SSEL —** Slave Select for SSP0.
P9_1	N6	x	-	66	-	-	[3]	I; PU	I/O	**GPIO4[13] —** General purpose digital input/output pin.
									O	**MCOA2 —** Motor control PWM channel 2, output A.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**I2S0_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									I	**ENET_RX_ER —** Ethernet receive error (MII interface).
									I/O	**SGPIO1 —** General purpose digital input/output pin.
									I/O	**SSP0_MISO —** Master In Slave Out for SSP0.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P9_2	N8	x	-	70	-	-	[3]	I; PU	I/O	**GPIO4[14] —** General purpose digital input/output pin.
									O	**MCOB2 —** Motor control PWM channel 2, output B.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**I2S0_TX_SDA —** I2S transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									I	**ENET_RXD3 —** Ethernet receive data 3 (MII interface).
									I/O	**SGPIO2 —** General purpose digital input/output pin.
									I/O	**SSP0_MOSI —** Master Out Slave in for SSP0.
P9_3	M6	x	-	79	-	-	[3]	I; PU	I/O	**GPIO4[15] —** General purpose digital input/output pin.
									O	**MCOA0 —** Motor control PWM channel 0, output A.
									O	**USB1_IND1 —** USB1 Port indicator LED control output 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**ENET_RXD2 —** Ethernet receive data 2 (MII interface).
									I/O	**SGPIO9 —** General purpose digital input/output pin.
									O	**U3_TXD —** Transmitter output for USART3.
P9_4	N10	x	-	92	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**MCOB0 —** Motor control PWM channel 0, output B.
									O	**USB1_IND0 —** USB1 Port indicator LED control output 0.
									-	**R —** Function reserved.
									I/O	**GPIO5[17] —** General purpose digital input/output pin.
									O	**ENET_TXD2 —** Ethernet transmit data 2 (MII interface).
									I/O	**SGPIO4 —** General purpose digital input/output pin.
									I	**U3_RXD —** Receiver input for USART3.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
P9_5	M9	x	-	98	69	-	[3]	I; PU	-	**R —** Function reserved.
									O	**MCOA1 —** Motor control PWM channel 1, output A.
									O	**USB1_PPWR —** VBUS drive signal (towards external charge pump or power management unit); indicates that VBUS must be driven (active high). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the USB_PPWR used on other NXP LPC parts.
									-	**R —** Function reserved.
									I/O	**GPIO5[18] —** General purpose digital input/output pin.
									O	**ENET_TXD3 —** Ethernet transmit data 3 (MII interface).
									I/O	**SGPIO3 —** General purpose digital input/output pin.
									O	**U0_TXD —** Transmitter output for USART0.
P9_6	L11	x	-	103	72	-	[3]	I; PU	I/O	**GPIO4[11] —** General purpose digital input/output pin.
									O	**MCOB1 —** Motor control PWM channel 1, output B.
									O	**USB1_PWR_FAULT —** USB1 Port power fault signal indicating over-current condition; this signal monitors over-current on the USB1 bus (external circuitry required to detect over-current condition).
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**ENET_COL —** Ethernet Collision detect (MII interface).
									I/O	**SGPIO8 —** General purpose digital input/output pin.
									I	**U0_RXD —** Receiver input for USART0.
PA_0	L12	x	-	126	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**I2S1_RX_MCLK —** I2S1 receive master clock.
									O	**CGU_OUT1 —** CGU spare clock output 1.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state [2]	Type	Description
PA_1	J14	x	-	134	-	-	[4]	I; PU	I/O	**GPIO4[8] —** General purpose digital input/output pin.
									I	**QEI_IDX —** Quadrature Encoder Interface INDEX input.
									-	**R —** Function reserved.
									O	**U2_TXD —** Transmitter output for USART2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PA_2	K15	x	-	136	-	-	[4]	I; PU	I/O	**GPIO4[9] —** General purpose digital input/output pin.
									I	**QEI_PHB —** Quadrature Encoder Interface PHB input.
									-	**R —** Function reserved.
									I	**U2_RXD —** Receiver input for USART2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PA_3	H11	x	-	147	-	-	[4]	I; PU	I/O	**GPIO4[10] —** General purpose digital input/output pin.
									I	**QEI_PHA —** Quadrature Encoder Interface PHA input.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PA_4	G13	x	-	151	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_9 —** SCT output 9. Match output 1 of timer 2.
									-	**R —** Function reserved.
									I/O	**EMC_A23 —** External memory address line 23.
									I/O	**GPIO5[19] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PB_0	B15	x	-	164	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_10 —** SCT output 10. Match output 2 of timer 2.
									O	**LCD_VD23 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[20] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PB_1	A14	x	-	175	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**USB1_ULPI_DIR —** ULPI link DIR signal. Controls the ULP data line direction.
									O	**LCD_VD22 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[21] —** General purpose digital input/output pin.
									O	**CTOUT_6 —** SCT output 6. Match output 2 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PB_2	B12	x	-	177	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D7 —** ULPI link bidirectional data line 7.
									O	**LCD_VD21 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[22] —** General purpose digital input/output pin.
									O	**CTOUT_7 —** SCT output 7. Match output 3 of timer 1.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PB_3	A13	x	-	178	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D6 —** ULPI link bidirectional data line 6.
									O	**LCD_VD20 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[23] —** General purpose digital input/output pin.
									O	**CTOUT_8 —** SCT output 8. Match output 0 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PB_4	B11	x	-	180	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D5 —** ULPI link bidirectional data line 5.
									O	**LCD_VD15 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[24] —** General purpose digital input/output pin.
									I	**CTIN_5 —** SCT input 5. Capture input 2 of timer 2.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PB_5	A12	x	-	181	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D4 —** ULPI link bidirectional data line 4.
									O	**LCD_VD14 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[25] —** General purpose digital input/output pin.
									I	**CTIN_7 —** SCT input 7.
									O	**LCD_PWR —** LCD panel power enable.
									-	**R —** Function reserved.
PB_6	A6	x	-	-	-	-	[6]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D3 —** ULPI link bidirectional data line 3.
									O	**LCD_VD13 —** LCD data.
									-	**R —** Function reserved.
									I/O	**GPIO5[26] —** General purpose digital input/output pin.
									I	**CTIN_6 —** SCT input 6. Capture input 1 of timer 3.
									O	**LCD_VD19 —** LCD data.
									-	**R —** Function reserved.
									I	**ADC0_6 —** ADC0, input channel 6.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PC_0	D4	x	-	7	-	-	[6]	I; PU	-	**R —** Function reserved.
									I	**USB1_ULPI_CLK —** ULPI link CLK signal. 60 MHz clock generated by the PHY.
									-	**R —** Function reserved.
									I/O	**ENET_RX_CLK —** Ethernet Receive Clock (MII interface).
									O	**LCD_DCLK —** LCD panel clock.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_CLK —** SD/MMC card clock.
									I	**ADC1_1 —** ADC1, input channel 1.
PC_1	E4	-	-	9	-	-	[3]	I; PU	I/O	**USB1_ULPI_D7 —** ULPI link bidirectional data line 7.
									-	**R —** Function reserved.
									I	**U1_RI —** Ring Indicator input for UART 1.
									O	**ENET_MDC —** Ethernet MIIM clock.
									I/O	**GPIO6[0] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**T3_CAP0 —** Capture input 0 of timer 3.
									O	**SD_VOLT0 —** SD/MMC bus voltage select output 0.
PC_2	F6	-	-	13	-	-	[3]	I; PU	I/O	**USB1_ULPI_D6 —** ULPI link bidirectional data line 6.
									-	**R —** Function reserved.
									I	**U1_CTS —** Clear to Send input for UART 1.
									O	**ENET_TXD2 —** Ethernet transmit data 2 (MII interface).
									I/O	**GPIO6[1] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**SD_RST —** SD/MMC reset signal for MMC4.4 card.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PC_3	F5	-	-	11	-	-	[6]	I; PU	I/O	**USB1_ULPI_D5 —** ULPI link bidirectional data line 5.
									-	**R —** Function reserved.
									O	**U1_RTS —** Request to Send output for UART 1. Can also be configured to be an RS-485/EIA-485 output enable signal for UART 1.
									O	**ENET_TXD3 —** Ethernet transmit data 3 (MII interface).
									I/O	**GPIO6[2] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**SD_VOLT1 —** SD/MMC bus voltage select output 1.
									I	**ADC1_0 —** ADC1, input channel 0.
PC_4	F4	-	-	16	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D4 —** ULPI link bidirectional data line 4.
									-	**R —** Function reserved.
										ENET_TX_EN — Ethernet transmit enable (RMII/MII interface).
									I/O	**GPIO6[3] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**T3_CAP1 —** Capture input 1 of timer 3.
									I/O	**SD_DAT0 —** SD/MMC data bus line 0.
PC_5	G4	-	-	20	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D3 —** ULPI link bidirectional data line 3.
									-	**R —** Function reserved.
									O	**ENET_TX_ER —** Ethernet Transmit Error (MII interface).
									I/O	**GPIO6[4] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**T3_CAP2 —** Capture input 2 of timer 3.
									I/O	**SD_DAT1 —** SD/MMC data bus line 1.
PC_6	H6	-	-	22	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D2 —** ULPI link bidirectional data line 2.
									-	**R —** Function reserved.
									I	**ENET_RXD2 —** Ethernet receive data 2 (MII interface).
									I/O	**GPIO6[5] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**T3_CAP3 —** Capture input 3 of timer 3.
									I/O	**SD_DAT2 —** SD/MMC data bus line 2.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PC_7	G5	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D1 —** ULPI link bidirectional data line 1.
									-	**R —** Function reserved.
									I	**ENET_RXD3 —** Ethernet receive data 3 (MII interface).
									I/O	**GPIO6[6] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT0 —** Match output 0 of timer 3.
									I/O	**SD_DAT3 —** SD/MMC data bus line 3.
PC_8	N4	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									I/O	**USB1_ULPI_D0 —** ULPI link bidirectional data line 0.
									-	**R —** Function reserved.
									I	**ENET_RX_DV —** Ethernet Receive Data Valid (RMII/MII interface).
									I/O	**GPIO6[7] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT1 —** Match output 1 of timer 3.
									I	**SD_CD —** SD/MMC card detect input.
PC_9	K2	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**USB1_ULPI_NXT —** ULPI link NXT signal. Data flow control signal from the PHY.
									-	**R —** Function reserved.
									I	**ENET_RX_ER —** Ethernet receive error (MII interface).
									I/O	**GPIO6[8] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT2 —** Match output 2 of timer 3.
									O	**SD_POW —** SD/MMC power monitor output.
PC_10	M5	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**USB1_ULPI_STP —** ULPI link STP signal. Asserted to end or interrupt transfers to the PHY.
									I	**U1_DSR —** Data Set Ready input for UART 1.
									-	**R —** Function reserved.
									I/O	**GPIO6[9] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**T3_MAT3 —** Match output 3 of timer 3.
									I/O	**SD_CMD —** SD/MMC command signal.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PC_11	L5	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**USB1_ULPI_DIR —** ULPI link DIR signal. Controls the ULP data line direction.
									I	**U1_DCD —** Data Carrier Detect input for UART 1.
									-	**R —** Function reserved.
									I/O	**GPIO6[10] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_DAT4 —** SD/MMC data bus line 4.
PC_12	L6	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**U1_DTR —** Data Terminal Ready output for UART 1. Can also be configured to be an RS-485/EIA-485 output enable signal for UART 1.
									-	**R —** Function reserved.
									I/O	**GPIO6[11] —** General purpose digital input/output pin.
									I/O	**SGPIO11 —** General purpose digital input/output pin.
									I/O	**I2S0_TX_SDA —** I2S transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									I/O	**SD_DAT5 —** SD/MMC data bus line 5.
PC_13	M1	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**U1_TXD —** Transmitter output for UART 1.
									-	**R —** Function reserved.
									I/O	**GPIO6[12] —** General purpose digital input/output pin.
									I/O	**SGPIO12 —** General purpose digital input/output pin.
									I/O	**I2S0_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									I/O	**SD_DAT6 —** SD/MMC data bus line 6.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PC_14	N1	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									I	**U1_RXD —** Receiver input for UART 1.
									-	**R —** Function reserved.
									I/O	**GPIO6[13] —** General purpose digital input/output pin.
									I/O	**SGPIO13 —** General purpose digital input/output pin.
									O	**ENET_TX_ER —** Ethernet Transmit Error (MII interface).
									I/O	**SD_DAT7 —** SD/MMC data bus line 7.
PD_0	N2	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_15 —** SCT output 15. Match output 3 of timer 3.
									O	**EMC_DQMOUT2 —** Data mask 2 used with SDRAM and static devices.
									-	**R —** Function reserved.
									I/O	**GPIO6[14] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO4 —** General purpose digital input/output pin.
PD_1	P1	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**EMC_CKEOUT2 —** SDRAM clock enable 2.
									-	**R —** Function reserved.
									I/O	**GPIO6[15] —** General purpose digital input/output pin.
									O	**SD_POW —** SD/MMC power monitor output.
									-	**R —** Function reserved.
									I/O	**SGPIO5 —** General purpose digital input/output pin.
PD_2	R1	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_7 —** SCT output 7. Match output 3 of timer 1.
									I/O	**EMC_D16 —** External memory data line 16.
									-	**R —** Function reserved.
									I/O	**GPIO6[16] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO6 —** General purpose digital input/output pin.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PD_3	P4	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_6 —** SCT output 7. Match output 2 of timer 1.
									I/O	**EMC_D17 —** External memory data line 17.
									-	**R —** Function reserved.
									I/O	**GPIO6[17] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO7 —** General purpose digital input/output pin.
PD_4	T2	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_8 —** SCT output 8. Match output 0 of timer 2.
									I/O	**EMC_D18 —** External memory data line 18.
									-	**R —** Function reserved.
									I/O	**GPIO6[18] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO8 —** General purpose digital input/output pin.
PD_5	P6	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_9 —** SCT output 9. Match output 1 of timer 2.
									I/O	**EMC_D19 —** External memory data line 19.
									-	**R —** Function reserved.
									I/O	**GPIO6[19] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO9 —** General purpose digital input/output pin.
PD_6	R6	-	-	68	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_10 —** SCT output 10. Match output 2 of timer 2.
									I/O	**EMC_D20 —** External memory data line 20.
									-	**R —** Function reserved.
									I/O	**GPIO6[20] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO10 —** General purpose digital input/output pin.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PD_7	T6	-	-	72	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_5 —** SCT input 5. Capture input 2 of timer 2.
									I/O	**EMC_D21 —** External memory data line 21.
									-	**R —** Function reserved.
									I/O	**GPIO6[21] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO11 —** General purpose digital input/output pin.
PD_8	P8	-	-	74	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_6 —** SCT input 6. Capture input 1 of timer 3.
									I/O	**EMC_D22 —** External memory data line 22.
									-	**R —** Function reserved.
									I/O	**GPIO6[22] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO12 —** General purpose digital input/output pin.
PD_9	T11	-	-	84	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_13 —** SCT output 13. Match output 1 of timer 3.
									I/O	**EMC_D23 —** External memory data line 23.
									-	**R —** Function reserved.
									I/O	**GPIO6[23] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SGPIO13 —** General purpose digital input/output pin.
PD_10	P11	-	-	86	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_1 —** SCT input 1. Capture input 1 of timer 0. Capture input 1 of timer 2.
									O	**$\overline{\text{EMC\_BLS3}}$ —** LOW active Byte Lane select signal 3.
									-	**R —** Function reserved.
									I/O	**GPIO6[24] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PD_11	N9	x	-	88	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**$\overline{\text{EMC\_CS3}}$ —** LOW active Chip Select 3 signal.
									-	**R —** Function reserved.
									I/O	**GPIO6[25] —** General purpose digital input/output pin.
									I/O	**USB1_ULPI_D0 —** ULPI link bidirectional data line 0.
									O	**CTOUT_14 —** SCT output 14. Match output 2 of timer 3.
									-	**R —** Function reserved.
PD_12	N11	x	-	94	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**$\overline{\text{EMC\_CS2}}$ —** LOW active Chip Select 2 signal.
									-	**R —** Function reserved.
									I/O	**GPIO6[26] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**CTOUT_10 —** SCT output 10. Match output 2 of timer 2.
									-	**R —** Function reserved.
PD_13	T14	x	-	97	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_0 —** SCT input 0. Capture input 0 of timer 0, 1, 2, 3.
									O	**$\overline{\text{EMC\_BLS2}}$ —** LOW active Byte Lane select signal 2.
									-	**R —** Function reserved.
									I/O	**GPIO6[27] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**CTOUT_13 —** SCT output 13. Match output 1 of timer 3.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PD_14	R13	x	-	99	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**EMC_DYCS2 —** SDRAM chip select 2.
									-	**R —** Function reserved.
									I/O	**GPIO6[28] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**CTOUT_11 —** SCT output 11. Match output 3 of timer 2.
									-	**R —** Function reserved.
PD_15	T15	x	-	101	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**EMC_A17 —** External memory address line 17.
									-	**R —** Function reserved.
									I/O	**GPIO6[29] —** General purpose digital input/output pin.
									I	**SD_WP —** SD/MMC card write protect input.
									O	**CTOUT_8 —** SCT output 8. Match output 0 of timer 2.
									-	**R —** Function reserved.
PD_16	R14	x	-	104	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**EMC_A16 —** External memory address line 16.
									-	**R —** Function reserved.
									I/O	**GPIO6[30] —** General purpose digital input/output pin.
									O	**SD_VOLT2 —** SD/MMC bus voltage select output 2.
									O	**CTOUT_12 —** SCT output 12. Match output 0 of timer 3.
									-	**R —** Function reserved.
PE_0	P14	x	-	106	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**EMC_A18 —** External memory address line 18.
									I/O	**GPIO7[0] —** General purpose digital input/output pin.
									O	**CAN1_TD —** CAN1 transmitter output.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PE_1	N14	x	-	112	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**EMC_A19 —** External memory address line 19.
									I/O	**GPIO7[1] —** General purpose digital input/output pin.
									I	**CAN1_RD —** CAN1 receiver input.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_2	M14	x	-	115	-	-	[3]	I; PU	I	**ADCTRIG0 —** ADC trigger input 0.
									I	**CAN0_RD —** CAN receiver input.
									-	**R —** Function reserved.
									I/O	**EMC_A20 —** External memory address line 20.
									I/O	**GPIO7[2] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_3	K12	x	-	118	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CAN0_TD —** CAN transmitter output.
									I	**ADCTRIG1 —** ADC trigger input 1.
									I/O	**EMC_A21 —** External memory address line 21.
									I/O	**GPIO7[3] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_4	K13	x	-	120	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**NMI —** External interrupt input to NMI.
									-	**R —** Function reserved.
									I/O	**EMC_A22 —** External memory address line 22.
									I/O	**GPIO7[4] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PE_5	N16	-	-	122	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_3 —** SCT output 3. Match output 3 of timer 0.
									O	**U1_RTS —** Request to Send output for UART 1. Can also be configured to be an RS-485/EIA-485 output enable signal for UART 1.
									I/O	**EMC_D24 —** External memory data line 24.
									I/O	**GPIO7[5] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_6	M16	-	-	124	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_2 —** SCT output 2. Match output 2 of timer 0.
									I	**U1_RI —** Ring Indicator input for UART 1.
									I/O	**EMC_D25 —** External memory data line 25.
									I/O	**GPIO7[6] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_7	F15	-	-	149	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_5 —** SCT output 5. Match output 1 of timer 1.
									I	**U1_CTS —** Clear to Send input for UART1.
									I/O	**EMC_D26 —** External memory data line 26.
									I/O	**GPIO7[7] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_8	F14	-	-	150	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_4 —** SCT output 4. Match output 0 of timer 0.
									I	**U1_DSR —** Data Set Ready input for UART 1.
									I/O	**EMC_D27 —** External memory data line 27.
									I/O	**GPIO7[8] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PE_9	E16	-	-	152	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_4 —** SCT input 4. Capture input 2 of timer 1.
									I	**U1_DCD —** Data Carrier Detect input for UART 1.
									I/O	**EMC_D28 —** External memory data line 28.
									I/O	**GPIO7[9] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_10	E14	-	-	154	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**CTIN_3 —** SCT input 3. Capture input 1 of timer 1.
									O	**U1_DTR —** Data Terminal Ready output for UART 1. Can also be configured to be an RS-485/EIA-485 output enable signal for UART 1.
									I/O	**EMC_D29 —** External memory data line 29.
									I/O	**GPIO7[10] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_11	D16	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_12 —** SCT output 12. Match output 0 of timer 3.
									O	**U1_TXD —** Transmitter output for UART 1.
									I/O	**EMC_D30 —** External memory data line 30.
									I/O	**GPIO7[11] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_12	D15	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_11 —** SCT output 11. Match output 3 of timer 2.
									I	**U1_RXD —** Receiver input for UART 1.
									I/O	**EMC_D31 —** External memory data line 31.
									I/O	**GPIO7[12] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PE_13	G14	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_14 —** SCT output 14. Match output 2 of timer 3.
									I/O	**I2C1_SDA —** I^2C1 data input/output (this pin does not use a specialized I^2C pad).
									O	**EMC_DQMOUT3 —** Data mask 3 used with SDRAM and static devices.
									I/O	**GPIO7[13] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_14	C15	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**$\overline{\text{EMC\_DYCS3}}$ —** SDRAM chip select 3.
									I/O	**GPIO7[14] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
PE_15	E13	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**CTOUT_0 —** SCT output 0. Match output 0 of timer 0.
									I/O	**I2C1_SCL —** I^2C1 clock input/output (this pin does not use a specialized I^2C pad).
									O	**EMC_CKEOUT3 —** SDRAM clock enable 3.
									I/O	**GPIO7[15] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PF_0	D12	-	-	159	-	-	[3]	O; PU	I/O	**SSP0_SCK —** Serial clock for SSP0.
									I	**GP_CLKIN —** General purpose clock input to the CGU.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**I2S1_TX_MCLK —** I2S1 transmit master clock.
PF_1	E11	-	-	-	-	-	[3]	I; PU	-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SSP0_SSEL —** Slave Select for SSP0.
									-	**R —** Function reserved.
									I/O	**GPIO7[16] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO0 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
PF_2	D11	-	-	168	-	-	[3]	I; PU	-	**R —** Function reserved.
									O	**U3_TXD —** Transmitter output for USART3.
									I/O	**SSP0_MISO —** Master In Slave Out for SSP0.
									-	**R —** Function reserved.
									I/O	**GPIO7[17] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO1 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
PF_3	E10	-	-	170	-	-	[3]	I; PU	-	**R —** Function reserved.
									I	**U3_RXD —** Receiver input for USART3.
									I/O	**SSP0_MOSI —** Master Out Slave in for SSP0.
									-	**R —** Function reserved.
									I/O	**GPIO7[18] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO2 —** General purpose digital input/output pin.
									-	**R —** Function reserved.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PF_4	D10	x	H4	172	120	83	[3]	O; PU	I/O	**SSP1_SCK —** Serial clock for SSP1.
									I	**GP_CLKIN —** General purpose clock input to the CGU.
									O	**TRACECLK —** Trace clock.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**I2S0_TX_MCLK —** I2S transmit master clock.
									I/O	**I2S0_RX_SCK —** I2S receive clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the *I^2S-bus specification*.
PF_5	E9	-	-	190	-	-	[6]	I; PU	-	**R —** Function reserved.
									I/O	**U3_UCLK —** Serial clock input/output for USART3 in synchronous mode.
									I/O	**SSP1_SSEL —** Slave Select for SSP1.
									O	**TRACEDATA[0] —** Trace data, bit 0.
									I/O	**GPIO7[19] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO4 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**ADC1_4 —** ADC1, input channel 4.
PF_6	E7	-	-	192	-	-	[6]	I; PU	-	**R —** Function reserved.
									I/O	**U3_DIR —** RS-485/EIA-485 output enable/direction control for USART3.
									I/O	**SSP1_MISO —** Master In Slave Out for SSP1.
									O	**TRACEDATA[1] —** Trace data, bit 1.
									I/O	**GPIO7[20] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO5 —** General purpose digital input/output pin.
									I/O	**I2S1_TX_SDA —** I2S1 transmit data. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the *I^2S-bus specification*.
									I	**ADC1_3 —** ADC1, input channel 3.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PF_7	B7	-	-	193	-	-	[6]	I; PU	-	**R —** Function reserved.
									I/O	**U3_BAUD —** Baud pin for USART3.
									I/O	**SSP1_MOSI —** Master Out Slave in for SSP1.
									O	**TRACEDATA[2] —** Trace data, bit 2.
									I/O	**GPIO7[21] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO6 —** General purpose digital input/output pin.
									I/O	**I2S1_TX_WS —** Transmit Word Select. It is driven by the master and received by the slave. Corresponds to the signal WS in the *I^2S-bus specification*.
									I/O	**ADC1_7 —** ADC1, input channel 7 or band gap output.
PF_8	E6	-	-	-	-	-	[6]	I; PU	-	**R —** Function reserved.
									I/O	**U0_UCLK —** Serial clock input/output for USART0 in synchronous mode.
									I	**CTIN_2 —** SCT input 2. Capture input 2 of timer 0.
									O	**TRACEDATA[3] —** Trace data, bit 3.
									I/O	**GPIO7[22] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO7 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**ADC0_2 —** ADC0, input channel 2.
PF_9	D6	-	-	203	-	-	[6]	I; PU	-	**R —** Function reserved.
									I/O	**U0_DIR —** RS-485/EIA-485 output enable/direction control for USART0.
									O	**CTOUT_1 —** SCT output 1. Match output 1 of timer 0.
									-	**R —** Function reserved.
									I/O	**GPIO7[23] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I/O	**SGPIO3 —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**ADC1_2 —** ADC1, input channel 2.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
PF_10	A3	-	-	205	-	98	[6]	I; PU	-	**R —** Function reserved.
									O	**U0_TXD —** Transmitter output for USART0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**GPIO7[24] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									I	**SD_WP —** SD/MMC card write protect input.
									-	**R —** Function reserved.
									I	**ADC0_5 —** ADC0, input channel 5.
PF_11	A2	-	-	207	-	100	[6]	I; PU	-	**R —** Function reserved.
									I	**U0_RXD —** Receiver input for USART0.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**GPIO7[25] —** General purpose digital input/output pin.
									-	**R —** Function reserved.
									O	**SD_VOLT2 —** SD/MMC bus voltage select output 2.
									-	**R —** Function reserved.
									I	**ADC1_5 —** ADC1, input channel 5.
Clock pins										
CLK0	N5	x	K3	62	45	31	[5]	O; PU	O	**EMC_CLK0 —** SDRAM clock 0.
									O	**CLKOUT —** Clock output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_CLK —** SD/MMC card clock.
									O	**EMC_CLK01 —** SDRAM clock 0 and clock 1 combined.
									I/O	**SSP1_SCK —** Serial clock for SSP1.
									I	**ENET_TX_CLK (ENET_REF_CLK) —** Ethernet Transmit Clock (MII interface) or Ethernet Reference Clock (RMII interface).

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
CLK1	T10	x	-	-	-	-	[5]	O; PU	O	**EMC_CLK1 —** SDRAM clock 1.
									O	**CLKOUT —** Clock output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**CGU_OUT0 —** CGU spare clock output 0.
									-	**R —** Function reserved.
									O	**I2S1_TX_MCLK —** I2S1 transmit master clock.
CLK2	D14	x	K6	141	99	68	[5]	O; PU	O	**EMC_CLK3 —** SDRAM clock 3.
									O	**CLKOUT —** Clock output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									I/O	**SD_CLK —** SD/MMC card clock.
									O	**EMC_CLK23 —** SDRAM clock 2 and clock 3 combined.
									O	**I2S0_TX_MCLK —** I2S transmit master clock.
									I/O	**I2S1_RX_SCK —** Receive Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I^2S-bus specification.
CLK3	P12	x	-	-	-	-	[5]	O; PU	O	**EMC_CLK2 —** SDRAM clock 2.
									O	**CLKOUT —** Clock output pin.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									-	**R —** Function reserved.
									O	**CGU_OUT1 —** CGU spare clock output 1.
									-	**R —** Function reserved.
									I/O	**I2S1_RX_SCK —** Receive Clock. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I^2S-bus specification.
Debug pins										
DBGEN	L4	x	A6	41	28	18	[3]	I	I	JTAG interface control signal. Also used for boundary scan.
TCK/SWDCLK	J5	x	H2	38	27	17	[3]	I; F	I	Test Clock for JTAG interface (default) or Serial Wire (SW) clock.
$\overline{\text{TRST}}$	M4	x	B4	42	29	19	[3]	I; PU	I	Test Reset for JTAG interface.
TMS/SWDIO	K6	x	C4	44	30	20	[3]	I; PU	I	Test Mode Select for JTAG interface (default) or SW debug data input/output.
TDO/SWO	K5	x	H3	46	31	21	[3]	O	O	Test Data Out for JTAG interface (default) or SW trace output.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
TDI	J4	x	G3	35	26	16	[3]	I; PU	I	Test Data In for JTAG interface.
USB0 pins										
USB0_DP	F2	x	E1	26	18	9	[7]	-	I/O	USB0 bidirectional D+ line.
USB0_DM	G2	x	E2	28	20	11	[7]	-	I/O	USB0 bidirectional D− line.
USB0_VBUS	F1	x	E3	29	21	12	[7] [8]	-	I/O	VBUS pin (power on USB cable). This pin includes an internal pull-down resistor of 64 kΩ (typical) ± 16 kΩ.
USB0_ID	H2	x	F1	30	22	13	[9]	-	I	Indicates to the transceiver whether connected as an A-device (USB0_ID LOW) or B-device (USB0_ID HIGH). For OTG this pin has an internal pull-up resistor.
USB0_RREF	H1	x	F3	32	24	15	[9]	-		12.0 kΩ (accuracy 1 %) on-board resistor to ground for current reference.
USB1 pins										
USB1_DP	F12	x	E9	129	89	59	[10]	-	I/O	USB1 bidirectional D+ line.
USB1_DM	G12	x	E10	130	90	60	[10]	-	I/O	USB1 bidirectional D− line.
I^2C-bus pins										
I2C0_SCL	L15	x	D6	132	92	62	[11]	I; F	I/O	I^2C clock input/output. Open-drain output (for I^2C-bus compliance).
I2C0_SDA	L16	x	E6	133	93	63	[11]	I; F	I/O	I^2C data input/output. Open-drain output (for I^2C-bus compliance).
Reset and wake-up pins										
$\overline{\text{RESET}}$	D9	x	B6	185	128	91	[12]	I; IA	I	External reset input: A LOW on this pin resets the device, causing I/O ports and peripherals to take on their default states, and processor execution to begin at address 0.
WAKEUP0	A9	x	A4	187	130	93	[12]	I; IA	I	External wake-up input; can raise an interrupt and can cause wake-up from any of the low power modes.
WAKEUP1	A10	x	-	-	-	-	[12]	I; IA	I	External wake-up input; can raise an interrupt and can cause wake-up from any of the low power modes.
WAKEUP2	C9	x	-	-	-	-	[12]	I; IA	I	External wake-up input; can raise an interrupt and can cause wake-up from any of the low power modes.
WAKEUP3	D8	x	-	-	-	-	[12]	I; IA	I	External wake-up input; can raise an interrupt and can cause wake-up from any of the low power modes.
ADC pins										
ADC0_0/ ADC1_0/DAC	E3	x	A2	8	6	4	[9]	I; IA	I	ADC input channel 0. Shared between 10-bit ADC0/1 and DAC.
ADC0_1/ ADC1_1	C3	x	A1	4	2	1	[9]	I; IA	I	ADC input channel 1. Shared between 10-bit ADC0/1.
ADC0_2/ ADC1_2	A4	x	B3	206	143	99	[9]	I; IA	I	ADC input channel 2. Shared between 10-bit ADC0/1.
ADC0_3/ ADC1_3	B5	x	A3	200	139	96	[9]	I; IA	I	ADC input channel 3. Shared between 10-bit ADC0/1.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
ADC0_4/ ADC1_4	C6	x	-	199	138	-	[9]	I; IA	I	ADC input channel 4. Shared between 10-bit ADC0/1.
ADC0_5/ ADC1_5	B3	x	-	208	144	-	[9]	I; IA	I	ADC input channel 5. Shared between 10-bit ADC0/1.
ADC0_6/ ADC1_6	A5	x	-	204	142	-	[9]	I; IA	I	ADC input channel 6. Shared between 10-bit ADC0/1.
ADC0_7/ ADC1_7	C5	x	-	197	136	-	[9]	I; IA	I	ADC input channel 7. Shared between 10-bit ADC0/1.
RTC										
RTC_ALARM	A11	x	C3	186	129	92	[12]	-	O	RTC controlled output.
RTCX1	A8	x	A5	182	125	88	[9]	-	I	Input to the RTC 32 kHz ultra-low power oscillator circuit.
RTCX2	B8	x	B5	183	126	89	[9]	-	O	Output from the RTC 32 kHz ultra-low power oscillator circuit.
Crystal oscillator pins										
XTAL1	D1	x	B1	18	12	5	[9]	-	I	Input to the oscillator circuit and internal clock generator circuits.
XTAL2	E1	x	C1	19	13	6	[9]	-	O	Output from the oscillator amplifier.
Power and ground pins										
USB0_VDDA 3V3_DRIVER	F3	x	D1	24	16	7		-	-	Separate analog 3.3 V power supply for driver.
USB0 _VDDA3V3	G3	x	D2	25	17	8		-	-	USB 3.3 V separate power supply voltage.
USB0_VSSA _TERM	H3	x	D3	27	19	10		-	-	Dedicated analog ground for clean reference for termination resistors.
USB0_VSSA _REF	G1	x	F2	31	23	14		-	-	Dedicated clean analog ground for generation of reference currents and voltages.
VDDA	B4	x	B2	198	137	95		-	-	Analog power supply and ADC reference voltage.
VBAT	B10	x	C5	184	127	90		-	-	RTC power supply: 3.3 V on this pin supplies power to the RTC.
VDDREG	F10, F9, L8, L7	x	E4, E5, F4	135, 188, 195, 82, 33	94, 131, 59, 25	-			-	Main regulator power supply. Tie the VDDREG and VDDIO pins to a common power supply to ensure the same ramp-up time for both supply voltages.
VPP	E8	-	-	-	-	-	[13]	-	-	OTP programming voltage.

Table 120. Pin description *...continued*

LCD, Ethernet, USB0, and USB1 functions are not available on all parts. See Table 2.

Symbol	LBGA256	TFBGA180[1]	TFBGA100	LQFP208[1]	LQFP144	LQFP100[1]		Reset state[2]	Type	Description
VDDIO	D7, E12, F7, F8, G10, H10, J6, J7, K7, L9, L10, N7, N13	x	F10, K5	6, 52, 57, 102, 110, 155, 160, 202	5, 36, 41, 71, 77, 107, 111, 141	-	[13]	-	-	I/O power supply. Tie the VDDREG and VDDIO pins to a common power supply to ensure the same ramp-up time for both supply voltages.
VDD	-	-	-	-	-	3, 24, 27, 49, 52, 74, 77, 97				Power supply for main regulator, I/O, and OTP.
VSS	G9, H7, J10, J11, K8	x	-	-	-	2, 26, 51, 76	[14] [15]	-	-	Ground.
VSSIO	C4, D13, G6, G7, G8, H8, H9, J8, J9, K9, K10, M13, P7, P13	x	C8, D4, D5, G8, J3, J6	5, 56, 109, 157	4, 40, 76, 109	-	[14] [15]	-	-	Ground.
VSSA	B2	x	C2	196	135	94		-	-	Analog ground.
Not connected										
-	B9	-	-	-	-	-		-	-	n.c.

[1] x = available; - = not pinned out.

[2] I = input, O = output, IA = inactive; PU = pull-up enabled (weak pull-up resistor pulls up pin to $V_{DD(IO)}$); F = floating; Reset state reflects the pin state at reset without boot code operation.

[3] 5 V tolerant pad with 15 ns glitch filter (5 V tolerant if $V_{DD(IO)}$ present; if $V_{DD(IO)}$ not present, do not exceed 3.3 V); provides digital I/O functions with TTL levels and hysteresis; normal drive strength.

[4] 5 V tolerant pad with 15 ns glitch filter(5 V tolerant if $V_{DD(IO)}$ present; if $V_{DD(IO)}$ not present, do not exceed 3.3 V); provides digital I/O functions with TTL levels, and hysteresis; high drive strength.

[5] 5 V tolerant pad with 15 ns glitch filter(5 V tolerant if $V_{DD(IO)}$ present; if $V_{DD(IO)}$ not present, do not exceed 3.3 V); provides high-speed digital I/O functions with TTL levels and hysteresis.

[6] 5 V tolerant pad providing digital I/O functions (with TTL levels and hysteresis) and analog input or output (5 V tolerant if $V_{DD(IO)}$ present; if $V_{DD(IO)}$ not present, do not exceed 3.3 V). When configured as a ADC input or DAC output, the pin is not 5 V tolerant and the digital section of the pad must be disabled by setting the pin to an input function and disabling the pull-up resistor through the pin's SFSP register.

[7] 5 V tolerant transparent analog pad.

[8] For maximum load C_L = 6.5 μF and maximum resistance R_{pd} = 80 kΩ, the VBUS signal takes about 2 s to fall from VBUS = 5 V to VBUS = 0.2 V when it is no longer driven.

[9] Transparent analog pad. Not 5 V tolerant.

[10] Pad provides USB functions 5 V tolerant if $V_{DD(IO)}$ present; if $V_{DD(IO)}$ not present, do not exceed 3.3 V. It is designed in accordance with the USB specification, revision 2.0 (Full-speed and Low-speed mode only). This pad is not 5 V tolerant.

[11] Open-drain 5 V tolerant digital I/O pad, compatible with I^2C-bus Fast Mode Plus specification. This pad requires an external pull-up to provide output functionality. When power is switched off, this pin connected to the I^2C-bus is floating and does not disturb the I^2C lines.

[12] 5 V tolerant pad with 20 ns glitch filter; provides digital I/O functions with open-drain output with weak pull-up resistor and hysteresis.

[13] On the TFBGA100 and LQFP208 packages, VPP is internally connected to VDDIO.

[14] On the LQFP144 package, VSSIO and VSS are connected to a common ground plane.

[15] On the TFBGA100 and LQFP100/208 packages, VSS is internally connected to VSSIO.

15.1 How to read this chapter

The following peripherals are not available on all parts, and the corresponding bit values that select those functions in the SFSP registers are reserved:

- Ethernet: available on LPC4350/30 only.
- USB0: available on LPC4350/30/20 only.
- USB1: available on LPC4350/30 only.
- LCD: available on LPC4350 only.

15.2 Basic configuration

The SCU is configured as follows:

- See Table 121 for clocking and power control.
- The SCU is reset by the SCU_RST (reset # 9).

Table 121. SCU clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to SCU register interface	BASE_M4_CLK	CLK_M4_SCU	up to 204 MHz

Remark: Before using any of the multiplexed pins or the I2C0 pins **as inputs**, set the corresponding pin configuration registers as follows:

- Enable the input buffer by setting bit EZI to 1.
- For high-frequency signals, disable the input glitch filter by setting bit ZIF to 1.

15.3 General description

The system control unit determines the function and electrical mode of most digital pins. By default the digital function 0 is selected for all pins with pull-up enabled.

Remark: Some pads support pin muxing of digital and analog functions. All analog I/Os for the ADC and DAC are also pinned out on analog-only pads without pin muxing.

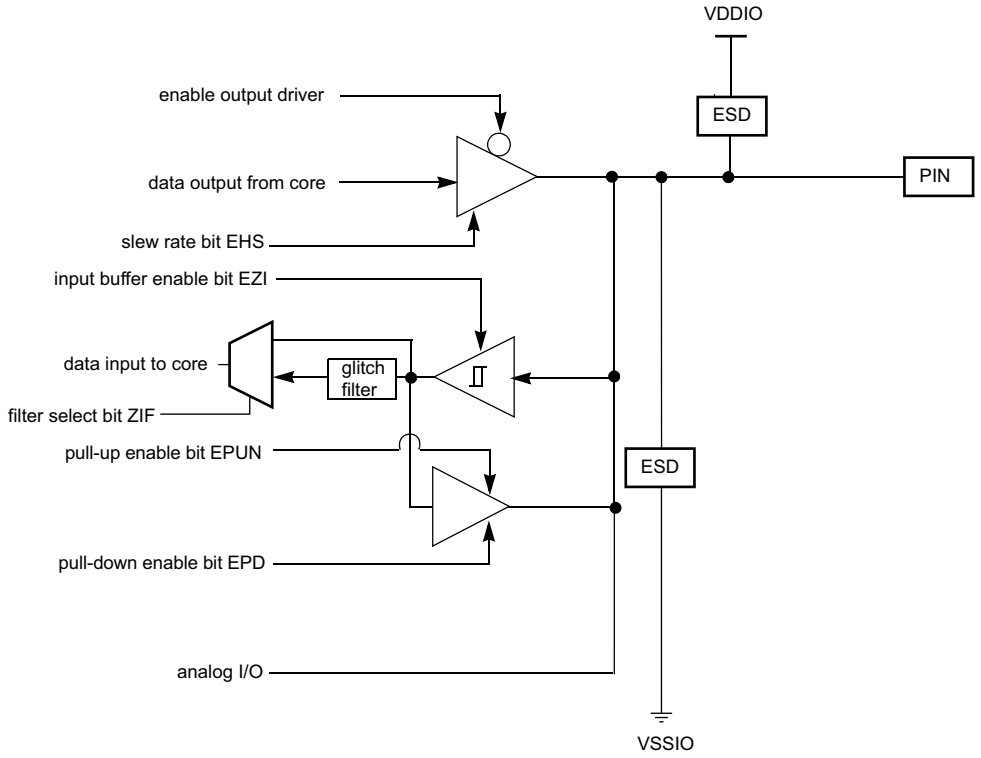


Fig 27. Block diagram of the I/O pad

15.3.1 Digital pin function

The FUNC bits in the SFS registers control the function of each pin. Each pin can have up to 8 digital functions and some pins support an additional analog function. If the function is GPIO, the DIR registers determine whether the pin is configured as an input or output (see Table 191). For any peripheral function, the pin direction is controlled automatically depending on the pin's functionality. The GPIO DIR registers do not affect peripheral functions.

15.3.2 Digital pin mode

The EPUN and EPD bits (see Figure 27) in the SFS registers allow the selection of weak on-chip pull-up or pull-down resistors with a typical value of 50 kΩ for each pin or the selection of the repeater mode.

The possible on-chip resistor configurations are pull-up enabled, pull-down enabled, or no pull-up/pull-down. The default value is pull-up enabled.

The repeater mode enables the pull-up resistor if the pin is at a logic HIGH and enables the pull-down resistor if the pin is at a logic LOW. This causes the pin to retain its last known state if it is configured as an input and is not driven externally. Repeater mode may typically be used to prevent a pin from floating (and potentially using significant power if it floats to an indeterminate state) if it is temporarily not driven.

To select the repeater mode, configure both the pull-up and the pull-down resistor in the SFS registers.

15.3.3 Input buffer

To be able to receive a digital signal, the input buffer must be enabled through bit EZI in the pin configuration registers (see Figure 27). By default, the input buffer is disabled.

For pads that support both a digital and an analog function, the input buffer must be disabled before enabling the analog function (see Section 15.4.6 to Section 15.4.8).

15.3.4 Programmable glitch filter

All digital pins support a programmable glitch filter (bit ZIF), which can be switched on or off (see Figure 27). By default, the glitch filter is on. The glitch filter should be disabled for clocking signals with frequencies higher than 30 MHz.

15.3.5 Programmable slew rate

Normal-drive and high-speed pins support a programmable slew rate (bit EHS) to select between lower noise and speed or higher noise and speed (see Figure 27). The typical frequencies supported are 50 MHz/80 MHz for normal-drive pins and 75 MHz/204 MHz for high-speed pins.

15.3.6 High-speed pins

The clock pins CLK0 to CLK3 and P3_3 support a programmable high-speed output with typical frequencies of 75 MHz or 204 MHz depending on the slew rate setting (see Section 15.3.5).

15.3.7 High-drive pins

Selected pins (see Section 15.4.2) support a high-drive output with four programmable levels.

High-drive pins support the programmable glitch filter but not the programmable slew rate.

15.3.8 I^2C0-bus pins

The SFSI2C0 register (Table 127) allows to configure different modes for I^2C0-bus interface:

- Standard mode/Fast-mode I^2C with an open-drain output according to the I^2C-bus specification. This is the default mode.
- Fast-mode Plus mode with an open-drain output according to the I^2C-bus specification.

The I2C0 pins use a programmable glitch filter (bit ZIF).

Remark: The input buffer must be enabled for the I2C0 pins SDA and SCL for proper operation.

15.3.9 USB1 USB1_DP/USB1_DM pins

The input signal to the USB1 is controlled by the SFSUSB register (Table 126). The USB_ESEA bit in this register must be set to one to enable the USB1 block.

15.3.10 EMC signal delay control

The SCU contains a programmable delay control for all EMC SDRAM clocks (seeTable 134).

15.3.11 Pin multiplexing

Multiplexed digital pins are grouped into 16 pin groups, named P0 to P9 and PA to PF, with up to 20 pins used per group. Each digital pin can support up to eight different digital functions, including General Purpose I/O (GPIO), selectable through the SCU registers. In addition, some pins support an analog function (ADC inputs and DAC output) selectable through the ENAIO registers. Note that the pin name is not indicative of the GPIO port assigned to it.

15.4 Register description

The system control unit contains the registers to configure the pin function of multiplexed digital pins, the EMC clock delays, and the GPIO pin interrupts.

Remark: The boot loader configures the pins involved in the boot process (see Section 5.3.2) when the part starts up. The reset values given in Table 122 for the pin configuration registers and EMC delay registers do not take into account the changes performed by the boot loader.

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
Pins P0_n							
SFSP0_0	R/W	0x000	Pin configuration register for pin P0_0	0x00	0x00	0x00	Table 123
SFSP0_1	R/W	0x004	Pin configuration register for pin P0_1	0x00	0x00	0x00	Table 123
-	-	0x008 - 0x07C	Reserved	-	-	-	
Pins P1_n							
SFSP1_0	R/W	0x080	Pin configuration register for pin P1_0	0x00	0xD2	0x00	Table 123
SFSP1_1	R/W	0x084	Pin configuration register for pin P1_1	0x00	0xD2	0x00	Table 123
SFSP1_2	R/W	0x088	Pin configuration register for pin P1_2	0x00	0xD2	0x00	Table 123
SFSP1_3	R/W	0x08C	Pin configuration register for pin P1_3	0x00	0xD3	0x00	Table 123
SFSP1_4	R/W	0x090	Pin configuration register for pin P1_4	0x00	0xD3	0x00	Table 123
SFSP1_5	R/W	0x094	Pin configuration register for pin P1_5	0x00	0xD3	0x00	Table 123
SFSP1_6	R/W	0x098	Pin configuration register for pin P1_6	0x00	0xD3	0x00	Table 123
SFSP1_7	R/W	0x09C	Pin configuration register for pin P1_7	0x00	0xD3	0x00	Table 123
SFSP1_8	R/W	0x0A0	Pin configuration register for pin P1_8	0x00	0xD3	0x00	Table 123
SFSP1_9	R/W	0x0A4	Pin configuration register for pin P1_9	0x00	0xD3	0x00	Table 123
SFSP1_10	R/W	0x0A8	Pin configuration register for pin P1_10	0x00	0xD3	0x00	Table 123
SFSP1_11	R/W	0x0AC	Pin configuration register for pin P1_11	0x00	0xD3	0x00	Table 123

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
SFSP1_12	R/W	0x0B0	Pin configuration register for pin P1_12	0x00	0xD3	0x00	Table 123
SFSP1_13	R/W	0x0B4	Pin configuration register for pin P1_13	0x00	0xD3	0x00	Table 123
SFSP1_14	R/W	0x0B8	Pin configuration register for pin P1_14	0x00	0xD3	0x00	Table 123
SFSP1_15	R/W	0x0BC	Pin configuration register for pin P1_15	0x00	0x00	0x00	Table 123
SFSP1_16	R/W	0x0C0	Pin configuration register for pin P1_16	0x00	0x00	0x00	Table 123
SFSP1_17	R/W	0x0C4	Pin configuration register for pin P1_17	0x00	0x00	0x00	Table 124
SFSP1_18	R/W	0x0C8	Pin configuration register for pin P1_18	0x00	0x00	0x00	Table 123
SFSP1_19	R/W	0x0CC	Pin configuration register for pin P1_19	0x00	0x00	0x00	Table 123
SFSP1_20	R/W	0x0D0	Pin configuration register for pin P1_20	0x00	0x00	0x00	Table 123
-	-	0x0D4 - 0x0FC	Reserved	-	-	-	
Pins P2_n							
SFSP2_0	R/W	0x100	Pin configuration register for pin P2_0	0x00	0xD8	0xD1 (UART 0)	Table 123
SFSP2_1	R/W	0x104	Pin configuration register for pin P2_1	0x00	0xD2	0xD1 (UART 0)	Table 123
SFSP2_2	R/W	0x108	Pin configuration register for pin P2_2	0x00	0xD2		Table 123
SFSP2_3	R/W	0x10C	Pin configuration register for pin P2_3	0x00	0x00	0xD2 (UART 3)	Table 124
SFSP2_4	R/W	0x110	Pin configuration register for pin P2_4	0x00	0x00	0xD2 (UART 3)	Table 124
SFSP2_5	R/W	0x114	Pin configuration register for pin P2_5	0x00	0x00	0x00	Table 124
SFSP2_6	R/W	0x118	Pin configuration register for pin P2_6	0x00	0xD2	0x00	Table 123
SFSP2_7	R/W	0x11C	Pin configuration register for pin P2_7	0x00	0xD3	0x00	Table 123
SFSP2_8	R/W	0x120	Pin configuration register for pin P2_8	0x00	0xD3	0x00	Table 123
SFSP2_9	R/W	0x124	Pin configuration register for pin P2_9	0x00	0xD3	0x00	Table 123

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
SFSP2_10	R/W	0x128	Pin configuration register for pin P2_10	0x00	0xD3	0x00	Table 123
SFSP2_11	R/W	0x12C	Pin configuration register for pin P2_11	0x00	0xD3	0x00	Table 123
SFSP2_12	R/W	0x130	Pin configuration register for pin P2_12	0x00	0xD3	0x00	Table 123
SFSP2_13	R/W	0x134	Pin configuration register for pin P2_13	0x00	0xD3	0x00	Table 123
-	-	0x138 - 0x17C	Reserved	-	-	-	-
Pins P3_n							
SFSP3_0	R/W	0x180	Pin configuration register for pin P3_0	0x00	0x00	0x00	Table 123
SFSP3_1	R/W	0x184	Pin configuration register for pin P3_1	0x00	0x00	0x00	Table 123
SFSP3_2	R/W	0x188	Pin configuration register for pin P3_2	0x00	0x00	0x00	Table 123
SFSP3_3	R/W	0x18C	Pin configuration register for pin P3_3	0x00	0x00	0x00	Table 125
SFSP3_4	R/W	0x190	Pin configuration register for pin P3_4	0x00	0x00	0x00	Table 123
SFSP3_5	R/W	0x194	Pin configuration register for pin P3_5	0x00	0x00	0x00	Table 123
SFSP3_6	R/W	0x198	Pin configuration register for pin P3_6	0x00	0x00	0x00	Table 123
SFSP3_7	R/W	0x19C	Pin configuration register for pin P3_7	0x00	0x00	0x00	Table 123
SFSP3_8	R/W	0x1A0	Pin configuration register for pin P3_8	0x00	0x00	0x00	Table 123
-	-	0x1A4 - 0x1FC	Reserved	-	-	-	
Pins P4_n							
SFSP4_0	R/W	0x200	Pin configuration register for pin P4_0	0x00	0x00	0x00	Table 123
SFSP4_1	R/W	0x204	Pin configuration register for pin P4_1	0x00	0x00	0x00	Table 123/ Table 129
SFSP4_2	R/W	0x208	Pin configuration register for pin P4_2	0x00	0x00	0x00	Table 123
SFSP4_3	R/W	0x20C	Pin configuration register for pin P4_3	0x00	0x00	0x00	Table 123/ Table 129
SFSP4_4	R/W	0x210	Pin configuration register for pin P4_4	0x00	0x00	0x00	Table 123/ Table 133
SFSP4_5	R/W	0x214	Pin configuration register for pin P4_5	0x00	0x00	0x00	Table 123
SFSP4_6	R/W	0x218	Pin configuration register for pin P4_6	0x00	0x00	0x00	Table 123
SFSP4_7	R/W	0x21C	Pin configuration register for pin P4_7	0x00	0x00	0x00	Table 123
SFSP4_8	R/W	0x220	Pin configuration register for pin P4_8	0x00	0x00	0x00	Table 123
SFSP4_9	R/W	0x224	Pin configuration register for pin P4_9	0x00	0x00	0x00	Table 123
SFSP4_10	R/W	0x228	Pin configuration register for pin P4_10	0x00	0x00	0x00	Table 123
-	-	0x22C - 0x27C	Reserved	-	-	-	
Pins P5_n							
SFSP5_0	R/W	0x280	Pin configuration register for pin P5_0	0x00	0xD2	0x00	Table 123
SFSP5_1	R/W	0x284	Pin configuration register for pin P5_1	0x00	0xD2	0x00	Table 123
SFSP5_2	R/W	0x288	Pin configuration register for pin P5_2	0x00	0xD2	0x00	Table 123

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
SFSP5_3	R/W	0x28C	Pin configuration register for pin P5_3	0x00	0xD2	0x00	Table 123
SFSP5_4	R/W	0x290	Pin configuration register for pin P5_4	0x00	0xD2	0x00	Table 123
SFSP5_5	R/W	0x294	Pin configuration register for pin P5_5	0x00	0xD2	0x00	Table 123
SFSP5_6	R/W	0x298	Pin configuration register for pin P5_6	0x00	0xD2	0x00	Table 123
SFSP5_7	R/W	0x29C	Pin configuration register for pin P5_7	0x00	0xD2	0x00	Table 123
-	-	0x2A0 - 0x2FC	Reserved	-	-	-	-
Pins P6_n							
SFSP6_0	R/W	0x300	Pin configuration register for pin P6_0	0x00	0x00	0x00	Table 123
SFSP6_1	R/W	0x304	Pin configuration register for pin P6_1	0x00	0x00	0x00	Table 123
SFSP6_2	R/W	0x308	Pin configuration register for pin P6_2	0x00	0x00	0x00	Table 123
SFSP6_3	R/W	0x30C	Pin configuration register for pin P6_3	0x00	0x00	0x00	Table 123
SFSP6_4	R/W	0x310	Pin configuration register for pin P6_4	0x00	0x00	0x00	Table 123
SFSP6_5	R/W	0x314	Pin configuration register for pin P6_5	0x00	0x00	0x00	Table 123
SFSP6_6	R/W	0x318	Pin configuration register for pin P6_6	0x00	0xD1	0x00	Table 123
SFSP6_7	R/W	0x31C	Pin configuration register for pin P6_7	0x00	0xD8	0x00	Table 123
SFSP6_8	R/W	0x320	Pin configuration register for pin P6_8	0x00	0xD8	0x00	Table 123
SFSP6_9	R/W	0x324	Pin configuration register for pin P6_9	0x00	0x00	0x00	Table 123
SFSP6_10	R/W	0x328	Pin configuration register for pin P6_10	0x00	0x00	0x00	Table 123
SFSP6_11	R/W	0x32C	Pin configuration register for pin P6_11	0x00	0x00	0x00	Table 123
SFSP6_12	R/W	0x330	Pin configuration register for pin P6_12	0x00	0x00	0x00	Table 123
-	-	0x334 - 0x37C	Reserved	-	-	-	-
Pins P7_n							
SFSP7_0	R/W	0x380	Pin configuration register for pin P7_0	0x00	0x00	0x00	Table 123
SFSP7_1	R/W	0x384	Pin configuration register for pin P7_1	0x00	0x00	0x00	Table 123
SFSP7_2	R/W	0x388	Pin configuration register for pin P7_2	0x00	0x00	0x00	Table 123
SFSP7_3	R/W	0x38C	Pin configuration register for pin P7_3	0x00	0x00	0x00	Table 123
SFSP7_4	R/W	0x390	Pin configuration register for pin P7_4	0x00	0x00	0x00	Table 123/ Table 129
SFSP7_5	R/W	0x394	Pin configuration register for pin P7_5	0x00	0x00	0x00	Table 123/ Table 129
SFSP7_6	R/W	0x398	Pin configuration register for pin P7_6	0x00	0x00	0x00	Table 123
SFSP7_7	R/W	0x39C	Pin configuration register for pin P7_7	0x00	0x00	0x00	Table 123/ Table 131
-	-	0x3A0 - 0x3FC	Reserved	-	-	-	-
Pins P8_n							
SFSP8_0	R/W	0x400	Pin configuration register for pin P8_0	0x00	0x00	0x00	Table 124

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
SFSP8_1	R/W	0x404	Pin configuration register for pin P8_1	0x00	0x00	0x00	Table 124
SFSP8_2	R/W	0x408	Pin configuration register for pin P8_2	0x00	0x00	0x00	Table 124
SFSP8_3	R/W	0x40C	Pin configuration register for pin P8_3	0x00	0x00	0x00	Table 123
SFSP8_4	R/W	0x410	Pin configuration register for pin P8_4	0x00	0x00	0x00	Table 123
SFSP8_5	R/W	0x414	Pin configuration register for pin P8_5	0x00	0x00	0x00	Table 123
SFSP8_6	R/W	0x418	Pin configuration register for pin P8_6	0x00	0x00	0x00	Table 123
SFSP8_7	R/W	0x41C	Pin configuration register for pin P8_7	0x00	0x00	0x00	Table 123
SFSP8_8	R/W	0x420	Pin configuration register for pin P8_8	0x00	0x00	0x00	Table 123
-	-	0x424 - 0x47C	Reserved	-	-	-	-
Pins P9_n							
SFSP9_0	R/W	0x480	Pin configuration register for pin P9_0	0x00	0x00	0x00	Table 123
SFSP9_1	R/W	0x484	Pin configuration register for pin P9_1	0x00	0x00	0x00	Table 123
SFSP9_2	R/W	0x488	Pin configuration register for pin P9_2	0x00	0x00	0x00	Table 123
SFSP9_3	R/W	0x49C	Pin configuration register for pin P9_3	0x00	0x00	0x00	Table 123
SFSP9_4	R/W	0x490	Pin configuration register for pin P9_4	0x00	0x00	0x00	Table 123
SFSP9_5	R/W	0x494	Pin configuration register for pin P9_5	0x00	0x00	0x00	Table 123
SFSP9_6	R/W	0x498	Pin configuration register for pin P9_6	0x00	0x00	0x00	Table 123
-	-	0x49C - 0x4FC	Reserved	-	-	-	-
Pins PA_n							
SFSPA_0	R/W	0x500	Pin configuration register for pin PA_0	0x00	0x00	0x00	Table 123
SFSPA_1	R/W	0x504	Pin configuration register for pin PA_1	0x00	0x00	0x00	Table 124
SFSPA_2	R/W	0x508	Pin configuration register for pin PA_2	0x00	0x00	0x00	Table 124
SFSPA_3	R/W	0x50C	Pin configuration register for pin PA_3	0x00	0x00	0x00	Table 124
SFSPA_4	R/W	0x510	Pin configuration register for pin PA_4	0x00	0xD8	0x00	Table 123
-	-	0x514 - 0x57C	Reserved	-	-	-	-
Pins PB_n							
SFSPB_0	R/W	0x580	Pin configuration register for pin PB_0	0x00	0x00	0x00	Table 123
SFSPB_1	R/W	0x584	Pin configuration register for pin PB_1	0x00	0x00	0x00	Table 123
SFSPB_2	R/W	0x588	Pin configuration register for pin PB_2	0x00	0x00	0x00	Table 123
SFSPB_3	R/W	0x58C	Pin configuration register for pin PB_3	0x00	0x00	0x00	Table 123
SFSPB_4	R/W	0x590	Pin configuration register for pin PB_4	0x00	0x00	0x00	Table 123
SFSPB_5	R/W	0x594	Pin configuration register for pin PB_5	0x00	0x00	0x00	Table 123
SFSPB_6	R/W	0x598	Pin configuration register for pin PB_6	0x00	0x00	0x00	Table 123/ Table 129

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
-	-	0x59C - 0x5FC	Reserved	-	-	-	-
Pins PC_n							
SFSPC_0	R/W	0x600	Pin configuration register for pin PC_0	0x00	0x00	0x00	Table 123/ Table 131
SFSPC_1	R/W	0x604	Pin configuration register for pin PC_1	0x00	0x00	0x00	Table 123
SFSPC_2	R/W	0x608	Pin configuration register for pin PC_2	0x00	0x00	0x00	Table 123
SFSPC_3	R/W	0x60C	Pin configuration register for pin PC_3	0x00	0x00	0x00	Table 123/ Table 131
SFSPC_4	R/W	0x610	Pin configuration register for pin PC_4	0x00	0x00	0x00	Table 123
SFSPC_5	R/W	0x614	Pin configuration register for pin PC_5	0x00	0x00	0x00	Table 123
SFSPC_6	R/W	0x618	Pin configuration register for pin PC_6	0x00	0x00	0x00	Table 123
SFSPC_7	R/W	0x61C	Pin configuration register for pin PC_7	0x00	0x00	0x00	Table 123
SFSPC_8	R/W	0x620	Pin configuration register for pin PC_8	0x00	0x00	0x00	Table 123
SFSPC_9	R/W	0x624	Pin configuration register for pin PC_9	0x00	0x00	0x00	Table 123
SFSPC_10	R/W	0x628	Pin configuration register for pin PC_10	0x00	0x00	0x00	Table 123
SFSPC_11	R/W	0x62C	Pin configuration register for pin PC_11	0x00	0x00	0x00	Table 123
SFSPC_12	R/W	0x630	Pin configuration register for pin PC_12	0x00	0x00	0x00	Table 123
SFSPC_13	R/W	0x634	Pin configuration register for pin PC_13	0x00	0x00	0x00	Table 123
SFSPC_14	R/W	0x638	Pin configuration register for pin PC_14	0x00	0x00	0x00	Table 123
-	-	0x63C - 0x67C	Reserved	-	-	-	-
Pins PD_n							
SFSPD_0	R/W	0x680	Pin configuration register for pin PD_0	0x00	0x00	0x00	Table 123
SFSPD_1	R/W	0x684	Pin configuration register for pin PD_1	0x00	0x00	0x00	Table 123
SFSPD_2	R/W	0x688	Pin configuration register for pin PD_2	0x00	0x00	0x00	Table 123
SFSPD_3	R/W	0x68C	Pin configuration register for pin PD_3	0x00	0x00	0x00	Table 123
SFSPD_4	R/W	0x690	Pin configuration register for pin PD_4	0x00	0x00	0x00	Table 123
SFSPD_5	R/W	0x694	Pin configuration register for pin PD_5	0x00	0x00	0x00	Table 123
SFSPD_6	R/W	0x698	Pin configuration register for pin PD_6	0x00	0x00	0x00	Table 123
SFSPD_7	R/W	0x69C	Pin configuration register for pin PD_7	0x00	0x00	0x00	Table 123
SFSPD_8	R/W	0x6A0	Pin configuration register for pin PD_8	0x00	0x00	0x00	Table 123
SFSPD_9	R/W	0x6A4	Pin configuration register for pin PD_9	0x00	0x00	0x00	Table 123
SFSPD_10	R/W	0x6A8	Pin configuration register for pin PD_10	0x00	0xD2	0x00	Table 123
SFSPD_11	R/W	0x6AC	Pin configuration register for pin PD_11	0x00	0x00	0x00	Table 123
SFSPD_12	R/W	0x6B0	Pin configuration register for pin PD_12	0x00	0x00	0x00	Table 123
SFSPD_13	R/W	0x6B4	Pin configuration register for pin PD_13	0x00	0xD2	0x00	Table 123
SFSPD_14	R/W	0x6B8	Pin configuration register for pin PD_14	0x00	0x00	0x00	Table 123

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
SFSPD_15	R/W	0x6BC	Pin configuration register for pin PD_15	0x00	0xD8	0x00	Table 123
SFSPD_16	R/W	0x6C0	Pin configuration register for pin PD_16	0x00	0xD8	0x00	Table 123
-	-	0x6C4 - 0x6FC	Reserved	-			
Pins PE_n							
SFSPE_0	R/W	0x700	Pin configuration register for pin PE_0	0x00	0xD8	0x00	Table 123
SFSPE_1	R/W	0x704	Pin configuration register for pin PE_1	0x00	0xD8	0x00	Table 123
SFSPE_2	R/W	0x708	Pin configuration register for pin PE_2	0x00	0xD8	0x00	Table 123
SFSPE_3	R/W	0x70C	Pin configuration register for pin PE_3	0x00	0xD8	0x00	Table 123
SFSPE_4	R/W	0x710	Pin configuration register for pin PE_4	0x00	0xD8	0x00	Table 123
SFSPE_5	R/W	0x714	Pin configuration register for pin PE_5	0x00	0x00	0x00	Table 123
SFSPE_6	R/W	0x718	Pin configuration register for pin PE_6	0x00	0x00	0x00	Table 123
SFSPE_7	R/W	0x71C	Pin configuration register for pin PE_7	0x00	0x00	0x00	Table 123
SFSPE_8	R/W	0x720	Pin configuration register for pin PE_8	0x00	0x00	0x00	Table 123
SFSPE_9	R/W	0x724	Pin configuration register for pin PE_9	0x00	0x00	0x00	Table 123
SFSPE_10	R/W	0x728	Pin configuration register for pin PE_10	0x00	0x00	0x00	Table 123
SFSPE_11	R/W	0x72C	Pin configuration register for pin PE_11	0x00	0x00	0x00	Table 123
SFSPE_12	R/W	0x730	Pin configuration register for pin PE_12	0x00	0x00	0x00	Table 123
SFSPE_13	R/W	0x734	Pin configuration register for pin PE_13	0x00	0x00	0x00	Table 123
SFSPE_14	R/W	0x738	Pin configuration register for pin PE_14	0x00	0x00	0x00	Table 123
SFSPE_15	R/W	0x73C	Pin configuration register for pin PE_15	0x00	0x00	0x00	Table 123
-	-	0x740 - 0x77C	Reserved	-			
Pins PF_n							
SFSPF_0	R/W	0x780	Pin configuration register for pin PF_0	0x00	0x00	0x00	Table 123
SFSPF_1	R/W	0x784	Pin configuration register for pin PF_1	0x00	0x00	0x00	Table 123
SFSPF_2	R/W	0x788	Pin configuration register for pin PF_2	0x00	0x00	0x00	Table 123
SFSPF_3	R/W	0x78C	Pin configuration register for pin PF_3	0x00	0x00	0x00	Table 123
SFSPF_4	R/W	0x790	Pin configuration register for pin PF_4	0x00	0x00	0x00	Table 123
SFSPF_5	R/W	0x794	Pin configuration register for pin PF_5	0x00	0x00	0x00	Table 123/ Table 131
SFSPF_6	R/W	0x798	Pin configuration register for pin PF_6	0x00	0x00	0x00	Table 123/ Table 131
SFSPF_7	R/W	0x79C	Pin configuration register for pin PF_7	0x00	0x00	0x00	Table 123/ Table 131
SFSPF_8	R/W	0x7A0	Pin configuration register for pin PF_8	0x00	0x00	0x00	Table 123/ Table 129
SFSPF_9	R/W	0x7A4	Pin configuration register for pin PF_9	0x00	0x00	0x00	Table 123/ Table 131

Table 122. Register overview: System Control Unit (SCU) (base address 0x4008 6000)

…continued

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reset value after UART boot	Reference
SFSPF_10	R/W	0x7A8	Pin configuration register for pin PF_10	0x00	0x00	0x00	Table 123/ Table 129
SFSPF_11	R/W	0x7AC	Pin configuration register for pin PF_11	0x00	0x00	0x00	Table 123/ Table 131
-	-	0x7B0 - 0xBFC	Reserved	-	-	-	-
CLKn pins							
SFSCLK0	R/W	0xC00	Pin configuration register for pin CLK0	0x00	<tbd>	0x00	Table 125
SFSCLK1	R/W	0xC04	Pin configuration register for pin CLK1	0x00	<tbd>	0x00	Table 125
SFSCLK2	R/W	0xC08	Pin configuration register for pin CLK2	0x00	<tbd>	0x00	Table 125
SFSCLK3	R/W	0xC0C	Pin configuration register for pin CLK3	0x00	<tbd>	0x00	Table 125
-	-	0xC10 - 0xC7C	Reserved	-	-	-	-
USB1 USB1_DP/USB1_DM pins and I^2C-bus open-drain pins							
SFSUSB	R/W	0xC80	Pin configuration register for pins USB1_DM and USB1_DP	0x02	0x00	0x00	Table 126
SFSI2C0	R/W	0xC84	Pin configuration register for I^2C0-bus pins	0x00	0x00	0x00	Table 127
ADC pin select registers							
ENAIO0	R/W	0xC88	ADC0 function select register	0x00	0x00	0x00	Table 129
ENAIO1	R/W	0xC8C	ADC1 function select register	0x00	0x00	0x00	Table 131
ENAIO2	R/W	0xC90	Analog function select register	0x00	0x00	0x00	Table 133
EMC delay registers							
EMCDELAYCLK	R/W	0xD00	EMC clock delay register	0x00	0x00	0x00	Table 134
Pin interrupt select registers							
PINTSEL0	R/W	0xE00	Pin interrupt select register for pin interrupts 0 to 3.	0x00	0x00	0x00	Table 135
PINTSEL1	R/W	0xE04	Pin interrupt select register for pin interrupts 4 to 7.	0x00	0x00	0x00	Table 136

15.4.1 Pin configuration registers for normal-drive pins

Each digital pin and each clock pin on the LPC43xx have an associated pin configuration register which determines the pin's function and electrical characteristics. The assigned functions for each pin are listed in <tbd>.

The pin configuration registers for normal-drive pins control the following pins:

- P0_0 and P0_1
- P1_0 to P1_16 and P1_18 to P1_20
- P2_0 to P2_2 and P2_6 to P2_13

- P3_0 to P3_2 and P3_4 to P3_8
- P4_0 to P4_10
- P5_0 to P5_7
- P6_0 to P6_12
- P7_0 to P7_7
- P8_3 to P8_8
- P9_0 to P9_6
- PA_0 and PA_4
- PB_0 to PB_6
- PC_0 to PC_14
- PE_0 to PE_15
- PF_0 to PF_11

Table 123. Pin configuration registers for normal-drive pins (SFS, address 0x4008 6000 (SPSP0_0) to 0x4008 67AC (SFSPF_11)) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	MODE		Select pin function.	0	R/W
		0x0	Function 0 (default)		
		0x1	Function 1		
		0x2	Function 2		
		0x3	Function 3		
		0x4	Function 4		
		0x5	Function 5		
		0x6	Function 6		
		0x7	Function 7		
3	EPD		Enable pull-down resistor at pad.	0	R/W
		0	Disable pull-down.		
		1	Enable pull-down.Enable both pull-down resistor and pull-up resistor for repeater mode.		
4	EPUN		Disable pull-up resistor at pad. By default, the pull-up resistor is enabled at reset.	0	R/W
		0	Enable pull-up. Enable both pull-down resistor and pull-up resistor for repeater mode.		
		1	Disable pull-up.		
5	EHS		Select Slew rate.	0	R/W
		0	Slow (low noise with medium speed)		
		1	Fast (medium noise with fast speed)		
6	EZI		Input buffer enable. The input buffer is disabled by default at reset and must be enabled for receiving.	0	R/W
		0	Disable input buffer		
		1	Enable input buffer		

Table 123. Pin configuration registers for normal-drive pins (SFS, address 0x4008 6000 (SPSP0_0) to 0x4008 67AC (SFSPF_11)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
7	ZIF		Input glitch filter. Disable the input glitch filter for clocking signals higher than 30 MHz.	0	R/W
		0	Enable input glitch filter		
		1	Disable input glitch filter		
31:8	-		Reserved	-	-

15.4.2 Pin configuration registers for high-drive pins

Each digital pin and each clock pin on the LPC43xx have an associated pin configuration register which determines the pin's function and electrical characteristics. The assigned functions for each pin are listed in <tbd>.

The pin configuration registers for high-drive pins control the following pins:

- P1_17
- P2_3 to P2_5
- P8_0 to P8_2
- PA_1 to PA_3

Table 124. Pin configuration registers for high-drive pins (SFS, address 0x4008 60C4 (SFSP1_17) to 0x4008 650C (SFSPA_3) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	MODE		Select pin function.	0	R/W
		0x0	Function 0 (default)		
		0x1	Function 1		
		0x2	Function 2		
		0x3	Function 3		
		0x4	Function 4		
		0x5	Function 5		
		0x6	Function 6		
		0x7	Function 7		
3	EPD		Enable pull-down resistor at pad.	0	R/W
		0	Disable pull-down.		
		1	Enable pull-down. Enable both pull-down resistor and pull-up resistor for repeater mode.		
4	EPUN		Disable pull-up resistor at pad. By default, the pull-up resistor is enabled at reset.	0	R/W
		0	Enable pull-up. Enable both pull-down resistor and pull-up resistor for repeater mode.		
		1	Disable pull-up		
5	-		Reserved	-	-

Table 124. Pin configuration registers for high-drive pins (SFS, address 0x4008 60C4 (SFSP1_17) to 0x4008 650C (SFSPA_3) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
6	EZI		Input buffer enable. The input buffer is disabled by default at reset but must be enabled to transfer data from the I/O buffer to the pad.	0	R/W
		0	Disable input buffer		
		1	Enable input buffer		
7	ZIF		Input glitch filter. Disable the input glitch filter for clocking signals higher than 30 MHz.	0	R/W
		0	Enable input glitch filter		
		1	Disable input glitch filter		
9:8	EHD		Select drive strength.	0	R/W
		0x0	Normal-drive: 4 mA drive strength		
		0x1	Medium-drive: 8 mA drive strength		
		0x2	High-drive: 14 mA drive strength		
		0x3	Ultra high-drive: 20 mA drive strength		
31:10	-		Reserved	-	-

15.4.3 Pin configuration registers for high-speed pins

Each digital pin and each clock pin on the LPC43xx have an associated pin configuration register which determines the pin's function and electrical characteristics. The assigned functions for each pin are listed in <tbd>.

This register controls the following pins: P3_3 and pins CLK0 to CLK3.

Table 125. Pin configuration registers for high-speed pins (SFS, address 0x4008 618C (SPSP3_3); 0x4008 6C00 (SFSCLK0) to 0x4008 6C0C (SFSCLK3)) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	MODE		Select pin function.	0	R/W
		0x0	Function 0 (default)		
		0x1	Function 1		
		0x2	Function 2		
		0x3	Function 3		
		0x4	Function 4		
		0x5	Function 5		
		0x6	Function 6		
		0x7	Function 7		
3	EPD		Enable pull-down resistor at pad.	0	R/W
		0	Disable pull-down.		
		1	Enable pull-down. Enable both pull-down resistor and pull-up resistor for repeater mode.		

Table 125. Pin configuration registers for high-speed pins (SFS, address 0x4008 618C (SPSP3_3); 0x4008 6C00 (SFSCLK0) to 0x4008 6C0C (SFSCLK3)) bit description

Bit	Symbol	Value	Description	Reset value	Access
4	EPUN		Disable pull-up resistor at pad. By default, the pull-up resistor is enabled at reset.	0	R/W
		0	Enable pull-up. Enable both pull-down resistor and pull-up resistor for repeater mode.		
		1	Disable pull-up.		
5	EHS		Slew rate	0	R/W
		0	Fast (low noise with fast speed)		
		1	High-speed (medium noise with high speed)		
6	EZI		Input buffer enable. The input buffer is disabled by default at reset and must be enabled for receiving.	0	R/W
		0	Disable input buffer		
		1	Enable input buffer		
7	ZIF		Input glitch filter. Disable the input glitch filter for clocking signals higher than 30 MHz.	0	R/W
		0	Enable input filter		
		1	Disable input filter		
31:8	-		Reserved	-	-

15.4.4 Pin configuration register for USB1 pins USB1_DP/USB1_DM

Remark: The USB_ESEA bit must be set to one to use USB1.

Table 126. Pin configuration for pins USB1_DP/USB1_DM register (SFSUSB, address 0x4008 6C80) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	USB_AIM		Differential data input AIP/AIM.	0	R/W
		0	Going LOW with full speed edge rate		
		1	Going HIGH with full speed edge rate		
1	USB_ESEA		Control signal for differential input or single input.	1	R/W
		0	Reserved. Do not use.		
		1	Single input. Enables USB1. Use with the on-chip full-speed PHY.		
2	USB_EPD		Enable pull-down connect.	0	R/W
		0	Pull-down disconnected		
		1	Pull-down connected		
3	-		Reserved	-	-
4	USB_EPWR		Power mode.	0	R/W
		0	Power saving mode (Suspend mode)		
		1	Normal mode		

Table 126. Pin configuration for pins USB1_DP/USB1_DM register (SFSUSB, address 0x4008 6C80) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
5	USB_VBUS		Enable the vbus_valid signal. This signal is monitored by the USB1 block. Use this bit for software de-bouncing of the VBUS sense signal or to indicate the VBUS state to the USB1 controller when the VBUS signal is present but the USB1_VBUS function is not connected in the SFSP2_5 register. **Remark:** The setting of this bit has no effect if the USB1_VBUS function of pin P2_5 is enabled through the SFSP2_5 register.	0	R/W
		0	VBUS signal LOW or inactive		
		1	VBUS signal HIGH or active		
31:6	-		Reserved	-	-

15.4.5 Pin configuration register for open-drain I²C-bus pins

Table 127. Pin configuration for open-drain I²C-bus pins register (SFSI2C0, address 0x4008 6C84) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	SCL_EFP		Select input glitch filter time constant for the SCL pin.	0	R/W
		0	50 ns glitch filter		
		1	3 ns glitch filter		
1	-		Reserved. Always write a 0 to this bit.	0	R/W
2	SCL_EHD		Select I2C mode for the SCL pin.	0	R/W
		0	Standard/Fast mode transmit		
		1	Fast-mode Plus transmit		
3	SCL_EZI		Enable the input receiver for the SCL pin. Always write a 1 to this bit when using the I2C0.	0	R/W
		0	Disabled		
		1	Enabled		
6:4	-		Reserved	-	-
7	SCL_ZIF		Enable or disable input glitch filter for the SCL pin. The filter time constant is determined by bit EFP.	0	R/W
		0	Enable input filter		
		1	Disable input filter		
8	SDA_EFP		Select input glitch filter time constant for the SDA pin.	0	R/W
		0	50 ns glitch filter		
		1	3 ns glitch filter		
9	-		Reserved. Always write a 0 to this bit.	0	R/W

Table 127. Pin configuration for open-drain I²C-bus pins register (SFSI2C0, address 0x4008 6C84) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
10	SDA_EHD		Select I2C mode for the SDA pin.	0	R/W
		0	Standard/Fast mode transmit		
		1	Fast-mode Plus transmit		
11	SDA_EZI		Enable the input receiver for the SDA pin. Always write a 1 to this bit when using the I2C0.	0	R/W
		0	Disabled		
		1	Enabled		
14:12	-		Reserved	-	-
15	SDA_ZIF		Enable or disable input glitch filter for the SDA pin. The filter time constant is determined by bit SDA_EFP.	0	R/W
		0	Enable input filter		
		1	Disable input filter		
31:16	-		Reserved	-	-

15.4.6 ADC0 function select register

For pins with digital and analog functions, this register selects the input channel of the ADC0 over any of the possible digital functions. This option is not available for channel ADC0_7.

In addition, each analog function is pinned out on a dedicated analog pin which is not affected by this register.

The following pins are controlled by the ENAIO0 register:

Table 128. Pins controlled by the ENAIO0 register

Pin	ADC function	ENAIO0 register bit
P4_3	ADC0_0	0
P4_1	ADC0_1	1
PF_8	ADC0_2	2
P7_5	ADC0_3	3
P7_4	ADC0_4	4
PF_10	ADC0_5	5
PB_6	ADC0_6	6

By default, all pins are connected to their digital function 0 and only the digital pad is available.

To select the analog function, the pad must be set as follows using the corresponding SFSP register:

1. Tri-state the output driver by selecting an input at the pinmux e.g. GPIO function in input mode.

2. Disable the receiver by setting the EZI bit to zero (see Table 123 or Table 124). This is the default setting.
3. Disable the pull-up resistor by setting the EPUN bit to one, and disable the pull-down resistor by setting the EPD bit to zero.
4. Set the bit corresponding to the analog function to 1 in the ENAIO0 register.

Table 129. ADC0 function select register (ENAIO0, address 0x4008 6C88) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ADC0_0		Select ADC0_0	0	R/W
		0	Digital function selected on pin P4_3.		
		1	Analog function ADC0_0 selected on pin P4_3		
1	ADC0_1		Select ADC0_1	0	R/W
		0	Digital function selected on pin P4_1.		
		1	Analog function ADC0_1 selected on pin P4_1.		
2	ADC0_2		Select ADC0_2	0	R/W
		0	Digital function selected on pin PF_8.		
		1	Analog function ADC0_2 selected on pin PF_8.		
3	ADC0_3		Select ADC0_3	0	R/W
		0	Digital function selected on pin P7_5.		
		1	Analog function ADC0_3 selected on pin P7_5.		
4	ADC0_4		Select ADC0_4	0	R/W
		0	Digital function selected on pin P7_4.		
		1	Analog function ADC0_4 selected on pin P7_4.		
5	ADC0_5		Select ADC0_5	0	R/W
		0	Digital function selected on pin PF_10.		
		1	Analog function ADC0_5 selected on pin PF_10.		
6	ADC0_6		Select ADC0_6	0	R/W
		0	Digital function selected on pin PB_6.		
		1	Analog function ADC0_6 selected on pin PB_6.		
31:7			Reserved	-	-

15.4.7 ADC1 function select register

For pins with digital and analog functions, this register selects the ADC1 function over any of the possible digital functions.

In addition, each analog function is pinned out on a dedicated analog pin which is not affected by this register.

The following pins are controlled by the ENAIO1 register:

Table 130. Pins controlled by the ENAIO1 register

Pin	ADC function	ENAIO1 register bit
PC_3	ADC1_0	0
PC_0	ADC1_1	1
PF_9	ADC1_2	2

Table 130. Pins controlled by the ENAIO1 register

Pin	ADC function	ENAIO1 register bit
PF_6	ADC1_3	3
PF_5	ADC1_4	4
PF_11	ADC1_5	5
P7_7	ADC1_6	6
PF_7	ADC1_7	7

By default, all pins are connected to their digital function 0 and only the digital pad is available.

To select the analog function, the pad must be set as follows using the corresponding SFSP register:

1. Tri-state the output driver by selecting an input at the pinmux e.g. GPIO function in input mode.
2. Disable the receiver by setting the EZI bit to zero (see Table 123 or Table 124). This is the default setting.
3. Disable the pull-up resistor by setting the EPUN bit to one, and disable the pull-down resistor by setting the EPD bit to zero.
4. Set the bit corresponding to the analog function to 1 in the ENAIO1 register.

Table 131. ADC1 function select register (ENAIO1, address 0x4008 6C8C) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	ADC1_0		Select ADC1_0	0	R/W
		0	Digital function selected on pin PC_3.		
		1	Analog function ADC1_0 selected on pin PC_3.		
1	ADC1_1		Select ADC1_1	0	R/W
		0	Digital function selected on pin PC_0.		
		1	Analog function ADC1_1 selected on pin PC_0.		
2	ADC1_2		Select ADC1_2	0	R/W
		0	Digital function selected on pin PF_9.		
		1	Analog function ADC1_2 selected on pin PF_9.		
3	ADC1_3		Select ADC1_3	0	R/W
		0	Digital function selected on pin PF_6.		
		1	Analog function ADC1_3 selected on pin PF_6.		
4	ADC1_4		Select ADC1_4	0	R/W
		0	Digital function selected on pin PF_5.		
		1	Analog function ADC1_4 selected on pin PF_5.		
5	ADC1_5		Select ADC1_5	0	R/W
		0	Digital function selected on pin PF_11.		
		1	Analog function ADC1_5 selected on pin PF_11.		
6	ADC1_6		Select ADC1_6	0	R/W
		0	Digital function selected on pin P7_7.		
		1	Analog function ADC1_6 selected on pin P7_7.		

Table 131. ADC1 function select register (ENAIO1, address 0x4008 6C8C) bit description

Bit	Symbol	Value	Description	Reset value	Access
7	ADC1_7		Select ADC1_7.	0	R/W
		0	Digital function selected on pin PF_7.		
		1	Analog function ADC1_7 selected on pin PF_7.		
31:8			Reserved	-	-

15.4.8 Analog function select register

For pins which have digital and analog functions, this register selects the analog DAC and band gap function over any of the possible digital functions.

In addition, the DAC function is pinned out on a dedicated analog pin which is not affected by this register.

The following pins are controlled by the ENAIO2 register:

Table 132. Pins controlled by the ENAIO2 register

Pin	ADC function	ENAIO2 register bit
P4_4	DAC	0
PF_7	BG (band gap output)	4

By default, all pins are connected to their digital function 0 and only the digital pad is available.

To select the analog function, the pad must be set as follows using the corresponding SFSP register:

1. Tri-state the output driver by selecting an input at the pinmux e.g. GPIO function in input mode.
2. Disable the receiver by setting the EZI bit to zero (see Table 123 or Table 124). This is the default setting.
3. Disable the pull-up resistor by setting the EPUN bit to one, and disable the pull-down resistor by setting the EPD bit to zero.
4. Set the bit corresponding to the analog function to 1in the ENAIO2 register.

Table 133. Analog function select register (ENAIO2, address 0x4008 6C90) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DAC		Select DAC	0	R/W
		0	Digital function selected on pin P4_4.		
		1	Analog function DAC selected on pin P4_4.		
3:1			Reserved	-	-

Table 133. Analog function select register (ENAIO2, address 0x4008 6C90) bit description

Bit	Symbol	Value	Description	Reset value	Access
4	BG		Select band gap output. To measure the band gap, disable the pull-up on pin PF_7 and connect PF_7 to the digital pad. Do not use the digital pad nor the ADC1_7 on the board when measuring the band gap (see Section 15.4.8.1).	0	R/W
		0	Digital function selected on pin PF_7.		
		1	Band gap output selected for pin PF_7.		
31:5			Reserved	-	-

15.4.8.1 Measuring the band gap

To measure the band gap, set up the pin configuration and ADC function select registers for pin PF_7 as follows:

1. Disable the pull-up and the input buffer: Set register SFSPF_7 at 0x4008 679C to 0x10.
2. Connect the ADC1_7 input to the digital pad: Set register ENAIO1at 0x4008 6C8C to 0x80.
3. Connect the band gap to the digital pad: Set register ENAIO2 at 0x4008 6C90 to 0x10.
4. Do not connect pin PF_7 on the board.

15.4.9 EMC clock delay register

This register provides a programmable delay for the EMC clock outputs. The delay for all EMC_CLKn clock outputs is the same and increases in approximately 0.5 ns steps from 0 (CLK_DELAY = 0 to 3.5 ns (CLK_DELAY = 0x7777). The exact values of the delays vary over temperature and processing.

Table 134. EMC clock delay register (EMCDELAYCLK, address 0x4008 6D00) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CLK_DELAY	EMC_CLKn SDRAM clock output delay. 0x0 = no delay 0x1111 ≈ 0.5 ns delay 0x2222 ≈ 1.0 ns delay 0x3333 ≈ 1.5 ns delay 0x4444 ≈ 2.0 ns delay 0x5555 ≈ 2.5 ns delay 0x6666 ≈ 3.0 ns delay 0x7777 ≈ 3.5 ns delay	0	R/W
31:16	-	Reserved. Do not write ones to reserved register bits.	-	-

15.4.10 Pin interrupt select register 0

This register selects one GPIO pin from all GPIO pins on all ports as the source for pin interrupts 0 to 3.

Example: For pin interrupt 1, INTPIN1 = 0xA and PORTSEL1 = 1 select GPIO pin GPIO1[10] located on pin P2_9 to generate an interrupt. Each pin interrupt must be enabled in the NVIC.

To enable each pin interrupt and configure its edge or level sensitivity, use the GPIO pin interrupt registers (see Section 17.4.1).

Table 135. Pin interrupt select register 0 (PINTSEL0, address 0x4008 6E00) bit description

Bit	Symbol	Value	Description	Reset value
4:0	INTPIN0		Pint interrupt 0: Select the pin number within the GPIO port selected by the PORTSEL0 bit in this register.	0
7:5	PORTSEL0		Pin interrupt 0: Select the port for the pin number to be selected in the INTPIN0 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	
12:8	INTPIN1		Pint interrupt 1: Select the pin number within the GPIO port selected by the PORTSEL1 bit in this register.	0
15:13	PORTSEL1		Pin interrupt 1: Select the port for the pin number to be selected in the INTPIN1 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	
20:16	INTPIN2		Pint interrupt 2: Select the pin number within the GPIO port selected by the PORTSEL2 bit in this register.	0
23:21	PORTSEL2		Pin interrupt 2: Select the port for the pin number to be selected in the INTPIN2 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	

Table 135. Pin interrupt select register 0 (PINTSEL0, address 0x4008 6E00) bit description

Bit	Symbol	Value	Description	Reset value
28:24	INTPIN3		Pint interrupt 3: Select the pin number within the GPIO port selected by the PORTSEL3 bit in this register.	0
31:29	PORTSEL3		Pin interrupt 3: Select the port for the pin number to be selected in the INTPIN3 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	

15.4.11 Pin interrupt select register 1

This register selects one GPIO pin from all GPIO pins on all ports as the source for pin interrupts 4 to 7.

Example: For pin interrupt 4, INTPIN4 = 0xA and PORTSEL4 = 1 select GPIO pin GPIO1[10] located on pin P2_9 to generate an interrupt. Each pin interrupt must be enabled in the NVIC using interrupt slots 32 to 39.

To enable each pin interrupt and configure its edge or level sensitivity, use the GPIO pin interrupt registers (see Section 17.4.1).

Table 136. Pin interrupt select register 1 (PINTSEL1, address 0x4008 6E04) bit description

Bit	Symbol	Value	Description	Reset value
6:0	INTPIN4		Pint interrupt 4: Select the pin number within the GPIO port selected by the PORTSEL4 bit in this register.	0
7:5	PORTSEL4		Pin interrupt 4: Select the port for the pin number to be selected in the INTPIN4 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	
12:8	INTPIN5		Pint interrupt 5: Select the pin number within the GPIO port selected by the PORTSEL5 bit in this register.	0

Table 136. Pin interrupt select register 1 (PINTSEL1, address 0x4008 6E04) bit description

Bit	Symbol	Value	Description	Reset value
15:13	PORTSEL5		Pin interrupt 5: Select the port for the pin number to be selected in the INTPIN5 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	
20:16	INTPIN6		Pint interrupt 6: Select the pin number within the GPIO port selected by the PORTSEL6 bit in this register.	0
23:21	PORTSEL6		Pin interrupt 6: Select the port for the pin number to be selected in the INTPIN6 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	
28:24	INTPIN7		Pint interrupt 7: Select the pin number within the GPIO port selected by the PORTSEL7 bit in this register.	0
31:29	PORTSEL7		Pin interrupt 7: Select the port for the pin number to be selected in the INTPIN7 bits of this register.	0
		0x0	GPIO Port 0	
		0x1	GPIO Port 1	
		0x2	GPIO Port 2	
		0x3	GPIO Port 3	
		0x4	GPIO Port 4	
		0x5	GPIO Port 5	
		0x6	GPIO Port 6	
		0x7	GPIO Port 7	

16.1 How to read this chapter

Remark: The VADC block is not available on the LPC4350/30/20/10 Rev ‘A’.

16.2 Basic configuration

The GIMA is configured as follows:

- See Table 137 for clocking and power control.
- The GIMA is reset by the BUS_RST (reset #8). Do not reset the GIMA during normal operation
- The GIMA outputs are connected to the timer, SCT, ADC, and event router peripherals (see Figure 28).
- To configure the GIMA inputs for the timers and SCT, set the CTOUTCTRL bit in CREG6 (Table 43). This bit controls whether the SCT outputs are ORed with the timer match outputs or whether the SCT outputs only are used.

Table 137. GIMA clocking and power control

	Base clock	Branch clock	Maximum frequency
Clock to GIMA register interface	BASE_M4_CLK	CLK_M4_BUS	204 MHz

16.3 General description

The Global Input Multiplexer Array (GIMA) connects events to various event triggered peripherals such as the ADCs, the SCT, or the timers.

Each output of the GIMA is connected to a peripheral function (for example, a timer capture input or an ADC conversion trigger input) and configured through one register, which selects the event triggers and configures the clock synchronization.

For example, an ADC conversion can be triggered on either an SCT output or a timer match output. To select the trigger event, use GIMA output 28 which is connected to the ADC0 and ADC1 start0 conversion inputs. The corresponding GIMA output register ADCSTART0_IN selects SCT output 15 or the match output 0 of timer 0 as conversion triggers (see Table 138).

16.3.1 GIMA event input selection

Events that can trigger a peripheral function (e.g. an ADC conversion or a timer capture) can be selected from the following sources:

- Timer capture pins
- SCT input pins
- Timer0/1/2/3 match outputs
- SCT outputs

- I2S0/1 MWS signal
- USART0/2/3 RX/TX active signal
- USB0/1 SOF signal

The following peripheral functions are connected to GIMA outputs:

- Timer0/1/2/3 capture inputs
- SCT inputs
- ADC0/1 start of conversion
- Event router events

Figure 28 shows the peripherals which are connected through the GIMA. For details, see Table 138.

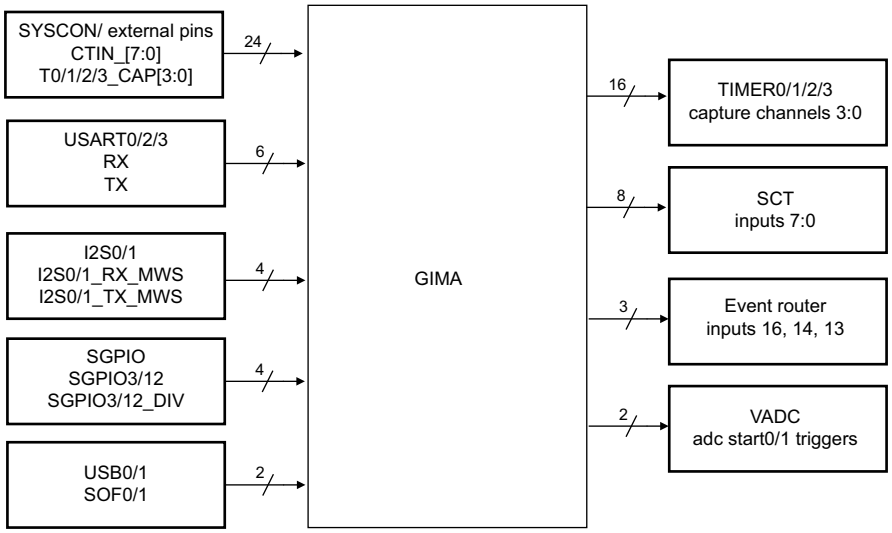


Fig 28. Connections between GIMA and peripherals

Table 138 shows the GIMA output number, the peripheral function the GIMA output is connected to, and the events that can be selected for this GIMA output. For signals that originate from an external pin, select a pin from the pinout (more than one pins may be possible) and program the corresponding pin function in the pin configuration register. Inputs from external pins include any of the timer capture pins (T0/1/2/3_CAPx and SCT input pins CTIN_x).

Each GIMA output uses one register (see Table 139) to control the synchronization stage and to select an event from a subset of all available event trigger signals as shown in Table 138.

Table 138. GIMA outputs

GIMA output	Peripheral	GIMA inputs				Reference
0	T0 capture channel 0	pin CTIN_0	SGPIO3	pin T0_CAP0	-	Table 140
1	T0 capture channel 1	pin CTIN_1	USART2 TX active	pin T0_CAP1	-	Table 141
2	T0 capture channel 2	pin CTIN_2	SGPIO3_DIV	pin T0_CAP2	-	Table 142
3	T0 capture channel 3	SCT output 15 or T3 match channel 3	pin T0_cap3	T3 match channel 3	-	Table 143
4	T1 capture channel 0	pin CTIN_0	SGPIO12	pin T1_CAP0	-	Table 144
5	T1 capture cannel 1	pin CTIN_3	USART0 TX active	pin T1_CAP1	-	Table 145
6	T1 capture channel 2	pin CTIN_4	USART0 RX active	pin T1_CAP2	-	Table 146
7	T1 capture channel 3	SCT output 3 or T0 match channel 3	pin T1_CAP3	T0 match channel 3	-	Table 147
8	T2 capture channel 0	pin CTIN_0	SGPIO12_DIV	pin T2_CAP0	-	Table 148
9	T2 capture channel 1	pin CTIN_1	USART2 TX active	I2S1_RX_MSW	pin T2_CAP1	Table 149
10	T2 capture channel 2	pin CTIN_5	USART2 RX active	I2S1_TX_MSW	pin T2_CAP2	Table 150
11	T2 capture channel 3	SCT output 7 or T1 match channel 3	pin T2_CAP3		-	Table 151
12	T3 capture channel 0	pin CTIN_0	I2S0_RX_MSW	pin T3_CAP0	-	Table 152
13	T3 capture channel 1	pin CTIN_6	USART3 TX active	I2S0_TX_MSW	pin T3_CAP1	Table 153
14	T3 capture channel 2	pin CTIN_7	USART3 RX active	SOF0	pin T3_CAP2	Table 154
15	T3 capture channel 3	SCT output 11 or T2 match channel 3	SOF1	pin T3_CAP3	-	Table 155
16	SCT input 0	pin CTIN_0	SGPIO3	SGPIO3_DIV	-	Table 156
17	SCT input 1	pin CTIN_1	USART2 TX active	SGPIO12	-	Table 157
18	SCT input 2	pin CTIN_2	SGPIO12	SGPIO12_DIV	-	Table 158
19	SCT input 3	pin CTIN_3	USART0 TX active	I2S1_RX_MSW	I2S1_TX_MSW	Table 159
20	SCT input 4	pin CTIN_4	USART0 RX active	I2S1_RX_MSW	I2S1_TX_MSW	Table 160
21	SCT input 5	pin CTIN_5	USART2 TX active	SGPIO12_DIV	-	Table 161
22	SCT input 6	pin CTIN_6	USART3 TX active	I2S0_RX_MSW	I2S0_TX_MSW	Table 162
23	SCT input 7	pin CTIN_7	USART3 RX active	SOF0	SOF1	Table 163
24	VADC trigger	see Table 164				Table 164
25	Event router input 13	SCT output 2 or T0 match channel 2	SGPIO3	T0 match channel 2	-	Table 165

Table 138. GIMA outputs

GIMA output	Peripheral	GIMA inputs				Reference
26	Event router input 14	SCT output 6 or T1 match channel 2	SGPIO12	T1 match channel 2	-	Table 166
27	Event router input 16	SCT output 14 or T3 match channel 2	T3 match channel 2	-	-	Table 167
28	ADC start0 input (ADC CR register START bits = 0x2)	SCT output 15 or T3 match channel 3	T0 match channel 0	-	-	Table 168
29	ADC start1 input (ADC CR register bit START = 0x3)	SCT output 8 or T2 match channel 0	T2 match channel 0	-	-	Table 169

16.3.2 GIMA clock synchronization

The clock synchronization control for each GIMA output consists of five stages (Figure 29):

1. Input selection
2. Input inversion: inverts the path between source and destination.
3. Asynchronous capture
4. Synchronization to peripheral clock
5. Pulse generation

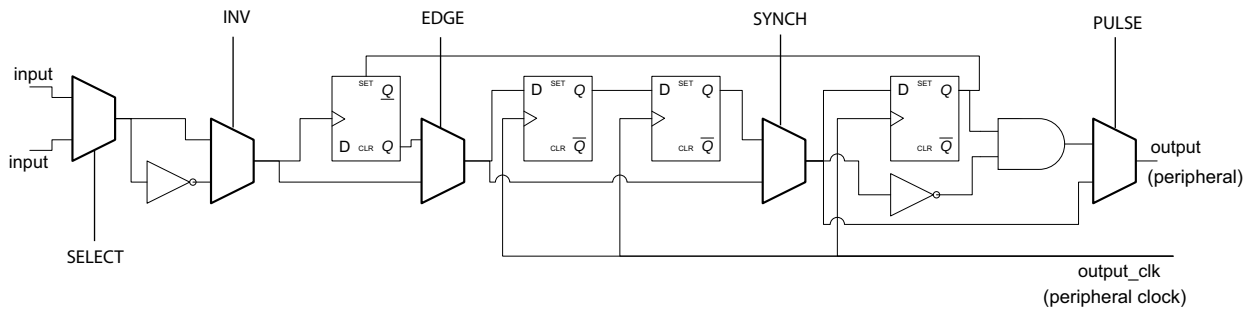


Fig 29. GIMA input stages

16.4 Register description

Table 139. Register overview: GIMA (base address: 0x400C 7000)

Name	Access	Address offset	Description	Reset value	Reference
CAP0_0_IN	R/W	0x000	Timer 0 CAP0_0 capture input multiplexer (GIMA output 0)	0	Table 140
CAP0_1_IN	R/W	0x004	Timer 0 CAP0_1 capture input multiplexer (GIMA output 1)	0	Table 141
CAP0_2_IN	R/W	0x008	Timer 0 CAP0_2 capture input multiplexer (GIMA output 2)	0	Table 142

Table 139. Register overview: GIMA (base address: 0x400C 7000)

Name	Access	Address offset	Description	Reset value	Reference
CAP0_3_IN	R/W	0x00C	Timer 0 CAP0_3 capture input multiplexer (GIMA output 3)	0	Table 143
CAP1_0_IN	R/W	0x010	Timer 1 CAP1_0 capture input multiplexer (GIMA output 4)	0	Table 144
CAP1_1_IN	R/W	0x014	Timer 1 CAP1_1 capture input multiplexer (GIMA output 5)	0	Table 145
CAP1_2_IN	R/W	0x018	Timer 1 CAP1_2 capture input multiplexer (GIMA output 6)	0	Table 146
CAP1_3_IN	R/W	0x01C	Timer 1 CAP1_3 capture input multiplexer (GIMA output 7)	0	Table 147
CAP2_0_IN	R/W	0x020	Timer 2 CAP2_0 capture input multiplexer (GIMA output 8)	0	Table 148
CAP2_1_IN	R/W	0x024	Timer 2 CAP2_1 capture input multiplexer (GIMA output 9)	0	Table 149
CAP2_2_IN	R/W	0x028	Timer 2 CAP2_2 capture input multiplexer (GIMA output 10)	0	Table 150
CAP2_3_IN	R/W	0x02C	Timer 2 CAP2_3 capture input multiplexer (GIMA output 11)	0	Table 151
CAP3_0_IN	R/W	0x030	Timer 3 CAP3_0 capture input multiplexer (GIMA output 12)	0	Table 152
CAP3_1_IN	R/W	0x034	Timer 3 CAP3_1 capture input multiplexer (GIMA output 13)	0	Table 153
CAP3_2_IN	R/W	0x038	Timer 3 CAP3_2 capture input multiplexer (GIMA output 14)	0	Table 154
CAP3_3_IN	R/W	0x03C	Timer 3 CAP3_3 capture input multiplexer (GIMA output 15)	0	Table 155
CTIN_0_IN	R/W	0x040	SCT CTIN_0 capture input multiplexer (GIMA output 16)	0	Table 156
CTIN_1_IN	R/W	0x044	SCT CTIN_1 capture input multiplexer (GIMA output 17)	0	Table 157
CTIN_2_IN	R/W	0x048	SCT CTIN_2 capture input multiplexer (GIMA output 18)	0	Table 158
CTIN_3_IN	R/W	0x04C	SCT CTIN_3 capture input multiplexer (GIMA output 19)	0	Table 159
CTIN_4_IN	R/W	0x050	SCT CTIN_4 capture input multiplexer (GIMA output 20)	0	Table 160
CTIN_5_IN	R/W	0x054	SCT CTIN_5 capture input multiplexer (GIMA output 21)	0	Table 161
CTIN_6_IN	R/W	0x058	SCT CTIN_6 capture input multiplexer (GIMA output 22)	0	Table 162
CTIN_7_IN	R/W	0x05C	SCT CTIN_7 capture input multiplexer (GIMA output 23)	0	Table 163
VADC_TRIGGER_IN	R/W	0x060	VADC trigger input multiplexer (GIMA output 24)	0	Table 164
EVENTROUTER_13_IN	R/W	0x064	Event router input 13 multiplexer (GIMA output 25)	0	Table 165
EVENTROUTER_14_IN	R/W	0x068	Event router input 14 multiplexer (GIMA output 26)	0	Table 166

Table 139. Register overview: GIMA (base address: 0x400C 7000)

Name	Access	Address offset	Description	Reset value	Reference
EVENTROUTER_16_IN	R/W	0x06C	Event router input 16 multiplexer (GIMA output 27)	0	Table 167
ADCSTART0_IN	R/W	0x070	ADC start0 input multiplexer (GIMA output 28)	0	Table 168
ADCSTART1_IN	R/W	0x074	ADC start1 input multiplexer (GIMA output 29)	0	Table 169

16.4.1 Timer 0 CAP0_0 capture input multiplexer (CAP0_0_IN)

Table 140. Timer 0 CAP0_0 capture input multiplexer (CAP0_0_IN, address 0x400C 7000) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_0	
		0x1	SGPIO3	
		0x2	T0_CAP0	
31:8	-		Reserved	-

16.4.2 Timer 0 CAP0_1 capture input multiplexer (CAP0_1_IN)

Table 141. Timer 0 CAP0_1 capture input multiplexer (CAP0_1_IN, address 0x400C 7004) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	

Table 141. Timer 0 CAP0_1 capture input multiplexer (CAP0_1_IN, address 0x400C 7004) bit description

Bit	Symbol	Value	Description	Reset value
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_1	
		0x1	USART2 TX active	
		0x2	T0_CAP1	
31:8	-		Reserved	-

16.4.3 Timer 0 CAP0_2 capture input multiplexer (CAP0_2_IN)

Table 142. Timer 0 CAP0_2 capture input multiplexer (CAP0_2_IN, address 0x400C 7008) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_2	
		0x1	SGPIO3_DIV	
		0x2	T0_CAP2	
31:8	-		Reserved	-

16.4.4 Timer 0 CAP0_3 capture input multiplexer (CAP0_3_IN)

Table 143. Timer 0 CAP0_3 capture input multiplexer (CAP0_3_IN, address 0x400C 700C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTOUT_15 or T3_MAT3	
		0x1	T0_CAP3	
		0x2	T3_MAT3	
31:8	-		Reserved	-

16.4.5 Timer 1 CAP1_0 capture input multiplexer (CAP1_0_IN)

Table 144. Timer 1 CAP1_0 capture input multiplexer (CAP1_0_IN, address 0x400C 7010) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	

Table 144. Timer 1 CAP1_0 capture input multiplexer (CAP1_0_IN, address 0x400C 7010) bit description

Bit	Symbol	Value	Description	Reset value
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_0	
		0x1	SGPIO12	
		0x2	T1_CAP0	
31:8	-		Reserved	-

16.4.6 Timer 1 CAP1_1 capture input multiplexer (CAP1_1_IN)

Table 145. Timer 1 CAP1_1 capture input multiplexer (CAP1_1_IN, address 0x400C 7014) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_3	
		0x1	USART0 TX active	
		0x2	T1_CAP1	
31:8	-		Reserved	-

16.4.7 Timer 1 CAP1_2 capture input multiplexer (CAP1_2_IN)

Table 146. Timer 1 CAP1_2 capture input multiplexer (CAP1_2_IN, address 0x400C 7018) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	

Table 146. Timer 1 CAP1_2 capture input multiplexer (CAP1_2_IN, address 0x400C 7018) bit description

Bit	Symbol	Value	Description	Reset value
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_4	
		0x1	USART0 RX active	
		0x2	T1_CAP2	
31:8	-		Reserved	-

16.4.8 Timer 1 CAP1_3 capture input multiplexer (CAP1_3_IN)

Table 147. Timer 1 CAP1_3 capture input multiplexer (CAP1_3_IN, address 0x400C 701C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTOUT_3 or T0_MAT3	
		0x1	T1_CAP3	
		0x2	T0_MAT3	
31:8	-		Reserved	-

16.4.9 Timer 2 CAP2_0 capture input multiplexer (CAP2_0_IN)

Table 148. Timer 2 CAP2_0 capture input multiplexer (CAP2_0_IN, address 0x400C 7020) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_0	
		0x1	SGPIO12_DIV	
		0x2	T2_CAP0	
31:8	-		Reserved	-

16.4.10 Timer 2 CAP2_1 capture input multiplexer (CAP2_1_IN)

Table 149. Timer 2 CAP2_1 capture input multiplexer (CAP2_1_IN, address 0x400C 7024) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	

Table 149. Timer 2 CAP2_1 capture input multiplexer (CAP2_1_IN, address 0x400C 7024) bit description

Bit	Symbol	Value	Description	Reset value
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_1	
		0x1	USART2 TX active	
		0x2	<tbd> - I2S1_RX_MWS	
		0x3	T2_CAP1	
31:8	-		Reserved	-

16.4.11 Timer 2 CAP2_2 capture input multiplexer (CAP2_2_IN)

Table 150. Timer 2 CAP2_2 capture input multiplexer (CAP2_2_IN, address 0x400C 7028) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_5	
		0x1	USART2 RX active	
		0x2	<tbd> - I2S1_TX_MWS	
		0x3	T2_CAP2	
31:8	-		Reserved	-

16.4.12 Timer 2 CAP2_3 capture input multiplexer (CAP2_3_IN)

Table 151. Timer 2 CAP2_3 capture input multiplexer (CAP2_3_IN, address 0x400C 702C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	

Table 151. Timer 2 CAP2_3 capture input multiplexer (CAP2_3_IN, address 0x400C 702C) bit description

Bit	Symbol	Value	Description	Reset value
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTOUT_7 or T1_MAT3	
		0x1	T2_CAP3	
		0x2	T1_MAT3	
31:8	-		Reserved	-

16.4.13 Timer 3 CAP3_0 capture input multiplexer (CAP3_0_IN)

Table 152. Timer 3 CAP3_0 capture input multiplexer (CAP3_0_IN, address 0x400C 7030) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_0	
		0x1	<tbd> I2S0_RX_MWS	
		0x2	T3_CAP0	
31:8	-		Reserved	-

16.4.14 Timer 3 CAP3_1 capture input multiplexer (CAP3_1_IN)

Table 153. Timer 3 CAP3_1 capture input multiplexer (CAP3_1_IN, address 0x400C 7034) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_6	
		0x1	USART3 TX active	
		0x2	TBD - I2S0_TX_MWS	
		0x3	T3_CAP1	
31:8	-		Reserved	-

16.4.15 Timer 3 CAP3_2 capture input multiplexer (CAP3_2_IN)

Table 154. Timer 3 CAP3_2 capture input multiplexer (CAP3_2_IN, address 0x400C 7038) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	

Table 154. Timer 3 CAP3_2 capture input multiplexer (CAP3_2_IN, address 0x400C 7038) bit description

Bit	Symbol	Value	Description	Reset value
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_7	
		0x1	USART3 RX active	
		0x2	SOF0 (Start-Of-Frame USB0)	
		0x3	T3_CAP2	
31:8	-		Reserved	-

16.4.16 Timer 3 CAP3_3 capture input multiplexer (CAP3_3_IN)

Table 155. Timer 3 CAP3_3 capture input multiplexer (CAP3_3_IN, address 0x400C 703C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTOUT11 or T2_MAT3	
		0x1	SOF1 (Start-Of-Frame USB1)	
		0x2	T3_CAP3	
		0x3	T2_MAT3	
31:8	-		Reserved	-

16.4.17 SCT CTIN_0 capture input multiplexer (CTIN_0_IN)

Table 156. SCT CTIN_0 capture input multiplexer (CTIN_0_IN, address 0x400C 7040) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	

Table 156. SCT CTIN_0 capture input multiplexer (CTIN_0_IN, address 0x400C 7040) bit description

Bit	Symbol	Value	Description	Reset value
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_0	
		0x1	SGPIO3	
		0x2	SGPIO3_DIV	
31:8	-		Reserved	-

16.4.18 SCT CTIN_1 capture input multiplexer (CTIN_1_IN)

Table 157. SCT CTIN_1 capture input multiplexer (CTIN_1_IN, address 0x400C 7044) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_1	
		0x1	USART2 TX active	
		0x2	SGPIO12	
31:8	-		Reserved	-

16.4.19 SCT CTIN_2 capture input multiplexer (CTIN_2_IN)

Table 158. SCT CTIN_2 capture input multiplexer (CTIN_2_IN, address 0x400C 7048) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_2	
		0x1	SGPIO12	
		0x2	SGPIO12_DIV	
31:8	-		Reserved	-

16.4.20 SCT CTIN_3 capture input multiplexer (CTIN_3_IN)

Table 159. SCT CTIN_3 capture input multiplexer (CTIN_3_IN, address 0x400C 704C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	

Table 159. SCT CTIN_3 capture input multiplexer (CTIN_3_IN, address 0x400C 704C) bit description

Bit	Symbol	Value	Description	Reset value
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_3	
		0x1	USART0 TX active	
		0x2	Reserved	
		0x3	Reserved	
31:8	-		Reserved	-

16.4.21 SCT CTIN_4 capture input multiplexer (CTIN_4_IN)

Table 160. SCT CTIN_4 capture input multiplexer (CTIN_4_IN, address 0x400C 7050) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_4	
		0x1	USART0 RX active	
		0x2	<tbd> - I2S1_RX_MWS1	
		0x3	<tbd> - I2S1_TX_MWS1	
31:8	-		Reserved	-

16.4.22 SCT CTIN_5 capture input multiplexer (CTIN_5_IN)

Table 161. SCT CTIN_5 capture input multiplexer (CTIN_5_IN, address 0x400C 7054) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	

Table 161. SCT CTIN_5 capture input multiplexer (CTIN_5_IN, address 0x400C 7054) bit description

Bit	Symbol	Value	Description	Reset value
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTIN_5	
		0x1	USART2 RX active	
		0x2	SGPIO12_DIV	
31:8	-		Reserved	-

16.4.23 SCT CTIN_6 capture input multiplexer (CTIN_6_IN)

Table 162. SCT CTIN_6 capture input multiplexer (CTIN_6_IN, address 0x400C 7058) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_6	
		0x1	USART3 TX active	
		0x2	<tbd> - I2S0_RX_MWS	
		0x3	<tbd> - I2S0_TX_MWS	
31:8	-		Reserved	-

16.4.24 SCT CTIN_7 capture input multiplexer (CTIN_7_IN)

Table 163. SCT CTIN_7 capture input multiplexer (CTIN_7_IN, address 0x400C 705C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x4 to 0xF are reserved.	0
		0x0	CTIN_7	
		0x1	USART3 RX active	
		0x2	SOF0 (Start-Of-Frame USB0)	
		0x3	SOF1 (Start-Of-Frame USB1)	
31:8	-		Reserved	-

16.4.25 VADC trigger input multiplexer (VADC_TRIGGER_IN)

Table 164. ADC trigger input multiplexer (VADC_TRIGGER_IN, address 0x400C 7060) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	

Table 164. ADC trigger input multiplexer (VADC_TRIGGER_IN, address 0x400C 7060) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
7:4	SELECT		Select input. Values 0xA to 0xF are reserved.	0
		0x0	GPIO6[28]	
		0x1	GPIO5[3]	
		0x2	SGPIO10	
		0x3	SGPIO12	
		0x4	Reserved	
		0x5	MCOB2	
		0x6	CTOUT_0 or T0_MAT0	
		0x7	CTOUT_8 or T2_MAT0	
		0x8	T0_MAT0	
		0x9	T2_MAT0	
31:8	-		Reserved	-

16.4.26 Event router input 13 multiplexer (EVENTROUTER_13_IN)

Table 165. Event router input 13 multiplexer (EVENTROUTER_13_IN, address 0x400C 7064) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTOUT_2 or T0_MAT2	
		0x1	SGPIO3	
		0x2	T0_MAT2	
31:8	-		Reserved	-

16.4.27 Event router input 14 multiplexer (EVENTROUTER_14_IN)

Table 166. Event router input 14 multiplexer (EVENTROUTER_14_IN, address 0x400C 7068) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x3 to 0xF are reserved.	0
		0x0	CTOUT_6 or T1_MAT2	
		0x1	SGPIO12	
		0x2	T1_MAT2	
31:8	-		Reserved	-

16.4.28 Event router input 16 multiplexer (EVENTROUTER_16_IN)

Table 167. Event router input 16multiplexer (EVENTROUTER_16_IN, address 0x400C 706C) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x2 to 0xF are reserved.	0
		0x0	CTOUT_14 or T3_MAT2	
		0x1	T3_MAT2	
31:8	-		Reserved	-

16.4.29 ADC start0 input multiplexer (ADCSTART0_IN)

Table 168. ADC start0 input multiplexer (ADCSTART0_IN, address 0x400C 7070) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x2 to 0xF are reserved.	0
		0x0	CTOUT_15 or T3_MAT3	
		0x1	T0_MAT0	
31:8	-		Reserved	-

16.4.30 ADC start1 input multiplexer (ADCSTART1_IN)

Table 169. ADC start1 input multiplexer (ADCSTART1_IN, address 0x400C 7074) bit description

Bit	Symbol	Value	Description	Reset value
0	INV		Invert input	0
		0	Not inverted.	
		1	Input inverted.	
1	EDGE		Enable rising edge detection	0
		0	No edge detection.	
		1	Rising edge detection enabled.	
2	SYNCH		Enable synchronization	0
		0	Disable synchronization.	
		1	Enable synchronization.	
3	PULSE		Enable single pulse generation.	0
		0	Disable single pulse generation.	
		1	Enable single pulse generation.	
7:4	SELECT		Select input. Values 0x2 to 0xF are reserved.	0
		0x0	CTOUT_8 or T2_MAT0	
		0x1	T2_MAT0	
31:8	-		Reserved	-

17.1 How to read this chapter

All GPIO register bit descriptions refer to up to 31 pins on each GPIO port. Depending on the package type, not all pins are available, and the corresponding bits in the GPIO registers are reserved (see Table 170).

Table 170. GPIO pins for different pin packages

	LBGA256	TFBGA180	TFBGA100	LQFP208	LQFP144	LQFP100
GPIO Port 0	GPIO0[15:0]	GPIO0[15:0]	GPIO0[4:0]; GPIO0[15:6]	GPIO0[15:0]	GPIO0[15:0]	GPIO0[4:0]; GPIO0[15:6]
GPIO Port 1	GPIO1[15:0]	GPIO1[15:0]	GPIO1[15:0]	GPIO1[15:0]	GPIO1[15:0]	GPIO1[15:0]
GPIO Port 2	GPIO2[15:0]	GPIO2[15:0]	-	GPIO2[15:0]	GPIO2[15:0]	-
GPIO Port 3	GPIO3[15:0]	GPIO3[15:0]	GPIO3[1:0]; GPIO3[5:3]; GPIO3[7]	GPIO3[15:0]	GPIO3[15:0]	GPIO3[1:0]; GPIO3[5:3]; GPIO3[7]
GPIO Port 4	GPIO4[15:0]	GPIO4[15:0]	-	GPIO4[15:0]	GPIO4[11]	-
GPIO Port 5	GPIO5[26:0]	GPIO5[26:0]	GPIO5[11:0]	GPIO5[25:0]	GPIO5[16:0]; GPIO5[18]	GPIO5[11:0]
GPIO Port 6	GPIO6[30:0]	GPIO6[30:25]	-	GPIO6[5:0]; GPIO6[30:20]	-	-
GPIO Port 7	GPIO7[25:0]	GPIO7[4:0]	-	GPIO7[10:0]; GPIO7[21:17]; GPIO7[25:23]	-	-

17.2 Basic configuration

The GPIO blocks share a common clock and reset connection and are configured as follows:

- See Table 171 for clocking and power control.
- The GPIO is reset by a GPIO_RST (reset #28).
- All GPIO pins are set to input by default.
- For the pin interrupts, select up to 8 external interrupt pins from all GPIO port pins in the SCU (see Table 135 and Table 136). The pin interrupts must be enabled in the NVIC (see Table 21).
- The GPIO group interrupts must be enabled in the NVIC (see Table 21).
- GPIO port registers can be accessed by the GPDMA as memory-to-memory transfer.

Table 171. GPIO clocking and power control

	Base clock	Branch clock	Operating frequency
GPIO, GPIO pin interrupt, GPIO group0 interrupt, GPIO group1 interrupt	BASE_M4_CLK	CLK_M4_GPIO	up to 204 MHz

17.3 Features

17.3.1 GPIO pin interrupt features

- Up to 8 pins can be selected from all GPIO pins as edge- or level-sensitive interrupt requests. Each request creates a separate interrupt in the NVIC.
- Edge-sensitive interrupt pins can interrupt on rising or falling edges or both.
- Level-sensitive interrupt pins can be HIGH- or LOW-active.

17.3.2 GPIO group interrupt features

- The inputs from any number of GPIO pins can be enabled to contribute to a combined group interrupt.
- The polarity of each input enabled for the group interrupt can be configured HIGH or LOW.
- Enabled interrupts can be logically combined through an OR or AND operation.
- Two group interrupts are supported to reflect two distinct interrupt patterns.
- The GPIO group interrupts can wake up the part from sleep, deep-sleep or power-down modes.

17.3.3 GPIO port features

- GPIO pins can be configured as input or output by software.
- All GPIO pins default to inputs with interrupt disabled at reset.
- Pin registers allow pins to be sensed and set individually.

17.4 Introduction

The GPIO pins can be used in several ways to set pins as inputs or outputs and use the inputs as combinations of level and edge sensitive interrupts.

17.4.1 GPIO pin interrupts

From all available GPIO pins, up to eight pins can be selected in the system control block to serve as external interrupt pins (see <tbd>). The external interrupt pins are connected to eight individual interrupts in the NVIC and are created based on rising or falling edges or on the input level on the pin.

17.4.2 GPIO group interrupt

For each port/pin connected to one of the two the GPIO Grouped Interrupt blocks (GROUP0 and GROUP1), the GPIO grouped interrupt registers determine which pins are enabled to generate interrupts and what the active polarities of each of those inputs are.

The GPIO grouped interrupt registers also select whether the interrupt output will be level or edge triggered and whether it will be based on the OR or the AND of all of the enabled inputs.

When the designated pattern is detected on the selected input pins, the GPIO grouped interrupt block will generate an interrupt. If the part is in a power-savings mode it will first asynchronously wake the part up prior to asserting the interrupt request. The interrupt request line can be cleared by writing a one to the interrupt status bit in the control register.

17.4.3 GPIO port

The GPIO port registers can be used to configure each GPIO pin as input or output and read the state of each pin if the pin is configured as input or set the state of each pin if the pin is configured as output.

17.5 Register description

The GPIO consists of the following blocks:

- The GPIO pin interrupts block at address 0x4008 7000. Registers in this block enable the up to 8 pin interrupts selected in Table 135 or Table 136 and configure the level and edge sensitivity for each selected pin interrupt. The GPIO interrupt registers are listed in <tbd>Table 176 to Table 185.
- The GPIO GROUP0 interrupt block at address 0x4008 8000. Registers in this block allow to configure any pin on port 0 and 1 to contribute to a combined interrupt. The GPIO GROUP0 registers are listed in Table 173 and Section 17.5.2.
- The GPIO GROUP1 interrupt block at address 0x4008 9000. Registers in this block allow to configure any pin on port 0 and 1 to contribute to a combined interrupt. The GPIO GROUP1 registers are listed in Table 174 and Section 17.5.2.
- The GPIO port block at address 0x400F 4000. Registers in this block allow to read and write to port pins and configure port pins as inputs or outputs.The GPIO port registers are listed in Table 175 and Section 17.5.3.

Note: In all GPIO registers, bits that are not shown are **reserved**.

Table 172. Register overview: GPIO pin interrupts (base address: 0x4008 7000)

Name	Access	Address offset	Description	Reset value	Reference
ISEL	R/W	0x000	Pin Interrupt Mode register	0	Table 176
IENR	R/W	0x004	Pin Interrupt Enable (Rising) register	0	Table 177
SIENR	WO	0x008	Set Pin Interrupt Enable (Rising) register	NA	Table 178
CIENR	WO	0x00C	Clear Pin Interrupt Enable (Rising) register	NA	Table 179
IENF	R/W	0x010	Pin Interrupt Enable Falling Edge / Active Level register	0	Table 180
SIENF	WO	0x014	Set Pin Interrupt Enable Falling Edge / Active Level register	NA	Table 181
CIENF	WO	0x018	Clear Pin Interrupt Enable Falling Edge / Active Level address	NA	Table 182
RISE	R/W	0x01C	Pin Interrupt Rising Edge register	0	Table 183
FALL	R/W	0x020	Pin Interrupt Falling Edge register	0	Table 184
IST	R/W	0x024	Pin Interrupt Status register	0	Table 185

Table 173. Register overview: GPIO GROUP0 interrupt (base address 0x4008 8000)

Name	Access	Address offset	Description	Reset value	Reference
CTRL	R/W	0x000	GPIO grouped interrupt control register	0	Table 186
PORT_POL0	R/W	0x020	GPIO grouped interrupt port 0 polarity register	0xFFFF FFFF	Table 187
PORT_POL1	R/W	0x024	GPIO grouped interrupt port 1 polarity register	0xFFFF FFFF	Table 187
PORT_POL2	R/W	0x028	GPIO grouped interrupt port 2 polarity register	0xFFFF FFFF	Table 187
PORT_POL3	R/W	0x02C	GPIO grouped interrupt port 3 polarity register	0xFFFF FFFF	Table 187
PORT_POL4	R/W	0x030	GPIO grouped interrupt port 4 polarity register	0xFFFF FFFF	Table 187
PORT_POL5	R/W	0x034	GPIO grouped interrupt port 5 polarity register	0xFFFF FFFF	Table 187

Table 173. Register overview: GPIO GROUP0 interrupt (base address 0x4008 8000)

Name	Access	Address offset	Description	Reset value	Reference
PORT_POL6	R/W	0x038	GPIO grouped interrupt port 6 polarity register	0xFFFF FFFF	Table 187
PORT_POL7	R/W	0x03C	GPIO grouped interrupt port 7 polarity register	0xFFFF FFFF	Table 187
PORT_ENA0	R/W	0x040	GPIO grouped interrupt port 0 enable register	0	Table 188
PORT_ENA1	R/W	0x044	GPIO grouped interrupt port 1 enable register	0	Table 188
PORT_ENA2	R/W	0x048	GPIO grouped interrupt port 2 enable register	0	Table 188
PORT_ENA3	R/W	0x04C	GPIO grouped interrupt port 3 enable register	0	Table 188
PORT_ENA4	R/W	0x050	GPIO grouped interrupt port 4 enable register	0	Table 188
PORT_ENA5	R/W	0x054	GPIO grouped interrupt port 5 enable register	0	Table 188
PORT_ENA6	R/W	0x058	GPIO grouped interrupt port 5 enable register	0	Table 188
PORT_ENA7	R/W	0x05C	GPIO grouped interrupt port 5 enable register	0	Table 188

Table 174. Register overview: GPIO GROUP1 interrupt (base address 0x4008 9000)

Name	Access	Address offset	Description	Reset value	Reference
CTRL	R/W	0x000	GPIO grouped interrupt control register	0	Table 186
PORT_POL0	R/W	0x020	GPIO grouped interrupt port 0 polarity register	0xFFFF FFFF	Table 187
PORT_POL1	R/W	0x024	GPIO grouped interrupt port 1 polarity register	0xFFFF FFFF	Table 187
PORT_POL2	R/W	0x028	GPIO grouped interrupt port 2 polarity register	0xFFFF FFFF	Table 187
PORT_POL3	R/W	0x02C	GPIO grouped interrupt port 3 polarity register	0xFFFF FFFF	Table 187
PORT_POL4	R/W	0x030	GPIO grouped interrupt port 4 polarity register	0xFFFF FFFF	Table 187
PORT_POL5	R/W	0x034	GPIO grouped interrupt port 5 polarity register	0xFFFF FFFF	Table 187
PORT_POL6	R/W	0x038	GPIO grouped interrupt port 6 polarity register	0xFFFF FFFF	Table 187
PORT_POL7	R/W	0x03C	GPIO grouped interrupt port 7 polarity register	0xFFFF FFFF	Table 187
PORT_ENA0	R/W	0x040	GPIO grouped interrupt port 0 enable register	0	Table 188
PORT_ENA1	R/W	0x044	GPIO grouped interrupt port 1 enable register	0	Table 188
PORT_ENA2	R/W	0x048	GPIO grouped interrupt port 2 enable register	0	Table 188
PORT_ENA3	R/W	0x04C	GPIO grouped interrupt port 3 enable register	0	Table 188
PORT_ENA4	R/W	0x050	GPIO grouped interrupt port 4 enable register	0	Table 188
PORT_ENA5	R/W	0x054	GPIO grouped interrupt port 5 enable register	0	Table 188
PORT_ENA6	R/W	0x058	GPIO grouped interrupt port 5 enable register	0	Table 188
PORT_ENA7	R/W	0x05C	GPIO grouped interrupt port 5 enable register	0	Table 188

GPIO port addresses can be read and written as bytes, halfwords, or words.

Table 175. Register overview: GPIO port (base address 0x400F 4000)

x denotes the highest pin number on each port and depends on package size (see Table 170).

Name	Access	Address offset	Description	Reset value	Width	Reference
B0 to B31	R/W	0x0000 to x001F	Byte pin registers port 0; pins PIO0_0 to PIO0_31	ext[1]	byte (8 bit)	Table 189
B32 to Bx	R/W	0x0020 to 0x003F	Byte pin registers port 1	ext[1]	byte (8 bit)	Table 189
B64 to Bx	R/W	0x0040 to 0x005F	Byte pin registers port 2	ext[1]	byte (8 bit)	

Table 175. Register overview: GPIO port (base address 0x400F 4000)

x denotes the highest pin number on each port and depends on package size (see Table 170).

Name	Access	Address offset	Description	Reset value	Width	Reference
B96 to Bx	R/W	0x0060 to 0x007F	Byte pin registers port 3	ext[1]	byte (8 bit)	Table 189
B128 to Bx	R/W	0x0080 to 0x009F	Byte pin registers port 4	ext[1]	byte (8 bit)	Table 189
B160 to Bx	R/W	0x00A0 to 0x00BF	Byte pin registers port 5	ext[1]	byte (8 bit)	Table 189
B192 to Bx	R/W	0x00C0 to0x00DF	Byte pin registers port 6	ext[1]	byte (8 bit)	Table 189
B224 to Bx	R/W	0x00E0 to 0x00FC	Byte pin registers port 7	ext[1]	byte (8 bit)	Table 189
W0 to Wx	R/W	0x1000 to 0x107C	Word pin registers port 0	ext[1]	word (32 bit)	Table 190
W32 to Wx	R/W	0x1080 to 0x10FC	Word pin registers port 1	ext[1]	word (32 bit)	Table 190
W64 to Wx	R/W	0x1100 to 0x11FC	Word pin registers port 2	ext[1]	word (32 bit)	Table 190
W96 to Wx	R/W	0x1180 to 0x11FC	Word pin registers port 3	ext[1]	word (32 bit)	Table 190
W128 to Wx	R/W	0x1200 to 0x12FC	Word pin registers port 4	ext[1]	word (32 bit)	Table 190
W160 to Wx	R/W	0x1280 to 0x12FC	Word pin registers port 5	ext[1]	word (32 bit)	Table 190
W192 to Wx	R/W	0x1300 to 0x137C	Word pin registers port 6	ext[1]	word (32 bit)	Table 190
W224 to Wx	R/W	0x1380 to 0x13FC	Word pin registers port 7	ext[1]	word (32 bit)	Table 190
DIR0	R/W	0x2000	Direction registers port 0	0	word (32 bit)	Table 191
DIR1	R/W	0x2004	Direction registers port 1	0	word (32 bit)	Table 191
DIR2	R/W	0x2008	Direction registers port 2	0	word (32 bit)	Table 191
DIR3	R/W	0x200C	Direction registers port 3	0	word (32 bit)	Table 191
DIR4	R/W	0x2010	Direction registers port 4	0	word (32 bit)	Table 191
DIR5	R/W	0x2014	Direction registers port 5	0	word (32 bit)	Table 191
DIR6	R/W	0x2018	Direction registers port 6	0	word (32 bit)	Table 191
DIR7	R/W	0x201C	Direction registers port 7	0	word (32 bit)	Table 191
MASK0	R/W	0x2080	Mask register port 0	0	word (32 bit)	Table 192
MASK1	R/W	0x2084	Mask register port 1	0	word (32 bit)	Table 192
MASK2	R/W	0x2088	Mask register port 2	0	word (32 bit)	Table 192
MASK3	R/W	0x208C	Mask register port 3	0	word (32 bit)	Table 192
MASK4	R/W	0x2090	Mask register port 4	0	word (32 bit)	Table 192
MASK5	R/W	0x2094	Mask register port 5	0	word (32 bit)	Table 192
MASK6	R/W	0x2098	Mask register port 6	0	word (32 bit)	Table 192
MASK7	R/W	0x209C	Mask register port 7	0	word (32 bit)	Table 192
PIN0	R/W	0x2100	Port pin register port 0	ext[1]	word (32 bit)	Table 193
PIN1	R/W	0x2104	Port pin register port 1	ext[1]	word (32 bit)	Table 193
PIN2	R/W	0x2108	Port pin register port 2	ext[1]	word (32 bit)	Table 193
PIN3	R/W	0x210C	Port pin register port 3	ext[1]	word (32 bit)	Table 193
PIN4	R/W	0x2110	Port pin register port 4	ext[1]	word (32 bit)	Table 193
PIN5	R/W	0x2114	Port pin register port 5	ext[1]	word (32 bit)	Table 193
PIN6	R/W	0x2118	Port pin register port 6	ext[1]	word (32 bit)	Table 193
PIN7	R/W	0x211C	Port pin register port 7	ext[1]	word (32 bit)	Table 193
MPIN0	R/W	0x2180	Masked port register port 0	ext[1]	word (32 bit)	Table 194
MPIN1	R/W	0x2184	Masked port register port 1	ext[1]	word (32 bit)	Table 194

Table 175. Register overview: GPIO port (base address 0x400F 4000)

x denotes the highest pin number on each port and depends on package size (see Table 170).

Name	Access	Address offset	Description	Reset value	Width	Reference
MPIN2	R/W	0x2188	Masked port register port 2	ext[1]	word (32 bit)	Table 194
MPIN3	R/W	0x218C	Masked port register port 3	ext[1]	word (32 bit)	Table 194
MPIN4	R/W	0x2190	Masked port register port 4	ext[1]	word (32 bit)	Table 194
MPIN5	R/W	0x2194	Masked port register port 5	ext[1]	word (32 bit)	Table 194
MPIN6	R/W	0x2198	Masked port register port 6	ext[1]	word (32 bit)	Table 194
MPIN7	R/W	0x219C	Masked port register port 7	ext[1]	word (32 bit)	Table 194
SET0	R/W	0x2200	Write: Set register for port 0 Read: output bits for port 0	0	word (32 bit)	Table 195
SET1	R/W	0x2204	Write: Set register for port 1 Read: output bits for port 1	0	word (32 bit)	Table 195
SET2	R/W	0x2208	Write: Set register for port 2 Read: output bits for port 2	0	word (32 bit)	Table 195
SET3	R/W	0x220C	Write: Set register for port 3 Read: output bits for port 3	0	word (32 bit)	Table 195
SET4	R/W	0x2210	Write: Set register for port 4 Read: output bits for port 4	0	word (32 bit)	Table 195
SET5	R/W	0x2214	Write: Set register for port 5 Read: output bits for port 5	0	word (32 bit)	Table 195
SET6	R/W	0x2218	Write: Set register for port 6 Read: output bits for port 6	0	word (32 bit)	Table 195
SET7	R/W	0x221C	Write: Set register for port 7 Read: output bits for port 7	0	word (32 bit)	Table 195
CLR0	WO	0x2280	Clear port 0	NA	word (32 bit)	Table 196
CLR1	WO	0x2284	Clear port 1	NA	word (32 bit)	Table 196
CLR2	WO	0x2288	Clear port 2	NA	word (32 bit)	Table 196
CLR3	WO	0x228C	Clear port 3	NA	word (32 bit)	Table 196
CLR4	WO	0x2290	Clear port 4	NA	word (32 bit)	Table 196
CLR5	WO	0x2294	Clear port 5	NA	word (32 bit)	Table 196
CLR6	WO	0x2298	Clear port 6	NA	word (32 bit)	Table 196
CLR7	WO	0x229C	Clear port 7	NA	word (32 bit)	Table 196
NOT0	WO	0x2300	Toggle port 0	NA	word (32 bit)	Table 197
NOT1	WO	0x2304	Toggle port 1	NA	word (32 bit)	Table 197
NOT2	WO	0x2308	Toggle port 2	NA	word (32 bit)	Table 197
NOT3	WO	0x230C	Toggle port 3	NA	word (32 bit)	Table 197
NOT4	WO	0x2310	Toggle port 4	NA	word (32 bit)	Table 197
NOT5	WO	0x2314	Toggle port 5	NA	word (32 bit)	Table 197
NOT6	WO	0x2318	Toggle port 6	NA	word (32 bit)	Table 197
NOT7	WO	0x231C	Toggle port 7	NA	word (32 bit)	Table 197

[1] “ext” in this table and subsequent tables indicates that the data read after reset depends on the state of the pin, which in turn may depend on an external source.

17.5.1 GPIO pin interrupts register description

17.5.1.1 Pin interrupt mode register

For each of the 8 pin interrupts selected in Table 135 and Table 136, one bit in the ISEL register determines whether the interrupt is edge or level sensitive.

Table 176. Pin interrupt mode register (ISEL, address 0x4008 7000) bit description

Bit	Symbol	Description	Reset value	Access
7:0	PMODE	Selects the interrupt mode for each pin interrupt. Bit n configures the pin interrupt selected in PINTSELn. 0 = Edge sensitive 1 = Level sensitive	0	R/W
31:8	-	Reserved.	-	-

17.5.1.2 Pin interrupt level (rising edge interrupt) enable register

For each of the 8 pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136), one bit in the IENR register enables the interrupt depending on the pin interrupt mode configured in the ISEL register:

- If the pin interrupt mode is edge sensitive (PMODE = 0), the rising edge interrupt is enabled.
- If the pin interrupt mode is level sensitive (PMODE = 1), the level interrupt is enabled. The PINTEN_F register configures the active level (HIGH or LOW) for this interrupt.

Table 177. Pin interrupt level (rising edge interrupt enable) register (IENR, address 0x4008 7004) bit description

Bit	Symbol	Description	Reset value	Access
7:0	ENRL	Enables the rising edge or level interrupt for each pin interrupt. Bit n configures the pin interrupt selected in PINTSELn. 0 = Disable rising edge or level interrupt. 1 = Enable rising edge or level interrupt.	0	R/W
31:8	-	Reserved.	-	-

17.5.1.3 Pin interrupt level (rising edge interrupt) set register

For each of the 8 pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136), one bit in the SIENR register sets the corresponding bit in the IENR register depending on the pin interrupt mode configured in the PINTMODE register:

- If the pin interrupt mode is edge sensitive (PMODE = 0), the rising edge interrupt is set.
- If the pin interrupt mode is level sensitive (PMODE = 1), the level interrupt is set.

Table 178. Pin interrupt level (rising edge interrupt) set register (SIENR, address 0x4008 7008) bit description

Bit	Symbol	Description	Reset value	Access
7:0	SETENRL	Ones written to this address set bits in the PINTEN_R, thus enabling interrupts. Bit n sets bit n in the PINTEN_R register. 0 = No operation. 1 = Enable rising edge or level interrupt.	NA	WO
31:8	-	Reserved.	-	-

17.5.1.4 Pin interrupt level (rising edge interrupt) clear register

For each of the 8 pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136), one bit in the CIENR register clears the corresponding bit in the IENR register depending on the pin interrupt mode configured in the ISEL register:

- If the pin interrupt mode is edge sensitive (PMODE = 0), the rising edge interrupt is cleared.
- If the pin interrupt mode is level sensitive (PMODE = 1), the level interrupt is cleared.

Table 179. Pin interrupt level (rising edge interrupt) clear register (PCIENR, address 0x4008 700C) bit description

Bit	Symbol	Description	Reset value	Access
7:0	CENRL	Ones written to this address clear bits in the IENR, thus disabling the interrupts. Bit n clears bit n in the IENR register. 0 = No operation. 1 = Disable rising edge or level interrupt.	NA	WO
31:8	-	Reserved.	-	-

17.5.1.5 Pin interrupt active level (falling edge interrupt enable) register

For each of the 8 pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136), one bit in the PINTSEN_F register enables the falling edge interrupt or the configures the level sensitivity depending on the pin interrupt mode configured in the ISEL register:

- If the pin interrupt mode is edge sensitive (PMODE = 0), the falling edge interrupt is enabled.
- If the pin interrupt mode is level sensitive (PMODE = 1), the active level of the level interrupt (HIGH or LOW) is configured.

Table 180. Pin interrupt active level (falling edge interrupt enable) register (IENF, address 0x4008 7010) bit description

Bit	Symbol	Description	Reset value	Access
7:0	ENAF	Enables the falling edge or configures the active level interrupt for each pin interrupt. Bit n configures the pin interrupt selected in PINTSELn. 0 = Disable falling edge interrupt or set active interrupt level LOW. 1 = Enable falling edge interrupt enabled or set active interrupt level HIGH.	0	R/W
31:8	-	Reserved.	-	-

17.5.1.6 Pin interrupt active level (falling edge interrupt) set register

For each of the 8 pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136), one bit in the SIENF register sets the corresponding bit in the IENF register depending on the pin interrupt mode configured in the ISEL register:

- If the pin interrupt mode is edge sensitive (PMODE = 0), the falling edge interrupt is set.
- If the pin interrupt mode is level sensitive (PMODE = 1), the HIGH-active interrupt is selected.

Table 181. Pin interrupt active level (falling edge interrupt) set register (SIENF, address 0x4008 7014) bit description

Bit	Symbol	Description	Reset value	Access
7:0	SETENAF	Ones written to this address set bits in the IENF, thus enabling interrupts. Bit n sets bit n in the IENF register. 0 = No operation. 1 = Select HIGH-active interrupt or enable falling edge interrupt.	NA	WO
31:8	-	Reserved.	-	-

17.5.1.7 Pin interrupt active level (falling edge interrupt) clear register

For each of the 8 pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136), one bit in the CIENF register sets the corresponding bit in the IENF register depending on the pin interrupt mode configured in the ISEL register:

- If the pin interrupt mode is edge sensitive (PMODE = 0), the falling edge interrupt is cleared.
- If the pin interrupt mode is level sensitive (PMODE = 1), the LOW-active interrupt is selected.

Table 182. Pin interrupt active level (falling edge interrupt) clear register (CIENF, address 0x4008 7018) bit description

Bit	Symbol	Description	Reset value	Access
7:0	CENAF	Ones written to this address clears bits in the IENF, thus disabling interrupts. Bit n clears bit n in the IENF register. 0 = No operation. 1 = LOW-active interrupt selected or falling edge interrupt disabled.	NA	WO
31:8	-	Reserved.	-	-

17.5.1.8 Pin interrupt rising edge register

This register contains ones for pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136) on which a rising edge has been detected. Writing ones to this register clears rising edge detection. Ones in this register assert an interrupt request for pins that are enabled for rising-edge interrupts. All edges are detected for all pins selected by the PINTSEL registers, regardless of whether they are interrupt-enabled.

Table 183. Pin interrupt rising edge register (RISE, address 0x4008 701C) bit description

Bit	Symbol	Description	Reset value	Access
7:0	RDET	Rising edge detect. Bit n detects the rising edge of the pin selected in PINTSELn. Read 0: No rising edge has been detected on this pin since Reset or the last time a one was written to this bit. Write 0: no operation. Read 1: a rising edge has been detected since Reset or the last time a one was written to this bit. Write 1: clear rising edge detection for this pin.	0	R/W
31:8	-	Reserved.	-	-

17.5.1.9 Pin interrupt falling edge register

This register contains ones for pin interrupts selected in the PINTSEL registers (see Table 135 and Table 136) on which a falling edge has been detected. Writing ones to this register clears falling edge detection. Ones in this register assert an interrupt request for pins that are enabled for falling-edge interrupts. All edges are detected for all pins selected by the PINTSEL registers, regardless of whether they are interrupt-enabled.

Table 184. Pin interrupt falling edge register (FALL, address 0x4008 7020) bit description

Bit	Symbol	Description	Reset value	Access
7:0	FDET	Falling edge detect. Bit n detects the falling edge of the pin selected in PINTSELn. Read 0: No falling edge has been detected on this pin since Reset or the last time a one was written to this bit. Write 0: no operation. Read 1: a falling edge has been detected since Reset or the last time a one was written to this bit. Write 1: clear falling edge detection for this pin.	0	R/W
31:8	-	Reserved.	-	-

17.5.1.10 Pin interrupt status register

Reading this register returns ones for pin interrupts that are currently requesting an interrupt. For pins identified as edge-sensitive in the Interrupt Select register, writing ones to this register clears both rising- and falling-edge detection for the pin. For level-sensitive pins, writing ones inverts the corresponding bit in the Active level register, thus switching the active level on the pin.

Table 185. Pin interrupt status register (IST address 0x4008 7024) bit description

Bit	Symbol	Description	Reset value	Access
7:0	PSTAT	Pin interrupt status. Bit n returns the status, clears the edge interrupt, or inverts the active level of the pin selected in PINTSELn. Read 0: interrupt is not being requested for this interrupt pin. Write 0: no operation. Read 1: interrupt is being requested for this interrupt pin. Write 1 (edge-sensitive): clear rising- and falling-edge detection for this pin. Write 1 (level-sensitive): switch the active level for this pin (in the PINTENT_F register).	0	R/W
31:8	-	Reserved.	-	-

17.5.2 GPIO GROUP0/GROUP1 interrupt register description

17.5.2.1 Grouped interrupt control register

Table 186. GPIO grouped interrupt control register (CTRL, addresses 0x4008 8000 (GROUP0 INT) and 0x4008 9000 (GROUP1 INT)) bit description

Bit	Symbol	Value	Description	Reset value
0	INT		Group interrupt status. This bit is cleared by writing a one to it. Writing zero has no effect.	0
		0	No interrupt request is pending.	
		1	Interrupt request is active.	
1	COMB		Combine enabled inputs for group interrupt	0
		0	OR functionality: A grouped interrupt is generated when any one of the enabled inputs is active (based on its programmed polarity).	
		1	AND functionality: An interrupt is generated when all enabled bits are active (based on their programmed polarity).	
2	TRIG		Group interrupt trigger	0
		0	Edge-triggered	
		1	Level-triggered	
31:3	-	-	Reserved	0

17.5.2.2 GPIO grouped interrupt port polarity registers

The grouped interrupt port polarity registers determine how the polarity of each enabled pin contributes to the grouped interrupt. Each port n (n = 0 to 7) is associated with its own port polarity register, and the values of all registers together determine the grouped interrupt.

Table 187. GPIO grouped interrupt port polarity registers (PORT_POL, addresses 0x4008 8020 (PORT_POL0) to 0x4008 803C (PORT_POL7) (GROUP0 INT) and 0x4008 9020 (PORT_POL0) to 0x4008 903C (PORT_POL7) (GROUP1 INT)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	POL	Configure pin polarity of port n pins for group interrupt. Bit m corresponds to pin GPIOn[m] of port n. 0 = the pin is active LOW. If the level on this pin is LOW, the pin contributes to the group interrupt. 1 = the pin is active HIGH. If the level on this pin is HIGH, the pin contributes to the group interrupt.	1	-

17.5.2.3 GPIO grouped interrupt port enable registers

The grouped interrupt port enable registers enable the pins which contribute to the grouped interrupt. Each port n (n = 0 to 7) is associated with its own port enable register, and the values of all registers together determine which pins contribute to the grouped interrupt.

Table 188. GPIO grouped interrupt port n enable registers (PORT_ENA, addresses 0x4008 8040 (PORT_ENA0) to 0x4008 805C (PORT_ENA7) (GROUP0 INT) and 0x4008 9040 (PORT_ENA0) to 0x4008 905C (PORT_ENA7) (GROUP1 INT)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	ENA	Enable port n pin for group interrupt. Bit m corresponds to pin GPIOPn[m] of port n. 0 = the port n pin is disabled and does not contribute to the grouped interrupt. 1 = the port n pin is enabled and contributes to the grouped interrupt.	0	-

17.5.3 GPIO port register description

17.5.3.1 GPIO port byte pin registers

Each GPIO pin GPIOn[m] has a byte register in this address range. The byte pin registers of GPIO port 0 correspond to registers B0 to B31, the byte pin registers of GPIO port 1 correspond to registers B32 to B63, etc.. Byte addresses are reserved for unused GPIO port pins (see Table 170).

Software typically reads and writes bytes to access individual pins but also can read or write halfwords to sense or set the state of two pins, and read or write words to sense or set the state of four pins.

Table 189. GPIO port byte pin registers (B, addresses 0x400F 4000 (B0) to 0x400F 00FC (B255)) bit description

Bit	Symbol	Description	Reset value	Access
0	PBYTE	Read: state of the pin GPIOn[m], regardless of direction, masking, or alternate function. Pins configured as analog I/O always read as 0. Write: loads the pin's output bit.	ext	R/W
7:1		Reserved (0 on read, ignored on write)	0	-

17.5.3.2 GPIO port word pin registers

Each GPIO pin GPIOn[m] has a word register in this address range. The word pin registers of GPIO port 0 correspond to registers W0 to W31, the word pin registers of GPIO port 1 correspond to registers W32 to W63, etc.. Word addresses are reserved for unused GPIO port pins (see Table 170).

Any byte, halfword, or word read in this range will be all zeros if the pin is low or all ones if the pin is high, regardless of direction, masking, or alternate function, except that pins configured as analog I/O always read as zeros. Any write will clear the pin's output bit if the value written is all zeros, else it will set the pin's output bit.

Table 190. GPIO port word pin registers (W, addresses 0x400F 5000 (W0) to 0x400F 13FC (W255)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	PWORD	Read 0: pin GPIOn[m] is LOW. Write 0: clear output bit. Read 0xFFFF FFFF: pin is HIGH. Write any value 0x0000 0001 to 0xFFFF FFFF: set output bit. **Remark:** Only 0 or 0xFFFF FFFF can be read. Writing any value other than 0 will set the output bit.	ext	R/W

17.5.3.3 GPIO port direction registers

Each GPIO port n (n = 0 to 7) has one direction register for configuring the port pins as inputs or outputs.

Table 191. GPIO port direction register (DIR, addresses 0x400F 6000 (DIR0) to 0x400F 601C (DIR7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	DIR	Selects pin direction for pin GPIOn[m] (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]). 0 = input. 1 = output.	0	R/W

17.5.3.4 GPIO port mask registers

Each GPIO port has one mask register. The mask registers affect writing and reading the MPORT registers. Zeroes in these registers enable reading and writing; ones disable writing and result in zeros in corresponding positions when reading.

Table 192. GPIO port mask register (MASK, addresses 0x400F 6080 (MASK0) to 0x400F 609C (MASK7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	MASK	Controls which bits corresponding to GPIOn[m] are active in the MPORT register (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]). 0 = Read MPORT: pin state; write MPORT: load output bit. 1 = Read MPORT: 0; write MPORT: output bit not affected.	0	R/W

17.5.3.5 GPIO port pin registers

Each GPIO port has one port pin register. Reading these registers returns the current state of the pins read, regardless of direction, masking, or alternate functions, except that pins configured as analog I/O always read as 0s. Writing these registers loads the output bits of the pins written to, regardless of the Mask register.

Table 193. GPIO port pin register (PIN, addresses 0x400F 6100 (PIN0) to 0x400F 611C (PIN7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	PORT	Reads pin states or loads output bits (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]). 0 = Read: pin is LOW; write: clear output bit. 1 = Read: pin is HIGH; write: set output bit.	ext	R/W

17.5.3.6 GPIO masked port pin registers

Each GPIO port has one masked port pin register. These registers are similar to the PORT registers, except that the value read is masked by ANDing with the inverted contents of the corresponding MASK register, and writing to one of these registers only affects output register bits that are enabled by zeros in the corresponding MASK register.

Table 194. GPIO masked port pin register (MPIN, addresses 0x400F 6180 (MPIN0) to 0x400F 619C (MPIN7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	MPORT	Masked port register (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]). 0 = Read: pin is LOW and/or the corresponding bit in the MASK register is 1; write: clear output bit if the corresponding bit in the MASK register is 0. 1 = Read: pin is HIGH and the corresponding bit in the MASK register is 0; write: set output bit if the corresponding bit in the MASK register is 0.	ext	R/W

17.5.3.7 GPIO port set registers

Each GPIO port has one port set register. Output bits can be set by writing ones to these registers, regardless of MASK registers. Reading from these register returns the port's output bits, regardless of pin directions.

Table 195. GPIO port set register (SET, addresses 0x400F 6200 (SET0) to 0x400F 621C (SET7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	SET	Read or set output bits (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]). 0 = Read: output bit: write: no operation. 1 = Read: output bit; write: set output bit.	0	R/W

17.5.3.8 GPIO port clear registers

Each GPIO port has one output clear register. Output bits can be cleared by writing ones to these write-only registers, regardless of MASK registers.

Table 196. GPIO port clear register (CLR, addresses 0x400F 6280 (CLR0) to 0x400F 629C (CLR7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	CLR	Clear output bits (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]): 0 = No operation. 1 = Clear output bit.	NA	WO

17.5.3.9 GPIO port toggle registers

Each GPIO port has one output toggle register. Output bits can be toggled/inverted/complemented by writing ones to these write-only registers, regardless of MASK registers.

Table 197. GPIO port toggle register (NOT, addresses 0x400F 6300 (NOT0) to 0x400F 632C (NOT7)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	NOTP0	Toggle output bits (bit 0 = GPIOn[0], bit 1 = GPIOn[1], ..., bit 31 = GPIOn[31]): 0 = no operation. 1 = Toggle output bit.	NA	WO

17.6 Functional description

17.6.1 Reading pin state

Software can read the state of all GPIO pins except those selected for analog input or output in the “I/O Configuration” logic. A pin does not have to be selected for GPIO in “I/O Configuration” in order to read its state. There are four ways to read pin state:

- The state of a single pin can be read with 7 high-order zeros from a Byte Pin register.
- The state of a single pin can be read in all bits of a byte, halfword, or word from a Word Pin register.
- The state of multiple pins in a port can be read as a byte, halfword, or word from a PORT register.
- The state of a selected subset of the pins in a port can be read from a Masked Port (MPORT) register. Pins having a 1 in the port’s Mask register will read as 0 from its MPORT register.

17.6.2 GPIO output

Each GPIO pin has an output bit in the GPIO block. These output bits are the targets of write operations “to the pins”. Two conditions must be met in order for a pin’s output bit to be driven onto the pin:

1. The pin must be selected for GPIO operation in the “I/O Configuration” block, and

2. the pin must be selected for output by a 1 in its port's DIR register.

If either or both of these conditions is (are) not met, "writing to the pin" has no effect.

There are seven ways to change GPIO output bits:

- Writing to a Byte Pin register loads the output bit from the least significant bit.
- Writing to a Word Pin register loads the output bit with the OR of all of the bits written. (This feature follows the definition of "truth" of a multi-bit value in programming languages.)
- Writing to a port's PORT register loads the output bits of all the pins written to.
- Writing to a port's MPORT register loads the output bits of pins identified by zeros in corresponding positions of the port's MASK register.
- Writing ones to a port's SET register sets output bits.
- Writing ones to a port's CLR register clears output bits.
- Writing ones to a port's NOT register toggles/complements/inverts output bits.

The state of a port's output bits can be read from its SET register. Reading any of the registers described in 17.6.1 returns the state of pins, regardless of their direction or alternate functions.

17.6.3 Masked I/O

A port's MASK register defines which of its pins should be accessible in its MPORT register. Zeroes in MASK enable the corresponding pins to be read from and written to MPORT. Ones in MASK force a pin to read as 0 and its output bit to be unaffected by writes to MPORT. When a port's MASK register contains all zeros, its PORT and MPORT registers operate identically for reading and writing.

Users of previous NXP devices with similar GPIO blocks should be aware of an incompatibility: on the LPC43xx, writing to the SET, CLR, and NOT registers is not affected by the MASK register. On previous devices these registers were masked.

Applications in which interrupts can result in Masked GPIO operation, or in task switching among tasks that do Masked GPIO operation, must treat code that uses the Mask register as a protected/restricted region. This can be done by interrupt disabling or by using a semaphore.

The simpler way to protect a block of code that uses a MASK register is to disable interrupts before setting the MASK register, and re-enable them after the last operation that uses the MPORT or MASK register.

More efficiently, software can dedicate a semaphore to the MASK registers, and set/capture the semaphore controlling exclusive use of the MASK registers before setting the MASK registers, and release the semaphore after the last operation that uses the MPORT or MASK registers.

17.6.4 GPIO Interrupts

Two separate GPIO interrupt facilities are provided. With pin interrupts, up to eight GPIO pins can each have separately-vectored, edge- or level-sensitive interrupts.

With group interrupts, any subset of the pins in each port can be selected to contribute to a common interrupt. Any of the pin and port interrupts can be enabled to wake the part from Deep-sleep mode or Power-down mode.

17.6.4.1 Pin interrupts

In this interrupt facility, up to 8 pins are identified as interrupt sources by the Pin Interrupt Select registers (PINTSEL0-7). All of the other Pin Interrupt registers contain 8 bits, corresponding to the pins called out by the PINTSEL0-7 registers. The PINTMODE register defines whether each interrupt pin is edge- or level-sensitive. The PINTRISE and PINTFALL registers detect edges on each interrupt pin, and can be written to clear (and set) edge detection. The PINTST register indicates whether each interrupt pin is currently requesting an interrupt, and PINTST can be written to clear interrupts.

The other pin interrupt registers play different roles for edge-sensitive and level-sensitive pins, as described in Table 198.

Table 198. Pin interrupt registers for edge- and level-sensitive pins

Name	Edge-sensitive function	Level-sensitive function
PINTEN_R	Enables rising-edge interrupts.	Enables interrupts.
PINTSEN_R	Write to enable rising-edge interrupts.	Write to enable interrupts.
PINTCEN_R	Write to disable rising-edge interrupts.	Write to disable interrupts.
PINTEN_F	Enables falling-edge interrupts.	Selects active level.
PINTSEN_F	Write to enable falling-edge interrupts.	Write to select high-active.
PINTCEN_F	Write to disable falling-edge interrupts.	Write to select low-active.

17.6.4.2 Group interrupts

In this interrupt facility, an interrupt can be requested for each port, based on any selected subset of pins within each port. The pins that contribute to each port interrupt are selected by 1s in the port's Enable register, and an interrupt polarity can be selected for each pin in the port's Polarity register. The level on each pin is exclusive-ORed with its polarity bit and the result is ANDed with its enable bit, and these results are then inclusive-ORed among all the pins in the port, to create the port's raw interrupt request.

The raw interrupt request from each of the two group interrupts is sent to the NVIC, which can be programmed to treat it as level- or edge-sensitive (see Table 21).

17.6.5 Recommended practices

The following lists some recommended uses for using the GPIO port registers:

- For initial setup after Reset or re-initialization, write the PORT registers.
- To change the state of one pin, write a Byte Pin or Word Pin register.
- To change the state of multiple pins at a time, write the SET and/or CLR registers.
- To change the state of multiple pins in a tightly controlled environment like a software state machine, consider using the NOT register. This can require less write operations than SET and CLR.
- To read the state of one pin, read a Byte Pin or Word Pin register.
- To make a decision based on multiple pins, read and mask a PORT register.

18.1 How to read this chapter

The SGPIO is available on all LPC43xx parts. The 12-bit ADC is not available on parts LPC4350/30/20/10.

18.2 Basic configuration

The SGPIO is configured as follows:

- See Table 199 for clocking and power control.
- The SGPIO is reset by the SGPIO_RST (reset # 57).
- The SGPIO interrupt is connected to interrupt slot # 31 in the ARM Cortex-M4 NVIC.
- SGPIO outputs SGPIO3 and SGPIO12 are connected through the GIMA to the event recorder, the timers, and the SCT (see Table 138). SGPIO3 and SGPIO12 also support a divided-by-128 signal to the GIMA (SGPIO3_DIV and SGPIO12_DIV).
- SGPIO outputs 10 and 12 can trigger the 12-bit ADC.
- SGPIO outputs 14 and 15 can trigger a GPDMA request.

Table 199. SGPIO clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
SGPIO peripheral clock (SGPIO_CLOCK)	BASE_PERIPH_CLK	CLK_PERIPH_SGPIO	up to 204 MHz	This clock is asynchronous to the main clock and can be freely chosen to create a desired SGPIO data rate.

18.3 Features

- Each SGPIO input/output slice can be used to perform a serial to parallel or parallel to serial data conversion.
- 16 SGPIO input/output slices each with a 32-bit FIFO can shift the input value from a pin or an output value to a pin with every cycle of a shift clock.
- Each slice is double-buffered.
- Interrupt is generated on a full FIFO, shift clock, or pattern match.
- Slices can be concatenated to increase buffer size.
- Each slice has a 32-bit pattern match filter.

18.4 General description

Serial GPIO (SGPIO) offer standard GPIO functionality enhanced with features to accelerate serial stream processing. A data stream on a single SGPIO input or output or on a dual, quad, and byte lane data input/output is processed by using so called slices. Up to 16 slices are supported, and all 16 slices have the same basic feature set with some slices offering additional features.

- Slices:
  - Each slice supports a 32-bit FIFO, shifted in from a pin or out to a pin at every shift clock.
  - Some slices control the pin output enable.
  - Slices are double buffered; slices are swapped with the shadow register when the POS counter reaches 0x0.
  - Slices can be concatenated to increase the buffer size. Software should ensure that if slice n is concatenated with slice m, POS[n] and POS[m] are equal and run in phase.
  - Interrupt is raised when the POS counter reaches 0x0.
  - Support for parallel to multi-stream serial conversion and vice versa. Conversion supports interleaving; Up to 8 bits can be shifted at a time (n= 1, 2, 4 or 8 for serial, dual-serial, quad-serial and byte parallel IO).
- Clock:
  - 12-bit counter running at SGPIO_CLOCK creates a shift clock to capture input or create output values. Note that the input frequency should be less than half of the SGPIO_CLOCK frequency.
  - External input can be used as shift clock. Active edge can be rising or falling (not both).
  - External input can be used as shift clock qualifier. Can be active LOW or HIGH.
  - Internal signals can be used as shift clock qualifier.
  - POS counter running at shift clock to enable double buffering.
- Input:
  - Inputs are captured using the shift clock.
  - Inputs can raise interrupt on input level (LOW or HIGH) or transitions (rising, falling rising or falling). Interrupts can be masked.
  - Interrupts can be raised on a matching pattern. A pattern can be up to 32 bit long and can be masked (match value = don’t care). Four slices (A, I, H and P) support mask functionality.
  - Four inputs can be used as shift clock for other slices to capture other inputs.
- Outputs:
  - Create output: active level (LOW or HIGH) or tri-state. Note if a slice is used to create an output clock at rate f_{out} the slice clock should at least be 2x f_{out}. This limits the data rate per slice to half the SGPIO_CLOCK rate.
  - Output enable can be controlled by other slices. For multi-lane output, the LSB lane output enable can be controlled independently from the MSB lines.
  - Create output clock from shift clock counter.

- Output clock polarity can be inverted.

- Interface
  - The register memory map supports use of ARM Store Multiple and Load Multiple instructions. Slice functions that control the same features are mapped in consecutive registers.
  - The bit order is optimized for MSB first. Interfaces that require LSB first should use a software instruction (RBIT) to reverse the bit order.

18.4.1 Interrupts

- Interrupts are raised on the following events:
  - The shift clock interrupt is raised at the occurrence of a shift clock when COUNTx equals zero (see Table 222 to Table 227). This interrupt is generated at each shift bit.
  - The capture clock interrupt is raised when a slice swap occurs, that is at the occurrence of a capture clock when POSx equals zero (see Table 228 to Table 233).
  - The pattern match interrupt is raised when the input data is equal to the masked pattern (see Table 234 to Table 239).
  - The input bit match interrupt is raised when the input bit is equal to the conditions set in DATA_CAPTURE_MODE (Table 240 to Table 244).
- Each of the interrupts can be controlled through a set of registers in one of the following ways:
  - Disable or enable interrupts using registers CLR_EN/SET_EN.
  - Read interrupts via the register STATUS.
  - Read the interrupt mask via the register ENABLE.
  - Set interrupts via the register SET_STAT.
  - Clear interrupts via the register CTR_STAT.

18.5 Pin description

Table 200. SGPIO pin description

Pin function	Direction	Description
SGPIO[15:0]	I/O	Serial GPIO input/output

18.6 Register description

Table 201. Register overview: SGPIO (base address 0x4010 1000)

Name	Access	Address offset	Description	Reset value	Reference
OUT_MUXCFG0 to OUT_MUX_CFG15	R/W	0x0000 to 0x003C	Pin multiplexer configuration registers.	0	Table 202
SGPIO_MUX_CFG0 to SGPIO_MUX_CFG15	R/W	0x0040 to 0x007C	SGPIO multiplexer configuration registers.	0	Table 205
SLICE_MUX_CFG0 to SLICE_MUX_CFG15	R/W	0x0080 to 0x00BC	Slice multiplexer configuration registers.	0	Table 207
REG0 to REG15	R/W	0x00C0 to 0x00FC	Slice data registers. Each time COUNT0 reaches 0x0 the register shifts loading bit 31 with data captured from DIN(n). DOUT(n) is set to REG(0)	0	Table 208
REG_SS0 to REG_SS15	R/W	0x0100 to 0x013C	Slice data shadow registers. Each time POS reaches 0x0 the contents of REG_SS is exchanged with the content of REG	0	Table 209
PRESET0 to PRESET15	R/W	0x0140 to 0x17C	Reload value of COUNT0, loaded when COUNT0 reaches 0x0	0	Table 210
COUNT0 to COUNT15	R/W	0x0180 to 0x1BC	Down counter, counts down each clock cycle.	0	Table 211
POS0 to POS15	R/W	0x01C0 to 0x01FC	Each time COUNT0 reaches 0x0 POS counts down.	0	Table 212
MASK_A	R/W	0x0200	Mask for pattern match function of slice A	0	Table 213
MASK_H	R/W	0x0204	Mask for pattern match function of slice H	0	Table 214
MASK_I	R/W	0x0208	Mask for pattern match function of slice I	0	Table 215
MASK_P	R/W	0x020C	Mask for pattern match function of slice P	0	Table 216
GPIO_INREG	R	0x0210	GPIO input status register	0	Table 217
GPIO_OUTREG	R/W	0x0214	GPIO output control register	0	Table 218
GPIO_OENREG	R/W	0x0218	GPIO OE control register	0	Table 219
CTRL_ENABLE	R/W	0x021C	Enables the slice COUNT counter	0	Table 220
CTRL_DISABLE	R/W	0x0220	Disables the slice COUNT counter	0	Table 221
CLR_EN_0	W	0x0F00	Shift clock interrupt clear mask	0	Table 222
SET_EN_0	W	0x0F04	Shift clock interrupt set mask	0	Table 223
ENABLE_0	R	0x0F08	Shift clock interrupt enable	0	Table 224
STATUS_0	R	0x0F0C	Shift clock interrupt status	0	Table 225
CTR_STAT_0	W	0x0F10	Shift clock interrupt clear status	0	Table 226
SET_STAT_0	W	0x0F14	Shift clock interrupt set status	0	Table 227
CLR_EN_1	W	0x0F20	Capture clock interrupt clear mask	0	Table 228
SET_EN_1	W	0x0F24	Capture clock interrupt set mask	0	Table 229
ENABLE_1	R	0x0F28	Capture clock interrupt enable	0	Table 230
STATUS_1	R	0x0F2C	Capture clock interrupt status	0	Table 231
CTR_STAT_1	W	0x0F30	Capture clock interrupt clear status	0	Table 232
SET_STAT_1	W	0x0F34	Capture clock interrupt set status	0	Table 233
CLR_EN_2	W	0x0F40	Pattern match interrupt clear mask	0	Table 234

Table 201. Register overview: SGPIO (base address 0x4010 1000)

Name	Access	Address offset	Description	Reset value	Reference
SET_EN_2	W	0x0F44	Pattern match interrupt set mask	0	Table 235
ENABLE_2	R	0x0F48	Pattern match interrupt enable	0	Table 236
STATUS_2	R	0x0F4C	Pattern match interrupt status	0	Table 237
CTR_STAT_2	W	0x0F50	Pattern match interrupt clear status	0	Table 238
SET_STAT_2	W	0x0F54	Pattern match interrupt set status	0	Table 239
CLR_EN_3	W	0x0F60	Input interrupt clear mask	0	Table 240
SET_EN_3	W	0x0F64	Input bit match interrupt set mask	0	Table 241
ENABLE_3	R	0x0F68	Input bit match interrupt enable	0	Table 242
STATUS_3	R	0x0F6C	Input bit match interrupt status	0	Table 243
CTR_STAT_3	W	0x0F70	Input bit match interrupt clear status	0	Table 244
SET_STAT_3	W	0x0F74	Input bit match interrupt set status	0	Table 245

18.6.1 Pin multiplexer configuration registers (OUT_MUX_CFG0 to 15)

Each register controls one local output pin multiplexer: OUT_MUX_CFG0 to OUT_MUX_CTRL15 control pins SGPIO0 to SGPIO15.

Remark: Pin refers to a pin on the LPC43xx configured as SGPIOn pin through the pin configuration registers.

The P_OUT_CFG bits control the width of the data stream when data is generated by a slice or whether the output is used as GPIO or clock. Not all modes are supported by all slices. Table 203 shows how the SGPIOn pins are connected to slices. For example, the 8-bit wide mode is only supported by slices A/B, J/M, or L/N. When using wider than 1-bit modes, the number of bits shifted per clock should be set equal to the width for parallel mode (see bits PARALLEL_MODE in Table 207).

The P_OE_CFG bits control the output enable source.This can be done statically with register GPIO_OEREG or dynamically by another slice. Table 204 indicates which slices control the output enable of which pins. Note that for modes wider than 1 bit, the output enable is defined by 2 bits: one bit for the LSB and one bit for the MSBs. When using wider than 1-bit modes (P_OE_CFG = dout_oem2, dout_oem4, or dout_oem8), the number of bits shifted per clock should be set to 2 bits per clock.

Table 202. Pin multiplexer configuration registers (OUT_MUX_CFG0 to 15, addresses 0x4010 1000 to 0x4010 103C) bit description

Bit	Symbol	Value	Description	Reset value	Access
3:0	P_OUT_CFG		Output control of output SGPIOn. All other values are reserved.	0	R/W
		0x0	dout_doutm1 (1-bit mode)		
		0x1	dout_doutm2a (2-bit mode 2a)		
		0x2	dout_doutm2b (2-bit mode 2b)		
		0x3	dout_doutm2c (2-bit mode 2c)		
		0x4	gpio_out (level set by GPIO_OUTREG)		
		0x5	dout_doutm4a (4-bit mode 4a)		
		0x6	dout_doutm4b (4-bit mode 4b)		
		0x7	dout_doutm4c (4-bit mode 4c)		
		0x8	clk_out		
		0x9	dout_doutm8a (8-bit mode 8a)		
		0xA	dout_doutm8b (8-bit mode 8b)		
		0xB	dout_doutm8c (8-bit mode 8c)		
6:4	P_OE_CFG		Output enable source. All other values are reserved.	0	R/W
		0x0	gpio_oe (state set by GPIO_OEREG)		
		0x4	dout_oem1 (1-bit mode)		
		0x5	dout_oem2 (2-bit mode)		
		0x6	dout_oem4 (4-bit mode)		
		0x7	dout_oem8 (8-bit mode)		
31:7	-		Reserved.	-	-

Table 203. Output pin multiplexing

SGPIO pin	Output mode - register OUT_MUX_CFG, bits P_OUT_CFG (see Table 202)											
	1011	1010	1001	0111	0110	0101	0011	0010	0001	0000	1000	0100
	8-bit 8c	8-bit 8b	8-bit 8a	4-bit 4c	4-bit 4b	4-bit 4a	2-bit 2c	2-bit 2b	2-bit 2a	1-bit	clk	gpio
0	L0	J0	A0	J0	I0	A0	J0	I0	A0	A0	Bck	A
1	L1	J1	A1	J1	I1	A1	J1	I1	A1	I0	Dck	B
2	L2	J2	A2	J2	I2	A2	I0	J0	E0	E0	Eck	C
3	L3	J3	A3	J3	I3	A3	I1	J1	E1	J0	Hck	D
4	L4	J4	A4	L0	K0	C0	L0	K0	C0	C0	Cck	E
5	L5	J5	A5	L1	K1	C1	L1	K1	C1	K0	Fck	F
6	L6	J6	A6	L2	K2	C2	K0	L0	F0	F0	Ock	G
7	L7	J7	A7	L3	K3	C3	K1	L1	F1	L0	Pck	H
8	N0	M0	B0	N0	M0	B0	N0	M0	B0	B0	Ack	I
9	N1	M1	B1	N1	M1	B1	N1	M1	B1	M0	Mck	J
10	N2	M2	B2	N2	M2	B2	M0	N0	G0	G0	Gck	K

Table 203. Output pin multiplexing

SGPIO pin	Output mode - register OUT_MUX_CFG, bits P_OUT_CFG (see Table 202)											
	1011	1010	1001	0111	0110	0101	0011	0010	0001	0000	1000	0100
	8-bit 8c	8-bit 8b	8-bit 8a	4-bit 4c	4-bit 4b	4-bit 4a	2-bit 2c	2-bit 2b	2-bit 2a	1-bit	clk	gpio
11	N3	M3	B3	N3	M3	B3	M1	N1	G1	N0	Nck	L
12	N4	M4	B4	P0	O0	D0	P0	O0	D0	D0	Ick	M
13	N5	M5	B5	P1	O1	D1	P1	O1	D1	O0	Jck	N
14	N6	M6	B6	P2	O2	D2	O0	P0	H0	H0	Kck	O
15	N7	M7	B7	P3	O3	D3	O1	P1	H1	P0	Lck	P

Table 204. Output enable control

SGPIO pin	OE control - register OUT_MUX_CFG, bits P_OE_CFG				
	111	110	101	100	000
	8-bit	4-bit	2-bit	1-bit	gpio
0	H0	H0	H0	B0	A
1	H1	H1	H1	M0	B
2	H1	H1	D0	G0	C
3	H1	H1	D1	N0	D
4	H1	O0	G0	D0	E
5	H1	O1	G1	O0	F
6	H1	O1	O0	H0	G
7	H1	O1	O1	P0	H
8	P0	P0	P0	A0	I
9	P1	P1	P1	I0	J
10	P1	P1	B0	E0	K
11	P1	P1	B1	J0	L
12	P1	N0	N0	C0	M
13	P1	N1	N1	K0	N
14	P1	N1	M0	F0	O
15	P1	N1	M1	L0	P

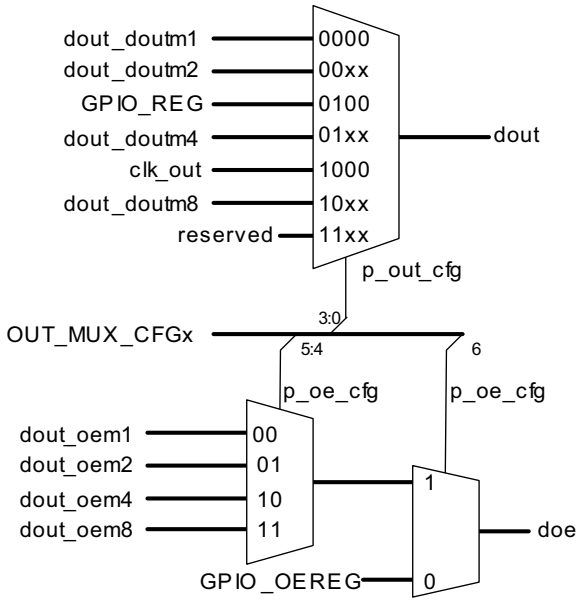


Fig 30. SGPIO local output pin multiplexer configuration

18.6.2 SGPIO multiplexer configuration registers (SGPIO_MUX_CFG0 to 15)

Each register controls one SGPIO multiplexer to select the following sources:

- External clock source (when COUNT is not used)
- External clock qualifier (static or controlled by a slice or external pin)
- Input concatenation structure

SGPIO_MUX_CFG0 to SGPIO_MUX_CFG15 control slices A (register 0) to P (register 15).

To avoid oscillation, slices that can be a clock source for other slices cannot support external slice clocks themselves (CLK_SOURCE_SLICE_MODE). These slices should not feed a clock higher than SGPIO_CLOCK/2 to the other slices.

Table 205. SGPIO multiplexer configuration registers (SGPIO_MUX_CFG0 to 15, addresses 0x4010 0040 to 0x4010 007C) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	EXT_CLK_ENABLE		Select clock signal.	0	R/W
		0x0	Internal clock signal (slice)		
		0x1	External clock signal (pin)		

Table 205. SGPIO multiplexer configuration registers (SGPIO_MUX_CFG0 to 15, addresses 0x4010 0040 to 0x4010 007C) bit description

…continued

Bit	Symbol	Value	Description	Reset value	Access
2:1	CLK_SOURCE_PIN_MODE		Select source clock pin.	0	R/W
		0x0	SGPIO8		
		0x1	SGPIO9		
		0x2	SGPIO10		
		0x3	SGPIO11		
4:3	CLK_SOURCE_SLICE_MODE		Select clock source slice. Note that slices D, H, O and P do not support this mode.	0	R/W
		0x0	Slice D		
		0x1	Slice H		
		0x2	Slice O		
		0x3	Slice P		
6:5	QUALIFIER_MODE		Select qualifier mode.	0	R/W
		0x0	Enable		
		0x1	Disable		
		0x2	Slice (see bits QUALIFIER_SLICE_MODE in this register)		
		0x3	External SGPIO pin (8, 9, 10, or 11)		
8:7	QUALIFIER_PIN_MODE		Select qualifier pin.	0	R/W
		0x0	SGPIO8		
		0x1	SGPIO9		
		0x2	SGPIO10		
		0x3	SGPIO11		
10:9	QUALIFIER_SLICE_MODE		Select qualifier slice.	0	R/W
		0x0	Slice A, but for slice A slice D is used.		
		0x1	Slice H, but for slice H slice O is used.		
		0x2	Slice I, but for slice I slice D is used.		
		0x3	Slice P, but for slice P slice O is used.		
11	CONCAT_ENABLE		Enable concatenation.	0	R/W
		0x0	External data pin		
		0x1	Concatenate data		
13:12	CONCAT_ORDER		Select concatenation order	0	R/W
		0x0	Self-loop		
		0x1	2 slices		
		0x2	4 slices		
		0x3	8 slices		
31:14	-		Reserved	-	-

Table 206. SGPIO multiplexer

slice	Slice Din					slice	Clock			
							Slice Din			
	CONCAT_ENABLE						CLK_SOURCE_ SLICE_MODE			
	0	1								
		CONCAT_ORDER								
		00	01	10	11		00	01	10	11
A	Pin 0	A	I	J	L	A	D	H	O	P
I	Pin 8	I	A	A	A	I	D	H	O	P
E	Pin 4	E	J	I	I	E	D	H	O	P
J	Pin 9	J	E	E	E	J	D	H	O	P
C	Pin 2	C	K	L	J	C	D	H	O	P
K	Pin 10	K	C	C	C	K	D	H	O	P
F	Pin 5	F	L	K	K	F	D	H	O	P
L	Pin 11	L	F	F	F	L	D	H	O	P
B	Pin 1	B	M	N	P	B	D	H	O	P
M	Pin 12	M	B	B	B	M	D	H	O	P
G	Pin 6	G	N	M	M	G	D	H	O	P
N	Pin 13	N	G	G	G	N	D	H	O	P
D	Pin 3	D	O	P	N	D	-	-	-	-
O	Pin 14	O	D	D	D	O	-	-	-	-
H	Pin 7	H	P	O	O	H	-	-	-	-
P	Pin15	P	H	H	H	P	-	-	-	-

18.6.3 Slice multiplexer configuration registers (SLICE_MUX_CFG0 to 15)

Each register controls a slice multiplexer shift clock and how many bits of the slice FIFO are shifted per clock.

Registers SLICE_MUX_CFG0 to SLICE_MUX_CFG15 control slices A (register 0) to P (register 15).

Bit MATCH_MODE selects whether the match filter is active or whether data is captured. Only slice A, H, I, and P support matching with a mask (register MASK_x). For other slices the pattern is not masked.

Bit CLKGEN_MODE selects as shift clock the clock generated by the slice counter or by an external pin or other slice.

Bit INV_OUT_CLK can invert the shift clock. This should only be used for external clocks coming from a pin.

Data_capture_mode is used to define which input data condition can generate an interrupt.

See Table 207 for connecting slice data to pins for the various setting of bits PARALLEL_MODE.

Table 207. Slice multiplexer configuration registers (SLICE_MUX_CFG0 to 15, addresses 0x4010 1080 to 0x4010 10BC) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	MATCH_MODE		Match mode	0	R/W
		0x0	Do not match data.		
		0x1	Match data.		
1	CLK_CAPTURE_MODE		Capture clock mode	0	R/W
		0x0	Use rising clock edge.		
		0x1	Use falling clock edge.		
2	CLKGEN_MODE		Clock generation mode	0	R/W
		0x0	Use clock internally generated by COUNTER.		
		0x1	Use external clock from a pin or other slice.		
3	INV_OUT_CLK		Invert output clock	0	R/W
		0x0	Normal clock.		
		0x1	Inverted clock.		
5:4	DATA_CAPTURE_MODE		Condition for input bit match interrupt	0	R/W
		0x0	Detect rising edge.		
		0x1	Detect falling edge.		
		0x2	Detect LOW level.		
		0x3	Detect HIGH level.		
7:6	PARALLEL_MODE		Parallel mode	0	R/W
		0x0	Shift 1 bit per clock.		
		0x1	Shift 2 bits per clock.		
		0x2	Shift 4 bits per clock.		
		0x3	Shift 1 byte per clock.		
8	INV_QUALIFIER		Inversion qualifier	0	R/W
		0x0	Use normal qualifier.		
		0x1	Use inverted qualifier.		
31:9	-	-	Reserved.	-	-

18.6.4 Slice data registers (REG0 to 15)

Each register contains the data for one slice: REG0 to REG15 contain data for slice A (register 0) to slice P (register 15).

At an active shift clock data is right shifted; captured data is shifted in at bit 31, and register data is shifted out from bit 0.

Table 208. Slice data registers (REG0 to 15, addresses 0x4010 10C0 to 0x4010 10FC) bit description

Bit	Symbol	Description	Reset value	Access
31:0	REG	At each active shift clock the register shifts right; loading REG(31) with data captured from DIN(n) and DOUT(n) is set to REG(0).	0	R/W

18.6.5 Slice data shadow registers (REG_SS0 to 15)

Each slice data shadow register contains the data for one slice: REG_SS0 to REG_SS15 contain data for slice A (register 0) to slice P (register 15).

This register controls when the shadow register REG_SS content is exchanged with the main register REG. The exchange occurs when the POS counter reaches 0x0.

Table 209. Slice data shadow registers (REG_SS0 to 15, addresses 0x4010 110 to 0x4010 113C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	REG_SS	Each time POS reaches 0x0 the contents of REG_SS is exchanged with the content of REG.	0	R/W

18.6.6 Reload registers (PRESET0 to 15)

Each register contains the reload value for one slice: PRESET0 to PRESET15 contain the reload value for slice A (register 0) to slice P (register 15).

This register controls the internally generated slice shift clock frequency: frequency(shift_clock) = frequency(SGPIO_CLK) / (PRESET+1).

Table 210. Reload registers (PRESET0 to 15, addresses 0x4010 1140 to 0x4010 117C) bit description

Bit	Symbol	Description	Reset value	Access
11:0	PRESET	Counter reload value; loaded when COUNT reaches 0x0.	0	R/W
31:12	-	Reserved.	-	-

18.6.7 Down counter registers (COUNT0 to 15)

This status register reflect the slice shift clock counter value. If the counter has to start with a defined phase then COUNT should be set to the desired value before the counter is enabled using CTRL_ENABLE.

Table 211. Down counter registers (COUNT0 to 15, addresses 0x4010 1180 to 0x4010 11BC) bit description

Bit	Symbol	Description	Reset value	Access
11:0	COUNT	Down counter, counts down each shift clock cycle. Next count after 0x0 is PRESET.	0	R/W
31:12	-	Reserved.	-	-

18.6.8 Position registers (POS0 to 15)

Each position register contains the position counter for one slice: POS0 to POS15 contain the counter for slice A (register 0) to slice P (register 15).

This register controls when the shadow register REG_SS content is exchanged with main register REG.

To exchange content every k x 32 bit, POS_PRESET should be 0x20 x k -1. This setting should be used when k slices are concatenated (CONCAT_ENABLE is set).

Remark: Before a slice is started (using CTRL_ENABLE), POS should be set to the POS_PRESET value.

Table 212. Position registers (POS0 to 15, addresses 0x4010 11C0 to 0x4010 11FC) bit description

Bit	Symbol	Description	Reset value	Access
7:0	POS	Each time COUNT reaches 0x0 POS counts down.	0	R/W
15:8	POS_RESET	Reload value for POS after POS reaches 0x0.	0	R/W
31:16	-	Reserved.	-	-

18.6.9 Slice A mask register (MASK_A)

Table 213. Slice A mask register (MASK_A, address 0x4010 1200) bit description

Bit	Symbol	Description	Reset value	Access
31:0	MASK_A	Mask for pattern match function of slice A 0 = No effect. 1 = Mask this bit.	0	R/W

18.6.10 Slice H mask register (MASK_H)

Table 214. Slice H mask register (MASK_H, address 0x4010 1204) bit description

Bit	Symbol	Description	Reset value	Access
31:0	MASK_H	Mask for pattern match function of slice H 0 = No effect. 1 = Mask this bit.	0	R/W

18.6.11 Slice I mask register (MASK_I)

Table 215. Slice I mask register (MASK_I, address 0x4010 1208) bit description

Bit	Symbol	Description	Reset value	Access
31:0	MASK_I	Mask for pattern match function of slice I 0 = No effect. 1 = Mask this bit.	0	R/W

18.6.12 Slice P mask register (MASK_P)

Table 216. Slice P mask register (MASK_P, address 0x4010 120C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	MASK_P	Mask for pattern match function of slice P 0 = No effect. 1 = Mask this bit.	0	R/W

18.6.13 GPIO input status register (GPIO_INREG)

Table 217. GPIO input status register (GPIO_INREG, address 0x4010 1210) bit description

Bit	Symbol	Description	Reset value	Access
15:0	GPIO_INi	Bit i reflects the input state of SGPIO pin i. 0 = LOW 1 = HIGH	0	R
31:16	-	Reserved.	-	-

18.6.14 GPIO output control register (GPIO_OUTREG)

Table 218. GPIO output control register (GPIO_OUTREG, address 0x4010 1214) bit description

Bit	Symbol	Description	Reset value	Access
15:0	GPIO_OUT	GPIO output register. Bit i sets the output of SGPIO pin i. 0 = LOW 1 = HIGH	0	R/W
31:16	-	Reserved.	-	-

18.6.15 GPIO output enable register (GPIO_OENREG)

Table 219. GPIO output enable register (GPIO_OENREG, address 0x4010 1218) bit description

Bit	Symbol	Description	Reset value	Access
15:0	GPIO_OE	Bit i selects the output enable state of SGPIO pin i. 0 = GPIO output i is tri-stated. 1 = GPIO output i is active.	0	R/W
31:16	-	Reserved.	-	-

18.6.16 Slice count enable register (CTRL_ENABLED)

Table 220. Slice count enable register (CTRL_ENABLED, address 0x4010 121C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTRL_ENABLED	Slice count enable. Bit n controls slice n (0 = slice A, ..., 15 = slice P). 0 = <tbd>. 1 = Enables COUNTn.	0	R/W
31:16	-	Reserved.	-	-

18.6.17 Slice count disable register (CTRL_DISABLED)

When this register is set, it synchronously disables the slice counter COUNTi when the POSi counter reaches a zero count.

When starting COUNTi (by setting CTRL_ENABLEi), this register should always be cleared. If only on POSi countdown is needed (when only one slice should be processed), then this register should be set after COUNTi is started with register CTRL_ENABLEi.

Table 221. Slice count disable register (CTRL_DISABLED, address 0x4010 1220) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTRL_DISABLEDn	Slice count disable. Bit n controls slice n, (0 = slice A, ..., 15 = slice P). 0 = <tbd> 1 = Disables COUNTn.	0	R/W
31:16	-	Reserved.	-	-

18.6.18 Shift clock interrupt clear mask register (CLR_EN_0)

This register clears the shift clock interrupt mask of a slice.

Table 222. Shift clock interrupt clear mask register (CLR_EN_0, address 0x4010 1F00) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CLR_SCI	1 = Shift clock interrupt clear mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.19 Shift clock interrupt set mask register (SET_EN_0)

This register masks the shift clock interrupt of a slice.

Table 223. Shift clock interrupt set mask register (SET_EN_0, address 0x4010 1F04) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SET_SCI	1 = Shift clock interrupt set mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.20 Shift clock interrupt enable register (ENABLE_0)

This register indicates whether the shift clock interrupt of a slice is enabled.

Table 224. Shift clock interrupt enable register (ENABLE_0, address 0x4010 1F08) bit description

Bit	Symbol	Description	Reset value	Access
15:0	ENABLE_SCI	1 = Shift clock interrupt enable of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.21 Shift clock interrupt status register (STATUS_0)

This register indicates the shift clock interrupt status of a slice.

Table 225. Shift clock interrupt status register (STATUS_0, address 0x4010 1F0C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	STATUS_SCI	Shift clock interrupt status of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.22 Shift clock interrupt clear status register (CTR_STATUS_0)

This register clears the shift clock interrupt of a slice.

Table 226. Shift clock interrupt clear status register (CTR_STATUS_0, address 0x4010 1F10) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_SCI	Shift clock interrupt clear status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.23 Shift clock interrupt set status register (SET_STATUS_0)

This register sets the shift clock interrupt of a slice.

Table 227. Shift clock interrupt set status register (SET_STATUS_0, address 0x4010 1F14) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_SCI	Shift clock interrupt set status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.24 Capture clock interrupt clear mask register (CLR_EN_1)

Table 228. Capture clock interrupt clear mask register (CLR_EN_1, address 0x4010 1F20) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CLR_EN_CCI	1 = Capture clock interrupt clear mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.25 Capture clock interrupt set mask register (SET_EN_1)

Table 229. Capture clock interrupt set mask register (SET_EN_1, address 0x4010 1F24) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SET_EN_CCI	1 = Capture clock interrupt set mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.26 Capture clock interrupt enable (ENABLE_1)

Table 230. Capture clock interrupt enable register (ENABLE_1, address 0x4010 1F28) bit description

Bit	Symbol	Description	Reset value	Access
15:0	ENABLE_CCI	Capture clock interrupt enable of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.27 Capture clock interrupt status register (STATUS_1)

Table 231. Capture clock interrupt status register (STATUS_1, address 0x4010 1F2C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	STATUS_CCI	Capture clock interrupt status of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.28 Capture clock interrupt clear status register (CTR_STATUS_1)

Table 232. Capture clock interrupt clear status register (CTR_STATUS_1, address 0x4010 1F30) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_CCI	Capture clock interrupt clear status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.29 Capture clock interrupt set status register (SET_STATUS_1)

Table 233. Capture clock interrupt set status register (SET_STATUS_1, address 0x4010 1F34) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_CCI	Capture clock interrupt set status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.30 Pattern match interrupt clear mask register (CLR_EN_2)

Table 234. Pattern match interrupt clear mask register (CLR_EN2, address 0x4010 1F40) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CLR_EN2_PMI	1 = Match interrupt clear mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.31 Pattern match interrupt set mask register (SET_EN_2)

Table 235. Pattern match interrupt set mask register (SET_EN_2, address 0x4010 1F44) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SET_EN_PMI	1 = Match interrupt set mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.32 Pattern match interrupt enable (ENABLE_2)

Table 236. Pattern match interrupt enable register (ENABLE_2, address 0x4010 1F48) bit description

Bit	Symbol	Description	Reset value	Access
15:0	ENABLE_PMI	Match interrupt enable of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.33 Pattern match interrupt status register (STATUS_2)

Table 237. Pattern match interrupt status register (STATUS_2, address 0x4010 1F4C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	STATUS_PMI	Match interrupt status of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.34 Pattern match interrupt clear status register (CTR_STATUS_2)

Table 238. Pattern match interrupt clear status register (CTR_STATUS_2, address 0x4010 1F50) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_PMI	Match interrupt clear status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.35 Pattern match interrupt set status register (SET_STATUS_2)

Table 239. Pattern match interrupt set status register (SET_STATUS_2, address 0x4010 1F54) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_PMI	Match interrupt set status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.36 Input interrupt clear mask register (CLR_EN_3)

Table 240. Input interrupt clear mask register (CLR_EN_3, address 0x4010 1F60 bit description

Bit	Symbol	Description	Reset value	Access
15:0	CLR_EN_INPI	1 = Input interrupt clear mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.37 Input bit match interrupt set mask register (SET_EN_3)

Table 241. Input interrupt set mask register (SET_EN_3, address 0x4010 1F64) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SET_EN_INPI	1 = Input interrupt set mask of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.38 Input bit match interrupt enable (ENABLE_3)

Table 242. Input interrupt enable register (ENABLE_3, address 0x4010 1F68) bit description

Bit	Symbol	Description	Reset value	Access
15:0	ENABLE3_INPI	Input interrupt enable of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.39 Input bit match interrupt status register (STATUS_3)

Table 243. Input interrupt status register (STATUS_3, address 0x4010 1F6C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	STATUS_INPI	Input interrupt status of slice n.	0	R
31:16	-	Reserved.	-	-

18.6.40 Input bit match interrupt clear status register (CTR_STATUS_3)

Table 244. Input interrupt clear status register (CTR_STATUS_3, address 0x4010 1F70) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_INPI	Input interrupt clear status of slice n.	0	W
31:16	-	Reserved.	-	-

18.6.41 Input bit match interrupt set status register (SET_STATUS_3)

Table 245. Shift clock interrupt set status register (SET_STATUS_3, address 0x4010 1F74) bit description

Bit	Symbol	Description	Reset value	Access
15:0	CTR_STATUS_INPI	Shift interrupt set status of slice n.	0	W
31:16	-	Reserved.	-	-

18.7 Functional description

Serial GPIO (SGPIO) offer standard GPIO functionality enhanced with features to accelerate serial stream processing. The enhanced features are made using so called slices. All 16 slices have the same basic feature set. Some slices offer additional features for pattern matching and processing 2-, 4- or 8-bit wide streams.

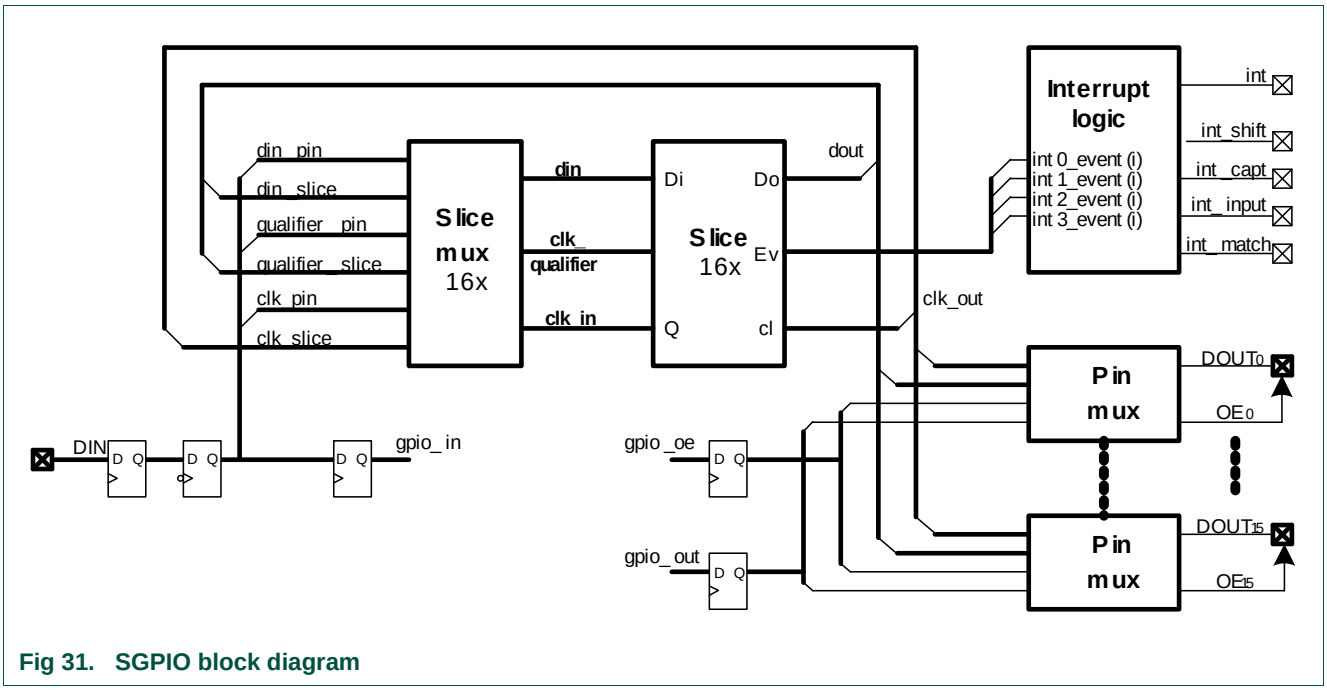


Fig 31. SGPIO block diagram

A slice performs parallel to serial data conversion (and vice versa). One slice contains 32 1-bit registers (REG) connected in a chain. Data is right shifted through the chain. Input data is shifted in at the MSB and shifted out from the LSB.

Input data is synchronized; this introduces a delay of one SGPIO_CLOCK cycle. This latency should be taken into account when changing a pin direction i.e. especially when emulating a high speed bi-directional bus.

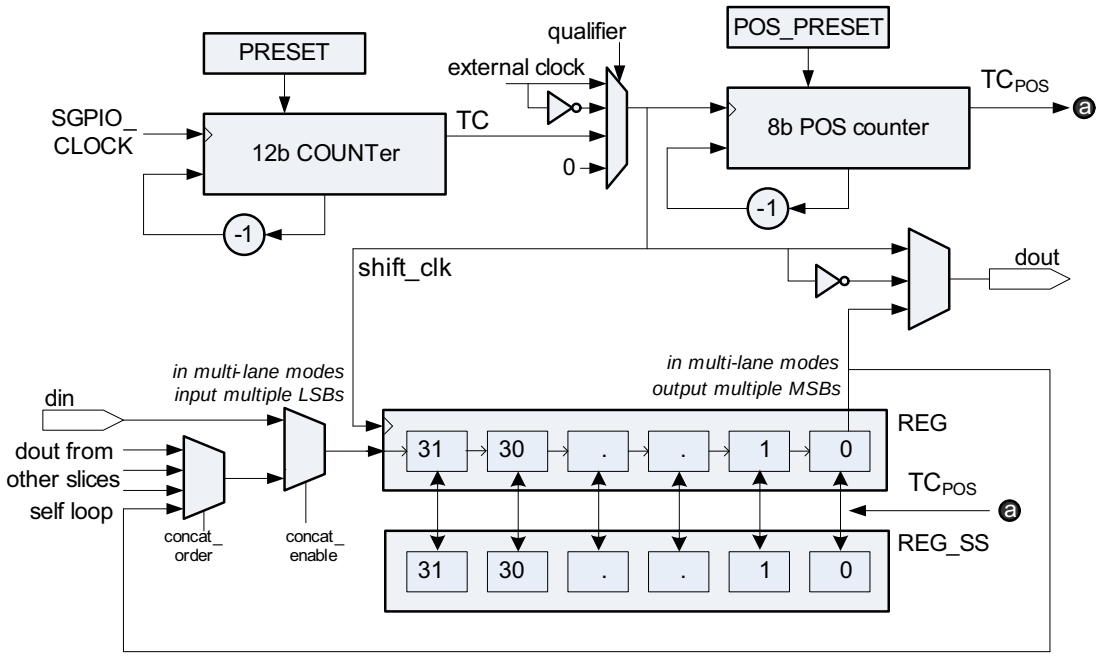


Fig 32. Basic operation of one slice

The shift clock controls the rate at which data is shifted through the chain. The bit shift increment is set by register parallel_mode to be 1, 2, 4 or 8 bits. Shifts of more than 1 are used in multi-lane modes.

The shift clock can be from an external source (see <tbd>) or generated locally.The shift clock can be qualified by an external pin or other slice. The local clock is made from a 12-bit down counter COUNT running at the SGPIO_CLOCK. The count duration can be preset with register PRESET. When COUNT reaches zero, it is loaded with PRESET.

The shift clock frequency is therefore equal to frequency(shift_clock) = frequency(SGPIO_CLK) / (PRESET+1).

Each time COUNT reaches zero the register shifts right; loading bit REG[31] with data captured from DIN and loading DOUT with bit REG[0]. Thus COUNT controls the serial data rate. When several slices are used to create an interface port the phase between the different slices can be controlled by using different initial COUNT value.

A second down counter (POS) controls when parallel data is loaded to and stored from the chain. The POS counter consists of an 8-bit counter that is decremented when COUNT equals zero. When POS reaches zero it is loaded with the PRESET_POS value. Slice data is exchanged using a double buffering scheme. When POS reaches zero the slice register REG content is swapped with the slice buffer REG_SS content. This gives the CPU more time to process the REG_SS buffer content. PRESET_POS can be used to indicate word boundaries for words that are not a multiple of 32 bit. A PRESET_POS value of zero will cause a REG swap with REG_SS after every shift clock.

18.7.1 Concatenation

Slices can be concatenated to increase the buffer size beyond REG_SS. This feature also enables creating PWM streams by implementing reverse catenation.

Concatenation is set by register SGPIO_MUX_CFG. The field CONCAT_ENABLE enables this feature. When multiple slices are concatenated, this field should be set the same for all involved slices. Field CONCAT_ORDER sets the concatenation size. For a size of zero, the self- loop mode, the slice output is routed back to the slice input. This mode can be used to create a repeating data stream.

Figure 33 below shows which slices can be concatenated. Slices are ordered in four groups, starting with slice A, B, C and D. Each group consists of four concatenated slices. For example, the slice A input can be extended with three extra slices using slice I, E and J. Or for output mode, slice I can be extended by slice A, I and E. Slices are furthermore ordered in four sub-groups starting with E, F, G and H. These sub-groups support concatenation of two slices. Slices A and B support concatenation of up to seven slices.

If multiple slices are concatenated then the concat_order for those slices should be consistent. For the above example, input slice A should be set as input (concat_enable=0). When concat_enable is not set, the concat_order value is a don't care. For slices I, E and J set concat_enable=1 and use concat_order =10 (4 slices).

Concatenated slices should have consistent shift clock and bit shift settings (in SLICE_MUX_CFG). When k slices are concatenated, PRESET_POS should be set to 0x20 x k - 1.

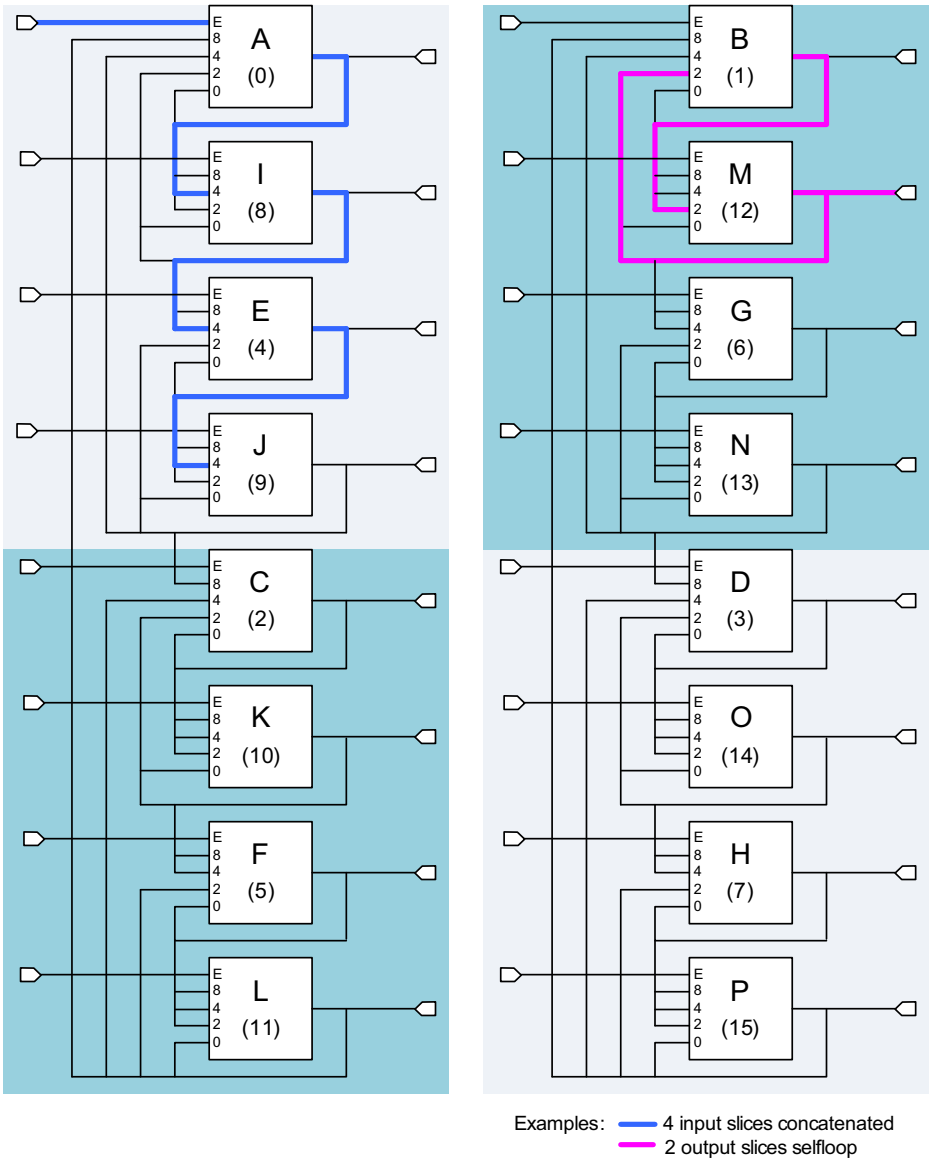


Fig 33. Concatenation interconnections

18.7.2 Pattern match

All slices feature pattern match functionality. This can be used for example to detect a start code. To use this functionality, REG_SS must be programmed with the pattern to be matched (Note that REG_SS will not be swapped with REG when POS reaches zero!). The MATCH_MODE bit must be set to 1. The input data is now compared to the programmed pattern. When a match is found the pattern match interrupt is raised.

Four slices (A, H, I and P) also support masking the pattern; MASK_x must be set for the pattern bits to be compared ('1' is compare). E.g. when looking for pattern 0x1234.xxxx.5678.9ABC, then REG should be set to 0x1234.xxxx.5678.9ABC and MASK to 0xFFFF.0000.FFFF.FFFF.

18.7.3 Pin multiplexing

The slice input and output are connected via a local pin multiplexer to the global pin multiplexer (Chapter 15).

Each pin can also be controlled for normal GPIO functions (input, output, output enable) using registers GPIO_INREG, GPIO_OUTREG and GPIO_OEREG.

Table 203 shows the output multiplexing scheme. The settings are controlled by register OUT_MUX_CFG. Table 244 shows the input multiplexing scheme. The settings are controlled by register SGPIO_MUX_CFG.

The 16 slices are denoted as A to P. A suffix indicates which slice bit connects to a pin, e.g. in 8-bit parallel input mode pins SGPIO0 to 7 connect to slice A bits 31-24.

Not all features are available for all pins. For example only pins SGPIO9 to 11 can be used as clock source input or as qualifier input or qualifier output.

18.7.4 Slice multiplexer

The slice multiplexer selects the external slice clock and slice clock qualifier. It is configured by register SGPIO_MUX_CFG.

The clock source can be from pins 8, 9, 10 or 11 or from slice D, H, O or P. Note that the slices that can be used as clock source cannot be sourced from themselves or other slices. The external clock is synchronized with the internal SGPIO clock. To prevent aliasing the source clock frequency should be less than half the SGPIO frequency.

The qualifier signals come from pins 8, 9, 10 or 11 or from slice A, H, I or P. Since a slice cannot feedback a qualifier to itself some other slices are used as well. Slice D can be qualifier for slice A and I and slice O for slice H and P.

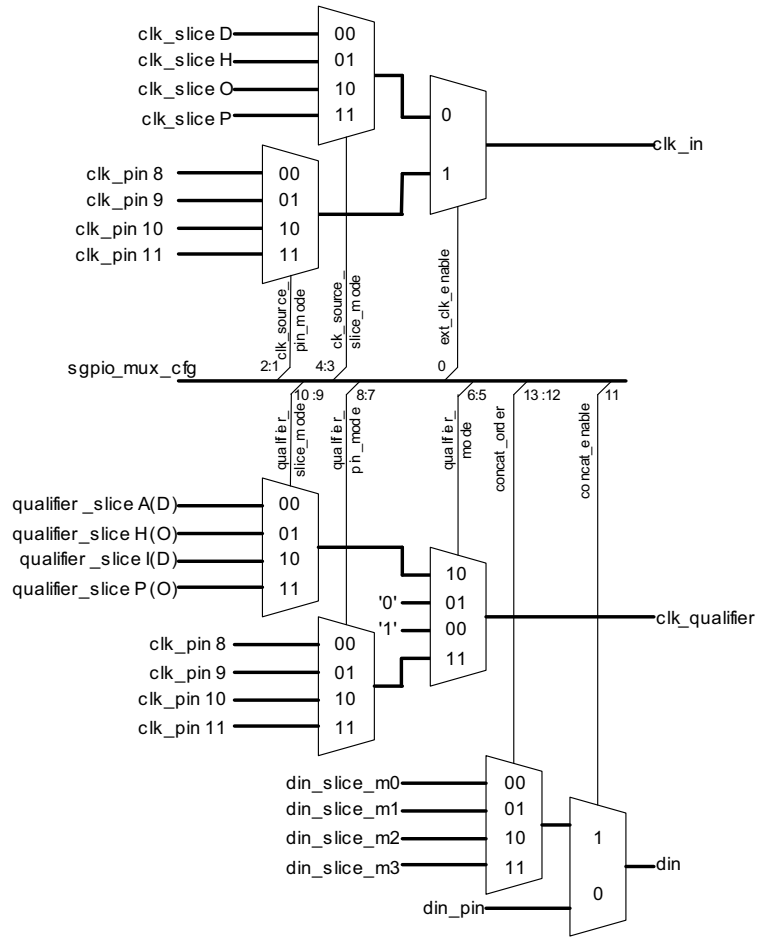


Fig 34. SGPIO_MUX_CFG slice multiplexer settings

Table 246. Slice I/O multiplexing

x = external; cl = clock; q = qualifier

SGPIO Pin	Input mode Parallel mode					
	8-bit	4-bit	2-bit	1-bit	Clock	Q
0	A24	A28	A30	A31		
1	A25	A29	A31	I31		
2	A26	A30	E30	E31		
3	A27	A31	E31	J31		
4	A28	C28	C30	C31		
5	A29	C29	C31	K31		
6	A30	C30	F30	F31		
7	A31	C31	F31	L31		
8	B31	B28	B30	B31	xcl	xq
9	B30	B29	B31	M31	xcl	xq

Table 246. Slice I/O multiplexing
x = external; cl = clock; q = qualifier

SGPIO Pin	Input mode Parallel mode					
	8-bit	4-bit	2-bit	1-bit	Clock	Q
10	B29	B30	G30	G31	xcl	xq
11	B28	B31	G31	N31	xcl	xq
12	B27	D28	D30	D31		
13	B26	D29	D31	O31		
14	B25	D30	H30	H31		
15	B24	D31	H31	P31		

18.7.5 Internal connections

SGPIO pins 10 and 12 can trigger the 12-bit ADC.

SGPIO pins 14 and 15 can trigger a GPDMA reset.

SGPIO pins 3 and 12 connect to SCT and timer capture inputs. For I2S applications, the SGPIO3 and SGPIO12 pins also support a signal which is divided by 128 (SGPIO3_DIV and SGPIO12_DIV).

A trigger signal can be created by the associated slice (for example pin 15 is controlled by slice P in 1-bit output mode) or by using these pins in GPIO mode.

18.8 Examples

Table 247. SGPIO applications on the LPC43xx

Example	Description
Logic analyzer	Capture up to 16 lines as fast as possible
Pattern generator	Shift out up to 16 lines as fast as possible, data input looped back (if wanted)
PWM	Similar to pattern generator, now looped. Slices can be concatenated to support longer patterns. Fixed frequency changing duty cycle is also possible by setting more bits high.
SPIFI	Uses 4-bit shift mode for data. Use oe to turn around data bus direction. GPIO for CS and one slice for clock.
I2C	Use oe path as data, tie the output data to 0.
I2S	Uses one slice per stereo channel. Use local or external clock for master or slave mode. Bit clock and master clock run at an integer multiple of the data clock.
SPDIF	Uses one slice per SPDIF audio interface.
TDM	Uses one slice per TDM audio interface.
UART	Use match feature to detect start bit, use qualifier if needed to capture first data.

18.8.1 Multi-channel I2S

18.8.1.1 I2S slice selection

A 5.1 channel I2S output interface in master mode requires 3 data outputs (SD[2:0]), 1 word select output (WS) and 1 clock output (SCK). In slave mode SCK becomes an input. This means that output SCK should also support clock input. This is supported by pins 8-10 which are mapped in serial output mode to slices B, M, G or N, let's use slice B. These slices should therefore not be used for SD or WS signals.

If an oversampled slave clock (MCK) is needed, use a slice that is capable to create clocks for other slices (D, H, O or P), e.g. use slice D. In slave mode MCK is an input. MCK is divided down to create the shift clock for Data, WS and SCK. In master mode MCK is an output.

The output audio data rate is 6 x Fs. For Fs = 192 kHz this becomes 1.152 MWps and thus relative low for a CPU frequency of 100+ MHz. Single slices can be used (without concatenation), for example slices A, I and E. The WS is made by slice J. This results in the following mapping:

Table 248. SGPIO Slice mapping for I2S 5.1

Slice: function	A,I,E : SD[2:0]	J : WS	B : SCK	D : MCK

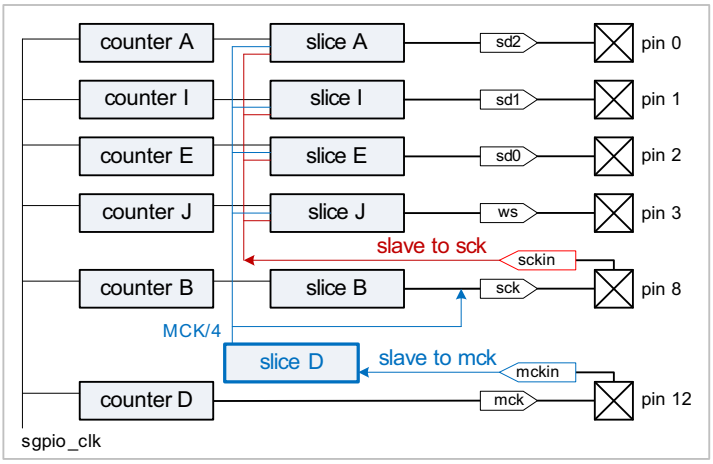


Fig 35. 5.1 channel I2S output mapped to SGPIO slices

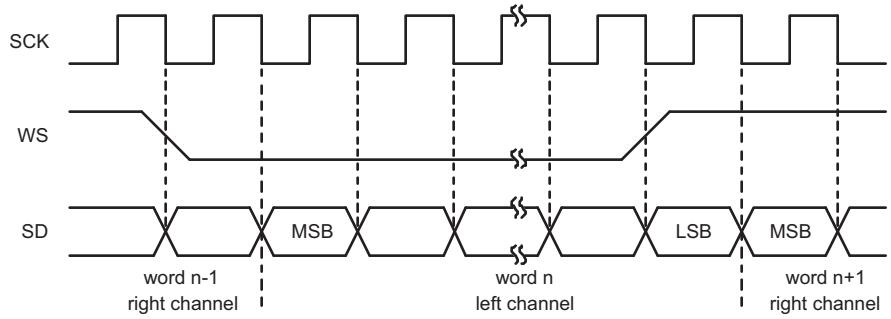


Fig 36. I2S configuration

18.8.1.2 I2S slice configuration

Using FS = 192 kHz and 32-bit audio samples provides the following parameters:

- WS=FS=192 kHz
- SCK = 2 x sample_width x FS = 2 x 32 x 192 kHz = 12.288 MHz, The slice shift clock should be twice this rate.
- MCK = oversampling_rate x SCK= 4 x SCK = 49.152 MHz.

Set the SGPIO IP clock (SGPIO_CLOCK) to 2 x MCK = 98.304 MHz.

The output width, 1-bit serial, and the output enable, statically controlled by GPIO are set as shown in Table 249.

Table 249. SGPIO setting for I2S 5.1, OUT_MUX_CFG register

OUT_MUX_CFGi	A,I,E (i=0,8,4)	J (i=9)	B (i=1)	D (i=3)
P_out_cfg	0000: dout_doutm1	dout_doutm1	dout_doutm1	dout_doutm1
P_oe_cfg	000: gpio_oe	gpio_oe	gpio_oe	gpio_oe
GPIO_OUTREG	1	1	1	1

In Master mode the shift clocks are generated by the local slice COUNTERs. The WS and CK slices contain repeating patterns and are concatenated in self-loop mode.

Table 250. SGPIO setting for I2S 5.1, SGPIO_MUX_CFG register

SGPIO_MUX_CFGi	A,I,E (i=0,8,4)	J (i=9)	B (i=1)	D (i=3)
ext_clk_enable	x	x	x	x
qualifier_mode	00: enable	00: enable	00: enable	00: enable
concat_enable	0; no	1; yes	1; yes	1; yes
concat_order	x	00; self loop	00; self loop	00; self loop

The data width is 1 bit and all slices are shifted 1 bit per clock.

Table 251. SGPIO setting for I2S 5.1, SLICE_MUX_CFG register

SLICE_MUX_CFGi	A,I,E (i=0,8,4)	J (i=9)	B (i=1)	D (i=3)
match_mode	0: no	0: no	0: no	0: no
clk_gen_mode	0: use COUNTER clk	0: use COUNTER clk	0: use COUNTER clk	0: use COUNTER clk
parallel_mode	00: 1bit per clock	00: 1bit per clock	00: 1bit per clock	00: 1bit per clock

All slice counters start at the same phase at value 0.

All data slices use for POS, the slice register length, and the audio sample width. For 32, 16, or 8-bit samples, POS is set to width-1: 0x1F, 0x0F or 0x07. For 16 (8)-bit samples POS can also be set to 0x1F and process 2 (4) samples. Slices that self-loop should always use the complete 32-bit size.

Table 252. SGPIO setting for I2S 5.1, slice

SLICE	A,I,E (i=0,8,4)	J (i=9)	B (i=1)	D (i=3)
PRESETi	7: f=clk/8	7: f=clk/8	3: f=clk/4	0: f=clk
COUNTi	0	0	0*[1]	0
POS_PRESETi	0x1F/0x0F/0x07	0x1F	0x1F	0x1F

[1] A value of 1 or 2 can be used to change the SCK phase relative to Data and WS.

In I2S slave mode, an external clock input is used. There are two cases:

1. SCK master mode: SCK is supplied at pin 8, this clock falling edge is used as shift clock for all slices. No MCK is used.
2. MCK master mode: MCK is supplied at pin 12 to slice D, where this clock is divided by the oversampling rate (=4). The output of slice D is used as shift clock for the other slices.

In SCK master mode, the SCK input is shift clock for the SD and WS signals. MCK is not used. The slice settings that are different for slave mode (1) with SCK supplied at pin 8 are shown in Table 253:

Table 253. SGPIO setting for I2S 5.1 (master mode, pin 8)

OUT_MUX_CFGi	A,I,E (i=0,8,4)	J (i=9)	B (i=1)	D (i=3)
P_out_cfg	0000: dout_doutm1	dout_doutm1	slice not used	slice not used
P_oe_cfg	000: gpio_oe	gpio_oe	slice not used	slice not used
GPIO_OUTREG	1	1	0	0
SGPIO_MUX_CFGi				
ext_clk_enable	1: pin	1: pin	slice not used	slice not used
clk_source_pin	00: pin8	00: pin8	slice not used	slice not used
qualifier_mode	00: enable	00: enable	slice not used	slice not used
SLICE_MUX_CFGi				
clk_gen_mode	1: use external clock	1: use external clock	slice not used	slice not used
inv_out_clk	1: inverted clock	1: inverted clock	slice not used	slice not used

Table 253. SGPIO setting for I2S 5.1 (master mode, pin 8)

PRESETi	counter not used	counter not used	counter not used	counter not used
COUNTi	counter not used	counter not used	counter not used	counter not used
POS_PRESETi	0x1F/0x0F/0x07	0x1F	slice not used	slice not used

In MCK master mode, the MCK input is divided down to generate the SCK and shift the SD and WS signals. The slice setting that are different for slave mode (1) with 4x oversampled clock MCK supplied at pin 9 are shown in :

Table 254. SGPIO setting for I2S 5.1 (master mode, pin 9)

OUT_MUX_CFGi	A,I,E (i=0,8,4)	J (i=9)	B (i=1)	D (i=3)
P_out_cfg	0000: dout_doutm1	0000: dout_doutm1	1000: clk	x
P_oe_cfg	000: gpio_oe	000: gpio_oe	000: gpio_oe	x
GPIO_OUTREG	1	1	1	0
SGPIO_MUX_CFGi				
ext_clk_enable	0: internal clock	0: internal clock	0: internal clock	1: pin
clk_source_pin	x	x	x	01: pin9
clk_source_slice	00: slice D	slice D	slice D	x
qualifier_mode	00: enable	00: enable	00: enable	00: enable
SLICE_MUX_CFGi				
clk_gen_mode	1: use external clock	1: use external clock	1: use external clock	1: use external clock
clk_capture_mode	1: use falling clock	1: use falling clock	not used	x
PRESETi	not used	not used	not used	3: f=MCK/4
COUNTi	not used	not used	not used	0
POS_PRESETi	0x1F/0x0F/0x07	0x1F	not used	0x1F

18.8.1.3 I2S slice programming

After configuration the data patterns are loaded in REGi and REG_SSi.

For SD the audio samples are loaded in REGi and RE_SSi. Data is shifted out starting from the LSB. After one sample is processed POS reaches countdown and REG_SSi is swapped with REGi to load the next sample. The CPU should update REG_SSi with a new sample before the next POS countdown when the current sample finishes.

The WS pattern repeats every 64-bit and is stored in REG9 and REG_SS9. To create a WS pattern as shown in Figure 35 for a 32-bit data width set REG9 = 0x0000.0001 and REG_SS9=0xFFFF.FFFE.

The SCK pattern is static and stored in REG3 and REG_SS3. To create a SCK pattern as shown in Figure 35 set REG1 = 0x5555.5555 and REG_SS1 = 0x5555.5555. To invert the clock phase use patterns 0xAAAA.AAAA instead.

MCK is not phase aligned to the other I2S signals. To toggle the output set REG3 = 0x5555.5555 and REG_SS3 = 0x5555.5555. In slave mode MCK should be divided by 4 to create SCK and the D and WS shift clock, this requires the pattern 11001100… hence REG3 = REG_SS3=0xCCCC.CCCC

All slices are started by enabling the COUNTERs by writing CTRL_ENABLE = 0x031B.

18.8.2 Camera interface example

The camera interface uses the following input signals:

- DIN[0:7] Data inputs
- HSYNC Horizontal synchronization input
- VSYNC Vertical synchronization input
- PIXCLX Pixel clock input

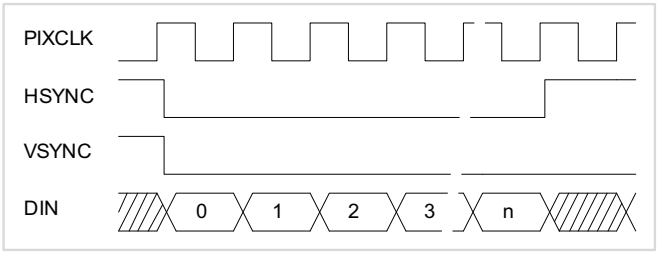


Fig 37. SGPIO camera interface configuration

The DIN input is captured on an 8-bit input. Wider inputs (10-bit or 12-bit) require an additional 2- or 4-bit input. Combining the 8- and 2- (or 4-) bit data to single 10- or 12-bit words must be done in software.

From slice A and B supporting the 8-bit input mode, select slice A. To minimize the CPU real time load, 8 slices are concatenated: A, I, E, J, C, K, F and L.

Inputs are captured on the PIXCLK falling edge. From pins 8-11 which can be used as clock input, choose pin 8. HSCYNC is used as qualifier for input data. Pins 8-11 can be used as qualifier; pin 8 is already used, therefore use pin 9. Pin 9 is also input for slice M. VSYNC uses one of the remaining free slices; for example G.

Output SGPIO15 (slice P) is used, as internal signal, to request a DMA transfer after a block of data has been transferred to local SRAM.

Table 255. SGPIO Slice mapping for camera interface

Slice: function	A,I,E,J,C,K,F,L: DIN[7:0]	Pin 8: PIXCLK	pin9/M: HSCYNC	G: VSYNC	P: DMA_REQ

18.8.2.1 Camera interface slice configuration

All interface signals are input only. The output configuration is don’t care.

Table 256. SGPIO setting for camera interface (OUT_MUX_CFG registers)

OUT_MUX_CFGi	A...L	Pin 8	M (i=12)	G (i=6)
P_out_cfg	x	x	x	x
P_oe_cfg	x	x	x	x
GPIO_OUTREG	0	0	0	0

Data is shifted in at PIXCLK (pin 8) using HSYNC (pin 9) as qualifier.

Table 257. SGPIO setting for camera interface (SGPIO_MUX_CFG registers)

SGPIO_MUX_CFGi	A...L	M (i=12)	G (i=6)
ext_clk_enable	1=pin	1=pin	1=pin
clk_source_pin_mode	00=Pin 8	00=Pin 8	00=Pin 8
qualifier_mode	11: external pin	00: enable	00: enable
qualifier_pin_mode	01: Pin 9	x	x
concat_enable	1: yes	0: no	0:; no
concat_order	11: 8 slices concat	x	x

The data width is 8 bit and all data slices are shifted 8 bit per clock.

Because an external clock is used as shift clock PRESET and COUNT are not used.

HSYNC and VSYNC are falling edge detected to raise an interrupt and indicate a start of line or frame. DIN is captured when HSYNC is low. Software should discard not needed data lines.

Alternatively the embedded codes 0xFF.00.00.YY (line start/end, frame start/end) are detected with slice A set to match_mode, YY contains the actual code. Note that in match_mode the shadow buffer is not used. To use the shadow buffer match_mode should be disabled once an embedded code has been detected.

Eight slices are concatenated giving 8 x 32/8 = 32 shifts until REG need to be shadowed, hence POS = 0x1F.

Table 258. SGPIO setting for camera interface (SLICE_MUX_CFG registers)

SLICE_MUX_CFGi	A...L	M (i=12)	G (i=6)
match_mode	0: no	0: no	0: no
clk_capture_mode	1: falling edge	1: falling edge	1: falling edge
clkgen_mode	1: use pin clock	1: use pin clock	1: use pin clock
data_capture_mode	x	01: detect falling edge	01: detect falling edge
parallel_mode	11: 1 byte/clk	x	x
inv_qualifier	1: inverted qualifier	x	x
PRESETi	x	x	x
COUNTi	x	x	x
POS_PRESETi	0x1F	x	x

The interface is started by starting slices A, I, E, J, C, K, F, L and M by writing CTRL_ENABLE=0x1F3F.

Data is captured at a falling PIXCLK when HSYNC is low. At a POS interrupt 32 data words are read from REG_SS and written to the data SRAM. Then SGPIO15 is toggled to request a GP-DMA transfer of 32 words from the data SRAM to the final destination. When DIN per line (or frame) is not a multiple of 32 data words then software should read at the end of a line (or frame) the POS counter to determine whether all data has been captured in REG.

19.1 How to read this chapter

The GPDMA is available on all LPC43xx parts.

Remark: The VADC is not available on parts LPC4350/30/20/10.

19.2 Basic configuration

The GPDMA is configured as follows:

- See Table 259 for clocking and power control.
- The GPDMA is reset by the DMA_RST (reset # 19).
- The DMAMUX register in the CREG block (see Table 41) selects between up to three peripherals for each GPDMA-to-peripheral line.
- The GPIO block, the WWDT, and the timers can be accessed by the GPDMA as memory-to-memory transfers.

Table 259. GPDMA clocking and power control

	Base clock	Branch clock	Operating frequency
GPDMA	BASE_M4_CLK	CLK_M4_DMA	204 MHz

19.3 Features

- Eight DMA channels. Each channel can support an unidirectional transfer.
- 16 DMA request lines.
- Single DMA and burst DMA request signals. Each peripheral connected to the DMA Controller can assert either a burst DMA request or a single DMA request. The DMA burst size is set by programming the DMA Controller.
- Memory-to-memory, memory-to-peripheral, peripheral-to-memory, and peripheral-to-peripheral transfers are supported.
- The GPIO block, the WWDT, and the timers can be accessed by the GPDMA as memory-to-memory transfers.
- Scatter or gather DMA is supported through the use of linked lists. This means that the source and destination areas do not have to occupy contiguous areas of memory.
- Hardware DMA channel priority.
- AHB slave DMA programming interface. The DMA Controller is programmed by writing to the DMA control registers over the AHB slave interface.
- Two AHB bus masters for transferring data. These interfaces transfer data when a DMA request goes active. Master 1 can access memories and peripherals, master 0 can access memories only.
- 32-bit AHB master bus width.

- Incrementing or non-incrementing addressing for source and destination.
- Programmable DMA burst size. The DMA burst size can be programmed to more efficiently transfer data.
- Internal four-word FIFO per channel.
- Supports 8, 16, and 32-bit wide transactions.
- Big-endian and little-endian support. The DMA Controller defaults to little-endian mode on reset.
- An interrupt to the processor can be generated on a DMA completion or when a DMA error has occurred.
- Raw interrupt status. The DMA error and DMA count raw interrupt status can be read prior to masking.

19.4 General description

The DMA controller allows peripheral-to memory, memory-to-peripheral, peripheral-to-peripheral, and memory-to-memory transactions. Each DMA stream provides unidirectional serial DMA transfers for a single source and destination. For example, a bi-directional port requires one stream for transmit and one for receives. The source and destination areas can each be either a memory region or a peripheral for master 1. Master 0 can only access memory (see Section 3.5).

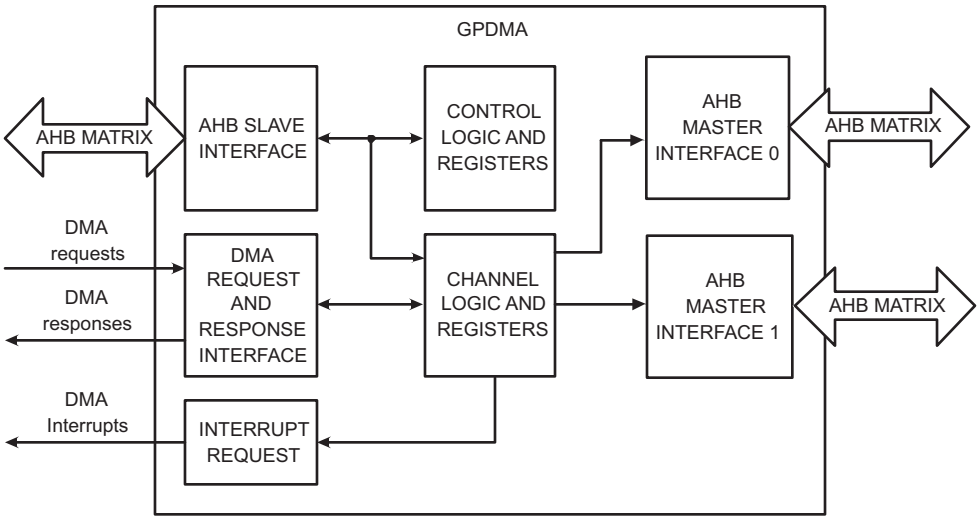


Fig 38. DMA controller block diagram

19.5 DMA system connections

The connection of the DMA Controller to supported peripheral devices is shown in Table 260. The LPC43xx supports multiple muxing options for each channel to connect peripherals to the DMA. The DMAMUX register in the CREG block controls which option is used (see Table 41).

Table 260. Peripheral connections to the DMA controller and matching flow control signals

Peripheral Number	DMA muxing option (see Table 41)	SREQ	BREQ
0	0x0	SPIFI	SPIFI
	0x1	-	SCT match 2
	0x2	-	SGPIO14
	0x3	-	Timer3 match 1
1	0x0	-	Timer 0 match 0
	0x1	-	USART0 transmit
	0x2	Reserved	Reserved
	0x3	Reserved	Reserved
2	0x0	-	Timer0 match 1
	0x1	-	USART0 receive
	0x2	Reserved	Reserved
	0x3	Reserved	Reserved
3	0x0	-	Timer1 match 0
	0x1	-	UART1 transmit
	0x2	-	I2S1 DMA request 0
	0x3	SSP1 transmit	SSP1 transmit
4	0x0	-	Timer1 match 1
	0x1	-	UART1 receive
	0x2	-	I2S1 DMA request 1
	0x3	SSP1 receive	SSP1 receive
5	0x0	-	Timer 2 match 0
	0x1	-	USART2 transmit
	0x2	SSP1 transmit	SSP1 transmit
	0x3	-	SGPIO15
6	0x0	-	Timer2 match 1
	0x1	-	USART2 receive
	0x2	SSP1 receive	SSP1 receive
	0x3	-	SGPIO14
7	0x0	-	Timer3 match 0
	0x1	-	USART3 transmit
	0x2	-	SCT DMA request 0
	0x3	Reserved	VADC write
8	0x0	-	Timer3 match 1
	0x1	-	USART3 receive
	0x2	-	SCT DMA request 1
	0x3	Reserved	VADC read

Table 260. Peripheral connections to the DMA controller and matching flow control signals

Peripheral Number	DMA muxing option (see Table 41)	SREQ	BREQ
9	0x0	SSP0 receive	SSP0 receive
	0x1	-	I2S0 DMA request 0
	0x2	-	SCT DMA request 1
	0x3	Reserved	Reserved
10	0x0	SSP0 transmit	SSP0 transmit
	0x1	-	I2S0 DMA request 1
	0x2	-	SCT match 0
	0x3	Reserved	Reserved
11	0x0	SSP1 receive	SSP1 receive
	0x1	-	SGPIO14
	0x2	-	USART0 transmit
	0x3	Reserved	Reserved
12	0x0	SSP1 transmit	SSP1 transmit
	0x1	-	SGPIO15
	0x2	-	USART0 receive
	0x3	Reserved	Reserved
13	0x0	-	ADC0
	0x1	Reserved	Reserved
	0x2	SSP1 receive	SSP1 receive
	0x3	-	USART3 receive
14	0x0	-	ADC1
	0x1	Reserved	Reserved
	0x2	SSP1 transmit	SSP1 transmit
	0x3	-	USART3 transmit
15	0x0	-	DAC
	0x1	-	SCT match 3
	0x2	-	SGPIO15
	0x3	-	Timer3 match 0

In addition to the peripherals listed in Table 260, the GPIOs, the WWDT, and the timers can be accessed by the GPDMA as a memory-to-memory transaction with no flow control.

19.5.1 DMA request signals

The DMA request signals are used by peripherals to request a data transfer. The DMA request signals indicate whether a single or burst transfer of data is required and whether the transfer is the last in the data packet. The DMA available request signals are:

BREQ[15:0] — Burst request signals. These cause a programmed burst number of data to be transferred.

SREQ[15:0] — Single transfer request signals. These cause a single data to be transferred. The DMA controller transfers a single transfer to or from the peripheral.

LBREQ[15:0] — Last burst request signals.

LSREQ[15:0] — Last single transfer request signals.

Note that most peripherals do not support all request types.

19.5.2 DMA response signals

The DMA response signals indicate whether the transfer initiated by the DMA request signal has completed. The response signals can also be used to indicate whether a complete packet has been transferred. The DMA response signals from the DMA controller are:

CLR[15:0] — DMA clear or acknowledge signals. The CLR signal is used by the DMA controller to acknowledge a DMA request from the peripheral.

TC[15:0] — DMA terminal count signals. The TC signal can be used by the DMA controller to indicate to the peripheral that the DMA transfer is complete.

19.6 Register description

The DMA Controller supports 8 channels. Each channel has registers specific to the operation of that channel. Other registers controls aspects of how source peripherals relate to the DMA Controller. There are also global DMA control and status registers.

Table 261. Register overview: GPDMA (base address 0x4000 2000)

Name	Access	Address offset	Description	Reset value	Reference
General registers					
INTSTAT	RO	0x000	DMA Interrupt Status Register	0x0000 0000	Table 262
INTTCSTAT	RO	0x004	DMA Interrupt Terminal Count Request Status Register	0x0000 0000	Table 263
INTTCCLEAR	WO	0x008	DMA Interrupt Terminal Count Request Clear Register	-	Table 264
INTERRSTAT	RO	0x00C	DMA Interrupt Error Status Register	0x0000 0000	Table 265
INTERRCLR	WO	0x010	DMA Interrupt Error Clear Register	-	Table 266
RAWINTTCSTAT	RO	0x014	DMA Raw Interrupt Terminal Count Status Register	0x0000 0000	Table 267
RAWINTERR STAT	RO	0x018	DMA Raw Error Interrupt Status Register	0x0000 0000	Table 268
ENBLDCHNS	RO	0x01C	DMA Enabled Channel Register	0x0000 0000	Table 269
SOFTBREQ	R/W	0x020	DMA Software Burst Request Register	0x0000 0000	Table 270
SOFTSREQ	R/W	0x024	DMA Software Single Request Register	0x0000 0000	Table 271
SOFTLBREQ	R/W	0x028	DMA Software Last Burst Request Register	0x0000 0000	Table 272
SOFTLSREQ	R/W	0x02C	DMA Software Last Single Request Register	0x0000 0000	Table 273
CONFIG	R/W	0x030	DMA Configuration Register	0x0000 0000	Table 274
SYNC	R/W	0x034	DMA Synchronization Register	0x0000 0000	Table 275
Channel 0 registers					

Table 261. Register overview: GPDMA (base address 0x4000 2000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
C0SRCADDR	R/W	0x100	DMA Channel 0 Source Address Register	0x0000 0000	Table 276
C0DESTADDR	R/W	0x104	DMA Channel 0 Destination Address Register	0x0000 0000	Table 277
C0LLI	R/W	0x108	DMA Channel 0 Linked List Item Register	0x0000 0000	Table 278
C0CONTROL	R/W	0x10C	DMA Channel 0 Control Register	0x0000 0000	Table 279
C0CONFIG	R/W	0x110	DMA Channel 0 Configuration Register	0x0000 0000[1]	Table 280
Channel 1 registers					
C1SRCADDR	R/W	0x120	DMA Channel 1 Source Address Register	0x0000 0000	Table 276
C1DESTADDR	R/W	0x124	DMA Channel 1 Destination Address Register	0x0000 0000	Table 277
C1LLI	R/W	0x128	DMA Channel 1 Linked List Item Register	0x0000 0000	Table 278
C1CONTROL	R/W	0x12C	DMA Channel 1 Control Register	0x0000 0000	Table 279
C1CONFIG	R/W	0x130	DMA Channel 1 Configuration Register	0x0000 0000[1]	Table 280
Channel 2 registers					
C2SRCADDR	R/W	0x140	DMA Channel 2 Source Address Register	0x0000 0000	Table 276
C2DESTADDR	R/W	0x144	DMA Channel 2 Destination Address Register	0x0000 0000	Table 277
C2LLI	R/W	0x148	DMA Channel 2 Linked List Item Register	0x0000 0000	Table 278
C2CONTROL	R/W	0x14C	DMA Channel 2 Control Register	0x0000 0000	Table 279
C2CONFIG	R/W	0x150	DMA Channel 2 Configuration Register	0x0000 0000[1]	Table 280
Channel 3 registers					
C3SRCADDR	R/W	0x160	DMA Channel 3 Source Address Register	0x0000 0000	Table 276
C3DESTADDR	R/W	0x164	DMA Channel 3 Destination Address Register	0x0000 0000	Table 277
C3LLI	R/W	0x168	DMA Channel 3 Linked List Item Register	0x0000 0000	Table 278
C3CONTROL	R/W	0x16C	DMA Channel 3 Control Register	0x0000 0000	Table 279
C3CONFIG	R/W	0x170	DMA Channel 3 Configuration Register	0x0000 0000[1]	Table 280
Channel 4 registers					
C4SRCADDR	R/W	0x180	DMA Channel 4 Source Address Register	0x0000 0000	Table 276
C4DESTADDR	R/W	0x184	DMA Channel 4 Destination Address Register	0x0000 0000	Table 277
C4LLI	R/W	0x188	DMA Channel 4 Linked List Item Register	0x0000 0000	Table 278
C4CONTROL	R/W	0x18C	DMA Channel 4 Control Register	0x0000 0000	Table 279
C4CONFIG	R/W	0x190	DMA Channel 4 Configuration Register	0x0000 0000[1]	Table 280
Channel 5 registers					
C5SRCADDR	R/W	0x1A0	DMA Channel 5 Source Address Register	0x0000 0000	Table 276
C5DESTADDR	R/W	0x1A4	DMA Channel 5 Destination Address Register	0x0000 0000	Table 277
C5LLI	R/W	0x1A8	DMA Channel 5 Linked List Item Register	0x0000 0000	Table 278
C5CONTROL	R/W	0x1AC	DMA Channel 5 Control Register	0x0000 0000	Table 279
C5CONFIG	R/W	0x1B0	DMA Channel 5 Configuration Register	0x0000 0000[1]	Table 280
Channel 6 registers					

Table 261. Register overview: GPDMA (base address 0x4000 2000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
C6SRCADDR	R/W	0x1C0	DMA Channel 6 Source Address Register	0x0000 0000	Table 276
C6DESTADDR	R/W	0x1C4	DMA Channel 6 Destination Address Register	0x0000 0000	Table 277
C6LLI	R/W	0x1C8	DMA Channel 6 Linked List Item Register	0x0000 0000	Table 278
C6CONTROL	R/W	01CC	DMA Channel 6 Control Register	0x0000 0000	Table 279
C6CONFIG	R/W	0x1D0	DMA Channel 6 Configuration Register	0x0000 0000[1]	Table 280
Channel 7 registers					
C7SRCADDR	R/W	0x1E0	DMA Channel 7 Source Address Register	0x0000 0000	Table 276
C7DESTADDR	R/W	0x1E4	DMA Channel 7 Destination Address Register	0x0000 0000	Table 277
C7LLI	R/W	0x1E8	DMA Channel 7 Linked List Item Register	0x0000 0000	Table 278
C7CONTROL	R/W	0x1EC	DMA Channel 7 Control Register	0x0000 0000	Table 279
C7CONFIG	R/W	0x1F0	DMA Channel 7 Configuration Register	0x0000 0000[1]	Table 280

[1] Bit 17 of this register is a read-only status flag.

19.6.1 DMA Interrupt Status Register

The IntStat Register is read-only and shows the status of the interrupts after masking. A HIGH bit indicates that a specific DMA channel interrupt request is active. The request can be generated from either the error or terminal count interrupt requests.

Table 262. DMA Interrupt Status register (INTSTAT, address 0x4000 2000) bit description

Bit	Symbol	Description	Reset value	Access
7:0	INTSTAT	Status of DMA channel interrupts after masking. Each bit represents one channel: 0 - the corresponding channel has no active interrupt request. 1 - the corresponding channel does have an active interrupt request.	0x00	RO
31:8	-	Reserved. Read undefined.	-	-

19.6.2 DMA Interrupt Terminal Count Request Status Register

The INTTCSTAT Register is read-only and indicates the status of the terminal count after masking.

Table 263. DMA Interrupt Terminal Count Request Status Register (INTTCSTAT, address 0x4000 2004) bit description

Bit	Symbol	Description	Reset value	Access
7:0	INTTCSTAT	Terminal count interrupt request status for DMA channels. Each bit represents one channel: 0 - the corresponding channel has no active terminal count interrupt request. 1 - the corresponding channel does have an active terminal count interrupt request.	0x00	RO
31:8	-	Reserved. Read undefined.	-	-

19.6.3 DMA Interrupt Terminal Count Request Clear Register

The INTTCCLEAR Register is write-only and clears one or more terminal count interrupt requests. When writing to this register, each data bit that is set HIGH causes the corresponding bit in the status register (IntTCStat) to be cleared. Data bits that are LOW have no effect.

Table 264. DMA Interrupt Terminal Count Request Clear Register (INTTCCLEAR, address 0x4000 2008) bit description

Bit	Symbol	Description	Reset value	Access
7:0	INTTCCLEAR	Allows clearing the Terminal count interrupt request (IntTCStat) for DMA channels. Each bit represents one channel: 0 - writing 0 has no effect. 1 - clears the corresponding channel terminal count interrupt.	0x00	WO
31:8	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

19.6.4 DMA Interrupt Error Status Register

The INTERRSTAT Register is read-only and indicates the status of the error request after masking.

Table 265. DMA Interrupt Error Status Register (INTERRSTAT, address 0x4000 200C) bit description

Bit	Symbol	Description	Reset value	Access
7:0	INTERRSTAT	Interrupt error status for DMA channels. Each bit represents one channel: 0 - the corresponding channel has no active error interrupt request. 1 - the corresponding channel does have an active error interrupt request.	0x00	RO
31:8	-	Reserved. Read undefined.	-	-

19.6.5 DMA Interrupt Error Clear Register

The INTERRCLR Register is write-only and clears the error interrupt requests. When writing to this register, each data bit that is HIGH causes the corresponding bit in the status register to be cleared. Data bits that are LOW have no effect on the corresponding bit in the register.

Table 266. DMA Interrupt Error Clear Register (INTERRCLR, address 0x4000 2010) bit description

Bit	Symbol	Description	Reset value	Access
7:0	INTERRCLR	Writing a 1 clears the error interrupt request (IntErrStat) for DMA channels. Each bit represents one channel: 0 - writing 0 has no effect. 1 - clears the corresponding channel error interrupt.	0x00	WO
31:8	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

19.6.6 DMA Raw Interrupt Terminal Count Status Register

The RAWINTTCSTAT Register is read-only and indicates which DMA channel is requesting a transfer complete (terminal count interrupt) prior to masking. (Note: the IntTCStat Register contains the same information after masking.) A HIGH bit indicates that the terminal count interrupt request is active prior to masking.

Table 267. DMA Raw Interrupt Terminal Count Status Register (RAWINTTCSTAT, address 0x4000 2014) bit description

Bit	Symbol	Description	Reset value	Access
7:0	RAWINTTCSTAT	Status of the terminal count interrupt for DMA channels prior to masking. Each bit represents one channel: 0 - the corresponding channel has no active terminal count interrupt request. 1 - the corresponding channel does have an active terminal count interrupt request.	0x00	RO
31:8	-	Reserved. Read undefined.	-	-

19.6.7 DMA Raw Error Interrupt Status Register

The RAWINTERRSTAT Register is read-only and indicates which DMA channel is requesting an error interrupt prior to masking. (Note: the IntErrStat Register contains the same information after masking.) A HIGH bit indicates that the error interrupt request is active prior to masking.

Table 268. DMA Raw Error Interrupt Status Register (RAWINTERRSTAT, address 0x4000 2018) bit description

Bit	Symbol	Description	Reset value	Access
7:0	RAWINTERRSTAT	Status of the error interrupt for DMA channels prior to masking. Each bit represents one channel: 0 - the corresponding channel has no active error interrupt request. 1 - the corresponding channel does have an active error interrupt request.	0x00	RO
31:8	-	Reserved. Read undefined.	-	-

19.6.8 DMA Enabled Channel Register

The ENBLDCHNS Register is read-only and indicates which DMA channels are enabled, as indicated by the Enable bit in the CCONFIG Register. A HIGH bit indicates that a DMA channel is enabled. A bit is cleared on completion of the DMA transfer.

Table 269. DMA Enabled Channel Register (ENBLDCHNS, address 0x4000 201C) bit description

Bit	Symbol	Description	Reset value	Access
7:0	ENABLEDCHANNELS	Enable status for DMA channels. Each bit represents one channel: 0 - DMA channel is disabled. 1 - DMA channel is enabled.	0x00	RO
31:8	-	Reserved. Read undefined.	-	-

19.6.9 DMA Software Burst Request Register

The SOFTBREQ Register is read/write and enables DMA burst requests to be generated by software. A DMA request can be generated for each source by writing a 1 to the corresponding register bit. A register bit is cleared when the transaction has completed. Reading the register indicates which sources are requesting DMA burst transfers. A request can be generated from either a peripheral or the software request register. Each bit is cleared when the related transaction has completed.

Table 270. DMA Software Burst Request Register (SOFTBREQ, address 0x4000 2020) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SOFTBREQ	Software burst request flags for each of 16 possible sources. Each bit represents one DMA request line or peripheral function (refer to Table 260 for peripheral hardware connections to the DMA controller): 0 - writing 0 has no effect. 1 - writing 1 generates a DMA burst request for the corresponding request line.	0x00	R/W
31:16	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

Note: It is recommended that software and hardware peripheral requests are not used at the same time.

19.6.10 DMA Software Single Request Register

The SOFTSREQ Register is read/write and enables DMA single transfer requests to be generated by software. A DMA request can be generated for each source by writing a 1 to the corresponding register bit. A register bit is cleared when the transaction has completed. Reading the register indicates which sources are requesting single DMA transfers. A request can be generated from either a peripheral or the software request register.

Table 271. DMA Software Single Request Register (SOFTSREQ, address 0x4000 2024) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SOFTSREQ	Software single transfer request flags for each of 16 possible sources. Each bit represents one DMA request line or peripheral function: 0 - writing 0 has no effect. 1 - writing 1 generates a DMA single transfer request for the corresponding request line.	0x00	R/W
31:16	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

19.6.11 DMA Software Last Burst Request Register

The SOFTLBREQ Register is read/write and enables DMA last burst requests to be generated by software. A DMA request can be generated for each source by writing a 1 to the corresponding register bit. A register bit is cleared when the transaction has completed. Reading the register indicates which sources are requesting last burst DMA transfers. A request can be generated from either a peripheral or the software request register.

Table 272. DMA Software Last Burst Request Register (SOFTLBREQ, address 0x4000 2028) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SOFTLBREQ	Software last burst request flags for each of 16 possible sources. Each bit represents one DMA request line or peripheral function: 0 - writing 0 has no effect. 1 - writing 1 generates a DMA last burst request for the corresponding request line.	0x00	R/W
31:16	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

19.6.12 DMA Software Last Single Request Register

The SOFTLSREQ Register is read/write and enables DMA last single requests to be generated by software. A DMA request can be generated for each source by writing a 1 to the corresponding register bit. A register bit is cleared when the transaction has completed. Reading the register indicates which sources are requesting last single DMA transfers. A request can be generated from either a peripheral or the software request register.

Table 273. DMA Software Last Single Request Register (SOFTLSREQ, address 0x4000 202C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	SOFTLSREQ	Software last single transfer request flags for each of 16 possible sources. Each bit represents one DMA request line or peripheral function: 0 - writing 0 has no effect. 1 - writing 1 generates a DMA last single transfer request for the corresponding request line.	0x00	R/W
31:16	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

19.6.13 DMA Configuration Register

The CONFIG Register is read/write and configures the operation of the DMA Controller. The endianness of the AHB master interface can be altered by writing to the M bit of this register. The AHB master interface is set to little-endian mode on reset.

Table 274. DMA Configuration Register (CONFIG, address 0x4000 2030) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	E		DMA Controller enable:	0x00	R/W
		0	Disabled (default). Disabling the DMA Controller reduces power consumption.		
		1	Enabled		
1	M0		AHB Master 0 endianness configuration:	0x00	R/W
		0	Little-endian mode (default).		
		1	Big-endian mode.		
2	M1		AHB Master 1 endianness configuration:	0x00	R/W
		0	Little-endian mode (default).		
		1	Big-endian mode.		
31:3	-		Reserved. Read undefined. Write reserved bits as zero.		

19.6.14 DMA Synchronization Register

The Sync Register is read/write and enables or disables synchronization logic for the DMA request signals. The DMA request signals consist of the BREQ[15:0], SREQ[15:0], LBREQ[15:0], and LSREQ[15:0]. A bit set to 0 enables the synchronization logic for a particular group of DMA requests. A bit set to 1 disables the synchronization logic for a particular group of DMA requests. This register is reset to 0, synchronization logic enabled.

Table 275. DMA Synchronization Register (SYNC, address 0x4000 2034) bit description

Bit	Symbol	Description	Reset value	Access
15:0	DMACSYNC	Controls the synchronization logic for DMA request signals. Each bit represents one set of DMA request lines as described in the preceding text: 0 - synchronization logic for the corresponding DMA request signals are disabled. 1 - synchronization logic for the corresponding request line signals are enabled.	0x00	R/W
31:16	-	Reserved. Read undefined. Write reserved bits as zero.	-	-

19.6.15 DMA Channel registers

The channel registers are used to program the eight DMA channels. These registers consist of:

- Eight CSRCADDR Registers.
- Eight CDESTADDR Registers.
- Eight CLLI Registers.
- Eight CCONTROL Registers.
- Eight CCONFIG Registers.

When performing scatter/gather DMA, the first four of these are automatically updated.

19.6.16 DMA Channel Source Address Registers

The eight read/write CSRCADDR Registers (C0SRCADDR to C7SRCADDR) contain the current source address (byte-aligned) of the data to be transferred. Each register is programmed directly by software before the appropriate channel is enabled. When the DMA channel is enabled this register is updated:

- As the source address is incremented.
- By following the linked list when a complete packet of data has been transferred.

Reading the register when the channel is active does not provide useful information. This is because by the time software has processed the value read, the address may have progressed. It is intended to be read only when the channel has stopped, in which case it shows the source address of the last item read.

Note: The source and destination addresses must be aligned to the source and destination widths.

Table 276. DMA Channel Source Address Registers (CSRCADDR, 0x4000 2100 (C0SRCADDR) to 0x4000 21E0 (C7SRCADDR)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	SRCADDR	DMA source address. Reading this register will return the current source address.	0x0000 0000	R/W

19.6.17 DMA Channel Destination Address registers

The eight read/write CDESTADDR Registers (C0DESTADDR to C7DESTADDR) contain the current destination address (byte-aligned) of the data to be transferred. Each register is programmed directly by software before the channel is enabled. When the DMA channel is enabled the register is updated as the destination address is incremented and by following the linked list when a complete packet of data has been transferred. Reading the register when the channel is active does not provide useful information. This is because by the time that software has processed the value read, the address may have progressed. It is intended to be read only when a channel has stopped, in which case it shows the destination address of the last item read.

Table 277. DMA Channel Destination Address registers (CDESTADDR, 0x4000 2104 (C0DESTADDR) to 0x4000 21E4 (C7DESTADDR)) bit description

Bit	Symbol	Description	Reset value	Access
31:0	DESTADDR	DMA Destination address. Reading this register will return the current destination address.	0x0000 0000	R/W

19.6.18 DMA Channel Linked List Item registers

The eight read/write CLLI Registers (C0LLI to C7LLI) contain a word-aligned address of the next Linked List Item (LLI). If the LLI is 0, then the current LLI is the last in the chain, and the DMA channel is disabled when all DMA transfers associated with it are completed. Programming this register when the DMA channel is enabled may have unpredictable side effects.

Table 278. DMA Channel Linked List Item registers (CLLI, 0x4000 2108 (C0LLI) to 0x4000 21E8 (C7LLI)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	LM		AHB master select for loading the next LLI:	0	R/W
		0	AHB Master 0.		
		1	AHB Master 1.		
1	R		Reserved, and must be written as 0, masked on read.	0	R/W
31:2	LLI		Linked list item. Bits [31:2] of the address for the next LLI. Address bits [1:0] are 0.	0x0000 0000	R/W

19.6.19 DMA channel control registers

The eight read/write CCONTROL Registers (C0CONTROL to C7CONTROL) contain DMA channel control information such as the transfer size, burst size, and transfer width. Each register is programmed directly by software before the DMA channel is enabled. When the channel is enabled the register is updated by following the linked list when a complete packet of data has been transferred. Reading the register while the channel is active does not give useful information. This is because by the time software has processed the value read, the channel may have advanced. It is intended to be read only when a channel has stopped.

Table 279. DMA Channel Control registers (CCONTROL, 0x4000 210C (C0CONTROL) to 0x4000 21EC (C7CONTROL)) bit description

Bit	Symbol	Value	Description	Reset value	Access
11:0	TRANSFERSIZE		Transfer size in number of transfers. A write to this field sets the size of the transfer when the DMA Controller is the flow controller. The transfer size value must be set before the channel is enabled. Transfer size is updated as data transfers are completed. A read from this field indicates the number of transfers completed on the destination bus. Reading the register when the channel is active does not give useful information because by the time that the software has processed the value read, the channel might have progressed. It is intended to be used only when a channel is enabled and then disabled. The transfer size value is not used if the DMA Controller is not the flow controller.	0x0	R/W
14:12	SBSIZE		Source burst size. Indicates the number of transfers that make up a source burst. This value must be set to the burst size of the source peripheral, or if the source is memory, to the memory boundary size (see Figure 3). The burst size is the amount of data that is transferred when the BREQ signal goes active in the source peripheral.	0x0	R/W
		0x0	Source burst size = 1		
		0x1	Source burst size = 4		
		0x2	Source burst size = 8		
		0x3	Source burst size = 16		
		0x4	Source burst size = 32		
		0x5	Source burst size = 64		
		0x6	Source burst size = 128		
		0x7	Source burst size = 256		
17:15	DBSIZE		Destination burst size. Indicates the number of transfers that make up a destination burst transfer request. This value must be set to the burst size of the destination peripheral or, if the destination is memory, to the memory boundary size. The burst size is the amount of data that is transferred when the BREQ signal goes active in the destination peripheral.	0x0	R/W
		0x0	Destination burst size = 1		
		0x1	Destination burst size = 4		
		0x2	Destination burst size = 8		
		0x3	Destination burst size = 16		
		0x4	Destination burst size = 32		
		0x5	Destination burst size = 64		
		0x6	Destination burst size = 128		
		0x7	Destination burst size = 256		

Table 279. DMA Channel Control registers (CCONTROL, 0x4000 210C (C0CONTROL) to 0x4000 21EC (C7CONTROL)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
20:18	SWIDTH		Source transfer width. Transfers wider than the AHB master bus width are illegal. The source and destination widths can be different from each other. The hardware automatically packs and unpacks the data as required. 0x3 to 0x7 - Reserved.	0x0	R/W
		0x0	Byte (8-bit)		
		0x1	Halfword (16-bit)		
		0x2	Word (32-bit)		
23:21	DWIDTH		Destination transfer width. Transfers wider than the AHB master bus width are not supported. The source and destination widths can be different from each other. The hardware automatically packs and unpacks the data as required. 0x3 to 0x7 - Reserved.	0x0	R/W
		0x0	Byte (8-bit)		
		0x1	Halfword (16-bit)		
		0x2	Word (32-bit)		
24	S		Source AHB master select:	0	R/W
		0	AHB Master 0 selected for source transfer.		
		1	AHB Master 1 selected for source transfer.		
25	D		Destination AHB master select: **Remark:** Only Master1 can access a peripheral. Master0 can only access memory.	0	R/W
		0	AHB Master 0 selected for destination transfer.		
		1	AHB Master 1 selected for destination transfer.		
26	SI		Source increment:	0	R/W
		0	The source address is not incremented after each transfer.		
		1	The source address is incremented after each transfer.		
27	DI		Destination increment:	0	R/W
		0	The destination address is not incremented after each transfer.		
		1	The destination address is incremented after each transfer.		
28	PROT1		Indicates that the access is in user mode or privileged mode:	0	R/W
		0	Access is in user mode		
		1	Access is in privileged mode.		
29	PROT2		Indicates that the access is bufferable or not bufferable:	0	R/W
		0	Access is not bufferable.		
		1	Access is bufferable.		
30	PROT3		Indicates that the access is cacheable or not cacheable:	0	R/W
		0	Access is not cacheable.		
		1	Access is cacheable.		
31	I		Terminal count interrupt enable bit.	0	R/W
		0	The terminal count interrupt is disabled.		
		1	The terminal count interrupt is enabled.		

19.6.19.1 Protection and access information

AHB access information is provided to the source and destination peripherals when a transfer occurs. The transfer information is provided by programming the DMA channel (the Prot bits of the CCONTROL Register, and the Lock bit of the CCONFIG Register). These bits are programmed by software. Peripherals can use this information if necessary.

19.6.20 Channel Configuration registers

The eight CCONFIG Registers (C0CONFIG to C7CONFIG) are read/write with the exception of bit[17] which is read-only. Used these to configure the DMA channel. The registers are not updated when a new LLI is requested.

Table 280. DMA Channel Configuration registers (CCONFIG, 0x4000 2110 (C0CONFIG) to 0x4000 21F0 (C7CONFIG)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	E		Channel enable. Reading this bit indicates whether a channel is currently enabled or disabled: The Channel Enable bit status can also be found by reading the ENBLDCHNS Register. A channel can be disabled by clearing the Enable bit. This causes the current AHB transfer (if one is in progress) to complete and the channel is then disabled. Any data in the FIFO of the relevant channel is lost. Restarting the channel by setting the Channel Enable bit has unpredictable effects, the channel must be fully re-initialized. The channel is also disabled, and Channel Enable bit cleared, when the last LLI is reached, the DMA transfer is completed, or if a channel error is encountered. If a channel must be disabled without losing data in the FIFO, the Halt bit must be set so that further DMA requests are ignored. The Active bit must then be polled until it reaches 0, indicating that there is no data left in the FIFO. Finally, the Channel Enable bit can be cleared.	0	R/W
		0	Channel disabled.		
		1	Channel enabled.		

Table 280. DMA Channel Configuration registers (CCONFIG, 0x4000 2110 (C0CONFIG) to 0x4000 21F0 (C7CONFIG)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
5:1	SRCPERIPHERAL		Source peripheral. This value selects the DMA source request peripheral. This field is ignored if the source of the transfer is from memory. See Table 260 for details.		R/W
		0x0	SPIFI/SCT match3/SGPIO14/Timer3 match 1		
		0x1	Timer0 match 0/USART0 transmit		
		0x2	Timer0 match 1/USART0 receive		
		0x3	Timer1 match 0/UART1 transmit/I2S1 DMA request 0/SSP1 transmit		
		0x4	Timer1 match 1/UART1 receive/I2S1 DMA request 1/SSP1 receive		
		0x5	Timer2 match 0/USART2 transmit/SSP1 transmit/SGPIO15		
		‘	Timer2 match 1/USART2 receive/SSP1 receive/SGPIO14		
		0x7	Timer3 match 0/UART3 transmit/SCT match 0/VADC write		
		0x8	Timer3 match 1/UART3 receive/SCT match 1/VADC read		
		0x9	SSP0 receive/I2S0 DMA request 0/SCT match1		
		0xA	SSP0 transmit/I2S DMA request 1/SCT match0		
		0xB	SSP1 receive/SGPIO14/USART0 transmit		
		0xC	SSP1 transmit/SGPIO15/USART0 receive		
		0xD	ADC0/SSP1 receive/USART3 receive		
		0xE	ADC1/SSP1 transmit/USART3 transmit		
		0xF	DAC/SCT match 3/SGPIO15/Timer3 match 0		

Table 280. DMA Channel Configuration registers (CCONFIG, 0x4000 2110 (C0CONFIG) to 0x4000 21F0 (C7CONFIG)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
10:6	DESTPERIPHERAL		Destination peripheral. This value selects the DMA destination request peripheral. This field is ignored if the destination of the transfer is to memory. See Table 260 for details.		R/W
		0x0	SPIFI/SCT match3/SGPIO14/Timer3 match 1		
		0x1	Timer0 match 0/USART0 transmit		
		0x2	Timer0 match 1/USART0 receive		
		0x3	Timer1 match 0/UART1 transmit/I2S1 DMA request 0/SSP1 transmit		
		0x4	Timer1 match 1/UART1 receive/I2S1 DMA request 1/SSP1 receive		
		0x5	Timer2 match 0/USART2 transmit/SSP1 transmit/SGPIO15		
		0x6	Timer2 match 1/USART2 receive/SSP1 receive/SGPIO14		
		0x7	Timer3 match 0/UART3 transmit/SCT match 0/VADC write		
		0x8	Timer3 match 1/UART3 receive/SCT match 1/VADC read		
		0x9	SSP0 receive/I2S0 DMA request 0/SCT match1		
		0xA	SSP0 transmit/I2S DMA request 1/SCT match0		
		0xB	SSP1 receive/SGPIO14/USART0 transmit		
		0xC	SSP1 transmit/SGPIO15/USART0 receive		
		0xD	ADC0/SSP1 receive/USART3 receive		
		0xE	ADC1/SSP1 transmit/USART3 transmit		
		0xF	DAC/SCT match 3/SGPIO15/Timer3 match 0		
13:11	FLOWCNTRL		Flow control and transfer type. This value indicates the flow controller and transfer type. The flow controller can be the DMA Controller, the source peripheral, or the destination peripheral. The transfer type can be memory-to-memory, memory-to-peripheral, peripheral-to-memory, or peripheral-to-peripheral. Refer to Table 281 for the encoding of this field.		R/W
		0x0	Memory to memory (DMA control)		
		0x1	Memory to peripheral (DMA control)		
		0x2	Peripheral to memory (DMA control)		
		0x3	Source peripheral to destination peripheral (DMA control)		
		0x4	Source peripheral to destination peripheral (destination control)		
		0x5	Memory to peripheral (peripheral control)		
		0x6	Peripheral to memory (peripheral control)		
		0x7	Source peripheral to destination peripheral (source control)		
14	IE		Interrupt error mask. When cleared, this bit masks out the error interrupt of the relevant channel.		R/W
15	ITC		Terminal count interrupt mask. When cleared, this bit masks out the terminal count interrupt of the relevant channel.		R/W
16	L		Lock. When set, this bit enables locked transfers.		R/W

Table 280. DMA Channel Configuration registers (CCONFIG, 0x4000 2110 (C0CONFIG) to 0x4000 21F0 (C7CONFIG)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
17	A		Active: 0 = there is no data in the FIFO of the channel. 1 = the channel FIFO has data. This value can be used with the Halt and Channel Enable bits to cleanly disable a DMA channel. This is a read-only bit.		RO
18	H		Halt: 0 = enable DMA requests. 1 = ignore further source DMA requests. The contents of the channel FIFO are drained. This value can be used with the Active and Channel Enable bits to cleanly disable a DMA channel.		R/W
		0	Enable DMA requests.		
		1	Ignore further source DMA requests.		
31:19	-		Reserved, do not modify, masked on read.		-

19.6.20.1 Lock control

The lock control may set the lock bit by writing a 1 to bit 16 of the CCONFIG Register. When a burst occurs, the AHB arbiter will not de-grant the master during the burst until the lock is deasserted. The DMA Controller can be locked for a a single burst such as a long source fetch burst or a long destination drain burst. The DMA Controller does not usually assert the lock continuously for a source fetch burst followed by a destination drain burst.

There are situations when the DMA Controller asserts the lock for source transfers followed by destination transfers. This is possible when internal conditions in the DMA Controller permit it to perform a source fetch followed by a destination drain back-to-back.

19.6.20.2 Flow control and transfer type

Table 281 lists the bit values of the three flow control and transfer type bits identified in Table Table 280.

Table 281. Flow control and transfer type bits

Bit value	Transfer type	Controller
000	Memory to memory	DMA
001	Memory to peripheral	DMA
010	Peripheral to memory	DMA
011	Source peripheral to destination peripheral	DMA
100	Source peripheral to destination peripheral	Destination peripheral
101	Memory to peripheral	Peripheral
110	Peripheral to memory	Peripheral
111	Source peripheral to destination peripheral	Source peripheral

19.7 Functional description

19.7.1 DMA controller functional description

The DMA Controller enables peripheral-to-memory, memory-to-peripheral, peripheral-to-peripheral, and memory-to-memory transactions. Each DMA stream provides unidirectional serial DMA transfers for a single source and destination. For example, a bidirectional port requires one stream for transmit and one for receive. The source and destination areas can each be either a memory region or a peripheral, and can be accessed through the AHB master. Figure 38 shows a block diagram of the DMA Controller.

The functions of the DMA Controller are described in the following sections.

19.7.1.1 AHB slave interface

All transactions to DMA Controller registers on the AHB slave interface are 32 bits wide. Eight bit and 16-bit accesses are not supported and will result in an exception.

19.7.1.2 Control logic and register bank

The register block stores data written or to be read across the AHB interface.

19.7.1.3 DMA request and response interface

See DMA Interface description for information on the DMA request and response interface.

19.7.1.4 Channel logic and channel register bank

The channel logic and channel register bank contains registers and logic required for each DMA channel.

19.7.1.5 Interrupt request

The interrupt request generates the interrupt to the ARM processor.

19.7.1.6 AHB master interface

The DMA Controller contains two AHB master interfaces. Each AHB master is capable of dealing with all types of AHB transactions, including:

- Split, retry, and error responses from slaves. If a peripheral performs a split or retry, the DMA Controller stalls and waits until the transaction can complete.
- Locked transfers for source and destination of each stream.
- Setting of protection bits for transfers on each stream.

19.7.1.6.1 Bus and transfer widths

The physical width of the AHB bus is 32 bits. Source and destination transfers can be of differing widths and can be the same width or narrower than the physical bus width. The DMA Controller packs or unpacks data as appropriate.

19.7.1.6.2 Endian behavior

The DMA Controller can cope with both little-endian and big-endian addressing. Software can set the endianness of each AHB master individually.

Internally the DMA Controller treats all data as a stream of bytes instead of 16-bit or 32-bit quantities. This means that when performing mixed-endian activity, where the endianness of the source and destination are different, byte swapping of the data within the 32-bit data bus is observed.

Note: If byte swapping is not required, then use of different endianness between the source and destination addresses must be avoided. Table 282 shows endian behavior for different source and destination combinations.

Table 282. Endian behavior

Source endian	Destination endian	Source width	Destination width	Source transfer no/byte lane	Source data	Destination transfer no/byte lane	Destination data
Little	Little	16	16	1/[7:0] 1/[15:8] 2/[23:16] 2/[31:24]	21 43 65 87	1/[15:0] 2/[31:16]	43214321 87658765
Little	Little	16	32	1/[7:0] 1/[15:8] 2/[23:16] 2/[31:24]	21 43 65 87	1/[31:0]	87654321
Little	Little	8	8	1/[7:0]	21	1/[7:0]	21212121
Little	Little	32	8	1/[7:0] 2/[15:8] 1/[15:8] 3/[23:16] 1/[23:16] 4/[31:24] 1/[31:24]	21 43 43 65 65 87 87	1/[7:0] 2/[15:8] 3/[23:16] 4/[31:24]	21212121 43434343 65656565 87878787
Little	Little	8	16	1/[7:0]	21	1/[15:0]	43214321
Little	Little	32	16	1/[7:0] 2/[15:8] 1/[15:8] 3/[23:16] 1/[23:16] 4/[31:24] 1/[31:24]	21 43 43 65 65 87 87	1/[15:0] 2/[31:16]	43214321 87658765
Little	Little	8	32	1/[7:0] 2/[15:8] 3/[23:16] 4/[31:24]	21 43 65 87	1/[31:0]	87654321
Little	Little	16	8	1/[7:0] 1/[15:8] 2/[23:16] 2/[31:24]	21 43 65 87	1/[7:0] 2/[15:8] 3/[23:16] 4/[31:24]	21212121 43434343 65656565 87878787

Little	Little	16	16	1/[7:0] 1/[15:8] 2/[23:16] 2/[31:24]	21 43 65 87	1/[15:0] 2/[31:16]	43214321 87658765
Little	Little	16	32	1/[7:0] 1/[15:8] 2/[23:16] 2/[31:24]	21 43 65 87	1/[31:0]	87654321
Little	Little	32	8	1/[7:0] 1/[15:8] 1/[23:16] 1/[31:24]	21 43 65 87	1/[7:0] 2/[15:8] 3/[23:16] 4/[31:24]	21212121 43434343 65656565 87878787
Little	Little	32	16	1/[7:0] 1/[15:8] 1/[23:16] 1/[31:24]	21 43 65 87	1/[15:0] 2/[31:16]	43214321 87658765

Table 282. Endian behavior *…continued*

Source endian	Destination endian	Source width	Destination width	Source transfer no/byte lane	Source data	Destination transfer no/byte lane	Destination data
Big	Big	16	32	1/[31:24]	12	1/[31:0]	12345678
				1/[23:16]	34		
				2/[15:8]	56		
				2/[7:0]	78		
Big	Big	32	8	1/[31:24]	12	1/[31:24]	12121212
				1/[23:16]	34	2/[23:16]	34343434
				1/[15:8]	56	3/[15:8]	56565656
				1/[7:0]	78	4/[7:0]	78787878
Big	Big	32	16	1/[31:24]	12	1/[15:0]	12341234
				1/[23:16]	34	2/[31:16]	56785678
				1/[15:8]	56		
				1/[7:0]	78		
Big	Big	32	32	1/[31:24]	12	1/[31:0]	12345678
				1/[23:16]	34		
				1/[15:8]	56		
				1/[7:0]	78		

19.7.1.6.3 Error conditions

An error during a DMA transfer is flagged directly by the peripheral by asserting an Error response on the AHB bus during the transfer. The DMA Controller automatically disables the DMA stream after the current transfer has completed, and can optionally generate an error interrupt to the CPU. This error interrupt can be masked.

19.7.1.7 Channel hardware

Each stream is supported by a dedicated hardware channel, including source and destination controllers, as well as a FIFO. This enables better latency than a DMA controller with only a single hardware channel shared between several DMA streams and simplifies the control logic.

19.7.1.8 DMA request priority

DMA channel priority is fixed. DMA channel 0 has the highest priority and DMA channel 7 has the lowest priority.

If the DMA Controller is transferring data for the lower priority channel and then the higher priority channel goes active, it completes the number of transfers delegated to the master interface by the lower priority channel before switching over to transfer data for the higher priority channel. In the worst case this is as large as a one quadword.

It is recommended that memory-to-memory transactions use the lowest priority channel. Otherwise other AHB bus masters are prevented from accessing the bus during DMA Controller memory-to-memory transfer.

19.7.1.9 Interrupt generation

A combined interrupt output is generated as an OR function of the individual interrupt requests of the DMA Controller and is connected to the interrupt controller.

19.8 Using the DMA controller

19.8.1 Programming the DMA controller

All accesses to the DMA Controller internal register must be word (32-bit) reads and writes.

19.8.1.1 Enabling the DMA controller

To enable the DMA controller set the Enable bit in the CONFIG register.

19.8.1.2 Disabling the DMA controller

To disable the DMA controller:

- Read the ENBLDCHNS register and ensure that all the DMA channels have been disabled. If any channels are active, see Disabling a DMA channel.
- Disable the DMA controller by writing 0 to the DMA Enable bit in the CONFIG register.

19.8.1.3 Enabling a DMA channel

To enable the DMA channel set the channel enable bit in the relevant DMA channel configuration register. Note that the channel must be fully initialized before it is enabled.

19.8.1.4 Disabling a DMA channel

A DMA channel can be disabled in three ways:

- By writing directly to the channel enable bit. Any outstanding data in the FIFO's is lost if this method is used.
- By using the active and halt bits in conjunction with the channel enable bit.
- By waiting until the transfer completes. This automatically clears the channel.

Disabling a DMA channel and losing data in the FIFO

Clear the relevant channel enable bit in the relevant channel configuration register. The current AHB transfer (if one is in progress) completes and the channel is disabled. Any data in the FIFO is lost.

Disabling the DMA channel without losing data in the FIFO

- Set the halt bit in the relevant channel configuration register. This causes any future DMA request to be ignored.
- Poll the active bit in the relevant channel configuration register until it reaches 0. This bit indicates whether there is any data in the channel that has to be transferred.
- Clear the channel enable bit in the relevant channel configuration register

19.8.1.5 Setting up a new DMA transfer

To set up a new DMA transfer:

If the channel is not set aside for the DMA transaction:

1. Read the ENBLDCHNS controller register and find out which channels are inactive.
2. Choose an inactive channel that has the required priority.

3. Program the DMA controller

19.8.1.6 Halting a DMA channel

Set the halt bit in the relevant DMA channel configuration register. The current source request is serviced. Any further source DMA request is ignored until the halt bit is cleared.

19.8.1.7 Programming a DMA channel

1. Choose a free DMA channel with the priority needed. DMA channel 0 has the highest priority and DMA channel 7 the lowest priority.
2. Clear any pending interrupts on the channel to be used by writing to the IntTCClear and INTERRCLEAR register. The previous channel operation might have left interrupt active.
3. Write the source address into the CSRCADDR register.
4. Write the destination address into the CDESTADDR register.
5. Write the address of the next LLI into the CLLI register. If the transfer comprises of a single packet of data then 0 must be written into this register.
6. Write the control information into the CCONTROL register.
7. Write the channel configuration information into the CCONFIG register. If the enable bit is set then the DMA channel is automatically enabled.

19.8.2 Flow control

The peripheral that controls the length of the packet is known as the flow controller. The flow controller is usually the DMA Controller where the packet length is programmed by software before the DMA channel is enabled. If the packet length is unknown when the DMA channel is enabled, either the source or destination peripherals can be used as the flow controller.

For simple or low-performance peripherals that know the packet length (that is, when the peripheral is the flow controller), a simple way to indicate that a transaction has completed is for the peripheral to generate an interrupt and enable the processor to reprogram the DMA channel.

The transfer size value (in the CCONTROL register) is ignored if a peripheral is configured as the flow controller.

When the DMA transfer is completed:

1. The DMA Controller issues an acknowledge to the peripheral in order to indicate that the transfer has finished.
2. A TC interrupt is generated, if enabled.
3. The DMA Controller moves on to the next LLI.

The following sections describe the DMA Controller data flow sequences for the four allowed transfer types:

- Memory-to-peripheral (master 1 only).
- Peripheral-to-memory (master 1 only).
- Memory-to-memory.

- Peripheral-to-peripheral (master 1 only).

Each transfer type can have either the peripheral or the DMA Controller as the flow controller so there are eight possible control scenarios.

Table 283 indicates the request signals used for each type of transfer.

Table 283. DMA request signal usage

Transfer direction	Request generator	Flow controller
Memory-to-peripheral	Peripheral	DMA Controller
Memory-to-peripheral	Peripheral	Peripheral
Peripheral-to-memory	Peripheral	DMA Controller
Peripheral-to-memory	Peripheral	Peripheral
Memory-to-memory	DMA Controller	DMA Controller
Source peripheral to destination peripheral	Source peripheral and destination peripheral	Source peripheral
Source peripheral to destination peripheral	Source peripheral and destination peripheral	Destination peripheral
Source peripheral to destination peripheral	Source peripheral and destination peripheral	DMA Controller

19.8.2.1 Peripheral-to-memory or memory-to-peripheral DMA flow

For a peripheral-to-memory or memory-to-peripheral DMA flow, the following sequence occurs:

1. Program and enable the DMA channel.
2. Wait for a DMA request.
3. The DMA Controller starts transferring data when:
   - The DMA request goes active.
   - The DMA stream has the highest pending priority.
   - The DMA Controller is the bus master of the AHB bus.
4. If an error occurs while transferring the data, an error interrupt is generated and disables the DMA stream, and the flow sequence ends.
5. Decrement the transfer count if the DMA Controller is performing the flow control.
6. If the transfer has completed (indicated by the transfer count reaching 0, if the DMA Controller is performing flow control, or by the peripheral sending a DMA request, if the peripheral is performing flow control):
   - The DMA Controller responds with a DMA acknowledge.
   - The terminal count interrupt is generated (this interrupt can be masked).
   - If the CLLI Register is not 0, then reload the CSRCADDR, CDESTADDR, CLLI, and CCONTROL registers and go to back to step 2. However, if CLLI is 0, the DMA stream is disabled and the flow sequence ends.

19.8.2.2 Peripheral-to-peripheral DMA flow

For a peripheral-to-peripheral DMA flow, the following sequence occurs:

1. Program and enable the DMA channel.
2. Wait for a source DMA request.
3. The DMA Controller starts transferring data when:

- The DMA request goes active.
- The DMA stream has the highest pending priority.
- The DMA Controller is the bus master of the AHB bus.

4. If an error occurs while transferring the data an error interrupt is generated, the DMA stream is disabled, and the flow sequence ends.
5. Decrement the transfer count if the DMA Controller is performing the flow control.
6. If the transfer has completed (indicated by the transfer count reaching 0 if the DMA Controller is performing flow control, or by the peripheral sending a DMA request if the peripheral is performing flow control):
   - The DMA Controller responds with a DMA acknowledge to the source peripheral.
   - Further source DMA requests are ignored.
7. When the destination DMA request goes active and there is data in the DMA Controller FIFO, transfer data into the destination peripheral.
8. If an error occurs while transferring the data, an error interrupt is generated, the DMA stream is disabled, and the flow sequence ends.
9. If the transfer has completed it is indicated by the transfer count reaching 0 if the DMA Controller is performing flow control, or by the sending a DMA request if the peripheral is performing flow control. The following happens:
   - The DMA Controller responds with a DMA acknowledge to the destination peripheral.
   - The terminal count interrupt is generated (this interrupt can be masked).
   - If the CLLI Register is not 0, then reload the CSRCADDR, CDESTADDR, CLLI, and CCONTROL Registers and go to back to step 2. However, if CLLI is 0, the DMA stream is disabled and the flow sequence ends.

19.8.2.3 Memory-to-memory DMA flow

For a memory-to-memory DMA flow the following sequence occurs:

1. Program and enable the DMA channel.
2. Transfer data whenever the DMA channel has the highest pending priority and the DMA Controller gains mastership of the AHB bus.
3. If an error occurs while transferring the data, generate an error interrupt and disable the DMA stream.
4. Decrement the transfer count.
5. If the count has reached zero:
   - Generate a terminal count interrupt (the interrupt can be masked).
   - If the CLLI Register is not 0, then reload the CSRCADDR, CDESTADDR, CLLI, and CCONTROL Registers and go to back to step 2. However, if CLLI is 0, the DMA stream is disabled and the flow sequence ends.

Note: Memory-to-memory transfers should be programmed with a low channel priority, otherwise other DMA channels cannot access the bus until the memory-to-memory transfer has finished, or other AHB masters cannot perform any transaction.

19.8.3 Interrupt requests

Interrupt requests can be generated when an AHB error is encountered or at the end of a transfer (terminal count), after all the data corresponding to the current LLI has been transferred to the destination. The interrupts can be masked by programming bits in the relevant CCONTROL and CCONFIG Channel Registers. Interrupt status registers are provided which group the interrupt requests from all the DMA channels prior to interrupt masking (RAWINTTCSTAT and RAWINTERRSTAT), and after interrupt masking (INTTCSTAT and INTERRSTAT). The INTSTAT Register combines both the INTTCSTAT and INTERRSTAT requests into a single register to enable the source of an interrupt to be quickly found. Writing to the INTTCCLEAR or the INTERRCLR Registers with a bit set HIGH enables selective clearing of interrupts.

19.8.3.1 Hardware interrupt sequence flow

When a DMA interrupt request occurs, the Interrupt Service Routine needs to:

1. Read the INTTCSTAT Register to determine whether the interrupt was generated due to the end of the transfer (terminal count). A HIGH bit indicates that the transfer completed. If more than one request is active, it is recommended that the highest priority channels be checked first.
2. Read the INTERRSTAT Register to determine whether the interrupt was generated due to an error occurring. A HIGH bit indicates that an error occurred.
3. Service the interrupt request.
4. For a terminal count interrupt, write a 1 to the relevant bit of the INTTCCLR Register. For an error interrupt write a 1 to the relevant bit of the INTERRCLR Register to clear the interrupt request.

19.8.4 Address generation

Address generation can be either incrementing or non-incrementing (address wrapping is not supported).

Some devices, especially memories, disallow burst accesses across certain address boundaries. The DMA controller assumes that this is the case with any source or destination area, which is configured for incrementing addressing. This boundary is assumed to be aligned with the specified burst size. For example, if the channel is set for 16-transfer burst to a 32-bit wide device then the boundary is 64-bytes aligned (that is address bits [5:0] equal 0). If a DMA burst is to cross one of these boundaries, then, instead of a burst, that transfer is split into separate AHB transactions.

Note: When transferring data to or from the SDRAM, the SDRAM access must always be programmed to 32 bit accesses. The SDRAM memory controller does not support AHB-INCR4 or INCR8 bursts using halfword or byte transfer-size. Start address in SDRAM should always be aligned to a burst boundary address.

19.8.4.1 Word-aligned transfers across a boundary

The channel is configured for 16-transfer bursts, each transfer 32-bits wide, to a destination for which address incrementing is enabled. The start address for the current burst is 0x0C000024, the next boundary (calculated from the burst size and transfer width) is 0x0C000040.

The transfer will be split into two AHB transactions:

- a 7-transfer burst starting at address 0x0C000024
- a 9-transfer burst starting at address 0x0C000040.

19.8.5 Scatter/gather

Scatter/gather is supported through the use of linked lists. This means that the source and destination areas do not have to occupy contiguous areas in memory. Where scatter/gather is not required, the CLLI Register must be set to 0.

The source and destination data areas are defined by a series of linked lists. Each Linked List Item (LLI) controls the transfer of one block of data, and then optionally loads another LLI to continue the DMA operation, or stops the DMA stream. The first LLI is programmed into the DMA Controller.

The data to be transferred described by a LLI (referred to as the packet of data) usually requires one or more DMA bursts (to each of the source and destination).

19.8.5.1 Linked list items

A Linked List Item (LLI) consists of four words. These words are organized in the following order:

1. CSRCADDR
2. CDESTADDR
3. CLLI
4. CCONTROL

Note: The CCONFIG DMA channel Configuration Register is not part of the linked list item.

19.8.5.1.1 Programming the DMA controller for scatter/gather DMA

To program the DMA Controller for scatter/gather DMA:

1. Write the LLIs for the complete DMA transfer to memory. Each linked list item contains four words:
   - Source address.
   - Destination address.
   - Pointer to next LLI.
   - Control word.

   The last LLI has its linked list word pointer set to 0.
2. Choose a free DMA channel with the priority required. DMA channel 0 has the highest priority and DMA channel 7 the lowest priority.
3. Write the first linked list item, previously written to memory, to the relevant channel in the DMA Controller.
4. Write the channel configuration information to the channel Configuration Register and set the Channel Enable bit. The DMA Controller then transfers the first and then subsequent packets of data as each linked list item is loaded.

5. An interrupt can be generated at the end of each LLI depending on the Terminal Count bit in the CCONTROL Register. If this bit is set an interrupt is generated at the end of the relevant LLI. The interrupt request must then be serviced and the relevant bit in the INTTCCLEAR Register must be set to clear the interrupt.

19.8.5.1.2 Example of scatter/gather DMA

See Figure 39 for an example of an LLI. A section of memory is to be transferred to a peripheral. The addresses of each LLI entry are given, in hexadecimal, at the left-hand side of the figure. The right side of the figure shows the memory containing the data to be transferred.

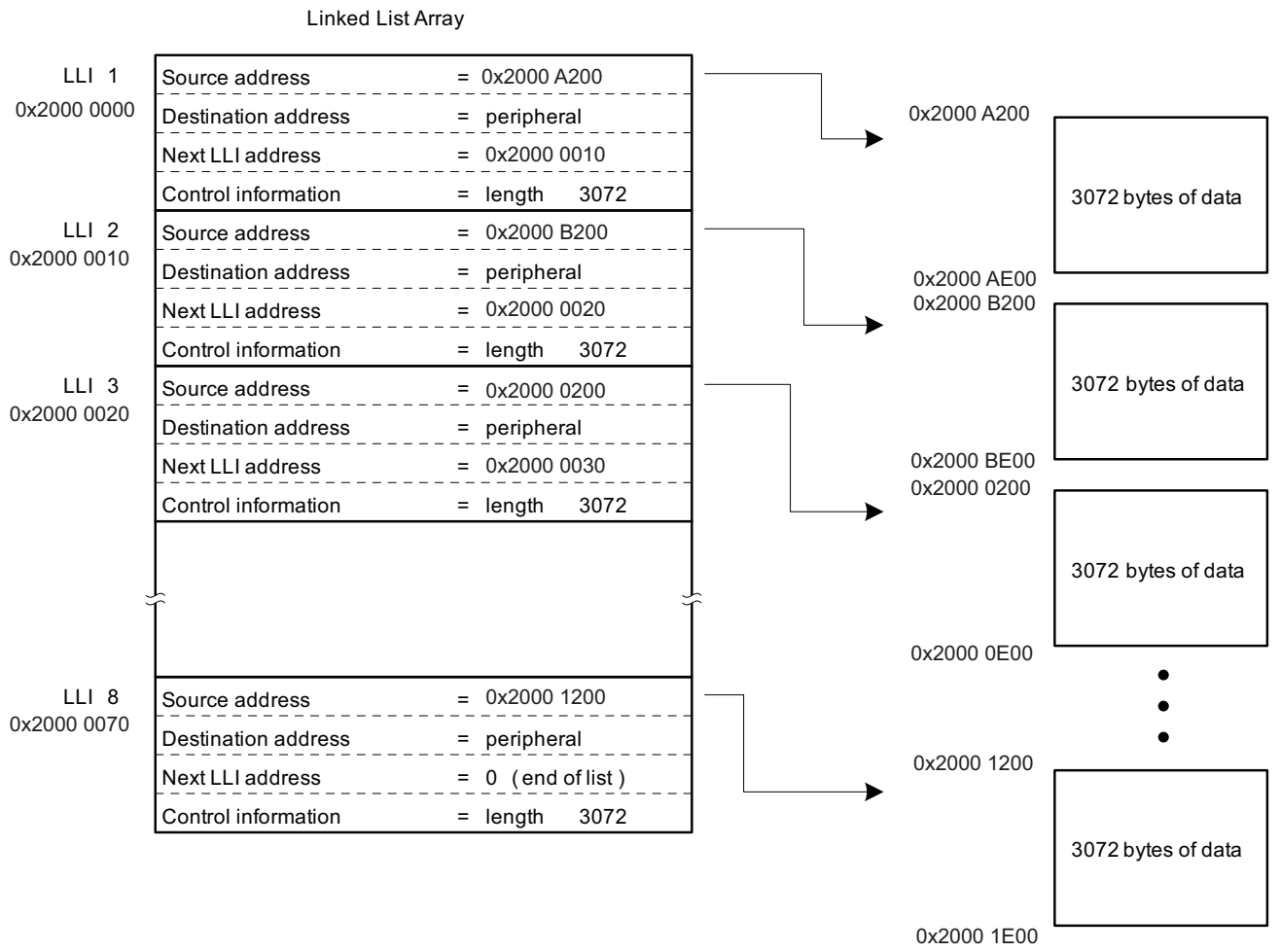


Fig 39. LLI example

The first LLI, stored at 0x2000 0000, defines the first block of data to be transferred, which is the data stored from address 0x2000 A200 to 0x2000 AE00:

- Source start address 0x2000 A200.
- Destination address set to the destination peripheral address.
- Transfer width, word (32-bit).
- Transfer size, 3072 bytes (0xC00).

- Source and destination burst sizes, 16 transfers.
- Next LLI address, 0x2000 0010.

The second LLI, stored at 0x2000 0010, describes the next block of data to be transferred:

- Source start address 0x2000 B200.
- Destination address set to the destination peripheral address.
- Transfer width, word (32-bit).
- Transfer size, 3072 bytes (0xC00).
- Source and destination burst sizes, 16 transfers.
- Next LLI address, 0x2000 0020.

A chain of descriptors is built up, each one pointing to the next in the series. To initialize the DMA stream, the first LLI, 0x2000 0000, is programmed into the DMA Controller. When the first packet of data has been transferred the next LLI is automatically loaded.

The final LLI is stored at 0x2000 0070 and contains:

- Source start address 0x2000 1200.
- Destination address set to the destination peripheral address.
- Transfer width, word (32-bit).
- Transfer size, 3072 bytes (0xC00).
- Source and destination burst sizes, 16 transfers.
- Next LLI address, 0x0.

Because the next LLI address is set to zero, this is the last descriptor, and the DMA channel is disabled after transferring the last item of data. The channel is probably set to generate an interrupt at this point to indicate to the ARM processor that the channel can be reprogrammed.

20.1 How to read this chapter

The SD/MMC card interface is available on all LPC43xx parts.

20.2 Basic configuration

Table 284. SDIO clocking and power control

	Base clock	Branch clock	Operating frequency
SDIO register interface	BASE_M4_CLK	CLK_M4_SDIO	up to 204 MHz
SDIO bit rate clock	BASE_SDIO_CLK	CLK_SDIO	The maximum operating frequency of a SD card is 25 MHz, 26 MHz or 52 MHz for a MMC-Ver4.0 card.

The SDIO is reset by the SD_RST (reset # 20).

20.3 Features

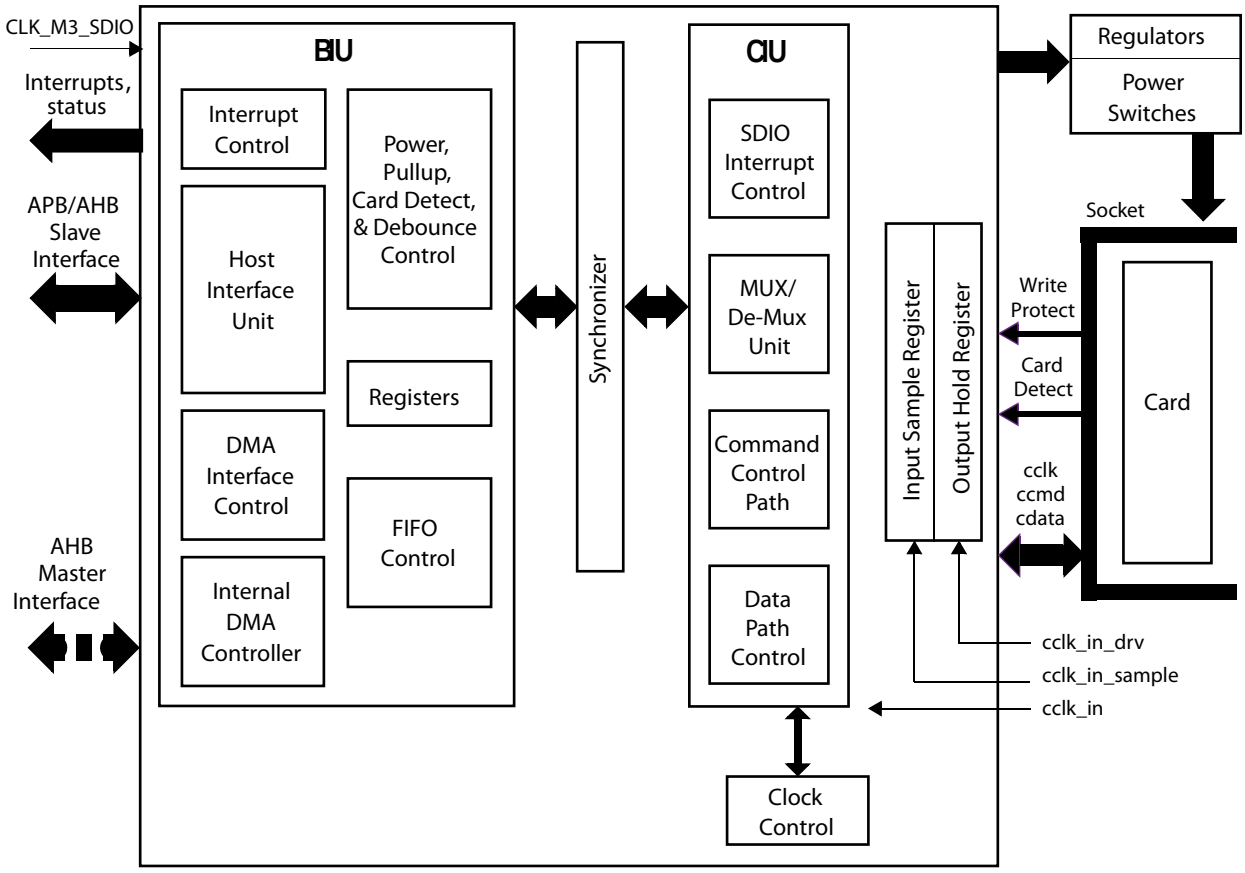
The SD/MMC card interface supports the following modes:

- Secure Digital memory (SD version 3.0)
- Secure Digital I/O (SDIO version 2.0)
- Consumer Electronics Advanced Transport Architecture (CE-ATA version 1.1)
- Multimedia Cards (MMC version 4.4)

20.4 General description

The SD/MMC care interface consists of the following main functional blocks:

- Bus Interface Unit (BIU) - Provides AHB and DMA interfaces for register and data read/writes.
- Card Interface Unit (CIU) - Handles the card protocols and provides clock management.
- SD/MMC DMA - Integrated DMA controller



Note: The card_detect and write-protect signals are from the SD/MMC card socket and not from the SD/MMC card.

Fig 40. SD/MMC block diagram

20.5 Pin description

Table 285. SDIO pin description

Pin function	Direction	Description
SD_CLK	O	SD/SDIO/MMC clock
$\overline{\text{SD\_CD}}$	I	SDIO card detect for single slot
SD_WP	I	SDIO card write protect
SD_LED	O	LED On signal. This signal cautions the user not to remove the SD card while it is accessed.
SD_CMD	I/O	Command input/output
SD_D[7:0]	I/O	Data input/output for data lines DAT[7:0]

Table 285. SDIO pin description

Pin function	Direction	Description
SD_VOLT[2:0]	O	SD/MMC bus voltage select output 2:0.
SD_RST	O	SD/MMC reset signal for MMC4.4 card.
SD_POW	O	<tbd>

<tbd>

20.6 Register description

Table 286. Register overview: SDMMC (base address: 0x4000 4000)

Name	Access	Address offset	Description	Reset value	Reference
CTRL	R/W	0x000	Control Register	0	Table 287
PWREN	R/W	0x004	Power Enable Register	0	Table 288
CLKDIV	R/W	0x008	Clock Divider Register	0	Table 289
CLKSRC	R/W	0x00C	SD Clock Source Register	0	Table 290
CLKENA	R/W	0x010	Clock Enable Register	0	Table 291
TMOUT	R/W	0x014	Time-out Register		Table 292
CTYPE	R/W	0x018	Card Type Register	0	Table 293
BLKSIZ	R/W	0x01C	Block Size Register	0x200	Table 294
BYTCNT	R/W	0x020	Byte Count Register	0x200	Table 295
INTMASK	R/W	0x024	Interrupt Mask Register		Table 296
CMDARG	R/W	0x028	Command Argument Register	0x00000000	Table 297
CMD	R/W	0x02C	Command Register	0x00000000	Table 298
RESP0	R	0x030	Response Register 0	0x00000000	Table 299
RESP1	R	0x034	Response Register 1	0x00000000	Table 300
RESP2	R	0x038	Response Register 2	0	Table 301
RESP3	R	0x03C	Response Register 3	0	Table 302
MINTSTS	r	0x040	Masked Interrupt Status Register	Reset value	Table 303
RINTSTS	R/W	0x044	Raw Interrupt Status Register	0	Table 304
STATUS	R	0x048	Status Register		Table 305
FIFOTH	R/W	0x04C	FIFO Threshold Watermark Register		Table 306
CDETECT	R	0x050	Card Detect Register		Table 307
WRTPRT	R	0x054	Write Protect Register		Table 308
-	-	0x058	Reserved	-	-
TCBCNT	R	0x05C	Transferred CIU Card Byte Count Register	0x00000000	Table 309
TBBCNT	R	0x060	Transferred Host to BIU-FIFO Byte Count Register	0	Table 310
DEBNCE	R/W	0x064	Debounce Count Register		Table 311
-	-	0x068	Reserved	-	-
-	-	0x06C	Reserved	-	-
UHS_REG	R/W	0x074	UHS-1 Register	0x00000000	Table 312
RST_N	R/W	0x078	Hardware Reset		Table 313

Table 286. Register overview: SDMMC (base address: 0x4000 4000)

Name	Access	Address offset	Description	Reset value	Reference
-	-	0x07C	Reserved	-	-
BMOD	R/W	0x080	Bus Mode Register	0x00000000	Table 314
PLDMND	W	0x084	Poll Demand Register	0x00000000	Table 315
DBADDR	R/W	0x088	Descriptor List Base Address Register	0x00000000	Table 316
IDSTS	R/W	0x08C	Internal DMAC Status Register	0x00000000	Table 317
IDINTEN	R/W	0x090	Internal DMAC Interrupt Enable Register	0x00000000	Table 318
DSCADDR	R	0x094	Current Host Descriptor Address Register	0x00000000	Table 319
BUFADDR	R	0x098	Current Buffer Descriptor Address Register	0x00000000	Table 320
DATA	R/W	≥ 0x100	Data FIFO read/write; if address is equal or greater than 0x100, then FIFO is selected as long as device is selected. Address 0x100 and above are mapped to the data FIFO. More than one address is mapped to the data FIFO so that the FIFO can be accessed using bursts.		-

20.6.1 Control Register (CTRL)

Table 287. Control Register (CTRL, address 0x4000 4000) bit description

Bit	Symbol	Value	Description	Reset value
0	CONTROLLER_RESET		Controller reset. To reset controller, firmware should set bit to 1. This bit is auto-cleared after two AHB and two cclk_in clock cycles. This resets: - BIU/CIU interface - CIU and state machines - abort_read_data, send_irq_response, and read_wait bits of Control register - start_cmd bit of Command register Does not affect any registers or DMA interface, or FIFO or host interrupts	0
		0	No change	
		1	Reset DWC_mobile_storage controller	
1	FIFO_RESET		Fifo reset. To reset FIFO, firmware should set bit to 1. This bit is auto-cleared after completion of reset operation. auto-cleared after two AHB clocks.	0
		0	No change	
		1	Reset to data FIFO To reset FIFO pointers	
2	DMA_RESET		dma_reset. To reset DMA interface, firmware should set bit to 1. This bit is auto-cleared after two AHB clocks.	0
		0	No change	
		1	Reset internal DMA interface control logic	
3	-		Reserved	-
4	INT_ENABLE		Global interrupt enable/disable bit. The int port is 1 only when this bit is 1 and one or more unmasked interrupts are set.	0
		0	Disable interrupts	
		1	Enable interrupts	

Table 287. Control Register (CTRL, address 0x4000 4000) bit description

Bit	Symbol	Value	Description	Reset value
5	DMA_ENABLE		DMA enable. Valid only if DWC_mobile_storage configured for External DMA interface. Even when DMA mode is enabled, host can still push/pop data into or from FIFO; this should not happen during the normal operation. If there is simultaneous FIFO access from host/DMA, the data coherency is lost. Also, there is no arbitration inside the SD/MMC interface to prioritize simultaneous host/DMA access.	0
		0	Disable DMA transfer mode	
		1	Enable DMA transfer mode	
6	READ_WAIT		Read/wait. For sending read-wait to SDIO cards.	0
		0	Clear read wait	
		1	Assert read wait	
7	SEND_IRQ_RESPONSE		Send irq response. Bit automatically clears once response is sent. To wait for MMC card interrupts, host issues CMD40, and DWC_mobile_storage waits for interrupt response from MMC card(s). In meantime, if host wants the SD/MMC interface to exit waiting for interrupt state, it can set this bit, at which time the SD/MMC interface command state-machine sends CMD40 response on bus and returns to idle state.	0
		0	No change	
		1	Send auto IRQ response	
8	ABORT_READ_DATA		Abort read data. Used in SDIO card suspend sequence.	0
		0	No change	
		1	After suspend command is issued during read-transfer, software polls card to find when suspend happened. Once suspend occurs, software sets bit to reset data state-machine, which is waiting for next block of data. Bit automatically clears once data state machine resets to idle.	
9	SEND_CCSD		Send ccsd. When set, DWC_mobile_storage sends CCSD to CE-ATA device. Software sets this bit only if current command is expecting CCS (that is, RW_BLK) and interrupts are enabled in CE-ATA device. Once the CCSD pattern is sent to device, the SD/MMC interface automatically clears send_ccsd bit. It also sets Command Done (CD) bit in RINTSTS register and generates interrupt to host if Command Done interrupt is not masked. NOTE: Once send_ccsd bit is set, it takes two card clock cycles to drive the CCSD on the CMD line. Due to this, during the boundary conditions it may happen that CCSD is sent to the CE-ATA device, even if the device signalled CCS.	0
		0	Clear bit if DWC_mobile_storage does not reset the bit.	
		1	Send Command Completion Signal Disable (CCSD) to CE-ATA device	

Table 287. Control Register (CTRL, address 0x4000 4000) bit description

Bit	Symbol	Value	Description	Reset value
10	SEND_AUTO_STOP_CCSD		Send auto stop ccsd. NOTE: Always set send_auto_stop_ccsd and send_ccsd bits together; send_auto_stop_ccsd should not be set independent of send_ccsd. When set, the SD/MMC interface automatically sends internallygenerated STOP command (CMD12) to CE-ATA device. After sending internally-generated STOP command, Auto Command Done (ACD) bit in RINTSTS is set and generates interrupt to host if Auto Command Done interrupt is not masked. After sending the CCSD, the SD/MMC interface automatically clears send_auto_stop_ccsd bit.	0
		0	Clear bit if DWC_mobile_storage does not reset the bit.	
		1	Send internally generated STOP after sending CCSD to CE-ATA device.	
11	CEATA_DEVICE_INTERRUPT _STATUS		CEATA device interrupt status. Software should appropriately write to this bit after power-on reset or any other reset to CE-ATA device. After reset, usually CE-ATA device interrupt is disabled (nIEN = 1). If the host enables CE-ATA device interrupt, then software should set this bit.	0
		0	Interrupts not enabled in CE-ATA device (nIEN = 1 in ATA control register)	
		1	Interrupts are enabled in CE-ATA device (nIEN = 0 in ATA control register)	
15:12	-		Reserved	-
19:16	CARD_VOLTAGE_A		<tbd>	0
23:20	CARD_VOLTAGE_B		<tbd>	0
24	ENABLE_OD_PULLUP		<tbd>	1
		0	Disable	
		1	Enable	
25	USE_INTERNAL_DMAC		SD/MMC DMA use.	0
		0	The host performs data transfers through the slave interface	
		1	Internal DMA used for data transfer	
31:26	-		Reserved	-

20.6.2 Power Enable Register (PWREN)

Table 288. Power Enable Register (PWREN, address 0x4000 4004) bit description

Bit	Symbol	Description	Reset value
0	POWER_ENABLE	Power on/off switch for card; once power is turned on, firmware should wait for regulator/switch ramp-up time before trying to initialize card. 0 - power off 1 - power on <tbd>	0
31:1	-	Reserved	-

20.6.3 Clock Divider Register (CLKDIV)

Table 289. Clock Divider Register (CLKDIV, address 0x4000 4008) bit description

Bit	Symbol	Description	Reset value
7:0	CLK_DIVIDER0	Clock divider-0 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of ff means divide by 2*255 = 510, and so on.	0
15:8	CLK_DIVIDER1	Clock divider-1 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of ff means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported.	0
23:16	CLK_DIVIDER2	Clock divider-2 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of ff means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported.	0
31:24	CLK_DIVIDER3	Clock divider-3 value. Clock division is 2*n. For example, value of 0 means divide by 2*0 = 0 (no division, bypass), a value of 1 means divide by 2*1 = 2, a value of ff means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported. divide by 2*0 = 0 (no division, bypass), value of 1 means divide by 2*1 = 2, value of ff means divide by 2*255 = 510, and so on. In MMC-Ver3.3-only mode, bits not implemented because only one clock divider is supported.	0

20.6.4 SD Clock Source Register (CLKSRC)

Table 290. SD Clock Source Register (CLKSRC, address 0x4000 400C) bit description

Bit	Symbol	Description	Reset value
1:0	CLK_SOURCE	Clock divider source for SD card. 00 - Clock divider 0 01 - Clock divider 1 10 - Clock divider 2 11 - Clock divider 3 In MMC-Ver3.3-only controller, only one clock divider supported. The cclk_out is always from clock divider 0, and this register is not implemented.	0
31:1	-	Reserved	-

20.6.5 Clock Enable Register (CLKENA)

Table 291. Clock Enable Register (CLKENA, address 0x4000 4010) bit description

Bit	Symbol	Description	Reset value
0	CCLK_ENABLE	Low-power control for SD card clock. One MMC card clock supported. 0 - Non-low-power mode 1 - Low-power mode; stop clock when card in IDLE (should be normally set to only MMC and SD memory cards; for SDIO cards, if interrupts must be detected, clock should not be stopped). For MMC-Ver3.3-only mode: 0 - Clock disabled 1 - Clock enabled	0
15:1	-	Reserved	-
16	CCLK_LOW_POWER	Clock-enable control for SD card clock. One MMC card clock supported. 0 - Clock disabled 1 - Clock enabled In MMC-Ver3.3-only mode: 0 - Non-low-power mode 1 - Low-power mode; stop clock when card in IDLE (should be normally set to only MMC and SD memory cards; for SDIO cards, if interrupts must be detected, clock should not be stopped).	0
31:17	-	Reserved	-

20.6.6 Time-out Register (TMOUT)

Table 292. Time-out Register (TMOUT, address 0x4000 4014) bit description

Bit	Symbol	Description	Reset value
7:0	RESPONSE_TIMEOUT	Response time-out value. Value is in number of card output clocks - cclk_out.	0x40
31:8	DATA_TIMEOUT	Value for card Data Read time-out; same value also used for Data Starvation by Host time-out. Value is in number of card output clocks - cclk_out of selected card. Starvation by Host time-out. Value is in number of card output clocks - cclk_out of selected card.	0xFFFFFF

20.6.7 Card Type Register (CTYPE)

Table 293. Card Type Register (CTYPE, address 0x4000 4018) bit description

Bit	Symbol	Description	Reset value
0	CARD_WIDTH0	Indicates if card is 1-bit or 4-bit: 0 - 1-bit mode 1 - 4-bit mode	0
15:1	-	Reserved	-
16	CARD_WIDTH1	Indicates if card is 8-bit: 0 - Non 8-bit mode 1 - 8-bit mode.	0
31:17	-	Reserved	-

20.6.8 Block Size Register (BLKSIZ)

Table 294. Block Size Register (BLKSIZ, address 0x4000 401C) bit description

Bit	Symbol	Description	Reset value
15:0	BLOCK_SIZE	Block size	0x200
31:16	-	Reserved	-

20.6.9 Byte Count Register (BYTCNT)

Table 295. Byte Count Register (BYTCNT, address 0x4000 4020) bit description

Bit	Symbol	Description	Reset value
31:0	BYTE_COUNT	Number of bytes to be transferred; should be integer multiple of Block Size for block transfers. For undefined number of byte transfers, byte count should be set to 0. When byte count is set to 0, it is responsibility of host to explicitly send stop/abort command to terminate data transfer.	0x200

20.6.10 Interrupt Mask Register (INTMASK)

Table 296. Interrupt Mask Register (INTMASK, address 0x4000 4024) bit description

Bit	Symbol	Description	Reset value
0	CDET	Card detect. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
1	RE	Response error. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
2	CDONE	Command done. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
3	DTO	Data transfer over. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
4	TXDR	Transmit FIFO data request. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0

Table 296. Interrupt Mask Register (INTMASK, address 0x4000 4024) bit description

Bit	Symbol	Description	Reset value
5	RXDR	Receive FIFO data request. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
6	RCRC	Response CRC error. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
7	DCRC	Data CRC error. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
8	RTO	Response time-out. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
9	DRTO	Data read time-out. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
10	HTO	Data starvation-by-host time-out (HTO) /Volt_switch_int. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
11	FRUN	FIFO underrun/overrun error. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
12	HLE	Hardware locked write error. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
13	SBE	Start-bit error. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
14	ACD	Auto command done. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
15	EBE	End-bit error (read)/Write no CRC. Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt.	0
16	SDIO_INT_MASK	Mask SDIO interrupt. When masked, SDIO interrupt detection for card is disabled. A 0 masks an interrupt, and 1 enables an interrupt. In MMC-Ver3.3-only mode, these bit is always 0.	0
31:17	-	Reserved	-

20.6.11 Command Argument Register (CMDARG)

Table 297. Command Argument Register (CMDARG, address 0x4000 4028) bit description

Bit	Symbol	Description	Reset value
31:0	CMD_ARG	Value indicates command argument to be passed to card.	0

20.6.12 Command Register (CMD)

Table 298. Command Register (CMD, address 0x4000 402C) bit description

Bit	Symbol	Value	Description	Reset value
5:0	CMD_INDEX		Command index	0
6	RESPONSE_EXPECT		Response expect	0
		0	No response expected from card	
		1	Response expected from card	
7	RESPONSE_ LENGTH		Response length	0
		0	Short response expected from card	
		1	Long response expected from card	
8	CHECK_RESPONSE_CRC		Check response crc. Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller.	0
		0	Do not check response CRC	
		1	Check response CRC	
9	DATA_EXPECTED		Data expected	0
		0	No data transfer expected (read/write)	
		1	Data transfer expected (read/write)	
10	READ_WRITE		read/write. Don't care if no data expected from card.	0
		0	Read from card	
		1	Data transfer expected (read/write)	
11	TRANSFER_MODE		Transfer mode. Don't care if no data expected.	0
		0	Block data transfer command	
		1	Stream data transfer command	
12	SEND_AUTO_STOP		Send auto stop. When set, the SD/MMC interface sends stop command to SD_MMC_CEATA cards at end of data transfer. Refer to Table 321 to determine: - when send_auto_stop bit should be set, since some data transfers do not need explicit stop commands - open-ended transfers that software should explicitly send to stop command Additionally, when resume is sent to resume - suspended memory access of SD-Combo card - bit should be set correctly if suspended data transfer needs send_auto_stop. Don't care if no data expected from card.	0
		0	No stop command sent at end of data transfer	
		1	Send stop command at end of data transfer	
13	WAIT_PRVDATA_COMPLETE		Wait prvdata complete. The wait_prvdata_complete = 0 option typically used to query status of card during data transfer or to stop current data transfer; card_number should be same as in previous command.	0
		0	Send command at once, even if previous data transfer has not completed.	
		1	Wait for previous data transfer completion before sending command.	

Table 298. Command Register (CMD, address 0x4000 402C) bit description

Bit	Symbol	Value	Description	Reset value
14	STOP_ABORT_CMD		Stop abort cmd. When open-ended or predefined data transfer is in progress, and host issues stop or abort command to stop data transfer, bit should be set so that command/data state-machines of CIU can return correctly to idle state. This is also applicable for Boot mode transfers. To Abort boot mode, this bit should be set along with CMD[26] = disable_boot.	0
		0	Neither stop nor abort command to stop current data transfer in progress. If abort is sent to function-number currently selected or not in data-transfer mode, then bit should be set to 0.	
		1	Stop or abort command intended to stop current data transfer in progress.	
15	SEND_INITIALIZATION		Send initialization. After power on, 80 clocks must be sent to card for initialization before sending any commands to card. Bit should be set while sending first command to card so that controller will initialize clocks before sending command to card. This bit should not be set for either of the boot modes (alternate or mandatory).	0
		0	Do not send initialization sequence (80 clocks of 1) before sending this command.	
		1	Send initialization sequence before sending this command.	
20:16	-		Reserved	0
21	UPDATE_CLOC_REGISTERS_ONLY		Update clock registers only. Following register values transferred into card clock domain: CLKDIV, CLRSRC, CLKENA. Changes card clocks (change frequency, truncate off or on, and set low-frequency mode); provided in order to change clock frequency or stop clock without having to send command to cards. During normal command sequence, when update_clock_registers_only = 0, following control registers are transferred from BIU to CIU: CMD, CMDARG, TMOUT, CTYPE, BLKSIZ, BYTCNT. CIU uses new register values for new command sequence to card(s). When bit is set, there are no Command Done interrupts because no command is sent to SD_MMC_CEATA cards. registers_only.	0
		0	Normal command sequence	
		1	Do not send commands, just update clock register value into card clock domain	
22	READ_CEATA_DEVICE		Read ceata device. Software should set this bit to indicate that CE-ATA device is being accessed for read transfer. This bit is used to disable read data time-out indication while performing CE-ATA read transfers. Maximum value of I/O transmission delay can be no less than 10 seconds.The SD/MMC interface should not indicate read data time-out while waiting for data from CE-ATA device.	0
		0	Host is not performing read access (RW_REG or RW_BLK) towards CE-ATA device.	
		1	Host is performing read access (RW_REG or RW_BLK) towards CE-ATA device.	

Table 298. Command Register (CMD, address 0x4000 402C) bit description

Bit	Symbol	Value	Description	Reset value
23	CCS_EXPECTED		CCS expected. If the command expects Command Completion Signal (CCS) from the CE-ATA device, the software should set this control bit. DWC_mobile_storage sets Data Transfer Over (DTO) bit in RINTSTS register and generates interrupt to host if Data Transfer Over interrupt is not masked.	0
		0	Interrupts are not enabled in CE-ATA device (nIEN = 1 in ATA control register), or command does not expect CCS from device.	
		1	Interrupts are enabled in CE-ATA device (nIEN = 0), and RW_BLK command expects command completion signal from CE-ATA device.	
24	ENABLE_BOOT		Enable Boot - this bit should be set only for mandatory boot mode. When Software sets this bit along with start_cmd, CIU starts the boot sequence for the corresponding card by asserting the CMD line low. Do NOT set disable_boot and enable_boot together.	0
25	EXPECT_BOOT_ACK		Expect Boot Acknowledge. When Software sets this bit along with enable_boot, CIU expects a boot acknowledge start pattern of 0-1-0 from the selected card.	0
26	DISABLE_BOOT		Disable Boot. When software sets this bit along with start_cmd, CIU terminates the boot operation. Do NOT set disable_boot and enable_boot together.	0
27	BOOT_MODE		Boot Mode	0
		0	Mandatory Boot operation	
		1	Alternate Boot operation	
28	VOLT_SWITCH		Voltage switch bit	0
		0	No voltage switching	
		1	Voltage switching enabled; must be set for CMD11 only	
30:29	-		Reserved	
31	START_CMD		Start command. Once command is taken by CIU, bit is cleared. When bit is set, host should not attempt to write to any command registers. If write is attempted, hardware lock error is set in raw interrupt register. Once command is sent and response is received from SD_MMC_CEATA cards, Command Done bit is set in raw interrupt register.	

20.6.13 Response Register 0 (RESP0)

Table 299. Response Register 0 (RESP0, address 0x4000 4030) bit description

Bit	Symbol	Description	Reset value
31:0	RESPONSE0	Bit[31:0] of response	0

20.6.14 Response Register 1 (RESP1)

Table 300. Response Register 1 (RESP1, address 0x4000 4034) bit description

Bit	Symbol	Description	Reset value
31:0	RESPONSE1	Register represents bit[63:32] of long response. When CIU sends auto-stop command, then response is saved in register. Response for previous command sent by host is still preserved in Response 0 register. Additional auto-stop issued only for data transfer commands, and response type is always short for them. For information on when CIU sends auto-stop commands, refer to Auto-Stop <tbd>.	0

20.6.15 Response Register 2 (RESP2)

Table 301. Response Register 2 (RESP2, address 0x4000 4038) bit description

Bit	Symbol	Description	Reset value
31:0	RESPONSE2	Bit[95:64] of long response	0

20.6.16 Response Register 3 (RESP3)

Table 302. Response Register 3 (RESP3, address 0x4000 403C) bit description

Bit	Symbol	Description	Reset value
31:0	RESPONSE3	Bit[127:96] of long response	0

20.6.17 Masked Interrupt Status Register (MINTSTS)

Table 303. Masked Interrupt Status Register (MINTSTS, address 0x4000 4040) bit description

Bit	Symbol	Description	Reset value
0	CDET	Card detect. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
1	RE	Response error. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
2	CDONE	Command done. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
3	DTO	Data transfer over. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
4	TXDR	Transmit FIFO data request. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
5	RXDR	Receive FIFO data request. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
6	RCRC	Response CRC error. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
7	DCRC	Data CRC error. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
8	RTO	Response time-out. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0

Table 303. Masked Interrupt Status Register (MINTSTS, address 0x4000 4040) bit description

Bit	Symbol	Description	Reset value
9	DRTO	Data read time-out. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
10	HTO	Data starvation-by-host time-out (HTO). Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
11	FRUN	FIFO underrun/overrun error. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
12	HLE	Hardware locked write error. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
13	SBE	Start-bit error. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
14	ACD	Auto command done. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
15	EBE	End-bit error (read)/write no CRC. Interrupt enabled only if corresponding bit in interrupt mask register is set.	0
16	SDIO_INTERRUPT	Interrupt from SDIO card. SDIO interrupt for card enabled only if corresponding sdio_int_mask bit is set in Interrupt mask register (mask bit 1 enables interrupt; 0 masks interrupt). 0 - No SDIO interrupt from card 1 - SDIO interrupt from card In MMC-Ver3.3-only mode, this bit is always 0.	-
31:17	-	Reserved	-

20.6.18 Raw Interrupt Status Register (RINTSTS)

Table 304. Raw Interrupt Status Register (RINTSTS, address 0x4000 4044) bit description

Bit	Symbol	Description	Reset value
0	CDET	Card detect. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
1	RE	Response error. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
2	CDONE	Command done. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
3	DTO	Data transfer over. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
4	TXDR	Transmit FIFO data request. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
5	RXDR	Receive FIFO data request. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0

Table 304. Raw Interrupt Status Register (RINTSTS, address 0x4000 4044) bit description

Bit	Symbol	Description	Reset value
6	RCRC	Response CRC error. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
7	DCRC	Data CRC error. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
8	RTO_BAR	Response time-out (RTO)/Boot Ack Received (BAR). Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
9	DRTO_BDS	Data read time-out (DRTO)/Boot Data Start (BDS). Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
10	HTO	Data starvation-by-host time-out (HTO). Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status./Volt_switch_int	0
11	FRUN	FIFO underrun/overrun error. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
12	HLE	Hardware locked write error. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
13	SBE	Start-bit error. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
14	ACD	Auto command done. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
15	EBE	End-bit error (read)/write no CRC. Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status.	0
16	SDIO_INTERRUPT	Interrupt from SDIO card. Writes to these bits clear them. Value of 1 clears bit and 0 leaves bit intact. 0 - No SDIO interrupt from card 1 - SDIO interrupt from card In MMC-Ver3.3-only mode, bits always 0. Bits are logged regardless of interrupt-mask status.	0
31:17	-	Reserved.	-

20.6.19 Status Register (STATUS)

Table 305. Status Register (STATUS, address 0x4000 4048) bit description

Bit	Symbol	Description	Reset value
0	FIFO_RX_WATERMARK	FIFO reached Receive watermark level; not qualified with data	0
1	FIFO_TX_WATERMARK	FIFO reached Transmit watermark level; not qualified with data transfer.	1
2	FIFO_EMPTY	FIFO is empty status	1
3	FIFO_FULL	FIFO is full status	0
7:4	CMDFSMSTATES	Command FSM states: 0 - Idle 1 - Send init sequence 2 - Tx cmd start bit 3 - Tx cmd tx bit 4 - Tx cmd index + arg 5 - Tx cmd crc7 6 - Tx cmd end bit 7 - Rx resp start bit 8 - Rx resp IRQ response 9 - Rx resp tx bit 10 - Rx resp cmd idx 11 - Rx resp data 12 - Rx resp crc7 13 - Rx resp end bit 14 - Cmd path wait NCC 15 - Wait; CMD-to-response turnaround NOTE: The command FSM state is represented using 19 bits. The STATUS Register(7:4) has 4 bits to represent the command FSM states. Using these 4 bits, only 16 states can be represented. Thus three states cannot be represented in the STATUS(7:4) register. The three states that are not represented in the STATUS Register(7:4) are: - Bit 16 - Wait for CCS - Bit 17 - Send CCSD - Bit 18 - Boot Mode Due to this, while command FSM is in Wait for CCS state or Send CCSD or Boot Mode?, the Status register indicates status as 0 for the bit field 7:4.	0
8	DATA_3_STATUS	Raw selected card_data[3]; checks whether card is present 0 - card not present 1 - card present	
9	DATA_BUSY	Inverted version of raw selected card_data[0] 0 - card data not busy 1 - card data busy	
10	DATA_STATE_MC_BUSY	Data transmit or receive state-machine is busy	1
16:11	RESPONSE_INDEX	Index of previous response, including any auto-stop sent by core.	0
29:17	FIFO_COUNT	FIFO count - Number of filled locations in FIFO	0
30	DMA_ACK	DMA acknowledge signal state; either dw_dma_ack or ge_dma_ack, depending on DW-DMA or Generic-DMA selection.	0
31	DMA_REQ	DMA request signal state; either dw_dma_req or ge_dma_req, depending on DW-DMA or Generic-DMA selection.	0

20.6.20 FIFO Threshold Watermark Register (FIFOTH)

Table 306. FIFO Threshold Watermark Register (FIFOTH, address 0x4000 404C) bit description

Bit	Symbol	Value	Description	Reset value
11:0	TX_WMARK		FIFO threshold watermark level when transmitting data to card. When FIFO data count is less than or equal to this number, DMA/FIFO request is raised. If Interrupt is enabled, then interrupt occurs. During end of packet, request or interrupt is generated, regardless of threshold programming. In non-DMA mode, when transmit FIFO threshold (TXDR) interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, on last interrupt, host is responsible for filling FIFO with only required remaining bytes (not before FIFO is full or after CIU completes data transfers, because FIFO may not be empty). In DMA mode, at end of packet, if last transfer is less than burst size, DMA controller does single cycles until required bytes are transferred. 12 bits - 1 bit less than FIFO-count of status register, which is 13 bits. Limitation: TX_WMark >= 1; Recommended: FIFO_DEPTH/2; (means less than or equal to FIFO_DEPTH/2).	
15:12	-		Reserved.	
27:16	RX_WMARK		FIFO threshold watermark level when receiving data to card. When FIFO data count reaches greater than this number, DMA/FIFO request is raised. During end of packet, request is generated regardless of threshold programming in order to complete any remaining data. In non-DMA mode, when receiver FIFO threshold (RXDR) interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, interrupt is not generated if threshold programming is larger than any remaining data. It is responsibility of host to read remaining bytes on seeing Data Transfer Done interrupt. In DMA mode, at end of packet, even if remaining bytes are less than threshold, DMA request does single transfers to flush out any remaining bytes before Data Transfer Done interrupt is set. 12 bits - 1 bit less than FIFO-count of status register, which is 13 bits. Limitation: RX_WMark less than FIFO_DEPTH-2 Recommended: (FIFO_DEPTH/2) - 1; (means greater than (FIFO_DEPTH/2) - 1) NOTE: In DMA mode during CCS time-out, the DMA does not generate the request at the end of packet, even if remaining bytes are less than threshold. In this case, there will be some data left in the FIFO. It is the responsibility of the application to reset the FIFO after the CCS time-out.	

Table 306. FIFO Threshold Watermark Register (FIFOTH, address 0x4000 404C) bit description

Bit	Symbol	Value	Description	Reset value
30:28	DW_DMA_MUTIPLE_TRANSACTION_SIZE		Burst size of multiple transaction; should be programmed same as DW-DMA controller multiple-transaction-size SRC/DEST_MSIZE.The units for transfers is the H_DATA_WIDTH parameter. A single transfer (dw_dma_single assertion in case of Non DW DMA interface) would be signalled based on this value. Value should be sub-multiple of (RX_WMark + 1)* (F_DATA_WIDTH/H_DATA_WIDTH) and (FIFO_DEPTH - TX_WMark)* (F_DATA_WIDTH/ H_DATA_WIDTH) For example, if FIFO_DEPTH = 16, FDATA_WIDTH == H_DATA_WIDTH Allowed combinations for MSize and TX_WMark are: MSize = 1, TX_WMARK = 1-15 MSize = 4, TX_WMark = 8 MSize = 4, TX_WMark = 4 MSize = 4, TX_WMark = 12 MSize = 8, TX_WMark = 8 MSize = 8, TX_WMark = 4. Allowed combinations for MSize and RX_WMark are: MSize = 1, RX_WMARK = 0-14 MSize = 4, RX_WMark = 3 MSize = 4, RX_WMark = 7 MSize = 4, RX_WMark = 11 MSize = 8, RX_WMark = 7 MSize = 8, RX_WMark = 11 Recommended: MSize = 8, TX_WMark = 8, RX_WMark = 7	0
		0x0	1 transfer	
		0x1	4 transfers	
		0x2	8 transfers	
		0x3	16 transfers	
		0x4	32 transfers	
		0x5	64 transfers	
		0x6	128 transfers	
		0x7	256 transfers	
31	-		Reserved	

20.6.21 Card Detect Register (CDETECT)

Table 307. Card Detect Register (CDETECT, address 0x4000 4050) bit description

Bit	Symbol	Description	Reset value
0	CARD_DETECT	Card detect. 0 represents presence of card.	0
31:1	-	Reserved	-

20.6.22 Write Protect Register (WRTPRT)

Table 308. Write Protect Register (WRTPRT, address 0x4000 4054) bit description

Bit	Symbol	Description	Reset value
0	WRITE_PROTECT	Write protect. 1 represents write protection.	0
31:1	-	Reserved	-

20.6.23 Transferred CIU Card Byte Count Register (TCBCNT)

Table 309. Transferred CIU Card Byte Count Register (TCBCNT, address 0x4000 405C) bit description

Bit	Symbol	Description	Reset value
31:0	TRANS_CARD_BYTE_COUNT	Number of bytes transferred by CIU unit to card. When AREA_OPTIMIZED parameter is 1, register should be read only after data transfer completes; during data transfer, register returns 0.	0

20.6.24 Transferred Host to BIU-FIFO Byte Count Register (TBBCNT)

Table 310. Transferred Host to BIU-FIFO Byte Count Register (TBBCNT, address 0x4000 4060) bit description

Bit	Symbol	Description	Reset value
31:0	TRANS_FIFO_BYTE_COUNT	Number of bytes transferred between Host/DMA memory and BIU FIFO.	0

20.6.25 Debounce Count Register (DEBNCE)

Table 311. Debounce Count Register (DEBNCE, address 0x4000 4064) bit description

Bit	Symbol	Description	Reset value
23:0	DEBOUNCE_COUNT	Number of host clocks (clk) used by debounce filter logic; typical debounce time is 5-25 ms.	0xFFFFFF
31:24	-	Reserved	

20.6.26 UHS-1 Register (UHS_REG)

Table 312. UHS-1 Register (UHS_REG, address 0x4000 4074) bit description

Bit	Symbol	Description	Reset value
0	VOLT_REG	High Voltage mode. Determines the voltage fed to the buffers by an external voltage regulator. 0 - Buffers supplied with 3.3V Vdd 1 - Buffers supplied with 1.8V Vdd These bits function as the output of the host controller and are fed to an external voltage regulator. The voltage regulator must switch the voltage of the buffers of a particular card to either 3.3V or 1.8V, depending on the value programmed in the register. VOLT_REG[0] should be set to 1 for card number 0 in order to make it operate for 1.8V.	0
15:1	-	Reserved.	-
16	DDR_REG	DDR mode. Determines the voltage fed to the buffers by an external voltage regulator. 0 - Non-DDR mode 1 - DDR mode UHS_REG [16] should be set.	0
31:17	-	Reserved.	-

20.6.27 Hardware Reset (RST_N)

Table 313. Hardware Reset (RST_N, address 0x4000 4078) bit description

Bit	Symbol	Description	Reset value
0	CARD_RESET	Hardware reset. 1 - Active mode 0 - Reset These bits cause the cards to enter pre-idle state, which requires them to be re-initialized.	1
31:1	-	Reserved	

20.6.28 Bus Mode Register (BMOD)

Table 314. Bus Mode Register (BMOD, address 0x4000 4080) bit description

Bit	Symbol	Value	Description	Reset value
0	SWR		Software Reset. When set, the DMA Controller resets all its internal registers. SWR is read/write. It is automatically cleared after 1 clock cycle.	0
1	FB		Fixed Burst. Controls whether the AHB Master interface performs fixed burst transfers or not. When set, the AHB will use only SINGLE, INCR4, INCR8 or INCR16 during start of normal burst transfers. When reset, the AHB will use SINGLE and INCR burst transfer operations. FB is read/write.	0
6:2	DSL		Descriptor Skip Length. Specifies the number of HWord/Word/Dword to skip between two unchained descriptors. This is applicable only for dual buffer structure. DSL is read/write.	0
7	DE		SD/MMC DMA Enable. When set, the SD/MMC DMA is enabled. DE is read/write.	

Table 314. Bus Mode Register (BMOD, address 0x4000 4080) bit description

Bit	Symbol	Value	Description	Reset value
10:8	PBL		Programmable Burst Length. These bits indicate the maximum number of beats to be performed in one SD/MMC DMA transaction. The SD/MMC DMA will always attempt to burst as specified in PBL each time it starts a Burst transfer on the host bus. The permissible values are 1, 4, 8, 16, 32, 64, 128 and 256. This value is the mirror of MSIZE of FIFOTH register. In order to change this value, write the required value to FIFOTH register. This is an encode value as follows.Transfer unit is either 16, 32, or 64 bits, based on HDATA_WIDTH. PBL is a read-only value.	0
		0x0	1 transfer	
		0x1	4 transfers	
		0x2	8 transfers	
		0x3	16 transfers	
		0x4	32 transfers	
		0x5	64 transfers	
		0x6	128 transfers	
		0x7	256 transfers	
31:11	-		Reserved	

20.6.29 Poll Demand Register (PLDMND)

Table 315. Poll Demand Register (PLDMND, address 0x4000 4084) bit description

Bit	Symbol	Description	Reset value
31:0	PD	Poll Demand. If the OWN bit of a descriptor is not set, the FSM goes to the Suspend state. The host needs to write any value into this register for the SD/MMC DMA state machine to resume normal descriptor fetch operation. This is a write only register. PD bit is write-only.	

20.6.30 Descriptor List Base Address Register (DBADDR)

Table 316. Descriptor List Base Address Register (DBADDR, address 0x4000 4088) bit description

Bit	Symbol	Description	Reset value
31:0	SDL	Start of Descriptor List. Contains the base address of the First Descriptor. The LSB bits [0/1/2:0] for 16/32/64-bit bus-width) are ignored and taken as all-zero by the SD/MMC DMA internally. Hence these LSB bits are read-only.	0

20.6.31 Internal DMAC Status Register (IDSTS)

Table 317. Internal DMAC Status Register (IDSTS, address 0x4000 408C) bit description

Bit	Symbol	Description	Reset value
0	TI	Transmit Interrupt. Indicates that data transmission is finished for a descriptor. Writing a 1 clears this bit.	0
1	RI	Receive Interrupt. Indicates the completion of data reception for a descriptor. Writing a 1 clears this bit.	0
2	FBE	Fatal Bus Error Interrupt. Indicates that a Bus Error occurred (IDSTS[12:10]). When this bit is set, the DMA disables all its bus accesses. Writing a 1 clears this bit.	0
3	-	Reserved	
4	DU	Descriptor Unavailable Interrupt. This bit is set when the descriptor is unavailable due to OWN bit = 0 (DES0[31] =0). Writing a 1 clears this bit.	0
5	CES	Card Error Summary. Indicates the status of the transaction to/from the card; also present in RINTSTS. Indicates the logical OR of the following bits: EBE - End Bit Error RTO - Response Time-out/Boot Ack Time-out RCRC - Response CRC SBE - Start Bit Error DRTO - Data Read Time-out/BDS time-out DCRC - Data CRC for Receive RE - Response Error Writing a 1 clears this bit.	0
7:6	-	Reserved	
8	NIS	Normal Interrupt Summary. Logical OR of the following: IDSTS[0] - Transmit Interrupt IDSTS[1] - Receive Interrupt Only unmasked bits affect this bit. This is a sticky bit and must be cleared each time a corresponding bit that causes NIS to be set is cleared. Writing a 1 clears this bit.	0
9	AIS	Abnormal Interrupt Summary. Logical OR of the following: IDSTS[2] - Fatal Bus Interrupt IDSTS[4] - DU bit Interrupt IDSTS[5] - Card Error Summary Interrupt Only unmasked bits affect this bit. This is a sticky bit and must be cleared each time a corresponding bit that causes AIS to be set is cleared. Writing a 1 clears this bit.	0

Table 317. Internal DMAC Status Register (IDSTS, address 0x4000 408C) bit description

Bit	Symbol	Description	Reset value
12:10	EB	Error Bits. Indicates the type of error that caused a Bus Error. Valid only with Fatal Bus Error bit (IDSTS[2]) set. This field does not generate an interrupt. 001 - Host Abort received during transmission 010 - Host Abort received during reception Others: Reserved EB is read-only.	0
16:13	FSM	DMAC state machine present state. 0 - DMA_IDLE 1 - DMA_SUSPEND 2 - DESC_RD 3 - DESC_CHK 4 - DMA_RD_REQ_WAIT 5 - DMA_WR_REQ_WAIT 6 - DMA_RD 7 - DMA_WR 8 - DESC_CLOSE This bit is read-only.	0
31:17	-	Reserved	

20.6.32 Internal DMAC Interrupt Enable Register (IDINTEN)

Table 318. Internal DMAC Interrupt Enable Register (IDINTEN, address 0x4000 4090) bit description

Bit	Symbol	Description	Reset value
0	TI	Transmit Interrupt Enable. When set with Normal Interrupt Summary Enable, Transmit Interrupt is enabled. When reset, Transmit Interrupt is disabled.	0
1	RI	Receive Interrupt Enable. When set with Normal Interrupt Summary Enable, Receive Interrupt is enabled. When reset, Receive Interrupt is disabled.	0
2	FBE	Fatal Bus Error Enable. When set with Abnormal Interrupt Summary Enable, the Fatal Bus Error Interrupt is enabled. When reset, Fatal Bus Error Enable Interrupt is disabled.	0
3	-	Reserved	
4	DU	Descriptor Unavailable Interrupt. When set along with Abnormal Interrupt Summary Enable, the DU interrupt is enabled.	0
5	CES	Card Error summary Interrupt Enable. When set, it enables the Card Interrupt summary.	0
7:6	-	Reserved	

Table 318. Internal DMAC Interrupt Enable Register (IDINTEN, address 0x4000 4090) bit description

Bit	Symbol	Description	Reset value
8	NIS	Normal Interrupt Summary Enable. When set, a normal interrupt is enabled. When reset, a normal interrupt is disabled. This bit enables the following bits: IDINTEN[0] - Transmit Interrupt IDINTEN[1] - Receive Interrupt	0
9	AIS	Abnormal Interrupt Summary Enable. When set, an abnormal interrupt is enabled. This bit enables the following bits: IDINTEN[2] - Fatal Bus Error Interrupt IDINTEN[4] - DU Interrupt IDINTEN[5] - Card Error Summary Interrupt	0
31:10	-	Reserved	

20.6.33 Current Host Descriptor Address Register (DSCADDR)

Table 319. Current Host Descriptor Address Register (DSCADDR, address 0x4000 4094) bit description

Bit	Symbol	Description	Reset value
31:0	HDA	Host Descriptor Address Pointer. Cleared on reset. Pointer updated by IDMAC during operation. This register points to the start address of the current descriptor read by the SD/MMC DMA.	0

20.6.34 Current Buffer Descriptor Address Register (BUFADDR)

Table 320. Current Buffer Descriptor Address Register (BUFADDR, address 0x4000 4098) bit description

Bit	Symbol	Description	Reset value
31:0	HBA	Host Buffer Address Pointer. Cleared on Reset. Pointer updated by IDMAC during operation. This register points to the current Data Buffer Address being accessed by the SD/MMC DMA.	0

20.7 Functional description

20.7.1 Auto-Stop

The auto-stop command helps to send an exact number of data bytes using a stream read or write for the MMC, and a multiple-block read or write for SD memory transfer for SD cards. The module internally generates a stop command and is loaded in the command path when the SEND_AUTO_STOP bit is set in the Command register.

The software should set the SEND_AUTO_STOP bit according to details listed in table below:

Table 321. SEND_AUTO_STOP bit

Card Type	Transfer Type	Byte Count	SEND_AUTO_STOP bit set	Comments
MMC	Stream read	0	No	Open-ended stream
MMC	Stream read	>0	Yes	Auto-stop after all bytes transfer
MMC	Stream read	0	No	Open-ended stream
MMC	Stream read	>0	Yes	Auto-stop after all bytes transfer
MMC	Single-block read	>0	No	Byte count = 0 is illegal
MMC	Single-block write	>0	No	Byte count = 0 is illegal
MMC	Multiple-block read	0	No	Open-ended multiple block
MMC	Multiple-block read	>0	Yes[1]	Pre-defined multiple block
MMC	Multiple-block write	0	No	Open-ended multiple block
MMC	Multiple-block write	>0	Yes[1]	Pre-defined multiple block
SDMEM	Single-block read	>0	No	Byte count = 0 is illegal
SDMEM	Single-block write	>0	No	Byte count = 0 is illegal
SDMEM	Multiple-block read	0	No	Open-ended multiple block
SDMEM	Multiple-block read	>0	Yes	Auto-stop after all bytes transfer
SDMEM	Multiple-block write	0	No	Open-ended multiple block
SDMEM	Multiple-block write	>0	Yes	Auto-stop after all bytes transfer
SDIO	Single-block read	>0	No	Byte count = 0 is illegal
SDIO	Single-block write	>0	No	Byte count = 0 is illegal
SDIO	Multiple-block read	0	No	Open-ended multiple block
SDIO	Multiple-block read	>0	No	Pre-defined multiple block
SDIO	Multiple-block write	0	No	Open-ended multiple block
SDIO	Multiple-block write	>0	No	Pre-defined multiple block

[1] The condition under which the transfer mode is set to block transfer and byte_count is equal to block size is treated as a single-block data transfer command for both MMC and SD cards. If byte_count = n × block_size (n = 2, 3, …), the condition is treated as a predefined multiple-block data transfer command. In the case of an MMC card, the cpu software can perform a predefined data transfer in two ways: 1) Issue the CMD23 command before issuing CMD18/CMD25 commands to the card – in this case, issue CMD18/CMD25 commands without setting the send_auto_stop bit. 2) Issue CMD18/CMD25 commands without issuing CMD23 command to the card, with the send_auto_stop bit set. In this case, the multiple-block data transfer is terminated by an internally-generated auto-stop command after the programmed byte count.

The following list conditions for the auto-stop command.

- Stream read for MMC card with byte count greater than 0 - The Module generates an internal stop command and loads it into the command path so that the end bit of the stop command is sent out when the last byte of data is read from the card and no extra data byte is received. If the byte count is less than 6 (48 bits), a few extra data bytes are received from the card before the end bit of the stop command is sent.
- Stream write for MMC card with byte count greater than 0 - The Module generates an internal stop command and loads it into the command path so that the end bit of the stop command is sent when the last byte of data is transmitted on the card bus and no extra data byte is transmitted. If the byte count is less than 6 (48 bits), the data path transmits the data last in order to meet the above condition.

- Multiple-block read memory for SD card with byte count greater than 0 - If the block size is less than 4 (single-bit data bus), 16 (4-bit data bus), or 32 (8-bit data bus), the auto-stop command is loaded in the command path after all the bytes are read. Otherwise, the top command is loaded in the command path so that the end bit of the stop command is sent after the last data block is received.
- Multiple-block write memory for SD card with byte count greater than 0 - If the block size is less than 3 (single-bit data bus), 12 (4-bit data bus), or 24 (8-bit data bus), the auto-stop command is loaded in the command path after all data blocks are transmitted. Otherwise, the stop command is loaded in the command path so that the end bit of the stop command is sent after the end bit of the CRC status is received.
- Precaution for cpu software during auto-stop - Whenever an auto-stop command is issued, the cpu software should not issue a new command to the Module until the auto-stop is sent by the Module and the data transfer is complete. If the cpu issues a new command during a data transfer with the auto-stop in progress, an auto-stop command may be sent after the new command is sent and its response is received; this can delay sending the stop command, which transfers extra data bytes. For a stream write, extra data bytes are erroneous data that can corrupt the card data. If the cpu wants to terminate the data transfer before the data transfer is complete, it can issue a stop or abort command, in which case the Module does not generate an auto-stop command.

20.7.2 Software/hardware restrictions

Only one data transfer command should be issued at one time. For CE-ATA devices, if CE-ATA device interrupts are enabled (nIEN=0), only one RW_MULTIPLE_BLOCK command (RW_BLK) should be issued; no other commands (including a new RW_BLK) should be issued before the Data Transfer. Over status is set for the outstanding RW_BLK.

Before issuing a new data transfer command, the software should ensure that the card is not busy due to any previous data transfer command. Before changing the card clock frequency, the software must ensure that there are no data or command transfers in progress.

To avoid glitches in the card clock outputs (cclk_out), the software should use the following steps when changing the card clock frequency:

1. Update the Clock Enable register to disable all clocks. To ensure completion of any previous command before this update, send a command to the CIU to update the clock registers by setting:
   - start_cmd bit
   - "update clock registers only" bits
   - "wait_previous data complete" bit

   Wait for the CIU to take the command by polling for 0 on the start_cmd bit.
2. Set the start_cmd bit to update the Clock Divider and/or Clock Source registers, and send a command to the CIU in order to update the clock registers; wait for the CIU to take the command.
3. Set start_cmd to update the Clock Enable register in order to enable the required clocks and send a command to the CIU to update the clock registers; wait for the CIU to take the command.

In non-DMA mode, while reading from a card, the Data Transfer Over (RINTSTS[3]) interrupt occurs as soon as the data transfer from the card is over. There still could be some data left in the FIFO, and the RX_WMark interrupt may or may not occur, depending on the remaining bytes in the FIFO. Software should read any remaining bytes upon seeing the Data Transfer Over (DTO) interrupt. In DMA mode while reading from a card, the DTO interrupt occurs only after all the FIFO data is flushed to memory by the DMA Interface unit.

While writing to a card in DMA mode, if an undefined-length transfer is selected by setting the Byte Count register to 0, the DMA logic will likely request more data than it will send to the card, since it has no way of knowing at which point the software will stop the transfer. The DMA request stops as soon as the DTO is set by the CIU.

If the software issues a controller_reset command by setting control register bit[0] to 1, all the CIU state machines are reset; the FIFO is not cleared. The DMA sends all remaining bytes to the cpu. In addition to a card-reset, if a FIFO reset is also issued, then:

- Any pending DMA transfer on the bus completes correctly
- DMA data read is ignored
- Write data is unknown (x)

Additionally, if dma_reset is also issued, any pending DMA transfer is abruptly terminated. The DMA controller channel should also be reset and reprogrammed.

If any of the previous data commands do not properly terminate, then the software should issue the FIFO reset in order to remove any residual data, if any, in the FIFO. After asserting the FIFO reset, you should wait until this bit is cleared.

One data-transfer requirement between the FIFO and cpu is that the number of transfers should be a multiple of the FIFO data width (F_DATA_WIDTH), which is 32. So if you want to write only 15 bytes to an SD/MMC/CE-ATA card (BYTCNT), the cpu should write 16 bytes to the FIFO or program the DMA to do 16-byte transfers, if DMA mode is enabled. The software can still program the Byte Count register to only 15, at which point only 15 bytes will be transferred to the card. Similarly, when 15 bytes are read from a card, the cpu should still read all 16 bytes from the FIFO.

It is recommended that you do not change the FIFO threshold register in the middle of data transfers.

20.7.3 Programming sequence

20.7.3.1 Initialization

Once the power and clocks are stable, reset_n should be asserted (active-low) for at least two clocks of clk or cclk_in, whichever is slower. The reset initializes the registers, ports, FIFO-pointers, DMA interface controls, and state-machines in the design. After power-on reset, the software should do the following:

1. After power on reset, configure the SD/MMC pins using the SFSP registers in the syscon block (Table 117).

2. Set masks for interrupts by clearing appropriate bits in the Interrupt Mask register @0x024. Set the global int_enable bit of the Control register @0x00. It is recommended that you write 0xffff_ffff to the Raw Interrupt register @0x044 in order to clear any pending interrupts before setting the int_enable bit.
3. Enumerate card stack - Each card is enumerated according to card type; for details, refer to "Enumerated Card Stack". For enumeration, you should restrict the clock frequency to 400 KHz in accordance with SD_MMC/CE-ATA standards.
4. Changing clock. The cards operate at a maximum of 26 MHz (at maximum of 52 MHz in high-speed mode).
5. Set other IP parameters, which normally do not need to be changed with every command, with a typical value such as time-out values in cclk_out according to SD_MMC/CE-ATA specifications.

   ResponseTimeOut = 0x40

   DataTimeOut = highest of one of the following:

   – (10 × ((TAAC × Fop) + (100 × NSAC))
   – Cpu FIFO read/write latency from FIFO empty/full

   FIFO threshold value in bytes in the FIFOTH register @0x04C. Typically, the threshold value can be set to half the FIFO depth (=32); that is:

   – RX_WMark = (FIFO_DEPTH/2) - 1;
   – TX_WMark = FIFO_DEPTH/2
6. If the software decides to handle the interrupts provided by the IP core, you should create another thread to handle interrupts.

20.7.3.2 Enumerated Card Stack

The card stack does the following:

- Enumerates all connected cards
- Sets the RCA for the connected cards
- Reads card-specific information
- Stores card-specific information locally

Enumerate_Card_Stack - Enumerates the card connected on the module. The card can be of the type MMC, CE-ATA, SD, or SDIO. All types of SDIO cards are supported; that is, SDIO_IO_ONLY, SDIO_MEM_ONLY, and SDIO_COMBO cards. The enumeration sequence includes the following steps:

1. Check if the card is connected.
2. Clear the bits in the card_type register. Clear the register bit for a 1-bit, 4-bit, or 8-bit bus width.
3. Identify the card type; that is, SD, MMC, or SDIO.
   – Send CMD5 first. If a response is received, then the card is SDIO
   – If not, send ACMD41; if a response is received, then the card is SD.
   – Otherwise, the card is an MMC or CE-ATA
4. Enumerate the card according to the card type.

Use a clock source with a frequency = Fod (that is, 400 KHz) and use the following enumeration command sequence:

- SD card - Send CMD0, ACMD41, CMD2, CMD3.
- SDHC card - send CMD0, SDCMD8, ACMD41, CMD2, CMD3
- SDIO - Send CMD5; if the function count is valid, CMD3. For the SDIO memory section, follow the same commands as for the SD card.
- MMC - Send CMD0, CMD1, CMD2, CMD3

5. Identify the MMC/CE-ATA device.
   - Selecting ATA mode for a CE-ATA device.
   - Cpu should query the byte 504 (S_CMD_SET) of EXT_CSD register by sending CMD8. If bit 4 is set to 1, then the device supports ATA mode.
   - If ATA mode is supported, the cpu should select the ATA mode by setting the ATA bit (bit 4) of the EXT_CSD register slice 191(CMD_SET) to activate the ATA command set for use. The cpu selects the command set using the SWITCH (CMD6) command.
   - The current mode selected is shown in byte 191 of the EXT_CSD register.

     If the device does not support ATA mode, then the device can be an MMC device or a CE-ATA v1.0 device.
   - Send RW_REG; if a response is received and the response data contains CE-ATA signature, the device is a CE-ATA device.
   - Otherwise the device is an MMC card.
6. You can change the card clock frequency after enumeration.

20.7.3.3 Clock Programming

The clock programming has to be done in the CGU. The cclk_in has to be equal to the cclk_out. Therefore the registers that support this have to be:

- CLKDIV @0x08 = 0x0 (bypass of clock divider).
- CLKSRC @0x0C = 0x0
- CLKENA @0x10 =0x0 or 0x1. This register enables or disables clock for the card and enables low-power mode, which automatically stops the clock to a card when the card is idle for more than 8 clocks.

The Module loads each of these registers only when the start_cmd bit and the Update_clk_regs_only bit in the CMD register are set. When a command is successfully loaded, the Module clears this bit, unless the Module already has another command in the queue, at which point it gives an HLE (Hardware Locked Error); for details on HLEs, refer to "Error Handling".

Software should look for the start_cmd and the Update_clk_regs_only bits, and should also set the wait_prvdata_complete bit to ensure that clock parameters do not change during data transfer. Note that even though start_cmd is set for updating clock registers, the Module does not raise a command_done signal upon command completion.

20.7.3.4 No-Data Command With or Without Response Sequence

To send any non-data command, the software needs to program the CMD register @0x2C and the CMDARG register @0x28 with appropriate parameters. Using these two registers, the Module forms the command and sends it to the command bus. The Module reflects the errors in the command response through the error bits of the RINTSTS register.

When a response is received - either erroneous or valid - the Module sets the command_done bit in the RINTSTS register. A short response is copied in Response Register0, while a long response is copied to all four response registers @0x30, 0x34, 0x38, and 0x3C. The Response3 register bit 31 represents the MSB, and the Response0 register bit 0 represents the LSB of a long response.

For basic commands or non-data commands, follow these steps:

1. Program the Command register @0x28 with the appropriate command argument parameter.
2. Program the Command register @0x2C with the settings in Table 322.
3. Wait for command acceptance by cpu. The following happens when the command is loaded into the Module:
   - Module accepts the command for execution and clears the start_cmd bit in the CMD register, unless one command is in process, at which point the Module can load and keep the second command in the buffer.
   - If the Module is unable to load the command - that is, a command is already in progress, a second command is in the buffer, and a third command is attempted - then it generates an HLE (hardware-locked error).
   - Check if there is an HLE.
   - Wait for command execution to complete. After receiving either a response from a card or response time-out, the Module sets the command_done bit in the RINTSTS register. Software can either poll for this bit or respond to a generated interrupt.
   - Check if response_timeout error, response_CRC error, or response error is set. This can be done either by responding to an interrupt raised by these errors or by polling bits 1, 6, and 8 from the RINTSTS register @0x44. If no response error is received, then the response is valid. If required, the software can copy the response from the response registers @0x30-0x3C.

Software should not modify clock parameters while a command is being executed.

Table 322. CMD register settings for No-Data Command

Name	Value	Comment
start_cmd	1	
update_clock_ registers_only	0	No clock parameters update command
card_number	0	Card number in use. Only zero is possible because one card is support.
Data_expected	0	No data command.
Send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	Can be 1 for commands to stop data transfer, such as CMD12

Table 322. CMD register settings for No-Data Command

Name	Value	Comment
Cmd_index	Command index	
Response_length	0	Can be 1 for R2 (long) response
Response_expect	1	Can be 0 for commands with no response; for example, CMD0, CMD4, CMD15, and so on
User-selectable		
Wait_prvdata_complete	1	Before sending command on command line, cpu should wait for completion of any data command in process, if any (recommended to always set this bit, unless the current command is to query status or stop data transfer when transfer is in progress)
Check_response_crc	1	0 – Do not check response CRC 1 – Check response CRC Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller.

20.7.3.5 Data Transfer Commands

Data transfer commands transfer data between the memory card and the Module. To send a data command, the Module needs a command argument, total data size, and block size. Software can receive or send data through the FIFO.

Before a data transfer command, software should confirm that the card is not busy and is in a transfer state, which can be done using the CMD13 and CMD7 commands, respectively.

For the data transfer commands, it is important that the same bus width that is programmed in the card should be set in the card type register @0x18. Therefore, in order to change the bus width, you should always use the following supplied APIs as appropriate for the type of card:

- Set_SD_Mode() - SD/SDIO card
- Set_HSmodeSettings() - HSMMC card

The Module generates an interrupt for different conditions during data transfer, which are reflected in the RINTSTS register @0x44 as:

1. Data_Transfer_Over (bit 3) - When data transfer is over or terminated. If there is a response time-out error, then the Module does not attempt any data transfer and the "Data Transfer Over" bit is never set.
2. Transmit_FIFO_Data_request (bit 4) - FIFO threshold for transmitting data was reached; software is expected to write data, if available, in FIFO.
3. Receive_FIFO_Data_request (bit 5) - FIFO threshold for receiving data was reached; software is expected to read data from FIFO.

4. Data starvation by Cpu time-out (bit 10) - FIFO is empty during transmission or is full during reception. Unless software writes data for empty condition or reads data for full condition, the Module cannot continue with data transfer. The clock to the card has been stopped.
5. Data read time-out error (bit 9) - Card has not sent data within the time-out period.
6. Data CRC error (bit 7) - CRC error occurred during data reception.
7. Start bit error (bit 13) - Start bit was not received during data reception.
8. End bit error (bit 15) - End bit was not received during data reception or for a write operation; a CRC error is indicated by the card.

   Conditions 6, 7, and 8 indicate that the received data may have errors. If there was a response time-out, then no data transfer occurred.

20.7.3.6 Single-Block or Multiple-Block Read

Steps involved in a single-block or multiple-block read are:

1. Write the data size in bytes in the BYTCNT register @0x20.
2. Write the block size in bytes in the BLKSIZ register @0x1C. The Module expects data from the card in blocks of size BLKSIZ each.
3. Program the CMDARG register @0x28 with the data address of the beginning of a data read. Program the Command register with the parameters listed in Table 323. For SD and MMC cards, use CMD17 for a single-block read and CMD18 for a multiple-block read. For SDIO cards, use CMD53 for both single-block and multiple-block transfers.

   After writing to the CMD register, the Module starts executing the command; when the command is sent to the bus, the command_done interrupt is generated.
4. Software should look for data error interrupts; that is, bits 7, 9, 13, and 15 of the RINTSTS register. If required, software can terminate the data transfer by sending a STOP command.
5. Software should look for Receive_FIFO_Data_request and/or data starvation by cpu time-out conditions. In both cases, the software should read data from the FIFO and make space in the FIFO for receiving more data.
6. When a Data_Transfer_Over interrupt is received, the software should read the remaining data from the FIFO.

Table 323. CMD register settings for Single-block or Multiple-block Read

Name	Value	Comment
start_cmd	1	
update_clock_ registers_only	0	No clock parameters update command
card_number	0	Card number in use. Only zero is possible because one card is support.
Data_expected	1	
Send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	Can be 1 for commands to stop data transfer, such as CMD12
Send_auto_stop	0/1	Set according to <tbd>
Transfer_mode	0	Block transfer

Table 323. CMD register settings for Single-block or Multiple-block Read

Name	Value	Comment
Read_write	0	Read from card
Cmd_index	Command index	
Response_length	0	Can be 1 for R2 (long) response
Response_expect	1	Can be 0 for commands with no response; for example, CMD0, CMD4, CMD15, and so on
User-selectable		
Wait_prvdata_complete	1	Before sending command on command line, cpu should wait for completion of any data command in process, if any (recommended to always set this bit, unless the current command is to query status or stop data transfer when transfer is in progress)
Check_response_crc	1	0 – Do not check response CRC 1 – Check response CRC Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller.

20.7.3.7 Single-Block or Multiple-Block Write

Steps involved in a single-block or multiple-block write are:

1. Write the data size in bytes in the BYTCNT register @0x20.
2. Write the block size in bytes in the BLKSIZ register @0x1C; the Module sends data in blocks of size BLKSIZ each.
3. Program CMDARG register @0x28 with the data address to which data should be written.
4. Write data in the FIFO; it is usually best to start filling data the full depth of the FIFO.
5. Program the Command register with the parameters listed in Table 324. For SD and MMC cards, use CMD24 for a single-block write and CMD25 for a multiple-block write. For SDIO cards, use CMD53 for both single-block and multiple-block transfers.

   After writing to the CMD register, Module starts executing a command; when the command is sent to the bus, a command_done interrupt is generated.
6. Software should look for data error interrupts; that is, for bits 7, 9, and 15 of the RINTSTS register. If required, software can terminate the data transfer by sending the STOP command.
7. Software should look for Transmit_FIFO_Data_request and/or time-out conditions from data starvation by the cpu. In both cases, the software should write data into the FIFO.

8. When a Data_Transfer_Over interrupt is received, the data command is over. For an open-ended block transfer, if the byte count is 0, the software must send the STOP command. If the byte count is not 0, then upon completion of a transfer of a given number of bytes, the Module should send the STOP command, if necessary. Completion of the AUTO-STOP command is reflected by the Auto_command_done interrupt - bit 14 of the RINTSTS register. A response to AUTO_STOP is stored in RESP1 @0x34.

Table 324. CMD register settings for Single-block or Multiple-block write

Name	Value	Comments
start_cmd	1	
update_clock_ registers_only	0	No clock parameters update command
card_number	0	Card number in use. Only zero is possible because one card is support.
Data_expected	1	
Send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	Can be 1 for commands to stop data transfer, such as CMD12
Send_auto_stop	0/1	Set according to XXXXXXXXX
Transfer_mode	0	Block transfer
Read_write	1	Write to card
Cmd_index	Command index	
Response_length	0	Can be 1 for R2 (long) response
Response_expect	1	Can be 0 for commands with no response; for example, CMD0, CMD4, CMD15, and so on
User-selectable		
Wait_prvdata_complete	1	Before sending command on command line, cpu should wait for completion of any data command in process, if any (recommended to always set this bit, unless the current command is to query status or stop data transfer when transfer is in progress)
Check_response_crc	1	0 – Do not check response CRC 1 – Check response CRC Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller.

20.7.3.8 Stream Read

A stream read is like the block read mentioned in "Single-Block or Multiple-Block Read", except for the following bits in the Command register:

transfer_mode = 1; //Stream transfer

cmd_index = CMD20;

A stream transfer is allowed for only a single-bit bus width.

20.7.3.9 Stream Write

A stream write is exactly like the block write mentioned in "Single-Block or Multiple-Block Write", except for the following bits in the Command register:

- transfer_mode = 1;//Stream transfer
- cmd_index = CMD11;

In a stream transfer, if the byte count is 0, then the software must send the STOP command. If the byte count is not 0, then when a given number of bytes completes a transfer, the Module sends the STOP command. Completion of this AUTO_STOP command is reflected by the Auto_command_done interrupt. A response to an AUTO_STOP is stored in the RESP1 register @0x34.

A stream transfer is allowed for only a single-bit bus width.

20.7.3.10 Sending Stop or Abort in Middle of Transfer

The STOP command can terminate a data transfer between a memory card and the Module, while the ABORT command can terminate an I/O data transfer for only the SDIO_IOONLY and SDIO_COMBO cards.

- Send STOP command - Can be sent on the command line while a data transfer is in progress; this command can be sent at any time during a data transfer. For information on sending this command, refer to "No-Data Command With or Without Response Sequence".

  You can also use an additional setting for this command in order to set the Command register bits (5-0) to CMD12 and set bit 14 (stop_abort_cmd) to 1. If stop stop_abort_cmd is not set to 1, the user stopped a data transfer. Reset bit 13 of the Command register (wait_prvdata_complete) to 0 in order to make the Module send the command at once, even though there is a data transfer in progress.

  – Send ABORT command - Can be used with only an SDIO_IOONLY or SDIO_COMBO card. To abort the function that is transferring data, program the function number in ASx bits (CCCR register of card, address 0x06, bits (0-2) using CMD52.

    This is a non-data command. For information on sending this command, refer to "No-Data Command With or Without Response Sequence".

    Program the CMDARG register @0x28 with the appropriate command argument parameters listed in Table 325.

  – Program the Command register using the command index as CMD52. Similar to the STOPcommand, set bit 14 of the Command register (stop_abort_cmd) to 1, which must be done in order to inform the Module that the user aborted the data transfer. Reset bit 13 (wait_prvdata_complete) of the Command register to 0 in order to make the Module send the command at once, even though a data transfer is in progress.

  – Wait for command_transfer_over.

- Check response (R5) for errors.

- During an open-ended card write operation, if the card clock is stopped because the FIFO is empty, the software must first fill the data into the FIFO and start the card clock before issuing a stop/abort command to the card.

Table 325. Parameters for CMDARG register

Bits	Contents	Value
31	R/W flag	1
30-28	Function number	0, for CCCR access
27	RAW flag	1, if needed to read after write
26	Don't care	-
25-9	Register address	0x06
8	Don't care	-
7-0	Write data	Function number to be aborted

20.7.4 Suspend or Resume Sequence

In an SDIO card, the data transfer between an I/O function and the Module can be temporarily halted using the SUSPEND command; this may be required in order to perform a high-priority data transfer with another function. When desired, the data transfer can be resumed using the RESUME command.

The following functions can be implemented by programming the appropriate bits in the CCCR register (Function 0) of the SDIO card. To read from or write to the CCCR register, use the CMD52 command.

1. SUSPEND data transfer - Non-data command.
   - Check if the SDIO card supports the SUSPEND/RESUME protocol; this can be done through the SBS bit in the CCCR register @0x08 of the card.

     Check if the data transfer for the required function number is in process; the function number that is currently active is reflected in bits 0-3 of the CCCR register @0x0D. Note that if the BS bit (address 0xc::bit 0) is 1, then only the function number given by the FSx bits is valid.

     To suspend the transfer, set BR (bit 2) of the CCCR register @0x0C.

     Poll for clear status of bits BR (bit 1) and BS (bit 0) of the CCCR @0x0C. The BS (Bus Status) bit is 1 when the currently-selected function is using the data bus; the BR (Bus Release) bit remains 1 until the bus release is complete. When the BR and BS bits are 0, the data transfer from the selected function has been suspended.

     During a read-data transfer, the Module can be waiting for the data from the card. If the data transfer is a read from a card, then the Module must be informed after the successful completion of the SUSPEND command. The Module then resets the data state machine and comes out of the wait state. To accomplish this, set abort_read_data (bit 8) in the Control register.

     Wait for data completion. Get pending bytes to transfer by reading the TCBCNT register @0x5C.
2. RESUME data transfer - This is a data command.

– Check that the card is not in a transfer state, which confirms that the bus is free for data transfer.

  If the card is in a disconnect state, select it using CMD7. The card status can be retrieved in response to CMD52/CMD53 commands.

  Check that a function to be resumed is ready for data transfer; this can be confirmed by reading the RFx flag in CCCR @0x0F. If RF = 1, then the function is ready for data transfer.

  To resume transfer, use CMD52 to write the function number at FSx bits (0-3) in the CCCR register @0x0D. Form the command argument for CMD52 and write it in CMDARG @0x28; bit values are listed in Table 326.

– Write the block size in the BLKSIZ register @0x1C; data will be transferred in units of this block size.

  Write the byte count in the BYTCNT register @0x20. This is the total size of the data; that is, the remaining bytes to be transferred. It is the responsibility of the software to handle the data.

  Program Command register; similar to a block transfer. For details, refer to "Single-Block or Multiple-Block Read" and "Single-Block or Multiple-Block Write".

  When the Command register is programmed, the command is sent and the function resumes data transfer. Read the DF flag (Resume Data Flag). If it is 1, then the function has data for the transfer and will begin a data transfer as soon as the function or memory is resumed. If it is 0, then the function has no data for the transfer.

  If the DF flag is 0, then in case of a read, the Module waits for data. After the data time-out period, it gives a data time-out error.

Table 326. Parameters for CMDARG register

Bits	Contents	Value
31	R/W flag	1
30-28	Function number	0, for CCCR access
27	RAW flag	1, read after write
26	Don't care	-
25-9	Register address	0x0D
8	Don't care	-
7-0	Write data	Function number to be aborted

20.7.4.1 Read_Wait Sequence

Read_wait is used with only the SDIO card and can temporarily stall the data transfer-either from function or memory-and allow the cpu to send commands to any function within the SDIO device. The cpu can stall this transfer for as long as required. The Module provides the facility to signal this stall transfer to the card. The steps for doing this are:

1. Check if the card supports the read_wait facility; read SRW (bit 2) of the CCCR register @0x08. If this bit is 1, then all functions in the card support the read_wait facility. Use CMD52 to read this bit.
2. If the card supports the read_wait signal, then assert it by setting the read_wait (bit 6) in the CTRL register @0x00.

3. Clear the read_wait bit in the CTRL register.

20.7.4.2 CE-ATA Data Transfer Commands

This section describes the CE-ATA data transfer commands. For information on the basic settings and interrupts generated for different conditions, refer to "Data Transfer Commands".

20.7.4.2.1 Reset and Device Recovery

Before starting CE-ATA operations, the cpu should perform an MMC reset and initialization procedure. The cpu and device should negotiate the MMC TRAN state (defined by the MultiMedia Card System Specification) before the device enters the MMC TRAN state. The cpu should follow the existing MMC Card enumeration procedure in order to negotiate the MMC

TRAN state. After completing normal MMC reset and initialization procedures, the cpu should query the initial ATA Task File values using RW_REG/CMD39.

By default, the MMC block size is 512 bytes-indicated by bits 1:0 of the srcControl register inside the CE-ATA device. The cpu can negotiate the use of a 1KB or 4KB MMC block size. The device indicates MMC block sizes that it can support through the srcCapabilities register; the cpu reads this register in order to negotiate the MMC block size. Negotiation is complete when the cpu controller writes the MMC block size into the srcControl register bits 1:0 of the device.

20.7.4.2.2 ATA Task File Transfer

ATA task file registers are mapped to addresses 0x00h-0x10h in the MMC register space. RW_REG is used to issue the ATA command, and the ATA task file is transmitted in a single RW_REG MMC command sequence.

The cpu software stack should write the task file image to the FIFO before setting the CMDARG and CMD registers. The cpu processor then sets the address and byte count in the CMDARG-offset 0x28 in the BIU register space-before setting the CMD (offset 0x2C) register bits.

For RW_REG, there is no command completion signal from the CE-ATA device

ATA Task File Transfer Using RW_MULTIPLE_REGISTER (RW_REG)

This command involves data transfer between the CE-ATA device and the Module. To send a data command, the Module needs a command argument, total data size, and block size. Software can receive or send data through the FIFO.

Steps involved in an ATA Task file transfer (read or write) are:

1. Write the data size in bytes in the BYTCNT register @0x20.
2. Write the block size in bytes in the BLKSIZ register @0x1C; the Module expects a single block transfer.
3. Program the CMDARG register @0x28 with the beginning register address.

You should program the CMDARG, CMD, BLKSIZ, and BYTCNT registers according to the following tables.

- Program the Command Argument (CMDARG) register as shown below.

Table 327. Parameters for CMDARG register

Bits	Contents	Value
31	R/W flag	1 (write) or 0 (read)
30-24	Reserved	0
23:18	Starting register address for read/write; Dword aligned	0
17:16	Register address; Dword aligned	0
15-8	Reserved; bits cleared to 0 by CPU	0
7:2	Number of bytes to read/write; integral number of Dwords	16
1:0	Byte count in integral number of Dwords	0

- Program the Command (CMD) register as shown below.

Table 328. CMD register settings

Name	Value	Comment
start_cmd	1	
Css_expect	0	Command Completion Signal is not expected
Read_ceata_device	0/1	1 – If RW_BLK or RW_REG read
update_clock_ registers_only	0	No clock parameters update command
card_number	0	Card number in use. Only zero is possible because one card is support.
Data_expected	1	
Send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	
Send_auto_stop	0	
Transfer_mode	0	Block transfer
Read_write	0/1	0 read from card, 1 - Write to card
Cmd_index	Command index	
Response_length	0	
Response_expect	1	

Table 328. CMD register settings

Name	Value	Comment
User-selectable		
Wait_prvdata_complete	1	0 – Sends command immediately 1 – Sends command after previous data transfer over
Check_response_crc	1	0 – Do not check response CRC 1 – Check response CRC

- Program the block size (BLKSIZ) register as shown below.

Table 329. BLKSIZ register

Bits	Value	Comment
31:16	0	Reserved bits as zeroes (0)
15:0	16	For accessing entire task file (16, 8-bit registers); block size of 16 bytes

- Program the Byte Count (BYTCNT) register as shown below.

Table 330. BYTCNT register

Bits	Value	Comment
31:0	16	For accessing entire task file(16, 8 bit registers); byte count value of 16 is used with the block size set to 16

20.7.4.2.3 ATA Payload Transfer Using RW_MULTIPLE_BLOCK (RW_BLK)

This command involves data transfer between the CE-ATA device and the Module. To send a data command, the Module needs a command argument, total data size, and block size. Software can receive or send data through the FIFO.

Steps involved in an ATA payload transfer (read or write) are:

1. Write the data size in bytes in the BYTCNT register @0x20.
2. Write the block size in bytes in the BLKSIZ register @0x1C. The Module expects a single/multiple block transfer.
3. Program the CMDARG register @0x28 to indicate the Data Unit Count.

   You should program the CMDARG, CMD, BLKSIZ, and BYTCNT registers according to the following tables.

   – Program the Command Argument (CMDARG) register as shown below.

Table 331. Parameters for CMDARG register

Bits	Contents	Value
31	R/W flag	1 (write) or 0 (read)
30-24	Reserved	0
23:16	Reserved	0
15:8	Data Count Unit [15:8]	Data count
1:0	Data Count Unit [7:0]	Data count

- Program the Command (CMD) register as shown below.

Table 332. CMD register settings

Name	Value	Comment
start_cmd	1	-
Css_expect	1	Command Completion Signal is expected; set for RW_BLK if interrupts are enabled in CE-ATA device, nIEN = 0
Read_ceata_device	0/1	1 – If RW_BLK or RW_REG read
update_clock_ registers_only	0	No clock parameters update command
card_number	0	Card number in use. Only zero is possible because one card is support.
Data_expected	1	
Send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	
Send_auto_stop	0	
Transfer_mode	0	Block transfer
Read_write	0/1	0 read from card, 1 - Write to card
Cmd_index	Command index	
Response_length	0	
Response_expect	1	
User-selectable		
Wait_prvdata_complete	1	0 – Sends command immediately 1 – Sends command after previous data transfer over
Check_response_crc	1	0 – Do not check response CRC 1 – Check response CRC

- Program the block size (BLKSIZ) register as shown below.

Table 333. BLKSIZ register

Bits	Value	Comment
31:16	0	Reserved bits as zeroes (0)
15:0	512, 1024, 4096	MMC block size can be 512, 1024, or 4096 bytes as negotiated by CPU

- Program the Byte Count (BYTCNT) register as shown below.

Table 334. BYTCNT register

Bits	Value	Comment
31:0	N × block_size	byte_count should be integral multiple of block size; for ATA media access commands, byte count should be multiple of 4KB. (N × block_size = X × 4KB, where N and X are integers)

20.7.4.2.4 Sending Command Completion Signal Disable

While waiting for the Command Completion Signal (CCS) for an outstanding RW_BLK, the cpu can send a Command Completion Signal Disable (CCSD).

- Send CCSD - Module sends CCSD to the CE-ATA device if the send_ccsd bit is set in the CTRL register; this bit is set only after a response is received for the RW_BLK.
- Send internal Stop command - Send internally generated STOP (CMD12) command after sending the CCSD pattern. If send_auto_stop_ccsd bit is also set when the controller is programmed to send the CCSD pattern, the Module sends the internally generated STOP command on the CMD line. After sending the STOP command, the Module sets the Auto Command Done bit in the RINTSTS register.

20.7.4.2.5 Recovery after Command Completion Signal Time-out

If time-out happened while waiting for Command Completion Signal (CCS), the cpu needs to send Command Completion Signal Disable (CCSD) followed by a STOP command to abort the pending ATA command. The cpu can program the Module to send internally generated STOP command after sending the CCSD pattern

- Send CCSD - Set the send_ccsd bit in the CTRL register.
- Reset bit 13 of the Command register (wait_prvdata_complete) to 0 in order to make the Module send the command at once, even though there is a data transfer in progress.
- Send internal STOP command - Set send_auto_stop_ccsd bit in the CTRL register, which programs the cpu controller to send the internally generated STOP command. After sending the STOP command, the Module sets the Auto Command Done bit in the RINTSTS register.

20.7.4.2.6 Reduced ATA Command Set

It is necessary for the CE-ATA device to support the reduced ATA command subset. The following details discuss this reduced command set.

- IDENTIFY DEVICE - Returns 512-byte data structure to the cpu that describes device-specific information and capabilities. The cpu issues the IDENTIFY DEVICE command only if the MMC block size is set to 512 bytes; any other MMC block size has indeterminate results.

The cpu issues RW_REG for the ATA command, and the data is retrieved through RW_BLK.

The cpu controller uses the following settings while sending RW_REG for the IDENTIFY DEVICE ATA command. The following lists the primary bit

- CMD register setting - data_expected field set to 0
- CMDARG register settings:
  - Bit [31] set to 0
  - Bits [7:2] set to 128.
- Task file settings:
  - Command field of the ATA task file set to ECh
  - Reserved fields of the task file cleared to 0

- BLKSIZ register bits [15:0] and BYTCNT register - Set to 16

  The cpu controller uses the following settings for data retrieval (RW_BLK):\

- CMD register settings:
  - ccs_expect set to 1
  - data_expected set to 1
- CMDARG register settings:
  - Bit [31] set to 0 (Read operation)

    Data Count set to 1 (16'h0001)
- BLKSIZ register bits [15:0] and BYTCNT register - Set to 512 IDENTIFY DEVICE can be aborted as a result of the CPU issued CMD12.
  - READ DMA EXT - Reads a number of logical blocks of data from the device using the Data-In data transfer protocol. The cpu uses RW_REG to issue the ATA command and RW_BLK for the data transfer.
  - WRITE DMA EXT - Writes a number of logical blocks of data to the device using the Data-Out data transfer protocol. The cpu uses RW_REG to issue the ATA command and RW_BLK for the data transfer.
  - STANDBY IMMEDIATE - No data transfer (RW_BLK) is expected for this ATA command, which causes the device to immediately enter the most aggressive power management mode that still retains internal device context.
- CMD Register setting - data_expected field set to 0

  CMDARG register settings:
  - Bit [31] set to 1
  - Bits [7:2] set to 4
- Task file settings:
  - Command field of the ATA task file set to E0h
  - Reserved fields of the task file cleared to 0
- BLKSIZ register bits [15:0] and BYTCNT register - Set to 16
  - FLUSH CACHE EXT - No data transfer (RW_BLK) is expected for this ATA command. For devices that buffer/cache written data, the FLUSH CACHE EXT command ensures that buffered data is written to the device media. For devices that do not buffer written data, FLUSH CACHE EXT returns a success status. The cpu issues RW_REG for the ATA command, and the status is retrieved through CMD39/RW_REG; there can be error status for this ATA command, in which case fields other than the status field of the ATA task file are valid.
- The CPU uses the following settings while sending the RW_REG for STANDBY IMMEDIATE ATA command:
- CMD register setting - data_expected field set to 0
- CMDARG register settings:
- Bit [31] set to 1
- Bits [7:2] set to 4
- Task file settings:

- Command field of the ATA task file set to EAh
- Reserved fields of the task file cleared to 0
- BLKSIZ register bits [15:0] and BYTCNT register - Set to 16

20.7.4.3 Controller/DMA/FIFO Reset Usage

Communication with the card involves the following:

- Controller - Controls all functions of the Module.
- FIFO - Holds data to be sent or received.
- DMA - If DMA transfer mode is enabled, then transfers data between system memory and the FIFO.
- Controller reset - Resets the controller by setting the controller_reset bit (bit 0) in the CTRL register; this resets the CIU and state machines, and also resets the BIU-to-CIU interface. Since this reset bit is self-clearing, after issuing the reset, wait until this bit is cleared.
- FIFO reset - Resets the FIFO by setting the fifo_reset bit (bit 1) in the CTRL register; this resets the FIFO pointers and counters of the FIFO. Since this reset bit is self-clearing, after issuing the reset, wait until this bit is cleared.

  DMA reset - Resets the internal DMA controller logic by setting the dma_reset bit (bit 2) in the CTRL register, which abruptly terminates any DMA transfer in process. Since this reset bit is self-clearing, after issuing the reset, wait until this bit is cleared.

The following are recommended methods for issuing reset commands:

- Non-DMA transfer mode - Simultaneously sets controller_reset and fifo_reset; clears the RAWINTS register @0x44 using another write in order to clear any resultant interrupt.
- Generic DMA mode - Simultaneously sets controller_reset, fifo_reset, and dma_reset; clears the RAWINTS register @0x44 by using another write in order to clear any resultant interrupt. If a "graceful" completion of the DMA is required, then it is recommended to poll the status register to see whether the dma request is 0 before resetting the DMA interface control and issuing an additional FIFO reset.
- In DMA transfer mode, even when the FIFO pointers are reset, if there is a DMA transfer in progress, it could push or pop data to or from the FIFO; the DMA itself completes correctly. In order to clear the FIFO, the software should issue an additional FIFO reset and clear any FIFO underrun or overrun errors in the RAWINTS register caused by the DMA transfers after the FIFO was reset.

20.7.4.4 Error Handling

The Module implements error checking; errors are reflected in the RAWINTS register @0x44 and can be communicated to the software through an interrupt, or the software can poll for these bits. Upon power-on, interrupts are disabled (int_enable in the CTRL register is 0), and all the interrupts are masked (bits 0-31 of the INTMASK register; default is 0). Error handling:

- Response and data time-out errors - For response time-out, software can retry the command. For data time-out, the Module has not received the data start bit - either for the first block or the intermediate block - within the time-out period, so software can

either retry the whole data transfer again or retry from a specified block onwards. By reading the contents of the TCBCNT later, the software can decide how many bytes remain to be copied.

- Response errors - Set when an error is received during response reception. In this case, the response that copied in the response registers is invalid. Software can retry the command.
- Data errors - Set when error in data reception are observed; for example, data CRC, start bit not found, end bit not found, and so on. These errors could be set for any block-first block, intermediate block, or last block. On receipt of an error, the software can issue a STOP or ABORT command and retry the command for either whole data or partial data.
- Hardware locked error - Set when the Module cannot load a command issued by software. When software sets the start_cmd bit in the CMD register, the Module tries to load the command. If the command buffer is already filled with a command, this error is raised. The software then has to reload the command.
- FIFO underrun/overrun error - If the FIFO is full and software tries to write data in the FIFO, then an overrun error is set. Conversely, if the FIFO is empty and the software tries to read data from the FIFO, an underrun error is set. Before reading or writing data in the FIFO, the software should read
- the fifo_empty or fifo_full bits in the Status register.
- Data starvation by cpu time-out - Raised when the Module is waiting for software intervention to transfer the data to or from the FIFO, but the software does not transfer within the stipulated time-out period. Under this condition and when a read transfer is in process, the software
- Should read data from the FIFO and create space for further data reception. When a transmit operation is in process, the software should fill data in the FIFO in order to start transferring data to the card.
- CRC Error on Command - If a CRC error is detected for a command, the CE-ATA device does not send a response, and a response time-out is expected from the Module. The ATA layer is notified that an MMC transport layer error occurred.
- Write operation - Any MMC Transport layer error known to the device causes an outstanding ATA command to be terminated. The ERR bits are set in the ATA status registers and the appropriate error code is sent to the ATA Error register.
- If nIEN=0, then the Command Completion Signal (CCS) is sent to the cpu.

If device interrupts are not enabled (nIEN=1), then the device completes the entire Data Unit Count if the cpu controller does not abort the ongoing transfer.

During a multiple-block data transfer, if a negative CRC status is received from the device, the data path signals a data CRC error to the BIU by setting the data CRC error bit in the RINTSTS register. It then continues further data transmission until all the bytes are transmitted.

- Read operation - If MMC transport layer errors are detected by the cpu controller, the cpu completes the ATA command with an error status.

The cpu controller can issue a Command Completion Signal Disable (CCSD) followed by a STOP TRANSMISSION (CMD12) to abort the read transfer. The cpu can also transfer the entire Data Unit Count bytes without aborting the data transfer.

20.7.5 DMA descriptors

The SD/MMC DMA controller uses the following descriptor structures:

- Dual-Buffer Structure – The distance between two descriptors is determined by the Skip Length value programmed in the Descriptor Skip Length (DSL) field of the Bus Mode Register (BMOD).
- Chain Structure – Each descriptor points to a unique buffer and the next descriptor.

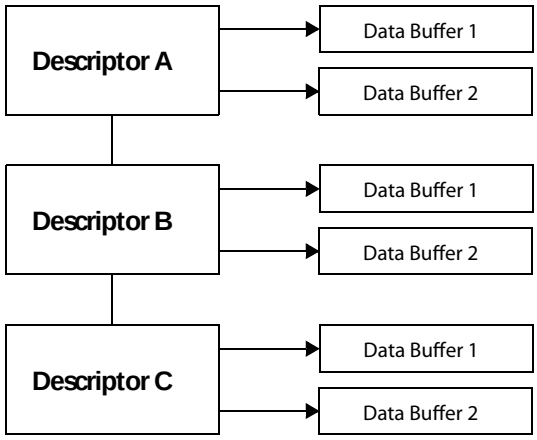


Fig 41. Dual-buffer descriptor structure

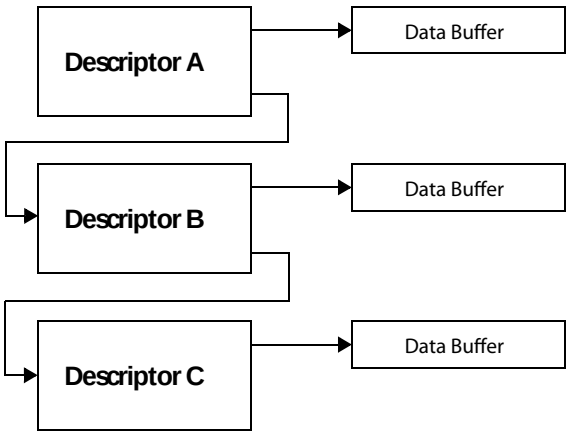


Fig 42. Chain descriptor structure

20.7.5.1 SD/MMC DMA descriptors

20.7.5.1.1 SD/MMC DMA descriptor DESC0

The DES0 descriptor contains control and status information.

Table 335. SD/MMC DMA DESC0 descriptor

Bit	Symbol	Description
0	-	Reserved
1	DIC	Disable Interrupt on Completion When set, this bit will prevent the setting of the TI/RI bit of the IDMAC Status Register (IDSTS) for the data that ends in the buffer pointed to by this descriptor.
2	LD	Last Descriptor When set, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the data.
3	FS	First Descriptor When set, this bit indicates that this descriptor contains the first buffer of the data. If the size of the first buffer is 0, next Descriptor contains the beginning of the data.
4	CH	Second Address Chained When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When this bit is set, BS2 (DES1[25:13]) should be all zeros.
5	ER	End of Ring When set, this bit indicates that the descriptor list reached its final descriptor. The IDMAC returns to the base address of the list, creating a Descriptor Ring. This is meaningful for only a dual-buffer descriptor structure.
29:6	-	Reserved
30	CES	Card Error Summary These error bits indicate the status of the transaction to or from the card. These bits are also present in RINTSTS Indicates the logical OR of the following bits: • EBE: End Bit Error • RTO: Response Time-out • RCRC: Response CRC • SBE: Start Bit Error • DRTO: Data Read Time-out • DCRC: Data CRC for Receive • RE: Response Error
31	OWN	When set, this bit indicates that the descriptor is owned by the SD/MMC DMA. When this bit is reset, it indicates that the descriptor is owned by the Host. The SD/MMC DMA clears this bit when it completes the data transfer.

20.7.5.1.2 SD/MMC DMA descriptor DESC1

The DES1 descriptor contains the buffer size.

Table 336. SD/MMC DMA DESC1 descriptor

Bit	Symbol	Description
12:0	BS1	Buffer 1 Size Indicates the data buffer byte size, which must be a multiple of 4 bytes. In the case where the buffer size is not a multiple of 4, the resulting behavior is undefined. If this field is 0, the DMA ignores this buffer and proceeds to the next descriptor in case of a chain structure, or to the next buffer in case of a dual-buffer structure. **Remark:** If there is only one descriptor and only one buffer to be programmed, use only the Buffer 1 and not Buffer 2.
25:13	BS2	Buffer 2 Size These bits indicate the second data buffer byte size. The buffer size must be a multiple of 4. Otherwise, the resulting behavior is undefined. This field is not valid if DES0[4] is set.
31:26	-	Reserved

20.7.5.1.3 SD/MMC DMA descriptor DESC2

The DES2 descriptor contains the address pointer to the data buffer;

Table 337. SD/MMC DMA DESC2 descriptor

Bit	Symbol	Description
31:0	BAP1	Buffer Address Pointer 1 These bits indicate the physical address of the first data buffer. The SD/MMC DMA ignores DES2 [2/1/0:0], corresponding to the bus width of 64/32/16, internally.

20.7.5.1.4 SD/MMC DMA descriptor DESC3

The DES3 descriptor contains the address pointer to the next descriptor if the present descriptor is not the last descriptor in a chained descriptor structure or the second buffer address for a dual-buffer structure.

Table 338. SD/MMC DMA DESC3 descriptor

Bit	Symbol	Description
31:0	BAP2	Buffer Address Pointer 2/ Next Descriptor Address These bits indicate the physical address of the second buffer when the dual-buffer structure is used. If the Second Address Chained (DES0[4]) bit is set, then this address contains the pointer to the physical memory where the Next Descriptor is present. If this is not the last descriptor, then the Next Descriptor address pointer must be bus-width aligned (DES3[2/1/0:0] = 0 , internally the LSBs are ignored).

20.7.5.2 Initialization

For the SD/MMC DMA initialization, follow these steps:

1. Write to the Bus Mode Register (BMOD) to set the Host bus access parameters.
2. Write to the Interrupt Enable Register (IDINTEN) to mask unnecessary interrupt causes.
3. The software driver creates either the Transmit or the Receive descriptor list. Then it writes to Descriptor List Base Address Register (DBADDR), providing the IDMAC with the starting address of the list.

4. The SD/MMC DMA engine attempts to acquire descriptors from the descriptor lists.

20.7.5.3 Host bus burst access

The SD/MMC DMA attempts to execute fixed-length burst transfers on the AHB Master interface if configured using the FB bit of the IDMAC Bus Mode register. The maximum burst length is indicated and limited by the PBL field. The descriptors are always accessed in the maximum possible burst-size for the 16-bytes to be read: 16*8/bus-width.

The SD/MMC DMA initiates a data transfer only when sufficient space to accommodate the configured burst is available in the FIFO or the number of bytes to the end of data, when less than the configured burst-length. The SD/MMC DMA indicates the start address and the number of transfers required to the AHB Master Interface. When the AHB Interface is configured for fixed-length bursts, then it transfers data using the best combination of INCR4/8/16 and SINGLE transactions. Otherwise, in no fixed-length bursts, it transfers data using INCR (undefined length) and SINGLE transactions.

20.7.5.4 Host data buffer alignment

The transmit and receive data buffers in host memory must be 32-bit aligned.

20.7.5.5 Buffer size calculations

The driver knows the amount of data to transmit or receive. For transmitting to the card, the IDMAC transfers the exact number of bytes to the FIFO, indicated by the buffer size field of DES1.

If a descriptor is not marked as last - LS bit of DES0 - then the corresponding buffers of the descriptor are full, and the amount of valid data in a buffer is accurately indicated by its buffer size field. If a descriptor is marked as last, then the buffer cannot be full, as indicated by the buffer size in DES1. The driver is aware of the number of locations that are valid in this case.

20.7.5.6 Transmission

The SD/MMC transmission occurs as follows:

1. The Host sets up the Descriptor (DES0-DES3) for transmission and sets the OWN bit (DES0[31]). The Host also prepares the data buffer.
2. The Host programs the write data command in the CMD register in BIU.
3. The Host will also program the required transmit threshold level (TX_WMark field in FIFOTH register).
4. The SD/MMC DMA determines that a write data transfer needs to be done as a consequence of step 2.
5. The SD/MMC DMA engine fetches the descriptor and checks the OWN bit. If the OWN bit is not set, it means that the host owns the descriptor. In this case the SD/MMC DMA enters suspend state and asserts the Descriptor Unable interrupt in the SD/MMC DMA status register (IDSTS). In such a case, the host needs to release the SD/MMC DMA by writing any value to the poll demand register.
6. It will then wait for Command Done (CD) bit and no errors from BIU which indicates that a transfer can be done.

7. The SD/MMC DMA engine will now wait for a DMA interface request from BIU. This request will be generated based on the programmed transmit threshold value. For the last bytes of data which can't be accessed using a burst, SINGLE transfers are performed on AHB Master Interface.
8. The SD/MMC DMA fetches the Transmit data from the data buffer in the Host memory and transfers to the FIFO for transmission to card.
9. When data spans across multiple descriptors, the SD/MMC DMA will fetch the next descriptor and continue with its operation with the next descriptor. The Last Descriptor bit in the descriptor indicates whether the data spans multiple descriptors or not.
10. When data transmission is complete, status information is updated in SD/MMC DMA status register (IDSTS) by setting Transmit Interrupt, if enabled. Also, the OWN bit is cleared by the SD/MMC DMA by performing a write transaction to DES0.

20.7.5.7 Reception

The SD/MMC reception occurs as follows:

1. The Host sets up the Descriptor (DES0-DES3) for reception, sets the OWN (DES0[31]).
2. The Host programs the read data command in the CMD register in BIU.
3. The Host will program the required receive threshold level (RX_WMark field in FIFOTH register).
4. The SD/MMC DMA determines that a read data transfer needs to be done as a consequence of step 2.
5. The SD/MMC DMA engine fetches the descriptor and checks the OWN bit. If the OWN bit is not set, it means that the host owns the descriptor. In this case the DMA enters suspend state and asserts the Descriptor Unable interrupt in the SD/MMC DMA status register (IDSTS). In such a case, the host needs to release the SD/MMC DMA by writing any value to the poll demand register.
6. It will then wait for Command Done (CD) bit and no errors from BIU which indicates that a transfer can be done.
7. The SD/MMC DMA engine will now wait for a DMA interface request (dw_dma_req) from BIU. This request will be generated based on the programmed receive threshold value. For the last bytes of data which can't be accessed using a burst, SINGLE transfers are performed on AHB.
8. The SD/MMC DMA fetches the data from the FIFO and transfer to Host memory.
9. When data spans across multiple descriptors, the SD/MMC DMA will fetch the next descriptor and continue with its operation with the next descriptor. The Last Descriptor bit in the descriptor indicates whether the data spans multiple descriptors or not.
10. When data reception is complete, status information is updated in SD/MMC DMA status register (IDSTS) by setting Receive Interrupt, if enabled. Also, the OWN bit is cleared by the SD/MMC DMA by performing a write transaction to DES0.

20.7.5.8 Interrupts

Interrupts can be generated as a result of various events. The SD/MMC DMA Status Register (IDSTS) contains all the bits that might cause an interrupt. The SD/MMC DMA Interrupt Enable Register (IDINTEN) contains an Enable bit for each of the events that can cause an interrupt.

There are two groups of summary interrupts - Normal and Abnormal - as outlined in Status Register (IDSTS). Interrupts are cleared by writing a 1 to the corresponding bit position. When all the enabled interrupts within a group are cleared, the corresponding summary bit is cleared. When both the summary bits are cleared, the interrupt signal is de-asserted.

Interrupts are not queued and if the interrupt event occurs before the driver has responded to it, no additional interrupts are generated. For example, Receive Interrupt (IDSTS[1]) indicates that one or more data was transferred to the Host buffer.

An interrupt is generated only once for simultaneous, multiple events. The driver must scan the SD/MMC DMA Status Register for the interrupt cause.

Remark: The final interrupt (int) signal from created is a logical OR of the interrupt from BIU and SD/MMC DMA.

20.7.5.9 Abort

When the host issues CMD12 when a data transfer on the card data lines is in progress, the FSM closes the present descriptor after completing the transfer of data until a DTO interrupt is asserted. Once an abort command is issued, the DMA performs single burst transfers:

1. When the host issues CMD12 when a data transfer on the card data lines is in progress, the FSM closes the present descriptor after completing the transfer of data until a DTO interrupt is asserted. Once an abort command is issued, the DMAC performs single burst transfers.
2. For a card read, the SD/MMC DMA keeps popping data from FIFO and writes to the host memory until a DTO interrupt is generated. This is required since DTO interrupt is not generated until and unless all the FIFO data is emptied.

Remark: The following scenarios apply for closing the descriptors:

- In case of an FBE, the current descriptor and the remaining unread descriptors are not closed by the SD/MMC DMA.
- In case of a write abort, only the current descriptor during which an abort occurred is closed by the SD/MMC DMA . The remaining unread descriptors are not closed by the IDMAC.
- In case of a read abort, the SD/MMC DMA pops the data out of the FIFO and writes them to the corresponding descriptor data buffers. The remaining unread descriptors are not closed.

20.7.5.10 FBE scenarios

An FBE occurs due to an AHB error response on the AHB bus. This is a system error, so the software driver should not perform any further programming to the SD/MMC The only recovery mechanism from such scenarios is to do one of the following:

- Issue a hard reset by asserting the reset_n signal .
- Do a program controller reset by writing to the CTRL[0] register .

20.7.5.11 FIFO overflow and underflow

During normal data transfer conditions, FIFO overflow and underflow will not occur. However if there is a programming error, then FIFO overflow/underflow can result. For example, consider the following scenarios.

For transmit: PBL=4, Tx watermark = 1. For these programming values, if the FIFO has only one location empty, it issues a dw_dma_req to DMA state machine. Due to PBL value=4, the DMA performs 4 pushes into the FIFO. This will result in a FIFO overflow interrupt.

For receive: PBL=4, Rx watermark = 1. For these programming values, if the FIFO has only one location filled, it issues a dw_dma_req to the DMA state machine. Due to PBL value=4, the DMA performs 4 pops to the FIFO. This will result in a FIFO underflow interrupt.

The driver should ensure that the number of bytes to be transferred as indicated in the descriptor should be a multiple of 4 bytes. For example, if the BYTCNT = 13, the number of bytes indicated in the descriptor should be 16.

20.7.5.12 Programming of PBL and watermark levels

The SD/MMC DMA performs data transfers depending on the programmed PBL and threshold values. Table 339 lists the allowed programming values.

Table 339. PBL and watermark levels

PBL (number of transfers)	Transmit/receive watermark value
1	greater than or equal to 1
4	greater than or equal to 4
8	greater than or equal to 8
16	greater than or equal to 16
32	greater than or equal to 32
64	greater than or equal to 64
128	greater than or equal to 128
264	greater than or equal to 256

21.1 How to read this chapter

The EMC is available on all LPC43xx parts.

The memory and address bus widths depend on package size (see Table 340).

Table 340. EMC pinout for different packages

Function	LBGA256	TFBGA180	TFBGA100	LQFP208	LQFP144	LQFP100
A	EMC_A[22:0]	EMC_A[22:0]	EMC_A[13:0]	EMC_A[22:0]	EMC_A[15:0]	EMC_A[13:0]
D	EMC_D[31:0]	EMC_D[15:0]	EMC_D[7:0]	EMC_D[29:0]	EMC_D[15:0]	EMC_D[7:0]
$\overline{BLS}$	$\overline{EMC\_BLS[3:0]}$	$\overline{EMC\_BLS[3:0]}$	BLS0	$\overline{EMC\_BLS[3:0]}$	$\overline{EMC\_BLS[1:0]}$	BLS0
$\overline{CS}$	$\overline{EMC\_CS[3:0]}$	$\overline{EMC\_CS[3:0]}$	CS0	$\overline{EMC\_CS[3:0]}$	$\overline{EMC\_CS[1:0]}$	CS0
$\overline{OE}$	$\overline{EMC\_OE}$	$\overline{EMC\_OE}$	$\overline{EMC\_OE}$	$\overline{EMC\_OE}$	$\overline{EMC\_OE}$	$\overline{EMC\_OE}$
$\overline{WE}$	$\overline{EMC\_WE}$	$\overline{EMC\_WE}$	$\overline{EMC\_WE}$	$\overline{EMC\_WE}$	$\overline{EMC\_WE}$	$\overline{EMC\_WE}$
$\overline{CKEOUT}$	$\overline{EMC\_CKEOUT[3:0]}$	$\overline{EMC\_CKEOUT[1:0]}$	$\overline{EMC\_CKEOUT[1:0]}$	$\overline{EMC\_CKEOUT[1:0]}$	$\overline{EMC\_CKEOUT[1:0]}$	EMC_CKEOUT0
CLK	EMC_CLK[3:0]; EMC_CLK12, EMC_CLK23	EMC_CLK0, EMC_CLK3; EMC_CLK12, EMC_CLK23	EMC_CLK0, EMC_CLK3; EMC_CLK12, EMC_CLK23	EMC_CLK0, EMC_CLK3; EMC_CLK12, EMC_CLK23	EMC_CLK0, EMC_CLK3; EMC_CLK12, EMC_CLK23	EMC_CLK0, EMC_CLK3; EMC_CLK12, EMC_CLK23
$\overline{DQMOUT}$	$\overline{EMC\_DQMOUT[3:0]}$	$\overline{EMC\_DQMOUT[1:0]}$	-	$\overline{EMC\_DQMOUT[1:0]}$	$\overline{EMC\_DQMOUT[1:0]}$	-
$\overline{DYCS}$	$\overline{EMC\_DYCS[3:0]}$	$\overline{EMC\_DYCS[2:0]}$	$\overline{EMC\_DYCS[1:0]}$	$\overline{EMC\_DYCS[2:0]}$	$\overline{EMC\_DYCS[1:0]}$	$\overline{EMC\_DYCS0}$
$\overline{CAS}$	$\overline{EMC\_CAS}$	$\overline{EMC\_CAS}$	$\overline{EMC\_CAS}$	$\overline{EMC\_CAS}$	$\overline{EMC\_CAS}$	$\overline{EMC\_CAS}$
$\overline{RAS}$	$\overline{EMC\_RAS}$	$\overline{EMC\_RAS}$	$\overline{EMC\_RAS}$	$\overline{EMC\_RAS}$	$\overline{EMC\_RAS}$	$\overline{EMC\_RAS}$

21.2 Basic configuration

The External Memory Controller is configured as follows:

- See Table 341 for clocking and power control. Two clocks are supported: BASE_M3_CLK and 1/2 × BASE_M3_CLK.
- If the EMC CCLK is configured for 1/2 × BASE_M3_CLK, the CLK_M3_EMC_DIV branch clock must be configured for half-frequency clock operation in both the CREG6 register (Table 43) and the CCU1 CLK_EMCDIV_CFG register (Table 97).
- To use 16-bit wide and 32-bit wide SDRAM interfaces, select the EMC_CLK function and enable the input buffer (EZI = 1) in all four SFSCLKn registers in the SCU.
- The EMC is reset by the EMC_RST (reset # 21).
- Program the SDRAM Delay value for the EMC_CLKn lines in the EMCDELAYCLK register in the SCU block. (See Section 15.4.9.). Add the SDRAM delay for most SDRAM devices running at frequencies above 96 MHz under typical conditions. Add the SDRAM delay at any frequency to compensate for variations over temperature. For details, see the *LPC4350_30_20_10 data sheet.*

Table 341. EMC clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
EMC registers and EMC CCLK	BASE_M4_CLK	CLK_M4_EMC	up to 204 MHz	The maximum operating frequency depends on temperature and voltage settings and is typically 120 MHz for SDRAM devices. For details, see the *LPC4350_30_20_10 data sheet.*
EMC CCLK (divided clock)	BASE_M4_CLK	CLK_M4_EMC_DIV	up to 204 MHz	This is an alternative clock option for CCLK. This clock can run at the same frequency as BASE_M4_CLK or half the frequency of BASE_M4_CLK.

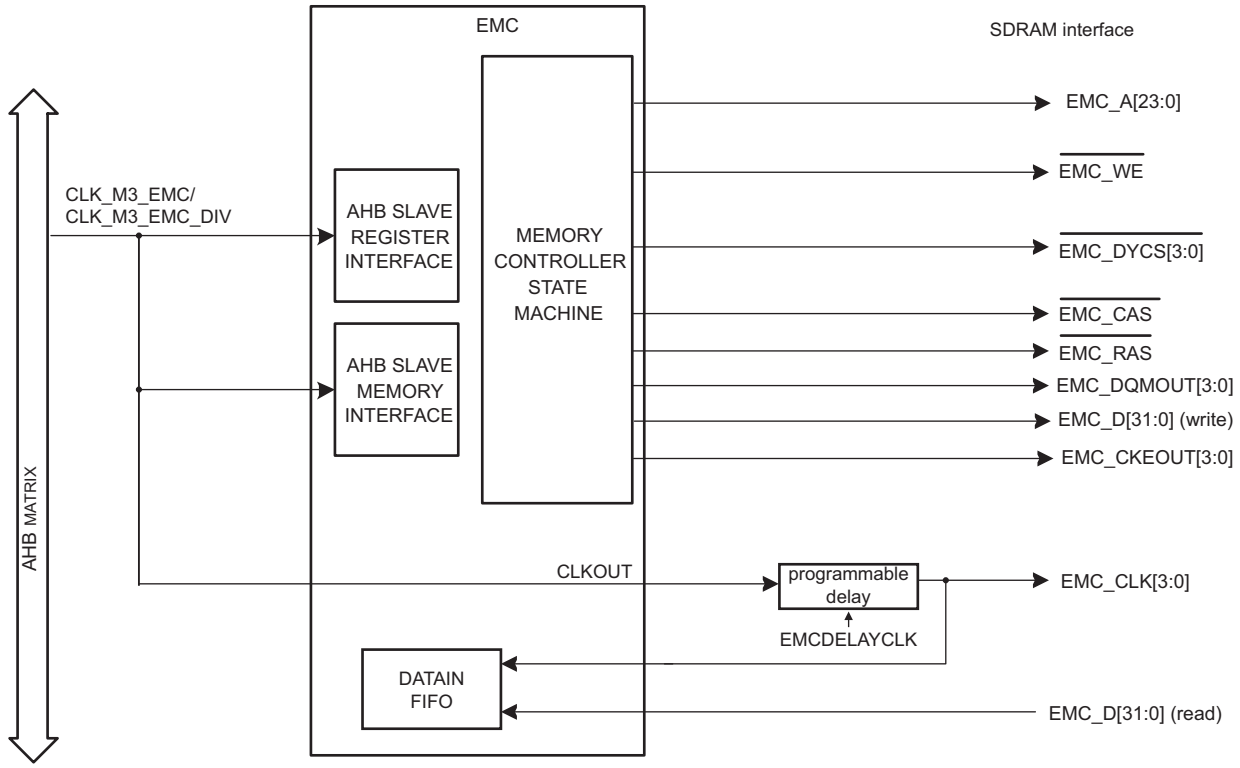
21.3 Features

- 8-bit, 16-bit, and 32-bit wide static memory support with up to four chip selects.
- Asynchronous static memory device support including RAM, ROM, and NOR Flash, with or without asynchronous page mode.
- Static memory features include:
  - Asynchronous page mode read
  - Programmable wait states
  - Bus turnaround delay
  - Output enable and write enable delays
  - Extended wait
- 16-bit and 32-bit wide chip select SDRAM memory support with up to four chip selects and up to 256 MB of data.
- Controller supports 2 kbit, 4 kbit, and 8 kbit row address synchronous memory parts. That is typical 512 MB, 256 MB, and 128 MB parts, with 4, 8, 16, or 32 data bits per device.
- Dynamic memory interface support including Single Data Rate SDRAM.
- Dynamic memory self-refresh mode controlled by software.
- Power-saving modes dynamically control EMC_CKEOUT and EMC_CLK to SDRAMs.
- Low transaction latency.
- Read and write buffers to reduce latency and to improve performance.
- Separate reset domains allow the for auto-refresh through a chip reset if desired.
- Programmable delay elements allow fine-tuning EMC timing.

Remark: Synchronous static memory devices (synchronous burst mode) are not supported.

21.4 General description

The LPC43xx External Memory Controller (EMC) is an ARM PrimeCell MultiPort Memory Controller peripheral offering support for asynchronous static memory devices such as RAM, ROM and Flash, as well as dynamic memories such as Single Data Rate SDRAM.



Program the EMC_CLKn delays in the EMCDELAYCLK register in the SYSCON block (Table 129), to adjust the delays for different SDRAM frequencies and operating conditions.

Fig 43. EMC block diagram (SDRAM)

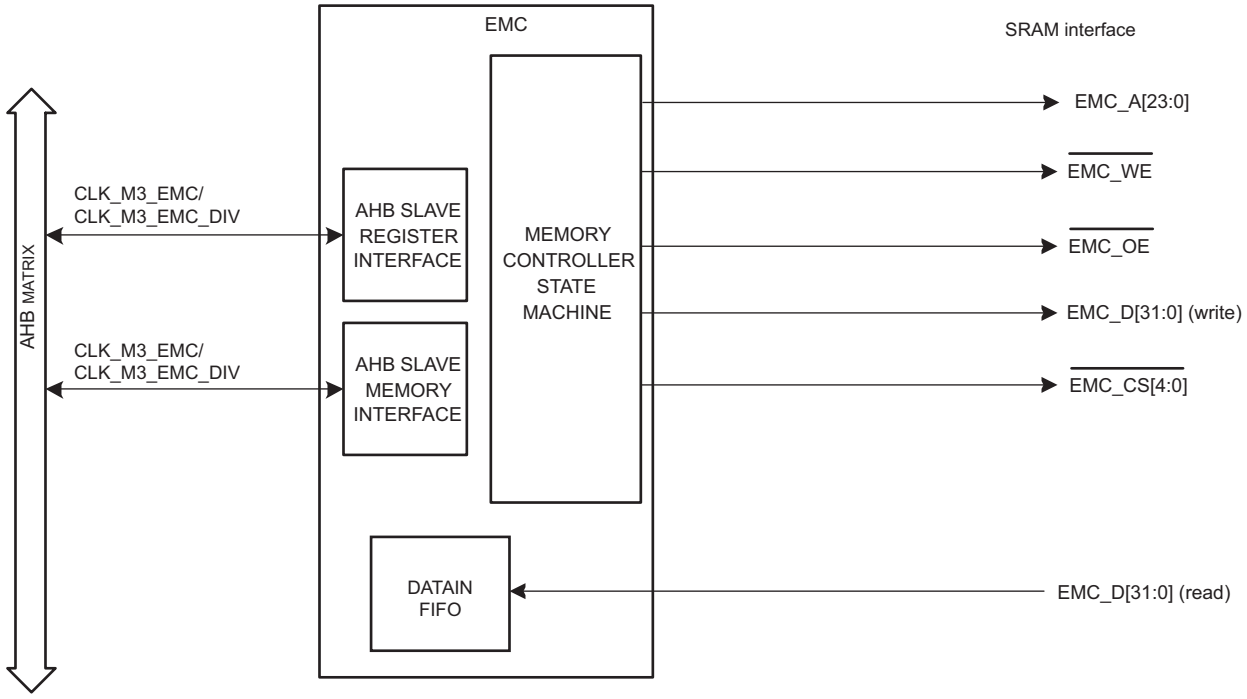


Fig 44. EMC block diagram (SRAM)

21.5 Memory bank select

Eight independently-configurable memory chip selects are supported:

- Pins $\overline{\text{EMC\_CS3}}$ to $\overline{\text{EMC\_CS0}}$ are used to select static memory devices.
- Pins $\overline{\text{EMC\_DYCS3}}$ to $\overline{\text{EMC\_DYCS0}}$ are used to select dynamic memory devices.

Static memory chip select ranges are each 16 Megabytes in size, while dynamic memory chip selects cover a range of 256 megabytes each. Table 342 shows the address ranges of the chip selects.

Table 342. Memory bank selection

Chip select pin	Address range	Memory type	Size of range
$\overline{\text{EMC\_CS0}}$	0x1C00 0000 - 0x1CFF FFFF	Static	16 MB
$\overline{\text{EMC\_CS1}}$	0x1D00 0000 - 01DFF FFFF	Static	16 MB
$\overline{\text{EMC\_CS2}}$	0x1E00 0000 - 0x1EFF FFFF	Static	16 MB
$\overline{\text{EMC\_CS3}}$	0x1F00 0000 - 0x1FFF FFFF	Static	16 MB
$\overline{\text{EMC\_DYCS0}}$	0x2800 0000 - 0x2FFF FFFF	Dynamic	128 MB
$\overline{\text{EMC\_DYCS1}}$	0x3000 0000 - 0x3FFF FFFF	Dynamic	256 MB
$\overline{\text{EMC\_DYCS2}}$	0x6000 0000 - 0x6FFF FFFF	Dynamic	256 MB
$\overline{\text{EMC\_DYCS3}}$	0x7000 0000 - 0x7FFF FFFF	Dynamic	256 MB

21.6 Pin description

Table 343. EMC pin description

Function name	Direction	Description
EMC_A[22:0]	O	Address bus
EMC_D[31:0]	I/O	Data bus
$\overline{\text{EMC\_BLS[3:0]}}$	O	Byte lane select
$\overline{\text{EMC\_CS[3:0]}}$	O	Static RAM memory bank select
$\overline{\text{EMC\_OE}}$	O	Output enable
$\overline{\text{EMC\_WE}}$	O	Write enable
$\overline{\text{EMC\_CKEOUT[3:0]}}$	O	SDRAM clock enable signals
$\overline{\text{EMC\_CLK[3:0]}}$	O	SDRAM clock signals
$\overline{\text{EMC\_DQMOUT[3:0]}}$	O	Data mask output to SDRAM memory banks
$\overline{\text{EMC\_DYCS[3:0]}}$	O	SDRAM memory bank select
$\overline{\text{EMC\_CAS}}$	O	Column address strobe
$\overline{\text{EMC\_RAS}}$	O	Row address strobe

21.7 Register description

This chapter describes the EMC registers and provides details required when programming the microcontroller. The EMC registers are shown in Table 344. Reset value reflects the data stored in used bits only. It does not include the content of reserved bits.

Table 344. Register overview: External memory controller (base address 0x4000 5000)

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reference
CONTROL	R/W	0x000	Controls operation of the memory controller.	0x3	[1]0x1	Table 345
STATUS	RO	0x004	Provides EMC status information.	0x5	0x5	Table 346
CONFIG	R/W	0x008	Configures operation of the memory controller.	0	0	Table 347
-	-	0x00C - 0x01C	Reserved.	-	-	-
DYNAMICCONTROL	R/W	0x020	Controls dynamic memory operation.	0x6	0x6	Table 348
DYNAMICREFRESH	R/W	0x024	Configures dynamic memory refresh operation.	0	0	Table 349
DYNAMICREADCONFIG	R/W	0x028	Configures the dynamic memory read strategy.	0	0	Table 350
-	-	0x02C	Reserved.	-	-	-
DYNAMICRP	R/W	0x030	Selects the precharge command period.	0xF	0xF	Table 351
DYNAMICRAS	R/W	0x034	Selects the active to precharge command period.	0xF	0xF	Table 352
DYNAMICSREX	R/W	0x038	Selects the self-refresh exit time.	0xF	0xF	Table 353

Table 344. Register overview: External memory controller (base address 0x4000 5000) *...continued*

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reference
DYNAMICAPR	R/W	0x03C	Selects the last-data-out to active command time.	0xF	0xF	Table 354
DYNAMICDAL	R/W	0x040	Selects the data-in to active command time.	0xF	0xF	Table 355
DYNAMICWR	R/W	0x044	Selects the write recovery time.	0xF	0xF	Table 356
DYNAMICRC	R/W	0x048	Selects the active to active command period.	0x1F	0x1F	Table 357
DYNAMICRFC	R/W	0x04C	Selects the auto-refresh period.	0x1F	0x1F	Table 358
DYNAMICXSR	R/W	0x050	Selects the exit self-refresh to active command time.	0x1F	0x1F	Table 359
DYNAMICRRD	R/W	0x054	Selects the active bank A to active bank B latency.	0xF	0xF	Table 360
DYNAMICMRD	R/W	0x058	Selects the load mode register to active command time.	0xF	0xF	Table 361
-	R/W	0x05C - 0x07C	Reserved.	-	-	-
STATICEXTENDEDWAIT	R/W	0x080	Selects time for long static memory read and write transfers.	0	0	Table 362
-	R/W	-	Reserved.	-	-	-
DYNAMICCONFIG0	R/W	0x100	Selects the configuration information for dynamic memory chip select 0.	0	0	Table 363
DYNAMICRASCAS0	R/W	0x104	Selects the RAS and CAS latencies for dynamic memory chip select 0.	0x303	0x303	Table 365
-		0x108 - 0x11C	Reserved.	-	-	-
DYNAMICCONFIG1	R/W	0x120	Selects the configuration information for dynamic memory chip select 1.	0	0	Table 363
DYNAMICRASCAS1	R/W	0x124	Selects the RAS and CAS latencies for dynamic memory chip select 1.	0x303	0x303	Table 365
-	-	0x128 - 0x13C	Reserved.	-	-	-
DYNAMICCONFIG2	R/W	0x140	Selects the configuration information for dynamic memory chip select 2.	0	0	Table 363
DYNAMICRASCAS2	R/W	0x144	Selects the RAS and CAS latencies for dynamic memory chip select 2.	0x303	0x303	Table 365
-	-	0x148 - 0x15C	Reserved.	-	-	-
DYNAMICCONFIG3	R/W	0x160	Selects the configuration information for dynamic memory chip select 3.	0	0	Table 363
DYNAMICRASCAS3	R/W	0x164	Selects the RAS and CAS latencies for dynamic memory chip select 3.	0x303	0x303	Table 365
-	-	0x168 - 0x1FC	Reserved.	-	-	-

Table 344. Register overview: External memory controller (base address 0x4000 5000) *...continued*

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reference
STATICCONFIG0	R/W	0x200	Selects the memory configuration for static chip select 0.	0	0x81	Table 366
STATICWAITWEN0	R/W	0x204	Selects the delay from chip select 0 to write enable.	0	0	Table 367
STATICWAITOEN0	R/W	0x208	Selects the delay from chip select 0 or address change, whichever is later, to output enable.	0	0	Table 368
STATICWAITRD0	R/W	0x20C	Selects the delay from chip select 0 to a read access.	0x1F	[2] 0xE	Table 369
STATICWAITPAGE0	R/W	0x210	Selects the delay for asynchronous page mode sequential accesses for chip select 0.	0x1F	0x1F	Table 370
STATICWAITWR0	R/W	0x214	Selects the delay from chip select 0 to a write access.	0x1F	0x1F	Table 371
STATICWAITTURN0	R/W	0x218	Selects the number of bus turnaround cycles for chip select 0.	0xF	0xF	Table 372
-	-	0x21C	Reserved.	-	-	-
STATICCONFIG1	R/W	0x220	Selects the memory configuration for static chip select 1.	0	0	Table 366
STATICWAITWEN1	R/W	0x224	Selects the delay from chip select 1 to write enable.	0	0	Table 367
STATICWAITOEN1	R/W	0x228	Selects the delay from chip select 1 or address change, whichever is later, to output enable.	0	0	Table 368
STATICWAITRD1	R/W	0x22C	Selects the delay from chip select 1 to a read access.	0x1F	0x1F	Table 369
STATICWAITPAGE1	R/W	0x230	Selects the delay for asynchronous page mode sequential accesses for chip select 1.	0x1F	0x1F	Table 370
STATICWAITWR1	R/W	0x234	Selects the delay from chip select 1 to a write access.	0x1F	0x1F	Table 371
STATICWAITTURN1	R/W	0x238	Selects the number of bus turnaround cycles for chip select 1.	0xF	0xF	Table 372
-	-	0x23C	Reserved.	-	-	-
STATICCONFIG2	R/W	0x240	Selects the memory configuration for static chip select 2.	0	0	Table 366
STATICWAITWEN2	R/W	0x244	Selects the delay from chip select 2 to write enable.	0	0	Table 367
STATICWAITOEN2	R/W	0x248	Selects the delay from chip select 2 or address change, whichever is later, to output enable.	0	0	Table 368
STATICWAITRD2	R/W	0x24C	Selects the delay from chip select 2 to a read access.	0x1F	0x1F	Table 369

Table 344. Register overview: External memory controller (base address 0x4000 5000) *...continued*

Name	Access	Address offset	Description	Reset value	Reset value after EMC boot	Reference
STATICWAITPAGE2	R/W	0x250	Selects the delay for asynchronous page mode sequential accesses for chip select 2.	0x1F	0x1F	Table 370
STATICWAITWR2	R/W	0x254	Selects the delay from chip select 2 to a write access.	0x1F	0x1F	Table 371
STATICWAITTURN2	R/W	0x258	Selects the number of bus turnaround cycles for chip select 2.	0xF	0xF	Table 372
-	-	0x25C	Reserved.	-	-	-
STATICCONFIG3	R/W	0x260	Selects the memory configuration for static chip select 3.	0	0	Table 366
STATICWAITWEN3	R/W	0x264	Selects the delay from chip select 3 to write enable.	0	0	Table 367
STATICWAITOEN3	R/W	0x268	Selects the delay from chip select 3 or address change, whichever is later, to output enable.	0	0	Table 368
STATICWAITRD3	R/W	0x26C	Selects the delay from chip select 3 to a read access.	0x1F	0x1F	Table 369
STATICWAITPAGE3	R/W	0x270	Selects the delay for asynchronous page mode sequential accesses for chip select 3.	0x1F	0x1F	Table 370
STATICWAITWR3	R/W	0x274	Selects the delay from chip select 3 to a write access.	0x1F	0x1F	Table 371
STATICWAITTURN3	R/W	0x278	Selects the number of bus turnaround cycles for chip select 3.	0xF	0xF	Table 372

[1] The reset value after warm reset for the CONTROL register is 0x0000 0001.

[2] If booting from EMC, see Section 5.3.4.3.

21.7.1 EMC Control register

The Control register is a read/write register that controls operation of the memory controller. The control bits can be altered during normal operation.

Table 345. EMC Control register (CONTROL - address 0x4000 5000) bit description

Bit	Symbol	Value	Description	Reset value
0	E		EMC Enable. Indicates if the EMC is enabled or disabled.Disabling the EMC reduces power consumption. When the memory controller is disabled the memory is not refreshed. The memory controller is enabled by setting the enable bit, or by reset. This bit must only be modified when the EMC is in idle state.[1]	1
		0	Disabled	
		1	Enabled (POR and warm reset value).	

Table 345. EMC Control register (CONTROL - address 0x4000 5000) bit description

Bit	Symbol	Value	Description	Reset value
1	M		Address mirror. Indicates normal or reset memory map. On POR, CS1 is mirrored to both CS0 and DYCS0 memory areas. Clearing the M bit enables CS0 and DYCS0 memory to be accessed.	1
		0	Normal memory map.	
		1	Reset memory map. Static memory CS1 is mirrored onto CS0 and DYCS0 (POR reset value).	
2	L		Low-power mode. Indicates normal, or low-power mode. Entering low-power mode reduces memory controller power consumption. Dynamic memory is refreshed as necessary. The memory controller returns to normal functional mode by clearing the low-power mode bit (L), or by POR. This bit must only be modified when the EMC is in idle state.[1]	0
		0	Normal mode (warm reset value).	
		1	Low-power mode.	
31:3	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] The external memory cannot be accessed in low-power or disabled state. If a memory access is performed an AHB error response is generated. The EMC registers can be programmed in low-power and/or disabled state.

21.7.2 EMC Status register

The read-only Status register provides EMC status information.

Table 346. EMC Status register (STATUS - address 0x4000 5004) bit description

Bit	Symbol	Value	Description	Reset value
0	B		Busy. This bit is used to ensure that the memory controller enters the low-power or disabled mode cleanly by determining if the memory controller is busy or not:	1
		0	EMC is idle (warm reset value).	
		1	EMC is busy performing memory transactions, commands, auto-refresh cycles, or is in self-refresh mode (POR reset value).	
1	S		Write buffer status. This bit enables the EMC to enter low-power mode or disabled mode cleanly:	0
		0	Write buffers empty (POR reset value)	
		1	Write buffers contain data.	
2	SA		Self-refresh acknowledge. This bit indicates the operating mode of the EMC:	1
		0	Normal mode	
		1	Self-refresh mode (POR reset value).	
31:3	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.3 EMC Configuration register

The Config register configures the operation of the memory controller. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This register is accessed with one wait state.

Table 347. EMC Configuration register (CONFIG - address 0x4000 5008) bit description

Bit	Symbol	Value	Description	Reset value
0	EM		Endian mode.	0
		0	Little-endian mode (POR reset value).	
		1	Big-endian mode.	
			On power-on reset, the value of the endian bit is 0. All data must be flushed in the EMC before switching between little-endian and big-endian modes.	
7:1	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
8	CR		Clock Ratio. CCLK: CLKOUT[1:0] ratio:	0
		0	1:1 (POR reset value)	
		1	1:2	
			This bit must contain 0 for proper operation of the EMC.	
31:9	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.4 Dynamic Memory Control register

The DynamicControl register controls dynamic memory operation. The control bits can be altered during normal operation.

Table 348. Dynamic Control register (DYNAMICCONTROL - address 0x4000 5020) bit description

Bit	Symbol	Value	Description	Reset value
0	CE		Dynamic memory clock enable.	0
		0	Clock enable of idle devices are deasserted to save power (POR reset value).	
		1	All clock enables are driven HIGH continuously.[1]	
1	CS		Dynamic memory clock control. When clock control is LOW the output clock CLKOUT is stopped when there are no SDRAM transactions. The clock is also stopped during self-refresh mode.	1
		0	CLKOUT stops when all SDRAMs are idle and during self-refresh mode.	
		1	CLKOUT runs continuously (POR reset value).	

Table 348. Dynamic Control register (DYNAMICCONTROL - address 0x4000 5020) bit description

Bit	Symbol	Value	Description	Reset value
2	SR		Self-refresh request, EMCSREFREQ. By writing 1 to this bit self-refresh can be entered under software control. Writing 0 to this bit returns the EMC to normal mode. The self-refresh acknowledge bit in the Status register must be polled to discover the current operating mode of the EMC.[2]	1
		0	Normal mode.	
		1	Enter self-refresh mode (POR reset value).	
4:3	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
5	MMC		Memory clock control.	0
		0	CLKOUT enabled (POR reset value).	
		1	CLKOUT disabled.[3]	
6	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
8:7	I		SDRAM initialization.	00
		0x0	Issue SDRAM NORMAL operation command (POR reset value).	
		0x1	Issue SDRAM MODE command.	
		0x2	Issue SDRAM PALL (precharge all) command.	
		0x3	Issue SDRAM NOP (no operation) command)	
12:9	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
13	DP		Low-power SDRAM deep-sleep mode.	0
		0	Normal operation (POR reset value).	
		1	Enter Deep-sleep mode.	
31:14	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] Clock enable must be HIGH during SDRAM initialization.

[2] The memory controller exits from power-on reset with the self-refresh bit HIGH. To enter normal functional mode set this bit LOW.

[3] Disabling CLKOUT can be performed if there are no SDRAM memory transactions. When enabled this bit can be used in conjunction with the dynamic memory clock control (CS) field.

Remark: Deep-sleep mode can be entered by setting the deep-sleep mode (DP) bit, the dynamic memory clock enable bit (CE), and the dynamic clock control bit (CS) to one. The device is then put into a low-power mode where the device is powered down and no longer refreshed. All data in the memory is lost.

21.7.5 Dynamic Memory Refresh Timer register

The DynamicRefresh register configures dynamic memory operation. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. However, these control bits can, if necessary, be altered during normal operation. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 349. Dynamic Memory Refresh Timer register (DYNAMICREFRESH - address 0x4000 5024) bit description

Bit	Symbol	Description	Reset value
10:0	REFRESH	Refresh timer. Indicates the multiple of 16 CCLKs between SDRAM refresh cycles. 0x0 = Refresh disabled (POR reset value). 0x1 - 0x7FF = n x16 = 16n CCLKs between SDRAM refresh cycles. For example: 0x1 = 1 x 16 = 16 CCLKs between SDRAM refresh cycles. 0x8 = 8 x 16 = 128 CCLKs between SDRAM refresh cycles	0
31:11	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

For example, for the refresh period of 16 µs, and a CCLK frequency of 50 MHz, the following value must be programmed into this register:

$(16 \times 10^{-6} \times 50 \times 10^{6}) / 16 = 50$ or 0x32

If auto-refresh through warm reset is requested (by setting the EMC_Reset_Disable bit), the timing of auto-refresh must be adjusted to allow a sufficient refresh rate when the clock rate is reduced during the wake-up period of a reset cycle. During this period, the EMC (and all other portions of the chip that are being clocked) run from the IRC oscillator at 12 MHz. The IRC oscillator frequency must be used as the CCLK rate for refresh calculations if auto-refresh through warm reset is requested.

Note: The refresh cycles are evenly distributed. However, there might be slight variations when the auto-refresh command is issued depending on the status of the memory controller.

21.7.6 Dynamic Memory Read Configuration register

The DynamicReadConfig register configures the dynamic memory read strategy. This register must be modified during system initialization with a bit value RD ≥ 1. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Remark: Choose command delay strategy (RD = 0x1) for SDRAM operation.

See Section 15.4.9 for programming the delay value for the EMC_CLKn delay.

Table 350. Dynamic Memory Read Configuration register (DYNAMICREADCONFIG - address 0x4000 5028) bit description

Bit	Symbol	Value	Description	Reset value
1:0	RD		Read data strategy.	0x0
		0x0	Do not use. POR reset value.	
		0x1	Command delayed by 1/2CCLK.	
		0x2	Command delayed by 1/2 CCLK plus one clock cycle.	
		0x3	Command delayed by1/2 CCLK plus two clock cycles,	
31:2	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.7 Dynamic Memory Precharge Command Period register

The DynamicTRP register enables you to program the precharge command period, tRP. This register must only be modified during system initialization. This value is normally found in SDRAM data sheets as tRP. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 351. Dynamic Memory Precharge Command Period register (DYNAMICRP - address 0x4000 5030) bit description

Bit	Symbol	Description	Reset value
3:0	tRP	Precharge command period. 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.8 Dynamic Memory Active to Precharge Command Period register

The DynamicTRAS register enables you to program the active to precharge command period, tRAS. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tRAS. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 352. Dynamic Memory Active to Precharge Command Period register (DYNAMICRAS - address 0x4000 5034) bit description

Bit	Symbol	Description	Reset value
3:0	tRAS	Active to precharge command period. 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.9 Dynamic Memory Self Refresh Exit Time register

The DynamicTSREX register enables you to program the self-refresh exit time, tSREX. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tSREX, for devices without this parameter you use the same value as tXSR. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 353. Dynamic Memory Self Refresh Exit Time register (DYNAMICSREX - address 0x4000 5038) bit description

Bit	Symbol	Description	Reset value
3:0	tSREX	Self-refresh exit time. 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.10 Dynamic Memory Last Data Out to Active Time register

The DynamicTAPR register enables you to program the last-data-out to active command time, tAPR. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tAPR. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 354. Dynamic Memory Last Data Out to Active Time register (DYNAMICAPR - address 0x4000 503C) bit description

Bit	Symbol	Description	Reset value
3:0	tAPR	Last-data-out to active command time. 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.11 Dynamic Memory Data In to Active Command Time register

The DynamicTDAL register enables you to program the data-in to active command time, tDAL. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tDAL, or tAPW. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 355. Dynamic Memory Data In to Active Command Time register (DYNAMICDAL - address 0x4000 5040) bit description

Bit	Symbol	Description	Reset value
3:0	tDAL	Data-in to active command. 0x0 - 0xE = n clock cycles. The delay is in CCLK cycles. 0xF = 15 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.12 Dynamic Memory Write Recovery Time register

The DynamicTWR register enables you to program the write recovery time, tWR. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tWR, tDPL, tRWL, or tRDL. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 356. Dynamic Memory Write Recovery Time register (DYNAMICWR - address 0x4000 5044) bit description

Bit	Symbol	Description	Reset value
3:0	tWR	Write recovery time. 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.13 Dynamic Memory Active to Active Command Period register

The DynamicTRC register enables you to program the active to active command period, tRC. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tRC. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 357. Dynamic Memory Active to Active Command Period register (DYNAMICRC - address 0x4000 5048) bit description

Bit	Symbol	Description	Reset value
4:0	tRC	Active to active command period. 0x0 - 0x1E = n + 1 clock cycles. The delay is in CCLK cycles. 0x1F = 32 clock cycles (POR reset value).	0x1F
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.14 Dynamic Memory Auto-refresh Period register

The DynamicTRFC register enables you to program the auto-refresh period, and auto-refresh to active command period, tRFC. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tRFC, or sometimes as tRC. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 358. Dynamic Memory Auto Refresh Period register (DYNAMICRFC - address 0x4000 504C) bit description

Bit	Symbol	Description	Reset value
4:0	tRFC	Auto-refresh period and auto-refresh to active command period. 0x0 - 0x1E = n + 1 clock cycles. The delay is in CCLK cycles. 0x1F = 32 clock cycles (POR reset value).	0x1F
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.15 Dynamic Memory Exit Self Refresh register

The DynamicTXSR register enables you to program the exit self-refresh to active command time, tXSR. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tXSR. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 359. Dynamic Memory Exit Self Refresh register (DYNAMICXSR - address 0x4000 5050) bit description

Bit	Symbol	Description	Reset value
4:0	tXSR	Exit self-refresh to active command time. 0x0 - 0x1E = n + 1 clock cycles. The delay is in CCLK cycles. 0x1F = 32 clock cycles (POR reset value).	0x1F
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.16 Dynamic Memory Active Bank A to Active Bank B Time register

The DynamicTRRD register enables you to program the active bank A to active bank B latency, tRRD. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tRRD. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 360. Dynamic Memory Active Bank A to Active Bank B Time register (DYNAMICRRD - address 0x4000 5054) bit description

Bit	Symbol	Description	Reset value
3:0	tRRD	Active bank A to active bank B latency 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.17 Dynamic Memory Load Mode register to Active Command Time

The DynamicTMRD register enables you to program the load mode register to active command time, tMRD. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This value is normally found in SDRAM data sheets as tMRD, or tRSA. This register is accessed with one wait state.

Note: This register is used for all four dynamic memory chip selects. Therefore the worst case value for all of the chip selects must be programmed.

Table 361. Dynamic Memory Load Mode register to Active Command Time (DYNAMICMRD - address 0x4000 5058) bit description

Bit	Symbol	Description	Reset value
3:0	tMRD	Load mode register to active command time. 0x0 - 0xE = n + 1 clock cycles. The delay is in CCLK cycles. 0xF = 16 clock cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.18 Static Memory Extended Wait register

ExtendedWait (EW) bit in the StaticConfig register is set. It is recommended that this register is modified during system initialization, or when there are no current or outstanding transactions. However, if necessary, these control bits can be altered during normal operation. This register is accessed with one wait state.

Table 362. Static Memory Extended Wait register (STATICEXTENDEDWAIT - address 0x4000 5080) bit description

Bit	Symbol	Description	Reset value
9:0	EXTENDEDWAIT	Extended wait time out. 16 clock cycles (POR reset value). The delay is in CCLK cycles. 0x0 = 16 clock cycles. 0x1 - 0x3FF = (n+1) x16 clock cycles.	0x0
31:10	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

For example, for a static memory read/write transfer time of 16 µs, and a CCLK frequency of 50 MHz, the following value must be programmed into this register:
$(16 \times 10^{-6} \times 50 \times 10^{6}) / 16 - 1 = 49$

21.7.19 Dynamic Memory Configuration registers

The DynamicConfig registers enable you to program the configuration information for the relevant dynamic memory chip select. These registers are normally only modified during system initialization. These registers are accessed with one wait state.

Table 363. Dynamic Memory Configuration registers (DYNAMICCONFIG, address 0x4000 5100 (DYNAMICCONFIG0), 0x4000 5120 (DYNAMICCONFIG1), 0x4000 5140 (DYNAMICCONFIG2), 0x4000 5160 (DYNAMICCONFIG3)) bit description

Bit	Symbol	Value	Description	Reset value
2:0	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
4:3	MD		Memory device.	00
		0x0	SDRAM (POR reset value).	
		0x1	Low-power SDRAM.	
		0x2	Reserved.	
		0x3	Reserved.	
6:5	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
12:7	AM0		Address mapping. See Table 364. 000000 = reset value.[1]	0
13	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
14	AM1		Address mapping See Table 364. 0 = reset value.	0
18:15	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
19	B		Buffer enable.	
		0	Buffer disabled for accesses to this chip select (POR reset value).	
		1	Buffer enabled for accesses to this chip select. After configuration of the dynamic memory, the buffer must be enabled for normal operation. [2]	
20	P		Write protect.	0
		0	Writes not protected (POR reset value).	
		1	Writes protected.	
31:21	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] The SDRAM column and row width and number of banks are computed automatically from the address mapping.

[2] The buffers must be disabled during SDRAM and SyncFlash initialization. They must also be disabled when performing SyncFlash commands. The buffers must be enabled during normal operation.

Address mappings that are not shown in Table 364 are reserved.

Table 364. Address mapping

14	12	11:9	8:7	Description
16 bit external bus high-performance address mapping (Row, Bank, Column)				
0	0	000	00	16 Mb (2Mx8), 2 banks, row length = 11, column length = 9
0	0	000	01	16 Mb (1Mx16), 2 banks, row length = 11, column length = 8
0	0	001	00	64 Mb (8Mx8), 4 banks, row length = 12, column length = 9
0	0	001	01	64 Mb (4Mx16), 4 banks, row length = 12, column length = 8
0	0	010	00	128 Mb (16Mx8), 4 banks, row length = 12, column length = 10
0	0	010	01	128 Mb (8Mx16), 4 banks, row length = 12, column length = 9
0	0	011	00	256 Mb (32Mx8), 4 banks, row length = 13, column length = 10
0	0	011	01	256 Mb (16Mx16), 4 banks, row length = 13, column length = 9
0	0	100	00	512 Mb (64Mx8), 4 banks, row length = 13, column length = 11
0	0	100	01	512 Mb (32Mx16), 4 banks, row length = 13, column length = 10
16 bit external bus low-power SDRAM address mapping (Bank, Row, Column)				
0	1	000	00	16 Mb (2Mx8), 2 banks, row length = 11, column length = 9
0	1	000	01	16 Mb (1Mx16), 2 banks, row length = 11, column length = 8
0	1	001	00	64 Mb (8Mx8), 4 banks, row length = 12, column length = 9
0	1	001	01	64 Mb (4Mx16), 4 banks, row length = 12, column length = 8
0	1	010	00	128 Mb (16Mx8), 4 banks, row length = 12, column length = 10
0	1	010	01	128 Mb (8Mx16), 4 banks, row length = 12, column length = 9
0	1	011	00	256 Mb (32Mx8), 4 banks, row length = 13, column length = 10
0	1	011	01	256 Mb (16Mx16), 4 banks, row length = 13, column length = 9
0	1	100	00	512 Mb (64Mx8), 4 banks, row length = 13, column length = 11
0	1	100	01	512 Mb (32Mx16), 4 banks, row length = 13, column length = 10
32 bit external bus high-performance address mapping (Row, Bank, Column)				
1	0	000	00	16 Mb (2Mx8), 2 banks, row length = 11, column length = 9
1	0	000	01	16 Mb (1Mx16), 2 banks, row length = 11, column length = 8
1	0	001	00	64 Mb (8Mx8), 4 banks, row length = 12, column length = 9
1	0	001	01	64 Mb (4Mx16), 4 banks, row length = 12, column length = 8
1	0	001	10	64 Mb (2Mx32), 4 banks, row length = 11, column length = 8
1	0	010	00	128 Mb (16Mx8), 4 banks, row length = 12, column length = 10
1	0	010	01	128 Mb (8Mx16), 4 banks, row length = 12, column length = 9
1	0	010	10	128 Mb (4Mx32), 4 banks, row length = 12, column length = 8
1	0	011	00	256 Mb (32Mx8), 4 banks, row length = 13, column length = 10
1	0	011	01	256 Mb (16Mx16), 4 banks, row length = 13, column length = 9
1	0	011	10	256 Mb (8Mx32), 4 banks, row length = 13, column length = 8
1	0	100	00	512 Mb (64Mx8), 4 banks, row length = 13, column length = 11
1	0	100	01	512 Mb (32Mx16), 4 banks, row length = 13, column length = 10
32 bit external bus low-power SDRAM address mapping (Bank, Row, Column)				
1	1	000	00	16 Mb (2Mx8), 2 banks, row length = 11, column length = 9
1	1	000	01	16 Mb (1Mx16), 2 banks, row length = 11, column length = 8
1	1	001	00	64 Mb (8Mx8), 4 banks, row length = 12, column length = 9
1	1	001	01	64 Mb (4Mx16), 4 banks, row length = 12, column length = 8

Table 364. Address mapping

14	12	11:9	8:7	Description
1	1	001	10	64 Mb (2Mx32), 4 banks, row length = 11, column length = 8
1	1	010	00	128 Mb (16Mx8), 4 banks, row length = 12, column length = 10
1	1	010	01	128 Mb (8Mx16), 4 banks, row length = 12, column length = 9
1	1	010	10	128 Mb (4Mx32), 4 banks, row length = 12, column length = 8
1	1	011	00	256 Mb (32Mx8), 4 banks, row length = 13, column length = 10
1	1	011	01	256 Mb (16Mx16), 4 banks, row length = 13, column length = 9
1	1	011	10	256 Mb (8Mx32), 4 banks, row length = 13, column length = 8
1	1	100	00	512 Mb (64Mx8), 4 banks, row length = 13, column length = 11
1	1	100	01	512 Mb (32Mx16), 4 banks, row length = 13, column length = 10

A chip select can be connected to a single memory device, in this case the chip select data bus width is the same as the device width. Alternatively the chip select can be connected to a number of external devices. In this case the chip select data bus width is the sum of the memory device data bus widths.

For example, for a chip select connected to:

- a 32-bit wide memory device, choose a 32-bit wide address mapping.
- a 16-bit wide memory device, choose a 16-bit wide address mapping.
- four x 8-bit wide memory devices, choose a 32-bit wide address mapping.
- two x 8-bit wide memory devices, choose a 16-bit wide address mapping.

The SDRAM bank select pins BA1 and BA0 are connected to address lines A14 and A13, respectively.

21.7.20 Dynamic Memory RAS & CAS Delay registers

The DynamicRasCas0:3 registers enable you to program the RAS and CAS latencies for the relevant dynamic memory. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. These registers are accessed with one wait state.

Note: The values programmed into these registers must be consistent with the values used to initialize the SDRAM memory device.

Table 365. Dynamic Memory RASCAS Delay registers (DYNAMICRASCAS, address 0x4000 5104 (DYNAMICRASCAS0), 0x4000 5124 (DYNAMICRASCAS1), 0x4000 5144 (DYNAMICRASCAS2), 0x4000 5164 (DYNAMICRASCAS3)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	RAS		RAS latency (active to read/write delay).	11
		0x0	Reserved.	
		0x1	One CCLK cycle.	
		0x2	Two CCLK cycles.	
		0x3	Three CCLK cycles (POR reset value).	

Table 365. Dynamic Memory RASCAS Delay registers (DYNAMICRASCAS, address 0x4000 5104 (DYNAMICRASCAS0), 0x4000 5124 (DYNAMICRASCAS1), 0x4000 5144 (DYNAMICRASCAS2), 0x4000 5164 (DYNAMICRASCAS3)) bit description

Bit	Symbol	Value	Description	Reset value
7:2	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
9:8	CAS		CAS latency.	11
		0x0	Reserved.	
		0x1	One CCLK cycle.	
		0x2	Two CCLK cycles.	
		0x3	Three CCLK cycles (POR reset value).	
31:10	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.21 Static Memory Configuration registers

The StaticConfig registers configure the static memory configuration. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. These registers are accessed with one wait state.

Table 366. Static Memory Configuration registers (STATICCONFIG, address 0x4000 5200 (STATICCONFIG0), 0x4000 5220 (STATICCONFIG1), 0x4000 5240 (STATICCONFIG2), 0x4000 5260 (STATICCONFIG3)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	MW		Memory width.	0
		0x0	8 bit (POR reset value).	
		0x1	16 bit.	
		0x2	32 bit.	
		0x3	Reserved.	
2	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
3	PM		Page mode. In page mode the EMC can burst up to four external accesses. Therefore devices with asynchronous page mode burst four or higher devices are supported. Asynchronous page mode burst two devices are not supported and must be accessed normally.	0
		0	Disabled (POR reset value).	
		1	Async page mode enabled (page length four).	
5:4	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
6	PC		Chip select polarity. The value of the chip select polarity on power-on reset is 0.	0
		0	Active LOW chip select.	
		1	Active HIGH chip select.	

Table 366. Static Memory Configuration registers (STATICCONFIG, address 0x4000 5200 (STATICCONFIG0), 0x4000 5220 (STATICCONFIG1), 0x4000 5240 (STATICCONFIG2), 0x4000 5260 (STATICCONFIG3)) bit description

Bit	Symbol	Value	Description	Reset value
7	PB		Byte lane state. The byte lane state bit, PB, enables different types of memory to be connected. For byte-wide static memories the BLSn[3:0] signal from the EMC is usually connected to $\overline{WE}$ (write enable). In this case for reads all the BLSn[3:0] bits must be HIGH. This means that the byte lane state (PB) bit must be LOW. 16 bit wide static memory devices usually have the BLSn[3:0] signals connected to the UBn and LBn (upper byte and lower byte) signals in the static memory. In this case a write to a particular byte must assert the appropriate UBn or LBn signal LOW. For reads, all the $\overline{UB}$ and $\overline{LB}$ signals must be asserted LOW so that the bus is driven. In this case the byte lane state (PB) bit must be HIGH. **Remark:** When PB is set to 0, the WE signal is undefined or 0. You must set PB to 1, to use the WE signal.	0
		0	For reads all the bits in BLSn[3:0] are HIGH. For writes the respective active bits in BLSn[3:0] are LOW (POR reset value).	
		1	For reads the respective active bits in BLSn[3:0] are LOW. For writes the respective active bits in BLSn[3:0] are LOW.	
8	EW		Extended wait. Extended wait (EW) uses the StaticExtendedWait register to time both the read and write transfers rather than the StaticWaitRd and StaticWaitWr registers. This enables much longer transactions.[1]	0
		0	Extended wait disabled (POR reset value).	
		1	Extended wait enabled.	
18:9	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
19	B		Buffer enable[2].	0
		0	Buffer disabled (POR reset value).	
		1	Buffer enabled.	
20	P		Write protect.	0
		0	Writes not protected (POR reset value).	
		1	Write protected.	
31:21	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] Extended wait and page mode cannot be selected simultaneously.

[2] EMC may perform burst read access even when the buffer enable bit is cleared.

21.7.22 Static Memory Write Enable Delay registers

The StaticWaitWen registers enable you to program the delay from the chip select to the write enable. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. These registers are accessed with one wait state.

Table 367. Static Memory Write Enable Delay registers (STATICWAITWEN, address 0x4000 5204 (STATICWAITWEN0), 0x4000 5224 (STATICWAITWEN1), 0x4000 5244 (STATICWAITWEN2), 0x4000 5264 (STATICWAITWEN3)) bit description

Bit	Symbol	Description	Reset value
3:0	WAITWEN	Wait write enable. Delay from chip select assertion to write enable. 0x0 = One CCLK cycle delay between assertion of chip select and write enable (POR reset value). 0x1 - 0xF = (n + 1) CCLK cycle delay. The delay is (WAITWEN +1) x tCCLK.	0x0
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.23 Static Memory Output Enable Delay registers

The StaticWaitOen registers enable you to program the delay from the chip select or address change, whichever is later, to the output enable. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. These registers are accessed with one wait state.

Table 368. Static Memory Output Enable delay registers (STATICWAITOEN, address 0x4000 5208 (STATICWAITOEN0), 0x4000 5228 (STATICWAITOEN1), 0x4000 5248 (STATICWAITOEN2), 0x4000 5268 (STATICWAITOEN3)) bit description

Bit	Symbol	Description	Reset value
3:0	WAITOEN	Wait output enable. Delay from chip select assertion to output enable. 0x0 = No delay (POR reset value). 0x1 - 0xF = n cycle delay. The delay is WAITOEN x tCCLK.	0x0
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.24 Static Memory Read Delay registers

The StaticWaitRd registers enable you to program the delay from the chip select to the read access. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. It is not used if the extended wait bit is enabled in the StaticConfig registers. These registers are accessed with one wait state.

Table 369. Static Memory Read Delay registers (STATICWAITRD, address 0x4000 520C (STATICWAITRD0), 0x4000 522C (STATICWAITRD1), 0x4000 524C (STATICWAITRD2), 0x4000 526C (STATICWAITRD3)) bit description

Bit	Symbol	Description	Reset value
4:0	WAITRD	Non-page mode read wait states or asynchronous page mode read first access wait state. Non-page mode read or asynchronous page mode read, first read only: 0x0 - 0x1E = (n + 1) CCLK cycles for read accesses. For non-sequential reads, the wait state time is (WAITRD + 1) x tCCLK. 0x1F = 32 CCLK cycles for read accesses (POR reset value).	0xB [1]
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] The reset value is 0x0B for the STATICWAITRD0 register only.

21.7.25 Static Memory Page Mode Read Delay registers

The StaticWaitPage registers enable you to program the delay for asynchronous page mode sequential accesses. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. This register is accessed with one wait state.

Table 370. Static Memory Page Mode Read Delay registers (STATICWAITPAGE, address 0x4000 5210 (STATICWAITPAGE0), 0x4000 5230 (STATICWAITPAGE1), 0x4000 5250 (STATICWAITPAGE2), 0x4000 5270 (STATICWAITPAGE3)) bit description

Bit	Symbol	Description	Reset value
4:0	WAITPAGE	Asynchronous page mode read after the first read wait states. Number of wait states for asynchronous page mode read accesses after the first read: 0x0 - 0x1E = (n+ 1) CCLK cycle read access time. For asynchronous page mode read for sequential reads, the wait state time for page mode accesses after the first read is (WAITPAGE + 1) x tCCLK. 0x1F = 32 CCLK cycle read access time (POR reset value).	0x1F
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.26 Static Memory Write Delay registers

The StaticWaitWr registers enable you to program the delay from the chip select to the write access. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode.These registers are not used if the extended wait (EW) bit is enabled in the StaticConfig register. These registers are accessed with one wait state.

Table 371. Static Memory Write Delay registers (STATICWAITWR, address 0x4000 5214 (STATICWAITWR0), 0x4000 5234 (STATICWAITWR1), 0x4000 5254 (STATICWAITWR2), 0x4000 5274 (STATICWAITWR3)) bit description

Bit	Symbol	Description	Reset value
4:0	WAITWR	Write wait states. SRAM wait state time for write accesses after the first read: 0x0 - 0x1E = (n + 2) CCLK cycle write access time. The wait state time for write accesses after the first read is WAITWR (n + 2) x tCCLK. 0x1F = 33 CCLK cycle write access time (POR reset value).	0x1F
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

21.7.27 Static Memory Turn Round Delay registers

The StaticWaitTurn registers enable you to program the number of bus turnaround cycles. It is recommended that these registers are modified during system initialization, or when there are no current or outstanding transactions. This can be ensured by waiting until the EMC is idle, and then entering low-power, or disabled mode. These registers are accessed with one wait state.

Table 372. Static Memory Turn Round Delay registers (STATICWAITTURN, address 0x4000 5218 (STATICWAITTURN0), 0x4000 5238 (STATICWAITTURN1), 0x4000 5258 (STATICWAITTURN2), 0x4000 5278 (STATICWAITTURN3)) bit description

Bit	Symbol	Description	Reset value
3:0	WAITTURN	Bus turnaround cycles. 0x0 - 0xE = (n + 1) CCLK turnaround cycles. Bus turnaround time is (WAITTURN + 1) x tCCLK. 0xF = 16 CCLK turnaround cycles (POR reset value).	0xF
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

To prevent bus contention on the external memory data bus, the WAITTURN field controls the number of bus turnaround cycles added between static memory read and write accesses. The WAITTURN field also controls the number of turnaround cycles between static memory and dynamic memory accesses.

21.8 Functional description

Figure 43 shows a block diagram of the EMC.

The functions of the EMC blocks are described in the following sections:

- AHB slave register interface.
- AHB slave memory interfaces.
- Data buffers.
- Memory controller state machine.
- Pad interface.

Note: For 32 bit wide chip selects data is transferred to and from dynamic memory in SDRAM bursts of four. For 16 bit wide chip selects SDRAM bursts of eight are used.

21.8.1 AHB slave register interface

The AHB slave register interface block enables the registers of the EMC to be programmed. This module also contains most of the registers and performs the majority of the register address decoding.

To eliminate the possibility of endianness problems, all data transfers to and from the registers of the EMC must be 32 bits wide.

Note: If an access is attempted with a size other than a word (32 bits), it causes an ERROR response to the AHB bus and the transfer is terminated.

21.8.2 AHB slave memory interface

The AHB slave memory interface allows access to external memories.

21.8.2.1 Memory transaction endianness

The endianness of the data transfers to and from the external memories is determined by the Endian mode (N) bit in the Config register.

Note: The memory controller must be idle (see the busy field of the Status Register) before endianness is changed, so that the data is transferred correctly.

21.8.2.2 Memory transaction size

Memory transactions can be 8, 16, or 32 bits wide. Any access attempted with a size greater than a word (32 bits) causes an ERROR response to the AHB bus and the transfer is terminated.

21.8.2.3 Write protected memory areas

Write transactions to write-protected memory areas generate an ERROR response to the AHB bus and the transfer is terminated.

21.8.3 Pad interface

The pad interface block provides the interface to the pads.

The EMC dynamic memory interface requires that all EMC_CLK signals are selected on the CLKn pins for 16-bit memory and for 32-bit memory.

For static memory larger delays are defined by in steps of one EMC clock cycle by the STATICWAIT registers (see Section 21.7.22 to Section 21.7.27).

21.8.4 Data buffers

The AHB interface reads and writes via buffers to improve memory bandwidth and reduce transaction latency. The EMC contains four 16-word buffers. The buffers can be used as read buffers, write buffers, or a combination of both. The buffers are allocated automatically.

The buffers must be disabled during SDRAM and SyncFlash initialization. They must also be disabled when performing SyncFlash commands. The buffers must be enabled during normal operation.

The buffers can be enabled or disabled for static memory using the StaticConfig Registers.

21.8.4.1 Write buffers

Write buffers are used to:

- Merge write transactions so that the number of external transactions are minimized.

  Buffer data until the EMC can complete the write transaction, improving AHB write latency.

  Convert all dynamic memory write transactions into quadword bursts on the external memory interface. This enhances transfer efficiency for dynamic memory.
- Reduce external memory traffic. This improves memory bandwidth and reduces power consumption.

Write buffer operation:

- If the buffers are enabled, an AHB write operation writes into the Least Recently Used (LRU) buffer, if empty.

  If the LRU buffer is not empty, the contents of the buffer are flushed to memory to make space for the AHB write data.
- If a buffer contains write data it is marked as dirty, and its contents are written to memory before the buffer can be reallocated.

The write buffers are flushed whenever:

- The memory controller state machine is not busy performing accesses to external memory.

  The memory controller state machine is not busy performing accesses to external memory, and an AHB interface is writing to a different buffer.

Note: For dynamic memory, the smallest buffer flush is a quadword of data. For static memory, the smallest buffer flush is a byte of data.

21.8.4.2 Read buffers

Read buffers are used to:

- Buffer read requests from memory. Future read requests that hit the buffer read the data from the buffer rather than memory, reducing transaction latency.

  Convert all read transactions into quadword bursts on the external memory interface. This enhances transfer efficiency for dynamic memory.
- Reduce external memory traffic. This improves memory bandwidth and reduces power consumption.

Read buffer operation:

- If the buffers are enabled and the read data is contained in one of the buffers, the read data is provided directly from the buffer.
- If the read data is not contained in a buffer, the LRU buffer is selected. If the buffer is dirty (contains write data), the write data is flushed to memory. When an empty buffer is available the read command is posted to the memory.

A buffer filled by performing a read from memory is marked as not-dirty (not containing write data) and its contents are not flushed back to the memory controller unless a subsequent AHB transfer performs a write that hits the buffer.

21.9 Low-power operation

In many systems, the contents of the memory system have to be maintained during low-power sleep modes. The EMC provides a mechanism to place the dynamic memories into self-refresh mode.

Self-refresh mode can be entered by software by setting the SREFREQ bit in the DynamicControl Register and polling the SREFACK bit in the Status Register.

Any transactions to memory that are generated while the memory controller is in self-refresh mode are rejected and an error response is generated to the AHB bus. Clearing the SREFREQ bit in the DynamicControl Register returns the memory to normal operation. See the memory data sheet for refresh requirements.

Note: The static memory can be accessed as normal when the SDRAM memory is in self-refresh mode.

21.9.1 Low-power SDRAM Deep-sleep Mode

The EMC supports JEDEC low-power SDRAM deep-sleep mode. Deep-sleep mode can be entered by setting the deep-sleep mode (DP) bit, the dynamic memory clock enable bit (CE), and the dynamic clock control bit (CS) in the DynamicControl register. The device is then put into a low-power mode where the device is powered down and no longer refreshed. All data in the memory is lost.

21.9.2 Low-power SDRAM partial array refresh

The EMC supports JEDEC low-power SDRAM partial array refresh. Partial array refresh can be programmed by initializing the SDRAM memory device appropriately. When the memory device is put into self-refresh mode only the memory banks specified are refreshed. The memory banks that are not refreshed lose their data contents.

21.10 External static memory interface

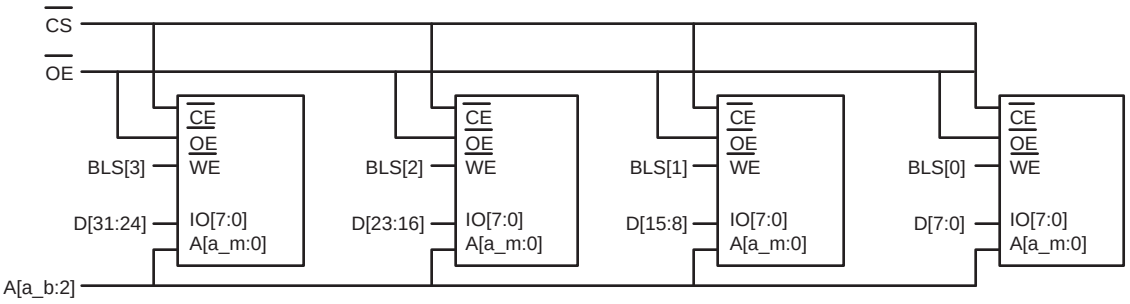
External memory interfacing depends on the bank width (32, 16 or 8 bit selected via MW bits in corresponding StaticConfig register).

If a memory bank is configured to be 32 bits wide, address lines A0 and A1 can be used as non-address lines. If a memory bank is configured to 16 bits wide, A0 is not required. However, 8 bit wide memory banks do require all address lines down to A0. Configuring A1 and/or A0 lines to provide address or non-address function is accomplished using the SYSCON registers.

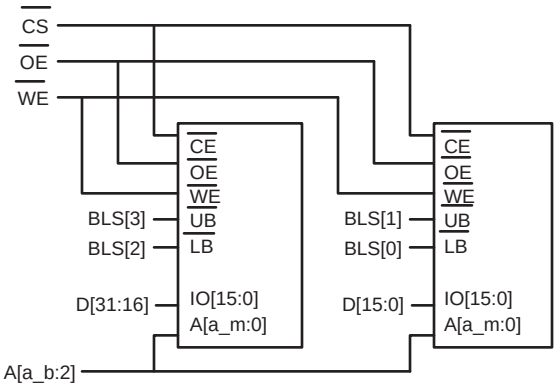
Symbol a_b in the following figures refers to the highest order address line in the data bus. Symbol a_m refers to the highest order address line of the memory chip used in the external memory interface.

If the external memory is used as external boot memory for flashless devices, refer to Section 5.2 on how to connect the EMC. The memory bank width for memory banks 1 and 2 is determined by the setting of the BOOT pins.

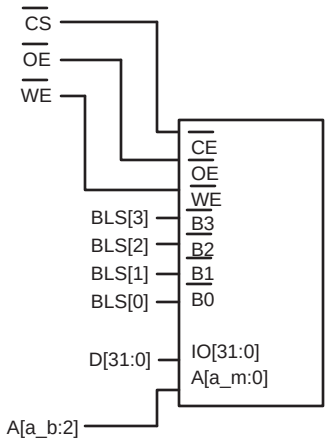
21.10.1 32-bit wide memory bank connection



a. 32 bit wide memory bank interfaced to four 8 bit memory chips



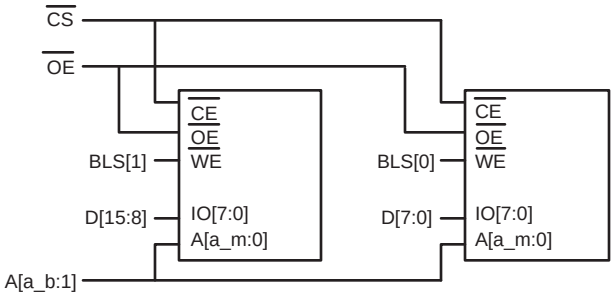
b. 32 bit wide memory bank interfaced to two 16 bit memory chips



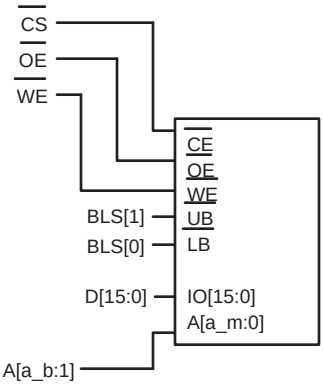
c. 32 bit wide memory bank interfaced to one 8 bit memory chip

Fig 45. 32 bit bank external memory interfaces (bits MW = 10)

21.10.2 16-bit wide memory bank connection



a. 16 bit wide memory bank interfaced to two 8 bit memory chips



b. 16 bit wide memory bank interfaced to a 16 bit memory chip

Fig 46. 16 bit bank external memory interfaces (bits MW = 01)

21.10.3 8-bit wide memory bank connection

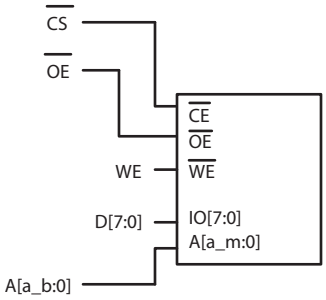


Fig 47. 8 bit bank external memory interface (bits MW = 00)

22.1 How to read this chapter

The SPIFI is available on all LPC43xx parts.

22.2 Basic configuration

The SPIFI is configured as follows:

- See Table 373 for clocking and power control.
- The SPIFI is reset by the SPIFI_RST (reset # 53).

Table 373. SPIFI clocking and power control

	Base clock	Branch clock	Operating frequency
SPIFI AHB register clock	BASE_M4_CLK	CLK_M4_SPIFI	up to 204 MHz
SPIFI serial clock input	BASE_SPIFI_CLK	SPIFI_CLK	120 MHz

Remark: All parts can use the SPIFI for booting. See Section 5.3.4.5.

22.3 Features

- Quad SPI Flash Interface (SPIFI) interface to external flash.
- Transfer rates of up to SPIFI_CLK/2 bytes per second.
- External flash is directly memory mapped for fast access.
- Supports 1-, 2-, and 4-bit bi-directional serial protocols.
- Half-duplex protocol compatible with various vendors and devices .
- The SPIFI memory is accessible by the GPDMA.
- Software driver library available on the LPCware website.

22.4 General description

The SPI Flash Interface (SPIFI) allows low-cost serial flash memories to be connected to the Cortex-M4 processor with little performance penalty compared to parallel flash devices with higher pin count.

Many serial flash devices use a half-duplex command-driven SPI protocol for device setup and initialization. Quad devices then use a half-duplex, command-driven 4-bit protocol for normal operation. Different serial flash vendors and devices accept or require different commands and command formats. SPIFI provides sufficient flexibility to be compatible with common flash devices, and includes extensions to help insure compatibility with future devices.

Serial flash devices respond to commands sent by software or automatically sent by the SPIFI when software reads either of the two read-only serial flash regions in the memory map (see Table 374).

Table 374. SPIFI flash memory map

Memory	Address
SPIFI data	0x1400 0000 to 0x17FF FFFF (Use this memory area for debugging code). 0x8000 0000 to 0x87FF FFFF (Debug will not work if the program counter is in this memory area). **Remark:** These are the spaces allocated to the SPIFI in the LPC43xx. The same data appears in the first area and the first half of the second area. These areas allow maxima of 64 MB and 128 MB of SPI flash (respectively) to be mapped into the Cortex-M4 memory space. In practice, the usable space is limited to the size of the connected device

22.5 Pin description

Table 375. SPIFI Pin description

Pin	Direction	Description
SPIFI_SCK	O	Serial clock for the flash memory, switched only during active bits on the MOSI/IO0, MISO/IO1, and IO3:2 lines.
$\overline{\text{SPIFI\_CS}}$	O	Chip select for the flash memory, driven low while a command is in progress, and high between commands. In the typical case of one serial slave, this signal can be connected directly to the device. If more than one serial slave is connected, software and off-chip hardware should use general-purpose I/O signals in combination with this signal to generate the chip selects for the various slaves.
SPIFI_MOSI or IO0	I/O	This is an output except in quad/dual input data fields. After a quad/dual input data field, it becomes an output again one serial clock period after $\overline{\text{CS}}$ goes high.
SPIFI_MISO or IO1	I/O	This is an output in quad/dual opcode, address, intermediate, and output data fields, and an input in SPI mode and in quad/dual input data fields. After an input data field in quad/dual mode, it becomes an output again one serial clock period after $\overline{\text{CS}}$ goes high.
SPIFI_SIO[3:2]	I/O	These are outputs in quad opcode, address, intermediate, and output data fields, and inputs in quad input data fields. If the flash memory does not have quad capability, these pins can be assigned to GPIO or other functions.

22.6 Supported QSPI devices

Table 224 shows a list of vendor QSPI devices which are verified to support the SPIFI API. Other devices can be used and will run in basic single SPI mode at lower speed.

Remark: All QSPI devices have been tested at an operating voltage of 3.3 V.

Table 376. Supported QSPI devices

Manufacturer	Device name
AMIC	A25L512, A25L010, A25L020, A25L040, A25L080, A25L016, A25L032, A25LQ032
Atmel	AT25F512B, AT25DF021, AT25DF041A, AT25DF081A, AT25DF161, AT25DQ161, AT25DF321A, AT25DF641
Chingis	Pm25LD256, Pm25LD512, Pm25LD010, Pm25LD020, Pm25LD040, Pm25LQ032
Elite (ESMT)	F25L08P, F25L16P, F25L32P, F25L32Q
Eon	EN25F10, EN25F20, EN25F40, EN25Q40, EN25F80, EN25Q80, EN25QH16, EN25Q32, EN25Q64, EN25Q128
Gigadevice	GD25Q512, GD25Q10, GD25Q20, GD25Q40, GD25Q80, GD25Q16, GD25Q32, GD25Q64
Macronix	MX25L8006, MX25L8035, MX25L8036, MX25U8035[1], MX25L1606, MX25L1633, MX25L1635, MX25L1636, MX25U1635[1], MX25L3206, MX25L3235, MX25L3236, MX25U3235[1], MX25L6436, MX25L6445, MX25L6465, MX25L12836, MX25L12845, MX25L12865, MX25L25635, MX25L25735
Numonyx	M25P10, M25P20, M25P40, M25P80, M25PX80, M25P16, M25PX16, M25P32, M25PX32, M25P64, M25PX64, N25Q032, N25Q064, N25Q128
Spansion	S25FL004K, S25FL008K, S25FL016K, S25FL032K, S25FL032P, S25FL064K, S25FL064P, S25FL129P
SST	SST26VF016, SST26VF032, SST25VF064
Winbond	W25Q40, W25Q80, W25Q16, W25Q32, W25Q64

[1] Level translation circuitry, which might affect performance, is required for these parts.

23.1 How to read this chapter

The USB0 Host/Device/OTG controller is available on parts LPC4350, LPC4330, and LPC4320.

23.2 Basic configuration

The USB0 Host/Device/OTG controller is configured as follows:

- See Table 377 for clocking and power control.
- The USB0 is reset by the USB0_RST (reset # 17).
- The USB0 OTG interrupt is connected to interrupt slot # 8 in the NVIC, and the wake-up request indicator is connected to slot # 9 in the Event router.
- Power to the USB0 on-chip PHY is controlled through the CREG block (see Table 38). The on-chip PHY is powered down by default unless the USB0 boot mode is selected. To use the USB0 controller, enable the PHY in the CREG0 register, bit 5.
- The SOF/VF indicator can be connected to Timer3 or the to SCT through the GIMA (see Section 23.7.7 and Table 138).

Table 377. USB0 clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
USB0 clock	BASE_USB0_CLK	CLK_USB0	480 MHz	Uses PLL0USB dedicated to USB0. CLK_USB0 must be set to the 480 MHz clock in all USB modes (low-speed, full-speed, and high-speed modes).
USB0 register interface clock	BASE_M4_CLK	CLK_M4_USB0	up to 204 MHz	

23.3 Features

- Contains on-chip high-speed UTMI+ compliant transceiver (PHY).
- Supports all high-speed, full-speed, and low-speed USB-compliant peripherals.
- Complies with Universal Serial Bus specification 2.0.
- Complies with USB On-The-Go supplement.
- Complies with Enhanced Host Controller Interface (EHCI) Specification.
- Supports auto USB 2.0 mode discovery.
- Supports software HNP and SRP for OTG peripherals.
- Supports power management.
- Supports six logical endpoints including one control endpoint for a total of 12 physical endpoints.

- This module has its own, integrated DMA engine.
- Can be used together with the audio PLL for USB streaming applications.

23.4 Introduction

Universal Serial Bus (USB) is a standard protocol developed to connect several types of devices to each other in order to exchange data or for other purposes. Many portable devices can benefit from the ability to communicate to each other over the USB interface without intervention of a host PC. The addition of the On-The-Go functionality to USB makes this possible without losing the benefits of the standard USB protocol. Examples of USB devices are: PC, mouse, keyboard, MP3 player, digital camera, USB storage device (USB stick).

23.4.1 Block diagram

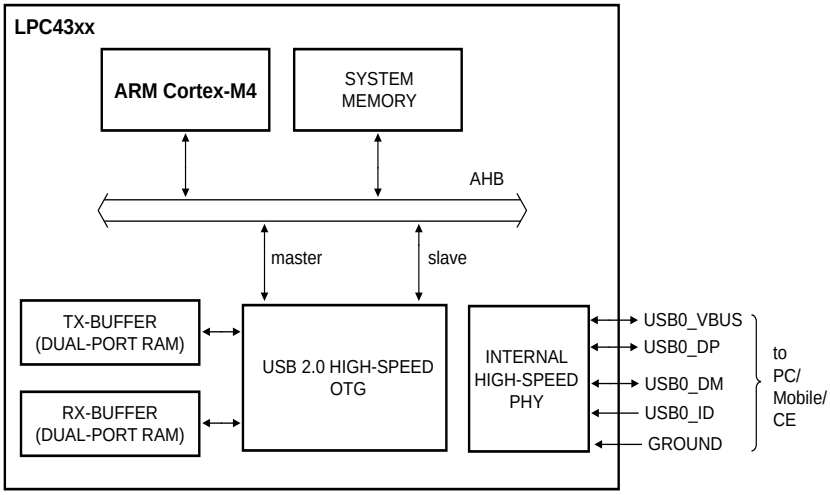


Fig 48. High-speed USB OTG block diagram

23.4.2 About USB On-The-Go

The USB On-The-Go block enables usage in both device mode and in host mode. This means that you can connect to a PC to exchange data, but also to another USB device such as a digital camera or MP3 player.

23.4.3 USB acronyms and abbreviations

Table 378. USB related acronyms

Acronym	Description
ATX	Analog Transceiver
DCD	Device Controller Driver
dQH	device Endpoint Queue Head
dTD	device Transfer Descriptor
EOP	End Of Packet

Table 378. USB related acronyms

Acronym	Description
EP	End Point
FS	Full Speed
HCD	Host Controller Driver
HS	High Speed
LS	Low Speed
MPS	Maximum Packet Size
NAK	Negative Acknowledge
OTG	On-The-Go
PID	Packet Identifier
QH	Queue Head
SE0	Single Ended 0
SOF	Start Of Frame
TT	Transaction Translator
USB	Universal Serial Bus

23.4.4 Transmit and receive buffers

The USB OTG controller contains a Transmit buffer to store data to be transmitted on the USB and an Receive buffer to store data received from the USB. The Receive buffer contains 256 words. The Transmit buffer contains 128 words for each endpoint in device mode and 512 words in host mode.

23.4.5 Fixed endpoint configuration

Table 379 shows the supported endpoint configurations. The Maximum Packet Size (see Table 380) is dependent on the type of endpoint and the device configuration (low-speed, full-speed, or high-speed).

Table 379. Fixed endpoint configuration

Logical endpoint	Physical endpoint	Endpoint type	Direction
0	0	Control	Out
0	1	Control	In
1	2	Interrupt/Bulk/Isochronous	Out
1	3	Interrupt/Bulk/Isochronous	In
2	4	Interrupt/Bulk/Isochronous	Out
2	5	Interrupt/Bulk/Isochronous	In
3	6	Interrupt/Bulk/Isochronous	Out
3	7	Interrupt/Bulk/Isochronous	In
4	8	Interrupt/Bulk/Isochronous	Out
4	9	Interrupt/Bulk/Isochronous	In
5	10	Interrupt/Bulk/Isochronous	Out
5	11	Interrupt/Bulk/Isochronous	In

Table 380. USB Packet size

Endpoint type	Speed	Packet size (byte)
Control	Low-speed	8
	Full-speed	8, 16, 32, or 64
	High-speed	64
Isochronous	Low-speed	n/a
	Full-speed	up to 1023
	High-speed	up to 1024
Interrupt	Low-speed	up to 8
	Full-speed	up to 64
	High-speed	up to 1024
Bulk	Low-speed	n/a
	Full-speed	8, 16, 32, or 64
	High-speed	8, 16, 32, 64, or 512

23.5 Pin description

Table 381. USB0 pin description

Pin function	Direction	Description
USB0_IND0	O	Port indicator LED control output.
USB0_IND1	O	Port indicator LED control output.
USB0_PWR_FAULT	O	Port power fault signal indicating overcurrent condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
USB0_PPWR	O	VBUS drive signal (towards external charge pump or power management unit); indicates that VBUS must be driven (active HIGH). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the USB_PPWR used on other NXP LPC parts.
USB0_DP	I/O	USB0 bidirectional D+ line. The D+ line has an internal 1.5 kΩ pull-up. This pull-up is enabled when software sets the RS bit (Bit 0) in the USBCMD register and the USB0 controller sees a valid VBUS voltage level (above ~1.8V) on the VBUS pin.
USB0_DM	I/O	USB0 bidirectional D− line. The D- line has an internal 1.5 kΩ pull-up. This pull-up is enabled when software sets RS bit (BIT_0) in the USBCMD register and the USB0 controller sees a valid VBUS voltage level (above ~1.8V) on the VBUS pin.
USB0_VBUS	I	VBUS pin (power on USB cable). This pin includes an internal pull-down resistor of 64 kOhm (typical) ± 16 kOhm. For maximum load C_L = 6.5 uF and maximum pull-down resistor R_{pd} = 80 kOhm, the VBUS signal takes about 2 s to fall from VBUS = 5 V to VBUS = 0.2 V when it is no longer driven.
USB0_ID	I	Indicates to the transceiver whether connected as an A-device (USB0_ID LOW) or B-device (USB0_ID HIGH). For OTG this pin has an internal pull-up resistor.
USB0_RREF		12.0 kΩ (accuracy 1%) on-board resistor to ground for current reference;
USB0_VDDA3V3_DRIVER		Separate analog 3.3 Vpower supply for driver.

Table 381. USB0 pin description

Pin function	Direction	Description
USB0_VDDA3V3		USB 3.3 V separate power supply voltage
USB0_VSSA_TERM		Dedicated analog ground for clean reference for termination resistors.
USB0_VSSA_REF		Dedicated clean analog ground for generation of reference currents and voltages.

23.6 Register description

Table 382. Register access abbreviations

Abbreviation	Description
R/W	Read/Write
R/WC	Read/Write one to Clear
R/WO	Read/Write Once
RO	Read Only
WO	Write Only

Table 383. Register overview: USB0 OTG controller (register base address 0x4000 6000)

Name	Access	Address offset	Description	Reset value	Reset value after USB0 boot	Reference
-	-	0x000 - 0x0FF	Reserved	-	-	
Device/host capability registers						
CAPLENGTH	RO	0x100	Capability register length	0x0100 0040	0x0100 0040	Table 384
HCSPARAMS	RO	0x104	Host controller structural parameters	0x0001 0011	0x0001 0011	Table 385
HCCPARAMS	RO	0x108	Host controller capability parameters	0x0000 0006	0x0000 0006	Table 386
DCIVERSION	RO	0x120	Device interface version number	0x0000 0001	0x0000 0001	Table 387
DCCPARAMS	RO	0x124	Device controller capability parameters	0x0000 0186	0x0000 0186	Table 388
-	-	0x128 - 0x13C	Reserved			-
Device/host operational registers						
USBCMD_D	R/W	0x140	USB command (device mode)	0x0008 0000	0	Table 389
USBCMD_H	R/W	0x140	USB command (host mode)	0x0008 0000	-	Table 390
USBSTS_D	R/W	0x144	USB status (device mode)	0	0x0000 0080	Table 392
USBSTS_H	R/W	0x144	USB status (host mode)	0	-	Table 393
USBINTR_D	R/W	0x148	USB interrupt enable (device mode)	0	0x0001 0147	Table 394
USBINTR_H	R/W	0x148	USB interrupt enable (host mode)	0	-	Table 395

Table 383. Register overview: USB0 OTG controller (register base address 0x4000 6000) *…continued*

Name	Access	Address offset	Description	Reset value	Reset value after USB0 boot	Reference
FRINDEX_D	R/W	0x14C	USB frame index (device mode)	0	0x0000 1F20	Table 396
FRINDEX_H	R/W	0x14C	USB frame index (host mode)	0	-	Table 397
-	-	0x150	Reserved			-
DEVICEADDR	R/W	0x154	USB device address (device mode)	0	0x0400 0000	Table 399
PERIODICLISTBASE	R/W	0x154	Frame list base address (host mode)	0	-	Table 400
ENDPOINTLISTADDR	R/W	0x158	Address of endpoint list in memory	0	0x2000 0000	Table 401
ASYNCLISTADDR	R/W	0x158	Asynchronous list address	0	0	Table 402
TTCTRL	R/W	0x15C	Asynchronous buffer status for embedded TT (host mode)	0	0	Table 403
BURSTSIZE	R/W	0x160	Programmable burst size	0	0	Table 404
TXFILLTUNING	R/W	0x164	Host transmit pre-buffer packet tuning (host mode)	0	0	Table 405
-	-	0x168 - 0x170	Reserved	-	-	
BINTERVAL	R/W	0x174	Length of virtual frame	0	0	Table 406
ENDPTNAK	R/W	0x178	Endpoint NAK (device mode)	0	0	Table 407
ENDPTNAKEN	R/W	0x17C	Endpoint NAK Enable (device mode)	0	0	Table 408
-	-	0x180	Reserved	-	-	
PORTSC1_D	R/W	0x184	Port 1 status/control (device mode)	0	0x3C00 0004	Table 409
PORTSC1_H	R/W	0x184	Port 1 status/control (host mode)	0	-	Table 410
-	-	0x188 - 0x1A0		-	-	-
OTGSC	R/W	0x1A4	OTG status and control	0	0x0020 0D09	Table 412
USBMODE_D	R/W	0x1A8	USB device mode (device mode)	0	0xA	Table 413
USBMODE_H	R/W	0x1A8	USB device mode (host mode)	0	-	Table 414
Device endpoint registers						
ENDPTSETUPSTAT	R/W	0x1AC	Endpoint setup status	0	0	Table 415
ENDPTPRIME	R/W	0x1B0	Endpoint initialization	0	0	Table 416
ENDPTFLUSH	R/W	0x1B4	Endpoint de-initialization	0	0	Table 417
ENDPTSTAT	RO	0x1B8	Endpoint status	0	0	Table 418
ENDPTCOMPLETE	R/W	0x1BC	Endpoint complete	0	0	Table 419
ENDPTCTRL0	R/W	0x1C0	Endpoint control 0	0	0	Table 420
ENDPTCTRL1	R/W	0x1C4	Endpoint control 1	0	0	Table 421
ENDPTCTRL2	R/W	0x1C8	Endpoint control 2	0	0	Table 421
ENDPTCTRL3	R/W	0x1CC	Endpoint control 3	0	0	Table 421
ENDPTCTRL4	R/W	0x1D0	Endpoint control 4	0	0	Table 421
ENDPTCTRL5	R/W	0x1D4	Endpoint control 5	0	0	Table 421

23.6.1 Use of registers

The register interface has bit functions described for device mode and bit functions described for host mode. However, during OTG operations it is necessary to perform tasks independent of the controller mode.

The only way to transition the controller mode out of host or device mode is by setting the controller reset bit. Therefore, it is also necessary for the OTG tasks to be performed independently of a controller reset as well as independently of the controller mode.

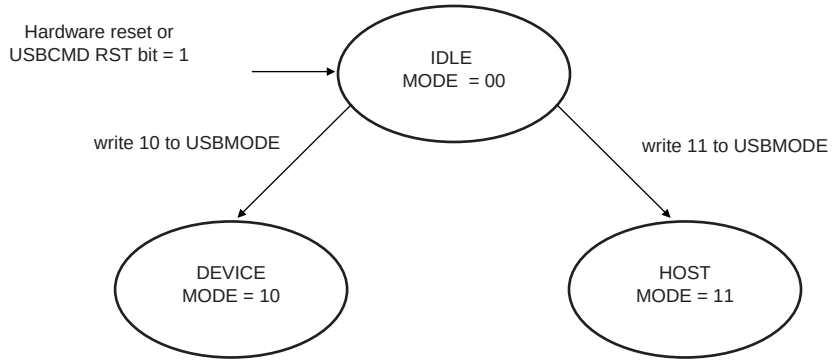


Fig 49. USB controller modes

The following registers and register bits are used for OTG operations. The values of these register bits are independent of the controller mode and are not affected by a write to the RESET bit in the USBCMD register.

- All identification registers
- All device/host capabilities registers
- All bits of the OTGSC register (Section 23.6.16)
- The following bits of the PORTSC register (Section 23.6.15):
  - PTS (parallel interface select)
  - STS (serial transceiver select)
  - PTW (parallel transceiver width)
  - PHCD (PHY low power suspend)
  - WKOC, WKDC, WKCN (wake signals)
  - PIC[1:0] (port indicators)
  - PP (port power)

23.6.2 Device/host capability registers

Table 384. CAPLENGTH register (CAPLENGTH - address 0x4000 6100) bit description

Bit	Symbol	Description	Reset value	Access
7:0	CAPLENGTH	Indicates offset to add to the register base address at the beginning of the Operational Register	0x40	RO
23:8	HCIVERSION	BCD encoding of the EHCI revision number supported by this host controller.	0x100	RO
31:24	-	These bits are reserved and should be set to zero.	-	-

Table 385. HCSPARAMS register (HCSPARAMS - address 0x4000 6104) bit description

Bit	Symbol	Description	Reset value	Access
3:0	N_PORTS	Number of downstream ports. This field specifies the number of physical downstream ports implemented on this host controller.	0x1	RO
4	PPC	Port Power Control. This field indicates whether the host controller implementation includes port power control.	0x1	RO
7:5	-	These bits are reserved and should be set to zero.	-	-
11:8	N_PCC	Number of Ports per Companion Controller. This field indicates the number of ports supported per internal Companion Controller.	0x0	RO
15:12	N_CC	Number of Companion Controller. This field indicates the number of companion controllers associated with this USB2.0 host controller.	0x0	RO
16	PI	Port indicators. This bit indicates whether the ports support port indicator control.	0x1	RO
19:17	-	These bits are reserved and should be set to zero.	-	-
23:20	N_PTT	Number of Ports per Transaction Translator. This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller.	0x0	RO
27:24	N_TT	Number of Transaction Translators. This field indicates the number of embedded transaction translators associated with the USB2.0 host controller.	0x0	RO
31:28	-	These bits are reserved and should be set to zero.	-	-

Table 386. HCCPARAMS register (HCCPARAMS - address 0x4000 6108) bit description

Bit	Symbol	Description	Reset value	Access
0	ADC	64-bit Addressing Capability. If zero, no 64-bit addressing capability is supported.	0	RO
1	PFL	Programmable Frame List Flag. If set to one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-boundary. This requirement ensures that the frame list is always physically contiguous.	1	RO
2	ASP	Asynchronous Schedule Park Capability. If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule.The feature can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the USBCMD register.	1	RO
7:4	IST	Isochronous Scheduling Threshold. This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule.	0	RO
15:8	EECP	EHCI Extended Capabilities Pointer. This optional field indicates the existence of a capabilities list.	0	RO
31:16	-	These bits are reserved and should be set to zero.	-	-

Table 387. DCIVERSION register (DCIVERSION - address 0x4000 6120) bit description

Bit	Symbol	Description	Reset value	Access
15:0	DCIVERSION	The device controller interface conforms to the two-byte BCD encoding of the interface version number contained in this register.	0x1	RO

Table 388. DCCPARAMS (address 0x4000 6124)

Bit	Symbol	Description	Reset value	Access
4:0	DEN	Device Endpoint Number.	0x4	RO
6:5	-	These bits are reserved and should be set to zero.	-	-
7	DC	Device Capable.	0x1	RO
8	HC	Host Capable.	0x1	RO
31:9	-	These bits are reserved and should be set to zero.	-	-

23.6.3 USB Command register (USBCMD)

The host/device controller executes the command indicated in this register.

23.6.3.1 Device mode

Table 389. USB Command register in device mode (USBCMD_D - address 0x4000 6140) bit description

Bit	Symbol	Value	Description	Access	Reset value
0	RS		Run/Stop	R/W	0
		0	Writing a 0 to this bit will cause a detach event.		
		1	Writing a one to this bit will cause the device controller to enable a pull-up on USB_DP and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the device controller has been properly initialized.		
1	RST		Controller reset. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register.	R/W	0
		0	Set to 0 by hardware when the reset process is complete.		
		1	When software writes a one to this bit, the Device Controller resets its internal pipelines, timers, counters, state machines etc. to their initial values. Writing a one to this bit when the device is in the attached state is not recommended, since the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.		
3:2	-	-	Not used in device mode.	-	0
4	-	-	Not used in device mode.	-	0
5	-	-	Not used in device mode.	-	0
6	-	-	Not used in device mode. Writing a one to this bit when the device mode is selected, will have undefined results.	-	-
7	-	-	Reserved. These bits should be set to 0.	-	-
9:8	-	-	Not used in Device mode.	-	-
10	-		Reserved.These bits should be set to 0.	-	0
11	-	-	Not used in Device mode.	-	
12	-		Reserved.These bits should be set to 0.	-	0
13	SUTW		Setup trip wire During handling a setup packet, this bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (see USBMODE register) then there exists a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software and will be cleared by hardware when a hazard exists. (See Section 23.10).	R/W	0
14	ATDTW		Add dTD trip wire This bit is used as a semaphore to ensure the to proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software during the process of adding a new dTD. See also Section 23.10. This bit shall also be cleared by hardware when its state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.	R/W	0

Table 389. USB Command register in device mode (USBCMD_D - address 0x4000 6140) bit description *...continued*

Bit	Symbol	Value	Description	Access	Reset value
15	-		Not used in device mode.	-	-
23:16	ITC		Interrupt threshold control. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. All other values are reserved. 0x0 = Immediate (no threshold) 0x1 = 1 micro frame. 0x2 = 2 micro frames. 0x8 = 8 micro frames. 0x10 = 16 micro frames. 0x20 = 32 micro frames. 0x40 = 64 micro frames.	R/W	0x8
31:24	-		Reserved		0

23.6.3.2 Host mode

Table 390. USB Command register in host mode (USBCMD_H - address 0x4000 6140) bit description - host mode

Bit	Symbol	Value	Description	Access	Reset value
0	RS		Run/Stop	R/W	0
		0	When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Host Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the host controller is in the Halted state (i.e. HCHalted in the USBSTS register is a one).		
		1	When set to a 1, the Host Controller proceeds with the execution of the schedule. The Host Controller continues execution as long as this bit is set to a one.		
1	RST		Controller reset. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register.	R/W	0
		0	This bit is set to zero by hardware when the reset process is complete.		
		1	When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior.		
2	FS0		Bit 0 of the Frame List Size bits. See Table 391. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index. Note that this field is made up from USBCMD bits 15, 3, and 2.		0
3	FS1		Bit 1 of the Frame List Size bits. See Table 391.		0

Table 390. USB Command register in host mode (USBCMD_H - address 0x4000 6140) bit description - host mode

Bit	Symbol	Value	Description	Access	Reset value
4	PSE		This bit controls whether the host controller skips processing the periodic schedule.	R/W	0
		0	Do not process the periodic schedule.		
		1	Use the PERIODICLISTBASE register to access the periodic schedule.		
5	ASE		This bit controls whether the host controller skips processing the asynchronous schedule.	R/W	0
		0	Do not process the asynchronous schedule.		
		1	Use the ASYNCLISTADDR to access the asynchronous schedule.		
6	IAA		This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule.	R/W	0
		0	The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one.		
		1	Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results.		
7	-	-	Reserved		0
9:8	ASP1_0		Asynchronous schedule park mode Contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 0x1 to 0x3. **Remark:** Software must not write 00 to this bit when Park Mode Enable is one as this will result in undefined behavior.	R/W	11
10	-	-	Reserved.	-	0
11	ASPE		Asynchronous Schedule Park Mode Enable	R/W	1
		0	Park mode is disabled.		
		1	Park mode is enabled.		
12	-	-	Reserved.	-	0
13	-	-	Not used in Host mode.	-	
14	-	-	Reserved.	-	0

Table 390. USB Command register in host mode (USBCMD_H - address 0x4000 6140) bit description - host mode

Bit	Symbol	Value	Description	Access	Reset value
15	FS2		Bit 2 of the Frame List Size bits. See Table 391.	-	0
23:16	ITC		Interrupt threshold control. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. All other values are reserved. 0x0 = Immediate (no threshold) 0x1 = 1 micro frame. 0x2 = 2 micro frames. 0x8 = 8 micro frames. 0x10 = 16 micro frames. 0x20 = 32 micro frames. 0x40 = 64 micro frames.	R/W	0x8
31:24	-		Reserved		0

Table 391. Frame list size values

USBCMD bit 15	USBCMD bit 3	USBCMD bit 2	Frame list size
0	0	0	1024 elements (4096 bytes) - default value
0	0	1	512 elements (2048 bytes)
0	1	0	256 elements (1024 bytes)
0	1	1	128 elements (512 bytes)
1	0	0	64 elements (256 bytes)
1	0	1	32 elements (128 bytes)
1	1	0	16 elements (64 bytes)
1	1	1	8 elements (32 bytes)

23.6.4 USB Status register (USBSTS)

This register indicates various states of the Host/Device controller and any pending interrupts. Software sets a bit to zero in this register by writing a one to it.

Remark: This register does not indicate status resulting from a transaction on the serial bus.

23.6.4.1 Device mode

Table 392. USB Status register in device mode (USBSTS_D - address 0x4000 6144) register bit description

Bit	Symbol	Value	Description	Reset value	Access
0	UI		USB interrupt	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
1	UEI		USB error interrupt	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set. The device controller detects resume signaling only (see Section 23.10.11.6).		
2	PCI		Port change detect.	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit (URI) and the DCSuspend bits (SLI) respectively.		
3	-		Not used in Device mode.		-
4	-		Reserved.	0	-
5	AAI		Not used in Device mode.	0	-
6	URI		USB reset received	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When the device controller detects a USB Reset and enters the default state, this bit will be set to a one.		

Table 392. USB Status register in device mode (USBSTS_D - address 0x4000 6144) register bit description

Bit	Symbol	Value	Description	Reset value	Access
7	SRI		SOF received	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1 ms in device FS mode and every 125 μs in HS mode and will be synchronized to the actual SOF that is received. Since the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp.		
8	SLI		DCSuspend	0	R/WC
		0	The device controller clears the bit upon exiting from a suspend state. This bit is cleared by software writing a one to it.		
		1	When a device controller enters a suspend state from an active state, this bit will be set to a one.		
11:9	-	-	Reserved. Software should only write 0 to reserved bits.		
12	-	-	Not used in Device mode.	0	
13	-	-	Not used in Device mode.	0	
14	-	-	Not used in Device mode.	0	
15	-	-	Not used in Device mode.	0	
16	NAKI		NAK interrupt bit	0	RO
		0	This bit is automatically cleared by hardware when the all the enabled TX/RX Endpoint NAK bits are cleared.		
		1	It is set by hardware when for a particular endpoint both the TX/RX Endpoint NAK bit and the corresponding TX/RX Endpoint NAK Enable bit are set.		
17	-	-	Reserved. Software should only write 0 to reserved bits.	0	-
18	-		Not used in Device mode.	0	-
19	-		Not used in Device mode.	0	-
31:20	-	-	Reserved. Software should only write 0 to reserved bits.		-

23.6.4.2 Host mode

Table 393. USB Status register in host mode (USBSTS_H - address 0x4000 6144) register bit description

Bit	Symbol	Value	Description	Reset value	Access
0	UI		USB interrupt (USBINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
1	UEI		USB error interrupt (USBERRINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set.		
2	PCI		Port change detect.	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port.		
3	FRI		Frame list roll-over	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX bit 13 toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FRINDEX bit 12 toggles (see Section 23.6.6).		
4	-		Reserved.		
5	AAI		Interrupt on async advance	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the USBCMD register. This status bit indicates the assertion of that interrupt source.		
6	-	-	Not used by the Host controller.	0	R/WC
7	SRI		SOF received	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	In host mode, this bit will be set every 125 μs and can be used by host controller driver as a time base.		
8	-	-	Not used by the Host controller.	-	-
11:9	-	-	Reserved.		

Table 393. USB Status register in host mode (USBSTS_H - address 0x4000 6144) register bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
12	HCH		HCHalted	1	RO
		0	The RS bit in USBCMD is set to zero. Set by the host controller.		
		1	The Host Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. because of an internal error).		
13	RCL		Reclamation	0	RO
		0	No empty asynchronous schedule detected.		
		1	An empty asynchronous schedule is detected. Set by the host controller.		
14	PS		Periodic schedule status This bit reports the current real status of the Periodic Schedule. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (if both are 1) or disabled (if both are 0).	0	RO
		0	The periodic schedule status is disabled.		
		1	The periodic schedule status is enabled.		
15	AS		Asynchronous schedule status This bit reports the current real status of the Asynchronous Schedule. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (if both are 1) or disabled (if both are 0).	0	
		0	Asynchronous schedule status is disabled.		
		1	Asynchronous schedule status is enabled.		
16	-		Not used on Host mode.	0	-
17	-		Reserved.		
18	UAI		USB host asynchronous interrupt (USBHSTASYNCINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set **and** the TD was from the asynchronous schedule. This bit is also set by the Host when a short packet is detected **and** the packet is on the asynchronous schedule. A short packet is when the actual number of bytes received was less than the expected number of bytes.		

Table 393. USB Status register in host mode (USBSTS_H - address 0x4000 6144) register bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
19	UPI		USB host periodic interrupt (USBHSTPERINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set **and** the TD was from the periodic schedule. This bit is also set by the Host Controller when a short packet is detected **and** the packet is on the periodic schedule. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
31:20	-				

23.6.5 USB Interrupt register (USBINTR)

The software interrupts are enabled with this register. An interrupt is generated when a bit is set and the corresponding interrupt is active. The USB Status register (USBSTS) still shows interrupt sources even if they are disabled by the USBINTR register, allowing polling of interrupt events by the software. All interrupts must be acknowledged by software by clearing (that is writing a 1 to) the corresponding bit in the USBSTS register.

23.6.5.1 Device mode

Table 394. USB Interrupt register in device mode (USBINTR_D - address 0x4000 6148) bit description

Bit	Symbol	Description	Reset value	Access
0	UE	USB interrupt enable When this bit is one, and the USBINT bit in the USBSTS register is one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit in USBSTS.	0	R/W
1	UEE	USB error interrupt enable When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit in the USBSTS register.	0	R/W
2	PCE	Port change detect enable When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host/device controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit in USBSTS.	0	R/W
3	-	Not used by the Device controller.		
4	-	Reserved	0	-
5	-	Not used by the Device controller.		
6	URE	USB reset enable When this bit is a one, and the USB Reset Received bit in the USBSTS register is a one, the device controller will issue an interrupt. The interrupt is acknowledged by software clearing the USB Reset Received bit.	0	R/W

Table 394. USB Interrupt register in device mode (USBINTR_D - address 0x4000 6148) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
7	SRE	SOF received enable When this bit is a one, and the SOF Received bit in the USBSTS register is a one, the device controller will issue an interrupt. The interrupt is acknowledged by software clearing the SOF Received bit.	0	R/W
8	SLE	Sleep enable When this bit is a one, and the DCSuspend bit in the USBSTS register transitions, the device controller will issue an interrupt. The interrupt is acknowledged by software writing a one to the DCSuspend bit.	0	R/W
15:9	-	Reserved	-	-
16	NAKE	NAK interrupt enable This bit is set by software if it wants to enable the hardware interrupt for the NAK Interrupt bit. If both this bit and the corresponding NAK Interrupt bit are set, a hardware interrupt is generated.	0	R/W
17	-	Reserved		
18	-	Not used by the Device controller.		
19	-	Not used by the Device controller.		
31:20	-	Reserved		

23.6.5.2 Host mode

Table 395. USB Interrupt register in host mode (USBINTR_H - address 0x4000 6148) bit description

Bit	Symbol	Description	Access	Reset value
0	UE	USB interrupt enable When this bit is one, and the USBINT bit in the USBSTS register is one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit in USBSTS.	R/W	0
1	UEE	USB error interrupt enable When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit in the USBSTS register.	R/W	0
2	PCE	Port change detect enable When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host/device controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit in USBSTS.	R/W	0
3	FRE	Frame list rollover enable When this bit is a one, and the Frame List Rollover bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit.		
4	-	Reserved	-	0
5	AAE	Interrupt on asynchronous advance enable When this bit is a one, and the Interrupt on Async Advance bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.	R/W	0
6	-	Not used by the Host controller.	-	0

Table 395. USB Interrupt register in host mode (USBINTR_H - address 0x4000 6148) bit description *...continued*

Bit	Symbol	Description	Access	Reset value
7	SRE	If this bit is one and the SRI bit in the USBSTS register is one, the host controller will issue an interrupt. In host mode, the SRI bit will be set every 125 μs and can be used by the host controller as a time base. The interrupt is acknowledged by software clearing the SRI bit in the USBSTS register.	-	0
8	-	Not used by the Host controller.	-	0
15:9	-	Reserved		
16	-	Not used by the host controller.	R/W	0
17	-	Reserved		
18	UAIE	USB host asynchronous interrupt enable When this bit is a one, and the USBHSTASYNCINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHSTASYNCINT bit.	R/W	0
19	UPIA	USB host periodic interrupt enable When this bit is a one, and the USBHSTPERINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHSTPERINT bit.	R/W	0
31:20	-	Reserved		

23.6.6 Frame index register (FRINDEX)

23.6.6.1 Device mode

In Device mode this register is read only, and the device controller updates the FRINDEX[13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX[13:3] will be checked against the SOF marker. If FRINDEX[13:3] is different from the SOF marker, FRINDEX[13:3] will be set to the SOF value and FRINDEX[2:0] will be set to zero (i.e. SOF for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX[2:0] will be incremented (i.e. SOF for 125 μs micro-frame) by hardware.

Table 396. USB frame index register in device mode (FRINDEX_D - address 0x4000 614C) bit description

Bit	Symbol	Description	Reset value	Access
2:0	FRINDEX2_0	Current micro frame number	N/A	RO
13:3	FRINDEX13_3	Current frame number of the last frame transmitted	N/A	RO
31:14	-	Reserved	N/A	

23.6.6.2 Host mode

This register is used by the host controller to index the periodic frame list. The register updates every 125 μs (once each micro-frame). Bits[N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the USBCMD register.

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit in the USBSTS register (host mode). A write to this register while the Run/Stop bit is set to a one produces undefined results. Writes to this register also affect the SOF value.

Table 397. USB frame index register in host (FRINDEX_H - address 0x4000 614C) bit description

Bit	Symbol	Description	Reset value	Access
2:0	FRINDEX2_0	Current micro frame number	N/A	R/W
12:3	FRINDEX12_3	Frame list current index.	N/A	R/W
31:13	-	Reserved	N/A	

Table 398. Number of bits used for the frame list index

USBCMD bit 15	USBCMD bit 3	USBCMD bit 2	Frame list size	N
0	0	0	1024 elements (4096 bytes). Default value.	12
0	0	1	512 elements (2048 bytes)	11
0	1	0	256 elements (1024 bytes)	10
0	1	1	128 elements (512 bytes)	9
1	0	0	64 elements (256 bytes)	8
1	0	1	32 elements (128 bytes)	7
1	1	0	16 elements (64 bytes)	6
1	1	1	8 elements (32 bytes)	5

23.6.7 Device address (DEVICEADDR - device) and Periodic List Base (PERIODICLISTBASE- host) registers

23.6.7.1 Device mode

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET_ADDRESS descriptor.

The USBADRA bit is used to accelerate the SET_ADDRESS sequence by allowing the DCD to preset the USBADR register bits before the status phase of the SET_ADDRESS descriptor.

Table 399. USB Device Address register in device mode (DEVICEADDR - address 0x4000 6154) bit description

Bit	Symbol	Value	Description	Reset value	Access
23:0	-		Reserved	0	-

Table 399. USB Device Address register in device mode (DEVICEADDR - address 0x4000 6154) bit description

Bit	Symbol	Value	Description	Reset value	Access
24	USBADRA		Device address advance		
		0	Any write to USBADR are instantaneous.		
		1	When the user writes a one to this bit at the same time or before USBADR is written, the write to USBADR fields is staged and held in a hidden register. After an IN occurs on endpoint 0 and is acknowledged, USBADR will be loaded from the holding register. Hardware will automatically clear this bit on the following conditions: • IN is ACKed to endpoint 0. USBADR is updated from the staging register. • OUT/SETUP occurs on endpoint 0. USBADR is **not** updated. • Device reset occurs. USBADR is set to 0. **Remark:** After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism will ensure this specification is met when the DCD can not write the device address within 2 ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR will be programmed instantly at the correct time and meet the 2 ms USB requirement.		
31:25	USBADR		USB device address	0	R/W

23.6.7.2 Host mode

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. The host controller driver (HCD) loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4 kB aligned. The contents of this register are combined with the Frame Index Register (FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Table 400. USB Periodic List Base register in host mode (PERIODICLISTBASE - address 0x4000 6154) bit description

Bit	Symbol	Description	Reset value	Access
11:0	-	Reserved	-	-
31:12	PERBASE31_12	Base Address (Low) These bits correspond to the memory address signals 31:12.	-	R/W

23.6.8 Endpoint List Address register (ENDPOINTLISTADDR - device) and Asynchronous List Address (ASYNCLISTADDR - host) registers

23.6.8.1 Device mode

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits[10:0] of this register cannot be modified by the system software and will always return a zero when read.The memory structure referenced by this physical memory pointer is assumed 64 byte aligned.

Table 401. USB Endpoint List Address register in device mode (ENDPOINTLISTADDR - address 0x4000 6158) bit description

Bit	Symbol	Description	Reset value	Access
10:0	-	reserved	0	-
31:11	EPBASE31_11	Endpoint list pointer (low) These bits correspond to memory address signals 31:11, respectively. This field will reference a list of up to 4 Queue Heads (QH). (i.e. one queue head per endpoint and direction.)	-	R/W

23.6.8.2 Host mode

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

Table 402. USB Asynchronous List Address register in host mode (ASYNCLISTADDR- address 0x4000 6158) bit description

Bit	Symbol	Description	Reset value	Access
4:0	-	Reserved	0	-
31:5	ASYBASE31_5	Link pointer (Low) LPL These bits correspond to memory address signals 31:5, respectively. This field may only reference a Queue Head (OH).	-	R/W

23.6.9 TT Control register (TTCTRL)

23.6.9.1 Device mode

This register is not used in device mode.

23.6.9.2 Host mode

This register contains parameters necessary for internal TT operations. This register is used by the host controller only. Writes must be in Dwords.

Table 403. USB TT Control register in host mode (TTCTRL - address 0x4000 615C) bit description

Bit	Symbol	Description	Reset value	Access
23:0	-	Reserved.	0	-
30:24	TTHA	Hub address when FS or LS device are connected directly.	N/A	R/W
31	-	Reserved.	0	

23.6.10 Burst Size register (BURSTSIZE)

This register is used to control and dynamically change the burst size used during data movement on the master interface of the USB DMA controller. Writes must be in Dwords.

The default for the length of a burst of 32-bit words for RX and TX DMA data transfers is 16 words each.

Table 404. USB burst size register (BURSTSIZE - address 0x4000 6160) bit description - device/host mode

Bit	Symbol	Description	Reset value	Access
7:0	RXPBURST	Programmable RX burst length This register represents the maximum length of a burst in 32-bit words while moving data from the USB bus to system memory.	0x10	R/W
15:8	TXPBURST	Programmable TX burst length This register represents the maximum length of a burst in 32-bit words while moving data from system memory to the USB bus.	0x10	R/W
31:16	-	Reserved.	-	-

23.6.11 Transfer buffer Fill Tuning register (TXFILLTUNING)

23.6.11.1 Device controller

This register is not used in device mode.

23.6.11.2 Host controller

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level

T_s = Total packet flight time (send-only) packet; $T_s = T_0 + T_1$

T_p = Total packet time (fetch and send) packet; $T_p = T_{ff} + T_0 + T_1$

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure T_p remains before the end of the (micro) frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is < T_s then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a mark will be made the scheduler health counter to note the occurrence of a “backoff” event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Backoffs can be minimized with use of the TSCHHEALTH (T_{ff}) described below.

Table 405. USB Transfer buffer Fill Tuning register in host mode (TXFILLTUNING - address 0x4000 6164) bit description

Bit	Symbol	Description	Reset value	Access
7:0	TXSCHOH	FIFO burst threshold This register controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on to the bus. The minimum value is 2 and this value should be a low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USBMODE register is set.	0x2	R/W
12:8	TXSCHEATLTH	Scheduler health counter This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame . This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter. The maximum value is 31.	0x0	R/W
15:13	-	reserved	-	-
21:16	TXFIFOTHRES	Scheduler overhead This register adds an additional fixed offset to the schedule time estimator described above as T_{ff}. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267 μs when a device is connected in High-Speed Mode for OTG and SPH. The time unit represented in this register is 6.333 μs when a device is connected in Low/Full Speed Mode for OTG and SPH.	0x0	R/W
31:22	-	reserved		

23.6.12 BINTERVAL register

This register defines the bInterval value which determines the length of the virtual frame (see Section 23.7.7). The BINTERVAL register divides the SOF signal by the value BINT.

Remark: The BINTERVAL register is not related to the bInterval endpoint descriptor field in the USB specification.

Table 406. USB BINTERVAL register (BINTERVAL - address 0x4000 6174) bit description

Bit	Symbol	Description	Reset value	Access
3:0	BINT	bInterval value (see Section 23.7.7)	0x00	R/W
31:4	-	reserved	-	-

23.6.13 USB Endpoint NAK register (ENDPTNAK)

23.6.13.1 Device mode

This register indicates when the device sends a NAK handshake on an endpoint. Each Tx and Rx endpoint has a bit in the EPTN and EPRN field respectively.

A bit in this register is cleared by writing a 1 to it.

Table 407. USB endpoint NAK register (ENDPTNAK - address 0x4000 6178) bit description

Bit	Symbol	Description	Reset value	Access
5:0	EPRN	Rx endpoint NAK Each RX endpoint has one bit in this field. The bit is set when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint. Bit 5 corresponds to endpoint 5. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/WC
15:6	-	Reserved	-	-
21:16	EPTN	Tx endpoint NAK Each TX endpoint has one bit in this field. The bit is set when the device sends a NAK handshake on a received IN token for the corresponding endpoint. Bit 5 corresponds to endpoint 5. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/WC
31:22	-	reserved	-	-

23.6.13.2 Host mode

This register is not used in host mode.

23.6.14 USB Endpoint NAK Enable register (ENDPTNAKEN)

23.6.14.1 Device mode

Each bit in this register enables the corresponding bit in the ENDPTNAK register. Each Tx and Rx endpoint has a bit in the EPTNE and EPRNE field respectively.

Table 408. USB Endpoint NAK Enable register (ENDPTNAKEN - address 0x4000 617C) bit description

Bit	Symbol	Description	Reset value	Access
5:0	EPRNE	Rx endpoint NAK enable Each bit enables the corresponding RX NAK bit. If this bit is set and the corresponding RX endpoint NAK bit is set, the NAK interrupt bit is set. Bit 5 corresponds to endpoint 5. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/W
15:6	-	Reserved	-	-
21:16	EPTNE	Tx endpoint NAK Each bit enables the corresponding TX NAK bit. If this bit is set and the corresponding TX endpoint NAK bit is set, the NAK interrupt bit is set. Bit 5 corresponds to endpoint 5. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/W
31:22	-	Reserved	-	-

23.6.14.2 Host mode

This register is not used in host mode.

23.6.15 Port Status and Control register (PORTSC1)

23.6.15.1 Device mode

The device controller implements one port register, and it does not support power control. Port control in device mode is used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling. This register allows software to put the PHY into low-power Suspend mode and disable the PHY clock.

Table 409. Port Status and Control register in device mode (PORTSC1_D - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	CCS		Current connect status	0	RO
		0	Device not attached A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.		
		1	Device attached. A one indicates that the device successfully attached and is operating in either high-speed mode or full-speed mode as indicated by the High Speed Port bit in this register.		
1	-	-	Not used in device mode	0	-
2	PE		Port enable. This bit is always 1. The device port is always enabled.	1	RO
3	PEC		Port enable/disable change This bit is always 0. The device port is always enabled.	0	RO

Table 409. Port Status and Control register in device mode (PORTSC1_D - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
5:4	-	-	Reserved	0	RO
6	FPR		Force port resume After the device has been in Suspend State for 5 ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. When this bit transitions to a one because a J-to-K transition detected, the Port Change Detect bit in the USBSTS register is set to one as well.	0	R/W
		0	No resume (K-state) detected/driven on port.		
		1	Resume detected/driven on port.		
7	SUSP		Suspend In device mode, this is a read-only status bit .	0	RO
		0	Port not in suspend state		
		1	Port in suspend state		
8	PR		Port reset In device mode, this is a read-only status bit. A device reset from the USB bus is also indicated in the USBSTS register.	0	RO
		0	Port is not in the reset state.		
		1	Port is in the reset state.		
9	HSP		High-speed status **Remark:** This bit is redundant with bits 27:26 (PSPD) in this register. It is implemented for compatibility reasons.	0	RO
		0	Host/device connected to the port is not in High-speed mode.		
		1	Host/device connected to the port is in High-speed mode.		
11:10	-	-	Not used in device mode.		
12	-	-	Not used in device mode.		
13	-	-	Reserved	-	-
15:14	PIC1_0		Port indicator control Writing to this field effects the value of the USB0_IND[1:0] pins.	00	R/W
		0x0	Port indicators are off.		
		0x1	amber		
		0x2	green		
		0x3	undefined		

Table 409. Port Status and Control register in device mode (PORTSC1_D - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
19:16	PTC3_0		Port test control Any value other than 0000 indicates that the port is operating in test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_HS/FS/LS values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. Values 0111 to 1111 are not valid.	0000	R/W
		0x0	TEST_MODE_DISABLE		
		0x1	J_STATE		
		0x2	K_STATE		
		0x3	SE0 (host)/NAK (device)		
		0x4	Packet		
		0x5	FORCE_ENABLE_HS		
		0x6	FORCE_ENABLE_FS		
20	-	-	Not used in device mode. This bit is always 0 in device mode.	0	-
21	-	-	Not used in device mode. This bit is always 0 in device mode.	0	-
22	-		Not used in device mode. This bit is always 0 in device mode.	0	-
23	PHCD		PHY low power suspend - clock disable (PLPSCD) In device mode, The PHY can be put into Low Power Suspend – Clock Disable when the device is not running (USBCMD Run/Stop = 0) or the host has signaled suspend (PORTSC SUSPEND = 1). Low power suspend will be cleared automatically when the host has signaled resume. Before forcing a resume from the device, the device controller driver must clear this bit.	0	R/W
		0	Writing a 0 enables the PHY clock. Reading a 0 indicates the status of the PHY clock (enabled).		
		1	Writing a 1 disables the PHY clock. Reading a 1 indicates the status of the PHY clock (disabled).		
24	PFSC		Port force full speed connect	0	R/W
		0	Port connects at any speed.		
		1	Writing this bit to a 1 will force the port to only connect at full speed. It disables the chirp sequence that allows the port to identify itself as High-speed. This is useful for testing FS configurations with a HS host, hub or device.		
25	-	-	reserved		
27:26	PSPD		Port speed This register field indicates the speed at which the port is operating.	0	RO
		0x0	Full-speed		
		0x1	invalid in device mode		
		0x2	High-speed		
31:28	-	-	Reserved	-	-

23.6.15.2 Host mode

The host controller uses one port. The register is only reset when power is initially applied or in response to a controller reset. The initial conditions of the port are:

- No device connected
- Port disabled

If the port has power control, this state remains until software applies power to the port by setting port power to one in the PORTSC register.

Table 410. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	CCS		Current connect status This value reflects the current state of the port and may not correspond directly to the event that caused the CSC bit to be set. This bit is 0 if PP (Port Power bit) is 0. Software clears this bit by writing a 1 to it.	0	R/WC
		0	No device is present.		
		1	Device is present on the port.		
1	CSC		Connect status change Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be setting an already-set bit (i.e., the bit will remain set). Software clears this bit by writing a one to it. This bit is 0 if PP (Port Power bit) is 0	0	R/WC
		0	No change in current status.		
		1	Change in current status.		
2	PE		Port enable. Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled. downstream propagation of data is blocked except for reset. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Port disabled.		
		1	Port enabled.		

Table 410. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
3	PEC		Port disable/enable change For the root hub, this bit gets set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See *Chapter 11 of the USB Specification*). Software clears this by writing a one to it. This bit is 0 if PP (Port Power bit) is 0,	0	R/WC
		0	No change.		
		1	Port enabled/disabled status has changed.		
4	OCA		Over-current active This bit will automatically transition from 1 to 0 when the over-current condition is removed.	0	RO
		0	The port does not have an over-current condition.		
		1	The port has currently an over-current condition.		
5	OCC		Over-current change This bit gets set to one when there is a change to Over-current Active. Software clears this bit by writing a one to this bit position.	0	R/WC
6	FPR		Force port resume Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the Port Change Detect bit in the USBSTS register is also set to one. This bit will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver. Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed K) is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no affect because the port controller will time the resume operation clear the bit the port control state switches to HS or FS idle. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	No resume (K-state) detected/driven on port.		
		1	Resume detected/driven on port.		

Table 410. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
7	SUSP		Suspend Together with the PE (Port enabled bit), this bit describes the port states, see Table 411. The host controller will unconditionally set this bit to zero when software sets the Force Port Resume bit to zero. The host controller ignores a write of zero to this bit. If host software sets this bit to a one when the port is not enabled (i.e. Port enabled bit is a zero) the results are undefined. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Port not in suspend state		
		1	Port in suspend state When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.		
8	PR		Port reset When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Port is not in the reset state.		
		1	Port is in the reset state.		
9	HSP		High-speed status	0	RO
		0	Host/device connected to the port is not in High-speed mode.		
		1	Host/device connected to the port is in High-speed mode.		
11:10	LS		Line status These bits reflect the current logical levels of the USB_DP and USB_DM signal lines. USB_DP corresponds to bit 11 and USB_DM to bit 10. In host mode, the use of linestate by the host controller driver is not necessary for this controller (unlike EHCI) because the controller hardware manages the connection of LS and FS.	0x3	RO
		0x0	SE0 (USB_DP and USB_DM LOW)		
		0x1	J-state (USB_DP HIGH and USB_DM LOW)		
		0x2	K-state (USB_DP LOW and USB_DM HIGH)		
		0x3	Undefined		

Table 410. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
12	PP	-	Port power control Host/OTG controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (i.e. PP equals a 0), the port is non-functional and will not report attaches, detaches, etc. When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitioned by the host controller driver from a one to a zero (removing power from the port).	0	R/W
		0	Port power off.		
		1	Port power on.		
13	-	-	Reserved	0	-
15:14	PIC1_0		Port indicator control Writing to this field effects the value of the pins USB0_IND1 and USB0_IND0.	00	R/W
		0x0	Port indicators are off.		
		0x1	Amber		
		0x2	Green		
		0x3	Undefined		
19:16	PTC3_0		Port test control Any value other than 0000 indicates that the port is operating in test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. Values 0x8 to 0xF are reserved.	0000	R/W
		0x0	TEST_MODE_DISABLE		
		0x1	J_STATE		
		0x2	K_STATE		
		0x3	SE0 (host)/NAK (device)		
		0x4	Packet		
		0x5	FORCE_ENABLE_HS		
		0x6	FORCE_ENABLE_FS		
		0x7	FORCE_ENABLE_LS		
20	WKCN		Wake on connect enable (WKCNNT_E) This bit is 0 if PP (Port Power bit) is 0	0	R/W
		0	Disables the port to wake up on device connects.		
		1	Writing this bit to a one enables the port to be sensitive to device connects as wake-up events.		

Table 410. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 6184) bit description

Bit	Symbol	Value	Description	Reset value	Access
21	WKDC		Wake on disconnect enable (WKDSCNNT_E) This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Disables the port to wake up on device disconnects.		
		1	Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events.		
22	WKOC		Wake on over-current enable (WKOC_E)	0	R/W
		0	Disables the port to wake up on over-current events.		
		1	Writing a one to this bit enabled the port to be sensitive to over-current conditions as wake-up events.		
23	PHCD		PHY low power suspend - clock disable (PLPSCD) In host mode, the PHY can be put into Low Power Suspend – Clock Disable when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software.	0	R/W
		0	Writing a 0 enables the PHY clock. Reading a 0 indicates the status of the PHY clock (enabled).		
		1	Writing a 1 disables the PHY clock. Reading a 1 indicates the status of the PHY clock (disabled).		
24	PFSC		Port force full speed connect	0	R/W
		0	Port connects at any speed.		
		1	Writing this bit to a 1 will force the port to only connect at Full Speed. It disables the chirp sequence that allows the port to identify itself as High Speed. This is useful for testing FS configurations with a HS host, hub or device.		
25	-	-	Reserved		
27:26	PSPD		Port speed This register field indicates the speed at which the port is operating. For HS mode operation in the host controller and HS/FS operation in the device controller the port routing steers data to the Protocol engine. For FS and LS mode operation in the host controller, the port routing steers data to the Protocol Engine w/ Embedded Transaction Translator.	0	RO
		0x0	Full-speed		
		0x1	Low-speed		
		0x2	High-speed		
31:28	-		Reserved	-	-

Table 411. Port states as described by the PE and SUSP bits in the PORTSC1 register

PE bit	SUSP bit	Port state
0	0 or 1	disabled
1	0	enabled
1	1	suspend

23.6.16 OTG Status and Control register (OTGSC)

The OTG register has four sections:

- OTG interrupt enables (R/W)
- OTG Interrupt status (R/WC)
- OTG status inputs (RO)
- OTG controls (R/W)

The status inputs are debounced using a 1 msec time constant. Values on the status inputs that do not persist for more than 1 msec will not cause an update of the status input register or cause an OTG interrupt.

Table 412. OTG Status and Control register (OTGSC - address 0x4000 61A4) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	VD		VBUS_Discharge Setting this bit to 1 causes VBUS to discharge through a resistor.	0	R/W
1	VC		VBUS_Charge Setting this bit to 1 causes the VBUS line to be charged. This is used for VBUS pulsing during SRP.	0	R/W
2	HAAR		Hardware assist auto_reset	0	R/W
		0	Disabled		
		1	Enable automatic reset after connect on host port.		
3	OT		OTG termination This bit must be set to 1 when the OTG controller is in device mode. This controls the pull-down on USB_DM.	0	R/W
4	DP		Data pulsing Setting this bit to 1 causes the pull-up on USB_DP to be asserted for data pulsing during SRP.	0	R/W
5	IDPU		ID pull-up. This bit provides control over the pull-up resistor.	1	R/W
		0	Pull-up off. The ID bit will not be sampled.		
		1	Pull-up on.		
6	HADP		Hardware assist data pulse Write a 1 to start data pulse sequence.	0	R/W
7	HABA		Hardware assist B-disconnect to A-connect	0	R/W
		0	Disabled.		
		1	Enable automatic B-disconnect to A-connect sequence.		
8	ID		USB ID	0	RO
		0	A-device		
		1	B-device		
9	AVV		A-VBUS valid Reading 1 indicates that VBUS is above the A-VBUS valid threshold.	0	RO
10	ASV		A-session valid Reading 1 indicates that VBUS is above the A-session valid threshold.	0	RO

Table 412. OTG Status and Control register (OTGSC - address 0x4000 61A4) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
11	BSV		B-session valid Reading 1 indicates that VBUS is above the B-session valid threshold.	0	RO
12	BSE		B-session end Reading 1 indicates that VBUS is below the B-session end threshold.	0	RO
13	MS1T		1 millisecond timer toggle This bit toggles once per millisecond.	0	RO
14	DPS		Data bus pulsing status Reading a 1 indicates that data bus pulsing is detected on the port.	0	RO
15	-	-	reserved	0	
16	IDIS		USB ID interrupt status This bit is set when a change on the ID input has been detected. Software must write a 1 to this bit to clear it.	0	R/WC
17	AVVIS		A-VBUS valid interrupt status This bit is set then VBUS has either risen above or fallen below the A-VBUS valid threshold (4.4 V on an A-device). Software must write a 1 to this bit to clear it.	0	R/WC
18	ASVIS		A-Session valid interrupt status This bit is set then VBUS has either risen above or fallen below the A-session valid threshold (0.8 V). Software must write a 1 to this bit to clear it.	0	R/WC
19	BSVIS		B-Session valid interrupt status This bit is set then VBUS has either risen above or fallen below the B-session valid threshold (0.8 V). Software must write a 1 to this bit to clear it.	0	R/WC
20	BSEIS		B-Session end interrupt status This bit is set then VBUS has fallen below the B-session end threshold. Software must write a 1 to this bit to clear it.	0	R/WC
21	ms1S		1 millisecond timer interrupt status This bit is set once every millisecond. Software must write a 1 to this bit to clear it.	0	R/WC
22	DPIS		Data pulse interrupt status This bit is set when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when the CM bit in USBMODE = Host (11) and the PortPower bit in PORTSC = Off (0). Software must write a 1 to this bit to clear it.	0	R/WC
23	-	-	reserved	0	
24	IDIE		USB ID interrupt enable Setting this bit enables the interrupt. Writing a 0 disables the interrupt.	0	R/W
25	AVVIE		A-VBUS valid interrupt enable Setting this bit enables the A-VBUS valid interrupt. Writing a 0 disables the interrupt.	0	R/W

Table 412. OTG Status and Control register (OTGSC - address 0x4000 61A4) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
26	ASVIE		A-session valid interrupt enable Setting this bit enables the A-session valid interrupt. Writing a 0 disables the interrupt	0	R/W
27	BSVIE		B-session valid interrupt enable Setting this bit enables the B-session valid interrupt. Writing a 0 disables the interrupt.	0	R/W
28	BSEIE		B-session end interrupt enable Setting this bit enables the B-session end interrupt. Writing a 0 disables the interrupt.	0	R/W
29	MS1E		1 millisecond timer interrupt enable Setting this bit enables the 1 millisecond timer interrupt. Writing a 0 disables the interrupt.	0	R/W
30	DPIE		Data pulse interrupt enable Setting this bit enables the data pulse interrupt. Writing a 0 disables the interrupt	0	R/W
31	-	-	Reserved	0	-

23.6.17 USB Mode register (USBMODE)

The USBMODE register sets the USB mode for the OTG controller. The possible modes are Device, Host, and Idle mode for OTG operations.

23.6.17.1 Device mode

Table 413. USB Mode register in device mode (USBMODE_D - address 0x4000 61A8) bit description

Bit	Symbol	Value	Description	Reset value	Access
1:0	CM1_0		Controller mode The controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. This register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the RESET bit in the USBCMD register before reprogramming this register.	00	R/ WO
		0x0	Idle		
		0x1	Reserved		
		0x2	Device controller		
		0x3	Host controller		
2	ES		Endian select This bit can change the byte ordering of the transfer buffers to match the host microprocessor bus architecture. The bit fields in the microprocessor interface and the DMA data structures (including the setup buffer within the device QH) are unaffected by the value of this bit, because they are based upon 32-bit words.	0	R/W
		0	Little endian: first byte referenced in least significant byte of 32-bit word.		
		1	Big endian: first byte referenced in most significant byte of 32-bit word.		

Table 413. USB Mode register in device mode (USBMODE_D - address 0x4000 61A8) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
3	SLOM		Setup Lockout mode In device mode, this bit controls behavior of the setup lock mechanism. See Section 23.10.8.	0	R/W
		0	Setup Lockouts on		
		1	Setup Lockouts Off (DCD requires the use of Setup Buffer Tripwire in USBCMD)		
4	SDIS		Stream disable mode **Remark:** The use of this feature substantially limits the overall USB performance that can be achieved.	0	R/W
		0	Not disabled		
		1	Disabled. Setting this bit to one disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received will be responded to with a NYET handshake when stream disable is active.		
5	-		Not used in device mode.	0	-
31:6	-	-	reserved		

23.6.17.2 Host mode

Table 414. USB Mode register in host mode (USBMODE_H - address 0x4000 61A8) bit description

Bit	Symbol	Value	Description	Reset value	Access
1:0	CM		Controller mode The controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. This register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the RESET bit in the USBCMD register before reprogramming this register.	00	R/ WO
		0x0	Idle		
		0x1	Reserved		
		0x2	Device controller		
		0x3	Host controller		
2	ES		Endian select This bit can change the byte ordering of the transfer buffers. The bit fields in the microprocessor interface and the DMA data structures (including the setup buffer within the device QH) are unaffected by the value of this bit, because they are based upon 32-bit words.	0	R/W
		0	Little endian: first byte referenced in least significant byte of 32-bit word.		
		1	Big endian: first byte referenced in most significant byte of 32-bit word.		
3	-		Not used in host mode	0	-

Table 414. USB Mode register in host mode (USBMODE_H - address 0x4000 61A8) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
4	SDIS		Stream disable mode **Remark:** The use of this feature substantially limits the overall USB performance that can be achieved.	0	R/W
		0	Not disabled		
		1	Disabled. Setting to a '1' ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the the TX latency is filled to capacity before the packet is launched onto the USB. Note: Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TXFILLTUNING to characterize the adjustments needed for the scheduler when using this feature.		
5	VBPS		VBUS power select	0	R/WO
		0	vbus_pwr_select is set LOW.		
		1	vbus_pwr_select is set HIGH		
31:6	-	-	reserved	-	-

23.6.18 USB Endpoint Setup Status register (ENDPSETUPSTAT)

Table 415. USB Endpoint Setup Status register (ENDPTSETUPSTAT - address 0x4000 61AC) bit description

Bit	Symbol	Description	Reset value	Access
5:0	ENDPTSETUPSTAT	Setup endpoint status for logical endpoints 0 to 5. For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lockout mechanism is engaged.	0	R/WC
31:6	-	reserved	-	-

23.6.19 USB Endpoint Prime register (ENDPTPRIME)

For each endpoint, software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed.

Remark: These bits will be momentarily set by hardware during hardware endpoint re-priming operations when a dTD is retired and the dQH is updated.

Table 416. USB Endpoint Prime register (ENDPTPRIME - address 0x4000 61B0) bit description

Bit	Symbol	Description	Reset value	Access
5:0	PERB	Prime endpoint receive buffer for physical OUT endpoints 5 to 0. For each OUT endpoint, a corresponding bit is set to 1 by software to request a buffer be prepared for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed. PERB0 = endpoint 0 ... PERB5 = endpoint 5	0	R/WS
15:6	-	Reserved	-	-
21:16	PETB	Prime endpoint transmit buffer for physical IN endpoints 5 to 0. For each IN endpoint a corresponding bit is set to one by software to request a buffer be prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed. PETB0 = endpoint 0 ... PETB5 = endpoint 5	0	R/WS
31:22	-	Reserved	-	-

23.6.20 USB Endpoint Flush register (ENDPTFLUSH)

Writing a one to a bit(s) in this register will cause the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer will continue until completion. Hardware will clear this register after the endpoint flush operation is successful.

Table 417. USB Endpoint Flush register (ENDPTFLUSH - address 0x4000 61B4) bit description

Bit	Symbol	Description	Reset value	Access
5:0	FERB	Flush endpoint receive buffer for physical OUT endpoints 5 to 0. Writing a one to a bit(s) will clear any primed buffers. FERB0 = endpoint 0 ... FERB5 = endpoint 5	0	R/WS

Table 417. USB Endpoint Flush register (ENDPTFLUSH - address 0x4000 61B4) bit description

Bit	Symbol	Description	Reset value	Access
15:6	-	Reserved	-	-
21:16	FETB	Flush endpoint transmit buffer for physical IN endpoints 5 to 0. Writing a one to a bit(s) will clear any primed buffers. FETB0 = endpoint 0 ... FETB5 = endpoint 5	0	R/WS
31:22	-	Reserved	-	-

23.6.21 USB Endpoint Status register (ENDPTSTAT)

One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set by hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There will always be a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register.

Remark: These bits will be momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired and the dQH is updated.

Table 418. USB Endpoint Status register (ENDPTSTAT - address 0x4000 61B8) bit description

Bit	Symbol	Description	Reset value	Access
5:0	ERBR	Endpoint receive buffer ready for physical OUT endpoints 5 to 0. This bit is set to 1 by hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. ERBR0 = endpoint 0 ... ERBR5 = endpoint 5	0	RO
15:6	-	Reserved	-	-
21:16	ETBR	Endpoint transmit buffer ready for physical IN endpoints 3 to 0. This bit is set to 1 by hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. ETBR0 = endpoint 0 ... ETBR5 = endpoint 5	0	RO
31:22	-	Reserved	-	-

23.6.22 USB Endpoint Complete register (ENDPTCOMPLETE)

Each bit in this register indicates that a received/transmit event occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit will be set simultaneously with the USBINT.

Writing a one will clear the corresponding bit in this register.

Table 419. USB Endpoint Complete register (ENDPTCOMPLETE - address 0x4000 61BC) bit description

Bit	Symbol	Description	Reset value	Access
5:0	ERCE	Endpoint receive complete event for physical OUT endpoints 5 to 0. This bit is set to 1 by hardware when receive event (OUT/SETUP) occurred. ERCE0 = endpoint 0 ... ERCE5 = endpoint 5	0	R/WC
15:6	-	reserved	-	-
21:16	ETCE	Endpoint transmit complete event for physical IN endpoints 5 to 0. This bit is set to 1 by hardware when a transmit event (IN/INTERRUPT) occurred. ETCE0 = endpoint 0 ... ETCE5 = endpoint 5	0	R/WC
31:22	-	Reserved	-	-

23.6.23 USB Endpoint 0 Control register (ENDPTCTRL0)

This register initializes endpoint 0 for control transfer. Endpoint 0 is always a control endpoint.

Table 420. USB Endpoint 0 Control register (ENDPTCTRL0 - address 0x4000 61C0) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RXS		Rx endpoint stall	0	R/W
		0	Endpoint ok.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDSETUPSTAT bit is cleared.[1]		
1	-	-	Reserved		
3:2	RXT1_0		Endpoint type Endpoint 0 is always a control endpoint.	00	R/W
6:4	-	-	reserved	-	-
7	RXE		Rx endpoint enable Endpoint enabled. Control endpoint 0 is always enabled. This bit is always 1.	1	RO
15:8	-	-	Reserved	-	-

Table 420. USB Endpoint 0 Control register (ENDPTCTRL0 - address 0x4000 61C0) bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
16	TXS		Tx endpoint stall		R/W
		0	Endpoint ok.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDSETUPSTAT bit is cleared.[1]		
17	-	-	Reserved		
19:18	TXT1_0		Endpoint type Endpoint 0 is always a control endpoint.	00	RO
22:20	-	-	Reserved		
23	TXE		Tx endpoint enable Endpoint enabled. Control endpoint 0 is always enabled. This bit is always 1.	1	RO
31:24	-	-	Reserved		

[1] There is a slight delay (50 clocks max) between the ENPTSETUPSTAT being cleared and hardware continuing to clear this bit. In most systems it is unlikely that the DCD software will observe this delay. However, should the DCD notice that the stall bit is not set after writing a one to it, software should continually write this stall bit until it is set or until a new setup has been received by checking the associated ENDPTSETUPSTAT bit.

23.6.24 Endpoint 1 to 5 control registers

Each endpoint that is not a control endpoint has its own register to set the endpoint type and enable or disable the endpoint.

Remark: The reset value for all endpoint types is the control endpoint. If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled, then the endpoint type of the unused direction must be changed from the control type to any other type (e.g. bulk). Leaving an unconfigured endpoint control will cause undefined behavior for the data PID tracking on the active endpoint.

Table 421. USB Endpoint 1 to 5 control registers (ENDPTCTRL - address 0x4000 61C4 (ENDPTCTRL1) to 0x4000 61D4 (ENDPTCTRL5)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RXS		Rx endpoint stall	0	R/W
		0	Endpoint ok. This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request.		

Table 421. USB Endpoint 1 to 5 control registers (ENDPTCTRL - address 0x4000 61C4 (ENDPTCTRL1) to 0x4000 61D4 (ENDPTCTRL5)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
1	-		Reserved	0	R/W
3:2	RXT		Endpoint type	00	R/W
		0x0	Control		
		0x1	Isochronous		
		0x2	Bulk		
		0x3	Interrupt		
4	-	-	Reserved		
5	RXI		Rx data toggle inhibit This bit is only used for test and should always be written as zero. Writing a one to this bit will cause this endpoint to ignore the data toggle sequence and always accept data packets regardless of their data PID.	0	R/W
		0	Disabled		
		1	Enabled		
6	RXR		Rx data toggle reset Write 1 to reset the PID sequence. Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PIDs between the host and device.	0	WS
7	RXE		Rx endpoint enable **Remark:** An endpoint should be enabled only after it has been configured.	0	R/W
		0	Endpoint disabled.		
		1	Endpoint enabled.		
15:8	-	-	reserved		
16	TXS		Tx endpoint stall	0	R/W
		0	Endpoint ok. This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint, and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request.		
17	-	-	Reserved	0	-
19:18	TXT1_0		Tx endpoint type	00	R/W
		0x0	Control		
		0x1	Isochronous		
		0x2	Bulk		
		0x3	Interrupt		
20	-	-	Reserved		

Table 421. USB Endpoint 1 to 5 control registers (ENDPTCTRL - address 0x4000 61C4 (ENDPTCTRL1) to 0x4000 61D4 (ENDPTCTRL5)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
21	TXI		Tx data toggle inhibit This bit is only used for test and should always be written as zero. Writing a one to this bit will cause this endpoint to ignore the data toggle sequence and always accept data packets regardless of their data PID.	0	R/W
		0	Enabled		
		1	Disabled		
22	TXR		Tx data toggle reset Write 1 to reset the PID sequence. Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.	1	WS
23	TXE		Tx endpoint enable **Remark:** An endpoint should be enabled only after it has been configured	0	R/W
		0	Endpoint disabled.		
		1	Endpoint enabled.		
31:24	-	-	Reserved	0	

23.7 Functional description

23.7.1 OTG core

The OTG core forms the main digital part of the USB-OTG. See the *USB EHCI specification* for details about this core.

23.7.2 Host data structures

See Chapter 4 of the *USB EHCI Specification for Universal Serial Bus 1.0*.

23.7.3 Host operational model

See Chapter 3 of the *USB EHCI Specification for Universal Serial Bus 1.0*.

23.7.4 ATX_RGEN module

The requirements for the reset signal towards the ATX transceiver are as follows:

- The clocks must be running for a reset to occur correctly.
- The ATX must see a rising edge of reset to correctly reset the clock generation module.
- The reset must be a minimum of 133 ns (4 × 30 MHz clock cycles) in duration to reset all logic correctly.

The ATX_RGEN module generates a reset signal towards the ATX fulfilling above 3 requirements, no matter how the AHB reset looks like.

23.7.5 ATX transceiver

The USB-OTG has a USB transceiver with UTMI+ interface. It contains the required transceiver OTG functionality; this includes:

- VBUS sensing for producing the session-valid and VBUS-valid signals.
- sampling of the USB_ID input for detection of A-device or B-device connection.
- charging and discharging of VBUS for starting and ending a session as B-device.

23.7.6 Modes of operation

In general, the USB-OTG can operate either in host mode or in device mode. Software must put the core in the appropriate mode by setting the USBMODE.CM field (‘11’ for host mode, ‘10’ for device mode).

The USBMODE.CM field can also be equal to ‘00’, which means that the core is in idle mode (neither host nor device mode). This will happen after the following:

- a hardware reset.
- a software reset via the USBCMD.RST bit. When switching from host mode to device mode as part of the HNP protocol (or vice versa), software must issue a software reset which sets the USB core to the idle state in a time frame dependent on the software.

23.7.7 SOF/VF indicator

See Section 23.2 for connecting the SOF signal to other peripherals on the LPC43xx.

The USB-OTG generates a SOF/VF indicator signal, which can be used by user specific external logic.

In FS mode, the SOF/VF indicator signal has a frequency equal to the frame frequency, which is about 1 kHz. The signal is high for half of the frame period and low for the other half of the frame period. The positive edge is aligned with the start of a frame (= SOF).

In HS mode, the SOF/VF indicator signal has a frequency equal to the virtual frame frequency. The positive edge indicates the start of the Virtual Frame and is aligned with the occurrence of an SOF token on the USB

The length of the virtual frame is defined as: VF = microframe $\times$ $2^{bInterval}$

bInterval is specified in the 4-bit programmable BINTERVAL.BINT register field. The minimum value of bInterval is 0, the maximum value is 15.

In suspend mode the SOF/VF indicator signal is turned off (= remains low).

23.7.8 Hardware assist

The hardware assist provides automated response and sequencing that may not be possible in software if there are significant interrupt latency response times. The use of this additional circuitry is optional and can be used to assist the following three state transitions by setting the appropriate bits in the OTGSC register:

- Auto reset (set bit HAAR).

- Data pulse (set bit HADP).
- B-disconnect to A-connect (set bit HABA).

23.7.8.1 Auto reset

When the HAAR in the OTGSC register is set to one, the host will automatically start a reset after a connect event. This shortcuts the normal process where software is notified of the connect event and starts the reset. Software will still receive notification of the connect event (CCS bit in the PORTSC register) but should not write the reset bit in the USBCMD register when the HAAR is set. Software will be notified again after the reset is complete via the enable change bit in the PORTSC register which causes a port change interrupt.

This assist will ensure the OTG parameter TB_ACON_BSE0_MAX = 1 ms is met (see *OTG specification* for an explanation of the OTG timing requirements).

23.7.8.2 Data pulse

Writing a one to HADP in the OTGSC register will start a data pulse of approximately 7 ms in duration and then automatically cease the data pulsing. During the data pulse, the DP bit will be set and then cleared. This automation relieves software from accurately controlling the data-pulse duration. During the data pulse, the HCD can poll to see that the HADP and DP bit have returned low to recognize the completion, or the HCD can simply launch the data pulse and wait to see if a VBUS Valid interrupt occurs when the A-side supplies bus power.

This assist will ensure data pulsing meets the OTG requirement of > 5 ms and < 10 ms.

23.7.8.3 B-disconnect to A-connect (Transition to the A-peripheral state)

During HNP, the B-disconnect occurs from the OTG A_suspend state, and within 3 ms, the A-device must enable the pull-up on the DP leg in the A-peripheral state. For the hardware assist to begin the following conditions must be met:

- HABA is set.
- Host controller is in suspend mode.
- Device is disconnecting.

The hardware assist consists of the following steps:

1. Hardware resets the OTG controller (writes 1 to the RST bit in USBCMD).
2. Hardware selects the device mode (writes 10 to bits CM[1:0] in USBMODE).
3. Hardware sets the RS bit in USBCMD and enables the necessary interrupts:
   - USB reset enable (URE) - enables interrupt on USB bus reset to device.
   - Sleep enable (SLE) - enables interrupt on device suspend.
   - Port change detect enable (PCE) - enables interrupt on device connect.

When software has enabled this hardware assist, it must not interfere during the transition and should not write any register in the OTG core until it gets an interrupt from the device controller signifying that a reset interrupt has occurred or until it has verified that the core has entered device mode. HCD/DCD must not activate the core soft reset at any time since this action is performed by hardware. During the transition, the software may see an

interrupt from the disconnect and/or other spurious interrupts (i.e. SOF/etc.) that may or may not cascade and my be cleared by the soft reset depending on the software response time.

After the core has entered device mode with help of the hardware assist, the DCD must ensure that the ENDPTLISTADDR is programmed properly before the host sends a setup packet. Since the end of the reset duration, which may be initiated quickly (a few microseconds) after connect, will require at a minimum 50 ms, this is the time for which the DCD must be ready to accept setup packets after having received notification that the reset has been detected or simply that the OTG is in device mode which ever occurs first.

If the A-peripheral fails to see a reset after the controller enters device mode and engages the D+-pull-up, the device controller interrupts the DCD signifying that a suspend has occurred. This assist will ensure the parameter TA_BDIS_ACON_MAX = 3ms is met.

23.8 Deviations from EHCI standard

For the purposes of a dual-role Host/Device controller with support for On-The-Go applications, it is necessary to deviate from the EHCI specification. Device operation and On-The-Go operation is not specified in the EHCI and thus the implementation supported in this core is specific to the LPC43xx. The host mode operation of the core is near EHCI compatible with few minor differences documented in this section.

The particulars of the deviations occur in the areas summarized here:

- Embedded Transaction Translator – Allows direct attachment of FS and LS devices in host mode without the need for a companion controller.
- Device operation - In host mode the device operational registers are generally disabled and thus device mode is mostly transparent when in host mode. However, there are a couple exceptions documented in the following sections.
- On-The-Go Operation - This design includes an On-The-Go controller.

23.8.1 Embedded Transaction Translator function

The USB-HS OTG controller supports directly connected full and low speed devices without requiring a companion controller by including the capabilities of a USB 2.0 high speed hub transaction translator. Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a high speed hub has been implemented within the DMA and Protocol engine blocks. The embedded transaction translator function is an extension to EHCI interface but makes use of the standard data structures and operational models that exist in the EHCI specification to support full and low speed devices.

23.8.1.1 Capability registers

The following items have been added to the capability registers to support the embedded Transaction Translator Function:

- N_TT bits added to HCSPARAMS – Host Control Structural Parameters (see Table 385).
- N_PTT added to HCSPARAMS – Host Control Structural Parameters (see Table 385).

23.8.1.2 Operational registers

The following items have been added to the operational registers to support the embedded TT:

- New register TTCTRL (see Section 23.6.9).
- Two-bit Port Speed (PSPD) bits added to the PORTSC1 register (see Section 23.6.15).

23.8.1.3 Discovery

In a standard EHCI controller design, the EHCI host controller driver detects a Full speed (FS) or Low speed (LS) device by noting if the port enable bit is set after the port reset operation. The port enable will only be set in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a High-Speed connection (i.e. Chirp completes successfully). Since this controller has an embedded Transaction Translator, the port enable will always be set after the port reset operation regardless of the result of the host device chirp result and the resulting port speed will be indicated by the PSPD field in PORTSC1 (see Section 23.6.15).

Table 422. Handling of directly connected full-speed and low-speed devices

Standard EHCI model	EHCI with embedded Transaction Translator
After the port enable bit is set following a connection and reset sequence, the device/hub is assumed to be HS.	After the port enable bit is set following a connection and reset sequence, the device/hub speed is noted from PORTSC1.
FS and LS devices are assumed to be downstream from a HS hub thus, all port-level control is performed through the Hub Class to the nearest Hub.	FS and LS device can be either downstream from a HS hub or directly attached. When the FS/LS device is downstream from a HS hub, then port-level control is done using the Hub Class through the nearest Hub. When a FS/LS device is directly attached, then port-level control is accomplished using PORTSC1.
FS and LS devices are assumed to be downstream from a HS hub with HubAddr=X, where HubAddr > 0 and HubAddr is the address of the Hub where the bus transitions from HS to FS/LS (i.e. Split target hub).	FS and LS device can be either downstream from a HS hub with HubAddr = X [HubAddr > 0] or directly attached, where HubAddr = TTHA (TTHA is programmable and defaults to 0) and HubAddr is the address of the Root Hub where the bus transitions from HS to FS/LS (i.e. Split target hub is the root hub).

23.8.1.4 Data structures

The same data structures used for FS/LS transactions though a HS hub are also used for transactions through the Root Hub with sm embedded Transaction Translator. Here it is demonstrated how the Hub Address and Endpoint Speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) – Async. (Bulk/Control Endpoints) Periodic (Interrupt)
   - Hub Address = TTHA (default TTHA = 0)
   - Transactions to direct attached device/hub: QH.EPS = Port Speed
   - Transactions to a device downstream from direct attached FS hub: QH.EPS = Downstream Device Speed

Remark: When QH.EPS = 01 (LS) and PORTSCx.PSPD = 00 (FS), a LS-pre-pid will be sent before the transmitting LS traffic.

Maximum Packet Size must be less than or equal 64 or undefined behavior may result.

2. siTD (for direct attach FS) – Periodic (ISO Endpoint)

   all FS ISO transactions:

   Hub Address = (default TTHA = 0)

   siTD.EPS = 00 (full speed)

   Maximum Packet Size must less than or equal to 1023 or undefined behavior may result.

23.8.1.5 Operational model

The operational models are well defined for the behavior of the Transaction Translator (see USB 2.0 specification) and for the EHCI controller moving packets between system memory and a USB-HS hub. Since the embedded Transaction Translator exists within the host controller there is no physical bus between EHCI host controller driver and the USB FS/LS bus. These sections will briefly discuss the operational model for how the EHCI and Transaction Translator operational models are combined without the physical bus between. The following sections assume the reader is familiar with both the EHCI and USB 2.0 Transaction Translator operational models.

23.8.1.5.1 Micro-frame pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the Host (H) and the Bus (B). The embedded Transaction Translator shall use the same pipeline algorithms specified in the USB 2.0 specification for a Hub-based Transaction Translator.

It is important to note that when programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded Transaction Translator, the EHCI host controller driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream Hub-based Transaction Translators. Once periodic transfers are exhausted, any stored asynchronous transfer will be moved. Asynchronous transfers are opportunistic in that they shall execute whenever possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer can not babble through the SOF (start of B-frame 0.)

23.8.1.6 Split state machines

The start and complete split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded Transaction Translator. Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete split operation is simple an internal operation to the embedded Transaction Translator. The following table summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

Table 423. Split state machine properties

	Condition	Emulate TT response
Start-split	All asynchronous buffers full.	NAK
	All periodic buffers full.	ERR
	Success for start of Async. Transaction.	ACK
	Start Periodic Transaction.	No Handshake (Ok)
Complete-split	Failed to find transaction in queue.	Bus Time Out
	Transaction in Queue is Busy.	NYET
	Transaction in Queue is Complete.	[Actual Handshake from LS/FS device]

23.8.1.7 Asynchronous Transaction scheduling and buffer management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

1. *USB 2.0 specification, section 11.17.3*: Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.
2. *USB 2.0 specification, section 11.17.4*: Transaction tracking for 2 data pipes.
3. *USB 2.0 specification, section 11.17.5*: Clear_TT_Buffer capability provided though the use of the TTCTRL register.

23.8.1.8 Periodic Transaction scheduling and buffer management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

1. *USB 2.0 specs, section 11.18.6.[1-2]*:
   - Abort of pending start-splits:

     EOF (and not started in micro-frames 6)

     Idle for more than 4 micro-frames
   - Abort of pending complete-splits:

     EOF

     Idle for more than 4 micro-frames
2. *USB 2.0 specs, section 11.18.6.[7-8]*:
   - Transaction tracking for up to 16 data pipes:

     Some applications may not require transaction tracking up to a maximum of 16 periodic data pipes. The option to limit the tracking to only 4 periodic data pipes exists in the by changing the configuration constant VUSB_HS_TT_PERIODIC_CONTEXTS to 4. The result is a significant gate count savings to the core given the limitations implied.

     Remark: Limiting the number of tracking pipes in the EMBedded TT to four (4) will impose the restriction that no more than 4 periodic transactions (INTERRUPT/ISOCHRONOUS) can be scheduled through the embedded TT per frame. The number 16 was chosen in the USB specification because it is sufficient to ensure that the high-speed to full- speed periodic pipeline can remain full.

keeping the pipeline full puts no constraint on the number of periodic transactions that can be scheduled in a frame and the only limit becomes the flight time of the packets on the bus.

– Complete-split transaction searching:

  There is no data schedule mechanism for these transactions other than micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

23.8.1.9 Multiple Transaction Translators

The maximum number of embedded Transaction Translators that is currently supported is one as indicated by the N_TT field in the HCSPARAMS – Host Control Structural Parameters register.

23.8.2 Device operation

The co-existence of a device operational controller within the host controller has little effect on EHCI compatibility for host operation except as noted in this section.

23.8.2.1 USBMODE register

Given that the dual-role controller is initialized in neither host nor device mode, the USBMODE register must be programmed for host operation before the EHCI host controller driver can begin EHCI host operations.

23.8.2.2 Non-Zero Fields the register file

Some of the reserved fields and reserved addresses in the capability registers and operational register have use in device mode, the following must be adhered to:

- Write operations to all EHCI reserved fields (some of which are device fields) with the operation registers should always be written to zero. This is an EHCI requirement of the device controller driver that must be adhered to.
- Read operations by the host controller must properly mask EHCI reserved fields (some of which are device fields) because fields that are used exclusive for device are undefined in host mode.

23.8.2.3 SOF interrupt

This SOF Interrupt used for device mode is shared as a free running 125us interrupt for host mode. EHCI does not specify this interrupt but it has been added for convenience and as a potential software time base. See USBSTS (Section 23.6.4) and USBINTR (Section 23.6.5) registers.

23.8.3 Miscellaneous variations from EHCI

23.8.3.1 Discovery

23.8.3.1.1 Port reset

The port connect methods specified by EHCI require setting the port reset bit in the PORTSCx register for a duration of 10 ms. Due to the complexity required to support the attachment of devices that are not high speed there are counter already present in the

design that can count the 10ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is then summarized as the following:

- [Port Change Interrupt] Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall write a '1' to the reset the device.
- Software shall write a '0' to the reset the device after 10 ms.

  This step, which is necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI host controller driver attempt to write a '0' to the reset bit while a reset is in progress the write will simple be ignored and the reset will continue until completion.
- [Port Change Interrupt] Port enable change occurs to notify the host controller that the device in now operational and at this point the port speed has been determined.

23.8.3.1.2 Port speed detection

After the port change interrupt indicates that a port is enabled, the EHCI stack should determine the port speed. Unlike the EHCI implementation which will re-assign the port owner for any device that does not connect at High-Speed, this host controller supports direct attach of non High-Speed devices. Therefore, the following differences are important regarding port speed detection:

- Port Owner is read-only and always reads 0.
- A 2-bit Port Speed indicator has been added to PORTSC to provide the current operating speed of the port to the host controller driver.
- A 1-bit High Speed indicator has been added to PORTSC to signify that the port is in High-Speed vs. Full/Low Speed – This information is redundant with the 2-bit Port Speed indicator above.

23.9 Device data structures

This section defines the interface data structures used to communicate control, status, and data between Device Controller Driver (DCD) Software and the Device Controller. The data structure definitions in this chapter support a 32-bit memory buffer address space.

Remark: The Software must ensure that no interface data structure reachable by the Device controller crosses a 4k-page boundary

The data structures defined in the chapter are (from the device controller's perspective) a mix of read-only and read/ writable fields. The device controller must preserve the read-only fields on all data structure writes.

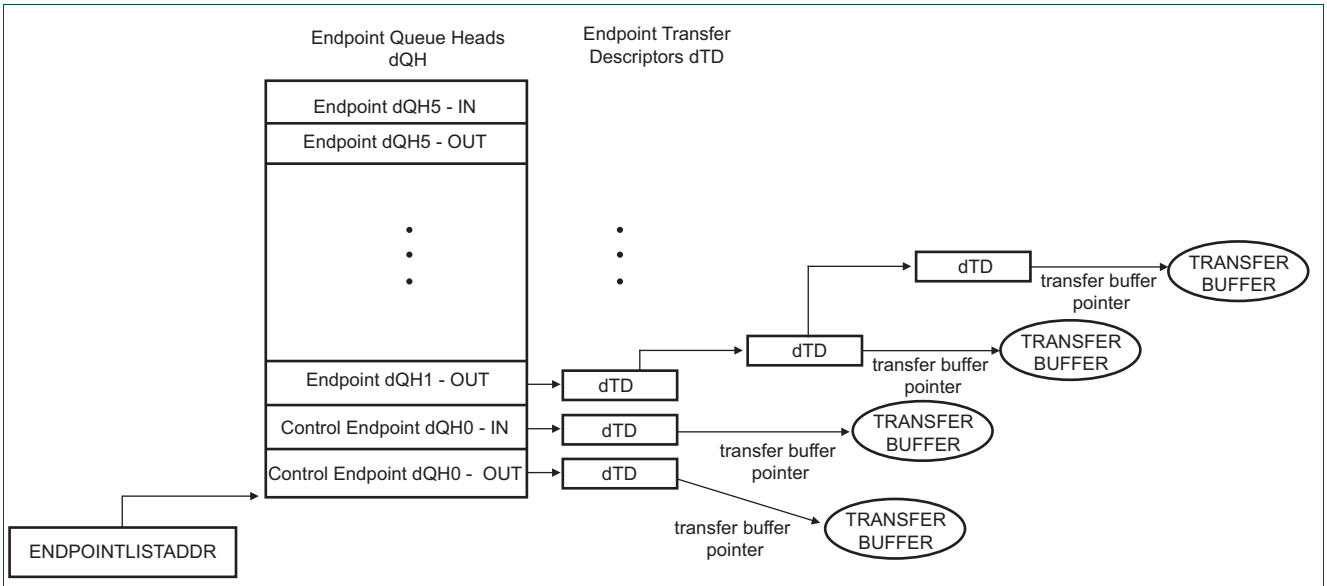


Fig 50. Endpoint queue head organization

Device queue heads are arranged in an array in a continuous area of memory pointed to by the ENDPOINTLISTADDR pointer. The even –numbered device queue heads in the list support receive endpoints (OUT/SETUP) and the odd-numbered queue heads in the list are used for transmit endpoints (IN/INTERRUPT). The device controller will index into this array based upon the endpoint number received from the USB bus. All information necessary to respond to transactions for all primed transfers is contained in this list so the Device Controller can readily respond to incoming requests without having to traverse a linked list.

Remark: The Endpoint Queue Head List must be aligned to a 2 kB boundary.

23.9.1 Endpoint queue head (dQH)

The device Endpoint Queue Head (dQH) is where all transfers are managed. The dQH is a 48-byte data structure, but must be aligned on 64-byte boundaries. During priming of an endpoint, the dTD (device transfer descriptor) is copied into the overlay area of the dQH, which starts at the nextTD pointer DWord and continues through the end of the buffer pointers DWords. After a transfer is complete, the dTD status DWord is updated in the dTD pointed to by the currentTD pointer. While a packet is in progress, the overlay area of the dQH is used as a staging area for the dTD so that the Device Controller can access needed information with little minimal latency.

23.9.1.1 Endpoint capabilities and characteristics

This DWord specifies static information about the endpoint, in other words, this information does not change over the lifetime of the endpoint. Device Controller software should not attempt to modify this information while the corresponding endpoint is enabled.

Table 424. Endpoint capabilities and characteristics

Access	Bit	Name	Description
RO	31:30	MULT	Number of packets executed per transaction descriptor 00 - Execute N transactions as demonstrated by the USB variable length protocol where N is computed using Max_packet_length and the Total_bytes field in the dTD. 01 - Execute one transaction 10 - Execute two transactions 11 - Execute three transactions **Remark:** Non-isochronous endpoints must set MULT = 00. **Remark:** Isochronous endpoints must set MULT = 01, 10, or 11 as needed.
RO	29	ZLT	Zero length termination select This bit is used for non-isochronous endpoints to indicate when a zero-length packet is received to terminate transfers in case the total transfer length is “multiple”. 0 - Enable zero-length packet to terminate transfers equal to a multiple of Max_packet_length (default). 1 - Disable zero-length packet on transfers that are equal in length to a multiple Max_packet_length.
RO	28:27	-	reserved
RO	26:16	Max_packet _length	Maximum packet size of the associated endpoint (< 1024)
RO	15	IOS	Interrupt on setup This bit is used on control type endpoints to indicate if USBINT is set in response to a setup being received.
RO	14:0	-	reserved

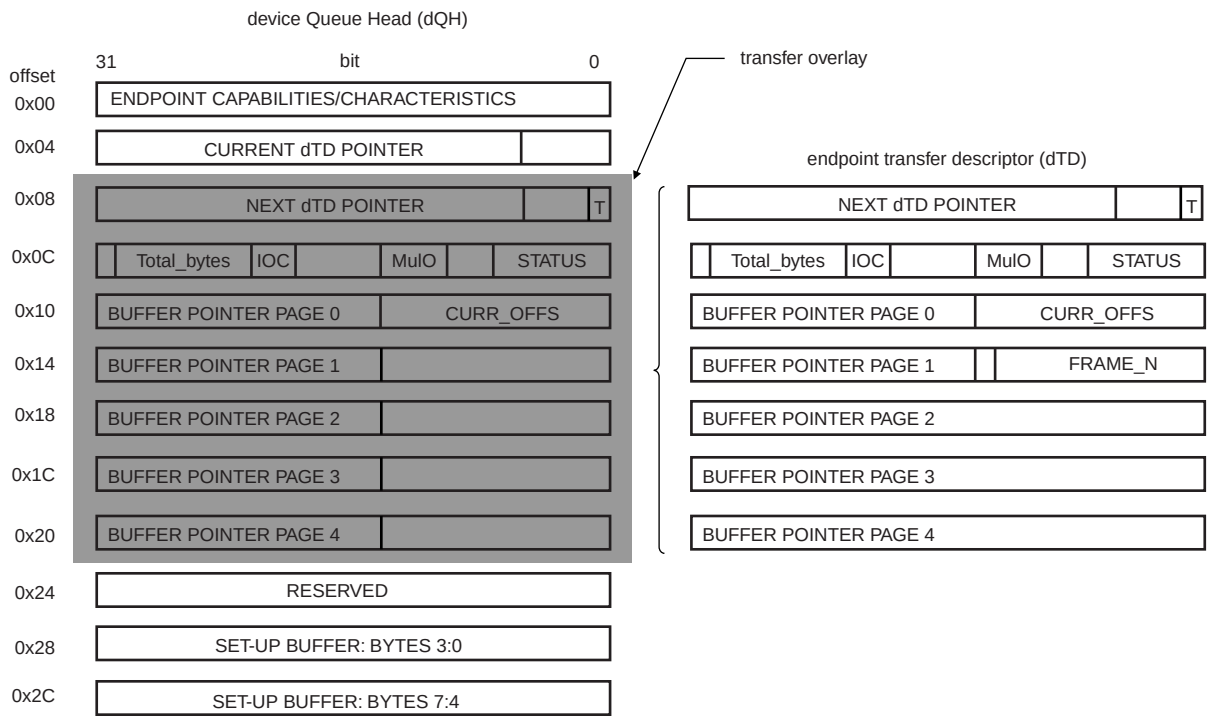


Fig 51. Endpoint queue head data structure

23.9.1.2 Transfer overlay

The seven DWords in the overlay area represent a transaction working space for the device controller. The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it will not read the associated endpoint.

After an endpoint is readied, the dTD will be copied into this queue head overlay area by the device controller. Until a transfer is expired, software must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller will write the results back to the original transfer descriptor and advance the queue. See dTD for a description of the overlay fields.

23.9.1.3 Current dTD pointer

The current dTD pointer is used by the device controller to locate the transfer in progress. This word is for Device Controller (hardware) use only and should not be modified by DCD software.

Table 425. Current dTD pointer

Access	Bit	Name	Description
R/W (hardware only)	31:5	Current_TD_pointer	Current dTD pointer This field is a pointer to the dTD that is represented in the transfer overlay area. This field will be modified by the device controller to the next dTD pointer during endpoint priming or queue advance.
-	4:0	-	reserved

23.9.1.4 Set-up buffer

The set-up buffer is dedicated storage for the 8-byte data that follows a set-up PID.

Remark: Each endpoint has a TX and an RX dQH associated with it, and only the RX queue head is used for receiving setup data packets.

Table 426. Set-up buffer

Dword	Access	Bit	Name	Description
1	R/W	31:0	BUF0	Setup buffer 0 This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller to be read by software.
2	R/W	31:0	BUF1	Setup buffer 1 This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller to be read by software.

23.9.2 Endpoint transfer descriptor (dTD)

The dTD describes to the device controller the location and quantity of data to be sent/received for given transfer. The DCD should not attempt to modify any field in an active dTD except the Next Link Pointer, which should only be modified as described in Section 23.10.11.

Table 427. Next dTD pointer

Access	Bit	Name	Description
RO	31:5	Next_link_pointer	Next link pointer This field contains the physical memory address of the next dTD to be processed. The field corresponds to memory address signals [31:5], respectively.
	4:1	-	reserved
	0	T	Terminate This bit indicates to the device controller when there are no more valid entries in the queue. 1 - pointer is invalid 0 - Pointer is valid, i.e. pointer points to a valid transfer element descriptor.

Table 428. dTD token

Access	Bit	Name	Description
-	31	-	reserved
R/W	30:16	Total_bytes	Total bytes This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction and it is decremented only when the transaction has been completed successfully. The maximum value software can write into this field is 0x5000 (5 x 4 kB) for the maximum number of bytes five page pointers can access. Although it is possible to create a transfer up to 20 kB this assumes that the first offset into the first page is zero. When the offset cannot be predetermined, crossing past the fifth page can be guaranteed by limiting the total bytes to 16 kB. Therefore, the maximum recommended Total-Bytes = 16 kB (0x4000). If Total_bytes = 0 when the host controller fetches this transfer descriptor and the active bit is set in the Status field of this dTD, the device controller executes a zero-length transaction and retires the dTD. **Remark:** For IN transfers, it is not a requirement that Total_bytes is an even multiple of Max_packet_length. If software builds such a dTD, the last transaction will always be less than Max_packet_length.
RO	15	IOC	Interrupt on complete This bit is used to indicate if USBINT will be set when the device controller is finished with this dTD. 1 - USBINT set. 0 - USBINT not set.
-	14:12	-	reserved

Table 428. dTD token *…continued*

Access	Bit	Name	Description
RO	11:10	MultO	Multiplier Override (see Section 23.9.2.1 for an example) This field can be used for transmit ISOs to override the MULT field in the dQH. This field must be zero for all packet types that are not transmit-ISO. 00 - Execute N transactions as demonstrated by the USB variable length protocol where N is computed using Max_packet_length and the Total_bytes field in the dTD. 01 - Execute one transaction 10 - Execute two transactions 11 - Execute three transactions **Remark:** Non-ISO and Non-TX endpoints must set MultO="00".
	9:8	-	reserved
R/W	7:0	Status	Status This field is used by the device controller to communicate individual execution states back to the software. This field contains the status of the last transaction performed on this dTD. Bit 7 = 1 - status: Active Bit 6 = 1 - status: Halted Bit 5 = 1 - status: Buffer Error Bit 4 - reserved Bit 3 = 1 - status: Transaction Error Bit 2 - reserved Bit 1 - reserved Bit 0 - reserved

Table 429. dTD buffer page pointer list

Access	Bit	Name	Description
RO	31:12	BUFF_P	Selects the page offset in memory for the packet buffer. Non-virtual memory systems will typically set the buffer pointers to a series of incrementing integers.
	page 0: 11:0	CURR_OFFS	Offset into the 4 kB buffer where the packet is to begin.
	page 1: 10:0	FRAME_N	Written by the device controller to indicate the frame number in which a packet finishes. This is typically used to correlate relative completion times of packets on an isochronous endpoint.

23.9.2.1 Determining the number of packets for Isochronous IN endpoints

The following examples show how the MULT field in the dQH and the MultO in the dTD are used to control the number of packets sent in an In-transaction for an isochronous endpoint:

Example 1

MULT = 3; Max_packet_size = 8; Total_bytes = 15; MultO = 0 (default)

In this case three packets are sent: Data2 (8 bytes), Data1 (7 bytes), Data0 (0 bytes).

Example 2

MULT = 3; Max_packet_size = 8; Total_bytes = 15; MultO = 2

In this case two packets are sent: Data1 (8 bytes), Data0 (7 bytes).

To optimize efficiency for IN transfers, software should compute MultO = greatest integer of (Total_bytes/Max_packet_size). If Total_bytes = 0, then MultO should be 1.

23.10 Device operational model

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus. Using a set of linked list transfer descriptors, pointed to by a queue head, the device controller will perform the data transfers. The following sections explain the use of the device controller from the device controller driver (DCD) point-of-view and further describe how specific USB bus events relate to status changes in the device controller programmer's interface.

23.10.1 Device controller initialization

After hardware reset, the device is disabled until the Run/Stop bit is set to a '1'. In the disabled state, the pull-up on the USB_DM is not active which prevents an attach event from occurring. At a minimum, it is necessary to have the queue heads setup for endpoint zero before the device attach occurs. Shortly after the device is enabled, a USB reset will occur followed by setup packet arriving at endpoint 0. A Queue head must be prepared so that the device controller can store the incoming setup packet.

In order to initialize a device, the software should perform the following steps:

1. Set Controller Mode in the USBMODE register to device mode.

   Remark: Transitioning from host mode to device mode requires a device controller reset before modifying USBMODE.

2. Allocate and Initialize device queue heads in system memory (see Section 23.9).

   Minimum: Initialize device queue heads 0 Tx & 0 Rx.

   Remark: All device queue heads associated with control endpoints must be initialized before the control endpoint is enabled. Non-Control device queue heads must be initialized before the endpoint is used and not necessarily before the endpoint is enabled.

3. Configure ENDPOINTLISTADDR Pointer (see Section 23.6.8).

4. Enable the microprocessor interrupt associated with the USB-HS core.

   Recommended: enable all device interrupts including: USBINT, USBERRINT, Port Change Detect, USB Reset Received, DCSuspend (see Table 394).

5. Set Run/Stop bit to Run Mode.

   After the Run bit is set, a device reset will occur. The DCD must monitor the reset event and adjust the software state as described in the Bus Reset section of the following Port State and Control section below.

Remark: Endpoint 0 is designed as a control endpoint only and does not need to be configured using ENDPTCTRL0 register.

It is also not necessary to initially prime Endpoint 0 because the first packet received will always be a setup packet. The contents of the first setup packet will require a response in accordance with USB device framework command set (see *USB Specification Rev. 2.0*, *chapter 9*).

23.10.2 Port state and control

From a chip or system reset, the device controller enters the powered state. A transition from the powered state to the attach state occurs when the Run/Stop bit is set to a '1'. After receiving a reset on the bus, the port will enter the defaultFS or defaultHS state in accordance with the reset protocol described in *Appendix C.2 of the USB Specification Rev. 2.0*. The following state diagram depicts the state of a USB 2.0 device.

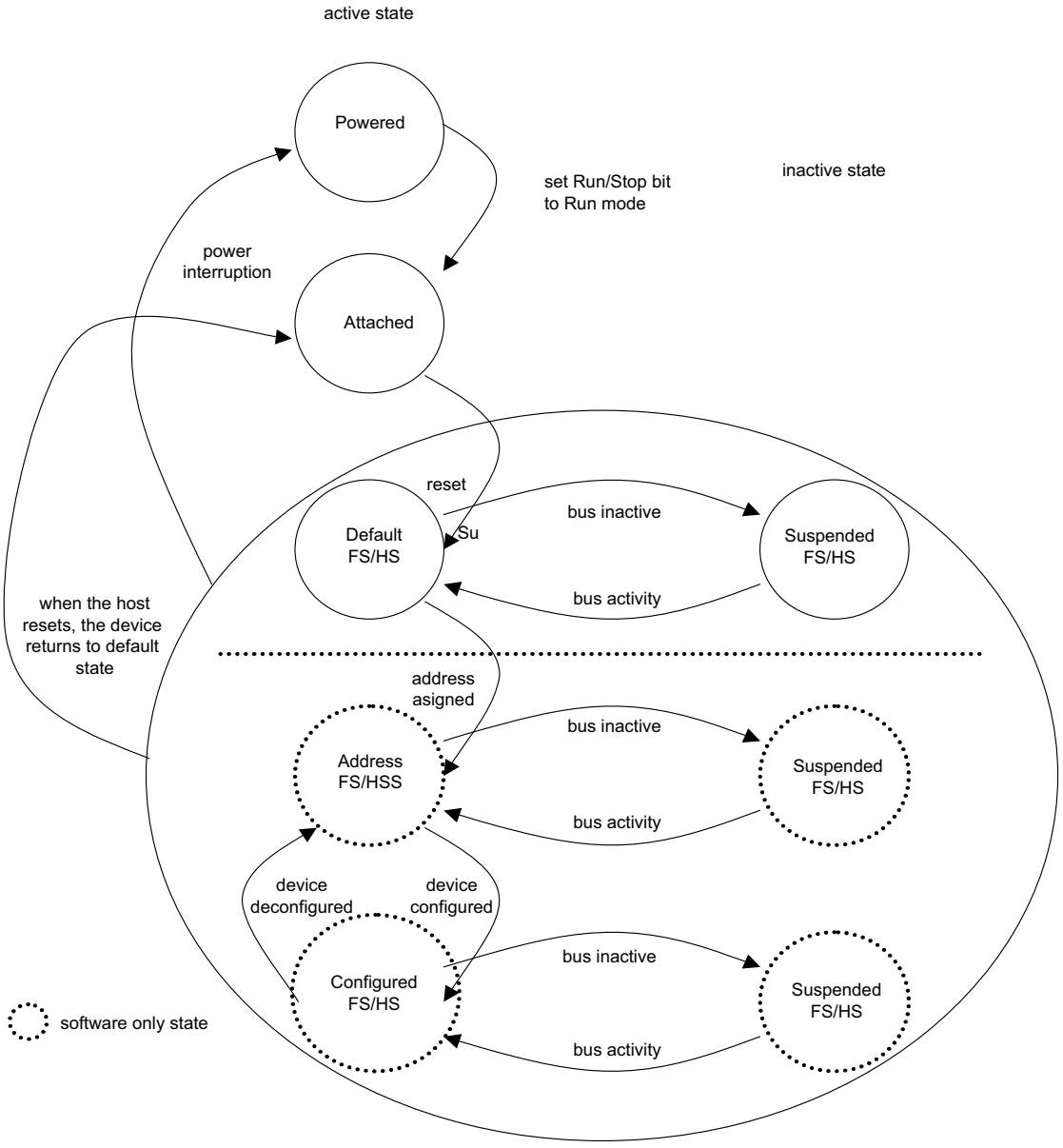


Fig 52. Device state diagram

The states powered, attach, default FS/HS, suspend FS/HS are implemented in the device controller and are communicated to the DCD using the following status bits:

- DCSuspend - see Table 392.
- USB reset received - see Table 392.
- Port change detect - see Table 392.
- High-speed port - see Table 409.

It is the responsibility of the DCD to maintain a state variable to differentiate between the DefaultFS/HS state and the Address/Configured states. Change of state from Default to Address and the configured states is part of the enumeration process described in the *device framework section of the USB 2.0 Specification*.

As a result of entering the Address state, the device address register (DEVICEADDR) must be programmed by the DCD.

Entry into the Configured indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the ENDPTCTRLx registers and initializing the associated queue heads.

23.10.3 Bus reset

A bus reset is used by the host to initialize downstream devices. When a bus reset is detected, the device controller will renegotiate its attachment speed, reset the device address to 0, and notify the DCD by interrupt (assuming the USB Reset Interrupt Enable is set). After a reset is received, all endpoints (except endpoint 0) are disabled and any primed transactions will be cancelled by the device controller. The concept of priming will be clarified below, but the DCD must perform the following tasks when a reset is received:

- Clear all setup token semaphores by reading the ENDPTSETUPSTAT register and writing the same value back to the ENDPTSETUPSTAT register.
- Clear all the endpoint complete status bits by reading the ENDPTCOMPLETE register and writing the same value back to the ENDPTCOMPLETE register.
- Cancel all primed status by waiting until all bits in the ENDPTPRIME are 0 and then writing 0xFFFFFFFF to ENDPTFLUSH.
- Read the reset bit in the PORTSCx register and make sure that it is still active. A USB reset will occur for a minimum of 3 ms and the DCD must reach this point in the reset cleanup before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare).

  Remark: A hardware reset can be performed by writing a one to the device controller reset bit in the USBCMD reset. Note: a hardware reset will cause the device to detach from the bus by clearing the Run/Stop bit. Thus, the DCD must completely re-initialize the device controller after a hardware reset.
- Free all allocated dTDs because they will no longer be executed by the device controller. If this is the first time the DCD is processing a USB reset event, then it is likely that no dTDs have been allocated. At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a Port Change Detect is indicated.
- After a Port Change Detect, the device has reached the default state and the DCD can read the PORTSCx to determine if the device is operating in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the *USB2.0 specification Chapter 9 - Device Framework*.

  Remark: The device DCD may use the FS/HS mode information to determine the bandwidth mode of the device.

In some applications, it may not be possible to enable one or more pipes while in FS mode. Beyond the data rate issue, there is no difference in DCD operation between FS and HS modes.

23.10.4 Suspend/resume

23.10.4.1 Suspend

In order to conserve power, USB devices automatically enter the suspended state when the device has observed no bus traffic for a specified period. When suspended, the USB device maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend at any time they are powered, regardless of if they have been assigned a non-default address, are configured, or neither Bus activity may cease due to the host entering a suspend mode of its own. In addition, a USB device shall also enter the suspended state when the hub port it is attached to is disabled.

A USB device exits suspend mode when there is bus activity. A USB device may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wake-up. The ability of a device to signal remote wake-up is optional. If the USB device is capable of remote wake-up signaling, the device must support the ability of the host to enable and disable this capability. When the device is reset, remote wake-up signaling must be disabled.

23.10.4.1.1 Operational model

The device controller moves into the suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the suspend state, the DCD is notified by an interrupt (assuming DC Suspend Interrupt is enabled). When the DCSuspend bit in the PORTSCx is set to a ‘1’, the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation. Information on the bus power limits in suspend state can be found in *USB 2.0 specification*.

23.10.4.2 Resume

If the device controller is suspended, its operation is resumed when any non-idle signaling is received on its upstream facing port. In addition, the device can signal the system to resume operation by forcing resume signaling to the upstream port. Resume signaling is sent upstream by writing a ‘1’ to the Resume bit in the in the PORTSCx while the device is in suspend state. Sending resume signal to an upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one more devices) back to the active condition.

Remark: Before resume signaling can be used, the host must enable it by using the Set Feature command defined in *device framework (chapter 9) of the USB 2.0 Specification*.

23.10.5 Managing endpoints

The *USB 2.0 specification* defines an endpoint, also called a device endpoint or an address endpoint as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device. The endpoint address is specified by the combination of the endpoint number and the endpoint direction.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a control type data channel used for device discovery and enumeration. Other types of endpoints support by USB include bulk, interrupt, and isochronous. Each endpoint type has specific behavior related to packet response and error handling. More detail on endpoint operation can be found in the *USB 2.0 specification*.

The LPC43xx supports up to six endpoints.

Each endpoint direction is essentially independent and can be configured with differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1- OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0 is, for example, is always a control endpoint and uses the pair of directions.

Each endpoint direction requires a queue head allocated in memory. If the maximum of 4 endpoint numbers, one for each endpoint direction are being used by the device controller, then 8 queue heads are required. The operation of an endpoint and use of queue heads are described later in this document.

23.10.5.1 Endpoint initialization

After hardware reset, all endpoints except endpoint zero are un-initialized and disabled. The DCD must configure and enable each endpoint by writing to configuration bit in the ENDPTCTRLx register (see Table 421). Each 32-bit ENDPTCTRLx is split into an upper and lower half. The lower half of ENDPTCTRLx is used to configure the receive or OUT endpoint and the upper half is likewise used to configure the corresponding transmit or IN endpoint. Control endpoints must be configured the same in both the upper and lower half of the ENDPTCTRLx register otherwise the behavior is undefined. The following table shows how to construct a configuration word for endpoint initialization.

Table 430. Device controller endpoint initialization

Field	Value
Data Toggle Reset	1
Data Toggle Inhibit	0
Endpoint Type	00 - control
	01 - isochronous
	10 - bulk
	11 - interrupt
Endpoint Stall	0

23.10.5.2 Stalling

There are two occasions where the device controller may need to return to the host a STALL:

1. The first occasion is the **functional stall**, which is a condition set by the DCD as described in the *USB 2.0 device framework (chapter 9)*. A functional stall is only used on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the ENDPTCTRLx register associated with the given endpoint and the given direction. In a functional stall condition, the device controller will continue to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.
2. A **protocol stal**l, unlike a function stall, is used on control endpoints is automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD should enable the stall bits (both directions) as a pair. A single write to the ENDPTCTRLx register can ensure that both stall bits are set at the same instant.

Remark: Any write to the ENDPTCTRLx register during operational mode must preserve the endpoint type field (i.e. perform a read-modify-write).

Table 431. Device controller stall response matrix

USB packet	Endpoint STALL bit	Effect on STALL bit	USB response
SETUP packet received by a non-control endpoint.	N/A	None	STALL
IN/OUT/PING packet received by a non-control endpoint.	1	None	STALL
IN/OUT/PING packet received by a non-control endpoint.	0	None	ACK/NAK/NYET
SETUP packet received by a control endpoint.	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint.	1	None	STALL
IN/OUT/PING packet received by a control endpoint.	0	None	ACK/NAK/NYET

23.10.5.3 Data toggle

Data toggle is a mechanism to maintain data coherency between host and device for any given data pipe. For more information on data toggle, refer *to the USB 2.0 specification.*

23.10.5.3.1 Data toggle reset

The DCD may reset the data toggle state bit and cause the data toggle sequence to reset in the device controller by writing a ‘1’ to the data toggle reset bit in the ENDPTCTRLx register. This should only be necessary when configuring/initializing an endpoint or returning from a STALL condition.

23.10.5.3.2 Data toggle inhibit

Remark: This feature is for test purposes only and should never be used during normal device controller operation.

Setting the data toggle Inhibit bit active (‘1’) causes the device controller to ignore the data toggle pattern that is normally sent and accept all incoming data packets regardless of the data toggle state. In normal operation, the device controller checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If Data PID does not match the data toggle state bit maintained by the device controller for that endpoint, the Data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the host controller from re-sending the same packet, the device controller will respond to the error packet by acknowledging it with either an ACK or NYET response.

23.10.6 Operational model for packet transfers

All transactions on the USB bus are initiated by the host and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 Specification. At USB 1.1 Full or Low Speed rates, this turnaround time was significant and the USB 1.1 device controllers were designed so that the device controller could access main memory or interrupt a host protocol processor in order to respond to the USB 1.1 transaction. The architecture of the USB 2.0 device controller must be different because same methods will not meet USB 2.0 High-speed turnaround time requirements by simply increasing clock rate.

A USB host will send requests to the device controller in an order that can not be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3 (transmit direction) is configured as a bulk pipe, then we can expect the host will send IN requests to that endpoint. This device controller is designed in such a way that it can prepare packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host initiated transaction on the bus is referred to as “priming” the endpoint. This term will be used throughout the following documentation to describe the device controller operation so the DCD can be designed properly to use priming. Further, note that the term “flushing” is used to describe the action of clearing a packet that was queued for execution.

23.10.6.1 Priming transmit endpoints

Priming a transmit endpoint will cause the device controller to fetch the device transfer descriptor (dTD) for the transaction pointed to by the device queue head (dQH). After the dTD is fetched, it will be stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to handle a request from the host without the need to follow the linked list, starting at the dQH when the host request is received. After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO is split into virtual channels so that the leading data can be stored for any endpoint up to four endpoints.

After a priming request is complete, an endpoint state of primed is indicated in the ENDPTSTATUS register. For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of High Speed USB. Since only the leading data is stored in the device controller FIFO, it is necessary for the device controller to begin filling in behind leading data after the

transaction starts. The FIFO must be sized to account for the maximum latency that can be incurred by the system memory bus. On the LPC43xx, 128 x 36 bit dual port memory FIFOs are used for each IN endpoint.

23.10.6.2 Priming receive endpoints

Priming receive endpoints is identical to priming of transmit endpoints from the point of view of the DCD. At the device controller the major difference in the operational model is that there is no data movement of the leading packet data simply because the data is to be received from the host. Note as part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

23.10.7 Interrupt/bulk endpoint operational model

The behaviors of the device controller for interrupt and bulk endpoints are identical. All valid IN and OUT transactions to bulk pipes will handshake with a NAK unless the endpoint had been primed. Once the endpoint has been primed, data delivery will commence.

A dTD will be retired by the device controller when the packets described in the transfer descriptor have been completed. Each dTD describes N packets to be transferred according to the USB Variable Length transfer protocol. The formula and table on the following page describe how the device controller computes the number and length of the packets to be sent/received by the USB vary according to the total number of bytes and maximum packet length.

With Zero Length Termination (ZLT) = 0

N = INT(Number Of Bytes/Max. Packet Length) + 1

With Zero Length Termination (ZLT) = 1

N = MAXINT(Number Of Bytes/Max. Packet Length)

Table 432. Variable length transfer protocol example (ZLT = 0)

Bytes (dTD)	Max Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	-
512	256	3	256	256	0
512	512	2	512	0	-

Table 433. Variable length transfer protocol example (ZLT = 1)

Bytes (dTD)	Max Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	-
512	256	2	256	256	-
512	512	1	512	-	-

Remark: The MULT field in the dQH must be set to “00” for bulk, interrupt, and control endpoints.

TX-dTD is complete when all packets described dTD were successfully transmitted.Total bytes in dTD will equal zero when this occurs.

RX-dTD is complete when:

- All packets described in dTD were successfully received. Total bytes in dTD will equal zero when this occurs.
- A short packet (number of bytes < maximum packet length) was received. This is a successful transfer completion; DCD must check Total Bytes in dTD to determine the number of bytes that are remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) OR (total bytes received > total bytes specified). This is an error condition. The device controller will discard the remaining packet, and set the Buffer Error bit in the dTD. In addition, the endpoint will be flushed and the USBERR interrupt will become active.

On the successful completion of the packet(s) described by the dTD, the active bit in the dTD will be cleared and the next pointer will be followed when the Terminate bit is clear. When the Terminate bit is set, the device controller will flush the endpoint/direction and cease operations for that endpoint/direction. On the unsuccessful completion of a packet (see long packet above), the dQH will be left pointing to the dTD that was in error. In order to recover from this error condition, the DCD must properly reinitialize the dQH by clearing the active bit and update the nextTD pointer before attempting to re-prime the endpoint.

Remark: All packet level errors such as a missing handshake or CRC error will be retried automatically by the device controller.

There is no required interaction with the DCD for handling such errors.

23.10.7.1 Interrupt/bulk endpoint bus response matrix

Table 434. Interrupt/bulk endpoint bus response matrix

Token type	STALL	Not primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	n/a	n/a
In	STALL	NAK	Transmit	BS error	n/a
Out	STALL	NAK	Receive and NYET/ACK	n/a	NAK
Ping	STALL	NAK	ACK	n/a	n/a
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

[1] BS error = Force Bit Stuff Error

[2] NYET/ACK – NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

[3] SYSERR – System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

23.10.8 Control endpoint operational model

23.10.8.1 Setup phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase. The device controller will always accept the setup phase unless the setup lockout is engaged.

The setup lockout will engage so that future setup packets are ignored. Lockout of setup packets ensures that while software is reading the setup packet stored in the queue head, that data is not written as it is being read potentially causing an invalid setup packet.

In hardware the setup lockout mechanism can be disabled and a new tripwire type semaphore will ensure that the setup packet payload is extracted from the queue head without being corrupted by an incoming setup packet. This is the preferred behavior because ignoring repeated setup packets due to long software interrupt latency would be a compliance issue.

23.10.8.1.1 Setup Packet Handling using setup lockout mechanism

After receiving an interrupt and inspecting USBMODE to determine that a setup packet was received on a particular pipe:

1. Duplicate contents of dQH.SsetupBuffer into local software byte array.
2. Write '1' to clear corresponding ENDPTSETUPSTAT bit and thereby disabling Setup Lockout (i.e. the Setup Lockout activates as soon as a setup arrives. By writing to the ENDPTSETUPSTAT, the device controller will accept new setup packets.).
3. Process setup packet using local software byte array copy and execute status/handshake phases.

   Remark: After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.
4. Before priming for status/handshake phases ensure that ENDPTSETUPSTAT is ‘0’. The time from writing a ‘1’ to ENDPTSETUPSTAT and reading back a ‘0’ may vary according to the type of traffic on the bus up to nearly a 1ms, however the it is absolutely necessary to ensure ENDPTSETUPSTAT has transitioned to ‘0’ after step 1) and before priming for the status/handshake phases.

Remark: To limit the exposure of setup packets to the setup lockout mechanism (if used), the DCD should designate the priority of responding to setup packets above responding to other packet completions

23.10.8.1.2 Setup Packet Handling using trip wire mechanism

- Disable Setup Lockout by writing ‘1’ to Setup Lockout Mode (SLOM) in USBMODE. (once at initialization). Setup lockout is not necessary when using the tripwire as described below.

  Remark: Leaving the Setup Lockout Mode As ‘0’ will result in pre-2.3 hardware behavior.
- After receiving an interrupt and inspecting ENDPTSETUPSTAT to determine that a setup packet was received on a particular pipe:

  a. Write '1' to clear corresponding bit ENDPTSETUPSTAT.

  b. Duplicate contents of dQH.SetupBuffer into local software byte array.

  c. Write ‘1’ to Setup Tripwire (SUTW) in USBCMD register.

  d. Read Setup TripWire (SUTW) in USBCMD register. (if set - continue; if cleared - go to b).

  e. Write '0' to clear Setup Tripwire (SUTW) in USBCMD register.

f. Process setup packet using local software byte array copy and execute status/handshake phases.

g. Before priming for status/handshake phases ensure that ENDPTSETUPSTAT is '0'.

- A poll loop should be used to wait until ENDPTSETUPSTAT transitions to '0' after step a) above and before priming for the status/handshake phases.
- The time from writing a '1' to ENDPTSETUPSTAT and reading back a '0' is very short (~1-2 us) so a poll loop in the DCD will not be harmful.

Remark: After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

23.10.8.2 Data phase

Following the setup phase, the DCD must create a device transfer descriptor for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet has not been received by reading the ENDPTSETUPSTAT register immediately verifying that the prime had completed. A prime will complete when the associated bit in the ENDPTPRIME register is zero and the associated bit in the ENDPTSTATUS register is a one. If a prime fails, i.e. The ENDPTPRIME bit goes to zero and the ENDPTSTATUS bit is not set, then the prime has failed. This can only be due to improper setup of the dQH, dTD or a setup arriving during the prime operation. If a new setup packet is indicated after the ENDPTPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller will automatically clear the prime status (ENDPTSTATUS) to enforce data coherency with the setup packet.

Remark: The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.

Remark: Error handling of data phase packets is the same as bulk packets described previously.

23.10.8.3 Status phase

Similar to the data phase, the DCD must create a transfer descriptor (with byte length equal zero) and prime the endpoint for the status phase. The DCD must also perform the same checks of the ENDPTSETUPSTAT as described above in the data phase.

Remark: The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.

Remark: Error handling of data phase packets is the same as bulk packets described previously.

23.10.8.4 Control endpoint bus response matrix

Shown in the following table is the device controller response to packets on a control endpoint according to the device controller state.

Table 435. Control endpoint bus response matrix

Token type	Endpoint sate					Setup lockout
	STALL	Not primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	n/a	SYSERR	-
In	STALL	NAK	Transmit	BS error	n/a	n/a
Out	STALL	NAK	Receive and NYET/ACK	n/a	NAK	n/a
Ping	STALL	NAK	ACK	n/a	n/a	n/a
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	ignore

[1] BS error = Force Bit Stuff Error

[2] NYET/ACK – NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

[3] SYSERR – System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

23.10.9 Isochronous endpoint operational model

Isochronous endpoints are used for real-time scheduled delivery of data, and their operational model is significantly different than the host throttled Bulk, Interrupt, and Control data pipes. Real time delivery by the device controller is accomplished by the following:

- Exactly MULT Packets per (micro) Frame are transmitted/received. Note: MULT is a two-bit field in the device Queue Head. The variable length packet protocol is not used on isochronous endpoints.
- NAK responses are not used. Instead, zero length packets are sent in response to an IN request to an unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro) frame. If the ISO-dTD is still active after that frame, then the ISO-dTD will be held ready until executed or canceled by the DCD.

An EHCI compatible host controller uses the periodic frame list to schedule data exchanges to Isochronous endpoints. The operational model for device mode does not use such a data structure. Instead, the same dTD used for Control/Bulk/Interrupt endpoints is also used for isochronous endpoints. The difference is in the handling of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint will become primed only after a SOF is received. After the DCD writes the prime bit, the prime bit will be cleared as usual to indicate to software that the device controller completed a priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD but occurs so that the device controller can match the dTD to a specific (micro) frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro) frame. Once an ISO transaction is started in a (micro) frame it will retire the corresponding dTD when MULT transactions occur or the device controller finds

a fulfillment condition. The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, the device controller will force retire the ISO-dTD and move to the next ISO-dTD.

It is important to note that fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction will stay primed indefinitely. This means it is up to software discard transmit ISO-dTDs that pile up from a failure of the host to move the data. Finally, the last difference with ISO packets is in the data level error handling. When a CRC error occurs on a received packet, the packet is not retried similar to bulk and control endpoints. Instead, the CRC is noted by setting the Transaction Error bit and the data is stored as usual for the application software to sort out.

TX packet retired

- MULT counter reaches zero.
- Fulfillment Error [Transaction Error bit is set].
- # Packets Occurred > 0 AND # Packets Occurred < MULT.

Remark: For TX-ISO, MULT Counter can be loaded with a lesser value in the dTD Multiplier Override field. If the Multiplier Override is zero, the MULT Counter is initialized to the Multiplier in the QH.

RX packet retired

- MULT counter reaches zero.
- Non-MDATA Data PID is received.

  Remark: Exit criteria only valid in hardware version 2.3 or later. Previous to hardware version 2.3, any PID sequence that did not match the MULT field exactly would be flagged as a transaction error due to PID mismatch or fulfillment error.
- Overflow Error:
  - Packet received is > maximum packet length. [Buffer Error bit is set].
  - Packet received exceeds total bytes allocated in dTD. [Buffer Error bit is set].
- Fulfillment error [Transaction Error bit is set]:

  # Packets Occurred > 0 AND # Packets Occurred < MULT.
- CRC Error [Transaction Error bit is set]

Remark: For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro) frame to (micro) frame operation the DCD should ensure that the dTD linked-list is out ahead of the device controller by at least two (micro) frames.

23.10.9.1 Isochronous pipe synchronization

When it is necessary to synchronize an isochronous data pipe to the host, the (micro) frame number (FRINDEX register) can be used as a marker. To cause a packet transfer to occur at a specific (micro) frame number [N], the DCD should interrupt on SOF during frame N-1. When the FRINDEX=N–1, the DCD must write the prime bit. The device controller will prime the isochronous endpoint in (micro) frame N–1 so that the device controller will execute delivery during (micro) frame N.

Remark: Priming an endpoint towards the end of (micro) frame N-1 will not guarantee delivery in (micro) frame N. The delivery may actually occur in (micro) frame N+1 if device controller does not have enough time to complete the prime before the SOF for packet N is received.

23.10.9.2 Isochronous endpoint bus response matrix

Table 436. Isochronous endpoint bus response matrix

Token type	STALL	Not primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	n/a	n/a
In	NULL packet	NULL packet	Transmit	BS error	n/a
Out	Ignore	Ignore	Receive	n/a	Drop packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

[1] BS error = Force Bit Stuff Error

[2] NULL packet = Zero length packet.

23.10.10 Managing queue heads

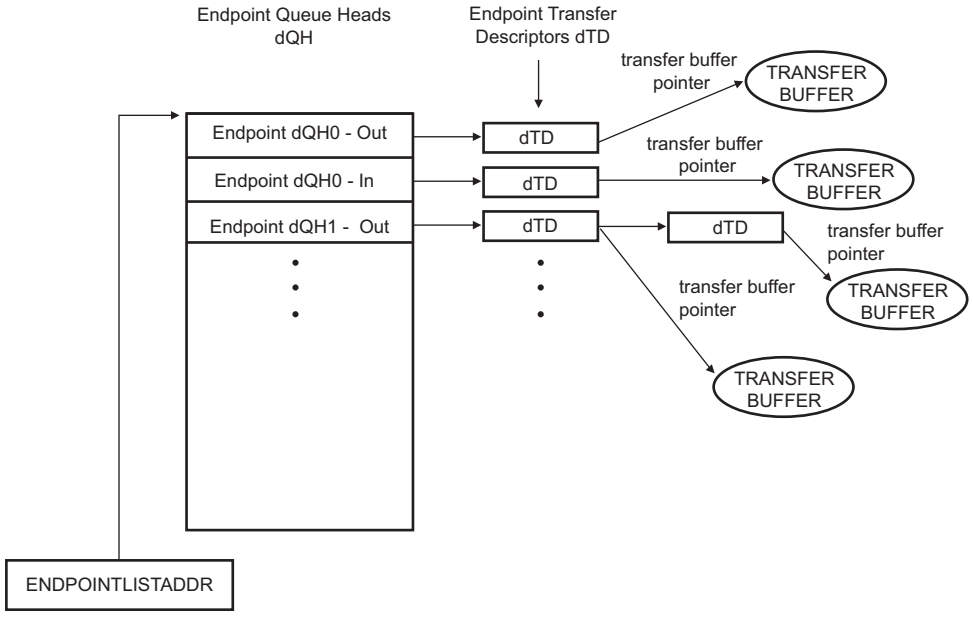


Fig 53. Endpoint queue head diagram

The device queue head (dQH) points to the linked list of transfer tasks, each depicted by the device Transfer Descriptor (dTD). An area of memory pointed to by ENDPOINTLISTADDR contains a group of all dQH's in a sequential list as shown Figure 53. The even elements in the list of dQH's are used for receive endpoints (OUT/SETUP) and the odd elements are used for transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. Once the dTD has been retired, it will no longer be part of the linked list

from the queue head. Therefore, software is required to track all transfer descriptors since pointers will no longer exist within the queue head once the dTD is retired (see Section 23.10.11.1).

In addition to the current and next pointers and the dTD overlay examined in section Operational Model For Packet Transfers, the dQH also contains the following parameters for the associated endpoint: Multiplier, Maximum Packet Length, Interrupt On Setup. The complete initialization of the dQH including these fields is demonstrated in the next section.

23.10.10.1 Queue head initialization

One pair of device queue heads must be initialized for each active endpoint. To initialize a device queue head:

- Write the wMaxPacketSize field as required by the *USB Chapter 9* or application specific protocol.
- Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1,2, or 3 as required bandwidth and in conjunction with the *USB Chapter 9 protocol*. Note: In FS mode, the multiplier field can only be 1 for ISO endpoints.
- Write the next dTD Terminate bit field to “1”.
- Write the Active bit in the status field to “0”.
- Write the Halt bit in the status field to “0”.

Remark: The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTD’s.

23.10.10.2 Operational model for setup transfers

As discussed in section Control Endpoint Operational Model (Section 23.10.8), setup transfer requires special treatment by the DCD. A setup transfer does not use a dTD but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD should handle the setup transfer as demonstrated here:

1. Copy setup buffer contents from dQH - RX to software buffer.
2. Acknowledge setup backup by writing a “1” to the corresponding bit in ENDPTSETUPSTAT.

   Remark: The acknowledge must occur before continuing to process the setup packet.

   Remark: After the acknowledge has occurred, the DCD must not attempt to access the setup buffer in the dQH – RX. Only the local software copy should be examined.
3. Check for pending data or status dTD’s from previous control transfers and flush if any exist as discussed in section Flushing/De-priming an Endpoint.

   Remark: It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.
4. Decode setup packet and prepare data phase [optional] and status phase transfer as require by the *USB Specification Chapter 9* or application specific protocol.

23.10.11 Managing transfers with transfer descriptors

23.10.11.1 Software link pointers

It is necessary for the DCD software to maintain head and tail pointers to the linked list of dTDs for each respective queue head. This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD to be executed. The operations described in next section for managing dTD will assume the DCD can use reference the head and tail of the dTD linked list.

Remark: To conserve memory, the reserved fields at the end of the dQH can be used to store the Head & Tail pointers but it still remains the responsibility of the DCD to maintain the pointers.

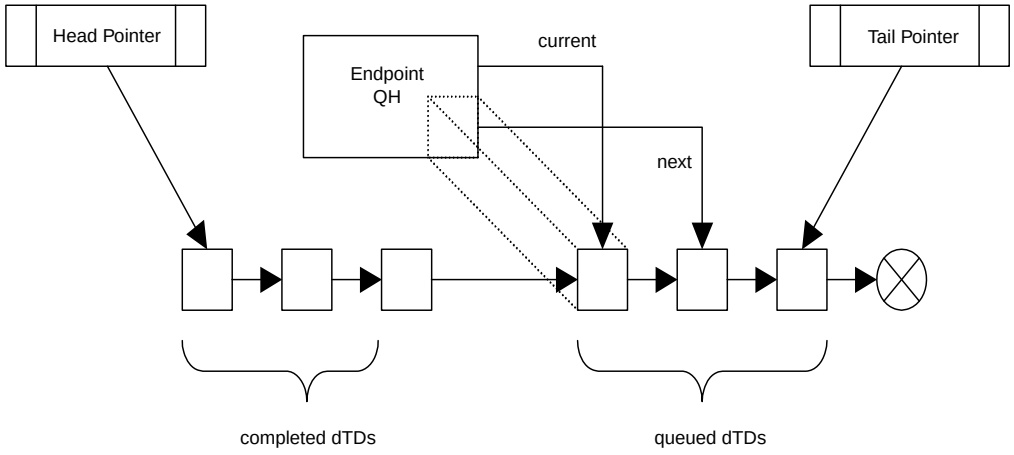


Fig 54. Software link pointers

23.10.11.2 Building a transfer descriptor

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer. Use the following procedure for building dTDs:

Allocate 8-DWord dTD block of memory aligned to 8-DWord boundaries. Example: bit address 4:0 would be equal to “00000”.

Write the following fields:

1. Initialize first 7 DWords to 0.
2. Set the terminate bit to “1”.
3. Fill in total bytes with transfer size.
4. Set the interrupt on complete if desired.
5. Initialize the status field with the active bit set to “1” and all remaining status bits set to “0”.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointer.

23.10.11.3 Executing a transfer descriptor

To safely add a dTD, the DCD must follow this procedure which will handle the event where the device controller reaches the end of the dTD list at the same time a new dTD is being added to the end of the list.

Determine whether the link list is empty: Check DCD driver to see if pipe is empty (internal representation of linked-list should indicate if any packets are outstanding).

Link list is empty

1. Write dQH next pointer AND dQH terminate bit to 0 as a single DWord operation.
2. Clear active and halt bits in dQH (in case set from a previous error).
3. Prime endpoint by writing ‘1’ to correct bit position in ENDPTPRIME.

Link list is not empty

1. Add dTD to end of the linked list.
2. Read correct prime bit in ENDPTPRIME – if ‘1’ DONE.
3. Set ATDTW bit in USBCMD register to ‘1’.
4. Read correct status bit in ENDPTSTAT. (Store in temp variable for later).
5. Read ATDTW bit in USBCMD register.
   - If ‘0’ go to step 3.
   - If ‘1’ continue to step 6.
6. Write ATDTW bit in USBCMD register to ‘0’.
7. If status bit read in step 4 (ENDPSTAT reg) indicates endpoint priming is DONE (corresponding ERBRx or ETBRx is one): DONE.
8. If status bit read in step 4 is 0 then go to Linked list is empty: Step 1.

23.10.11.4 Transfer completion

After a dTD has been initialized and the associated endpoint primed the device controller will execute the transfer upon the host-initiated request. The DCD will be notified with a USB interrupt if the Interrupt On Complete bit was set or alternately, the DCD can poll the endpoint complete register to find when the dTD had been executed. After a dTD has been executed, DCD can check the status bits to determine success or failure.

Remark: Multiple dTD can be completed in a single endpoint complete notification. After clearing the notification, DCD must search the dTD linked list and retire all dTDs that have finished (Active bit cleared).

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

Active = 0

Halted = 0

Transaction Error = 0

Data Buffer Error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in the Device Error Matrix (see Table 437).

In addition to checking the status bit, the DCD must read the Transfer Bytes field to determine the actual bytes transferred. When a transfer is complete, the Total Bytes transferred is decremented by the actual bytes transferred. For Transmit packets, a packet is only complete after the actual bytes reaches zero, but for receive packets, the host may send fewer bytes in the transfer according the USB variable length packet protocol.

23.10.11.5 Flushing/De-priming an endpoint

It is necessary for the DCD to flush to de-prime one more endpoints on a USB device reset or during a broken control transfer. There may also be application specific requirements to stop transfers in progress. The following procedure can be used by the DCD to stop a transfer in progress:

1. Write a ‘1’ to the corresponding bit(s) in ENDPTFLUSH.
2. Wait until all bits in ENDPTFLUSH are ‘0’.

   Remark: Software note: This operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.
3. Read ENDPTSTAT to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now ‘0’. If the corresponding bits are ‘1’ after step #2 has finished, then the flush failed as described in the following:

   In very rare cases, a packet is in progress to the particular endpoint when commanded flush using ENDPTFLUSH. A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush by repeating steps 1-3 until each endpoint is successfully flushed.

23.10.11.6 Device error matrix

The Table 437 summarizes packet errors that are not automatically handled by the Device Controller.

The following errors can occur:

Overflow: Number of bytes received exceeded max. packet size or total buffer length. This error will also set the Halt bit in the dQH, and if there are dTDs remaining in the linked list for the endpoint, then those will not be executed.

ISO packet error: CRC Error on received ISO packet. Contents not guaranteed to be correct.

ISO fulfillment error: Host failed to complete the number of packets defined in the dQH mult field within the given (micro) frame. For scheduled data delivery the DCD may need to readjust the data queue because a fulfillment error will cause Device Controller to cease data transfers on the pipe for one (micro) frame. During the “dead” (micro) frame, the Device Controller reports error on the pipe and primes for the following frame

Table 437. Device error matrix

Error	Direction	Packet type	Data buffer error bit	Transaction error bit
Overflow	Rx	Any	1	0
ISO packet error	Rx	ISO	0	1
ISO fulfillment error	Both	ISO	0	1

23.10.12 Servicing interrupts

The interrupt service routine must consider that there are high-frequency, low-frequency operations, and error operations and order accordingly.

23.10.12.1 High-frequency interrupts

High frequency interrupts in particular should be handed in the order below. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

Table 438. High-frequency interrupt events

Execution order	Interrupt	Action
1a	USB interrupt: ENDPTSETUPSTATUS [1]	Copy contents of setup buffer and acknowledge setup packet (as indicated in Section 23.10.10). Process setup packet according to *USB 2.0 Chapter 9* or application specific protocol.
1b	USB interrupt: ENDPTCOMPLETE[1]	Handle completion of dTD as indicated in Section 23.10.10.
2	SOF interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

[1] It is likely that multiple interrupts to stack up on any call to the Interrupt Service Routine AND during the Interrupt Service Routine.

23.10.12.2 Low-frequency interrupts

The low frequency events include the following interrupts. These interrupt can be handled in any order since they don't occur often in comparison to the high-frequency interrupts.

Table 439. Low-frequency interrupt events

Interrupt	Action
Port change	Change software state information.
Sleep enable (Suspend)	Change software state information. Low power handling as necessary.
Reset Received	Change software state information. Abort pending transfers.

23.10.12.3 Error interrupts

Error interrupts will be least frequent and should be placed last in the interrupt service routine.

Table 440. Error interrupt events

Interrupt	Action
USB error interrupt	This error is redundant because it combines USB Interrupt and an error status in the dTD. The DCD will more aptly handle packet-level errors by checking dTD status field upon receipt of USB Interrupt (w/ ENDPTCOMPLETE).
System error	Unrecoverable error. Immediate Reset of core; free transfers buffers in progress and restart the DCD.

23.11 USB power optimization

The USB-HS core is a fully synchronous static design. The power used by the design is dependent on the application usage of the core. Applications that transfer more data or use a greater number of packets to be sent will consume a greater amount of power.

The USB power consumption can be controlled by disabling the USB clocks and disabling the High-speed PHY (see Section 23.1).

A device may suspend operations autonomously by disconnecting from the USB, or, in response to the suspend signaling, the USB has moved it into the suspend state. A host can suspend operation autonomously, or it can command portions or the entire USB to transition into the suspend state.

23.11.1 USB power states

The USB provides a mechanism to place segments of the USB or the entire USB into a low-power suspend state. USB bus powered devices are required to respond to a 3ms lack of activity on the USB bus by going into a suspend state. In the USB-HS core software is notified of the suspend condition via the transition in the PORTSC register. Optionally an interrupt can be generated which is controlled by the port change Detect Enable bit in the USBINTR control register. Software then has 7 ms to transition a bus powered device into the suspend state. In the suspend state, a USB device has a maximum USB bus power budget of 500μA. In general, to achieve that level of power conservation, most of the device circuits will need to be switched off, or clock at an extremely low frequency. This can be accomplished by suspending the clock.

The implementation of low power states in the USB-HS core is dependant on the use of the device role (host or peripheral), whether the device is bus powered, and the selected clock architecture of the core.

Bus powered peripheral devices are required by the USB specification to support a low power suspend state. Self powered peripheral devices and hosts set their own power management strategies based on their system level requirements. The clocking architecture selected is important to consider as it determines what portions of the design will remain active when transitioned into the low power state.

Before the system clock is suspended or set to a frequency that is below the operational frequency of the USB-HS core, the core must be moved from the operational state to a low power state. The power strategies designed into the USB-HS core allow for the most challenging case, a self powered device that is clocked entirely by the transceiver clock.

23.11.2 Device power states

A bus powered peripheral device must move through the power states as directed by the host. Optionally autonomously directed low power states may be implemented.

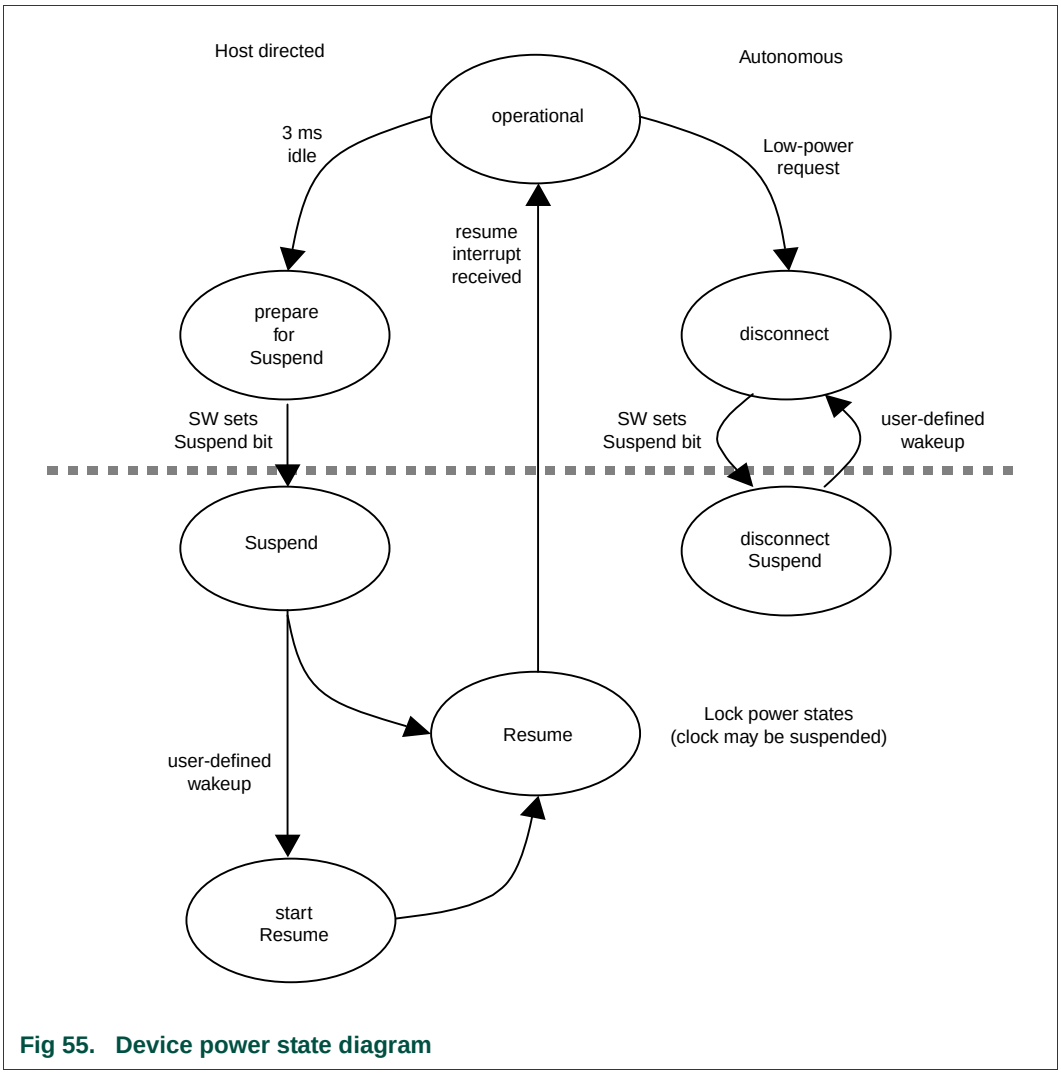


Fig 55. Device power state diagram

In the operational state both the transceiver clock and system clocks are running. Software can initiate a low power mode autonomously by disconnecting from the host to go into the disconnect state. Once in this state, the software can set the Suspend bit to turn off the transceiver clock putting the system in to the disconnect-suspend state. Since software cannot depend on the presents of a clock to clear the Suspend bit, a wake-up event must be defined which would clear the Suspend bit and allow the transceiver clock to resume.

The device can also go into suspend mode as a result of a suspend command from the host. Suspend is signaled on the bus by 3ms of idle time on the bus. This will generate a suspend interrupt to the software at which point the software must prepare to go into suspend then set the suspend bit. Once the Suspend bit is set the transceiver clock may turn off and the device will be in the suspended state. The device has two ways of getting out of suspend.

1. If remote wake-up is enabled, a wake-up event could be defined which would clear the Suspend bit. The software would then initiate the resume by setting the Resume bit in the port controller then waiting for a port change interrupt indicating that the port is in an operational state.
2. If the host puts resume signaling on the bus, it will clear the Suspend bit and generate a port change interrupt when the resume is finished.

23.11.3 Host power states

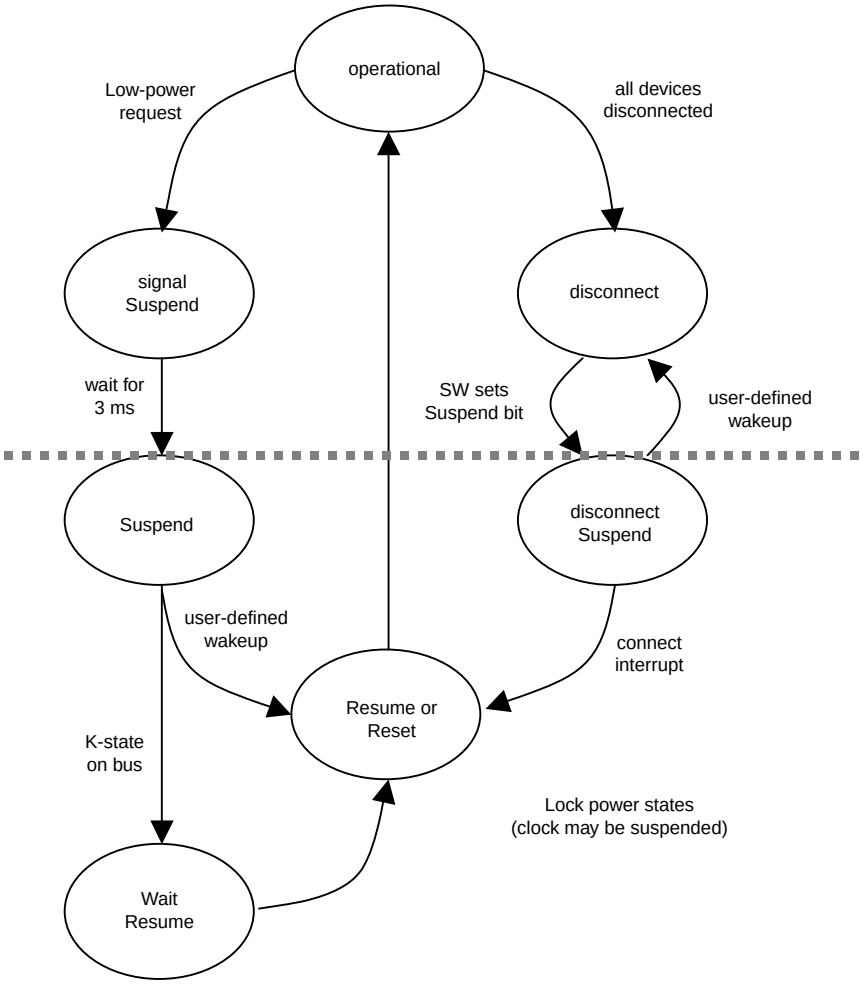


Fig 56. Host/OTG power state diagram

From an operational state when a host gets a low power request, it must set the suspend bit in the port controller. This will put an idle on the bus, block all traffic through the port, and turn off the transceiver clock. There are two ways for a host controller to get out of the suspend state. If it has enabled remote wake-up, a K-state on the bus will turn the transceiver clock and generate an interrupt. The software will then have to wait 20 ms for the resume to complete and the port to go back to an active state. Alternatively an external event could clear the suspend bit and start the transceiver clock running again. The software can then initiate a resume by setting the resume bit in the port controller, or force a reconnect by setting the reset bit in the port controller.

If all devices have disconnected from the host, the host can go into a low power mode by the software setting the suspend bit. From the disconnect-suspend state a connect event would start the transceiver clock and interrupt the software. The software would then need to set the reset bit to start the connect process.

23.11.4 Susp_CTRL module

The SUSP_CTRL module implements the power management logic of USB-OTG. It controls the suspend input of the transceiver. Asserting this suspend signal will put the transceiver in suspend mode and the generation of the 30 MHz clock and 60 MHz clock will be switched off.

A suspend control input of the transceiver (otg_on) that was previously tied high and prevented the transceiver to go into full suspend mode, has been connected to <tbd>. This bit is low by default and only needs to be set high in OTG Host mode operation.

In suspend mode, the transceiver will raise an output signal indicating that the PLL generating the 480 MHz clock can be switched off.

The SUSP_CTRL module also generates an output signal indicating whether the AHB clock is needed or not. If '0' the AHB clock is allowed to be switched off or reduced in frequency in order to save power.

The core will enter the low power state if:

- Software sets the PORTSC.PHCD bit.

When operating in host mode, the core will leave the low power state on one of the following conditions:

- software clears the PORTSC.PHCD bit
- a device is connected and the PORTSC.WKCN bit is set
- a device is disconnected an the PORTSC.WKDC bit is set
- an over-current condition occurs and the PORTSC.WKOC bit is set
- a remote wake-up from the attached device occurs (when USB bus was in suspend)
- a change on vbusvalid occurs (= VBUS threshold at 4.4 V is crossed)
- a change on bvalid occurs (=VBUS threshold at 4.0 V is crossed).

When operating in device mode, the core will leave the low power state on one of the following conditions:

- software clears the PORTSC.PHCD bit.
- a change on the USB data lines (dp/dm) occurs.
- a change on vbusvalid occurs (= VBUS threshold at 4.4 V is crossed).
- a change on bvalid occurs (= VBUS threshold at 4.0 V is crossed).

The vbusvalid and bvalid signals coming from the transceiver are not filtered in the SUSP_CTRL module. Any change on those signals will cause a wake-up event.

Input signals 'host_wakeup_n' and 'dev_wakeup_n' are extra external wake-up signals (for host mode and device mode respectively). However the detection of all USB related wake-up events is already handled in the SUSP_CTRL mode. Therefore in normal situations these signals can be tied high (= inactive).

24.1 How to read this chapter

The USB1 Host/Device controller is available on parts LPC4350 and LPC4330.

24.2 Basic configuration

The USB1 controller is configured as follows:

- See Table 441 for clocking and power control.
- The USB1 is reset by a USB1_RST (reset # 18).
- The USB1 <tbd> interrupt is connected to interrupt slot # 9 in the NVIC. The USB wake-up interrupt is connected to slot # 10 in the event router.
- In the SFSUSB register, the USB_ESEA bit must be set to 1 for the USB1 to operate (see Table 126).

Table 441. USB1 clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
USB1 clock	BASE_USB1_CLK	CLK_USB1	60 MHz	Uses PLL1. CLK_USB1 must be 60 MHz when the USB1 is operated IN low-speed and full-speed modes. In high-speed mode, the clock is provided by the ULPI PHY.
USB1 register interface clock	BASE_M4_CLK	CLK_M4_USB1	up to 204 MHz	

24.2.1 Full-speed mode without external PHY

In Full-speed mode, use CLK_USB1 to generate a clock for the USB1 interface.

24.2.2 High-speed mode with ULPI interface

In High-speed mode, the external PHY generates the clock for the USB1 interface, and the USB1_ULPI_CLK must be enabled on pins PC_0 or P8_8 through their respective pin configuration registers in the system configuration block. The USB1 branch clock CLK_USB1 must be disabled.

24.3 Features

- Supports all high-speed USB-compliant peripherals if connected to external ULPI PHY.
- Supports all full-speed USB-compliant peripherals.
- Complies with *Universal Serial Bus specification 2.0*.
- Complies with *Enhanced Host Controller Interface Specification*.

- Supports auto USB 2.0 mode discovery.
- Supports three logical endpoints plus one control endpoint for a total of 8 physical endpoints.
- This module has its own, integrated DMA engine.

24.4 General description

The USB1 Controller is a peripheral for embedded applications containing digital circuitry to provide USB2.0 On-The-Go functionality.

USB2.0 provides plug-and-play connection of peripheral devices to a host with three different data speeds: High-Speed with a data rate of 480 Mbps, Full-Speed with a data rate of 12 Mbps, Low-Speed with a data rate of 1.5 Mbps. Many portable devices can benefit from the ability to communicate to each other over the USB interface without intervention of a host PC. The addition of the On-The-Go functionality to USB makes this possible without losing the benefits of the standard USB protocol.

Support of the High-Speed data rate requires an external USB HS OTG PHY that connects to the USB controller via the ULPI interface. Full-Speed or Low-Speed is supported through the on-chip Full-speed PHY.

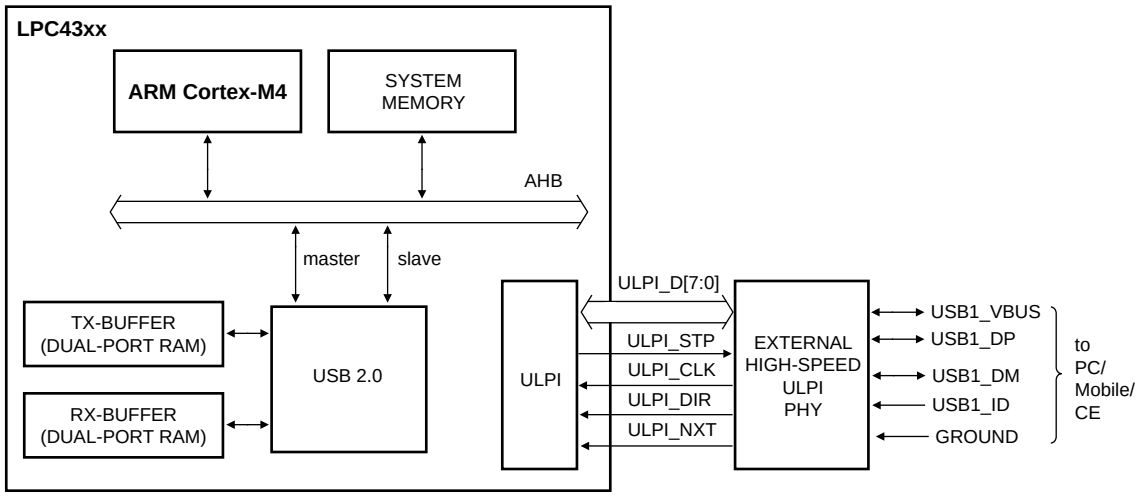


Fig 57. USB1 block diagram with ULPI

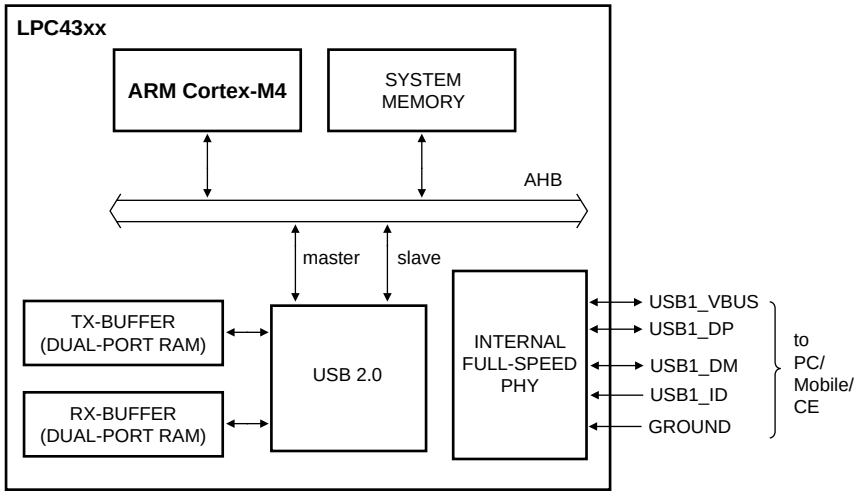


Fig 58. USB1 block diagram with internal full-speed PHY

24.5 Pin description

Table 442. USB1 pin description

Function name	Direction	Description
USB1_DP	I/O	USB1 bidirectional D+ line. The D+ line has an internal 1.5 kΩ pull-up. This pull-up is enabled when software sets the RS bit (Bit 0) in the USBCMD register. The USB1 controller checks whether the USB1_VBUS pin is pulled HIGH (there is no voltage monitoring). For applications which use the USB1_VBUS pin as GPIO, software can monitor VBUS through bit 5 in SFSUSB register (0x4008 6C80).
USB1_DM	I/O	USB1 bidirectional D− line. The D- line has an internal 1.5 kΩ pull-up. This pull-up is enabled when software sets the RS bit (Bit 0) in the USBCMD register. The USB1 controller checks whether the USB1_VBUS pin is pulled HIGH (there is no voltage monitoring). For applications which use the USB1_VBUS pin as GPIO, software can monitor VBUS through bit 5 in SFSUSB register (0x4008 6C80).
USB1_VBUS	I	VBUS pin (power on USB cable). If the USB1_VBUS function is not connected on pin P2_5, use the USB_VBUS bit in the SFSUSB register Table 126 to indicate the VBUS state to the USB1 controller.
USB1_PPWR	O	VBUS drive signal (towards external charge pump or power management unit); indicates that VBUS must be driven (active HIGH). Add a pull-down resistor to disable the power switch at reset. This signal has opposite polarity compared to the USB_PPWR used on other NXP LPC parts.
USB1_IND0	O	Port indicator LED control output 0.
USB1_IND1	O	Port indicator LED control output 1.
USB1_PWR_FAULT	I	Port power fault signal indicating over-current condition; this signal monitors over-current on the USB bus (external circuitry required to detect over-current condition).
ULPI pins		
ULPI_DATA[7:0]	I/O	ULPI link 8-bit bidirectional data bus timed on the rising clock edge.
ULPI_STP	O	ULPI link STP signal. Asserted to end or interrupt transfers to the PHY.

Table 442. USB1 pin description

Function name	Direction	Description
ULPI_NXT	I	ULPI link NXT signal. Data flow control signal from the PHY.
ULPI_DIR	I	ULPI link DIR signal. Controls the DATA bus direction.
ULPI_CLK	I	ULPI link CLK signal. 60 MHz clock generated by the PHY.

24.6 Register description

Remark: For Full-speed operation with on-chip Full-speed PHY, the pads of the PHY need to be configured. For configuration of these pads see Table 126.

Remark: For operations with an external PHY connected through the ULPI interface the interface needs to be selected in the PTS bits of the PORTSC1 register (Section 24.6.15).

Table 443. Register access abbreviations

Abbreviation	Description
R/W	Read/Write
R/WC	Read/Write one to Clear
R/WO	Read/Write Once
RO	Read Only
WO	Write Only

Table 444. Register overview: USB1 host/device controller (register base address 0x4000 7000)

Name	Access	Address offset	Description	Reset value	Reference
-	-	0x000 - 0x0FF	Reserved		
Device/host capability registers					
CAPLENGTH	RO	0x100	Capability register length	0x0001 0040	Table 445
HCSPARAMS	RO	0x104	Host controller structural parameters	0x0001 0011	Table 446
HCCPARAMS	RO	0x108	Host controller capability parameters	0x0000 0005	Table 447
DCIVERSION	RO	0x120	Device interface version number	0x0000 0001	Table 448
DCCPARAMS	RO	0x124	Device controller capability parameters	0x0000 0184	Table 449
-	-	0x128 - 0x13C	Reserved	-	-
Device/host operational registers					
USBCMD_D	R/W	0x140	USB command (device mode)	0x0004 0000	Table 450
USBCMD_H	R/W	0x140	USB command (host mode)	0x0004 00B0	Table 451
USBSTS_D	R/W	0x144	USB status (device mode)	0x0000 0000	Table 453
USBSTS_H	R/W	0x144	USB status (host mode)	0x0000 1000	Table 454
USBINTR_D	R/W	0x148	USB interrupt enable (device mode)	0x0000 0000	Table 455
USBINTR_H	R/W	0x148	USB interrupt enable (host mode)	0x0000 0000	Table 456
FRINDEX_D	RO	0x14C	USB frame index (device mode)	0x0000 0000	Table 457
FRINDEX_H	R/W	0x14C	USB frame index (host mode)	0x0000 0000	Table 458
-	-	0x150	Reserved		-
DEVICEADDR	R/W	0x154	USB device address	0x0000 0000	Table 460

Table 444. Register overview: USB1 host/device controller (register base address 0x4000 7000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
PERIODICLISTBASE	R/W	0x154	Frame list base address	0x0000 0000	Table 461
ENDPOINTLISTADDR	R/W	0x158	Address of endpoint list in memory	0x0000 0000	Table 462
ASYNCLISTADDR	R/W	0x158	Asynchronous list address	0x0000 0000	Table 463
TTCTRL	R/W	0x15C	Asynchronous buffer status for embedded TT (host mode)	0x0000 0000	Table 464
BURSTSIZE	R/W	0x160	Programmable burst size	0x0000 0000	Table 465
TXFILLTUNING	R/W	0x164	Host transmit pre-buffer packet tuning (host mode)	0x0000 0000	Table 466
-	-	0x168 - 0x16C	Reserved	-	-
ULPIVIEWPORT	R/W	0x170	ULPI viewport	0x0000 0000	Table 467
BINTERVAL	R/W	0x174	Length of virtual frame	0x0000 0000	Table 468
ENDPTNAK	R/W	0x178	Endpoint NAK (device mode)	0x0000 0000	Table 469
ENDPTNAKEN	R/W	0x17C	Endpoint NAK Enable (device mode)	0x0000 0000	Table 470
-	-	0x180	-	-	-
PORTSC1_D	R/W	0x184	Port 1 status/control (device mode)	0x0000 0000	Table 471
PORTSC1_H	R/W	0x184	Port 1 status/control (host mode)	0x0000 0000	Table 472
-	-	0x188 - 0x1A0	-	-	-
-	-	0x1A4	-	-	-
USBMODE_D	R/W	0x1A8	USB mode (device mode)	0x0000 0000	Table 474
USBMODE_H	R/W	0x1A8	USB mode (host mode)	0x0000 0000	Table 475
Device endpoint registers					
ENDPTSETUPSTAT	R/W	0x1AC	Endpoint setup status	0x0000 0000	Table 476
ENDPTPRIME	R/W	0x1B0	Endpoint initialization	0x0000 0000	Table 477
ENDPTFLUSH	R/W	0x1B4	Endpoint de-initialization	0x0000 0000	Table 478
ENDPTSTAT	RO	0x1B8	Endpoint status	0x0000 0000	Table 479
ENDPTCOMPLETE	R/W	0x1BC	Endpoint complete	0x0000 0000	Table 480
ENDPTCTRL0	R/W	0x1C0	Endpoint control 0	0x0000 0000	Table 481
ENDPTCTRL1	R/W	0x1C4	Endpoint control 1	0x0000 0000	Table 482
ENDPTCTRL2	R/W	0x1C8	Endpoint control 2	0x0000 0000	Table 482
ENDPTCTRL3	R/W	0x1CC	Endpoint control 3	0x0000 0000	Table 482

24.6.1 Device/host capability registers

Table 445. CAPLENGTH register (CAPLENGTH - address 0x4000 7100) bit description

Bit	Symbol	Description	Reset value	Access
7:0	CAPLENGTH	Indicates offset to add to the register base address at the beginning of the Operational Register	0x40	RO
23:8	HCIVERSION	BCD encoding of the EHCI revision number supported by this host controller.	0x100	RO
31:24	-	These bits are reserved and should be set to zero.	-	-

Table 446. HCSPARAMS register (HCSPARAMS - address 0x4000 7104) bit description

Bit	Symbol	Description	Reset value	Access
3:0	N_PORTS	Number of downstream ports. This field specifies the number of physical downstream ports implemented on this host controller.	0x1	RO
4	PPC	Port Power Control. This field indicates whether the host controller implementation includes port power control.	0x1	RO
7:5	-	These bits are reserved and should be set to zero.	-	-
11:8	N_PCC	Number of Ports per Companion Controller. This field indicates the number of ports supported per internal Companion Controller.	0x0	RO
15:12	N_CC	Number of Companion Controller. This field indicates the number of companion controllers associated with this USB2.0 host controller.	0x0	RO
16	PI	Port indicators. This bit indicates whether the ports support port indicator control.	0x1	RO
19:17	-	These bits are reserved and should be set to zero.	-	-
23:20	N_PTT	Number of Ports per Transaction Translator. This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller.	0x0	RO
27:24	N_TT	Number of Transaction Translators. This field indicates the number of embedded transaction translators associated with the USB2.0 host controller.	0x0	RO
31:28	-	These bits are reserved and should be set to zero.	-	-

Table 447. HCCPARAMS register (HCCPARAMS - address 0x4000 7108) bit description

Bit	Symbol	Description	Reset value	Access
0	ADC	64-bit Addressing Capability. If zero, no 64-bit addressing capability is supported.	0	RO
1	PFL	Programmable Frame List Flag. If set to one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-boundary. This requirement ensures that the frame list is always physically contiguous.	1	RO
2	ASP	Asynchronous Schedule Park Capability. If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule.The feature can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the USBCMD register.	1	RO
7:4	IST	Isochronous Scheduling Threshold. This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule.	0	RO
15:8	EECP	EHCI Extended Capabilities Pointer. This optional field indicates the existence of a capabilities list.	0	RO
31:16	-	These bits are reserved and should be set to zero.	-	-

Table 448. DCIVERSION register (DCIVERSION - address 0x4000 7120) bit description

Bit	Symbol	Description	Reset value	Access
15:0	DCIVERSION	The device controller interface conforms to the two-byte BCD encoding of the interface version number contained in this register.	0x1	RO
31:16	-	These bits are reserved and should be set to zero.	-	-

Table 449. DCCPARAMS (address 0x4000 7124)

Bit	Symbol	Description	Reset value	Access
4:0	DEN	Device Endpoint Number.	0x4	RO
6:5	-	These bits are reserved and should be set to zero.	-	-
7	DC	Device Capable.	0x1	RO
8	HC	Host Capable.	0x1	RO
31:9	-	These bits are reserved and should be set to zero.	-	-

24.6.2 USB Command register (USBCMD)

The host/device controller executes the command indicated in this register.

24.6.2.1 Device mode

Table 450. USB Command register in device mode (USBCMD_D - address 0x4000 7140) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RS		Run/Stop	0	R/W
		0	Writing a 0 to this bit will cause a detach event.		
		1	Writing a one to this bit will cause the device controller to enable a pull-up on USB_DP and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the device controller has been properly initialized.		
1	RST		Controller reset. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register.	0	R/W
		0	Set to 0 by hardware when the reset process is complete.		
		1	When software writes a one to this bit, the Device Controller resets its internal pipelines, timers, counters, state machines etc. to their initial values. Writing a one to this bit when the device is in the attached state is not recommended, since the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.		
3:2	-		Not used in device mode.	0	-
4	-		Not used in device mode.	0	-
5	-		Not used in device mode.	0	-
6	-		Not used in device mode. Writing a one to this bit when the device mode is selected, will have undefined results.	-	-
7	-	-	Reserved. These bits should be set to 0.	-	-
9:8	-	-	Not used in Device mode.	-	-
10	-		Reserved.These bits should be set to 0.	0	-
11	-	-	Not used in Device mode.		-
12	-		Reserved.These bits should be set to 0.	0	-
13	SUTW		Setup trip wire During handling a setup packet, this bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (see USBMODE register) then there exists a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software and will be cleared by hardware when a hazard exists. (See Section 23.10).	0	R/W
14	ATDTW		Add dTD trip wire This bit is used as a semaphore to ensure the to proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software during the process of adding a new dTD. See also Section 23.10. This bit shall also be cleared by hardware when its state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.	0	R/W

Table 450. USB Command register in device mode (USBCMD_D - address 0x4000 7140) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
15	FS2		Not used in device mode.	-	-
23:16	ITC		Interrupt threshold control. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. All other values are reserved. 0x0 = Immediate (no threshold) 0x1 = 1 micro frame. 0x2 = 2 micro frames. 0x8 = 8 micro frames. 0x10 = 16 micro frames. 0x20 = 32 micro frames. 0x40 = 64 micro frames.	0x8	R/W
31:24	-		Reserved	0	

24.6.2.2 Host mode

Table 451. USB Command register in host mode (USBCMD_H - address 0x4000 7140) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RS		Run/Stop	0	R/W
		0	When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Host Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the host controller is in the Halted state (i.e. HCHalted in the USBSTS register is a one).		
		1	When set to a 1, the Host Controller proceeds with the execution of the schedule. The Host Controller continues execution as long as this bit is set to a one.		
1	RST		Controller reset. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register.	0	R/W
		0	This bit is set to zero by hardware when the reset process is complete.		
		1	When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior.		
2	FS0		Bit 0 of the Frame List Size bits. See Table 452. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index. Note that this field is made up from USBCMD bits 15, 3, and 2.	0	
3	FS1		Bit 1 of the Frame List Size bits. See Table 452	0	

Table 451. USB Command register in host mode (USBCMD_H - address 0x4000 7140) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
4	PSE		This bit controls whether the host controller skips processing the periodic schedule.	0	R/W
		0	Do not process the periodic schedule.		
		1	Use the PERIODICLISTBASE register to access the periodic schedule.		
5	ASE		This bit controls whether the host controller skips processing the asynchronous schedule.	0	R/W
		0	Do not process the asynchronous schedule.		
		1	Use the ASYNCLISTADDR to access the asynchronous schedule.		
6	IAA		This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule.	0	R/W
		0	The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one.		
		1	Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results.		
7	-	-	Reserved	0	
9:8	ASP1_0		Asynchronous schedule park mode. Contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 0x1 to 0x3. **Remark:** Software must not write 00 to this bit when Park Mode Enable is one as this will result in undefined behavior.	11	R/W
10	-	-	Reserved.	0	-
11	ASPE		Asynchronous Schedule Park Mode Enable	1	R/W
		0	Park mode is disabled.		
		1	Park mode is enabled.		
12	-	-	Reserved.	0	-
13	-	-	Not used in Host mode.		-
14	-	-	Reserved.	0	-

Table 451. USB Command register in host mode (USBCMD_H - address 0x4000 7140) bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
15	FS2		Bit 2 of the Frame List Size bits. See Table 452.	0	-
23:16	ITC		Interrupt threshold control. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. All other values are reserved. 0x0 = Immediate (no threshold) 0x1 = 1 micro frame. 0x2 = 2 micro frames. 0x8 = 8 micro frames. 0x10 = 16 micro frames. 0x20 = 32 micro frames. 0x40 = 64 micro frames.	0x8	R/W
31:24	-		Reserved	0	

Table 452. Frame list size values

USBCMD bit 15	USBCMD bit 3	USBCMD bit 2	Frame list size
0	0	0	1024 elements (4096 bytes) - default value
0	0	1	512 elements (2048 bytes)
0	1	0	256 elements (1024 bytes)
0	1	1	128 elements (512 bytes)
1	0	0	64 elements (256 bytes)
1	0	1	32 elements (128 bytes)
1	1	0	16 elements (64 bytes)
1	1	1	8 elements (32 bytes)

24.6.3 USB Status register (USBSTS)

This register indicates various states of the Host/Device controller and any pending interrupts. Software sets a bit to zero in this register by writing a one to it.

Remark: This register does not indicate status resulting from a transaction on the serial bus.

24.6.3.1 Device mode

Table 453. USB Status register in device mode (USBSTS_D - address 0x4000 7144) register bit description

Bit	Symbol	Value	Description	Reset value	Access
0	UI		USB interrupt	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
1	UEI		USB error interrupt	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set. The device controller detects resume signaling only (see Section 23.10.11.6).		
2	PCI		Port change detect.	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit (URI) and the DCSuspend bits (SLI) respectively.		
3	-		Not used in Device mode.		
4	-	0	Reserved.		
5	-		Not used in Device mode.	0	-
6	URI		USB reset received	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When the device controller detects a USB Reset and enters the default state, this bit will be set to a one.		

Table 453. USB Status register in device mode (USBSTS_D - address 0x4000 7144) register bit description

Bit	Symbol	Value	Description	Reset value	Access
7	SRI		SOF received	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1 ms in device FS mode and every 125 μs in HS mode and will be synchronized to the actual SOF that is received. Since the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp.		
8	SLI		DCSuspend	0	R/WC
		0	The device controller clears the bit upon exiting from a suspend state. This bit is cleared by software writing a one to it.		
		1	When a device controller enters a suspend state from an active state, this bit will be set to a one.		
11:9	-	-	Reserved. Software should only write 0 to reserved bits.	0	
12	-	-	Not used in Device mode.	0	
13	-	-	Not used in Device mode.	0	
14	-	-	Not used in Device mode.	0	
15	-	-	Not used in Device mode.	0	
16	NAKI		NAK interrupt bit	0	RO
		0	This bit is automatically cleared by hardware when the all the enabled TX/RX Endpoint NAK bits are cleared.		
		1	It is set by hardware when for a particular endpoint both the TX/RX Endpoint NAK bit and the corresponding TX/RX Endpoint NAK Enable bit are set.		
17	-	-	Reserved. Software should only write 0 to reserved bits.	0	-
18	-		Not used in Device mode.	0	-
19	-		Not used in Device mode.	0	-
31:20	-	-	Reserved. Software should only write 0 to reserved bits.		-

24.6.3.2 Host mode

Table 454. USB Status register in host mode (USBSTS_H - address 0x4000 7144) register bit description

Bit	Symbol	Value	Description	Reset value	Access
0	UI		USB interrupt (USBINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
1	UEI		USB error interrupt (USBERRINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set.		
2	PCI		Port change detect.	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port.		
3	FRI		Frame list roll-over	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX [13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FRINDEX bit 12 toggles (see Section 24.6.5).		
4	-	0	Reserved.		
5	AAI		Interrupt on async advance	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the USBCMD register. This status bit indicates the assertion of that interrupt source.		
6	-	-	Not used by the Host controller.	0	R/WC
7	SRI		SOF received	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	In host mode, this bit will be set every 125 μs and can be used by host controller driver as a time base.		
8	SLI	-	Not used by the Host controller.	-	-
11:9	-	-	Reserved.		

Table 454. USB Status register in host mode (USBSTS_H - address 0x4000 7144) register bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
12	HCH		HCHalted	1	RO
		0	The RS bit in USBCMD is set to zero. Set by the host controller.		
		1	The Host Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. because of an internal error).		
13	RCL		Reclamation	0	RO
		0	No empty asynchronous schedule detected.		
		1	An empty asynchronous schedule is detected. Set by the host controller.		
14	PS		Periodic schedule status This bit reports the current real status of the Periodic Schedule. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (if both are 1) or disabled (if both are 0).	0	RO
		0	The periodic schedule status is disabled.		
		1	The periodic schedule status is enabled.		
15	AS		Asynchronous schedule status This bit reports the current real status of the Asynchronous Schedule. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (if both are 1) or disabled (if both are 0).	0	
		0	Asynchronous schedule status is disabled.		
		1	Asynchronous schedule status is enabled.		
16	-		Not used on Host mode.	0	-
17	-		Reserved.		
18	UAI		USB host asynchronous interrupt (USBHSTASYNCINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set **and** the TD was from the asynchronous schedule. This bit is also set by the Host when a short packet is detected **and** the packet is on the asynchronous schedule. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
19	UPI		USB host periodic interrupt (USBHSTPERINT)	0	R/WC
		0	This bit is cleared by software writing a one to it.		
		1	This bit is set by the Host Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set **and** the TD was from the periodic schedule. This bit is also set by the Host Controller when a short packet is detected **and** the packet is on the periodic schedule. A short packet is when the actual number of bytes received was less than the expected number of bytes.		
31:20	-		Reserved.	-	-

24.6.4 USB Interrupt register (USBINTR)

The software interrupts are enabled with this register. An interrupt is generated when a bit is set and the corresponding interrupt is active. The USB Status register (USBSTS) still shows interrupt sources even if they are disabled by the USBINTR register, allowing polling of interrupt events by the software. All interrupts must be acknowledged by software by clearing (that is writing a 1 to) the corresponding bit in the USBSTS register.

24.6.4.1 Device mode

Table 455. USB Interrupt register in device mode (USBINTR_D - address 0x4000 7148) bit description

Bit	Symbol	Description	Reset value	Access
0	UE	USB interrupt enable When this bit is one, and the USBINT bit in the USBSTS register is one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit in USBSTS.	0	R/W
1	UEE	USB error interrupt enable When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit in the USBSTS register.	0	R/W
2	PCE	Port change detect enable When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host/device controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit in USBSTS.	0	R/W
3	-	Not used by the Device controller.		
4	-	Reserved	0	-
5	-	Not used by the Device controller.		
6	URE	USB reset enable When this bit is a one, and the USB Reset Received bit in the USBSTS register is a one, the device controller will issue an interrupt. The interrupt is acknowledged by software clearing the USB Reset Received bit.	0	R/W
7	SRE	SOF received enable When this bit is a one, and the SOF Received bit in the USBSTS register is a one, the device controller will issue an interrupt. The interrupt is acknowledged by software clearing the SOF Received bit.	0	R/W
8	SLE	Sleep enable When this bit is a one, and the DCSuspend bit in the USBSTS register transitions, the device controller will issue an interrupt. The interrupt is acknowledged by software writing a one to the DCSuspend bit.	0	R/W
15:9	-	Reserved	-	-
16	NAKE	NAK interrupt enable This bit is set by software if it wants to enable the hardware interrupt for the NAK Interrupt bit. If both this bit and the corresponding NAK Interrupt bit are set, a hardware interrupt is generated.	0	R/W
17	-	Reserved		
18	UAIE	Not used by the Device controller.		
19	UPIA	Not used by the Device controller.		
31:20	-	Reserved		

24.6.4.2 Host mode

Table 456. USB Interrupt register in host mode (USBINTR_H - address 0x4000 7148) bit description

Bit	Symbol	Description	Access	Reset value
0	UE	USB interrupt enable When this bit is one, and the USBINT bit in the USBSTS register is one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit in USBSTS.	R/W	0
1	UEE	USB error interrupt enable When this bit is a one, and the USBERRINT bit in the USBSTS register is a one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit in the USBSTS register.	R/W	0
2	PCE	Port change detect enable When this bit is a one, and the Port Change Detect bit in the USBSTS register is a one, the host/device controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit in USBSTS.	R/W	0
3	FRE	Frame list rollover enable When this bit is a one, and the Frame List Rollover bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit.		
4	-	Reserved	-	0
5	AAE	Interrupt on asynchronous advance enable When this bit is a one, and the Interrupt on Async Advance bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.	R/W	0
6	-	Not used by the Host controller.	-	0
7	SRE	If this bit is one and the SRI bit in the USBSTS register is one, the host controller will issue an interrupt. In host mode, the SRI bit will be set every 125 μs and can be used by the host controller as a time base. The interrupt is acknowledged by software clearing the SRI bit in the USBSTS register.	-	0
8	-	Not used by the Host controller.	-	0
15:9	-	Reserved		
16	-	Not used by the host controller.	R/W	0
17	-	Reserved		
18	UAIE	USB host asynchronous interrupt enable When this bit is a one, and the USBHSTASYNCINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHSTASYNCINT bit.	R/W	0
19	UPIA	USB host periodic interrupt enable When this bit is a one, and the USBHSTPERINT bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBHSTPERINT bit.	R/W	0
31:20	-	Reserved		

24.6.5 Frame index register (FRINDEX)

24.6.5.1 Device mode

In Device mode this register is read only, and the device controller updates the FRINDEX[13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX[13:3] will be checked against the SOF marker. If FRINDEX[13:3] is different from the SOF marker, FRINDEX[13:3] will be set to the SOF value and FRINDEX[2:0] will be set to zero (i.e. SOF for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX[2:0] will be incremented (i.e. SOF for 125 μs micro-frame) by hardware.

Table 457. USB frame index register in device mode (FRINDEX_D - address 0x4000 714C) bit description

Bit	Symbol	Description	Reset value	Access
2:0	FRINDEX2_0	Current micro frame number	-	RO
13:3	FRINDEX13_3	Current frame number of the last frame transmitted	-	RO
31:14	-	Reserved	-	

24.6.5.2 Host mode

This register is used by the host controller to index the periodic frame list. The register updates every 125 μs (once each micro-frame). Bits[N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the USBCMD register.

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit in the USBSTS register (host mode). A write to this register while the Run/Stop bit is set to a one produces undefined results. Writes to this register also affect the SOF value.

Table 458. USB frame index register in host mode (FRINDEX_H - address 0x4000 714C) bit description

Bit	Symbol	Description	Reset value	Access
2:0	FRINDEX2_0	Current micro frame number	-	R/W
12:3	FRINDEX12_3	Frame list current index for 1024 elements.	-	R/W
31:13	-	Reserved	-	

Table 459. Number of bits used for the frame list index

USBCMD bit 15	USBCMD bit 3	USBCMD bit 2	Frame list size	Size of FRINDEX12_3 bit field
0	0	0	1024 elements (4096 bytes). Default value.	12
0	0	1	512 elements (2048 bytes)	11
0	1	0	256 elements (1024 bytes)	10
0	1	1	128 elements (512 bytes)	9
1	0	0	64 elements (256 bytes)	8

Table 459. Number of bits used for the frame list index

USBCMD bit 15	USBCMD bit 3	USBCMD bit 2	Frame list size	Size of FRINDEX12_3 bit field
1	0	1	32 elements (128 bytes)	7
1	1	0	16 elements (64 bytes)	6
1	1	1	8 elements (32 bytes)	5

24.6.6 Device address (DEVICEADDR) and Periodic List Base (PERIODICLISTBASE) registers

24.6.6.1 Device mode

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET_ADDRESS descriptor.

The USBADRA bit is used to accelerate the SET_ADDRESS sequence by allowing the DCD to preset the USBADR register bits before the status phase of the SET_ADDRESS descriptor.

Table 460. USB Device Address register in device mode (DEVICEADDR - address 0x4000 7154) bit description

Bit	Symbol	Value	Description	Reset value	Access
23:0	-		reserved	0	-
24	USBADRA		Device address advance		
		0	Any write to USBADR are instantaneous.		
		1	When the user writes a one to this bit at the same time or before USBADR is written, the write to USBADR fields is staged and held in a hidden register. After an IN occurs on endpoint 0 and is acknowledged, USBADR will be loaded from the holding register. Hardware will automatically clear this bit on the following conditions: • IN is ACKed to endpoint 0. USBADR is updated from the staging register. • OUT/SETUP occurs on endpoint 0. USBADR is **not** updated. • Device reset occurs. USBADR is set to 0. **Remark:** After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism will ensure this specification is met when the DCD can not write the device address within 2 ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR will be programmed instantly at the correct time and meet the 2 ms USB requirement.		
31:25	USBADR		USB device address	0	R/W

24.6.6.2 Host mode

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. The host controller driver (HCD) loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this

physical memory pointer is assumed to be 4 kB aligned. The contents of this register are combined with the Frame Index Register (FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Table 461. USB Periodic List Base register in host mode (PERIODICLISTBASE - address 0x4000 7154) bit description

Bit	Symbol	Description	Reset value	Access
11:0	-	Reserved	N/A	-
31:12	PERBASE31_12	Base Address (Low) These bits correspond to the memory address signals[31:12].	N/A	R/W

24.6.7 Endpoint List Address register (ENDPOINTLISTADDR) and Asynchronous List Address (ASYNCLISTADDR) registers

24.6.7.1 Device mode

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits[10:0] of this register cannot be modified by the system software and will always return a zero when read.The memory structure referenced by this physical memory pointer is assumed 64 byte aligned.

Table 462. USB Endpoint List Address register in device mode (ENDPOINTLISTADDR - address 0x4000 7158) bit description

Bit	Symbol	Description	Reset value	Access
10:0	-	reserved	0	-
31:11	EPBASE31_11	Endpoint list pointer (low) These bits correspond to memory address signals 31:11, respectively. This field will reference a list of up to 4 Queue Heads (QH). (i.e. one queue head per endpoint and direction.)	N/A	R/W

24.6.7.2 Host mode

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

Table 463. USB Asynchronous List Address register in host mode (ASYNCLISTADDR- address 0x4000 7158) bit description

Bit	Symbol	Description	Reset value	Access
4:0	-	Reserved	0	-
31:5	ASYBASE31_5	Link pointer (Low) LPL These bits correspond to memory address signals 31:5, respectively. This field may only reference a Queue Head (OH).	-	R/W

24.6.8 TT Control register (TTCTRL)

24.6.8.1 Device mode

This register is not used in device mode.

24.6.8.2 Host mode

This register contains parameters needed for internal TT operations. This register is used by the host controller only. Writes must be in Dwords.

Table 464. USB TT Control register in host mode (TTCTRL - address 0x4000 715C) bit description

Bit	Symbol	Description	Reset value	Access
23:0	-	Reserved.	0	-
30:24	TTHA	Hub address when FS or LS device are connected directly.	N/A	R/W
31	-	Reserved.	0	

24.6.9 Burst Size register (BURSTSIZE)

This register is used to control and dynamically change the burst size used during data movement on the master interface of the USB DMA controller. Writes must be in Dwords.

The default for the length of a burst of 32-bit words for RX and TX DMA data transfers is 16 words each.

Table 465. USB burst size register in device/host mode (BURSTSIZE - address 0x4000 7160) bit description

Bit	Symbol	Description	Reset value	Access
7:0	RXPBURST	Programmable RX burst length This register represents the maximum length of a burst in 32-bit words while moving data from the USB bus to system memory.	0x10	R/W
15:8	TXPBURST	Programmable TX burst length This register represents the maximum length of a burst in 32-bit words while moving data from system memory to the USB bus.	0x10	R/W
31:16	-	reserved	-	-

24.6.10 Transfer buffer Fill Tuning register (TXFILLTUNING)

24.6.10.1 Device controller

This register is not used in device mode.

24.6.10.2 Host controller

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level

T_s = Total packet flight time (send-only) packet; $T_s = T_0 + T_1$

T_p = Total packet time (fetch and send) packet; $T_p = T_{ff} + T_0 + T_1$

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure T_p remains before the end of the (micro) frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is < T_s then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a mark will be made the scheduler health counter to note the occurrence of a “backoff” event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Backoffs can be minimized with use of the TSCHHEALTH (T_{ff}) described below.

Table 466. USB Transfer buffer Fill Tuning register in host mode (TXFILLTUNING - address 0x4000 7164) bit description

Bit	Symbol	Description	Reset value	Access
7:0	TXSCHOH	FIFO burst threshold This register controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on to the bus. The minimum value is 2 and this value should be a low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USBMODE register is set.	0x2	R/W
12:8	TXSCHEATLTH	Scheduler health counter This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame . This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter. The maximum value is 31.	0x0	R/W
15:13	-	Reserved	-	-
21:16	TXFIFOTHRES	Scheduler overhead This register adds an additional fixed offset to the schedule time estimator described above as T_{ff}. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267 μs when a device is connected in High-Speed Mode. The time unit represented in this register is 6.333 μs when a device is connected in Low/Full Speed Mode.	0x0	R/W
31:22	-	Reserved		

24.6.11 USB ULPI viewport register (ULPIVIEWPORT)

The register provides indirect access to the ULPI PHY register set. Although the core performs access to the ULPI PHY register set, there may be extraordinary circumstances where software may need direct access.

Remark: Writes to the ULPI through the viewport can substantially harm standard USB operations. Currently no usage model has been defined where software should need to execute writes directly to the ULPI – see exception regarding optional features below.

Remark: Executing read operations though the ULPI viewport should have no harmful side effects to standard USB operations.

There are two operations that can be performed with the ULPI Viewport, wake-up and read /write operations. The wakeup operation is used to put the ULPI interface into normal operation mode and reenable the clock if necessary. A wakeup operation is required before accessing the registers when the ULPI interface is operating in low power mode, serial mode, or carkit mode. The ULPI state can be determined by reading the sync. state bit (ULPISS). If this bit is a one, then ULPI interface is running in normal operation mode and can accept read/write operations. If the ULPISS indicates a 0 then read/write operations will not be able execute. Undefined behavior will result if ULPISS = 0 and a read or write operation is performed. To execute a wakeup operation, write all 32 bits of the ULPI Viewport where ULPIPORT is constructed appropriately and the ULPIWU bit is a 1 and ULPIRUN bit is a 0. Poll the ULPI Viewport until ULPIWU is zero for the operation to complete.

To execute a read or write operation, write all 32-bits of the ULPI Viewport where ULPIDATWR, ULPIADDR, ULPIPORT, ULPIRW are constructed appropriately and the ULPIRUN bit is a 1. Poll the ULPI Viewport until ULPIRUN is zero for the operation to complete. Once ULPIRUN is zero, the ULPIDATRD will be valid if the operation was a read.

The polling method above could also be replaced and interrupt driven using the ULPI interrupt defined in the USBSTS and USBINTR registers. When a wakeup or read/write operation complete, the ULPI interrupt will be set.

Table 467. USB ULPI viewport register (ULPIVIEWPORT - address 0x4000 7170) bit description

Bit	Symbol	Value	Description	Access	Reset value
7:0	ULPIDATWR		When a write operation is commanded, the data to be sent is written to this field.	R/W	0
15:8	ULPIDATRD		After a read operation completes, the result is placed in this field.	R	0
23:16	ULPIADDR		When a read or write operation is commanded, the address of the operation is written to this field.	R/W	0
26:24	ULPIPORT		For the wakeup or read/write operation to be executed, this value must be written as 0.	R/W	000
27	ULPISS		ULPI sync state. This bit represents the state of the ULPI interface.	R	0
		0	In another state (ie. carkit, serial, low power)		
		1	Normal Sync. State.		
28	-	-	Reserved	-	-

Table 467. USB ULPI viewport register (ULPIVIEWPORT - address 0x4000 7170) bit description *...continued*

Bit	Symbol	Value	Description	Access	Reset value
29	ULPIRW		ULPI Read/Write control. This bit selects between running a read or write operation.	R/W	0
		0	Read		
		1	Write		
30	ULPIRUN		ULPI Read/Write Run. Writing the 1 to this bit will begin the read/write operation. The bit will automatically transition to 0 after the read/write is complete. Once this bit is set, the driver can not set it back to 0. **Remark:** The driver must never execute a wake-up and a read/write operation at the same time.	R/W	-
31	ULPIWU		ULPI Wake-up. Writing the 1 to this bit will begin the wake-up operation. The bit will automatically transition to 0 after the wake-up is complete. Once this bit is set, the driver can not set it back to 0. **Remark:** The driver must never execute a wake-up and a read/write operation at the same time.	R/W	0

24.6.12 BINTERVAL register

This register defines the bInterval value which determines the length of the virtual frame (see Section 23.7.7).

Remark: The BINTERVAL register is not related to the bInterval endpoint descriptor field in the USB specification.

Table 468. USB BINTERVAL register (BINTERVAL - address 0x4000 7174) bit description in device/host mode

Bit	Symbol	Description	Reset value	Access
3:0	BINT	bInterval value	0x00	R/W
31:4	-	Reserved	-	-

24.6.13 USB Endpoint NAK register (ENDPTNAK)

24.6.13.1 Device mode

This register indicates when the device sends a NAK handshake on an endpoint. Each Tx and Rx endpoint has a bit in the EPTN and EPRN field respectively.

A bit in this register is cleared by writing a 1 to it.

Table 469. USB endpoint NAK register in device mode (ENDPTNAK - address 0x4000 7178) bit description

Bit	Symbol	Description	Reset value	Access
3:0	EPRN	Rx endpoint NAK Each RX endpoint has one bit in this field. The bit is set when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint. Bit 3 corresponds to endpoint 3. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/WC
15:6	-	Reserved	-	-
19:16	EPTN	Tx endpoint NAK Each TX endpoint has one bit in this field. The bit is set when the device sends a NAK handshake on a received IN token for the corresponding endpoint. Bit 3 corresponds to endpoint 3. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/WC
31:20	-	Reserved	-	-

24.6.13.2 Host mode

This register is not used in host mode.

24.6.14 USB Endpoint NAK Enable register (ENDPTNAKEN)

24.6.14.1 Device mode

Each bit in this register enables the corresponding bit in the ENDPTNAK register. Each Tx and Rx endpoint has a bit in the EPTNE and EPRNE field respectively.

Table 470. USB Endpoint NAK Enable register in device mode (ENDPTNAKEN - address 0x4000 717C) bit description

Bit	Symbol	Description	Reset value	Access
3:0	EPRNE	Rx endpoint NAK enable Each bit enables the corresponding RX NAK bit. If this bit is set and the corresponding RX endpoint NAK bit is set, the NAK interrupt bit is set. Bit 3 corresponds to endpoint 3. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/W
15:4	-	Reserved	-	-
19:16	EPTNE	Tx endpoint NAK Each bit enables the corresponding TX NAK bit. If this bit is set and the corresponding TX endpoint NAK bit is set, the NAK interrupt bit is set. Bit 3 corresponds to endpoint 3. ... Bit 1 corresponds to endpoint 1. Bit 0 corresponds to endpoint 0.	0x00	R/W
31:20	-	Reserved	-	-

24.6.14.2 Host mode

This register is not used in host mode.

24.6.15 Port Status and Control register (PORTSC1)

24.6.15.1 Device mode

The device controller implements one port register, and it does not support power control. Port control in device mode is used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling. This register allows software to put the PHY into low-power Suspend mode and disable the PHY clock.

Table 471. Port Status and Control register in device mode (PORTSC1_D - address 0x4000 7184) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	CCS		Current connect status	0	RO
		0	Device not attached A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.		
		1	Device attached. A one indicates that the device successfully attached and is operating in either high-speed mode or full-speed mode as indicated by the High Speed Port bit in this register.		
1	CSC	-	Not used in device mode	0	-
2	PE	1	Port enable. This bit is always 1. The device port is always enabled.	1	RO

Table 471. Port Status and Control register in device mode (PORTSC1_D - address 0x4000 7184) bit description

Bit	Symbol	Value	Description	Reset value	Access
3	PEC	0	Port enable/disable change This bit is always 0. The device port is always enabled.	0	RO
5:4	-	-	Reserved	0	RO
6	FPR		Force port resume After the device has been in Suspend State for 5 ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. When this bit transitions to a one because a J-to-K transition detected, the Port Change Detect bit in the USBSTS register is set to one as well.	0	R/W
		0	No resume (K-state) detected/driven on port.		
		1	Resume detected/driven on port.		
7	SUSP		Suspend In device mode, this is a read-only status bit .	0	RO
		0	Port not in suspend state		
		1	Port in suspend state		
8	PR		Port reset In device mode, this is a read-only status bit. A device reset from the USB bus is also indicated in the USBSTS register.	0	RO
		0	Port is not in the reset state.		
		1	Port is in the reset state.		
9	HSP		High-speed status **Remark:** This bit is redundant with bits 27:26 (PSPD) in this register. It is implemented for compatibility reasons.	0	RO
		0	Host/device connected to the port is not in High-speed mode.		
		1	Host/device connected to the port is in High-speed mode.		
11:10	LS	-	Not used in device mode.		
12	PP	-	Not used in device mode.		
13	-	-	Reserved	-	-
15:14	PIC1_0		Port indicator control Writing to this field effects the value of the USB1_IND1:0 pins.	00	R/W
		0x0	Port indicators are off.		
		0x1	amber		
		0x2	green		
		0x3	undefined		

Table 471. Port Status and Control register in device mode (PORTSC1_D - address 0x4000 7184) bit description

Bit	Symbol	Value	Description	Reset value	Access
19:16	PTC3_0		Port test control Any value other than 0000 indicates that the port is operating in test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_HS/FS/LS values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. Values 0x7 to 0xF are reserved.	0000	R/W
		0x0	TEST_MODE_DISABLE		
		0x1	J_STATE		
		0x2	K_STATE		
		0x3	SE0 (host)/NAK (device)		
		0x4	Packet		
		0x5	FORCE_ENABLE_HS		
		0x6	FORCE_ENABLE_FS		
20	-	-	Not used in device mode. This bit is always 0 in device mode.	0	-
21	-	-	Not used in device mode. This bit is always 0 in device mode.	0	-
22	-		Not used in device mode. This bit is always 0 in device mode.	0	-
23	PHCD		PHY low power suspend - clock disable (PLPSCD) In device mode, The PHY can be put into Low Power Suspend – Clock Disable when the device is not running (USBCMD Run/Stop = 0) or the host has signaled suspend (PORTSC SUSPEND = 1). Low power suspend will be cleared automatically when the host has signaled resume. Before forcing a resume from the device, the device controller driver must clear this bit.	0	R/W
		0	Writing a 0 enables the PHY clock. Reading a 0 indicates the status of the PHY clock (enabled).		
		1	Writing a 1 disables the PHY clock. Reading a 1 indicates the status of the PHY clock (disabled).		
24	PFSC		Port force full speed connect	0	R/W
		0	Port connects at any speed.		
		1	Writing this bit to a 1 will force the port to only connect at full speed. It disables the chirp sequence that allows the port to identify itself as High-speed. This is useful for testing FS configurations with a HS host, hub or device.		
25	-	-	Reserved		
27:26	PSPD		Port speed This register field indicates the speed at which the port is operating.	0	RO
		0x1	Full-speed		
		0x2	invalid in device mode		
		0x3	High-speed		
29:28	-	-	Reserved	-	-
31:30	PTS		Parallel transceiver select. All other values are reserved.	<tbd>	R/W
		0x2	ULPI		
		0x3	Serial/ 1.1 PHY (Full-speed only)		

24.6.15.2 Host mode

The host controller uses one port. The register is only reset when power is initially applied or in response to a controller reset. The initial conditions of the port are:

- No device connected
- Port disabled

If the port has power control, this state remains until software applies power to the port by setting port power to one in the PORTSC register.

Table 472. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 7184) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	CCS		Current connect status This value reflects the current state of the port and may not correspond directly to the event that caused the CSC bit to be set. This bit is 0 if PP (Port Power bit) is 0. Software clears this bit by writing a 1 to it.	0	R/WC
		0	No device is present.		
		1	Device is present on the port.		
1	CSC		Connect status change Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be 'setting' an already-set bit (i.e., the bit will remain set). Software clears this bit by writing a one to it. This bit is 0 if PP (Port Power bit) is 0	0	R/WC
		0	No change in current status.		
		1	Change in current status.		
2	PE		Port enable. Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled. downstream propagation of data is blocked except for reset. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Port disabled.		
		1	Port enabled.		

Table 472. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 7184) bit description

…continued

Bit	Symbol	Value	Description	Reset value	Access
3	PEC	0	Port disable/enable change For the root hub, this bit gets set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See *Chapter 11 of the USB Specification*). Software clears this by writing a one to it. This bit is 0 if PP (Port Power bit) is 0,	0	R/WC
		0	No change.		
		1	Port enabled/disabled status has changed.		
4	OCA		Over-current active This bit will automatically transition from 1 to 0 when the over-current condition is removed.	0	RO
		0	The port does not have an over-current condition.		
		1	The port has currently an over-current condition.		
5	OCC		Over-current change This bit gets set to one when there is a change to Over-current Active. Software clears this bit by writing a one to this bit position.	0	R/WC
6	FPR		Force port resume Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the Port Change Detect bit in the USBSTS register is also set to one. This bit will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver. Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed ‘K’) is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no affect because the port controller will time the resume operation clear the bit the port control state switches to HS or FS idle. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	No resume (K-state) detected/driven on port.		
		1	Resume detected/driven on port.		

Table 472. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 7184) bit description

…continued

Bit	Symbol	Value	Description	Reset value	Access
7	SUSP		Suspend Together with the PE (Port enabled bit), this bit describes the port states, see Table 473 “Port states as described by the PE and SUSP bits in the PORTSC1 register”. The host controller will unconditionally set this bit to zero when software sets the Force Port Resume bit to zero. The host controller ignores a write of zero to this bit. If host software sets this bit to a one when the port is not enabled (i.e. Port enabled bit is a zero) the results are undefined. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Port not in suspend state		
		1	Port in suspend state When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.		
8	PR		Port reset When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver. This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Port is not in the reset state.		
		1	Port is in the reset state.		
9	HSP		High-speed status	0	RO
		0	Host/device connected to the port is not in High-speed mode.		
		1	Host/device connected to the port is in High-speed mode.		
11:10	LS		Line status These bits reflect the current logical levels of the USB_DP and USB_DM signal lines. USB_DP corresponds to bit 11 and USB_DM to bit 10. In host mode, the use of linestate by the host controller driver is not necessary for this controller (unlike EHCI) because the controller hardware manages the connection of LS and FS.	0x3	RO
		0x0	SE0 (USB_DP and USB_DM LOW)		
		0x1	J-state (USB_DP HIGH and USB_DM LOW)		
		0x2	K-state (USB_DP LOW and USB_DM HIGH)		
		0x3	Undefined		

Table 472. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 7184) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
12	PP	-	Port power control Host controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (i.e. PP equals a 0), the port is non-functional and will not report attaches, detaches, etc. When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitioned by the host controller driver from a one to a zero (removing power from the port).	0	R/W
		0	Port power off.		
		1	Port power on.		
13	-	-	Reserved	0	-
15:14	PIC1_0		Port indicator control Writing to this field controls the value of the pins USB1_IND1 and USB1_IND0.	00	R/W
		0x0	Port indicators are off.		
		0x1	Amber		
		0x2	Green		
		0x3	Undefined		
19:16	PTC3_0		Port test control Any value other than 0000 indicates that the port is operating in test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. Values 0x8 to 0xF are reserved.	0000	R/W
		0x0	TEST_MODE_DISABLE		
		0x1	J_STATE		
		0x2	K_STATE		
		0x3	SE0 (host)/NAK (device)		
		0x4	Packet		
		0x5	FORCE_ENABLE_HS		
		0x6	FORCE_ENABLE_FS		
		0x7	FORCE_ENABLE_LS		
20	WKCN		Wake on connect enable (WKCNNT_E) This bit is 0 if PP (Port Power bit) is 0	0	R/W
		0	Disables the port to wake up on device connects.		
		1	Writing this bit to a one enables the port to be sensitive to device connects as wake-up events.		

Table 472. Port Status and Control register in host mode (PORTSC1_H - address 0x4000 7184) bit description

…continued

Bit	Symbol	Value	Description	Reset value	Access
21	WKDC		Wake on disconnect enable (WKDSCNNT_E) This bit is 0 if PP (Port Power bit) is 0.	0	R/W
		0	Disables the port to wake up on device disconnects.		
		1	Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events.		
22	WKOC		Wake on over-current enable (WKOC_E)	0	R/W
		0	Disables the port to wake up on over-current events.		
		1	Writing a one to this bit enabled the port to be sensitive to over-current conditions as wake-up events.		
23	PHCD		PHY low power suspend - clock disable (PLPSCD) In host mode, the PHY can be put into Low Power Suspend – Clock Disable when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software.	0	R/W
		0	Writing a 0 enables the PHY clock. Reading a 0 indicates the status of the PHY clock (enabled).		
		1	Writing a 1 disables the PHY clock. Reading a 1 indicates the status of the PHY clock (disabled).		
24	PFSC		Port force full speed connect	0	R/W
		0	Port connects at any speed.		
		1	Writing this bit to a 1 will force the port to only connect at Full Speed. It disables the chirp sequence that allows the port to identify itself as High Speed. This is useful for testing FS configurations with a HS host, hub or device.		
25	-	-	Reserved		
27:26	PSPD		Port speed This register field indicates the speed at which the port is operating. For HS mode operation in the host controller and HS/FS operation in the device controller the port routing steers data to the Protocol engine. For FS and LS mode operation in the host controller, the port routing steers data to the Protocol Engine w/ Embedded Transaction Translator.	0	RO
		0x0	Full-speed		
		0x1	Low-speed		
		0x2	High-speed		
29:28	-	-	Reserved	-	-
31:30	PTS		Parallel transceiver select. All other values are reserved.	<tbd>	R/W
		0x2	ULPI		
		0x3	Serial/ 1.1 PHY (Full-speed only)		

Table 473. Port states as described by the PE and SUSP bits in the PORTSC1 register

PE bit	SUSP bit	Port state
0	0 or 1	disabled
1	0	enabled
1	1	suspend

24.6.16 USB Mode register (USBMODE)

The USBMODE register sets the USB mode for the USB controller. The possible modes are Device, Host, and Idle mode.

24.6.16.1 Device mode

Table 474. USB Mode register in device mode (USBMODE_D - address 0x4000 71A8) bit description

Bit	Symbol	Value	Description	Reset value	Access
1:0	CM1_0		Controller mode The controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. This register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the RESET bit in the USBCMD register before reprogramming this register.	00	R/ WO
		0x0	Idle		
		0x1	Reserved		
		0x2	Device controller		
		0x3	Host controller		
2	ES		Endian select This bit can change the byte ordering of the transfer buffers to match the host microprocessor bus architecture. The bit fields in the microprocessor interface and the DMA data structures (including the setup buffer within the device QH) are unaffected by the value of this bit, because they are based upon 32-bit words.	0	R/W
		0	Little endian: first byte referenced in least significant byte of 32-bit word.		
		1	Big endian: first byte referenced in most significant byte of 32-bit word.		
3	SLOM		Setup Lockout mode In device mode, this bit controls behavior of the setup lock mechanism. See Section 23.10.8.	0	R/W
		0	Setup Lockouts on		
		1	Setup Lockouts Off (DCD requires the use of Setup Buffer Tripwire in USBCMD)		

Table 474. USB Mode register in device mode (USBMODE_D - address 0x4000 71A8) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
4	SDIS		Stream disable mode **Remark:** The use of this feature substantially limits the overall USB performance that can be achieved.	0	R/W
		0	Not disabled		
		1	Disabled. Setting this bit to one disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received will be responded to with a NYET handshake when stream disable is active.		
5	-		Not used in device mode.	0	-
31:6	-	-	Reserved		

24.6.16.2 Host mode

Table 475. USB Mode register in host mode (USBMODE_H - address 0x4000 71A8) bit description

Bit	Symbol	Value	Description	Reset value	Access
1:0	CM1_0		Controller mode The controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. This register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the RESET bit in the USBCMD register before reprogramming this register.	00	R/ WO
		0x0	Idle		
		0x1	Reserved		
		0x2	Device controller		
		0x3	Host controller		
2	ES		Endian select This bit can change the byte ordering of the transfer buffers. The bit fields in the microprocessor interface and the DMA data structures (including the setup buffer within the device QH) are unaffected by the value of this bit, because they are based upon 32-bit words.	0	R/W
		0	Little endian: first byte referenced in least significant byte of 32-bit word.		
		1	Big endian: first byte referenced in most significant byte of 32-bit word.		
3	-		Not used in host mode	0	-

Table 475. USB Mode register in host mode (USBMODE_H - address 0x4000 71A8) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
4	SDIS		Stream disable mode **Remark:** The use of this feature substantially limits the overall USB performance that can be achieved.	0	R/W
		0	Not disabled		
		1	Disabled. Setting to a 1 ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB. Note: Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TXFILLTUNING to characterize the adjustments needed for the scheduler when using this feature.		
5	VBPS		VBUS power select	0	R/WO
		0	vbus_pwr_select is set LOW.		
		1	vbus_pwr_select is set HIGH		
31:6	-	-	Reserved	-	-

24.6.17 USB Endpoint Setup Status register (ENDPSETUPSTAT)

Table 476. USB Endpoint Setup Status register (ENDPTSETUPSTAT - address 0x4000 71AC) bit description

Bit	Symbol	Description	Reset value	Access
3:0	ENDPTSETUPSTAT	Setup endpoint status for logical endpoints. For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lockout mechanism is engaged.	0	R/WC
31:4	-	Reserved	-	-

24.6.18 USB Endpoint Prime register (ENDPTPRIME)

For each endpoint, software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed.

Remark: These bits will be momentarily set by hardware during hardware endpoint re-priming operations when a dTD is retired and the dQH is updated.

Table 477. USB Endpoint Prime register (ENDPTPRIME - address 0x4000 71B0) bit description

Bit	Symbol	Description	Reset value	Access
3:0	PERB	Prime endpoint receive buffer for physical OUT endpoints. For each OUT endpoint, a corresponding bit is set to 1 by software to request a buffer be prepared for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed. PERB0 = endpoint 0 ... PERB3 = endpoint 3	0	R/WS
15:4	-	Reserved	-	-
19:16	PETB	Prime endpoint transmit buffer for physical IN endpoints. For each IN endpoint a corresponding bit is set to one by software to request a buffer be prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed. PETB0 = endpoint 0 ... PETB3 = endpoint 3	0	R/WS
31:20	-	Reserved	-	-

24.6.19 USB Endpoint Flush register (ENDPTFLUSH)

Writing a one to a bit(s) in this register will cause the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer will continue until completion. Hardware will clear this register after the endpoint flush operation is successful.

Table 478. USB Endpoint Flush register (ENDPTFLUSH - address 0x4000 71B4) bit description

Bit	Symbol	Description	Reset value	Access
3:0	FERB	Flush endpoint receive buffer for physical OUT endpoints. Writing a one to a bit(s) will clear any primed buffers. FERB0 = endpoint 0 ... FERB3 = endpoint 3	0	R/WS

Table 478. USB Endpoint Flush register (ENDPTFLUSH - address 0x4000 71B4) bit description

Bit	Symbol	Description	Reset value	Access
15:4	-	Reserved	-	-
19:16	FETB	Flush endpoint transmit buffer for physical IN endpoints. Writing a one to a bit(s) will clear any primed buffers. FETB0 = endpoint 0 ... FETB3 = endpoint 3	0	R/WS
31:20	-	Reserved	-	-

24.6.20 USB Endpoint Status register (ENDPTSTAT)

One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set by hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There will always be a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register.

Remark: These bits will be momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired and the dQH is updated.

Table 479. USB Endpoint Status register (ENDPTSTAT - address 0x4000 71B8) bit description

Bit	Symbol	Description	Reset value	Access
3:0	ERBR	Endpoint receive buffer ready for physical OUT endpoints. This bit is set to 1 by hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. ERBR0 = endpoint 0 ... ERBR3 = endpoint 3	0	RO
15:4	-	Reserved	-	-
19:16	ETBR	Endpoint transmit buffer ready for physical IN endpoints 3 to 0. This bit is set to 1 by hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. ETBR0 = endpoint 0 ... ETBR3 = endpoint 3	0	RO
31:20	-	Reserved	-	-

24.6.21 USB Endpoint Complete register (ENDPTCOMPLETE)

Each bit in this register indicates that a received/transmit event occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit will be set simultaneously with the USBINT.

Writing a one will clear the corresponding bit in this register.

Table 480. USB Endpoint Complete register (ENDPTCOMPLETE - address 0x4000 71BC) bit description

Bit	Symbol	Description	Reset value	Access
3:0	ERCE	Endpoint receive complete event for physical OUT endpoints. This bit is set to 1 by hardware when receive event (OUT/SETUP) occurred. ERCE0 = endpoint 0 ... ERCE3 = endpoint 3	0	R/WC
15:4	-	Reserved	-	-
19:16	ETCE	Endpoint transmit complete event for physical IN endpoints. This bit is set to 1 by hardware when a transmit event (IN/INTERRUPT) occurred. ETCE0 = endpoint 0 ... ETCE3 = endpoint 3	0	R/WC
31:20	-	Reserved	-	-

24.6.22 USB Endpoint 0 Control register (ENDPTCTRL0)

This register initializes endpoint 0 for control transfer. Endpoint 0 is always a control endpoint.

Table 481. USB Endpoint 0 Control register (ENDPTCTRL0 - address 0x4000 71C0) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RXS		Rx endpoint stall	0	R/W
		0	Endpoint ok.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDSETUPSTAT bit is cleared.[1]		
1	-	-	Reserved		
3:2	RXT	0x0	Endpoint type Endpoint 0 is always a control endpoint.	0	R/W
6:4	-	-	Reserved	-	-
7	RXE	1	Rx endpoint enable Endpoint enabled. Control endpoint 0 is always enabled. This bit is always 1.	1	RO
15:8	-	-	Reserved	-	-

Table 481. USB Endpoint 0 Control register (ENDPTCTRL0 - address 0x4000 71C0) bit description *...continued*

Bit	Symbol	Value	Description	Reset value	Access
16	TXS		Tx endpoint stall		R/W
		0	Endpoint ok.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request. After receiving a SETUP request, this bit will continue to be cleared by hardware until the associated ENDSETUPSTAT bit is cleared.[1]		
17	-	-	Reserved		
19:18	TXT	0x0	Endpoint type Endpoint 0 is always a control endpoint.	0	RO
22:20	-	-	Reserved		
23	TXE	1	Tx endpoint enable Endpoint enabled. Control endpoint 0 is always enabled. This bit is always 1.	1	RO
31:24	-	-	Reserved		

[1] There is a slight delay (50 clocks max) between the ENPTSETUPSTAT being cleared and hardware continuing to clear this bit. In most systems it is unlikely that the DCD software will observe this delay. However, should the DCD notice that the stall bit is not set after writing a one to it, software should continually write this stall bit until it is set or until a new setup has been received by checking the associated ENDPTSETUPSTAT bit.

24.6.23 Endpoint 1 to 3 control registers

Each endpoint that is not a control endpoint has its own register to set the endpoint type and enable or disable the endpoint.

Remark: The reset value for all endpoint types is the control endpoint. If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled, then the endpoint type of the unused direction must be changed from the control type to any other type (e.g. bulk). Leaving an unconfigured endpoint control will cause undefined behavior for the data PID tracking on the active endpoint.

Table 482. USB Endpoint 1 to 3 control registers (ENDPTCTRL - address 0x4000 71C4 (ENDPTCTRL1) to 0x4000 71CC (ENDPTCTRL3)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	RXS		Rx endpoint stall	0	R/W
		0	Endpoint ok. This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request.[1]		

Table 482. USB Endpoint 1 to 3 control registers (ENDPTCTRL - address 0x4000 71C4 (ENDPTCTRL1) to 0x4000 71CC (ENDPTCTRL3)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
1	-		Reserved	0	R/W
3:2	RXT		Endpoint type	00	R/W
		0x0	Control		
		0x1	Isochronous		
		0x2	Bulk		
		0x3	Interrupt		
4	-	-	Reserved		
5	RXI		Rx data toggle inhibit This bit is only used for test and should always be written as zero. Writing a one to this bit will cause this endpoint to ignore the data toggle sequence and always accept data packets regardless of their data PID.	0	R/W
		0	Disabled		
		1	Enabled		
6	RXR		Rx data toggle reset Write 1 to reset the PID sequence. Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PIDs between the host and device.	0	WS
7	RXE		Rx endpoint enable **Remark:** An endpoint should be enabled only after it has been configured.	0	R/W
		0	Endpoint disabled.		
		1	Endpoint enabled.		
15:8	-	-	Reserved		
16	TXS		Tx endpoint stall	0	R/W
		0	Endpoint ok. This bit will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint, and this bit will continue to be cleared by hardware until the associated ENDPTSETUPSTAT bit is cleared.		
		1	Endpoint stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software, or it will automatically be cleared upon receipt of a new SETUP request.[1]		
17	-	-	Reserved	0	-
19:18	TXT		Tx endpoint type	00	R/W
		0x0	Control		
		0x1	Isochronous		
		0x2	Bulk		
		0x3	Interrupt		
20	-	-	Reserved		

Table 482. USB Endpoint 1 to 3 control registers (ENDPTCTRL - address 0x4000 71C4 (ENDPTCTRL1) to 0x4000 71CC (ENDPTCTRL3)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
21	TXI		Tx data toggle inhibit This bit is only used for test and should always be written as zero. Writing a one to this bit will cause this endpoint to ignore the data toggle sequence and always accept data packets regardless of their data PID.	0	R/W
		0	Enabled		
		1	Disabled		
22	TXR		Tx data toggle reset Write 1 to reset the PID sequence. Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.	1	WS
23	TXE		Tx endpoint enable **Remark:** An endpoint should be enabled only after it has been configured	0	R/W
		0	Endpoint disabled.		
		1	Endpoint enabled.		
31:24	-	-	Reserved	0	

[1] For control endpoints only: There is a slight delay (50 clocks max) between the ENPTSETUPSTAT being cleared and hardware continuing to clear this bit. In most systems it is unlikely that the DCD software will observe this delay. However, should the DCD notice that the stall bit is not set after writing a one to it, software should continually write this stall bit until it is set or until a new setup has been received by checking the associated ENDPTSETUPSTAT bit.

24.7 Functional description

For details on the device data structures, see Section 23.9. For the device operational model, see Section 23.10.

25.1 How to read this chapter

THe USB ROM API is available on parts LPC4350/30/20.

25.2 Introduction

The boot ROM contains a USB driver to simplify the USB application development. The USB driver implements the Communication Device Class (CDC), the Human Interface Device (HID), and the Mass Storage Device (MSC) device class. Only one device function can be used by the application software.

25.3 USB driver functions

The USB device driver ROM API consists of the following modules:

- Communication Device Class (CDC) function driver
  - Communication Device Class function driver initialization parameter data structure (Table 510 “USBD_CDC_INIT_PARAM class structure”).
  - CDC class API functions structure. This module exposes functions which interact directly with USB device controller hardware (Table 509 “USBD_CDC_API class structure”).
- USB core layer
  - struct (Table 506 “_WB_T class structure”)
  - union (Table 483 “__WORD_BYTE class structure”)
  - struct (Table 484 “_BM_T class structure”)
  - struct (Table 497 “_REQUEST_TYPE class structure”)
  - struct (Table 504 “_USB_SETUP_PACKET class structure”)
  - struct (Table 500 “_USB_DEVICE_QUALIFIER_DESCRIPTOR class structure”)
  - struct <tbd>USB device descriptor
  - struct (Table 500 “_USB_DEVICE_QUALIFIER_DESCRIPTOR class structure”)
  - struct <tbd>USB configuration descriptor
  - struct (Table 502 “_USB_INTERFACE_DESCRIPTOR class structure”)
  - struct <tbd>USB endpoint descriptor
  - struct (Table 505 “_USB_STRING_DESCRIPTOR class structure”)
  - struct (Table 498 “_USB_COMMON_DESCRIPTOR class structure”)
  - struct (Table 503 “_USB_OTHER_SPEED_CONFIGURATION class structure”)
  - USB descriptors data structure (Table 499 “_USB_CORE_DESCS_T class structure”)
  - USB device stack initialization parameter data structure (Table 508 “USBD_API_INIT_PARAM class structure”).

- USB device stack core API functions structure (Table 511 “USBD_CORE_API class structure”).

- Device Firmware Upgrade (DFU) class function driver
  - DFU descriptors data structure (Table 513 “USBD_DFU_INIT_PARAM class structure”).
  - DFU class API functions structure. This module exposes functions which interact directly with the USB device controller hardware (Table 512 “USBD_DFU_API class structure”).
- HID class function driver
  - struct (Table 492 “_HID_DESCRIPTOR class structure”).
  - struct (Table 494 “_HID_REPORT_T class structure”).
  - USB descriptors data structure (Table 515 “USBD_HID_INIT_PARAM class structure”).
  - HID class API functions structure. This structure contains pointers to all the functions exposed by the HID function driver module (Table 516 “USBD_HW_API class structure”).
- USB device controller driver
  - Hardware API functions structure. This module exposes functions which interact directly with the USB device controller hardware (Table 516 “USBD_HW_API class structure”).
- Mass Storage Class (MSC) function driver
  - Mass Storage Class function driver initialization parameter data structure (Table 518).
  - MSC class API functions structure. This module exposes functions which interact directly with the USB device controller hardware (Table 517).

25.4 Calling the USB device driver

A fixed location in ROM contains a pointer to the ROM driver table i.e. 0x1FFF 1FF8. The ROM driver table contains a pointer to the USB driver table. Pointers to the various USB driver functions are stored in this table. USB driver functions can be called by using a C structure. Figure 59 illustrates the pointer mechanism used to access the on-chip USB driver.

```
typedef struct USBD_API

{

const USBD_HW_API_T* hw;

const USBD_CORE_API_T* core;

const USBD_MSC_API_T* msc;

const USBD_DFU_API_T* dfu;

const USBD_HID_API_T* hid;

const USBD_CDC_API_T* cdc;
```

```
const uint32_t* reserved6;
const uint32_t version;
} USBD_API_T;
```

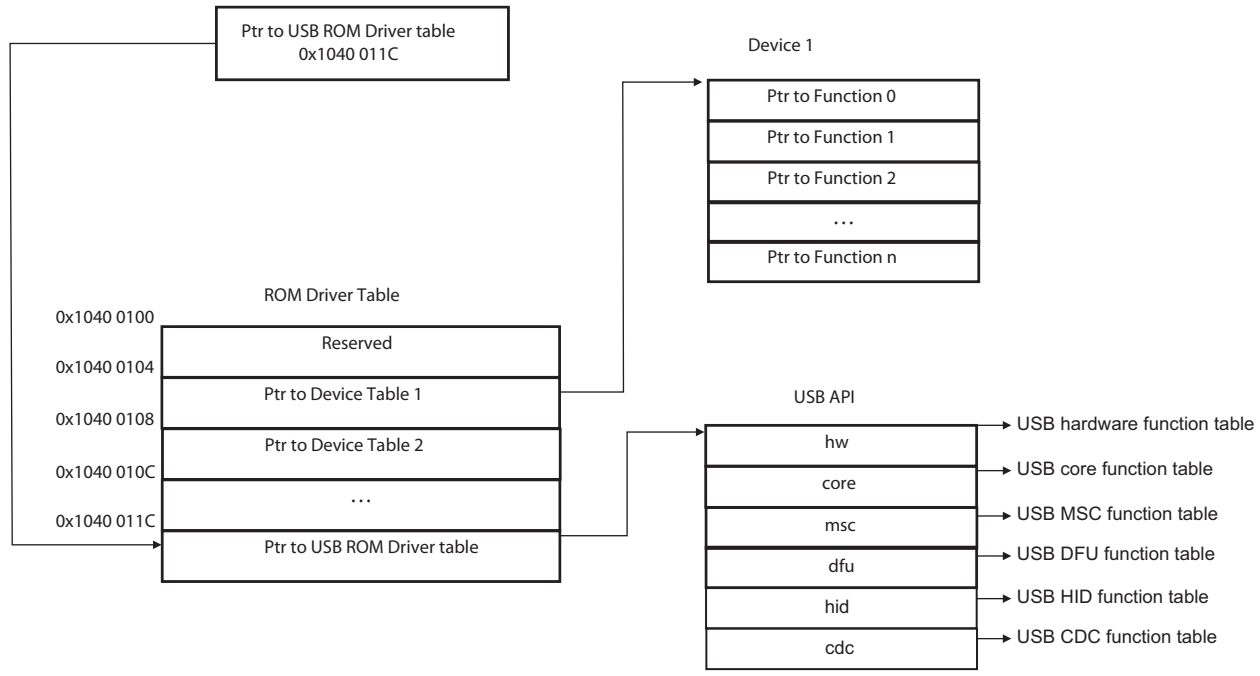


Fig 59. USB device driver pointer structure

25.5 USB API

25.5.1 __WORD_BYTE

Table 483. __WORD_BYTE class structure

Member	Description
W	uint16_tuint16_t __WORD_BYTE::W data member to do 16 bit access
WB	WB_TWB_T __WORD_BYTE::WB data member to do 8 bit access

25.5.2 _BM_T

Table 484. _BM_T class structure

Member	Description
Recipient	uint8_tuint8_t _BM_T::Recipient Recipient type.
Type	uint8_tuint8_t _BM_T::Type Request type.
Dir	uint8_tuint8_t _BM_T::Dir Direction type.

25.5.3 _CDC_ABSTRACT_CONTROL_MANAGEMENT_DESCRIPTOR

Table 485. _CDC_ABSTRACT_CONTROL_MANAGEMENT_DESCRIPTOR class structure

Member	Description
bFunctionLength	uint8_tuint8_t _CDC_ABSTRACT_CONTROL_MANAGEMENT_DESCRIPTOR::bFunctionLength
bDescriptorType	uint8_tuint8_t _CDC_ABSTRACT_CONTROL_MANAGEMENT_DESCRIPTOR::bDescriptorType
bDescriptorSubtype	uint8_tuint8_t _CDC_ABSTRACT_CONTROL_MANAGEMENT_DESCRIPTOR::bDescriptorSubtype
bmCapabilities	uint8_tuint8_t _CDC_ABSTRACT_CONTROL_MANAGEMENT_DESCRIPTOR::bmCapabilities

25.5.4 _CDC_CALL_MANAGEMENT_DESCRIPTOR

Table 486. _CDC_CALL_MANAGEMENT_DESCRIPTOR class structure

Member	Description
bFunctionLength	uint8_tuint8_t _CDC_CALL_MANAGEMENT_DESCRIPTOR::bFunctionLength
bDescriptorType	uint8_tuint8_t _CDC_CALL_MANAGEMENT_DESCRIPTOR::bDescriptorType
bDescriptorSubtype	uint8_tuint8_t _CDC_CALL_MANAGEMENT_DESCRIPTOR::bDescriptorSubtype
bmCapabilities	uint8_tuint8_t _CDC_CALL_MANAGEMENT_DESCRIPTOR::bmCapabilities
bDataInterface	uint8_tuint8_t _CDC_CALL_MANAGEMENT_DESCRIPTOR::bDataInterface

25.5.5 _CDC_HEADER_DESCRIPTOR

Table 487. _CDC_HEADER_DESCRIPTOR class structure

Member	Description
bFunctionLength	uint8_tuint8_t _CDC_HEADER_DESCRIPTOR::bFunctionLength
bDescriptorType	uint8_tuint8_t _CDC_HEADER_DESCRIPTOR::bDescriptorType
bDescriptorSubtype	uint8_tuint8_t _CDC_HEADER_DESCRIPTOR::bDescriptorSubtype
bcdCDC	uint16_tuint16_t _CDC_HEADER_DESCRIPTOR::bcdCDC

25.5.6 _CDC_LINE_CODING

Table 488. _CDC_LINE_CODING class structure

Member	Description
dwDTERate	uint32_tuint32_t _CDC_LINE_CODING::dwDTERate
bCharFormat	uint8_tuint8_t _CDC_LINE_CODING::bCharFormat
bParityType	uint8_tuint8_t _CDC_LINE_CODING::bParityType
bDataBits	uint8_tuint8_t _CDC_LINE_CODING::bDataBits

25.5.7 _CDC_UNION_1SLAVE_DESCRIPTOR

Table 489. _CDC_UNION_1SLAVE_DESCRIPTOR class structure

Member	Description
sUnion	CDC_UNION_DESCRIPTORCDC_UNION_DESCRIPTOR _CDC_UNION_1SLAVE_DESCRIPTOR::sUnion
bSlaveInterfaces	uint8_tuint8_t _CDC_UNION_1SLAVE_DESCRIPTOR::bSlaveInterfaces[1][1]

25.5.8 _CDC_UNION_DESCRIPTOR

Table 490. _CDC_UNION_DESCRIPTOR class structure

Member	Description
bFunctionLength	uint8_tuint8_t _CDC_UNION_DESCRIPTOR::bFunctionLength
bDescriptorType	uint8_tuint8_t _CDC_UNION_DESCRIPTOR::bDescriptorType
bDescriptorSubtype	uint8_tuint8_t _CDC_UNION_DESCRIPTOR::bDescriptorSubtype
bMasterInterface	uint8_tuint8_t _CDC_UNION_DESCRIPTOR::bMasterInterface

25.5.9 _DFU_STATUS

Table 491. _DFU_STATUS class structure

Member	Description
bStatus	uint8_tuint8_t _DFU_STATUS::bStatus
bwPollTimeout	uint8_tuint8_t _DFU_STATUS::bwPollTimeout[3][3]
bState	uint8_tuint8_t _DFU_STATUS::bState
iString	uint8_tuint8_t _DFU_STATUS::iString

25.5.10 _HID_DESCRIPTOR

HID class-specific HID Descriptor.

Table 492. _HID_DESCRIPTOR class structure

Member	Description
bLength	uint8_tuint8_t _HID_DESCRIPTOR::bLength Size of the descriptor, in bytes.
bDescriptorType	uint8_tuint8_t _HID_DESCRIPTOR::bDescriptorType Type of HID descriptor.
bcdHID	uint16_tuint16_t _HID_DESCRIPTOR::bcdHID BCD encoded version that the HID descriptor and device complies to.
bCountryCode	uint8_tuint8_t _HID_DESCRIPTOR::bCountryCode Country code of the localized device, or zero if universal.
bNumDescriptors	uint8_tuint8_t _HID_DESCRIPTOR::bNumDescriptors Total number of HID report descriptors for the interface.
DescriptorList	PRE_PACK struct POST_PACK _HID_DESCRIPTOR::_HID_DESCRIPTOR_LISTPRE_PACK struct POST_PACK _HID_DESCRIPTOR::_HID_DESCRIPTOR_LIST _HID_DESCRIPTOR::DescriptorList[1][1] Array of one or more descriptors

25.5.11 _HID_DESCRIPTOR::_HID_DESCRIPTOR_LIST

Table 493. _HID_DESCRIPTOR::_HID_DESCRIPTOR_LIST class structure

Member	Description
bDescriptorType	uint8_tuint8_t _HID_DESCRIPTOR::_HID_DESCRIPTOR_LIST::bDescriptorType Type of HID report.
wDescriptorLength	uint16_tuint16_t _HID_DESCRIPTOR::_HID_DESCRIPTOR_LIST::wDescriptorLength Length of the associated HID report descriptor, in bytes.

25.5.12 _HID_REPORT_T

HID report descriptor data structure.

Table 494. _HID_REPORT_T class structure

Member	Description
len	uint16_tuint16_t _HID_REPORT_T::len Size of the report descriptor in bytes.
idle_time	uint8_tuint8_t _HID_REPORT_T::idle_time This value is used by stack to respond to Set_Idle & GET_Idle requests for the specified report ID. The value of this field specified the rate at which duplicate reports are generated for the specified Report ID. For example, a device with two input reports could specify an idle rate of 20 milliseconds for report ID 1 and 500 milliseconds for report ID 2.
__pad	uint8_tuint8_t _HID_REPORT_T::__pad Padding space.
desc	uint8_t *uint8_t* _HID_REPORT_T::desc Report descriptor.

25.5.13 _MSC_CBW

Table 495. _MSC_CBW class structure

Member	Description
dSignature	uint32_tuint32_t _MSC_CBW::dSignature
dTag	uint32_tuint32_t _MSC_CBW::dTag
dDataLength	uint32_tuint32_t _MSC_CBW::dDataLength
bmFlags	uint8_tuint8_t _MSC_CBW::bmFlags
bLUN	uint8_tuint8_t _MSC_CBW::bLUN
bCBLength	uint8_tuint8_t _MSC_CBW::bCBLength
CB	uint8_tuint8_t _MSC_CBW::CB[16][16]

25.5.14 _MSC_CSW

Table 496. _MSC_CSW class structure

Member	Description
dSignature	uint32_tuint32_t _MSC_CSW::dSignature
dTag	uint32_tuint32_t _MSC_CSW::dTag
dDataResidue	uint32_tuint32_t _MSC_CSW::dDataResidue
bStatus	uint8_tuint8_t _MSC_CSW::bStatus

25.5.15 _REQUEST_TYPE

Table 497. _REQUEST_TYPE class structure

Member	Description
B	uint8_tuint8_t _REQUEST_TYPE::B byte wide access member
BM	BM_TBM_T _REQUEST_TYPE::BM bit field structure access member

25.5.16 _USB_COMMON_DESCRIPTOR

Table 498. _USB_COMMON_DESCRIPTOR class structure

Member	Description
bLength	uint8_tuint8_t _USB_COMMON_DESCRIPTOR::bLength Size of this descriptor in bytes
bDescriptorType	uint8_tuint8_t _USB_COMMON_DESCRIPTOR::bDescriptorType Descriptor Type

25.5.17 _USB_CORE_DESCS_T

USB descriptors data structure.

Table 499. _USB_CORE_DESCS_T class structure

Member	Description
device_desc	uint8_t *uint8_t* _USB_CORE_DESCS_T::device_desc Pointer to USB device descriptor
string_desc	uint8_t *uint8_t* _USB_CORE_DESCS_T::string_desc Pointer to array of USB string descriptors
full_speed_desc	uint8_t *uint8_t* _USB_CORE_DESCS_T::full_speed_desc Pointer to USB device configuration descriptor when device is operating in full speed mode.
high_speed_desc	uint8_t *uint8_t* _USB_CORE_DESCS_T::high_speed_desc Pointer to USB device configuration descriptor when device is operating in high speed mode. For full-speed only implementation this pointer should be same as full_speed_desc.
device_qualifier	uint8_t *uint8_t* _USB_CORE_DESCS_T::device_qualifier Pointer to USB device qualifier descriptor. For full-speed only implementation this pointer should be set to null (0).

25.5.18 _USB_DEVICE_QUALIFIER_DESCRIPTOR

Table 500. _USB_DEVICE_QUALIFIER_DESCRIPTOR class structure

Member	Description
bLength	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bLength Size of descriptor
bDescriptorType	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bDescriptorType Device Qualifier Type
bcdUSB	uint16_tuint16_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bcdUSB USB specification version number (e.g., 0x0200H for V2.00)
bDeviceClass	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bDeviceClass Class Code
bDeviceSubClass	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bDeviceSubClass SubClass Code
bDeviceProtocol	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bDeviceProtocol Protocol Code
bMaxPacketSize0	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bMaxPacketSize0 Maximum packet size for other speed
bNumConfigurations	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bNumConfigurations Number of Other-speed Configurations
bReserved	uint8_tuint8_t _USB_DEVICE_QUALIFIER_DESCRIPTOR::bReserved Reserved for future use, must be zero

25.5.19 _USB_DFU_FUNC_DESCRIPTOR

Table 501. _USB_DFU_FUNC_DESCRIPTOR class structure

Member	Description
bLength	uint8_tuint8_t _USB_DFU_FUNC_DESCRIPTOR::bLength
bDescriptorType	uint8_tuint8_t _USB_DFU_FUNC_DESCRIPTOR::bDescriptorType
bmAttributes	uint8_tuint8_t _USB_DFU_FUNC_DESCRIPTOR::bmAttributes
wDetachTimeOut	uint16_tuint16_t _USB_DFU_FUNC_DESCRIPTOR::wDetachTimeOut
wTransferSize	uint16_tuint16_t _USB_DFU_FUNC_DESCRIPTOR::wTransferSize
bcdDFUVersion	uint16_tuint16_t _USB_DFU_FUNC_DESCRIPTOR::bcdDFUVersion

25.5.20 _USB_INTERFACE_DESCRIPTOR

Table 502. _USB_INTERFACE_DESCRIPTOR class structure

Member	Description
bLength	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bLength Size of this descriptor in bytes
bDescriptorType	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bDescriptorType INTERFACE Descriptor Type
bInterfaceNumber	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bInterfaceNumber Number of this interface. Zero-based value identifying the index in the array of concurrent interfaces supported by this configuration.
bAlternateSetting	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bAlternateSetting Value used to select this alternate setting for the interface identified in the prior field
bNumEndpoints	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bNumEndpoints Number of endpoints used by this interface (excluding endpoint zero). If this value is zero, this interface only uses the Default Control Pipe.
bInterfaceClass	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bInterfaceClass Class code (assigned by the USB-IF).
bInterfaceSubClass	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bInterfaceSubClass Subclass code (assigned by the USB-IF).
bInterfaceProtocol	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::bInterfaceProtocol Protocol code (assigned by the USB).
iInterface	uint8_tuint8_t _USB_INTERFACE_DESCRIPTOR::iInterface Index of string descriptor describing this interface

25.5.21 _USB_OTHER_SPEED_CONFIGURATION

Table 503. _USB_OTHER_SPEED_CONFIGURATION class structure

Member	Description
bLength	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::bLength Size of descriptor
bDescriptorType	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::bDescriptorType Other_speed_Configuration Type
wTotalLength	uint16_tuint16_t _USB_OTHER_SPEED_CONFIGURATION::wTotalLength Total length of data returned
bNumInterfaces	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::bNumInterfaces Number of interfaces supported by this speed configuration
bConfigurationValue	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::bConfigurationValue Value to use to select configuration
IConfiguration	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::IConfiguration Index of string descriptor
bmAttributes	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::bmAttributes Same as Configuration descriptor
bMaxPower	uint8_tuint8_t _USB_OTHER_SPEED_CONFIGURATION::bMaxPower Same as Configuration descriptor

25.5.22 _USB_SETUP_PACKET

Table 504. _USB_SETUP_PACKET class structure

Member	Description
bmRequestType	REQUEST_TYPEREQUEST_TYPE _USB_SETUP_PACKET::bmRequestType This bit-mapped field identifies the characteristics of the specific request. _BM_T.
bRequest	uint8_tuint8_t _USB_SETUP_PACKET::bRequest This field specifies the particular request. The Type bits in the bmRequestType field modify the meaning of this field. USBD_REQUEST.
wValue	WORD_BYTEWORD_BYTE _USB_SETUP_PACKET::wValue Used to pass a parameter to the device, specific to the request.
wIndex	WORD_BYTEWORD_BYTE _USB_SETUP_PACKET::wIndex Used to pass a parameter to the device, specific to the request. The wIndex field is often used in requests to specify an endpoint or an interface.
wLength	uint16_tuint16_t _USB_SETUP_PACKET::wLength This field specifies the length of the data transferred during the second phase of the control transfer.

25.5.23 _USB_STRING_DESCRIPTOR

Table 505. _USB_STRING_DESCRIPTOR class structure

Member	Description
bLength	uint8_tuint8_t _USB_STRING_DESCRIPTOR::bLength Size of this descriptor in bytes
bDescriptorType	uint8_tuint8_t _USB_STRING_DESCRIPTOR::bDescriptorType STRING Descriptor Type
bString	uint16_tuint16_t _USB_STRING_DESCRIPTOR::bString UNICODE encoded string

25.5.24 _WB_T

Table 506. _WB_T class structure

Member	Description
L	uint8_tuint8_t _WB_T::L lower byte
H	uint8_tuint8_t _WB_T::H upper byte

25.5.25 USBD_API

Main USBD API functions structure.This structure contains pointer to various USB Device stack's sub-module function tables. This structure is used as main entry point to access various methods (grouped in sub-modules) exposed by ROM based USB device stack.

Table 507. USBD_API class structure

Member	Description
hw	const USBD_HW_API_T *const USBD_HW_API_T* USBD_API::hw Pointer to function table which exposes functions which interact directly with USB device stack's core layer.
core	const USBD_CORE_API_T *const USBD_CORE_API_T* USBD_API::core Pointer to function table which exposes functions which interact directly with USB device controller hardware.
msc	const USBD_MSC_API_T *const USBD_MSC_API_T* USBD_API::msc Pointer to function table which exposes functions provided by MSC function driver module.
dfu	const USBD_DFU_API_T *const USBD_DFU_API_T* USBD_API::dfu Pointer to function table which exposes functions provided by DFU function driver module.
hid	const USBD_HID_API_T *const USBD_HID_API_T* USBD_API::hid Pointer to function table which exposes functions provided by HID function driver module.

Table 507. USBD_API class structure

Member	Description
cdc	const USBD_CDC_API_T *const USBD_CDC_API_T* USBD_API::cdc Pointer to function table which exposes functions provided by CDC-ACM function driver module.
reserved6	const uint32_t *const uint32_t* USBD_API::reserved6 Reserved for future function driver module.
version	const uint32_tconst uint32_t USBD_API::version Version identifier of USB ROM stack. The version is defined as 0x0CHDMhCC where each nibble represents version number of the corresponding component. CC - 7:0 - 8bit core version number h - 11:8 - 4bit hardware interface version number M - 15:12 - 4bit MSC class module version number D - 19:16 - 4bit DFU class module version number H - 23:20 - 4bit HID class module version number C - 27:24 - 4bit CDC class module version number H - 31:28 - 4bit reserved

25.5.26 USBD_API_INIT_PARAM

USB device stack initilization parameter data structure.

Table 508. USBD_API_INIT_PARAM class structure

Member	Description
usb_reg_base	uint32_tuint32_t USBD_API_INIT_PARAM::usb_reg_base USB device controller's base register address.
mem_base	uint32_tuint32_t USBD_API_INIT_PARAM::mem_base Base memory location from where the stack can allocate data and buffers. **Remark:** The memory address set in this field should be accessible by USB DMA controller. Also this value should be aligned on 2048 byte boundary.
mem_size	uint32_tuint32_t USBD_API_INIT_PARAM::mem_size The size of memory buffer which stack can use. **Remark:** The mem_size should be greater than the size returned by USBD_HW_API::GetMemSize() routine.
max_num_ep	uint8_tuint8_t USBD_API_INIT_PARAM::max_num_ep max number of endpoints supported by the USB device controller instance (specified by
pad0	uint8_tuint8_t USBD_API_INIT_PARAM::pad0[3][3]
USB_Reset_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_Reset_Event Event for USB interface reset. This event fires when the USB host requests that the device reset its interface. This event fires after the control endpoint has been automatically configured by the library. **Remark:** This event is called from USB_ISR context and hence is time-critical. Having delays in this callback will prevent the device from enumerating correctly or operate properly.
USB_Suspend_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_Suspend_Event Event for USB suspend. This event fires when the USB host suspends the device by halting its transmission of Start Of Frame pulses to the device. This is generally hooked in order to move the device over to a low power state until the host wakes up the device. **Remark:** This event is called from USB_ISR context and hence is time-critical. Having delays in this callback will cause other system issues.

Table 508. USBD_API_INIT_PARAM class structure

Member	Description
USB_Resume_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_Resume_Event Event for USB wake up or resume. This event fires when a the USB device interface is suspended and the host wakes up the device by supplying Start Of Frame pulses. This is generally hooked to pull the user application out of a low power state and back into normal operating mode. **Remark:** This event is called from USB_ISR context and hence is time-critical. Having delays in this callback will cause other system issues.
reserved_sbz	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::reserved_sbz Reserved parameter should be set to zero.
USB_SOF_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_SOF_Event Event for USB Start Of Frame detection, when enabled. This event fires at the start of each USB frame, once per millisecond in full-speed mode or once per 125 microseconds in high-speed mode, and is synchronized to the USB bus. This event is time-critical; it is run once per millisecond (full-speed mode) and thus long handlers will significantly degrade device performance. This event should only be enabled when needed to reduce device wake-ups. This event is not normally active - it must be manually enabled and disabled via the USB interrupt register. **Remark:** This event is not normally active - it must be manually enabled and disabled via the USB interrupt register.
USB_WakeUpCfg	USB_PARAM_CB_TUSB_PARAM_CB_T USBD_API_INIT_PARAM::USB_WakeUpCfg Event for remote walk-up configuration, when enabled. This event fires when the USB host request the device to configure itself for remote wake-up capability. The USB host sends this request to device which report remote wake-up capable in their device descriptors, before going to low-power state. The application layer should implement this callback if they have any special on board circuit to triger remote wake-up event. Also application can use this callback to differentiate the following SUSPEND event is caused by cable plug-out or host SUSPEND request. The device can wake-up host only after receiving this callback and remote wake-up feature is enabled by host. To signal remote wake-up the device has to generate resume signaling on bus by calling usapi.hw->WakeUp() routine. Parameters: 1. hUsb = Handle to the USB device stack. 2. param1 = When 0 - Clear the wake-up configuration, 1 - Enable the wake-up configuration. Returns: The call back should return ErrorCode_t type to indicate success or error condition.
USB_Power_Event	USB_PARAM_CB_TUSB_PARAM_CB_T USBD_API_INIT_PARAM::USB_Power_Event Reserved parameter should be set to zero.
USB_Error_Event	USB_PARAM_CB_TUSB_PARAM_CB_T USBD_API_INIT_PARAM::USB_Error_Event Event for error condition. This event fires when USB device controller detect an error condition in the system. Parameters: 1. hUsb = Handle to the USB device stack. 2. param1 = USB device interrupt status register. Returns: The call back should return ErrorCode_t type to indicate success or error condition.

Table 508. USBD_API_INIT_PARAM class structure

Member	Description
USB_Configure_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_Configure_Event Event for USB configuration number changed. This event fires when a the USB host changes the selected configuration number. On receiving configuration change request from host, the stack enables/configures the endpoints needed by the new configuration before calling this callback function. **Remark:** This event is called from USB_ISR context and hence is time-critical. Having delays in this callback will prevent the device from enumerating correctly or operate properly.
USB_Interface_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_Interface_Event Event for USB interface setting changed. This event fires when a the USB host changes the interface setting to one of alternate interface settings. On receiving interface change request from host, the stack enables/configures the endpoints needed by the new alternate interface setting before calling this callback function. **Remark:** This event is called from USB_ISR context and hence is time-critical. Having delays in this callback will prevent the device from enumerating correctly or operate properly.
USB_Feature_Event	USB_CB_TUSB_CB_T USBD_API_INIT_PARAM::USB_Feature_Event Event for USB feature changed. This event fires when a the USB host send set/clear feature request. The stack handles this request for USB_FEATURE_REMOTE_WAKEUP, USB_FEATURE_TEST_MODE and USB_FEATURE_ENDPOINT_STALL features only. On receiving feature request from host, the stack handle the request appropriately and then calls this callback function. **Remark:** This event is called from USB_ISR context and hence is time-critical. Having delays in this callback will prevent the device from enumerating correctly or operate properly.
virt_to_phys	uint32_t(*uint32_t(* USBD_API_INIT_PARAM::virt_to_phys)(void *vaddr))(void *vaddr) Reserved parameter for future use. should be set to zero.
cache_flush	void(*void(* USBD_API_INIT_PARAM::cache_flush)(uint32_t *start_adr, uint32_t *end_adr))(uint32_t *start_adr, uint32_t *end_adr) Reserved parameter for future use. should be set to zero.

25.5.27 USBD_CDC_API

CDC class API functions structure.This module exposes functions which interact directly with USB device controller hardware.

Table 509. USBD_CDC_API class structure

Member	Description
GetMemSize	uint32_t(*uint32_t USBD_CDC_API::GetMemSize)(USBD_CDC_INIT_PARAM_T *param) Function to determine the memory required by the CDC function driver module. This function is called by application layer before calling pUsbApi->CDC->Init(), to allocate memory used by CDC function driver module. The application should allocate the memory which is accessible by USB controller/DMA controller. **Remark:** Some memory areas are not accessible by all bus masters. Parameters: 1. param = Structure containing CDC function driver module initialization parameters. Returns: Returns the required memory size in bytes.

Table 509. USBD_CDC_API class structure

Member	Description
init	ErrorCode_t(*ErrorCode_t USBD_CDC_API::init)(USBD_HANDLE_T hUsb, USBD_CDC_INIT_PARAM_T *param, USBD_HANDLE_T *phCDC) Function to initialize CDC function driver module. This function is called by application layer to initialize CDC function driver module. hUsbHandle to the USB device stack. paramStructure containing CDC function driver module initialization parameters. Parameters: 1. hUsb = Handle to the USB device stack. 2. param = Structure containing CDC function driver module initialization parameters. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_BAD_MEM_BUF = Memory buffer passed is not 4-byte aligned or smaller than required. 3. ERR_API_INVALID_PARAM2 = Either CDC_Write() or CDC_Read() or CDC_Verify() callbacks are not defined. 4. ERR_USBD_BAD_INTF_DESC = Wrong interface descriptor is passed. 5. ERR_USBD_BAD_EP_DESC = Wrong endpoint descriptor is passed.
SendNotification	ErrorCode_t(*ErrorCode_t USBD_CDC_API::SendNotification)(USBD_HANDLE_T hCdc, uint8_t bNotification, uint16_t data) Function to initialize CDC function driver module. This function is called by application layer to initialize CDC function driver module. hUsbHandle to the USB device stack. paramStructure containing CDC function driver module initialization parameters. Parameters: 1. hUsb = Handle to the USB device stack. 2. param = Structure containing CDC function driver module initialization parameters. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_BAD_MEM_BUF = Memory buffer passed is not 4-byte aligned or smaller than required. 3. ERR_API_INVALID_PARAM2 = Either CDC_Write() or CDC_Read() or CDC_Verify() callbacks are not defined. 4. ERR_USBD_BAD_INTF_DESC = Wrong interface descriptor is passed. 5. ERR_USBD_BAD_EP_DESC = Wrong endpoint descriptor is passed.

25.5.28 USBD_CDC_INIT_PARAM

Communication Device Class function driver initialization parameter data structure.

Table 510. USBD_CDC_INIT_PARAM class structure

Member	Description
mem_base	uint32_tuint32_t USBD_CDC_INIT_PARAM::mem_base Base memory location from where the stack can allocate data and buffers. **Remark:** The memory address set in this field should be accessible by USB DMA controller. Also this value should be aligned on 4 byte boundary.
mem_size	uint32_tuint32_t USBD_CDC_INIT_PARAM::mem_size The size of memory buffer which stack can use. **Remark:** The mem_size should be greater than the size returned by USBD_CDC_API::GetMemSize() routine.
cif_intf_desc	uint8_t *uint8_t* USBD_CDC_INIT_PARAM::cif_intf_desc Pointer to the control interface descriptor within the descriptor array (
dif_intf_desc	uint8_t *uint8_t* USBD_CDC_INIT_PARAM::dif_intf_desc Pointer to the data interface descriptor within the descriptor array (
CIC_GetRequest	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::CIC_GetRequest)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t *length))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t *length) Communication Interface Class specific get request callback function. This function is provided by the application software. This function gets called when host sends CIC management element get requests. The setup packet data (hCdcHandle to CDC function driver. pSetupPointer to setup packet received from host. pBufferPointer to a pointer of data buffer containing request data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. lengthAmount of data to be sent back to host. Parameters: 1. hCdc = Handle to CDC function driver. 2. pSetup = Pointer to setup packet received from host. 3. pBuffer = Pointer to a pointer of data buffer containing request data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 4. length = Amount of data to be sent back to host. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 510. USBD_CDC_INIT_PARAM class structure

Member	Description
CIC_SetRequest	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::CIC_SetRequest)(USBD_HANDLE_T hCdc, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t length))(USBD_HANDLE_T hCdc, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t length) Communication Interface Class specific set request callback function. This function is provided by the application software. This function gets called when host sends a CIC management element requests. The setup packet data (hCdcHandle to CDC function driver. pSetupPointer to setup packet received from host. pBufferPointer to a pointer of data buffer containing request data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. lengthAmount of data copied to destination buffer. Parameters: 1. hCdc = Handle to CDC function driver. 2. pSetup = Pointer to setup packet received from host. 3. pBuffer = Pointer to a pointer of data buffer containing request data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 4. length = Amount of data copied to destination buffer. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.
CDC_BulkIN_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::CDC_BulkIN_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Communication Device Class specific BULK IN endpoint handler. The application software should provide the BULK IN endpoint handler. Applications should transfer data depending on the communication protocol type set in descriptors. **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 510. USBD_CDC_INIT_PARAM class structure

Member	Description
CDC_BulkOUT_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::CDC_BulkOUT_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Communication Device Class specific BULK OUT endpoint handler. The application software should provide the BULK OUT endpoint handler. Applications should transfer data depending on the communication protocol type set in descriptors. **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.
SendEncpsCmd	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::SendEncpsCmd)(USBD_HANDLE_T hCDC, uint8_t *buffer, uint16_t len))(USBD_HANDLE_T hCDC, uint8_t *buffer, uint16_t len)
GetEncpsResp	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::GetEncpsResp)(USBD_HANDLE_T hCDC, uint8_t **buffer, uint16_t *len))(USBD_HANDLE_T hCDC, uint8_t **buffer, uint16_t *len)
SetCommFeature	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::SetCommFeature)(USBD_HANDLE_T hCDC, uint16_t feature, uint8_t *buffer, uint16_t len))(USBD_HANDLE_T hCDC, uint16_t feature, uint8_t *buffer, uint16_t len)
GetCommFeature	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::GetCommFeature)(USBD_HANDLE_T hCDC, uint16_t feature, uint8_t **pBuffer, uint16_t *len))(USBD_HANDLE_T hCDC, uint16_t feature, uint8_t **pBuffer, uint16_t *len)
ClrCommFeature	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::ClrCommFeature)(USBD_HANDLE_T hCDC, uint16_t feature))(USBD_HANDLE_T hCDC, uint16_t feature)
SetCtrlLineState	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::SetCtrlLineState)(USBD_HANDLE_T hCDC, uint16_t state))(USBD_HANDLE_T hCDC, uint16_t state)
SendBreak	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::SendBreak)(USBD_HANDLE_T hCDC, uint16_t mstime))(USBD_HANDLE_T hCDC, uint16_t mstime)

Table 510. USBD_CDC_INIT_PARAM class structure

Member	Description
SetLineCode	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::SetLineCode)(USBD_HANDLE_T hCDC, CDC_LINE_CODING *line_coding))(USBD_HANDLE_T hCDC, CDC_LINE_CODING *line_coding)
CDC_InterruptEP_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::CDC_InterruptEP_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional Communication Device Class specific INTERRUPT IN endpoint handler. The application software should provide the INT IN endpoint handler. Applications should transfer data depending on the communication protocol type set in descriptors. **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.
CDC_Ep0_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_CDC_INIT_PARAM::CDC_Ep0_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional user over ridable function to replace the default CDC class handler. The application software could override the default EP0 class handler with their own by providing the handler function address as this data member of the parameter structure. Application which like the default handler should set this data member to zero before calling the USBD_CDC_API::Init(). **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

25.5.29 USBD_CORE_API

USBD stack Core API functions structure.

Table 511. USBD_CORE_API class structure

Member	Description
RegisterClassHandler	ErrorCode_t(*ErrorCode_t USBD_CORE_API::RegisterClassHandler)(USBD_HANDLE_T hUsb, USB_EP_HANDLER_T pfn, void *data) Function to register class specific EP0 event handler with USB device stack. The application layer uses this function when it has to register the custom class's EP0 handler. The stack calls all the registered class handlers on any EP0 event before going through default handling of the event. This gives the class handlers to implement class specific request handlers and also to override the default stack handling for a particular event targeted to the interface. Check USB_EP_HANDLER_T for more details on how the callback function should be implemented. Also application layer could use this function to register EP0 handler which responds to vendor specific requests. hUsbHandle to the USB device stack. pfnClass specific EP0 handler function. dataPointer to the data which will be passed when callback function is called by the stack. Parameters: 1. hUsb = Handle to the USB device stack. 2. pfn = Class specific EP0 handler function. 3. data = Pointer to the data which will be passed when callback function is called by the stack. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_TOO_MANY_CLASS_HDLR(0x0004000c) = The number of class handlers registered is greater than the number of handlers allowed by the stack.
RegisterEpHandler	ErrorCode_t(*ErrorCode_t USBD_CORE_API::RegisterEpHandler)(USBD_HANDLE_T hUsb, uint32_t ep_index, USB_EP_HANDLER_T pfn, void *data) Function to register interrupt/event handler for the requested endpoint with USB device stack. The application layer uses this function to register the custom class's EP0 handler. The stack calls all the registered class handlers on any EP0 event before going through default handling of the event. This gives the class handlers to implement class specific request handlers and also to override the default stack handling for a particular event targeted to the interface. Check USB_EP_HANDLER_T for more details on how the callback function should be implemented. hUsbHandle to the USB device stack. ep_indexClass specific EP0 handler function. pfnClass specific EP0 handler function. dataPointer to the data which will be passed when callback function is called by the stack. Parameters: 1. hUsb = Handle to the USB device stack. 2. ep_index = Class specific EP0 handler function. 3. pfn = Class specific EP0 handler function. 4. data = Pointer to the data which will be passed when callback function is called by the stack. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_TOO_MANY_CLASS_HDLR(0x0004000c) = Too many endpoint handlers.

Table 511. USBD_CORE_API class structure

Member	Description
SetupStage	void(*void USBD_CORE_API::SetupStage)(USBD_HANDLE_T hUsb) Function to set EP0 state machine in setup state. This function is called by USB stack and the application layer to set the EP0 state machine in setup state. This function will read the setup packet received from USB host into stack's buffer. **Remark:** This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly.Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
DataInStage	void(*void USBD_CORE_API::DataInStage)(USBD_HANDLE_T hUsb) Function to set EP0 state machine in data_in state. This function is called by USB stack and the application layer to set the EP0 state machine in data_in state. This function will write the data present in EP0Data buffer to EP0 FIFO for transmission to host. **Remark:** This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly.Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
DataOutStage	void(*void USBD_CORE_API::DataOutStage)(USBD_HANDLE_T hUsb) Function to set EP0 state machine in data_out state. This function is called by USB stack and the application layer to set the EP0 state machine in data_out state. This function will read the control data (EP0 out packets) received from USB host into EP0Data buffer. **Remark:** This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly.Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.

Table 511. USBD_CORE_API class structure

Member	Description
StatusInStage	void(*void USBD_CORE_API::StatusInStage)(USBD_HANDLE_T hUsb) Function to set EP0 state machine in status_in state. This function is called by USB stack and the application layer to set the EP0 state machine in status_in state. This function will send zero length IN packet on EP0 to host, indicating positive status. **Remark:** This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly.Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
StatusOutStage	void(*void USBD_CORE_API::StatusOutStage)(USBD_HANDLE_T hUsb) Function to set EP0 state machine in status_out state. This function is called by USB stack and the application layer to set the EP0 state machine in status_out state. This function will read the zero length OUT packet received from USB host on EP0. **Remark:** This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly.Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
StallEp0	void(*void USBD_CORE_API::StallEp0)(USBD_HANDLE_T hUsb) Function to set EP0 state machine in stall state. This function is called by USB stack and the application layer to generate STALL signalling on EP0 endpoint. This function will also reset the EP0Data buffer. **Remark:** This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly.Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.

25.5.30 USBD_DFU_API

DFU class API functions structure.This module exposes functions which interact directly with USB device controller hardware.

Table 512. USBD_DFU_API class structure

Member	Description
GetMemSize	uint32_t(*uint32_t USBD_DFU_API::GetMemSize)(USBD_DFU_INIT_PARAM_T *param) Function to determine the memory required by the DFU function driver module. This function is called by application layer before calling pUsbApi->dfu->Init(), to allocate memory used by DFU function driver module. The application should allocate the memory which is accessible by USB controller/DMA controller. **Remark:** Some memory areas are not accessible by all bus masters. Parameters: 1. param = Structure containing DFU function driver module initialization parameters. Returns: Returns the required memory size in bytes.
init	ErrorCode_t(*ErrorCode_t USBD_DFU_API::init)(USBD_HANDLE_T hUsb, USBD_DFU_INIT_PARAM_T *param, uint32_t init_state) Function to initialize DFU function driver module. This function is called by application layer to initialize DFU function driver module. hUsbHandle to the USB device stack. paramStructure containing DFU function driver module initialization parameters. Parameters: 1. hUsb = Handle to the USB device stack. 2. param = Structure containing DFU function driver module initialization parameters. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_BAD_MEM_BUF = Memory buffer passed is not 4-byte aligned or smaller than required. 3. ERR_API_INVALID_PARAM2 = Either DFU_Write() or DFU_Done() or DFU_Read() callbacks are not defined. 4. ERR_USBD_BAD_DESC = USB_DFU_DESCRIPTOR_TYPE is not defined immediately after interface descriptor.wTransferSize in descriptor doesn't match the value passed in param->wTransferSize.DFU_Detach() is not defined while USB_DFU_WILL_DETACH is set in DFU descriptor. 5. ERR_USBD_BAD_INTF_DESC = Wrong interface descriptor is passed.

25.5.31 USBD_DFU_INIT_PARAM

USB descriptors data structure.

Table 513. USBD_DFU_INIT_PARAM class structure

Member	Description
mem_base	uint32_tuint32_t USBD_DFU_INIT_PARAM::mem_base Base memory location from where the stack can allocate data and buffers. **Remark:** The memory address set in this field should be accessible by USB DMA controller. Also this value should be aligned on 4 byte boundary.
mem_size	uint32_tuint32_t USBD_DFU_INIT_PARAM::mem_size The size of memory buffer which stack can use. **Remark:** The mem_size should be greater than the size returned by USBD_DFU_API::GetMemSize() routine.
wTransferSize	uint16_tuint16_t USBD_DFU_INIT_PARAM::wTransferSize DFU transfer block size in number of bytes. This value should match the value set in DFU descriptor provided as part of the descriptor array (
pad	uint16_tuint16_t USBD_DFU_INIT_PARAM::pad
intf_desc	uint8_t *uint8_t* USBD_DFU_INIT_PARAM::intf_desc Pointer to the DFU interface descriptor within the descriptor array (
DFU_Write	uint8_t(*uint8_t(* USBD_DFU_INIT_PARAM::DFU_Write)(uint32_t block_num, uint8_t **src, uint32_t length, uint8_t *bwPollTimeout))(uint32_t block_num, uint8_t **src, uint32_t length, uint8_t *bwPollTimeout) DFU Write callback function. This function is provided by the application software. This function gets called when host sends a write command. For application using zero-copy buffer scheme this function is called for the first time with block_numDestination start address. srcPointer to a pointer to the source of data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. bwPollTimeoutPointer to a 3 byte buffer which the callback implementer should fill with the amount of minimum time, in milliseconds, that the host should wait before sending a subsequent DFU_GETSTATUS request. lengthNumber of bytes to be written. Parameters: 1. block_num = Destination start address. 2. src = Pointer to a pointer to the source of data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 3. bwPollTimeout = Pointer to a 3 byte buffer which the callback implementer should fill with the amount of minimum time, in milliseconds, that the host should wait before sending a subsequent DFU_GETSTATUS request. 4. length = Number of bytes to be written. Returns: Returns DFU_STATUS_ values defined in mw_usbd_dfu.h.

Table 513. USBD_DFU_INIT_PARAM class structure

Member	Description
DFU_Read	uint32_t(*uint32_t(* USBD_DFU_INIT_PARAM::DFU_Read)(uint32_t block_num, uint8_t **dst, uint32_t length))(uint32_t block_num, uint8_t **dst, uint32_t length) DFU Read callback function. This function is provided by the application software. This function gets called when host sends a read command. block_numDestination start address. dstPointer to a pointer to the source of data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. lengthAmount of data copied to destination buffer. Parameters: 1. block_num = Destination start address. 2. dst = Pointer to a pointer to the source of data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 3. length = Amount of data copied to destination buffer. Returns: Returns DFU_STATUS_ values defined in mw_usbd_dfu.h.

Table 513. USBD_DFU_INIT_PARAM class structure

Member	Description
DFU_Done	void(*void(* USBD_DFU_INIT_PARAM::DFU_Done)(void))(void) DFU done callback function. This function is provided by the application software. This function gets called after download is finished. Nothing. Returns: Nothing.
DFU_Detach	void(*void(* USBD_DFU_INIT_PARAM::DFU_Detach)(USBD_HANDLE_T hUsb))(USBD_HANDLE_T hUsb) DFU detach callback function. This function is provided by the application software. This function gets called after USB_REQ_DFU_DETACH is received. Applications which set USB_DFU_WILL_DETACH bit in DFU descriptor should define this function. As part of this function application can call Connect() routine to disconnect and then connect back with host. For application which rely on WinUSB based host application should use this feature since USB reset can be invoked only by kernel drivers on Windows host. By implementing this feature host doesn't have to issue reset instead the device has to do it automatically by disconnect and connect procedure. hUsbHandle DFU control structure. Parameters: 1. hUsb = Handle DFU control structure. Returns: Nothing.
DFU_Ep0_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_DFU_INIT_PARAM::DFU_Ep0_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional user over ridable function to replace the default DFU class handler. The application software could override the default EP0 class handler with their own by providing the handler function address as this data member of the parameter structure. Application which like the default handler should set this data member to zero before calling the USBD_DFU_API::Init(). **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

25.5.32 USBD_HID_API

HID class API functions structure.This structure contains pointers to all the function exposed by HID function driver module.

Table 514. USBD_HID_API class structure

Member	Description
GetMemSize	uint32_t(*uint32_t USBD_HID_API::GetMemSize)(USBD_HID_INIT_PARAM_T *param) Function to determine the memory required by the HID function driver module. This function is called by application layer before calling pUsbApi->hid->Init(), to allocate memory used by HID function driver module. The application should allocate the memory which is accessible by USB controller/DMA controller. **Remark:** Some memory areas are not accessible by all bus masters. Parameters: 1. param = Structure containing HID function driver module initialization parameters. Returns: Returns the required memory size in bytes.
init	ErrorCode_t(*ErrorCode_t USBD_HID_API::init)(USBD_HANDLE_T hUsb, USBD_HID_INIT_PARAM_T *param) Function to initialize HID function driver module. This function is called by application layer to initialize HID function driver module. On successful initialization the function returns a handle to HID function driver module in passed param structure. hUsbHandle to the USB device stack. paramStructure containing HID function driver module initialization parameters. Parameters: 1. hUsb = Handle to the USB device stack. 2. param = Structure containing HID function driver module initialization parameters. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_BAD_MEM_BUF = Memory buffer passed is not 4-byte aligned or smaller than required. 3. ERR_API_INVALID_PARAM2 = Either HID_GetReport() or HID_SetReport() callback are not defined. 4. ERR_USBD_BAD_DESC = HID_HID_DESCRIPTOR_TYPE is not defined immediately after interface descriptor. 5. ERR_USBD_BAD_INTF_DESC = Wrong interface descriptor is passed. 6. ERR_USBD_BAD_EP_DESC = Wrong endpoint descriptor is passed.

25.5.33 USBD_HID_INIT_PARAM

USB descriptors data structure.

Table 515. USBD_HID_INIT_PARAM class structure

Member	Description
mem_base	uint32_tuint32_t USBD_HID_INIT_PARAM::mem_base Base memory location from where the stack can allocate data and buffers. **Remark:** The memory address set in this field should be accessible by USB DMA controller. Also this value should be aligned on 4 byte boundary.
mem_size	uint32_tuint32_t USBD_HID_INIT_PARAM::mem_size The size of memory buffer which stack can use. **Remark:** The mem_size should be greater than the size returned by USBD_HID_API::GetMemSize() routine.
max_reports	uint8_tuint8_t USBD_HID_INIT_PARAM::max_reports Number of HID reports supported by this instance of HID class driver.
pad	uint8_tuint8_t USBD_HID_INIT_PARAM::pad[3][3]
intf_desc	uint8_t *uint8_t* USBD_HID_INIT_PARAM::intf_desc Pointer to the HID interface descriptor within the descriptor array (
report_data	USB_HID_REPORT_T *USB_HID_REPORT_T* USBD_HID_INIT_PARAM::report_data Pointer to an array of HID report descriptor data structure (**Remark:** This array should be of global scope.
HID_GetReport	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_GetReport)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t *length))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t *length) HID get report callback function. This function is provided by the application software. This function gets called when host sends a HID_REQUEST_GET_REPORT request. The setup packet data (**Remark:** HID reports are sent via interrupt IN endpoint also. This function is called only when report request is received on control endpoint. Application should implement HID_EpIn_Hdlr to send reports to host via interrupt IN endpoint. Parameters: 1. hHid = Handle to HID function driver. 2. pSetup = Pointer to setup packet received from host. 3. pBuffer = Pointer to a pointer of data buffer containing report data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 4. length = Amount of data copied to destination buffer. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 515. USBD_HID_INIT_PARAM class structure

Member	Description
HID_SetReport	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_SetReport)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t length))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuffer, uint16_t length) HID set report callback function. This function is provided by the application software. This function gets called when host sends a HID_REQUEST_SET_REPORT request. The setup packet data (hHidHandle to HID function driver. pSetupPointer to setup packet received from host. pBufferPointer to a pointer of data buffer containing report data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. lengthAmount of data copied to destination buffer. Parameters: 1. hHid = Handle to HID function driver. 2. pSetup = Pointer to setup packet received from host. 3. pBuffer = Pointer to a pointer of data buffer containing report data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 4. length = Amount of data copied to destination buffer. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 515. USBD_HID_INIT_PARAM class structure

Member	Description
HID_GetPhysDesc	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_GetPhysDesc)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuf, uint16_t *length))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuf, uint16_t *length) Optional callback function to handle HID_GetPhysDesc request. The application software could provide this callback HID_GetPhysDesc handler to handle get physical descriptor requests sent by the host. When host requests Physical Descriptor set 0, application should return a special descriptor identifying the number of descriptor sets and their sizes. A Get_Descriptor request with the Physical Index equal to 1 should return the first Physical Descriptor set. A device could possibly have alternate uses for its items. These can be enumerated by issuing subsequent Get_Descriptor requests while incrementing the Descriptor Index. A device should return the last descriptor set to requests with an index greater than the last number defined in the HID descriptor. **Remark:** Applications which don't have physical descriptor should set this data member to zero before calling the USBD_HID_API::Init(). Parameters: 1. hHid = Handle to HID function driver. 2. pSetup = Pointer to setup packet received from host. 3. pBuf = Pointer to a pointer of data buffer containing physical descriptor data. If the physical descriptor is in USB accessible memory area application could just update the pointer or else it should copy the descriptor to the address pointed by this pointer. 4. length = Amount of data copied to destination buffer or descriptor length. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.
HID_SetIdle	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_SetIdle)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t idleTime))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t idleTime) Optional callback function to handle HID_REQUEST_SET_IDLE request. The application software could provide this callback to handle HID_REQUEST_SET_IDLE requests sent by the host. This callback is provided to applications to adjust timers associated with various reports, which are sent to host over interrupt endpoint. The setup packet data (**Remark:** Applications which don't send reports on Interrupt endpoint or don't have idle time between reports should set this data member to zero before calling the USBD_HID_API::Init(). Parameters: 1. hHid = Handle to HID function driver. 2. pSetup = Pointer to setup packet received from host. 3. idleTime = Idle time to be set for the specified report. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 515. USBD_HID_INIT_PARAM class structure

Member	Description
HID_SetProtocol	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_SetProtocol)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t protocol))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t protocol) Optional callback function to handle HID_REQUEST_SET_PROTOCOL request. The application software could provide this callback to handle HID_REQUEST_SET_PROTOCOL requests sent by the host. This callback is provided to applications to adjust modes of their code between boot mode and report mode. **Remark:** Applications which don't support protocol modes should set this data member to zero before calling the USBD_HID_API::Init(). Parameters: 1. hHid = Handle to HID function driver. 2. pSetup = Pointer to setup packet received from host. 3. protocol = Protocol mode. 0 = Boot Protocol 1 = Report Protocol Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.
HID_EpIn_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_EpIn_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional Interrupt IN endpoint event handler. The application software could provide Interrupt IN endpoint event handler. Application which send reports to host on interrupt endpoint should provide an endpoint event handler through this data member. This data member is ignored if the interface descriptor hUsbHandle to the USB device stack. dataHandle to HID function driver. eventType of endpoint event. See USBD_EVENT_T for more details. Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Handle to HID function driver. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should return ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 515. USBD_HID_INIT_PARAM class structure

Member	Description
HID_EpOut_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_EpOut_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional Interrupt OUT endpoint event handler. The application software could provide Interrupt OUT endpoint event handler. Application which receives reports from host on interrupt endpoint should provide an endpoint event handler through this data member. This data member is ignored if the interface descriptor hUsbHandle to the USB device stack. dataHandle to HID function driver. eventType of endpoint event. See USBD_EVENT_T for more details. Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Handle to HID function driver. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should return ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.
HID_GetReportDesc	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_GetReportDesc)(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuf, uint16_t *length))(USBD_HANDLE_T hHid, USB_SETUP_PACKET *pSetup, uint8_t **pBuf, uint16_t *length) Optional user over-ridable function to replace the default HID_GetReportDesc handler. The application software could override the default HID_GetReportDesc handler with their own by providing the handler function address as this data member of the parameter structure. Application which like the default handler should set this data member to zero before calling the USBD_HID_API::Init() and also provide report data array **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

Table 515. USBD_HID_INIT_PARAM class structure

Member	Description
HID_Ep0_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_HID_INIT_PARAM::HID_Ep0_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional user over-ridable function to replace the default HID class handler. The application software could override the default EP0 class handler with their own by providing the handler function address as this data member of the parameter structure. Application which like the default handler should set this data member to zero before calling the USBD_HID_API::Init(). **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

25.5.34 USBD_HW_API

Hardware API functions structure.This module exposes functions which interact directly with USB device controller hardware.

Table 516. USBD_HW_API class structure

Member	Description
GetMemSize	uint32_t(*uint32_t USBD_HW_API::GetMemSize)(USBD_API_INIT_PARAM_T *param) Function to determine the memory required by the USB device stack's DCD and core layers. This function is called by application layer before calling pUsbApi->hw-> **Remark:** Some memory areas are not accessible by all bus masters. Parameters: 1. param = Structure containing USB device stack initialization parameters. Returns: Returns the required memory size in bytes.
Init	ErrorCode_t(*ErrorCode_t USBD_HW_API::Init)(USBD_HANDLE_T *phUsb, USB_CORE_DESCS_T *pDesc, USBD_API_INIT_PARAM_T *param) Function to initialize USB device stack's DCD and core layers. This function is called by application layer to initialize USB hardware and core layers. On successful initialization the function returns a handle to USB device stack which should be passed to the rest of the functions. phUsbPointer to the USB device stack handle of type USBD_HANDLE_T. paramStructure containing USB device stack initialization parameters. Parameters: 1. phUsb = Pointer to the USB device stack handle of type USBD_HANDLE_T. 2. param = Structure containing USB device stack initialization parameters. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK(0) = On success 2. ERR_USBD_BAD_MEM_BUF(0x0004000b) = When insufficient memory buffer is passed or memory is not aligned on 2048 boundary.
Connect	void(*void USBD_HW_API::Connect)(USBD_HANDLE_T hUsb, uint32_t con) Function to make USB device visible/invisible on the USB bus. This function is called after the USB initialization. This function uses the soft connect feature to make the device visible on the USB bus. This function is called only after the application is ready to handle the USB data. The enumeration process is started by the host after the device detection. The driver handles the enumeration process according to the USB descriptors passed in the USB initialization function. hUsbHandle to the USB device stack. conStates whether to connect (1) or to disconnect (0). Parameters: 1. hUsb = Handle to the USB device stack. 2. con = States whether to connect (1) or to disconnect (0). Returns: Nothing.

Table 516. USBD_HW_API class structure

Member	Description
ISR	void(*void USBD_HW_API::ISR)(USBD_HANDLE_T hUsb) Function to USB device controller interrupt events. When the user application is active the interrupt handlers are mapped in the user flash space. The user application must provide an interrupt handler for the USB interrupt and call this function in the interrupt handler routine. The driver interrupt handler takes appropriate action according to the data received on the USB bus. hUsbHandle to the USB device stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
Reset	void(*void USBD_HW_API::Reset)(USBD_HANDLE_T hUsb) Function to Reset USB device stack and hardware controller. Reset USB device stack and hardware controller. Disables all endpoints except EP0. Clears all pending interrupts and resets endpoint transfer queues. This function is called internally by pUsbApi->hw->init() and from reset event. hUsbHandle to the USB device stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
ForceFullSpeed	void(*void USBD_HW_API::ForceFullSpeed)(USBD_HANDLE_T hUsb, uint32_t cfg) Function to force high speed USB device to operate in full speed mode. This function is useful for testing the behavior of current device when connected to a full speed only hosts. hUsbHandle to the USB device stack. cfgWhen 1 - set force full-speed or 0 - clear force full-speed. Parameters: 1. hUsb = Handle to the USB device stack. 2. cfg = When 1 - set force full-speed or 0 - clear force full-speed. Returns: Nothing.

Table 516. USBD_HW_API class structure

Member	Description
WakeUpCfg	void(*void USBD_HW_API::WakeUpCfg)(USBD_HANDLE_T hUsb, uint32_t cfg) Function to configure USB device controller to wake-up host on remote events. This function is called by application layer to configure the USB device controller to wake-up on remote events. It is recommended to call this function from users's USB_WakeUpCfg() callback routine registered with stack. **Remark:** User's USB_WakeUpCfg() is registered with stack by setting the USB_WakeUpCfg member of USBD_API_INIT_PARAM_T structure before calling pUsbApi->hw->Init() routine. Certain USB device controllers needed to keep some clocks always on to generate resume signaling through pUsbApi->hw->WakeUp(). This hook is provided to support such controllers. In most controllers cases this is an empty routine. Parameters: 1. hUsb = Handle to the USB device stack. 2. cfg = When 1 - Configure controller to wake on remote events or 0 - Configure controller not to wake on remote events. Returns: Nothing.
SetAddress	void(*void USBD_HW_API::SetAddress)(USBD_HANDLE_T hUsb, uint32_t adr) Function to set USB address assigned by host in device controller hardware. This function is called automatically when USB_REQUEST_SET_ADDRESS request is received by the stack from USB host. This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly. Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. hUsbHandle to the USB device stack. adrUSB bus Address to which the device controller should respond. Usually assigned by the USB host. Parameters: 1. hUsb = Handle to the USB device stack. 2. adr = USB bus Address to which the device controller should respond. Usually assigned by the USB host. Returns: Nothing.
Configure	void(*void USBD_HW_API::Configure)(USBD_HANDLE_T hUsb, uint32_t cfg) Function to configure device controller hardware with selected configuration. This function is called automatically when USB_REQUEST_SET_CONFIGURATION request is received by the stack from USB host. This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly. Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. hUsbHandle to the USB device stack. cfgConfiguration index. Parameters: 1. hUsb = Handle to the USB device stack. 2. cfg = Configuration index. Returns: Nothing.

Table 516. USBD_HW_API class structure

Member	Description
ConfigEP	void(*void USBD_HW_API::ConfigEP)(USBD_HANDLE_T hUsb, USB_ENDPOINT_DESCRIPTOR *pEPD) Function to configure USB Endpoint according to descriptor. This function is called automatically when USB_REQUEST_SET_CONFIGURATION request is received by the stack from USB host. All the endpoints associated with the selected configuration are configured. This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly. Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. hUsbHandle to the USB device stack. pEPDEndpoint descriptor structure defined in USB 2.0 specification. Parameters: 1. hUsb = Handle to the USB device stack. 2. pEPD = Endpoint descriptor structure defined in USB 2.0 specification. Returns: Nothing.
DirCtrlEP	void(*void USBD_HW_API::DirCtrlEP)(USBD_HANDLE_T hUsb, uint32_t dir) Function to set direction for USB control endpoint EP0. This function is called automatically by the stack on need basis. This interface is provided to users to invoke this function in other scenarios which are not handle by current stack. In most user applications this function is not called directly. Also this function can be used by users who are selectively modifying the USB device stack's standard handlers through callback interface exposed by the stack. hUsbHandle to the USB device stack. cfgWhen 1 - Set EP0 in IN transfer mode 0 - Set EP0 in OUT transfer mode Parameters: 1. hUsb = Handle to the USB device stack. 2. cfg = When 1 - Set EP0 in IN transfer mode 0 - Set EP0 in OUT transfer mode Returns: Nothing.

Table 516. USBD_HW_API class structure

Member	Description
EnableEP	void(*void USBD_HW_API::EnableEP)(USBD_HANDLE_T hUsb, uint32_t EPNum) Function to enable selected USB endpoint. This function enables interrupts on selected endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Returns: Nothing. This function enables interrupts on selected endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number corresponding to the event as per USB specification. ie. An EP1_IN is represented by 0x81 number. For device events set this param to 0x0. eventType of endpoint event. See USBD_EVENT_T for more details. enable1 - enable event, 0 - disable event. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number corresponding to the event as per USB specification. ie. An EP1_IN is represented by 0x81 number. For device events set this param to 0x0. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. 4. enable = 1 - enable event, 0 - disable event. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK(0) = - On success 2. ERR_USBD_INVALID_REQ(0x00040001) = - Invalid event type.
DisableEP	void(*void USBD_HW_API::DisableEP)(USBD_HANDLE_T hUsb, uint32_t EPNum) Function to disable selected USB endpoint. This function disables interrupts on selected endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Returns: Nothing.

Table 516. USBD_HW_API class structure

Member	Description
ResetEP	void(*void USBD_HW_API::ResetEP)(USBD_HANDLE_T hUsb, uint32_t EPNum) Function to reset selected USB endpoint. This function flushes the endpoint buffers and resets data toggle logic. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Returns: Nothing.
SetStallEP	void(*void USBD_HW_API::SetStallEP)(USBD_HANDLE_T hUsb, uint32_t EPNum) Function to STALL selected USB endpoint. Generates STALL signalling for requested endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Returns: Nothing.
ClrStallEP	void(*void USBD_HW_API::ClrStallEP)(USBD_HANDLE_T hUsb, uint32_t EPNum) Function to clear STALL state for the requested endpoint. This function clears STALL state for the requested endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. Returns: Nothing.

Table 516. USBD_HW_API class structure

Member	Description
SetTestMode	ErrorCode_t(*ErrorCode_t USBD_HW_API::SetTestMode)(USBD_HANDLE_T hUsb, uint8_t mode) Function to set high speed USB device controller in requested test mode. USB-IF requires the high speed device to be put in various test modes for electrical testing. This USB device stack calls this function whenever it receives USB_REQUEST_CLEAR_FEATURE request for USB_FEATURE_TEST_MODE. Users can put the device in test mode by directly calling this function. Returns ERR_USBD_INVALID_REQ when device controller is full-speed only. hUsbHandle to the USB device stack. modeTest mode defined in USB 2.0 electrical testing specification. Parameters: 1. hUsb = Handle to the USB device stack. 2. mode = Test mode defined in USB 2.0 electrical testing specification. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK(0) = - On success 2. ERR_USBD_INVALID_REQ(0x00040001) = - Invalid test mode or Device controller is full-speed only.
ReadEP	uint32_t(*uint32_t USBD_HW_API::ReadEP)(USBD_HANDLE_T hUsb, uint32_t EPNum, uint8_t *pData) Function to read data received on the requested endpoint. This function is called by USB stack and the application layer to read the data received on the requested endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. pDataPointer to the data buffer where data is to be copied. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. 3. pData = Pointer to the data buffer where data is to be copied. Returns: Returns the number of bytes copied to the buffer.
ReadReqEP	uint32_t(*uint32_t USBD_HW_API::ReadReqEP)(USBD_HANDLE_T hUsb, uint32_t EPNum, uint8_t *pData, uint32_t len) Function to queue read request on the specified endpoint. This function is called by USB stack and the application layer to queue a read request on the specified endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. pDataPointer to the data buffer where data is to be copied. This buffer address should be accessible by USB DMA master. lenLength of the buffer passed. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. 3. pData = Pointer to the data buffer where data is to be copied. This buffer address should be accessible by USB DMA master. 4. len = Length of the buffer passed. Returns: Returns the length of the requested buffer.

Table 516. USBD_HW_API class structure

Member	Description
ReadSetupPkt	uint32_t(*uint32_t USBD_HW_API::ReadSetupPkt)(USBD_HANDLE_T hUsb, uint32_t EPNum, uint32_t *pData) Function to read setup packet data received on the requested endpoint. This function is called by USB stack and the application layer to read setup packet data received on the requested endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP0_IN is represented by 0x80 number. pDataPointer to the data buffer where data is to be copied. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP0_IN is represented by 0x80 number. 3. pData = Pointer to the data buffer where data is to be copied. Returns: Returns the number of bytes copied to the buffer.
WriteEP	uint32_t(*uint32_t USBD_HW_API::WriteEP)(USBD_HANDLE_T hUsb, uint32_t EPNum, uint8_t *pData, uint32_t cnt) Function to write data to be sent on the requested endpoint. This function is called by USB stack and the application layer to send data on the requested endpoint. hUsbHandle to the USB device stack. EPNumEndpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. pDataPointer to the data buffer from where data is to be copied. cntNumber of bytes to write. Parameters: 1. hUsb = Handle to the USB device stack. 2. EPNum = Endpoint number as per USB specification. ie. An EP1_IN is represented by 0x81 number. 3. pData = Pointer to the data buffer from where data is to be copied. 4. cnt = Number of bytes to write. Returns: Returns the number of bytes written.
WakeUp	void(*void USBD_HW_API::WakeUp)(USBD_HANDLE_T hUsb) Function to generate resume signaling on bus for remote host wake-up. This function is called by application layer to remotely wake-up host controller when system is in suspend state. Application should indicate this remote wake-up capability by setting USB_CONFIG_REMOTE_WAKEUP in bmAttributes of Configuration Descriptor. Also this routine will generate resume signalling only if host enables USB_FEATURE_REMOTE_WAKEUP by sending SET_FEATURE request before suspending the bus. hUsbHandle to the USB device stack. Parameters: 1. hUsb = Handle to the USB device stack. Returns: Nothing.
EnableEvent	ErrorCode_t(*ErrorCode_t(* USBD_HW_API::EnableEvent)(USBD_HANDLE_T hUsb, uint32_t EPNum, uint32_t event_type, uint32_t enable))(USBD_HANDLE_T hUsb, uint32_t EPNum, uint32_t event_type, uint32_t enable)

25.5.35 USBD_MSC_API

MSC class API functions structure.This module exposes functions which interact directly with USB device controller hardware.

Table 517. USBD_MSC_API class structure

Member	Description
GetMemSize	uint32_t(*uint32_t USBD_MSC_API::GetMemSize)(USBD_MSC_INIT_PARAM_T *param) Function to determine the memory required by the MSC function driver module. This function is called by application layer before calling pUsbApi->msc->Init(), to allocate memory used by MSC function driver module. The application should allocate the memory which is accessible by USB controller/DMA controller. **Remark:** Some memory areas are not accessible by all bus masters. Parameters: 1. param = Structure containing MSC function driver module initialization parameters. Returns: Returns the required memory size in bytes.
init	ErrorCode_t(*ErrorCode_t USBD_MSC_API::init)(USBD_HANDLE_T hUsb, USBD_MSC_INIT_PARAM_T *param) Function to initialize MSC function driver module. This function is called by application layer to initialize MSC function driver module. hUsbHandle to the USB device stack. paramStructure containing MSC function driver module initialization parameters. Parameters: 1. hUsb = Handle to the USB device stack. 2. param = Structure containing MSC function driver module initialization parameters. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success 2. ERR_USBD_BAD_MEM_BUF = Memory buffer passed is not 4-byte aligned or smaller than required. 3. ERR_API_INVALID_PARAM2 = Either MSC_Write() or MSC_Read() or MSC_Verify() callbacks are not defined. 4. ERR_USBD_BAD_INTF_DESC = Wrong interface descriptor is passed. 5. ERR_USBD_BAD_EP_DESC = Wrong endpoint descriptor is passed.

25.5.36 USBD_MSC_INIT_PARAM

Mass Storage class function driver initialization parameter data structure.

Table 518. USBD_MSC_INIT_PARAM class structure

Member	Description
mem_base	uint32_tuint32_t USBD_MSC_INIT_PARAM::mem_base Base memory location from where the stack can allocate data and buffers. **Remark:** The memory address set in this field should be accessible by USB DMA controller. Also this value should be aligned on 4 byte boundary.
mem_size	uint32_tuint32_t USBD_MSC_INIT_PARAM::mem_size The size of memory buffer which stack can use. **Remark:** The mem_size should be greater than the size returned by USBD_MSC_API::GetMemSize() routine.
InquiryStr	uint8_t *uint8_t* USBD_MSC_INIT_PARAM::InquiryStr Pointer to the 28 character string. This string is sent in response to the SCSI Inquiry command. **Remark:** The data pointed by the pointer should be of global scope.
BlockCount	uint32_tuint32_t USBD_MSC_INIT_PARAM::BlockCount Number of blocks present in the mass storage device
BlockSize	uint32_tuint32_t USBD_MSC_INIT_PARAM::BlockSize Block size in number of bytes
MemorySize	uint32_tuint32_t USBD_MSC_INIT_PARAM::MemorySize Memory size in number of bytes
intf_desc	uint8_t *uint8_t* USBD_MSC_INIT_PARAM::intf_desc Pointer to the interface descriptor within the descriptor array (
MSC_Write	void(*void(* USBD_MSC_INIT_PARAM::MSC_Write)(uint32_t offset, uint8_t **src, uint32_t length))(uint32_t offset, uint8_t **src, uint32_t length) MSC Write callback function. This function is provided by the application software. This function gets called when host sends a write command. offsetDestination start address. srcPointer to a pointer to the source of data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. lengthNumber of bytes to be written. Parameters: 1. offset = Destination start address. 2. src = Pointer to a pointer to the source of data. Pointer-to-pointer is used to implement zero-copy buffers. See Zero-Copy Data Transfer model for more details on zero-copy concept. 3. length = Number of bytes to be written. Returns: Nothing.

Table 518. USBD_MSC_INIT_PARAM class structure

Member	Description
MSC_Read	void(*void(* USBD_MSC_INIT_PARAM::MSC_Read)(uint32_t offset, uint8_t **dst, uint32_t length))(uint32_t offset, uint8_t **dst, uint32_t length) MSC Read callback function. This function is provided by the application software. This function gets called when host sends a read command. offsetSource start address. dstPointer to a pointer to the source of data. The MSC function drivers implemented in stack are written with zero-copy model. Meaning the stack doesn't make an extra copy of buffer before writing/reading data from USB hardware FIFO. Hence the parameter is pointer to a pointer containing address buffer (uint8_t** dst). So that the user application can update the buffer pointer instead of copying data to address pointed by the parameter. /note The updated buffer address should be accessible by USB DMA master. If user doesn't want to use zero-copy model, then the user should copy data to the address pointed by the passed buffer pointer parameter and shouldn't change the address value. See Zero-Copy Data Transfer model for more details on zero-copy concept. lengthNumber of bytes to be read. Parameters: 1. offset = Source start address. 2. dst = Pointer to a pointer to the source of data. The MSC function drivers implemented in stack are written with zero-copy model. Meaning the stack doesn't make an extra copy of buffer before writing/reading data from USB hardware FIFO. Hence the parameter is pointer to a pointer containing address buffer (uint8_t** dst). So that the user application can update the buffer pointer instead of copying data to address pointed by the parameter. /note The updated buffer address should be accessible by USB DMA master. If user doesn't want to use zero-copy model, then the user should copy data to the address pointed by the passed buffer pointer parameter and shouldn't change the address value. See Zero-Copy Data Transfer model for more details on zero-copy concept. 3. length = Number of bytes to be read. Returns: Nothing.

Table 518. USBD_MSC_INIT_PARAM class structure

Member	Description
MSC_Verify	ErrorCode_t(*ErrorCode_t(* USBD_MSC_INIT_PARAM::MSC_Verify)(uint32_t offset, uint8_t buf[], uint32_t length))(uint32_t offset, uint8_t buf[], uint32_t length) MSC Verify callback function. This function is provided by the application software. This function gets called when host sends a verify command. The callback function should compare the buffer with the destination memory at the requested offset and offsetDestination start address. bufBuffer containing the data sent by the host. lengthNumber of bytes to verify. Parameters: 1. offset = Destination start address. 2. buf = Buffer containing the data sent by the host. 3. length = Number of bytes to verify. Returns: Returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = If data in the buffer matches the data at destination 2. ERR_FAILED = At least one byte is different.
MSC_GetWriteBuf	void(*void(* USBD_MSC_INIT_PARAM::MSC_GetWriteBuf)(uint32_t offset, uint8_t **buff_adr, uint32_t length))(uint32_t offset, uint8_t **buff_adr, uint32_t length) Optional callback function to optimize MSC_Write buffer transfer. This function is provided by the application software. This function gets called when host sends SCSI_WRITE10/SCSI_WRITE12 command. The callback function should update the offsetDestination start address. bufBuffer containing the data sent by the host. lengthNumber of bytes to write. Parameters: 1. offset = Destination start address. 2. buf = Buffer containing the data sent by the host. 3. length = Number of bytes to write. Returns: Nothing.

Table 518. USBD_MSC_INIT_PARAM class structure

Member	Description
MSC_Ep0_Hdlr	ErrorCode_t(*ErrorCode_t(* USBD_MSC_INIT_PARAM::MSC_Ep0_Hdlr)(USBD_HANDLE_T hUsb, void *data, uint32_t event))(USBD_HANDLE_T hUsb, void *data, uint32_t event) Optional user overridable function to replace the default MSC class handler. The application software could override the default EP0 class handler with their own by providing the handler function address as this data member of the parameter structure. Application which like the default handler should set this data member to zero before calling the USBD_MSC_API::Init(). **Remark:** Parameters: 1. hUsb = Handle to the USB device stack. 2. data = Pointer to the data which will be passed when callback function is called by the stack. 3. event = Type of endpoint event. See USBD_EVENT_T for more details. Returns: The call back should returns ErrorCode_t type to indicate success or error condition. Return values: 1. LPC_OK = On success. 2. ERR_USBD_UNHANDLED = Event is not handled hence pass the event to next in line. 3. ERR_USBD_xxx = For other error conditions.

26.1 How to read this chapter

The Ethernet controller is available on parts LPC4350 and LPC4330.

26.2 Basic configuration

The Ethernet controller is configured as follows:

- See Table 519 for clocking and power control.
- The Ethernet is reset by the ETHERNET_RST (reset # 22).
- The Ethernet interrupt is connected to interrupt slot # 5 in the NVIC.
- The Ethernet wake-up packet indicator is connected to slot # 8 in the event router.
- Set the Ethernet mode to RMII or MII in the CREG6 register in the CREG block (see Table 43).

Table 519. Ethernet clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
Ethernet register interface clock	BASE_M4_CLK	CLK_M4_ETHERNET	up to 204 MHz	-
Ethernet PHY clock	BASE_PHY_RX_CLK	-	75 MHz	Select the clock pin ENET_RX_CLK as clock source for this base clock in the OUTCLK_7_CTRL register in the CGU.
Ethernet PHY clock	BASE_PHY_TX_CLK	-	75 MHz	Select the clock pin ENET_TX_CLK as clock source for this base clock in the OUTCLK_8_CTRL register in the CGU.

26.3 Features

- 10/100 Mbit/s
- TCP/IP hardware checksum
- IP checksum
- DMA support
- IEEE 1588 time stamping block
- IEEE 1588 advanced time stamp support (IEEE 1588-2008 v2)
- Power management remote wake-up frame and magic packet detection
- Supports both full-duplex and half-duplex operation
  - Supports CSMA/CD Protocol for half-duplex operation.

- Supports IEEE 802.3x flow control for full-duplex operation.
- Optional forwarding of received pause control frames to the user application in full-duplex operation.
- Back-pressure support for half-duplex operation.
- Automatic transmission of zero-quanta pause frame on deassertion of flow control input in full-duplex operation.

26.4 General description

The Ethernet block enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2005 standard. The Ethernet interface contains a full featured 10 Mbps or 100 Mbps Ethernet MAC (Media Access Controller) designed to provide optimized performance through the use of DMA hardware acceleration.

Features include a generous suite of control registers, half or full duplex operation, flow control, control frames, hardware acceleration for transmit retry, receive packet filtering and wake-up on LAN activity, supporting both Wake-Up and Magic Packet frames. Automatic frame transmission and reception with Scatter-Gather DMA off-loads many operations from the CPU.

Additional features such as IEEE 1588 Time Stamping (IEEE 1588-2002) and IEEE Advanced Time Stamp support (IEEE 1588-2008 v2), and Hardware Checksum for IP (IPv4 or IPv6), TCP, UDP, and ICMP enrich the list of supported features.

The Ethernet block is an AHB master connected to the AHB Multilayer Matrix and has access to internal SRAM and memory connected to the External Memory Controller for Ethernet data, control, and status information. Other AHB traffic in the LPC43xx can take place using other masters, effectively separating Ethernet activity from the rest of the system.

The Ethernet block interfaces with an off-chip Ethernet PHY using the MII (Media Independent Interface) or RMII (reduced MII) protocol and with the on-chip MIIM (Media Independent Interface Management) serial bus.

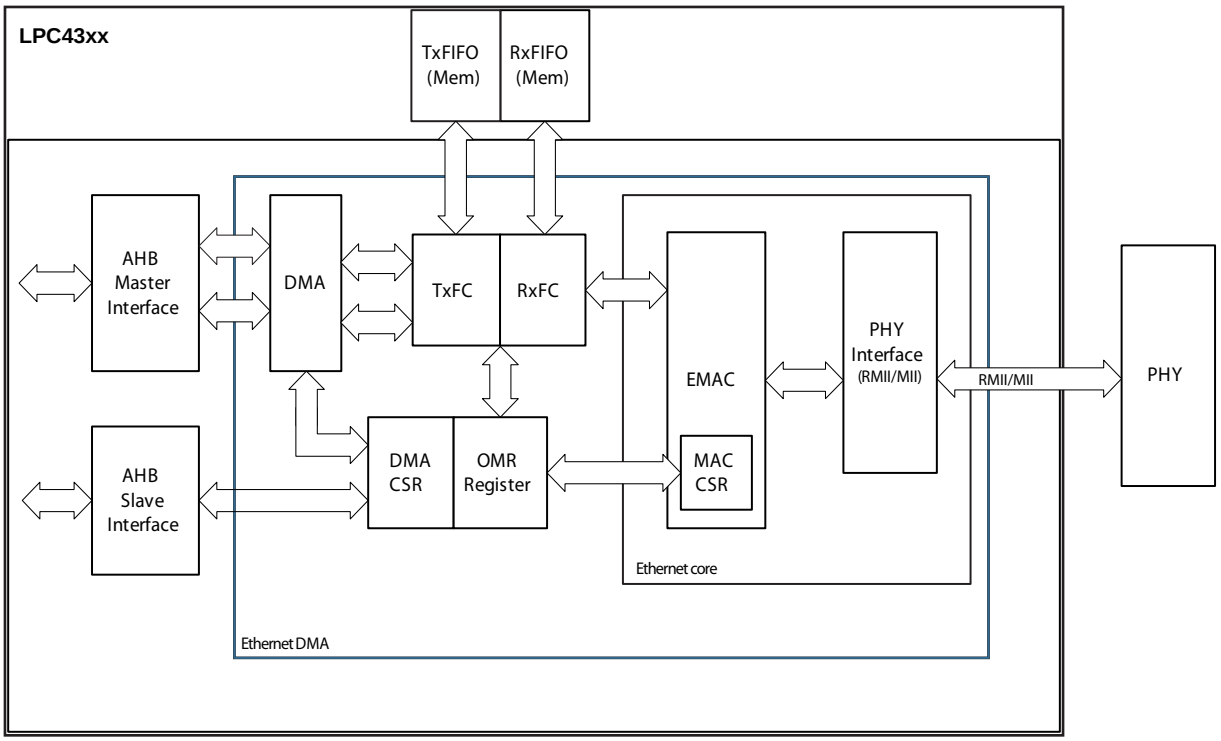


Fig 60. Ethernet block diagram

26.5 Pin description

Table 520. Ethernet pin description

Pin function	Direction	Description
MIIM interface		
ENET_MDIO	I/O	Ethernet MIIM Data Input and Output.
ENET_MDC	O	Ethernet MIIM Clock.
RMII interface		
ENET_RXD[1:0]	I	Ethernet Receive Data.
ENET_TXD[1:0]	O	Ethernet Transmit Data.
ENET_RX_DV	I	Ethernet Receive Data Valid.
ENET_REF_CLK	I	Ethernet Reference Clock.
ENET_TX_EN	O	Ethernet Transmit Data Enable.
MII interface		
ENET_RXD[3:0]	I	Ethernet Receive Data.
ENET_TXD[3:0]	O	Ethernet Transmit Data.
ENET_COL	I	Ethernet Collision detect.
ENET_CRS	I	Ethernet Carrier Sense.
ENET_TX_ER	O	Ethernet Transmit Error.
ENET_TX_CLK	I	Ethernet Transmit Clock.

Table 520. Ethernet pin description

Pin function	Direction	Description
ENET_RX_CLK	I	Ethernet Receive Clock.
ENET_RX_ER	I	Ethernet Receive Error.
ENET_TX_EN	O	Ethernet Transmit Enable.

26.6 Register description

Table 521. Register overview: Ethernet MAC and DMA (base address 0x4001 0000)

Name	Access	Address offset	Description	Reset value	Reference
MAC_CONFIG	R/W	0x0000	MAC configuration register	0x0000 8000	Table 522
MAC_FRAME_FILTER	R/W	0x0004	MAC frame filter	0x0000 0000	Table 523
MAC_HASHTABLE_HIGH	R/W	0x0008	Hash table high register	0x0000 0000	Table 524
MAC_HASHTABLE_LOW	R/W	0x000C	Hash table low register	0x0000 0000	Table 525
MAC_MII_ADDR	R/W	0x0010	MII address register	0x0000 0000	Table 526
MAC_MII_DATA	R/W	0x0014	MII data register	0x0000 0000	Table 528
MAC_FLOW_CTRL	R/W	0x0018	Flow control register	0x0000 0000	Table 529
MAC_VLAN_TAG	R/W	0x001C	VLAN tag register	0x0000 0000	Table 530
-	-	0x0020	Reserved	-	-
MAC_DEBUG	RO	0x0024	Debug register	0x0000 0000	Table 531
MAC_RWAKE_FRFLT	R/W	0x0028	Remote wake-up frame filter	0x0000 0000	Table 532
MAC_PMT_CTRL_STAT	R/W	0x002C	PMT control and status	0x0000 0000	Table 533
-	-	0x0030 - 0x0034	Reserved	-	-
MAC_INTR	RO	0x0038	Interrupt status register	0x0000 0000	Table 534
MAC_INTR_MASK	R/W	0x003C	Interrupt mask register	0x0000 0000	Table 535
MAC_ADDR0_HIGH	R/W	0x0040	MAC address 0 high register	0x8000 FFFF	Table 536
MAC_ADDR0_LOW	R/W	0x0044	MAC address 0 low register	0xFFFF FFFF	Table 537
-	-	0x0048 - 0x06FC	Reserved	-	-
MAC_TIMESTP_CTRL	R/W	0x0700	Time stamp control register	0x0000 2000	Table 538
SUBSECOND_INCR	R/W	0x0704	Sub-second increment register	0x0000 0000	Table 540
SECONDS	RO	0x0708	System time seconds register	0x0000 0000	Table 541
NANOSECONDS	RO	0x070C	System time nanoseconds register	0x0000 0000	Table 542
SECONDSUPDATE	R/W	0x0710	System time seconds update register	0x0000 0000	Table 543
NANOSECONDSUPDATE	R/W	0x0714	System time nanoseconds update register	0x0000 0000	Table 544
ADDEND	R/W	0x0718	Time stamp addend register	0x0000 0000	Table 545
TARGETSECONDS	R/W	0x071C	Target time seconds register	0x0000 0000	Table 546
TARGETNANOSECONDS	R/W	0x0720	Target time nanoseconds register	0x0000 0000	Table 547
HIGHWORD	R/W	0x0724	System time higher word seconds register	0x0000 0000	Table 548
TIMESTAMPSTAT	RO	0x0728	Time stamp status register	0x0000 0000	Table 549
PPSCTRL	R/W	0x072C	PPS control register	0x0000 0000	Table 550
AUXNANOSECONDS	RO	0x0730	Auxiliary time stamp nanoseconds register	0x0000 0000	Table 551

Table 521. Register overview: Ethernet MAC and DMA (base address 0x4001 0000)

Name	Access	Address offset	Description	Reset value	Reference
AUXSECONDS	RO	0x0734	Auxiliary time stamp seconds register	0x0000 0000	Table 552
-		0x0738 - 0x0FFC	Reserved		-
DMA_BUS_MODE	R/W	0x1000	Bus Mode Register	0x0002 0100	Table 553
DMA_TRANS_POLL_DEMAND	R/W	0x1004	Transmit poll demand register	0x0000 0000	Table 555
DMA_REC_POLL_DEMAND	R/W	0x1008	Receive poll demand register	0x0000 0000	Table 556
DMA_REC_DES_ADDR	R/W	0x100C	Receive descriptor list address register	0x0000 0000	Table 557
DMA_TRANS_DES_ADDR	R/W	0x1010	Transmit descriptor list address register	0x0000 0000	Table 558
DMA_STAT	R/W	0x1014	Status register	0x0000 0000	Table 559
DMA_OP_MODE	R/W	0x1018	Operation mode register	0x0000 0000	Table 560
DMA_INT_EN	R/W	0x101C	Interrupt enable register	0x0000 0000	Table 561
DMA_MFRM_BUFOF	RO	0x1020	Missed frame and buffer overflow register	0x0000 0000	Table 562
DMA_REC_INT_WDT	R/W	0x1024	Receive interrupt watchdog timer register	0x0000 0000	Table 563
-	-	0x1028 - 0x1044	Reserved	-	-
DMA_CURHOST_TRANS_DES	RO	0x1048	Current host transmit descriptor register	0x0000 0000	Table 564
DMA_CURHOST_REC_DES	RO	0x104C	Current host receive descriptor register	0x0000 0000	Table 565
DMA_CURHOST_TRANS_BUF	RO	0x1050	Current host transmit buffer address register	0x0000 0000	Table 566
DMA_CURHOST_REC_BUF	RO	0x1054	Current host receive buffer address register	0x0000 0000	Table 567
-		0x1058	-	-	-

26.6.1 MAC Configuration register

The MAC Configuration register establishes receive and transmit operating modes.

Table 522. MAC Configuration register (MAC_CONFIG, address 0x4001 0000) bit description

Bit	Symbol	Description	Reset value	Access
1:0	-	Reserved	00	RO
2	RE	Receiver enable When this bit is set, the receiver state machine of the MAC is enabled for receiving frames from the MII. When this bit is reset, the MAC receive state machine is disabled after the completion of the reception of the current frame, and will not receive any further frames from the MII.	0	R/W
3	TE	Transmitter Enable When this bit is set, the transmit state machine of the MAC is enabled for transmission on the MII. When this bit is reset, the MAC transmit state machine is disabled after the completion of the transmission of the current frame, and will not transmit any further frames.	0	R/W
4	DF	Deferral Check When this bit is set, the deferral check function is enabled in the MAC. The MAC will issue a Frame Abort status, along with the excessive deferral error bit set in the transmit frame status when the transmit state machine is deferred for more than 24,288 bit times in 10/100-Mbps mode. If the Core is configured for 1000 Mbps operation, or if the Jumbo frame mode is enabled in 10/100-Mbps mode, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but is prevented because of an active CRS (carrier sense) signal on the MII. Defer time is not cumulative. If the transmitter defers for 10,000 bit times, then transmits, collides, backs off, and then has to defer again after completion of back-off, the deferral timer resets to 0 and restarts. When this bit is reset, the deferral check function is disabled and the MAC defers until the CRS signal goes inactive. This bit is applicable only in Half-Duplex mode and is reserved (RO) in Full-Duplex-only configuration.	0	R/W
6:5	BL	Back-Off Limit The Back-Off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000 Mbps and 512 bit times for 10/100 Mbps) the MAC waits before rescheduling a transmission attempt during retries after a collision. This bit is applicable only to Half-Duplex mode and is reserved (RO) in Full-Duplex-only configuration. • 00: k = min (n, 10) • 01: k = min (n, 8) • 10: k = min (n, 4) • 11: k = min (n, 1) where n = retransmission attempt. The random integer r takes the value in the range $0 \leq r \leq 2^k$.	0	R/W
7	ACS	Automatic Pad/CRC Stripping When this bit is set, the MAC strips the Pad/FCS field on incoming frames only if the length's field value is less than or equal to 1,500 bytes. All received frames with length field greater than or equal to 1,501 bytes are passed to the application without stripping the Pad/FCS field. When this bit is reset, the MAC will pass all incoming frames to the Host unmodified.	0	R/W

Table 522. MAC Configuration register (MAC_CONFIG, address 0x4001 0000) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
8	-	Link Up/Down Indicates whether the link is up or down during the transmission of configuration in SMII interface: 0 = Link down 1 = Link up	0	R/W
9	DR	Disable Retry When this bit is set, the MAC will attempt only 1 transmission. When a collision occurs on the MII, the MAC will ignore the current frame transmission and report a Frame Abort with excessive collision error in the transmit frame status. When this bit is reset, the MAC will attempt retries based on the settings of BL. This bit is applicable only to Half-Duplex mode and is reserved (RO with default value) in Full- Duplex-only configuration.	0	R/W
10	IPC	Checksum Offload When this bit is set, the MAC calculates the 16-bit one's complement of the one's complement sum of all received Ethernet frame payloads. It also checks whether the IPv4 Header checksum (assumed to be bytes 25 to 26 or 29 to 30 (VLAN-tagged) of the received Ethernet frame) is correct for the received frame and gives the status in the receive status word. The MAC core also appends the 16-bit checksum calculated for the IP header datagram payload (bytes after the IPv4 header) and appends it to the Ethernet frame transferred to the application (when Type 2 COE is deselected). When this bit is reset, this function is disabled. When Type 2 COE is selected, this bit, when set, enables IPv4 checksum checking for received frame payload's TCP/UDP/ICMP headers. When this bit is reset, the COE function in the receiver is disabled and the corresponding PCE and IP HCE status bits are always cleared.	0	R/W
11	DM	Duplex Mode When this bit is set, the MAC operates in a Full-Duplex mode where it can transmit and receive simultaneously.	0	R/W
12	LM	Loopback Mode When this bit is set, the MAC operates in loopback mode at MII. The (G)MII Receive clock input is required for the loopback to work properly, as the Transmit clock is not looped-back internally.	0	R/W
13	DO	Disable Receive Own When this bit is set, the MAC disables the reception of frames in Half-Duplex mode. When this bit is reset, the MAC receives all packets that are given by the PHY while transmitting. This bit is not applicable if the MAC is operating in Full-Duplex mode.	0	R/W
14	FES	Speed Indicates the speed in Fast Ethernet (MII) mode: 0 = 10 Mbps 1 = 100 Mbps	0	
15	PS	Port select 1 = MII (100 Mbp) - this is the only allowed value.	1	RO

Table 522. MAC Configuration register (MAC_CONFIG, address 0x4001 0000) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
16	DCRS	Disable carrier sense during transmission When set high, this bit makes the MAC transmitter ignore the (G)MII CRS signal during frame transmission in Half-Duplex mode. This request results in no errors generated due to Loss of Carrier or No Carrier during such transmission. When this bit is low, the MAC transmitter generates such errors due to Carrier Sense and will even abort the transmissions.	0	R/W
19:17	IFG	Inter-frame gap These bits control the minimum IFG between frames during transmission. 000 = 96 bit times 001 = 88 bit times 010 = 80 bit times ... 000 = 40 bit times Note that in Half-Duplex mode, the minimum IFG can be configured for 64 bit times (IFG = 100) only. Lower values are not considered	000	R//W
20	JE	Jumbo Frame Enable When this bit is set, MAC allows Jumbo frames of 9,018 bytes (9,022 bytes for VLAN tagged frames) without reporting a giant frame error in the receive frame status.	0	R/W
21	-	Reserved.	0	RO
22	JD	Jabber Disable When this bit is set, the MAC disables the jabber timer on the transmitter, and can transfer frames of up to 16,384 bytes. When this bit is reset, the MAC cuts off the transmitter if the application sends out more than 2,048 bytes of data (10,240 if JE is set high) during transmission.	0	R/W
23	WD	Watchdog Disable When this bit is set, the MAC disables the watchdog timer on the receiver, and can receive frames of up to 16,384 bytes. When this bit is reset, the MAC allows no more than 2,048 bytes (10,240 if JE is set high) of the frame being received and cuts off any bytes received after that.	0	R/W
31:24	-	Reserved.	0x00	RO

26.6.2 MAC Frame filter register

The MAC Frame Filter register contains the filter controls for receiving frames. Some of the controls from this register go to the address check block of the MAC, which performs the first level of address filtering. The second level of filtering is performed on the incoming frame, based on other controls such as Pass Bad Frames and Pass Control Frames.

Table 523. MAC Frame filter register (MAC_FRAME_FILTER, address 0x4001 0004) bit description

Bit	Symbol	Description	Reset value	Access
0	PR	Promiscuous Mode When this bit is set, the Address Filter module passes all incoming frames regardless of its destination or source address. The SA/DA Filter Fails status bits of the Receive Status Word will always be cleared when PR is set.	0	R/W
1	-	Reserved	0	RO
2	-	Reserved	0	RO
3	DAIF	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast frames. When reset, normal filtering of frames is performed.	0	R/W
4	PM	Pass All Multicast When set, this bit indicates that all received frames with a multicast destination address (first bit in the destination address field is '1') are passed. When reset, filtering of multicast frame depends on HMC bit.	0	R/W
5	DBF	Disable Broadcast Frames When this bit is set, the AFM module filters all incoming broadcast frames. When this bit is reset, the AFM module passes all received broadcast frames.	0	R/W
7:6	PCF	Pass Control Frames These bits control the forwarding of all control frames (including unicast and multicast PAUSE frames). Note that the processing of PAUSE control frames depends only on RFE of Flow Control Register[2]. 00 = MAC filters all control frames from reaching the application. 01 = MAC forwards all control frames except PAUSE control frames to application even if they fail the Address filter. 10 = MAC forwards all control frames to application even if they fail the Address Filter. 11 = MAC forwards control frames that pass the Address Filter.	00	R/W
8	SAIF	SA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the SA address comparison. The frames whose SA matches the SA registers will be marked as failing the SA Address filter. When this bit is reset, frames whose SA does not match the SA registers will be marked as failing the SA Address filter.	0	R/W

Table 523. MAC Frame filter register (MAC_FRAME_FILTER, address 0x4001 0004) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
9	SAF	Source Address Filter Enable The MAC core compares the SA field of the received frames with the values programmed in the enabled SA registers. If the comparison matches, then the SAMatch bit of RxStatus Word is set high. When this bit is set high and the SA filter fails, the MAC drops the frame. When this bit is reset, then the MAC Core forwards the received frame to the application and with the updated SA Match bit of the RxStatus depending on the SA address comparison.	0	R/W
30:10	-	Reserved	0	RO
31	RA	Receive all When this bit is set, the MAC Receiver module passes to the Application all frames received irrespective of whether they pass the address filter. The result of the SA/DA filtering is updated (pass or fail) in the corresponding bits in the Receive Status Word. When this bit is reset, the Receiver module passes to the Application only those frames that pass the SA/DA address filter.	0	R/W

26.6.3 MAC Hash table high register

The 64-bit Hash table is used for group address filtering. For hash filtering, the contents of the destination address in the incoming frame is passed through the CRC logic, and the upper 6 bits of the CRC register are used to index the contents of the Hash table. The most significant bit determines the register to be used (Hash Table High/Hash Table Low), and the other 5 bits determine which bit within the register. A hash value of 00000 selects Bit 0 of the selected register, and a value of 11111 selects Bit 31 of the selected register.

For example, if the DA of the incoming frame is received as 0x1F52419CB6AF (0x1F is the first byte received on MII interface), then the internally calculated 6-bit Hash value is 0x2C and the HTH register bit[12] is checked for filtering. If the DA of the incoming frame is received as 0xA00A98000045, then the calculated 6- bit Hash value is 0x07 and the HTL register bit[7] is checked for filtering.

If the corresponding bit value of the register is 1, the frame is accepted. Otherwise, it is rejected. If the PM (Pass All Multicast) bit is set in the MAC_CONFIG register, then all multicast frames are accepted regardless of the multicast hash values.

If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in Little-Endian mode) or Bits[7:0] (in Big-Endian mode) of the Hash Table High/Low registers are written to. Please note that consecutive writes to these register should be performed only after at least 4 clock cycles in the destination clock domain when double synchronization is enabled.

The Hash Table High register contains the higher 32 bits of the Hash table.

Table 524. MAC Hash table high register (MAC_HASHTABLE_HIGH, address 0x4001 0008) bit description

Bit	Symbol	Description	Reset value	Access
31:0	HTH	Hash table high This field contains the upper 32 bits of Hash table.	0	R/W

26.6.4 MAC Hash table low register

The Hash Table Low register contains the lower 32 bits of the Hash table.

Table 525. MAC Hash table low register (MAC_HASHTABLE_LOW, address 0x4001 0008) bit description

Bit	Symbol	Description	Reset value	Access
31:0	HTL	Hash table low This field contains the upper 32 bits of Hash table.	0	R/W

26.6.5 MAC MII Address register

The MII Address register controls the management cycles to the external PHY through the management interface.

Table 526. MAC MII Address register (MAC_MII_ADDR, address 0x4001 0010) bit description

Bit	Symbol	Description	Reset value	Access
0	GB	MII busy This register field can be read by the application (Read), can be set to 1 by the application with a register write of 1 (Write Set), and is cleared to 0 by the core (Self Clear). The application cannot clear this type of field, and a register write of 0 to this bit has no effect on this field. This bit should read a logic 0 before writing to this register and the MAC_MII_DATA register. This bit must also be set to 0 during a Write to this register. During a PHY register access, this bit will be set to 1 by the Application to indicate that a Read or Write access is in progress. The MAC_MII_DATA register should be kept valid until this bit is cleared by the MAC during a PHY Write operation. The MAC_MII_DATA register is invalid until this bit is cleared by the MAC during a PHY Read operation. This register should not be written to until this bit is cleared.	0	R/W
1	W	MII write When set, this bit tells the PHY that this will be a Write operation using the MII Data register. If this bit is not set, this will be a Read operation, placing the data in the MII Data register.	0	R/W
5:2	CR	CSR clock range The CSR Clock Range selection determines the frequency of the MDC clock. The suggested range of CLK_M3_ETHERNET frequency applicable for each value below (when Bit[5] = 0) ensures that the MDC clock is approximately between the frequency range 1.0 MHz - 2.5 MHz. When bit 5 is set, you can achieve MDC clock of frequency higher than the IEEE 802.3 specified frequency limit of 2.5 MHz and program a clock divider of lower value. For example, when CLK_M3_ETHERNET is of frequency 100 MHz and you program these bits as 1010, then the resultant MDC clock will be of 12.5 MHz which is outside the limit of IEEE 802.3 specified range. Program the values given below only if the interfacing chips supports faster MDC clocks. See Table 527 for bit values.	0	R/W

Table 526. MAC MII Address register (MAC_MII_ADDR, address 0x4001 0010) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
10:6	GR	MII register These bits select the desired MII register in the selected PHY device.	0	R/W
15:11	PA	Physical layer address This field tells which of the 32 possible PHY devices are being accessed.	0	R/W
31:16	-	Reserved	0	RO

Table 527. CSR clock range values

Bits 5:2	CLK_M3_ETHERNET	MDC clock
0000	60 - 100 MHz	CLK_M3_ETHERNET/42
0001	100 - 150 MHz	CLK_M3_ETHERNET/62
0010	20 - 35 MHz	CLK_M3_ETHERNET/16
0011	35 - 60 MHz	CLK_M3_ETHERNET/26
0100	150 - 250 MHz	CLK_M3_ETHERNET/102
0101	250 - 300 MHz	CLK_M3_ETHERNET/124
0110, 0111	Reserved	-
1000	-	CLK_M3_ETHERNET/42
1001	-	CLK_M3_ETHERNET/62
1010	-	CLK_M3_ETHERNET/16
1011	-	CLK_M3_ETHERNET/26
1100	-	CLK_M3_ETHERNET/102
1101	-	CLK_M3_ETHERNET/124
1110	-	CLK_M3_ETHERNET/42
1111	-	CLK_M3_ETHERNET/62

26.6.6 MAC MII Data register

The MII Data register stores Write data to be written to the PHY register located at the address specified in the MAC_MII_ADDR register. This register also stores Read data from the PHY register located at the address specified by the MAC_MII_ADDR register.

Table 528. MII Data register (MAC_MII_DATA, address 0x4001 0014) bit description

Bit	Symbol	Description	Reset value	Access
15:0	GD	MII data This contains the 16-bit data value read from the PHY after a Management Read operation or the 16-bit data value to be written to the PHY before a Management Write operation.	0	R/W
31:16	-	Reserved	0	RO

26.6.7 MAC Flow control register

The Flow Control register controls the generation and reception of the Control (Pause Command) frames by the MAC's Flow control module. A Write to a register with the Busy bit set to 1 triggers the Flow Control block to generate a Pause Control frame. The fields

of the control frame are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control frame. The Busy bit remains set until the control frame is transferred onto the cable. The Host must make sure that the Busy bit is cleared before writing to the register.

Table 529. MAC Flow control register (MAC_FLOW_CTRL, address 0x4001 0018) bit description

Bit	Symbol	Description	Reset value	Access
0	FCB	Flow Control Busy/Backpressure Activate This register field can be read by the application (Read), can be set to 1 by the application with a register write of 1 (Write Set), and is cleared to 0 by the core (Self Clear). The application cannot clear this type of field, and a register write of 0 to this bit has no effect on this field. This bit initiates a Pause Control frame in Full-Duplex mode. In Full-Duplex mode, this bit should be read as 0 before writing to the Flow Control register. To initiate a Pause control frame, the Application must set this bit to 1. During a transfer of the Control Frame, this bit will continue to be set to signify that a frame transmission is in progress. After the completion of Pause control frame transmission, the MAC will reset this bit to 0. The Flow Control register should not be written to until this bit is cleared. In Half-Duplex mode, when this bit is set (and TFE is set), then backpressure is asserted by the MAC Core. During backpressure, when the MAC receives a new frame, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically ORed with the mti_flowctrl_i input signal for the backpressure function. When the MAC is configured to Full- Duplex mode, the BPA is automatically disabled.	0	R/W
1	TFE	Transmit Flow Control Enable In Full-Duplex mode, when this bit is set, the MAC enables the flow control operation to transmit Pause frames. When this bit is reset, the flow control operation in the MAC is disabled, and the MAC will not transmit any Pause frames. In Half-Duplex mode, when this bit is set, the MAC enables the back-pressure operation. When this bit is reset, the backpressure feature is disabled.	0	R/W
2	RFE	Receive Flow Control Enable When this bit is set, the MAC will decode the received Pause frame and disable its transmitter for a specified (Pause Time) time. When this bit is reset, the decode function of the Pause frame is disabled.	0	R/W
3	UP	Unicast Pause Frame Detect When this bit is set, the MAC will detect the Pause frames with the station's unicast address specified in MAC Address0 High Register and MAC Address0 Low Register, in addition to the detecting Pause frames with the unique multicast address. When this bit is reset, the MAC will detect only a Pause frame with the unique multicast address specified in the 802.3x standard.	0	R/W
5:4	PLT	Pause Low Threshold This field configures the threshold of the PAUSE timer at which the input flow control is checked for automatic retransmission of PAUSE Frame. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = Ox100 (256 slot-times), and PLT = 01, then a second PAUSE frame is automatically transmitted if the flow control signal is asserted at 228 (256 – 28) slot-times after the first PAUSE frame is transmitted.	00	R/W
6	-	Reserved	0x000	RO

Table 529. MAC Flow control register (MAC_FLOW_CTRL, address 0x4001 0018) bit description *...continued*

Bit	Symbol	Description	Reset value	Access
7	DZPQ	Disable Zero-Quanta Pause When set, this bit disables the automatic generation of Zero-Quanta Pause Control frames on the deassertion of the flow-control signal from the FIFO layer. When this bit is reset, normal operation with automatic Zero-Quanta Pause Control frame generation is enabled.	0	R/W
15:8	-	Reserved	0	RO
31:16	PT	Pause time This field holds the value to be used in the Pause Time field in the transmit control frame. If the Pause Time bits is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to this register should be performed only after at least 4 clock cycles in the destination clock domain.	0x0000	R/W

26.6.8 MAC VLAN tag register

The VLAN Tag register contains the IEEE 802.1Q VLAN Tag to identify the VLAN frames. The MAC compares the 13th and 14th bytes of the receiving frame (Length/Type) with 0x8100, and the following 2 bytes are compared with the VLAN tag; if a match occurs, it sets the received VLAN bit in the receive frame status. The legal length of the frame is increased from 1518 bytes to 1522 bytes.

If the VLAN Tag register is configured to be double-synchronized to the MII clock domain, then consecutive writes to these register should be performed only after at least 4 clock cycles in the destination clock domain.

Table 530. MAC VLAN tag register (MAC_VLAN_TAG, address 0x4001 01C) bit description

Bit	Symbol	Description	Reset value	Access
15:0	VL	VLAN Tag Identifier for Receive Frames This contains the 802.1Q VLAN tag to identify VLAN frames, and is compared to the fifteenth and sixteenth bytes of the frames being received for VLAN frames. Bits[15:13] are the User Priority, Bit[12] is the Canonical Format Indicator (CFI) and bits[11:0] are the VLAN tag's VLAN Identifier (VID) field. When the ETV bit is set, only the VID (Bits[11:0]) is used for comparison. If VL (VL[11:0] if ETV is set) is all zeros, the MAC does not check the fifteenth and sixteenth bytes for VLAN tag comparison, and declares all frames with a Type field value of 0x8100 to be VLAN frames.	0x0000	R/W
16	ETV	Enable 12-Bit VLAN Tag Comparison When this bit is set, a 12-bit VLAN identifier, rather than the complete 16-bit VLAN tag, is used for comparison and filtering. Bits[11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged frame. When this bit is reset, all 16 bits of the received VLAN frame's fifteenth and sixteenth bytes are used for comparison.	0	R/W
31:17	-	Reserved	0x0000	RO

26.6.9 MAC Debug register

This debug register gives the status of all the main modules of the transmit and receive data-paths and the FIFOs. An all-zero status indicates that the MAC core is in idle state (and FIFOs are empty) and no activity is going on in the data-paths.

Remark: Reset values in this register are valid only if the clocks to the Ethernet block are present during the reset operation.

Table 531. MAC Debug register (MAC_DEBUG, address 0x4001 0024) bit description

Bit	Symbol	Description	Reset value	Access
0	RXIDLESTAT	When high, it indicates that the MAC MII receive protocol engine is actively receiving data and not in IDLE state.	0	RO
2:1	FIFOSTAT0	When high, it indicates the active state of the small FIFO Read and Write controllers respectively of the MAC receive Frame Controller module.	0	RO
3	-	Reserved	-	RO
4	RXFIFOSTAT1	When high, it indicates that the MTL RxFIFO Write Controller is active and transferring a received frame to the FIFO.	0	RO
6:5	RXFIFOSTAT	State of the RxFIFO read Controller: 00 = idle state 01 = reading frame data 10 = reading frame status (or time stamp) 11 = flushing the frame data and status	0	RO
7	-	Reserved	-	RO
9:8	RXFIFOLVL	Status of the RxFIFO Fill-level 00 = RxFIFO Empty 01 = RxFIFO fill-level below flow-control de-activate threshold 10 = RxFIFO fill-level above flow-control activate threshold 11 = RxFIFO Full	0	RO
15:10	-	Reserved	-	RO
16	TXIDLESTAT	When high, it indicates that the MAC MII transmit protocol engine is actively transmitting data and not in IDLE state.	0	RO
18:17	TXSTAT	State of the MAC Transmit Frame Controller module: 00 = idle 01 = Waiting for Status of previous frame or IFG/backoff period to be over 10 = Generating and transmitting a PAUSE control frame (in full duplex mode) 11 = Transferring input frame for transmission	0	RO
19	PAUSE	When high, it indicates that the MAC transmitter is in PAUSE condition (in full-duplex only) and hence will not schedule any frame for transmission.	0	RO
21:20	TXFIFOSTAT	State of the TxFIFO read Controller 00 = idle state 01 = READ state (transferring data to MAC transmitter) 10 = Waiting for TxStatus from MAC transmitter 11 = Writing the received TxStatus or flushing the TxFIFO	0	RO
22	TXFIFOSTAT1	When high, it indicates that the MTL TxFIFO Write Controller is active and transferring data to the TxFIFO.	0	RO
23	-	Reserved	0	RO
24	TXFIFOLVL	When high, it indicates that the MTL TxFIFO is not empty and has some data left for transmission.	0	RO
25	TXFIFOFULL	When high, it indicates that the MTL TxStatus FIFO is full and hence the MTL will not be accepting any more frames for transmission.	0	RO
31:26	-	Reserved	-	RO

26.6.10 MAC Remote wake-up frame filter register

This is the address through which the remote Wake-up Frame Filter registers (WKUPFMFILTER) are written/read by the Application. WKUPFMFILTER is actually a pointer to eight (not transparent) such WKUPFMFILTER registers. Eight sequential Writes to this address (0x028) will write all WKUPFMFILTER registers. Eight sequential Reads from this address (0x028) will read all WKUPFMFILTER registers. See Section 26.7.1.1 for details.

Remark: Do not use bit-banding for this register.

Table 532. MAC Remote wake-up frame filter register (MAC_RWAKE_FRFLT, address 0x4001 0028) bit description

Bit	Symbol	Description	Reset value	Access
31:0	ADDR	WKUPFMFILTER address	-	R/W

26.6.11 MAC PMT control and status register

The PMT control and status registers programs the request wake-up events and monitors the wake-up events. See Section 26.7.1 for details.

Table 533. MAC PMT control and status register (MAC_PMT_CTRL_STAT, address 0x4001 002C) bit description

Bit	Symbol	Description	Reset value	Access
0	PD	Power-down This register field can be read by the application (Read), can be set to 1 by the application with a register write of 1 (Write Set), and is cleared to 0 by the core (Self Clear). The application cannot clear this type of field, and a register write of 0 to this bit has no effect on this field. When set, all received frames will be dropped. This bit is cleared automatically when a magic packet or Wake-Up frame is received, and Power-Down mode is disabled. Frames received after this bit is cleared are forwarded to the application.This bit must only be set when either the Magic Packet Enable or Wake- Up Frame Enable bit is set high.	0	R/W
1	MPE	Magic packet enable When set, enables generation of a power management event due to Magic Packet reception.	0	R/W
2	WFE	Wake-up frame enable When set, enables generation of a power management event due to wake-up frame reception.	0	R/W
4:3	-	Reserved	00	RO
5	MPR	Magic Packet Received This register field can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field. When set, this bit indicates the power management event was generated by the reception of a Magic Packet. This bit is cleared by a Read into this register.	0	RO

Table 533. MAC PMT control and status register (MAC_PMT_CTRL_STAT, address 0x4001 002C) bit description

Bit	Symbol	Description	Reset value	Access
6	WFR	Wake-up Frame Received This register field can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field. When set, this bit indicates the power management event was generated due to reception of a wake-up frame. This bit is cleared by a Read into this register.	0	RO
8:7	-	Reserved	0	RO
9	GU	Global Unicast When set, enables any unicast packet filtered by the MAC (DAF) address recognition to be a wake-up frame.	0	R/W
30:10	-	Reserved	0x00 0000	RO
31	WFFRPR	Wake-up Frame Filter Register Pointer Reset This register field can be read by the application (Read), can be set to 1 by the application with a register write of 1 (Write Set), and is cleared to 0 by the core (Self Clear). The application cannot clear this type of field, and a register write of 0 to this bit has no effect on this field. When set, resets the Remote Wake-up Frame Filter register pointer to 000. It is automatically cleared after 1 clock cycle.	0	R/W

26.6.12 MAC Interrupt status register

The Interrupt Status register contents identify the events in the MAC-CORE that can generate interrupt.

Table 534. MAC Interrupt status register (MAC_INTR, address 0x4001 0038) bit description

Bit	Symbol	Description	Reset value	Access
31:0	-	Reserved	0	RO

26.6.13 MAC Interrupt mask register

The Interrupt Mask Register bits enables the user to mask the interrupt signal due to the corresponding event in the Interrupt Status Register.

Table 535. MAC Interrupt mask register (MAC_INTR_MASK, address 0x4001 003C) bit description

Bit	Symbol	Description	Reset value	Access
2:0	-	Reserved	0	RO
3	PMTMSK	PMT Interrupt Mask This bit when set, will disable the assertion of the interrupt signal due to the setting of PMT Interrupt Status bit in Table 534.	0	R/W
31:4		Reserved	0	R/W

26.6.14 MAC Address 0 high register

The MAC Address 0 High register holds the upper 16 bits of the 6-byte first MAC address of the station. Note that the first DA byte that is received on the (G)MII interface corresponds to the LS Byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 is the first byte) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211.

If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in Little-Endian mode) or Bits[7:0] (in Big-Endian mode) of the MAC Address Low Register (Register 17) are written to. Please note that consecutive writes to this Address Low Register should be performed only after at least 4 clock cycles in the destination clock domain for proper synchronization updates.

Table 536. MAC Address 0 high register (MAC_ADDR0_HIGH, address 0x4001 0040) bit description

Bit	Symbol	Description	Reset value	Access
15:0	A47_32	MAC Address0 [47:32] This field contains the upper 16 bits (47:32) of the 6-byte first MAC address. This is used by the MAC for filtering for received frames and for inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.	0xFFFF	R/W
30:16	-	Reserved	0x0000	RO
31	MO	Always 1	1	RO

26.6.15 MAC Address 0 low register

The MAC Address 0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.

Table 537. MAC Address 0 low register (MAC_ADDR0_LOW, address 0x4001 0044) bit description

Bit	Symbol	Description	Reset value	Access
31:0	A31_0	MAC Address0 [31:0] This field contains the lower 32 bits of the 6-byte first MAC address. This is used by the MAC for filtering for received frames and for inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.	0xFFFF FFFF	R/W

26.6.16 MAC IEEE1588 time stamp control register

This register controls the operation of the System Time generator and the snooping of PTP packets for time-stamping in the Receiver.

Table 538. MAC IEEE1588 time stamp control register (MAC_TIMESTP_CTRL, address 0x4001 0700) bit description

Bit	Symbol	Description	Reset value	Access
0	TSENA	Time Stamp Enable When this bit, is set the timestamping is enabled for transmit and receive frames. When disabled timestamp is not added for transmit and receive frames and the TimeStamp Generator is also suspended. User has to always initialize the TimeStamp (system time) after enabling this mode.	0	R/W
1	TSCFUPDT	Time Stamp Fine or Coarse Update When set, indicates that the system times update to be done using fine update method. When reset it indicates the system time stamp update to be done using Coarse method. This bit is reserved if the fine correction option is not enabled.	0	R/W
2	TSINIT	Time Stamp Initialize This register field can be read and written by the application (Read and Write), and is cleared to 0 by the Ethernet core (Self Clear). When set, the system time is initialized (over-written) with the value specified in the Time Stamp High Update and Time Stamp Low Update registers. This register bit should be read zero before updating it. This bit is reset once the initialize is complete.	0	R/W
3	TSUPDT	Time Stamp Update This register field can be read and written by the application (Read and Write), and is cleared to 0 by the Ethernet core (Self Clear). When set, the system time is updated (added/subtracted) with the value specified in the Time Stamp High Update and Time Stamp Low Update registers. This register bit should be read zero before updating it. This bit is reset once the update is completed in hardware.	0	R/W
4	TSTRIG	Time Stamp Interrupt Trigger Enable This register field can be read and written by the application (Read and Write), and is cleared to 0 by the Ethernet core (Self Clear). When set, the Time Stamp interrupt is generated when the System Time becomes greater than the value written in Target Time register. This bit is reset after the generation of Time Stamp Trigger Interrupt.	0	R/W
5	TSADDREG	Addend Reg Update When set, the contents of the Time Stamp Addend register is updated in the PTP block for fine correction. This is cleared when the update is completed. This register bit should be zero before setting it. This is a reserved bit when only coarse correction option is selected.		
7:6	-	Reserved		
8	TSENALL	Enable Time Stamp for All Frames When set, the time stamp snapshot is enabled for all frames received by the core.	0	R/W
9	TSCTRLSSR	Time Stamp Digital or Binary rollover control When set, the Time Stamp Low register rolls over after 0x3B9A_C9FF value (i.e., 1 nanosecond accuracy) and increments the Time Stamp (High) seconds. When reset, the rollover value of sub-second register is 0x7FFF_FFFF. The sub-second increment has to be programmed correctly depending on the PTP reference clock frequency and this bit value.	0	R/W
10	TSVER2ENA	Enable PTP packet snooping for version 2 format When set, the PTP packets are snooped using the 1588 version 2 format else snooped using the version 1 format.	0	R/W

Table 538. MAC IEEE1588 time stamp control register (MAC_TIMESTP_CTRL, address 0x4001 0700) bit description

Bit	Symbol	Description	Reset value	Access
11	TSIPENA	Enable Time Stamp Snapshot for PTP over Ethernet frames When set, the time stamp snapshot is taken for frames which have PTP messages in Ethernet frames (PTP over Ethernet) also. By default snapshots are taken for UDP-IP-Ethernet PTP packets.	0	R/W
12	TSIPV6ENA	Enable Time Stamp Snapshot for IPv6 frames When set, the time stamp snapshot is taken for IPv6 frames.	0	R/W
13	TSIPV4ENA	Enable Time Stamp Snapshot for IPv4 frames When set, the time stamp snapshot is taken for IPv4 frames.	1	R/W
14	TSEVNTENA	Enable Time Stamp Snapshot for Event Messages When set, the time stamp snapshot is taken for event messages only. When reset snapshot is taken for all other messages except Announce, Management and Signaling.	0	R/W
15	TSMSTRENA	Enable Snapshot for Messages Relevant to Master When set, the snapshot is taken for messages relevant to master node only else snapshot is taken for messages relevant to slave node. This is valid only for ordinary clock and boundary clock node.	0	R/W
17:16	TSCLKTYPE	Select the type of clock node The following are the options to select the type of clock node: 00 = ordinary clock 01 = boundary clock 10 = end-to-end transparent clock 11 = peer-to-peer transparent clock	00	R/W
18	TSENMACADDR	Enable MAC address for PTP frame filtering When set, uses the DA MAC address (that matches any MAC Address register except the default MAC address 0) to filter the PTP frames when PTP is sent directly over Ethernet.	0	R/W
31:19				

Table 539 indicates the messages, for which a snapshot is taken depending on the clock, enable master and enable snapshot for event message register settings.

Table 539. Time stamp snapshot dependency on register bits

TSCLKTYPE	TSMSTRENA	TSEVNTENA	Messages for which snapshot is taken
00 or 01	x	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00 or 01	1	1	Delay_req
00 or 01	0	1	SYNC
10	N/A	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
10	N/A	1	SYNC, Follow_Up
11	N/A	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp
11	N/A	1	SYNC, Pdelay_Req, Pdelay_Resp

26.6.17 Sub-second increment register

This register contains the 8-bit value by which the Sub-Second register is incremented.

In Coarse Update mode (TSCFUPDT bit in Table 538), the value in this register is added to the system time every clock cycle. In Fine Update mode, the value in this register is added to the system time whenever the Accumulator gets an overflow.

Table 540. Sub-second increment register (SUBSECOND_INCR, address 0x4001 0704) bit description

Bit	Symbol	Description	Reset value	Access
7:0	SSINC	Sub-second increment value. The value programmed in this register is accumulated with the contents of the sub-second register. For example, to achieve an accuracy of 20 ns, the value to be programmed is 20. (0x14 with a 50 MHz reference clock if 1 ns accuracy is selected.)	0	R/W
31:8	-	Reserved.	0	RO

26.6.18 System time seconds register

This register contains the lower 32 bits of the seconds field of the system time.

The System Time - Seconds register, along with System Time - Nanoseconds register, indicates the current value of the system time maintained by the core. Though it is updated on a continuous basis, there is some delay from the actual time due to clock domain transfer latencies.

Table 541. System time seconds register (SECONDS, address 0x4001 0708) bit description

Bit	Symbol	Description	Reset value	Access
31:0	TSS	Time stamp second The value in this field indicates the current value in seconds of the System Time maintained by the core.	0	R/W

26.6.19 System time nanoseconds register

This register contains 32 bits of the nano-seconds field of the system time.

Table 542. System time nanoseconds register (NANOSECONDS, address 0x4001 070C) bit description

Bit	Symbol	Description	Reset value	Access
30:0	TSSS	Time stamp sub seconds The value in this field has the sub second representation of time, with an accuracy of 0.46 nano-second. (When TSCTRLSSR in the MAC_TIMESTAMP_CTRL register is set, each bit represents 1 ns and the maximum value will be 0x3B9A_C9FF, after which it rolls-over to zero).	0	RO
31	PSNT	Positive or negative time This bit indicates positive or negative time value. If the bit is reset, it indicates that the time representation is positive, and if it is set, it indicates negative time value. (This bit represents the 32nd bit of the nanoseconds value when the Advance Time Stamp feature is enabled).	0	RO

26.6.20 System time seconds update register

This register contains the lower 32 bits of the seconds field to be written to, added to, or subtracted from the System Time value.

The System Time - Seconds Update register, along with the System Time - Nanoseconds Update register, initialize or update the system time maintained by the core. You must write both of these registers before setting the TSINIT or TSUPDT bits in the Time Stamp Control register.

Table 543. System time seconds update register (SECONDSUPDATE, address 0x4001 0710) bit description

Bit	Symbol	Description	Reset value	Access
31:0	TSS	Time stamp second The value in this field indicates the time, in seconds, to be initialized or added to the system time.	0	R/W

26.6.21 System time nanoseconds update register

This register contains 32 bits of the nano-seconds field to be written to, added to, or subtracted from the System Time value.

Table 544. System time nanoseconds update register (NANOSECONDSUPDATE, address 0x4001 0714) bit description

Bit	Symbol	Description	Reset value	Access
30:0	TSSS	Time stamp sub seconds The value in this field has the sub second representation of time, with an accuracy of 0.46 nano-second. (When TSCTRLSSR is set in the time stamp control register, each bit represents 1 ns and the programmed value should not exceed 0x3B9A_C9FF.)	0	R/W
31	ADDSUB	Add or subtract time When this bit is set, the time value is subtracted with the contents of the update register. When this bit is reset, the time value is added with the contents of the update register.	0	R/W

26.6.22 Time stamp addend register

This register is used by the software to readjust the clock frequency linearly to match the master clock frequency.

This register value is used only when the system time is configured for Fine Update mode (TSCFUPDT bit in Table 538). This register content is added to a 32-bit accumulator in every clock cycle and the system time is updated whenever the accumulator overflows.

Table 545. Time stamp addend register (ADDEND, address 0x4001 0718) bit description

Bit	Symbol	Description	Reset value	Access
31:0	TSAR	Time stamp addend This register indicates the 32-bit time value to be added to the Accumulator register to achieve time synchronization.	0	R/W

26.6.23 Target time seconds register

This register contains the higher 32 bits of time to be compared with the system time for interrupt event generation.

The Target Time Seconds register, along with Target Time Nanoseconds register, are used to schedule an interrupt event (TSTARGT bit in Table 549 when Advanced Timestamping is enabled, or otherwise, TS interrupt bit in Table 534) when the system time exceeds the value programmed in these registers.

Table 546. Target time seconds register (TARGETSECONDS, address 0x4001 071C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	TSTR	Target time seconds register This register stores the time in seconds. When the time stamp value matches or exceeds both Target Time Stamp registers, the MAC, if enabled, generates an interrupt.	0	R/W

26.6.24 Target time nanoseconds register

This register contains the higher 32 bits of time to be compared with the system time for interrupt event generation.

Table 547. Target time nanoseconds register (TARGETNANOSECONDS, address 0x4001 0720) bit description

Bit	Symbol	Description	Reset value	Access
30:0	TSTR	Target time stamp low This register stores the time in (signed) nanoseconds. When the value of the Time Stamp matches the Target time stamp registers (both), the MAC will generate an interrupt if enabled. (This value should not exceed 0x3B9A_C9FF when TSCTRLSSR is set in the time stamp control register.)	0	RO
31	-	Reserved.	-	-

26.6.25 System time higher words seconds register

This register contains the most significant 16-bits of the time stamp seconds value.

Table 548. System time higher words seconds register (HIGHWORD, address 0x4001 0724) bit description

Bit	Symbol	Description	Reset value	Access
15:0	TSHWR	Time stamp higher word Contains the most significant 16-bits of the time stamp seconds value. The register is directly written to initialize the value. This register is incremented when there is an overflow from the 32-bits of the System Time - Seconds register.	0	R/W
31:16	-	Reserved.	-	-

26.6.26 Time stamp status register

This register contains the PTP status. All bits except Bits[27:25] gets cleared after this register is read by the host.

The register field can be read by the application (Read), can be set to 1 by the core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field.

Table 549. Time stamp status register (TIMESTAMPSTAT, address 0x4001 0728) bit description

Bit	Symbol	Description	Reset value	Access
0	TSSOVF	Time stamp seconds overflow When set, indicates that the seconds value of the time stamp (when supporting version 2 format) has overflowed beyond 0xFFFF_FFFF.	0	R/W
1	TSTARGT	Time stamp target reached When set, indicates the value of system time is greater or equal to the value specified in the Target Time High and Low registers	0	R/W
2	AUXSS	Auxiliary Time Stamp Trigger Snapshot This bit is set high when the auxiliary snapshot is written to the FIFO.	0	R/W
23:3	-	Reserved.	-	-
24	ATSSTM	Auxiliary Time Stamp Snapshot Trigger Missed This bit is set when the Auxiliary time stamp snapshot FIFO is full and external trigger was set. This indicates that the latest snapshot was not stored in the FIFO.	0	R/W
27:25	ATSNS	Auxiliary Time Stamp Number of Snapshots These bits indicate the number of Snapshots available in the FIFO. A value of 4 (100) indicates that the Auxiliary Snapshot FIFO is full. These bits are cleared (to 000) when the Auxiliary snapshot FIFO clear bit is set.	0	R/W
31:28	-	Reserved.	-	-

26.6.27 PPS control register

This register is used to control the interval of the PPS signal output, such that a duration of less than 1 second can be achieved between pulses.

Table 550. PPS control register (PPSCTRL, address 0x4001 072C) bit description

Bit	Symbol	Description	Reset value	Access
3:0	PPSCTRL	Control the duration between 2-pulses of PPS signal output This controls the duration between 2 pulses of the PPS output signal. By default the control is set to 1 second between pulses. The duration of the pulses for valid settings is mentioned below. 0000 = 1.0 sec (Binary & Digital Rollover) 0001 = 0.5 sec (Binary Rollover), 0.536 sec (Digital Rollover) 0010 = 0.25 sec (Binary Rollover), 0.26 sec (Digital Rollover) 0011 = 0.125 sec (Binary Rollover), 0.13 sec (Digital Rollover) ... 1111 = 15.28 μsec (Binary Rollover), 32.77 μsec (Digital Rollover)	0	R/W
31:4	-	Reserved.	-	-

26.6.28 Auxiliary time stamp nanoseconds register

This register contains the lower 32 bits (nano-seconds field) of the auxiliary time stamp register.

This register along with the Auxiliary time stamp seconds register (Table 552) gives the 64-bit timestamp stored as auxiliary snapshot. The two registers together form the read port of a 4-deep 64-bit wide FIFO. Multiple snapshots can be stored in this FIFO and the fill-level of this FIFO is indicated by ATSNS bits in the Time Stamp Status register. The top of the FIFO is removed only when the last byte of Table 552 is read. In Little-endian mode, this means when Bits[31:24] is read while in Big-endian mode it corresponds to the reading of Bits[7:0] of Table 552.

Table 551. Auxiliary time stamp nanoseconds register (AUXNANOSECONDS, address 0x4001 0730) bit description

Bit	Symbol	Description	Reset value	Access
31:0	AUXNS	Lower 32 bits (nano-seconds field) of the auxiliary time stamp.	0	RO

26.6.29 Auxiliary timestamp seconds register

This register contains the lower 32 bits of the Seconds field of the auxiliary time stamp register.

Table 552. Auxiliary timestamp seconds register (AUXSECONDS, address 0x4001 0734) bit description

Bit	Symbol	Description	Reset value	Access
31:0	AUXS	Lower 32 bits of the Seconds field of the auxiliary time stamp .	0	RO

26.6.30 DMA Bus mode register

The Bus Mode register establishes the bus operating modes for the DMA.

Table 553. DMA Bus mode register (DMA_BUS_MODE, address 0x4001 1000) bit description

Bit	Symbol	Description	Reset value	Access
0	SWR	Software reset This register field can be read by the application (Read), can be set to 1 by the application with a register write of 1 (Write Set), and is cleared to 0 by the Ethernet core (Self Clear). The application cannot clear this type of field, and a register write of 0 to this bit has no effect on this field. When this bit is set, the MAC DMA Controller resets all MAC Subsystem internal registers and logic. It is cleared automatically after the reset operation has completed in all of the core clock domains. Read a 0 value in this bit before re-programming any register of the core. **Remark:** The reset operation is completed only when all the resets in all the active clock domains are de-asserted. Hence it is essential that all the PHY inputs clocks (applicable for the selected PHY interface) are present for software reset completion.	0	R/W
1	DA	DMA arbitration scheme 0 = Round-robin with Rx:Tx priority given in bits [15:14] 1 = Rx has priority over Tx	0	R/W
6:2	DSL	Descriptor skip length This bit specifies the number of Word/Dword/Lword (depending on 32/64/128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of current descriptor to the start of next descriptor. When DSL value equals zero, then the descriptor table is taken as contiguous by the DMA, in Ring mode.	0	R/W
7	ATDS	Alternate descriptor size When set, the alternate descriptor (see Section 26.10.3) size is increased to 32 bytes (8 DWORDS). This is required when the Advanced Time-Stamp feature or Full IPC Offload Engine is enabled in the receiver. When reset, the descriptor size reverts back to 4 DWORDs (16 bytes).	0	R/W
13:8	PBL	Programmable burst length These bits indicate the maximum number of beats to be transferred in one DMA transaction. This will be the maximum value that is used in a single block Read/Write. The DMA will always attempt to burst as specified in PBL each time it starts a Burst transfer on the host bus. PBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value will result in undefined behavior. When USP is set high, this PBL value is applicable for TxDMA transactions only. The PBL values have the following limitations. The maximum number of beats (PBL) possible is limited by the size of the Tx FIFO and Rx FIFO in the MTL layer and the data bus width on the DMA. The FIFO has a constraint that the maximum beat supported is half the depth of the FIFO, except when specified (as given below). For different data bus widths and FIFO sizes, the valid PBL range (including x8 mode) is provided in the following table. If the PBL is common for both transmit and receive DMA, the minimum Rx FIFO and Tx FIFO depths must be considered. Do not program out-of-range PBL values, because the system may not behave properly.	1	R/W

Table 553. DMA Bus mode register (DMA_BUS_MODE, address 0x4001 1000) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
15:14	PR	Rx-to-Tx priority ratio RxDMA requests given priority over TxDMA requests in the following ratio. This is valid only when the DA bit is reset. 00 = 1-to-1 01 = 2-to-1 10 = 3-to-1 11 = 4-to-1	00	R/W
16	FB	Fixed burst This bit controls whether the AHB Master interface performs fixed burst transfers or not. When set, the AHB will use only SINGLE, INCR4, INCR8 or INCR16 during start of normal burst transfers. When reset, the AHB will use SINGLE and INCR burst transfer operations.	0	R/W
22:17	RPBL	RxDMA PBL These bits indicate the maximum number of beats to be transferred in one RxDMA transaction. This will be the maximum value that is used in a single block Read/Write. The RxDMA will always attempt to burst as specified in RPBL each time it starts a Burst transfer on the host bus. RPBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value will result in undefined behavior. These bits are valid and applicable only when USP is set high.	1	R/W
23	USP	Use separate PBL When set high, it configures the RxDMA to use the value configured in bits [22:17] as PBL while the PBL value in bits [13:8] is applicable to TxDMA operations only. When reset to low, the PBL value in bits [13:8] is applicable for both DMA engines.	0	R/W
24	PBL8X	8 x PBL mode When set high, this bit multiplies the PBL value programmed (bits [22:17] and bits [13:8]) eight times. Thus the DMA will transfer data in to a maximum of 8, 16, 32, 64, 128, and 256 beats depending on the PBL value. **Remark:** This bit function is not backward compatible. Before version 3.50a, this bit was 4xPBL.	0	R/W
25	AAL	Address-aligned beats When this bit is set high and the FB bit equals 1, the AHB interface generates all bursts aligned to the start address LS bits. If the FB bit equals 0, the first burst (accessing the data buffer's start address) is not aligned, but subsequent bursts are aligned to the address.	0	R/W
26	MB	Mixed burst When this bit is set high and FB bit is low, the AHB master interface will start all bursts of length more than 16 with INCR (undefined burst) whereas it will revert to fixed burst transfers (INCRx and SINGLE) for burst-length of 16 and below.	0	R/W
27	TXPR	When set, this bit indicates that the transmit DMA has higher priority than the receive DMA during arbitration for the system-side bus.	0	R/W
31:28	-	Reserved	0	RO

Table 554. Programmable burst length settings

Data bus width	FIFO depth	Valid PBL range in full duplex mode
32 bit	128 bytes	8 or less
	256 bytes	32 or less
	512 bytes	64 or less
	1 kB	128 or less
	2 kB and above	all

26.6.31 DMA Transmit poll demand register

The Transmit Poll Demand register enables the Transmit DMA to check whether or not the current descriptor is owned by DMA. The Transmit Poll Demand command is given to wake up the TxDMA if it is in Suspend mode. The TxDMA can go into Suspend mode due to an Underflow error in a transmitted frame or due to the unavailability of descriptors owned by Transmit DMA. You can give this command anytime and the TxDMA will reset this command once it starts re-fetching the current descriptor from host memory.

Table 555. DMA Transmit poll demand register (DMA_TRANS_POLL_DEMAND, address 0x4001 1004) bit description

Bit	Symbol	Description	Reset value	Access
31:0	TPD	Transmit poll demand This register field can be read by the application, and when a write operation is performed with any data value, an event is triggered. When these bits are written with any value, the DMA reads the current descriptor pointed to by the Current Host Transmit Descriptor register (Section 26.6.40). If that descriptor is not available (owned by Host), transmission returns to the Suspend state and bit 2 in the DMA_STAT Register is asserted. If the descriptor is available, transmission resumes.	0	R/W

26.6.32 DMA Receive poll demand register

The Receive Poll Demand register enables the receive DMA to check for new descriptors. This command is given to wake up the RxDMA from SUSPEND state. The RxDMA can go into SUSPEND state only due to the unavailability of descriptors owned by it.

Table 556. DMA Receive poll demand register (DMA_REC_POLL_DEMAND, address 0x4001 1008) bit description

Bit	Symbol	Description	Reset value	Access
31:0	RPD	Receive poll demand This register field can be read by the application, and when a write operation is performed with any data value, an event is triggered. When these bits are written with any value, the DMA reads the current descriptor pointed to by the Current Host Receive Descriptor register (Section 26.6.41). If that descriptor is not available (owned by Host), reception returns to the Suspended state and bit 7 in the DMA_STAT Register is not asserted. If the descriptor is available, the Receive DMA returns to active state.	0	R/W

26.6.33 DMA Receive descriptor list address register

The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word/Dword/Lword-aligned (for 32/64/128- bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to this register is permitted only when reception is stopped. When stopped, this register must be written to before the receive Start command is given.

Table 557. DMA Receive descriptor list address register (DMA_REC_DES_ADDR, address 0x4001 100C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	SRL	Start of receive list This field contains the base address of the First Descriptor in the Receive Descriptor list. The LSB bits [1/2/3:0] for 32/64/128-bit bus width) will be ignored and taken as all-zero by the DMA internally. Hence these LSB bits are Read Only.	0	R/W

26.6.34 DMA Transmit descriptor list address register

The Transmit Descriptor List Address register points to the start of the Transmit Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word/DWORD/LWORD-aligned (for 32/64/128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. Writing to this register is permitted only when transmission has stopped. When stopped, this register can be written before the transmission Start command is given.

Table 558. DMA Transmit descriptor list address register (DMA_TRANS_DES_ADDR, address 0x4001 1010) bit description

Bit	Symbol	Description	Reset value	Access
31:0	SRL	Start of transmit list This field contains the base address of the First Descriptor in the Transmit Descriptor list. The LSB bits [1/2/3:0] for 32/64/128-bit bus width) will be ignored and taken as all-zero by the DMA internally. Hence these LSB bits are Read Only.	0	R/W

26.6.35 DMA Status register

The Status register contains all the status bits that the DMA reports to the host. This register is usually read by the Software driver during an interrupt service routine or polling. Most of the fields in this register cause the host to be interrupted. The bits in this register are not cleared when read. Writing 1 to (unreserved) bits in this register (bits [16:0]) clears them and writing 0 has no effect. Each field (bits[16:0]) can be masked by masking the appropriate bit in the DMA_INT_EN register.

This fields in this register can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and can be cleared to 0 by the application with a register write of 1 (Write Clear). A register write of 0 has no effect on this field.

Table 559. DMA Status register (DMA_STAT, address 0x4001 1014) bit description

Bit	Symbol	Description	Reset value	Access
0	TI	Transmit interrupt This bit indicates that frame transmission is finished and TDES1[31] is set in the First Descriptor.	0	R/W
1	TPS	Transmit process stopped This bit is set when the transmission is stopped.	0	RW
2	TU	Transmit buffer unavailable This bit indicates that the Next Descriptor in the Transmit List is owned by the host and cannot be acquired by the DMA. Transmission is suspended. Bits[22:20] explain the Transmit Process state transitions. To resume processing transmit descriptors, the host should change the ownership of the bit of the descriptor and then issue a Transmit Poll Demand command.	0	R/W
3	TJT	Transmit jabber timeout This bit indicates that the Transmit Jabber Timer expired, meaning that the transmitter had been excessively active. The transmission process is aborted and placed in the Stopped state. This causes the Transmit Jabber Timeout TDES0[14] flag to assert.	0	R/W
4	OVF	Receive overflow This bit indicates that the Receive Buffer had an Overflow during frame reception. If the partial frame is transferred to application, the overflow status is set in RDES0[11].	0	R/W
5	UNF	Transmit underflow This bit indicates that the Transmit Buffer had an Underflow during frame transmission. Transmission is suspended and an Underflow Error TDES0[1] is set.	0	R/W

Table 559. DMA Status register (DMA_STAT, address 0x4001 1014) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
6	RI	Receive interrupt This bit indicates the completion of frame reception. Specific frame status information has been posted in the descriptor. Reception remains in the Running state.	0	R/W
7	RU	Receive buffer unavailable This bit indicates that the Next Descriptor in the Receive List is owned by the host and cannot be acquired by the DMA. Receive Process is suspended. To resume processing Receive descriptors, the host should change the ownership of the descriptor and issue a Receive Poll Demand command. If no Receive Poll Demand is issued, Receive Process resumes when the next recognized incoming frame is received. This bit is set only when the previous Receive Descriptor was owned by the DMA.	0	R/W
8	RPS	Received process stopped This bit is asserted when the Receive Process enters the Stopped state.	0	R/W
9	RWT	Receive watchdog timeout This bit is asserted when a frame with a length greater than 2,048 bytes is received (10,240 when Jumbo Frame mode is enabled).	0	R/W
10	ETI	Early transmit interrupt This bit indicates that the frame to be transmitted was fully transferred to the MTL Transmit FIFO.	0	R/W
12:11	-	Reserved	0	RO
13	FBI	Fatal bus error interrupt This bit indicates that a bus error occurred, as detailed in bits [25:23]. When this bit is set, the corresponding DMA engine disables all its bus accesses.	0	R/W
14	ERI	Early receive interrupt This bit indicates that the DMA had filled the first data buffer of the packet. Receive Interrupt bit 6 in this register automatically clears this bit.	0	R/W

Table 559. DMA Status register (DMA_STAT, address 0x4001 1014) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
15	AIE	Abnormal interrupt summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_INT_EN register: DMA_STAT register, bit 1: Transmit process stopped DMA_STAT register, bit 3: Transmit jabber timeout DMA_STAT register, bit 4: Receive overflow DMA_STAT register, bit 5: Transmit underflow DMA_STAT register, bit 7: Receiver buffer unavailable DMA_STAT register, bit 8: Receive process stopped DMA_STAT register, bit 9: Receive watchdog timeout DMA_STAT register, bit 10: Early transmit interrupt DMA_STAT register, bit 13: Fatal bus error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit and must be cleared each time a corresponding bit that causes AIS to be set is cleared.	0	R/W
16	NIS	Normal interrupt summary Normal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in the DMA_INT_EN register: DMA_STAT register, bit 0: Transmit interrupt DMA_STAT register, bit 2: Transmit buffer unavailable DMA_STAT register, bit 6: Receive interrupt DMA_STAT register, bit 14: Early receive interrupt Only unmasked bits affect the Normal Interrupt Summary bit. This is a sticky bit and must be cleared (by writing a 1 to this bit) each time a corresponding bit that causes NIS to be set is cleared.	0	R/W
31:17	-	Reserved	0	RO

26.6.36 DMA Operation mode register

The Operation Mode register establishes the Transmit and Receive operating modes and commands. This register should be the last CSR to be written as part of DMA initialization.

Table 560. DMA operation mode register (DMA_OP_MODE, address 0x4001 1018) bit description

Bit	Symbol	Description	Reset value	Access
0	-	Reserved	0	RO
1	SR	Start/stop receive When this bit is set, the Receive process is placed in the Running state. The DMA attempts to acquire the descriptor from the Receive list and processes incoming frames. Descriptor acquisition is attempted from the current position in the list, which is the address set by the DMA_REC_DES_ADDR register or the position retained when the Receive process was previously stopped. If no descriptor is owned by the DMA, reception is suspended and Receive Buffer Unavailable bit (bit 7 in DMA_STAT register) is set. The Start Receive command is effective only when reception has stopped. If the command was issued before setting the DMA_REC_DES_ADDR, DMA behavior is unpredictable.	0	R/W
2	OSF	Operate on second frame When this bit is set, this bit instructs the DMA to process a second frame of Transmit data even before status for first frame is obtained.	0	R/W
4:3	RTC	Receive threshold control These two bits control the threshold level of the MTL Receive FIFO. Transfer (request) to DMA starts when the frame size within the MTL Receive FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are transferred automatically. Note that value of 11 is not applicable if the configured Receive FIFO size is 128 bytes. These bits are valid only when the RSF bit is zero, and are ignored when the RSF bit is set to 1. 00 = 64 01 = 32 10 = 96 11 = 128	0	R/W
5	-	Reserved	0	RO
6	FUF	Forward undersized good frames When set, the Rx FIFO will forward Undersized frames (frames with no Error and length less than 64 bytes) including pad-bytes and CRC). When reset, the Rx FIFO will drop all frames of less than 64 bytes, unless it is already transferred due to lower value of Receive Threshold (e.g., RTC = 01).	0	R/W
7	FEF	Forward error frames When this bit is reset, the Rx FIFO drops frames with error status (CRC error, collision error, GMII_ER, giant frame, watchdog timeout, overflow). However, if the frame's start byte (write) pointer is already transferred to the read controller side (in Threshold mode), then the frames are not dropped. When FEF is set, all frames except runt error frames are forwarded to the DMA. But when RxFIFO overflows when a partial frame is written, then such frames are dropped even when FEF is set.	0	R/W
12:8	-	Reserved	0	RO

Table 560. DMA operation mode register (DMA_OP_MODE, address 0x4001 1018) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
13	ST	Start/Stop Transmission Command When this bit is set, transmission is placed in the Running state, and the DMA checks the Transmit List at the current position for a frame to be transmitted. Descriptor acquisition is attempted either from the current position in the list, which is the Transmit List Base Address set by the DMA_TRANS_DES_ADDR register or from the position retained when transmission was stopped previously. If the current descriptor is not owned by the DMA, transmission enters the Suspended state and Transmit Buffer Unavailable (DMA_STAT register, bit 2) is set. The Start Transmission command is effective only when transmission is stopped. If the command is issued before setting the DMA_TRANS_DES_ADDR register, then the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current frame. The Next Descriptor position in the Transmit List is saved, and becomes the current position when transmission is restarted. The stop transmission command is effective only the transmission of the current frame is complete or when the transmission is in the Suspended state.	0	R/W
16:14	TTC	Transmit threshold control These three bits control the threshold level of the MTL Transmit FIFO. Transmission starts when the frame size within the MTL Transmit FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are also transmitted. These bits are used only when the TSF bit (Bit 21) is reset. 000 = 64 001 = 128 010 = 192 011 = 256 100 = 40 101 = 32 110 = 24 111 = 16	0	R/W
19:17	-	Reserved	0	RO
20	FTF	Flush transmit FIFO This register field can be read by the application (Read), can be set to 1 by the application with a register write of 1 (Write Set), and is cleared to 0 by the Ethernet core (Self Clear). The application cannot clear this type of field, and a register write of 0 to this bit has no effect on this field. When this bit is set, the transmit FIFO controller logic is reset to its default values and thus all data in the Tx FIFO is lost/flushed. This bit is cleared internally when the flushing operation is completed fully. The Operation Mode register should not be written to until this bit is cleared. The data which is already accepted by the MAC transmitter will not be flushed. It will be scheduled for transmission and will result in underflow and runt frame transmission. **Remark:** The flush operation completes only after emptying the TxFIFO of its contents and all the pending Transmit Status of the transmitted frames are accepted by the host. In order to complete this flush operation, the PHY transmit clock is required to be active.	0	R/W

Table 560. DMA operation mode register (DMA_OP_MODE, address 0x4001 1018) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
21	TSF	Transmit store and forward When this bit is set, transmission starts when a full frame resides in the MTL Transmit FIFO. When this bit is set, the TTC values specified in this register (bits [16:14]) are ignored. This bit should be changed only when transmission is stopped.	0	R/W
23:22	-	Reserved	0	RO
24	DFF	Disable flushing of received frames When this bit is set, the RxDMA does not flush any frames due to the unavailability of receive descriptors/buffers as it does normally when this bit is reset. (See).	0	R/W
25	RSF	Receive store and forward When this bit is set, the MTL only reads a frame from the Rx FIFO after the complete frame has been written to it, ignoring RTC bits. When this bit is reset, the Rx FIFO operates in Cut-Through mode, subject to the threshold specified by the RTC bits.	0	R/W
26	DT	Disable Dropping of TCP/IP Checksum Error Frames When this bit is set, the core does not drop frames that only have errors detected by the Receive Checksum Offload engine. Such frames do not have any errors (including FCS error) in the Ethernet frame received by the MAC but have errors in the encapsulated payload only. When this bit is reset, all error frames are dropped if the FEF bit is reset.	0	R/W
31:27	-	Reserved	0	RO

26.6.37 DMA Interrupt enable register

The Interrupt Enable register enables the interrupts reported by the DMA_STAT register. Setting a bit to 1 enables a corresponding interrupt. After a hardware or software reset, all interrupts are disabled.

Table 561. DMA Interrupt enable register (DMA_INT_EN, address 0x4001 101C) bit description

Bit	Symbol	Description	Reset value	Access
0	TIE	Transmit interrupt enable When this bit is set with Normal Interrupt Summary Enable (bit 16 in this register), Transmit Interrupt is enabled. When this bit is reset, Transmit Interrupt is disabled.	0	R/W
1	TSE	Transmit stopped enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), Transmission Stopped Interrupt is enabled. When this bit is reset, Transmission Stopped Interrupt is disabled.	0	R/W
2	TUE	Transmit buffer unavailable enable When this bit is set with Normal Interrupt Summary Enable (bit 16 in this register), Transmit Buffer Unavailable Interrupt is enabled. When this bit is reset, Transmit Buffer Unavailable Interrupt is disabled.	0	R/W
3	TJE	Transmit jabber timeout enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), Transmit Jabber Timeout Interrupt is enabled. When this bit is reset, Transmit Jabber Timeout Interrupt is disabled.	0	R/W

Table 561. DMA Interrupt enable register (DMA_INT_EN, address 0x4001 101C) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
4	OVE	Overflow interrupt enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), Receive Overflow Interrupt is enabled. When this bit is reset, Overflow Interrupt is disabled.	0	R/W
5	UNE	Underflow interrupt enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), Transmit Underflow Interrupt is enabled. When this bit is reset, Underflow Interrupt is disabled.	0	R/W
6	RIE	Receive interrupt enable When this bit is set with Normal Interrupt Summary Enable (bit 16 in this register), Receive Interrupt is enabled. When this bit is reset, Receive Interrupt is disabled.	0	R/W
7	RUE	Receive buffer unavailable enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), Receive Buffer Unavailable Interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable Interrupt is disabled.	0	R/W
8	RSE	Received stopped enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), Receive Stopped Interrupt is enabled. When this bit is reset, Receive Stopped Interrupt is disabled.	0	R/W
9	RWE	Receive watchdog timeout enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), the Receive Watchdog Timeout Interrupt is enabled. When this bit is reset, Receive Watchdog Timeout Interrupt is disabled.	0	R/W
10	ETE	Early transmit interrupt enable When this bit is set with an Abnormal Interrupt Summary Enable (bit 15 in this register), Early Transmit Interrupt is enabled. When this bit is reset, Early Transmit Interrupt is disabled.	0	R/W
12:11	-	Reserved	0	RO
13	FBE	Fatal bus error enable When this bit is set with Abnormal Interrupt Summary Enable (bit 15 in this register), the Fatal Bus Error Interrupt is enabled. When this bit is reset, Fatal Bus Error Enable Interrupt is disabled.	0	R/W
14	ERE	Early receive interrupt enable When this bit is set with Normal Interrupt Summary Enable (bit 16 in this register), Early Receive Interrupt is enabled. When this bit is reset, Early Receive Interrupt is disabled.	0	R/W

Table 561. DMA Interrupt enable register (DMA_INT_EN, address 0x4001 101C) bit description *…continued*

Bit	Symbol	Description	Reset value	Access
15	AIE	Abnormal interrupt summary enable When this bit is set, an Abnormal Interrupt is enabled. When this bit is reset, an Abnormal Interrupt is disabled. This bit enables the following bits DMA_STAT register, bit 1: Transmit process stopped DMA_STAT register, bit 3: Transmit jabber timeout DMA_STAT register, bit 4: Receive overflow DMA_STAT register, bit 5: Transmit underflow DMA_STAT register, bit 7: Receiver buffer unavailable DMA_STAT register, bit 8: Receive process stopped DMA_STAT register, bit 9: Receive watchdog timeout DMA_STAT register, bit 10: Early transmit interrupt DMA_STAT register, bit 13: Fatal bus error	0	R/W
16	NIE	Normal interrupt summary enable When this bit is set, a normal interrupt is enabled. When this bit is reset, a normal interrupt is disabled. This bit enables the following bits: DMA_STAT register, bit 0: Transmit interrupt DMA_STAT register, bit 2: Transmit buffer unavailable DMA_STAT register, bit 6: Receive interrupt DMA_STAT register, bit 14: Early receive interrupt	0	R/W
31:17	-	Reserved	0	RO

The interrupt (sbd_intr_o_interrupt) is generated as shown in Figure 61. It is asserted when the NIS/AIS Status bit is asserted and the corresponding Interrupt Enable bits (NIE/AIE) are enabled.

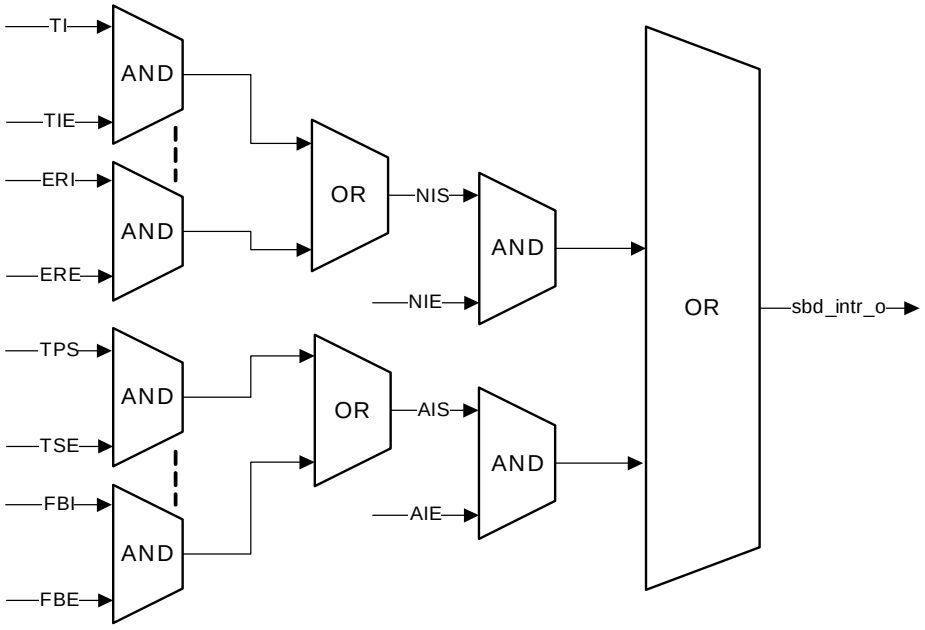


Fig 61. Interrupt generation

26.6.38 DMA Missed frame and buffer overflow counter register

The DMA maintains two counters to track the number of missed frames during reception. This register reports the current value of the counter. The counter is used for diagnostic purposes. Bits[15:0] indicate missed frames due to the host buffer being unavailable. Bits[27:17] indicate missed frames due to buffer overflow conditions (MTL and MAC) and runt frames (good frames of less than 64 bytes) dropped by the MTL.

Table 562. DMA Missed frame and buffer overflow counter register (DMA_MFRM_BUFOF, address 0x4001 1020) bit description

Bit	Symbol	Description	Reset value	Access
15:0	FMC	Number of frames missed This register field can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field. Indicates the number of frames missed by the controller due to the Host Receive Buffer being unavailable. This counter is incremented each time the DMA discards an incoming frame. The counter is cleared when this register is read with.	0	RO
16	OC	Overflow bit for missed frame counter This register field can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field.	0	RO
27:17	FMA	Number of frames missed by the application This register field can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field. Indicates the number of frames missed by the application. This counter is incremented each time the MTL asserts the sideband signal mtl_rxoverflow_o. The counter is cleared when this register is read with <tbd>.	0	RO
28	OF	Overflow bit for FIFO overflow counter This register field can be read by the application (Read), can be set to 1 by the Ethernet core on a certain internal event (Self Set), and is automatically cleared to 0 on a register read. A register write of 0 has no effect on this field.	0	RO
31:29	-	Reserved	0	RO

26.6.39 DMA Receive interrupt watchdog timer register

This register, when written with non-zero value, will enable the watchdog timer for RI (bit 6 in the DMA_STAT register).

Table 563. DMA Receive interrupt watchdog timer register (DMA_REC_INT_WDT, address 0x4001 1024) bit description

Bit	Symbol	Description	Reset value	Access
7:0	RIWT	RI watchdog timeout Indicates the number of system clock cycles multiplied by 256 for which the watchdog timer is set. The watchdog timer gets triggered with the programmed value after the RxDMA completes the transfer of a frame for which the RI status bit is not set due to the setting in the corresponding descriptor RDES1[31]. When the watch-dog timer runs out, the RI bit is set and the timer is stopped. The watchdog timer is reset when RI bit is set high due to automatic setting of RI as per RDES1[31] of any received frame.	0	R/W
31:8	-	Reserved	0	RO

26.6.40 DMA Current host transmit descriptor register

The Current Host Transmit Descriptor register points to the start address of the current Transmit Descriptor read by the DMA.

Table 564. DMA Current host transmit descriptor register (DMA_CURHOST_TRANS_DES, address 0x4001 1048) bit description

Bit	Symbol	Description	Reset value	Access
31:0	HTD	Host Transmit Descriptor Address Pointer Cleared on Reset. Pointer updated by DMA during operation.	0	RO

26.6.41 DMA Current host receive descriptor register

The Current Host Receive Descriptor register points to the start address of the current Receive Descriptor read by the DMA.

Table 565. DMA Current host receive descriptor register (DMA_CURHOST_REC_DES, address 0x4001 104C) bit description

Bit	Symbol	Description	Reset value	Access
31:0	HRD	Host Receive Descriptor Address Pointer Cleared on Reset. Pointer updated by DMA during operation.	0	RO

26.6.42 DMA Current host transmit buffer address register

The Current Host Transmit Buffer Address register points to the current Transmit Buffer Address being read by the DMA.

Table 566. DMA Current host transmit buffer address register (DMA_CURHOST_TRANS_BUF, address 0x4001 1050) bit description

Bit	Symbol	Description	Reset value	Access
31:0	HTB	Host Transmit Buffer Address Pointer Cleared on Reset. Pointer updated by DMA during operation.	0	RO

26.6.43 DMA Current host receive buffer address register

The Current Host Receive Buffer Address register points to the current Receive Buffer address being read by the DMA.

Table 567. DMA Current host receive buffer address register (DMA_CURHOST_REC_BUF, address 0x4001 1054) bit description

Bit	Symbol	Description	Reset value	Access
31:0	HRB	Host Receive Buffer Address Pointer Cleared on Reset. Pointer updated by DMA during operation.	0	RO

26.7 Functional description

26.7.1 Power management block

This section describes the power management (PMT) mechanisms supported by the MAC. PMT supports the reception of network (remote) wake-up frames and Magic Packet frames. PMT does not perform the clock gate function, but generates interrupts for wake-up frames and Magic Packets received by the MAC. The PMT block sits on the receiver path of the MAC and is enabled with remote wake-up frame enable and Magic Packet enable. These enables are in the PMT Control and Status register and are programmed by the Application.

When the power-down mode is enabled in the PMT, then all received frames are dropped by the core and they are not forwarded to the application. The core comes out of the power down mode only when either a Magic Packet or a Remote Wake-up frame is received and the corresponding detection is enabled.

26.7.1.1 Remote wake-up frame registers

The register wkupfmfilter_reg, address (0x028), loads the Wake-up Frame Filter register. To load values in a Wake-up Frame Filter register, the entire register (WKUPFMFILTER_REG) must be written. The WKUPFMFILTER_REG register is loaded by sequentially loading the eight register values in address (0x028) for WKUPFMFILTER_REG0, WKUPFMFILTER_REG1,... WKUPFMFILTER_REG7, respectively. WKUPFMFILTER_REG is read in the same way.

Remark: The internal counter to access the appropriate WKUPFMFILTER_REG is incremented when lane 3 (or lane 0 in big-endian) is accessed by the CPU. This should be kept in mind if you are accessing these registers in byte or half-word mode.

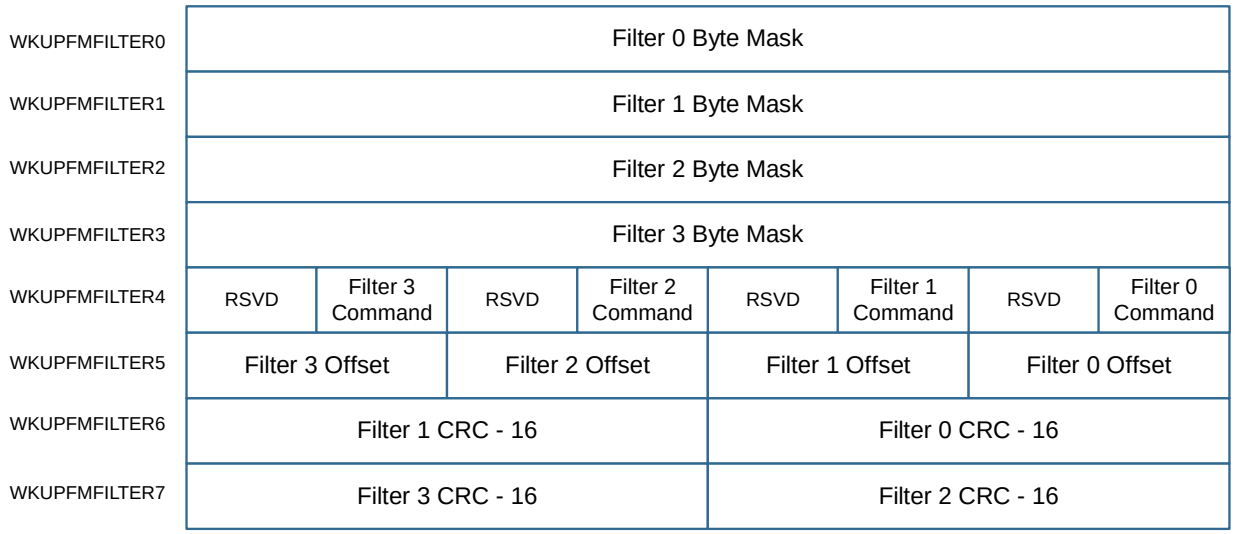


Fig 62. Wake-up frame filter register

Filter i byte mask

This register defines which bytes of the frame are examined by filter i (0, 1, 2, and 3) in order to determine whether or not the frame is a wake-up frame. The MSB (thirty-first bit) must be zero. Bit j [30:0] is the Byte Mask. If bit j (byte number) of the Byte Mask is set, then Filter i Offset + j of the incoming frame is processed by the CRC block; otherwise Filter i Offset + j is ignored.

Filter i command

This 4-bit command controls the filter i operation. Bit 3 specifies the address type, defining the pattern's destination address type. When the bit is set, the pattern applies to only multicast frames; when the bit is reset, the pattern applies only to unicast frame. Bit 2 and Bit 1 are reserved. Bit 0 is the enable for filter i; if Bit 0 is not set, filter i is disabled.

Filter i offset

This register defines the offset (within the frame) from which filter i examines the frames. This 8-bit pattern offset is the offset for the filter i first byte to be examined. The minimum allowed is 12, which refers to the 13th byte of the frame. The offset value 0 refers to the first byte of the frame.

Filter i CRC-16

This register contains the CRC_16 value calculated from the pattern, as well as the byte mask programmed to the wake-up filter register block.

26.7.1.2 Remote wake-up detection

When the MAC is in sleep mode and the remote wake-up bit is enabled in PMT Control and Status register (0x002C), normal operation is resumed after receiving a remote wake-up frame. The Application writes all eight wake-up filter registers by performing a sequential Write to address (0x0028). The Application enables remote wake-up by writing a 1 to Bit 2 of the PMT Control and Status register.

PMT supports four programmable filters that allow support of different receive frame patterns. If the incoming frame passes the address filtering of Filter Command, and if Filter CRC-16 matches the incoming examined pattern, then the wake-up frame is received.

Filter_offset (minimum value 12, which refers to the 13th byte of the frame) determines the offset from which the frame is to be examined. Filter Byte Mask determines which bytes of the frame must be examined. The thirty-first bit of Byte Mask must be set to zero.

The remote wake-up CRC block determines the CRC value that is compared with Filter CRC-16. The wake-up frame is checked only for length error, FCS error, dribble bit error, MII error, collision, and to ensure that it is not a runt frame. Even if the wake-up frame is more than 512 bytes long, if the frame has a valid CRC value, it is considered valid. Wake-up frame detection is updated in the PMT Control and Status register for every remote Wake-up frame received. A PMT interrupt to the Application triggers a Read to the PMT Control and Status register to determine reception of a wake-up frame.

26.7.1.3 Magic packet detection

The Magic Packet frame is based on a method that uses Advanced Micro Device's Magic Packet technology to power up the sleeping device on the network. The MAC receives a specific packet of information, called a Magic Packet, addressed to the node on the network.

Only Magic Packets that are addressed to the device or a broadcast address will be checked to determine whether they meet the wake-up requirements. Magic Packets that pass the address filtering (unicast or broadcast) will be checked to determine whether they meet the remote Wake-on-LAN data format of 6 bytes of all ones followed by a MAC Address appearing 16 times.

The application enables Magic Packet wake-up by writing a 1 to Bit 1 of the PMT Control and Status register. The PMT block constantly monitors each frame addressed to the node for a specific Magic Packet pattern. Each frame received is checked for a 0xFFFF FFFF FFFF pattern following the destination and source address field. The PMT block then checks the frame for 16 repetitions of the MAC address without any breaks or interruptions. In case of a break in the 16 repetitions of the address, the 0xFFFF FFFF FFFF pattern is scanned for again in the incoming frame. The 16 repetitions can be anywhere in the frame, but must be preceded by the synchronization stream (0xFFFF FFFF FFFF). The device will also accept a multicast frame, as long as the 16 duplications of the MAC address are detected.

If the MAC address of a node is 0x0011 2233 4455, then the MAC scans for the data sequence:

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55

00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55 00 11 22 33 44 55

...CRC

Magic Packet detection is updated in the PMT Control and Status register for Magic Packet received. A PMT interrupt to the Application triggers a read to the PMT CSR to determine whether a Magic Packet frame has been received.

26.7.1.4 System considerations during power-down

MAC neither gates nor stops clocks when Power-down mode is enabled. Power saving by clock gating must be done outside the core by the application. The receive data path must be clocked with ENET_RX_CLK during Power-down mode because it is involved in magic packet/wake-on-LAN frame detection. However, the transmit path and the application path clocks can be gated off during Power-down mode.

The PMT interrupt is asserted when a valid wake-up frame is received. This signal is generated in the receive clock domain

The recommended power-down and wake-up sequence is as follows.

1. Disable the Transmit DMA and wait for any previous frame transmissions to complete. These transmissions can be detected when Transmit Interrupt (see DMA_STAT register bit NIS; Table 559) is received.
2. Disable the MAC transmitter and MAC receiver by clearing the appropriate bits in the MAC Configuration register.
3. Wait until the Receive DMA empties all the frames from the Rx FIFO (a software timer may be required).
4. Enable Power-Down mode by appropriately configuring the PMT registers.
5. Enable the MAC Receiver and enter Power-Down mode.
6. Gate the application and transmit clock inputs to the core (and other relevant clocks in the system) to reduce power and enter Sleep mode.
7. On receiving a valid wake-up frame, the MAC PMT interrupt signal and exits Power-Down mode.
8. On receiving the interrupt, the system must enable the application and transmit clock inputs to the core.
9. Read the PMT Status register to clear the interrupt, then enable the other modules in the system and resume normal operation.

Remark:

26.7.2 DMA arbiter functions

If you have enabled the transmit (Tx) DMA and receive (Rx) DMA of a channel, you can specify which DMA gets the bus when the channel gets the control of the bus. You can set the priority between the corresponding Tx DMA and Rx DMA by using the bit 27 (TXPR: Transmit Priority) of the DMA Bus Mode Register). For round-robin arbitration, you can use the bits [15:14] (PR: Priority Ratio) of the Bus Mode Register to specify the weighted priority between the Tx DMA and Rx DMA. Table 568 provides information about the priority scheme between Tx DMA and Rx DMA.

Table 568. Priority scheme for transmit and receive DMA

Bit 27	Bit 15	Bit 14	Bit 1	Priority scheme
0	x	x	x	Rx always has priority over Tx
0	0	0	0	Tx and Rx have equal priority. Rx gets the access first on simultaneous requests.
0	0	1	0	Rx has priority over Tx in the ratio 2:1.
0	1	0	0	Rx has priority over Tx in the ratio 3:1.
0	1	1	0	Rx has priority over Tx in the ratio 4:1.
1	x	x	1	Tx always has priority over Rx.
1	0	0	0	Tx and Rx have equal priority. Tx gets the access first on simultaneous requests.
1	0	1	0	Tx has priority over Rx in the ratio 2:1.
1	1	0	0	Tx has priority over Rx in the ratio 3:1.
1	1	1	0	Tx has priority over Rx in the ratio 4:1.

26.7.3 IPC Receive checksum offload engine

In this mode, both IPv4 and IPv6 frames in the received Ethernet frames are detected and processed for data integrity.You can enable this module by setting the bit 10 (IPC) of the MAC configuration register (Section 26.6.1). The MAC receiver identifies IPv4 or IPv6 frames by checking for value 0x0800 or 0x86DD, respectively, in the received Ethernet frames. Type field. This identification applies to VLAN-tagged frames as well.

The Receive Checksum Offload engine calculates IPv4 header checksums and checks that they match the received IPv4 header checksums. The result of this operation (pass or fail) is given to the RFC module for insertion into the receive status word. The IP Header Error bit is set for any mismatch between the indicated payload type (Ethernet Type field) and the IP header version, or when the received frame does not have enough bytes, as indicated by the IPv4 header.s Length field (or when fewer than 20 bytes are available in an IPv4 or IPv6 header).

This engine also identifies a TCP, UDP, or ICMP payload in the received IP datagrams (IPv4 or IPv6) and calculates the checksum of such payloads properly, as defined in the TCP, UDP, or ICMP specifications. This engine includes the TCP/UDP/ICMPv6 pseudo-header bytes for checksum calculation and checks whether the received checksum field matches the calculated value. The result of this operation is given as a Payload Checksum Error bit in the receive status word. This status bit is also set if the length of the TCP, UDP, or ICMP payload does not match the expected payload length given in the IP header.

This engine bypasses the payload of fragmented IP datagrams, IP datagrams with security features, IPv6 routing headers, and payloads other than TCP, UDP or ICMP.

26.8 IEEE 1588-2002 timestamps

The IEEE 1588-2002 standard defines a protocol, Precision Time Protocol (PTP), that enables precise synchronization of clocks in measurement and control systems implemented with technologies such as network communication, local computing, and distributed objects. The PTP applies to systems communicating by local area networks supporting multicast messaging, including (but not limited to) Ethernet. This protocol

enables heterogeneous systems that include clocks of varying inherent precision, resolution, and stability to synchronize. The protocol supports system-wide synchronization accuracy in the sub-microsecond range with minimal network and local clock computing resources.

The PTP is transported over UDP/IP. The system or network is classified into Master and Slave nodes for distributing the timing/clock information. Figure 63 shows the process that PTP uses for synchronizing a slave node to a master node by exchanging PTP messages.

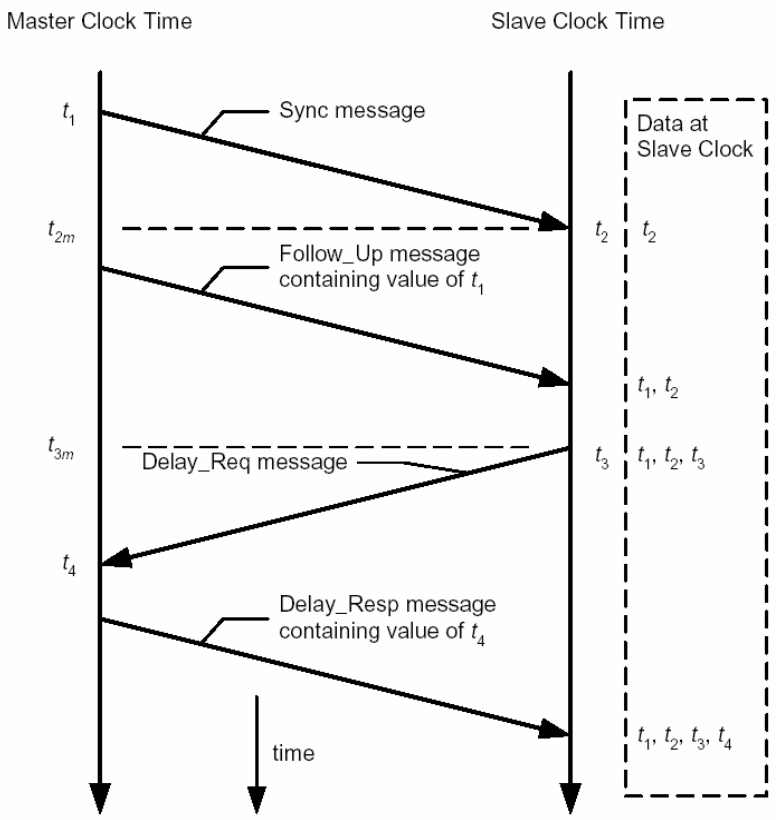


Fig 63. Networked time synchronization

As shown in Figure 63, the PTP uses the following process:

1. The master broadcasts the PTP Sync messages to all its nodes. The Sync message contains the master.s reference time information. The time at which this message leaves the master.s system is t1. This time must be captured, for Ethernet ports, at GMII/MII.
2. The slave receives the Sync message and also captures the exact time, t2, using its timing reference.
3. The master sends a Follow_up message to the slave, which contains t1 information for later use.
4. The slave sends a Delay_Req message to the master, noting the exact time, t3, at which this frame leaves the GMII/MII.

5. The master receives the message, capturing the exact time, t4, at which it enters its system.
6. The master sends the t4 information to the slave in the Delay_Resp message.
7. The slave uses the four values of t1, t2, t3, and t4 to synchronize its local timing reference to the master's timing reference.

Most of the PTP implementation is done in the software above the UDP layer. However, the hardware support is required to capture the exact time when specific PTP packets enter or leave the Ethernet port at the GMII/MII. This timing information must be captured and returned to the software for the proper implementation of PTP with high accuracy.

26.8.1 Reference timing source

To get a snapshot of the time, the MAC requires a reference time in 64-bit format as defined in the IEEE 1588 specification. The Ethernet MAC uses internal timing to provide a reference timing source by using the reference clock input to generate the Reference time (also called the System Time) internally and capture timestamps. The generation, update, and modification of the System Time are described in Section 26.8.2.

26.8.2 System time register module

The 64-bit time is maintained in this module and updated using the input reference clock. This time is the source for taking snapshots (timestamps) of Ethernet frames being transmitted or received at the GMII.

The System Time counter can be initialized or corrected using the coarse correction method. In this method, the initial value or the offset value is written to the Timestamp Update register (Table 538). For initialization, the System Time counter is written with the value in the

Timestamp Update registers, while for system time correction, the offset value is added to or subtracted from the system time.In the fine correction method, a slave clock's frequency drift with respect to the master clock (as defined in IEEE 1588) is corrected over a period of time instead of in one clock, as in coarse correction. This helps maintain linear time and does not introduce drastic changes (or a large jitter) in the reference time between PTP Sync message intervals. In this method, an accumulator sums up the contents of the Addend register, as shown in Figure 4-2. The arithmetic carry that the accumulator generates is used as a pulse to increment the system time counter. The accumulator and the addend are 32-bit registers. Here, the accumulator acts as a high-precision frequency multiplier or divider.

This algorithm is shown in Figure 64:

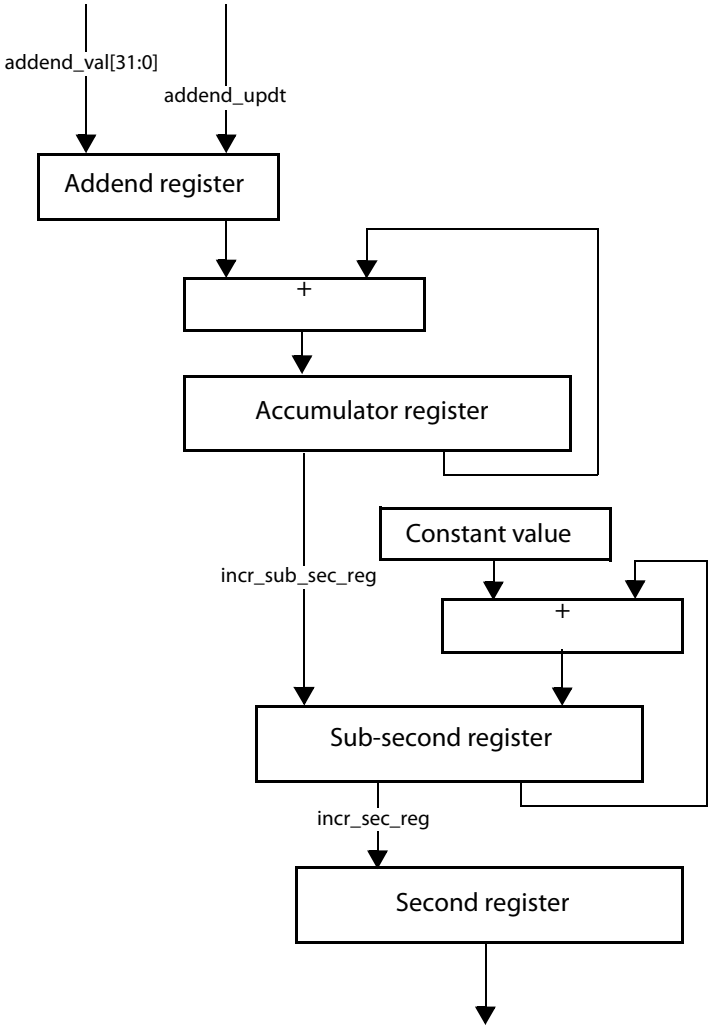


Fig 64. System update using fine method

The System Time Update logic requires a 50-MHz clock frequency to achieve 20-ns accuracy. The frequency division is the ratio of the reference clock frequency to the required clock frequency. Hence, if the reference clock is, for example, 66 MHz, this ratio is calculated as 66 MHz / 50 MHz = 1.32. Hence, the default addend value to be set in the register is 2^{32} / 1.32, 0xC1F07C1F.

If the reference clock drifts lower, to 65 MHz for example, the ratio is 65 / 50, or 1.3 and the value to set in the addend register is 2^{32} / 1.30, or 0xC4EC4EC4. If the clock drifts higher, to 67 MHz for example, the addend register must be set to 0xBF0B7672. When the clock drift is nil, the default addend value of 0xC1F07C1F (2^{32} / 1.32) must be programmed.

In Figure 64, the constant value used to accumulate the sub-second register is decimal 43, which achieves an accuracy of 20 ns in the system time (in other words, it is incremented in 20-ns steps).

The software must calculate the drift in frequency based on the Sync messages and update the Addend register accordingly.

Initially, the slave clock is set with FreqCompensationValue0 in the Addend register. This value is as follows:

FreqCompensationValue0 = 232 / FreqDivisionRatio

If MasterToSlaveDelay is initially assumed to be the same for consecutive Sync messages, the algorithm described below must be applied. After a few Sync cycles, frequency lock occurs. The slave clock can then determine a precise MasterToSlaveDelay value and re-synchronize with the master using the new value.

The algorithm is as follows:

- At time $MasterSyncTime_n$ the master sends the slave clock a Sync message. The slave receives this message when its local clock is $SlaveClockTime_n$ and computes $MasterClockTime_n$ as:

  $MasterClockTime_n = MasterSyncTime_n + MasterToSlaveDelay_n$
- The master clock count for current Sync cycle, $MasterClockCount_n$ is given by:

  $MasterClockCount_n = MasterClockTime_n . MasterClockTime_{n-1}$ (assuming that MasterToSlaveDelay is the same for Sync cycles n and n - 1)
- The slave clock count for current Sync cycle, $SlaveClockCount_n$ is given by:

  $SlaveClockCount_n = SlaveClockTimen_n - SlaveClockTime_{n-1}$
- The difference between master and slave clock counts for current Sync cycle, $ClockDiffCount_n$ is given by:

  $ClockDiffCount_n = MasterClockCount_n - SlaveClockCount_n$
- The frequency-scaling factor for slave clock, $FreqScaleFactor_n$ is given by:

  $FreqScaleFactor_n = (MasterClockCount_n + ClockDiffCount_n) / SlaveClockCount_n$
- The frequency compensation value for Addend register, $FreqCompensationValue_n$ is given by:

  $FreqCompensationValue_n = FreqScaleFactor_n * FreqCompensationValue_{n-1}$

In theory, this algorithm achieves lock in one Sync cycle; however, it may take several cycles, because of changing network propagation delays and operating conditions.

This algorithm is self-correcting: if for any reason the slave clock is initially set to a value from the master that is incorrect, the algorithm corrects it at the cost of more Sync cycles.

26.8.3 Transmit path functions

The MAC captures a timestamp when the Start Frame Delimiter (SFD) of a frame is sent on GMII/MII. The frames for which you want to capture timestamps are controllable on a per-frame basis. In other words, each transmit frame can be marked to indicate whether a timestamp should be captured for that frame.

The MAC does not process the transmitted frames to identify the PTP frames. You need to specify the frames for which you want to capture timestamps.

Use the control bits in the transmit descriptor (see Section 26.10.3.1). The MAC returns the timestamp to the software inside the corresponding transmit descriptor, thus connecting the timestamp automatically to the specific PTP frame. The 64-bit timestamp information is written to the TDES6 and TDES7 fields. The TDES7 field holds the 32 least significant bits of the timestamp.

26.8.4 Receive path functions

The MAC captures the timestamp of all frames received on the GMII/MII. The MAC does not process the received frames to identify the PTP frames in the default mode, that is, when the Advanced Timestamp feature is not selected.

The MAC gives the timestamp and the corresponding status on the MAC Receive Interface (MRI) along with the EOF data. The MTL provides the timestamp on the data bus after the EOF data has been transferred. The MTL sends a separate signal to validate the timestamp and to indicate the availability of the timestamp. Once the timestamp is transferred, the Status Valid signal is asserted as soon as status data is present on the data bus.

The DMA returns the timestamp to the software in the corresponding receive descriptor. The 64-bit timestamp information is written back to the RDES6 and RDES7 fields. The RDES2 holds the 32 least significant bits of the timestamp, except as mentioned in Section 26.10.3.2. The timestamp is written only to that receive descriptor for which the Last Descriptor status field has been set to 1 (the EOF marker). When the timestamp is not available (for example, because of an RxFIFO overflow), an all-ones pattern is written to the descriptors (RDES6 and RDES7), indicating that timestamp is not correct. If the software uses a control register bit to disable time stamping, the DMA does not alter RDES6 or RDES7.

26.8.5 Timestamp error margin

According to the IEEE 1588 specifications, a timestamp must be captured at the SFD of the transmitted and received frames at the GMII/MII interface. Because the reference timing source (the PTP clock) is different from the GMII/MII clocks, a small error margin is introduced, because of the transfer of information across asynchronous clock domains.

In the transmit path, the captured and reported timestamp has a maximum error margin of 2 PTP clocks. It means that the captured timestamp has the reference timing source value that is given within 2 clocks after the SFD has been transmitted on the GMII.

Similarly, in the receive path, the error margin is 3 GMII/MII clocks, plus up to 2 PTP clocks. You can ignore the error margin because of the 3 GMII/MII clocks by assuming that this constant delay is present in the system (or link) before the SFD data reaches the GMAC.s GMII/MII interface.

26.8.6 Frequency range of the reference timing clock

The timestamp information is transferred across asynchronous clock domains, that is, from MAC clock domain to application clock domain. Therefore, a minimum delay is required between two consecutive timestamp captures. This delay is 4 clock cycles of GMII/MII and 3 clock cycles of PTP clocks. If the delay between two timestamp captures is less than this delay, the MAC does not take a timestamp snapshot for the second frame.

The maximum PTP clock frequency is limited by the maximum resolution of the reference time (1 ns resulting in 1 GHz) and the timing constraints achievable for logic operating on the PTP clock. In addition, the resolution, or granularity, of the reference time source determines the accuracy of the synchronization. Therefore, a higher PTP clock frequency gives better system performance.

The minimum PTP clock frequency depends on the time required between two consecutive SFD bytes. Because the GMII/MII clock frequency is fixed by IEEE specification, the minimum PTP clock frequency required for proper operation depends upon the operating mode and operating speed of the MAC as shown in Table 4-1.

Table 569. Minimum PTP clock frequency cycle

Mode	Minimum gap between two SFDs	Minimum PTP frequency
100-Mbps full-duplex operation	168 MII clocks (128 clocks for a 64-byte frame + 24 clocks of min IFG + 16 clocks of preamble)	(3 * PTP) + (4 * MII) ≤ 168 * MII that is, ~0.5 MHz ((168 – 4) * 40 ns ÷ 3 = 2,180 ns period)

26.9 IEEE 1588-2008 advanced timestamps

In addition to the basic timestamp features (see Section 26.8) the ethernet controller supports the following advanced timestamp features defined in the IEEE 1588-2008 standard:

- Supports the IEEE 1588-2008 (version 2) timestamp format.
- Provides an option to take snapshot of all frames or only PTP type frames.
- Provides an option to take snapshot of only event messages.
- Provides an option to take the snapshot based on the clock type: ordinary, boundary, end-to-end, and peer-to-peer.
- Provides an option to select the node to be a Master or Slave for ordinary and boundary clock.
- Identifies the PTP message type, version, and PTP payload in frames sent directly over Ethernet and sends the status.
- Provides an option to measure sub-second time in digital or binary format.

26.9.1 Peer-to-Peer PTP Transparent Clock (P2P TC) Message Support

The IEEE 1588-2008 version supports Peer-to-Peer PTP (Pdelay) message in addition to SYNC, Delay Request, Follow-up, and Delay Response messages. Figure 65 shows the method to calculate the propagation delay in clocks supporting peer-to-peer path correction.

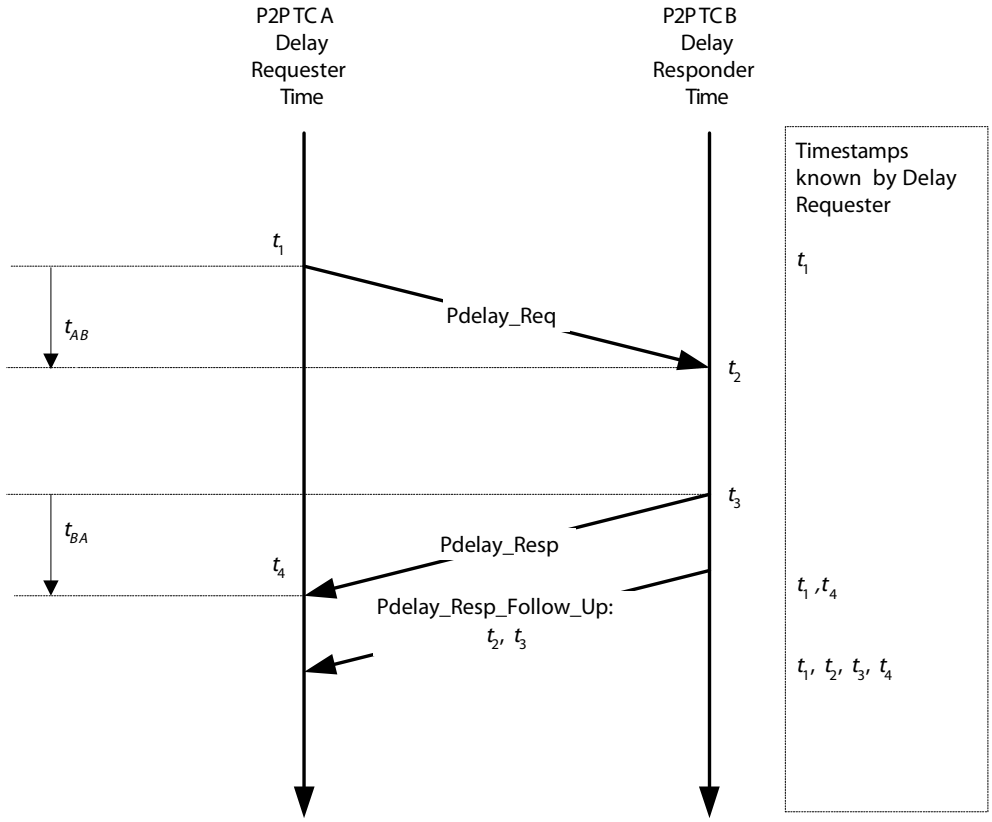


Fig 65. Propagation Delay Calculation in Clocks Supporting Peer-to-Peer Path Correction

As shown in Figure 65, the propagation delay is calculated in the following way:

1. Port-1 issues a Pdelay_Req message and generates a timestamp, t1, for the Pdelay_Req message .
2. Port-2 receives the Pdelay_Req message and generates a timestamp, t2, for this message.
3. Port-2 returns a Pdelay_Resp message and generates a timestamp, t3, for this message.

   To minimize errors because of any frequency offset between the two ports, Port-2 returns the Pdelay_Resp message as quickly as possible after the receipt of the Pdelay_Req message. The Port-2 returns any one of the following:

   - The difference between the timestamps t2 and t3 in the Pdelay_Resp message.
   - The difference between the timestamps t2 and t3 in the Pdelay_Resp_Follow_Up message.
   - The timestamps t2 and t3 in the Pdelay_Resp and Pdelay_Resp_Follow_Up messages respectively.
4. Port-1 generates a timestamp, t4, on receiving the Pdelay_Resp message.
5. Port-1 uses all four timestamps to compute the mean link delay.

26.9.2 Clock types

The Ethernet controller supports the following clock types defined in the IEEE 1588-2008 standard:

- Ordinary clock
- Boundary clock
- End-to-end transparent clock
- Peer-to-peer transparent clock

26.9.2.1 Ordinary clock

The ordinary clock in a domain supports a single copy of the protocol. The ordinary clock has a single PTP state and a single physical port. In typical industrial automation applications, an ordinary clock is associated with an application device such as a sensor or an actuator. In telecommunications applications, the ordinary clock can be associated with a timing demarcation device.

The ordinary clock can be a grandmaster or a slave clock. The ordinary clock supports the following features:

- Send and receives PTP messages. The timestamp snapshot can be controlled as described in Table 538.
- Maintains the data sets such as timestamp values.

Table 570. Ordinary clock: PTP messages for snapshot

Master	Slave
Delay_Req	SYNC

For an ordinary clock, you can take the snapshot of either one of the following PTP message types: version 1 or version 2. You cannot take the snapshots for both PTP message types. You can take the snapshot by setting the control bit (TSVER2ENA) and selecting the snapshot mode in Table 538.

26.9.2.2 Boundary clock

The boundary clock typically has several physical ports communicating with the network. The messages related to synchronization, master-slave hierarchy, and signaling terminate in the protocol engine of the boundary clock and are not forwarded. The PTP message type status given by the core (see .Section 26.8.4) helps you to identify the type of message and take appropriate action.

The boundary clock is similar to the ordinary clock except for the following features:

- The clock data sets are common to all ports of the boundary clock .
- The local clock is common to all ports of the boundary clock.

Therefore, the features of the ordinary clock are also applicable to the boundary clock.

26.9.2.3 End-to-end transparent clock

The end-to-end transparent clock supports the end-to-end delay measurement mechanism between slave clocks and the master clock. The end-to-end transparent clock forwards all messages like normal bridge, router, or repeater. The residence time of a PTP packet is the time taken by the PTP packet from the Ingress port to the Egress port.

The residence time of a SYNC packet inside the end-to-end transparent clock is updated in the correction field of the associated Follow_Up PTP packet before it is transmitted. Similarly, the residence time of a Delay_Req packet inside the end-to-end transparent clock is updated in the correction field of the associated Delay_Resp PTP packet before it is transmitted. Therefore, the snapshot needs to be taken at both Ingress and Egress ports only for the messages mentioned in Table 571.

You can take the snapshot by setting the snapshot select bits (SNAPTYPSEL) to 10 in Table 538.

Table 571. End-to-end transparent clock: PTP messages for snapshot

PTP messages
SYNC
Delay_Req

26.9.2.4 PTP processing and control

Table 4-5 shows the common message header for the PTP messages. This format is taken from IEEE standard 1588-2008 (Revision of IEEE Std. 1588-2002).

Table 572. Message format defined in IEEE 1588-2008

Bits								OCTETS	OFFSET
7	6	5	4	3	2	1	0		
transportSpecific				messageType				1	0
Reserved				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
Reserved								1	5
flagField								2	6
correctionField								8	8
Reserved								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField ([1])								1	32
logMessageInterva								1	33

[1] Field is used in version 1. In version 2, messageType field is used for detecting different message types.

There are some fields in the Ethernet payload that you can use to detect the PTP packet type and control the snapshot to be taken. These fields are different for the following PTP frames:

- PTP Frames Over IPv4
- PTP Frames Over IPv6

- PTP Frames Over Ethernet

26.9.2.4.1 PTP frames over IPv4

Table 4-6 provides information about the fields that are matched to control snapshot for the PTP packets sent over UDP over IPv4 for IEEE 1588 version 1 and 2. The octet positions for the tagged frames are offset by 4. This is based on Annex D of IEEE 1588-2008 standard and the message format defined in Table 571.

Table 573. IPv4-UDP PTP Frame Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Frame Type	12, 13	0x0800	IPv4 datagram
IP version and Header Length	14	0x45	IP version is IPv4
Layer 4 Protocol	23	0x11	UDP
IP Multicast Address (IEEE 1588 version 1)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (or 0x82 or 0x83 or 0x84)	Multicast IPv4 addresses allowed. 224.0.1.129 224.0.1.130 224.0.1.131 224.0.1.132
IP Multicast Address (IEEE 1588 version 2)	30, 31, 32, 33	0xE0, 0x00, 0x01, 0x81 (Hex) 0xE0, 0x00, 0x00, 0x6B (Hex)	PTP-Primary multicast address: 224.0.1.129 PTP-Pdelay multicast address: 224.0.0.107
UDP Destination Port	36, 37	0x013F, 0x0140	0x013F – PTP event message ([1]) 0x0140 – PTP general messages
PTP Control Field (IEEE version 1)	74	0x00/0x01/0x02/ 0x03/0x04	0x00 – SYNC, 0x01 – Delay_Req 0x02 – Follow_Up 0x03 – Delay_Resp 0x04 – Management
PTP Message Type Field (IEEE version 2)	42 (nibble)	0x0/0x1/0x2/0x3/ 0x8/0x9/0xB /0xC/0xD	0x0 – SYNC 0x1 – Delay_Req 0x2 – Pdelay_Req 0x3 – Pdelay_Resp 0x8 – Follow_Up 0x9 – Delay_Resp 0xA – Pdelay_Resp_Follow_Up 0xB – Announce 0xC – Signaling 0xD - Management
PTP Version	43 (nibble)	0x1 or 0x2	0x1 – Supports PTP version 1 0x2 – Supports PTP version 2

[1] PTP event messages are SYNC, Delay_Req (IEEE 1588 version 1 and 2) or Pdelay_Req, Pdelay_Resp (IEEE 1588 version 2 only).

26.9.2.4.2

26.9.2.4.3

26.9.2.4.4 PTP Frames Over IPv6

Table 574 provides information about the fields that are matched to control the snapshots for the PTP packets sent over UDP over IPv6 for IEEE 1588 version 1 and 2. The octet positions for the tagged frames are offset by 4. This is based on Annex D of IEEE 1588-2008 standard and the message format defined in Table 571.

Table 574. IPv6-UDP PTP Frame Fields Required for Control and Status

Field Matched	Octet Position	Matched Value	Description
MAC Frame Type	12, 13	0x86DD	IP datagram
IP version	14(bits [7:4])	0x6	IP version is IPv6
Layer 4 Protocol	20 ([1])	0x11	UDP
PTP Multicast Address	38 – 53	FF0x:0:0:0:0:0:0:181 (Hex) FF02:0:0:0:0:0:0:6B (Hex)	PTP – Primary multicast address: FF0x:0:0:0:0:0:0:0:181 (Hex) PTP – Pdelay multicast address: FF02:0:0:0:0:0:0:0:6B (Hex)
UDP Destination Port	56, 57 ([1])	0x013F, 0x140	0x013F – PTP event message 0x0140 – PTP general messages
PTP Control Field (IEEE version 1)	93 ([1])	0x00/0x01/0x02/0x03/0x04	0x00 – SYNC 0x01 – Delay_Req 0x02 – Follow_Up 0x03 – Delay_Resp 0x04 – Management (version1)
PTP Message Type Field (IEEE version 2)	74 ([1]) (nibble)	0x0/0x1/0x2/0x3/0x8/0x9/0xB/0xC/0xD	0x0 – SYNC 0x1 – Delay_Req 0x2 – Pdelay_Req 0x3 – Pdelay_Resp 0x8 – Follow_Up 0x9 – Delay_Resp 0xA – Pdelay_Resp_Follow_Up 0xB – Announce 0xC – Signaling 0xD - Management
PTP Version	75 (nibble)	0x1 or 0x2	0x1 – Supports PTP version 1 0x2 – Supports PTP version 2

[1] The Extension Header is not defined for PTP packets.

26.9.2.4.5 PTP frames over ethernet

Table 4-8 provides information about the fields that are matched to control the snapshots for the PTP packets sent over Ethernet for IEEE 1588 version 1 and 2. The octet positions for the tagged frames are offset by 4. This is based on Annex D of the IEEE 1588-2008 standard and the message format defined in Table 572.

Table 575. Ethernet PTP Frame Fields Required for Control And Status

Field Matched	Octet Position	Matched Value	Description
MAC Destination Multicast Address[1]	0-5	01-1B-19-00-00-00 01-80-C2-00-00-0E	All PTP messages can use any of the following multicast addresses[2]: 01-1B-19-00-00-00 01-80-C2-00-00-0E[3]
MAC Frame Type	12, 13	0x88F7	PTP Ethernet frame

Table 575. Ethernet PTP Frame Fields Required for Control And Status

Field Matched	Octet Position	Matched Value	Description
PTP Control Field (IEEE version 1)	45	0x00/0x01/0x02/ 0x03/0x04	0x00 – SYNC 0x01 – Delay_Req 0x02 – Follow_Up 0x03 – Delay_Resp 0x04 – Management
PTP Message Type Field (IEEE version 2)	14 (nibble)	0x0/0x1/0x2/0x3/ 0x8/0x9/0xB /0xC/0xD	0x0 – SYNC 0x1 – Delay_Req 0x2 – Pdelay_Req 0x3 – Pdelay_Resp 0x8 – Follow_Up 0x9 – Delay_Resp 0xA – Pdelay_Resp_Follow_Up 0xB – Announce 0xC – Signaling 0xD - Management
PTP Version	14 (nibble)	0x1 or 0x2	0x1 – Supports PTP version 1 0x2 – Supports PTP version 2

[1] The address match of destination addresses (DA) programmed in MAC address 1 to 31 is used if the control bit 18 (TSENMACADDR: Enable MAC address for PTP frame filtering) of the Timestamp Control register is set.

[2] IEEE standard 1588-2008, Annex F

[3] The Ethernet controller does not consider the PTP version 1 messages with Peer delay multicast address (01-80-C2-00-00- 0E) as valid PTP messages.

26.9.3 Reference timing source

The ethernet controller supports the following reference timing sources featured in the IEEE 1588-2008 standard:

- 48-Bit Seconds Field
- Pulse-Per-Second Output
- Auxiliary Snapshots with External Events

26.9.3.1 48-bit seconds field

The ethernet controller supports 80-bit timestamping. The timestamp has the following fields:

- UInteger48 secondsField

  The seconds field is the integer portion of the timestamp in units of seconds and is 48-bits wide. For example, 2.000000001 seconds are represented as secondsField = 0x0000_0000_0002.

- UInteger32 nanosecondsField

  The nanoseconds field is the fractional portion of the timestamp in units of nanoseconds. For example, 2.000000001 nanoseconds are represented as nanoSeconds = 0x0000_0001.

  The nanoseconds field supports the following two modes:

  – Digital rollover mode: In digital rollover mode, the maximum value in the nanoseconds field is 0x3B9A_C9FF, that is, (10e9-1) nanoseconds.

– Binary rollover mode: In binary rollover mode, the nanoseconds field rolls over and increments the seconds field after value 0x7FFF_FFFF. Accuracy is ~0.466 ns per bit.

You can set these modes by using the bit 9 (TSCTRLSSR) of Table 538.

When you select the advanced timestamp feature, the timestamp maintained in the core is still 64-bit wide. The overflow to the upper 16-bits of seconds register happens once in 130 years. You can read the values of the upper 16-bits of the seconds field from the CSR register.

26.9.3.2 Fixed pulse-per-second Output

The Ethernet controller supports the pulse-per-second (PPS) output that is given to indicate 1 second interval (default). You can change the frequency of the PPS output by setting the bits[3:0] in Table 550.

26.9.3.3 Flexible pulse-per-second output

The Ethernet controller also provides the flexibility to program the start or stop time, width, and interval of the pulse generated on the PPS output.

The Ethernet controller provides the following features with the flexible PPS output option:

- Supports programming the start or stop time in terms of advanced system time.
- Supports programming the start point of the single pulse and start and stop points of the pulse train in terms of 64-bit system time. The existing Target Time registers (register 455 and 456) are used to program the start and stop time.
- Supports programming the stop time in advance, that is, you can program the stop time before the

  actual start time has elapsed.
- Supports programming the width, between the rising edge and corresponding falling edge of PPS signal output, in terms of number of units of sub-second increment value programmed in the Sub-Second Increment Register (Table 540). You can program the width of pulse from 1 to 2^{32}-1 units of sub-second increment value.
- Supports programming the interval, between the rising edges of PPS signal, in terms of number of

  units of sub-second increment value. You can program the interval between pulses from 1 to 2^{32}-1 units of sub-second increment value.
- Provides the option to cancel the programmed PPS start or stop request.
- Indicates error if the start or stop time being programmed has already elapsed.

Remark: The PTP Reference clock mentioned in the following sections is the clock at which the system time gets updated. When the bit 1 (TSCFUPDT: Timestamp Fine or Coarse Update) of the Timestamp Control Register is set to 0, this clock is similar to the <tbd> clock. In the FineCorrection mode, this is the clock tick at which the system time gets updated (<tbd>).

26.9.3.4 PPS start and stop time

You can initially program the start time in the Target Time registers (Table 546 and Table 547). If required, you can again program the start or stop time, but you can do it only after the earlier programmed value is synchronized to the PTP clock domain. The bit 31

(TSTRBUSY) of Target Time Nanoseconds Register (Table 547) indicates that the synchronization is complete. This enables you to program the start or stop time in advance, even before the earlier stop or start time has elapsed.

The start or stop time should be programmed with advanced system time to ensure proper PPS signal output. If the application programs a start or stop time that has already elapsed, then the Ethernet controller sets an error status bit indicating the programming error. If enabled, the Ethernet controller sets the Target Time Reached interrupt event. The application can cancel the start or stop request only if the corresponding start or stop time has not elapsed. If the time has elapsed, the cancel command has no effect.

26.9.3.5 PPS width and interval

The PPS width and interval are programmed in terms of granularity of system time, that is, number of the units of sub-second increment value. For example, to have a PPS pulse width of 40 ns and interval of 100 ns, with the PTP reference clock of 50 MHz, you should program the width and interval to values 2 and 5 respectively. You can achieve smaller granularity by using a faster PTP reference clock.

Before giving the command to trigger a pulse or pulse train on the PPS output, you should program or update the interval and width of the PPS signal output.

26.9.3.6 Auxiliary snapshots with external events

The Ethernet controller supports taking auxiliary timestamp snapshots with an external event. For detailed information, see Section 26.9.3.6.

26.9.4 Transmit path functions

There is no change in the transmit path functions for the Advanced timestamp feature.

The structure of the descriptor changes when you enable the advanced timestamp feature. The advanced timestamp feature is supported only through Alternate (Enhanced) descriptors format. The descriptor is 32-bytes long (8 DWORDS) and the snapshot of the timestamp is written in descriptor TDES6 and TDES7. For detailed information about descriptors, see Section 26.10.3.

26.9.5 Receive path functions

When you select the advanced timestamp feature, the MAC processes the received frames to identify valid PTP frames. You can control the snapshot of the time, to be sent to the application, by using the following options of the timestamp control register (Section 26.6.16):

- When you select the advanced timestamp feature, the MAC processes the received frames to identify valid PTP frames. You can control the snapshot of the time, to be sent to the application, by using the following options of Register 448 (Timestamp Control Register).
- When you select the advanced timestamp feature, the MAC processes the received frames to identify valid PTP frames. You can control the snapshot of the time, to be sent to the application, by using the following options of Register 448 (Timestamp Control Register).
- Enable snapshot for PTP frames transmitted directly over Ethernet or UDP-IP-Ethernet.

- Enable timestamp snapshot for the received frame for IPv4 or IPv6.
- Enable timestamp snapshot for EVENT messages (SYNC, DELAY_REQ, PDELAY_REQ or PDELAY_RESP) only.
- Enable the node to be a Master or Slave and select the snapshot type. This controls the type of messages for which snapshots are taken.

Remark: The ethernet controller also supports PTP messages over VLAN frames.

The MAC provides the timestamp, along with EOF, on a 64-bit bus on MRI. An additional signal validates the presence of timestamp for the receive frame.

The MTL provides the timestamp on the data bus after the EOF data has been transferred. An additional signal validates the timestamp. This signal is asserted to indicate the availability of timestamp. Once the timestamp is transferred, the MTL sends the receive status to the application. In 32-bit datawidth mode, the MTL provides the additional status related to the timestamp and IP checksum offload (IPC) on the data bus after the normal status is read. The additional status is provided only when the bit 0 of the normal status is set and is validated. In 64-bit and 128-bit datawidth mode, the MTL provides the additional status by using the bits 55:32 of the normal status.

The DMA returns the timestamp to the software inside the corresponding Transmit and Receive Descriptor. The advanced timestamp feature is supported only with the 32-bytes long Alternate (Enhanced) descriptor. The extended status, containing the timestamp message status and the IPC status, is written in descriptor RDES4 and the snapshot of the timestamp is written in descriptors RDES6 and RDES7. For detailed information about descriptors, see Section 26.10.3.

26.9.6 Auxiliary snapshot

The auxiliary snapshot feature allows you to store a snapshot of the system time based on an external event. The event is considered to be the rising edge of the sideband signal <tbd>. This feature is independent of whether the system time is generated internally or given as input (on <tbd> bus).

The MAC stores these snapshots in a 4-deep FIFO. Only 64-bits of the timestamp are stored in the FIFO. You can read the upper 16-bits of seconds from the System Time - Higher Word Seconds Register (Table 548) when it is present. When a snapshot is stored, the MAC indicates this to the host with an interrupt. The value of the snapshot is read through a FIFO register access. If the FIFO becomes full and an external trigger

to take the snapshot is asserted, then a snapshot trigger-missed status (ATSSTM) is set in the Timestamp Status Register (Table 549). This indicates that the latest auxiliary snapshot of the timestamp was not stored in the FIFO. The latest snapshot is not written to the FIFO when it is full.

26.10 DMA controller description

The DMA has independent Transmit and Receive engines and a CSR space. The Transmit engine transfers data from system memory to the device port (MTL), while the Receive engine transfers data from the device port to the system memory. The controller uses descriptors to efficiently move data from source to destination with minimal Host CPU intervention. The DMA is designed for packet-oriented data transfers such as frames

in Ethernet. The controller can be programmed to interrupt the Host CPU for situations such as Frame Transmit and Receive transfer completion, and other normal/error conditions.

The DMA and the Host driver communicate through two data structures:

- Control and Status registers (CSR). See Section 26.6.
- Descriptor lists and data buffers. See Section 26.10.3.

The DMA transfers data frames received by the core to the Receive Buffer in the Host memory, and Transmit data frames from the Transmit Buffer in the Host memory. Descriptors that reside in the Host memory act as pointers to these buffers. There are two descriptor lists; one for reception, and one for transmission. The base address of each list is written into DMA Registers Table 557 and Table 558. A descriptor list is forward linked (either implicitly or explicitly). The last descriptor may point back to the first entry to create a ring structure. Explicit chaining of descriptors is accomplished by setting the second address chained in both Receive and Transmit descriptors (RDES1[24] and TDES1[24]). The descriptor lists resides in the Host physical memory address space. Each descriptor can point to a maximum of two buffers. This enables two buffers to be used, physically addressed, rather than contiguous buffers in memory.

A data buffer resides in the Host physical memory space, and consists of an entire frame or part of a frame, but cannot exceed a single frame. Buffers contain only data, buffer status is maintained in the descriptor. Data chaining refers to frames that span multiple data buffers. However, a single descriptor cannot span multiple frames. The DMA skips to the next frame buffer when end-of-frame is detected. Data chaining can be enabled or disabled.

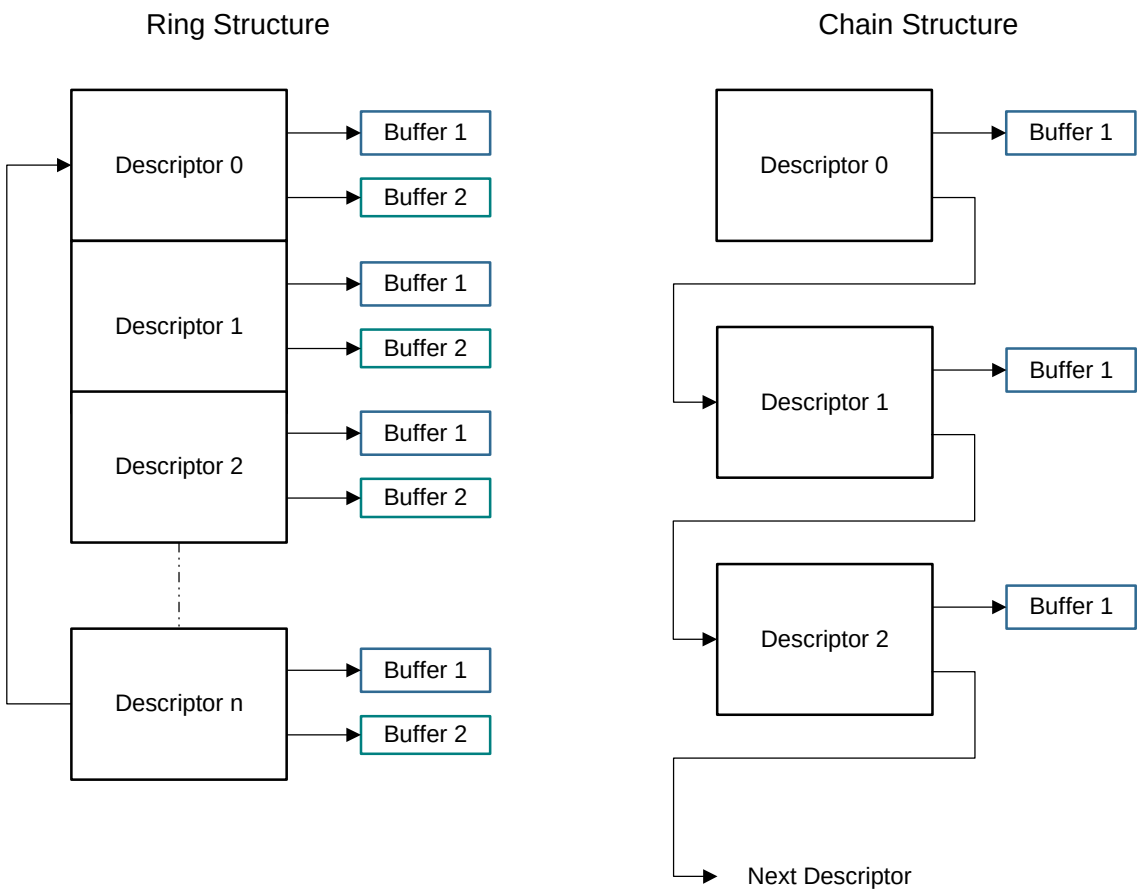


Fig 66. Descriptor ring and chain structure

26.10.1 Initialization

Follow these steps to initialize the ethernet controller:

1. Write to DMA Register Table 553 to set Host bus access parameters.
2. Write to DMA Register Table 561 to mask unnecessary interrupt causes.
3. The software driver creates the Transmit and Receive descriptor lists. Then it writes to both DMA Register Table 557 and DMA Register Table 558, providing the DMA with the starting address of each list.
4. Write to MAC Registers Table 523, Table 525, and Table 524 for desired filtering options.
5. Write to MAC Register Table 522 to configure the operating mode and enable the transmit operation (bit 3: Transmitter Enable). The PS and DM bits are set based on the auto-negotiation result (read from the PHY).
6. Write to DMA Register Table 560 to set bits 13 and 1 to start transmission and reception.
7. Write to MAC Register Table 522 to enable the Receive operation (bit 2: Receiver Enable).

The Transmit and Receive engines enter the Running state and attempt to acquire descriptors from the respective descriptor lists. The Receive and Transmit engines then begin processing Receive and Transmit operations. The Transmit and Receive processes are independent of each other and can be started or stopped separately.

26.10.1.1 Host bus burst access

The DMA attempts to execute fixed-length Burst transfers on the AHB Master interface if configured to do so (FB bit of DMA Register 0). The maximum Burst length is indicated and limited by the PBL field (DMA Register 0[13:8]). The Receive and Transmit descriptors are always accessed in the maximum possible (limited by PBL or 16 x 8/bus width) burst-size for the 16-bytes to be read.

The Transmit DMA initiates a data transfer only when sufficient space to accommodate the configured burst is available in MTL Transmit FIFO or the number of bytes till the end of frame (when it is less than the configured burst-length). The DMA indicates the start address and the number of transfers required to the AHB Master Interface. When the AHB Interface is configured for fixed-length burst, then it transfers data using the best combination of INCR4/8/16 and SINGLE transactions. Otherwise (no fixed-length burst), it transfers data using INCR (undefined length) and SINGLE transactions.

The Receive DMA initiates a data transfer only when sufficient data to accommodate the configured burst is available in MTL Receive FIFO or when the end of frame (when it is less than the configured burst-length) is detected in the Receive FIFO. The DMA indicates the start address and the number of transfers required to the AHB Master Interface. When the AHB Interface is configured for fixed-length burst, then it transfers data using the best combination of INCR4/8/16 and SINGLE transactions. If the end-of frame is reached before the fixed-burst ends on the AHB interface, then dummy transfers are performed in order to complete the fixed-burst. Otherwise (FB bit of DMA Register Table 553 is reset), it transfers data using INCR (undefined length) and SINGLE transactions.

When the AHB interface is configured for address-aligned beats, both DMA engines ensure that the first burst transfer the AHB initiates is less than or equal to the size of the configured PBL. Thus, all subsequent beats start at an address that is aligned to the configured PBL. The DMA can only align the address for beats up to size 16 (for PBL > 16), because the AHB interface does not support more than INCR16.

26.10.1.2 Host data buffer alignment

The Transmit and Receive data buffers do not have any restrictions on start address alignment. For example, in systems with 32-bit memory, the start address for the buffers can be aligned to any of the four bytes. However, the DMA always initiates transfers with address aligned to the bus width with dummy data for the byte lanes not required. This typically happens during the transfer of the beginning or end of an Ethernet frame.

Example: Buffer read

If the Transmit buffer address is 0x00000FF2 (for 32-bit data bus), and 15 bytes need to be transferred, then the DMA reads five full words from address 0x00000FF0, but when transferring data to the MTL Transmit FIFO, the extra bytes (the first two bytes) are dropped or ignored. Similarly, the last 3 bytes of the last transfer are also ignored. The DMA always ensures it transfers a full 32-bit data to the MTL Transmit FIFO, unless it is the end-of-frame.

Example: Buffer write

If the Receive buffer address is 0x0000FF2 (for 64-bit data bus) and 16 bytes of a received frame need to be transferred, then the DMA writes 3 full words from address 0x00000FF0. But the first 2 bytes of first transfer and the last 6 bytes of the third transfer have dummy data.

26.10.1.3 Buffer size calculations

The DMA does not update the size fields in the Transmit and Receive descriptors. The DMA updates only the status fields (RDES and TDES) of the descriptors. The driver has to perform the size calculations.

The transmit DMA transfers the exact number of bytes (indicated by buffer size field of TDES1) towards the MAC core. If a descriptor is marked as first (FS bit of TDES1 is set), then the DMA marks the first transfer from the buffer as the start of frame. If a descriptor is marked as last (LS bit of TDES1), then the DMA marks the last transfer from that data buffer as the end-of frame to the MTL.

The Receive DMA transfers data to a buffer until the buffer is full or the end-of frame is received from the MTL. If a descriptor is not marked as last (LS bit of RDES0), then the descriptor's corresponding buffer(s) are full and the amount of valid data in a buffer is accurately indicated by its buffer size field minus the data buffer pointer offset when the FS bit of that descriptor is set. The offset is zero when the data buffer pointer is aligned to the data bus width. If a descriptor is marked as last, then the buffer may not be full (as indicated by the buffer size in RDES1). To compute the amount of valid data in this final buffer, the driver must read the frame length (FL bits of RDES0[29:16]) and subtract the sum of the buffer sizes of the preceding buffers in this frame. The Receive DMA always transfers the start of next frame with a new descriptor.

Remark: Even when the start address of a receive buffer is not aligned to the system bus's data width, the system should allocate a receive buffer of a size aligned to the system bus width. For example, if the system allocates a 1,024-byte (1 KB) receive buffer starting from address 0x1000, the software can program the buffer start address in the Receive descriptor to have a 0x1002 offset. The Receive DMA writes the frame to this buffer with dummy data in the first two locations (0x1000 and 0x1001). The actual frame is written from location 0x1002. Thus, the actual useful space in this buffer is 1,022 bytes, even though the buffer size is programmed as 1,024 bytes, because of the start address offset.

26.10.1.4 DMA arbiter

The arbiter inside the DMA module performs the arbitration between the Transmit and Receive channel accesses to the AHB Master interface. Two types of arbitrations are possible: round-robin, and fixed-priority.

When round-robin arbitration is selected (DA bit of Register Table 553 (Bus Mode Register) is reset), the arbiter allocates the data bus in the ratio set by the PR bits of DMA Register Table 553, when both Transmit and Receive DMAs are requesting for access simultaneously. When the DA bit is set, the Receive DMA always gets priority over the Transmit DMA for data access by default. When the TXPR bit (bit 27 of DMA register Table 553) is also set, then the Transmit DMA gets priority over the Receive DMA.

26.10.2 Transmission

26.10.2.1 TxDMA operation: Default (non-OSF) mode

The transmit DMA engine in default mode proceeds as follows:

1. The Host sets up the transmit descriptor (TDES0-TDES3) and sets the Own bit (TDES0[31]) after setting up the corresponding data buffer(s) with Ethernet Frame data.
2. Once the ST bit (DMA Register) is set, the DMA enters the Run state.
3. While in the Run state, the DMA polls the Transmit Descriptor list for frames requiring transmission. After polling starts, it continues in either sequential descriptor ring order or chained order. If the DMA detects a descriptor flagged as owned by the Host, or if an error condition occurs, transmission is suspended and both the Transmit Buffer Unavailable (DMA Register Table 559) and Normal Interrupt Summary (DMA Register Table 559) bits are set. The Transmit Engine proceeds to Step 9.
4. If the acquired descriptor is flagged as owned by DMA (TDES0[31] = 1), the DMA decodes the Transmit Data Buffer address from the acquired descriptor.
5. The DMA fetches the Transmit data from the Host memory and transfers the data to the MTL for transmission.
6. If an Ethernet frame is stored over data buffers in multiple descriptors, the DMA closes the intermediate descriptor and fetches the next descriptor. Steps 3, 4, and 5 are repeated until the end-of-Ethernet-frame data is transferred to the MTL.
7. When frame transmission is complete, if IEEE 1588 time stamping was enabled for the frame (as indicated in the transmit status) the timestamp value obtained from MTL is written to the transmit descriptor (TDES2 and TDES3) that contains the end-of-frame buffer. The status information is then written to this transmit descriptor (TDES0). Because the Own bit is cleared during this step, the Host now owns this descriptor. If time stamping was not enabled for this frame, the DMA does not alter the contents of TDES2 and TDES3.
8. Transmit Interrupt (DMA Register Table 559) is set after completing transmission of a frame that has Interrupt on Completion (TDES1[31]) set in its Last Descriptor. The DMA engine then returns to Step 3.
9. In the Suspend state, the DMA tries to re-acquire the descriptor (and thereby return to Step 3) when it receives a Transmit Poll demand and the Underflow Interrupt Status bit is cleared.

The TxDMA transmission flow in default mode is shown in Figure 67.

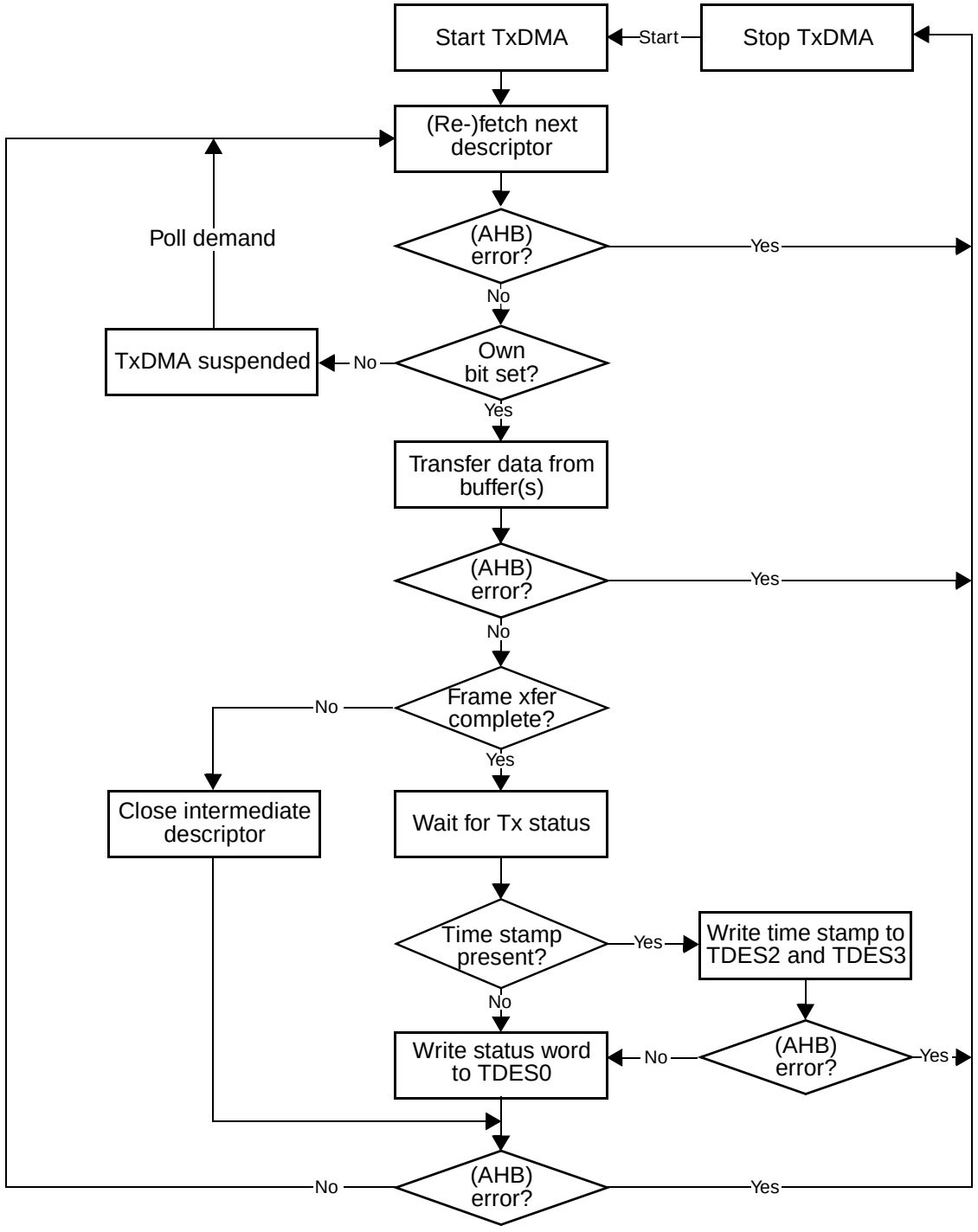


Fig 67. TxDMA operation in default mode

26.10.2.2 TxDMA operation: OSF mode

While in the Run state, the transmit process can simultaneously acquire two frames without closing the Status descriptor of the first (if the OSF bit is set in DMA Operation mode register, bit 2). As the transmit process finishes transferring the first frame, it

immediately polls the Transmit Descriptor list for the second frame. If the second frame is valid, the transmit process transfers this frame before writing the first frame's status information.

In OSF mode, the Run state Transmit DMA operates in the following sequence:

1. The DMA operates as described in steps 1 to 6 of the TxDMA (default mode).
2. Without closing the previous frame's last descriptor, the DMA fetches the next descriptor.
3. If the DMA owns the acquired descriptor, the DMA decodes the transmit buffer address in this descriptor. If the DMA does not own the descriptor, the DMA goes into Suspend mode and skips to Step 7.
4. The DMA fetches the Transmit frame from the Host memory and transfers the frame to the MTL until the End-of-Frame data is transferred, closing the intermediate descriptors if this frame is split across multiple descriptors.
5. The DMA waits for the previous frame's frame transmission status and time stamp. Once the status is available, the DMA writes the time stamp to TDES2 and TDES3, if such time stamp was captured (as indicated by a status bit). The DMA then writes the status, with a cleared Own bit, to the corresponding TDES0, thus closing the descriptor. If time stamping was not enabled for the previous frame, the DMA does not alter the contents of TDES2 and TDES3.
6. If enabled, the Transmit interrupt is set, the DMA fetches the next descriptor, then proceeds to Step 3 (when Status is normal). If the previous transmission status shows an underflow error, the DMA goes into Suspend mode (Step 7).
7. In Suspend mode, if a pending status and time stamp are received from the MTL, the DMA writes the time stamp (if enabled for the current frame) to TDES2 and TDES3, then writes the status to the corresponding TDES0. It then sets relevant interrupts and returns to Suspend mode.
8. The DMA can exit Suspend mode and enter the Run state (go to Step 1 or Step 2 depending on pending status) only after receiving a Transmit Poll demand (DMA Transmit Poll Demand register).

Remark: As the DMA fetches the next descriptor in advance before closing the current descriptor, the descriptor chain should have more than 2 different descriptors for correct and proper operation.

The basic flow is described in Figure 68.

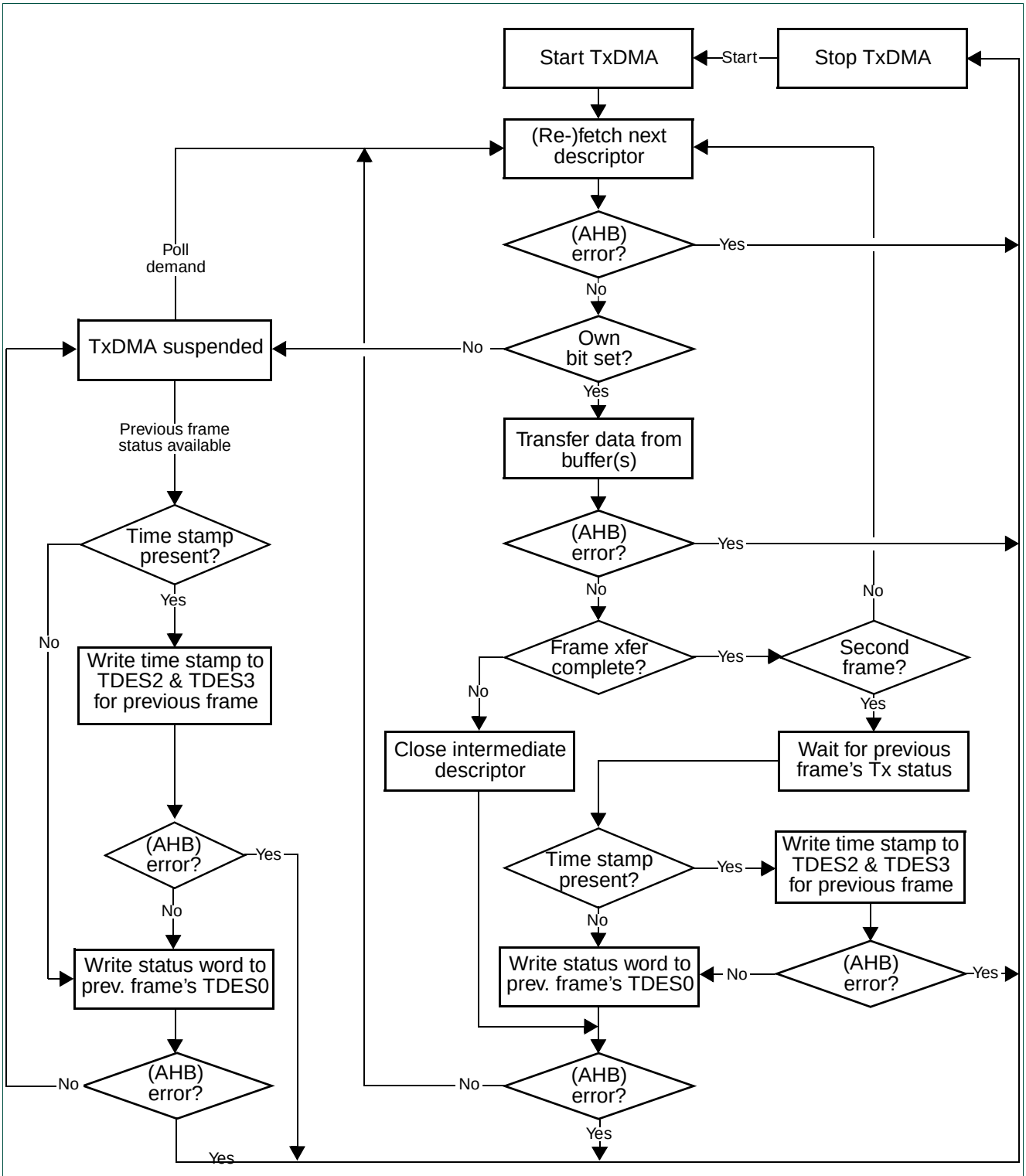


Fig 68. TxDMA operation in OSF mode

26.10.2.3 Transmit frame processing

The Transmit DMA expects that the data buffers contain complete Ethernet frames, excluding preamble, pad bytes, and FCS fields. The DA, SA, and Type/Len fields contain valid data. If the Transmit Descriptor indicates that the MAC core must disable CRC or PAD insertion, the buffer must have complete Ethernet frames (excluding preamble), including the CRC bytes.

Frames can be data-chained and can span several buffers. Frames must be delimited by the First Descriptor (TDES1[29]) and the Last Descriptor (TDES1[30]), respectively.

As transmission starts, the First Descriptor must have (TDES1[29]) set. When this occurs, frame data transfers from the Host buffer to the MTL Transmit FIFO. Concurrently, if the current frame has the Last Descriptor (TDES1[30]) clear, the Transmit Process attempts to acquire the Next Descriptor. The Transmit Process expects this descriptor to have TDES1[29] clear. If TDES1[30] is clear, it indicates an intermediary buffer. If TDES1[30] is set, it indicates the last buffer of the frame.

After the last buffer of the frame has been transmitted, the DMA writes back the final status information to the Transmit Descriptor 0 (TDES0) word of the descriptor that has the last segment set in Transmit Descriptor 1 (TDES1[30]). At this time, if Interrupt on Completion (TDES1[31]) was set, Transmit Interrupt (DMA Status register, bit 0) is set, the Next Descriptor is fetched, and the process repeats.

The actual frame transmission begins after the MTL Transmit FIFO has reached either a programmable transmit threshold (DMA Operation Mode register, bits [16:14]), or a full frame is contained in the FIFO. There is also an option for Store and Forward Mode (DMA Operation Mode register, bit [21]). Descriptors are released (Own bit TDES0[31] clears) when the DMA finishes transferring the frame.

Remark: To ensure proper transmission of a frame and the next frame, you must specify a non-zero buffer size for the transmit descriptor that has the Last Descriptor (TDES1[30]) set.

26.10.2.4 Transmit polling suspended

Transmit polling can be suspended by either of the following conditions:

- The DMA detects a descriptor owned by the Host (TDES0[31]=0). To resume, the driver must give descriptor ownership to the DMA and then issue a Poll Demand command.
- A frame transmission is aborted when a transmit error because of underflow is detected. The appropriate Transmit Descriptor 0 (TDES0) bit is set.

If the second condition occur, both Abnormal Interrupt Summary (DMA Status register Table 559) and Transmit Underflow bits (DMA Status register Table 559) are set, and the information is written to Transmit Descriptor 0, causing the suspension. If the DMA goes into SUSPEND state because of the first condition, then both Normal Interrupt Summary (DMA Status register Table 559) and Transmit Buffer Unavailable (DMA Status register Table 559) are set.

In both cases, the position in the Transmit List is retained. The retained position is that of the descriptor following the Last Descriptor closed by the DMA.

The driver must explicitly issue a Transmit Poll Demand command after rectifying the suspension cause.

26.10.2.5 Reception

The Receive DMA engine's reception sequence is shown in Figure 69 and proceeds as follows:

1. The host sets up Receive descriptors (RDES0-RDES3) and sets the Own bit (RDES0[31]).
2. Once the SR (DMA Operation Mode register Table 560) bit is set, the DMA enters the Run state. While in the Run state, the DMA polls the Receive Descriptor list, attempting to acquire free descriptors. If the fetched descriptor is not free (is owned by the host), the DMA enters the Suspend state and jumps to Step 9.
3. The DMA decodes the receive data buffer address from the acquired descriptors.
4. Incoming frames are processed and placed in the acquired descriptor's data buffers.
5. When the buffer is full or the frame transfer is complete, the Receive engine fetches the next descriptor.
6. If the current frame transfer is complete, the DMA proceeds to Step 7. If the DMA does not own the next fetched descriptor and the frame transfer is not complete (EOF is not yet transferred), the DMA sets the Descriptor Error bit in the RDES0 (unless flushing is disabled). The DMA closes the current descriptor (clears the Own bit) and marks it as intermediate by clearing the Last Segment (LS) bit in the RDES0 value (marks it as Last Descriptor if flushing is not disabled), then proceeds to Step 8. If the DMA does own the next descriptor but the current frame transfer is not complete, the DMA closes the current descriptor as intermediate and reverts to Step 4.
7. If IEEE 1588 time stamping is enabled, the DMA writes the timestamp (if available) to the current descriptor's RDES2 and RDES3. It then takes the receive frame's status from the MTL and writes the status word to the current descriptor's RDES0, with the Own bit cleared and the Last Segment bit set.
8. The Receive engine checks the latest descriptor's Own bit. If the host owns the descriptor (Own bit is 0) the Receive Buffer Unavailable bit (DMA Status register Table 559) is set and the DMA Receive engine enters the Suspended state (Step 9). If the DMA owns the descriptor, the engine returns to Step 4 and awaits the next frame.
9. Before the Receive engine enters the Suspend state, partial frames are flushed from the Receive FIFO (You can control flushing using Bit 24 of DMA Operation MOde register Table 560).
10. The Receive DMA exits the Suspend state when a Receive Poll demand is given or the start of next frame is available from the MTL's Receive FIFO. The engine proceeds to Step 2 and refetches the next descriptor.

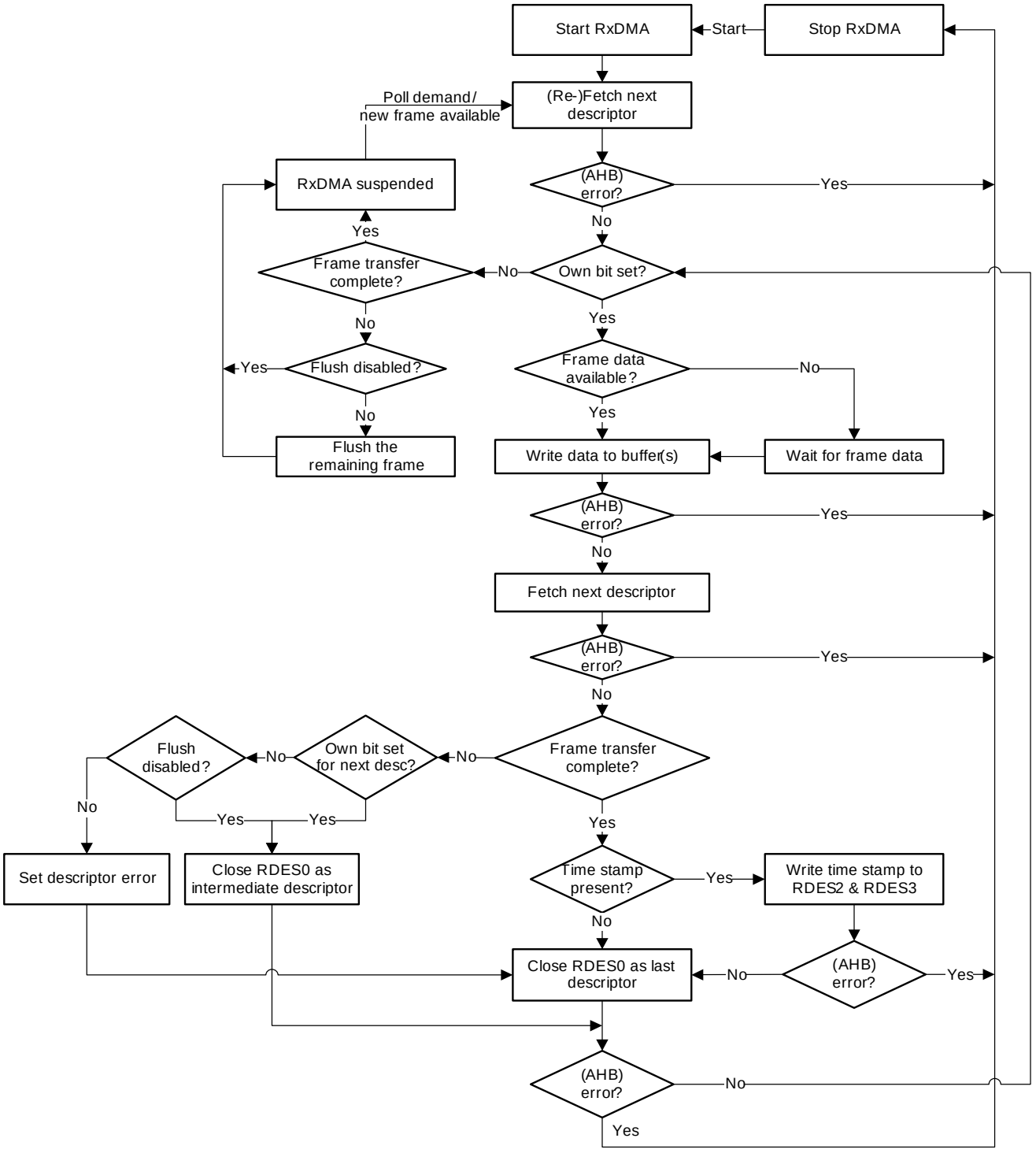


Fig 69. Receive DMA operation

The DMA does not acknowledge accepting the status from the MTL until it has completed the time stamp write-back and is ready to perform status write-back to the descriptor.

If software has enabled time stamping through CSR, when a valid time stamp value is not available for the frame (for example, because the receive FIFO was full before the time stamp could be written to it), the DMA writes all-ones to RDES2 and RDES3. Otherwise (that is, if time stamping is not enabled), the RDES2 and RDES3 remain unchanged.

26.10.2.6 Receive descriptor acquisition

The Receive Engine always attempts to acquire an extra descriptor in anticipation of an incoming frame. Descriptor acquisition is attempted if any of the following conditions is satisfied:

- The receive Start/Stop bit (DMA Operation Mode register Table 560) has been set immediately after being placed in the Run state.
- The data buffer of current descriptor is full before the frame ends for the current transfer.
- The controller has completed frame reception, but the current Receive Descriptor is not yet closed.
- The receive process has been suspended because of a host-owned buffer (RDES0[31] = 0) and a new frame is received.
- A Receive poll demand has been issued.

26.10.2.7 Receive frame processing

The MAC transfers the received frames to the Host memory only when the frame passes the address filter and frame size is greater than or equal to configurable threshold bytes set for the Receive FIFO of MTL, or when the complete frame is written to the FIFO in Store-and-Forward mode.

If the frame fails the address filtering, it is dropped in the MAC block itself (unless Receive All bit 31 is set in the MAC Frame Filter register; Table 523). Frames that are shorter than 64 bytes, because of collision or premature termination, can be purged from the MTL Receive FIFO.

After 64 (configurable threshold) bytes have been received, the MTL block requests the DMA block to begin transferring the frame data to the Receive Buffer pointed to by the current descriptor. The DMA sets First Descriptor (RDES0[9]) after the DMA Host Interface (AHB or MDC) becomes ready to receive a data transfer (if DMA is not fetching transmit data from the host), to delimit the frame. The descriptors are released when the Own (RDES[31]) bit is reset to 0, either as the Data buffer fills up or as the last segment of the frame is transferred to the Receive buffer. If the frame is contained in a single descriptor, both Last Descriptor (RDES[8]) and First Descriptor (RDES[9]) are set.

The DMA fetches the next descriptor, sets the Last Descriptor (RDES[8]) bit, and releases the RDES0 status bits in the previous frame descriptor. Then the DMA sets Receive Interrupt (Register 5[6]). The same process repeats unless the DMA encounters a descriptor flagged as being owned by the host. If this occurs, the Receive Process sets Receive Buffer Unavailable (DMA Status register Table 559) and then enters the Suspend state. The position in the receive list is retained.

26.10.2.8 Receive process suspended

If a new Receive frame arrives while the Receive Process is in Suspend state, the DMA refetches the current descriptor in the Host memory. If the descriptor is now owned by the DMA, the Receive Process re-enters the Run state and starts frame reception. If the descriptor is still owned by the host, by default, the DMA discards the current frame at the top of the MTL Rx FIFO and increments the missed frame counter. If more than one frame is stored in the MTL Rx FIFO, the process repeats.

The discarding or flushing of the frame at the top of the MTL Rx FIFO can be avoided by setting Operation Mode register bit 24 (DFF) in Table 560. In such conditions, the receive process sets the Receive Buffer Unavailable status and returns to the Suspend state.

26.10.2.9 Interrupts

Interrupts can be generated as a result of various events. The DMA Status register (Table 559) contains all the bits that might cause an interrupt. Table 561 contains an enable bit for each of the events that can cause an interrupt.

There are two groups of interrupts, Normal and Abnormal, as described in DMA Status register (Table 559). Interrupts are cleared by writing a 1 to the corresponding bit position. When all the enabled interrupts within a group are cleared, the corresponding summary bit is cleared. When both the summary bits are cleared, the interrupt signal is de-asserted. If the MAC core is the cause for assertion of the interrupt, then any of the GLI, GMI, or GPI bits of DMA Status register (Table 559) are set HIGH.

Remark: The DMA Status register (Table 559) is the (interrupt) status register. The interrupt pin is asserted because of any event in this status register only if the corresponding interrupt enable bit is set in DMA Interrupt Enable Register (Table 561).

Interrupts are not queued and if the interrupt event occurs before the driver has responded to it, no additional interrupts are generated. For example, the Receive Interrupt (bit 6 of the DMA Status Register (Table 559) indicates that one or more frames were transferred to the Host buffer. The driver must scan all descriptors, from the last recorded position to the first one owned by the DMA.

An interrupt is generated only once for simultaneous, multiple events. The driver must scan the DMA Status register (Table 559) for the cause of the interrupt. The interrupt is not generated again unless a new interrupting event occurs, after the driver has cleared the appropriate bit in DMA Status register. For example, the controller generates a DMA Receive interrupt (bit 6 of the DMA Status register), and the driver begins reading DMA Status register. Next, Receive Buffer Unavailable (bit 7 of DMA Status register (Status Register)) occurs. The driver clears the Receive interrupt. Even then, the sbd_intr_o signal is not de-asserted, because of the active or pending Receive Buffer Unavailable interrupt.

An interrupt timer RIWT (bits 7:0 in Receive Interrupt Watchdog Timer Register (Table 563)) is given for flexible control of Receive Interrupt. When this Interrupt timer is programmed with a non-zero value, it gets activated as soon as the RxDMA completes a transfer of a received frame to system memory without asserting the Receive Interrupt because it is not enabled in the corresponding Receive Descriptor (RDES1[31]. When this timer runs out as per the programmed value, RI bit is set and the interrupt is asserted if

the corresponding RI is enabled in DMA Interrupt Enable register (Table 561). This timer gets disabled before it runs out, when a frame is transferred to memory and the RI is set because it is enabled for that descriptor.

26.10.2.10 Error response to DMA

For any data transfer initiated by a DMA channel, if the slave replies with an error response, that DMA stops all operations and updates the error bits and the Fatal Bus Error bit in the DMA Status register (Table 559). That DMA controller can resume operation only after soft resetting or hard resetting the core and re-initializing the DMA. This DMA behavior is true for non-AHB interfaced DMAs that receive an error response.

26.10.3 Ethernet descriptors

The descriptor structure supports up to 8 DWORDS (32 bytes) and the IEEE 1588-2008 Advanced Timestamp feature or the AV feature. The features of the descriptor structure are:

- Descriptor size can be 4 DWORDS (16 bytes) or 8 DWORDS (32 bytes) depending on the setting of the ATDS bit in the DMA Bus Mode register (Table 553).
- Support buffers of up to 8 KB (useful for Jumbo frames).
- The transmit descriptor stores the timestamp in TDES6 and TDES7 when you select the Advanced Timestamp.
- This receive descriptor structure is also used for storing the extended status (RDES4) and timestamp (RDES6 and RDES7) when advanced timestamp feature or IPC full offload is selected.
- When the descriptor mode is selected, and the Timestamp feature is enabled, the software needs to allocate 32-bytes (8 DWORDS) of memory for every descriptor. When Timestamping or Receive IPC FullOffload engine are not enabled, the extended descriptors are not required and the SW can use alternate descriptors with the default size of 16 bytes. The core also needs to be configured for this change using the bit 7 (ATDS: Alternate Descriptor Size) of DMA Bus Mode register (Table 553).
- When a descriptor is chosen without Timestamp or Full IPC Offload feature, the descriptor size is always 4 DWORDs (DES0-DES3).

26.10.3.1 Transmit descriptor

The transmit descriptor structure is shown in Figure 70. The application software must program the control bits TDES0[31:20] during descriptor initialization. When the DMA updates the descriptor, it write backs all the control bits except the OWN bit (which it clears) and updates the status bits[19:0]. The contents of the transmitter descriptor word 0 (TDES0) through word 3 (TDES3) are given in Table 576 through Table 579, respectively.

With the advance timestamp support, the snapshot of the timestamp to be taken can be enabled for a given frame by setting bit TTSE: Transmit Timestamp Enable. (bit-25 of TDES0). When the descriptor is closed (i.e. when the OWN bit is cleared), the time-stamp is written into TDES6 and TDES7. This is indicated by the status bit TTSS: Transmit Timestamp Status. (bit-17 of TDES0). This is shown in Figure 70. The contents of TDES6 and TDES7 are mentioned in Table 580 to Table 581.

When either Advanced Timestamp or IPC Offload (Type 2) features is enabled, the SW should set the DMA Bus Mode register[7], so that the DMA operates with extended descriptor size. When this control bit is reset, the TDES4-TDES7 descriptor space are not valid.

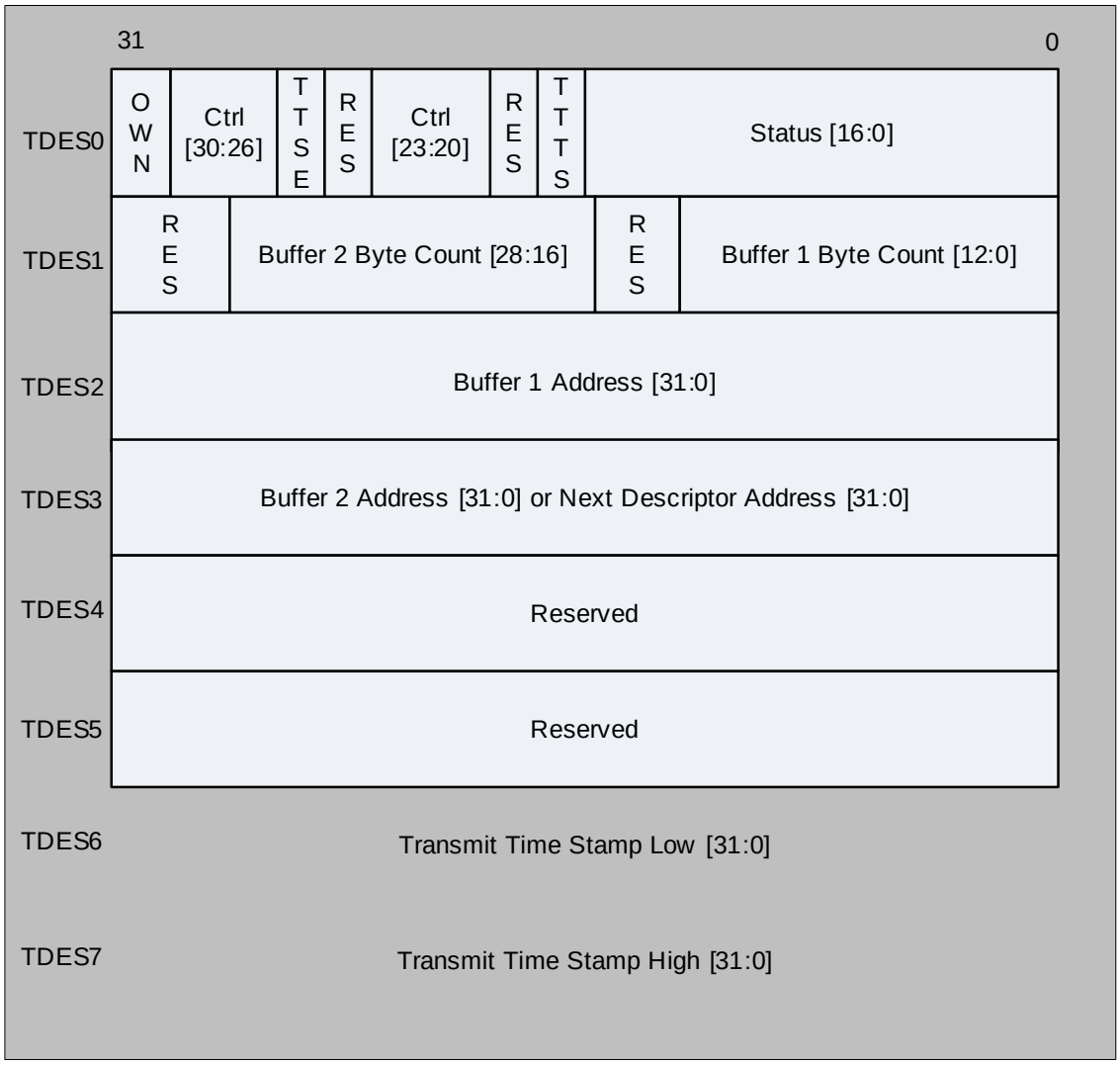


Fig 70. Transmitter descriptor fields

The DMA always reads or fetches four DWORDS of the descriptor from system memory to obtain the buffer and control information as shown in Figure 71. When Advanced timestamp feature support is enabled, TDES0 has additional control bits[6:3] for channel 1 and channel 2. For channel 0, the bits 6:3 are ignored. The bits 6:3 are described in Table 576.

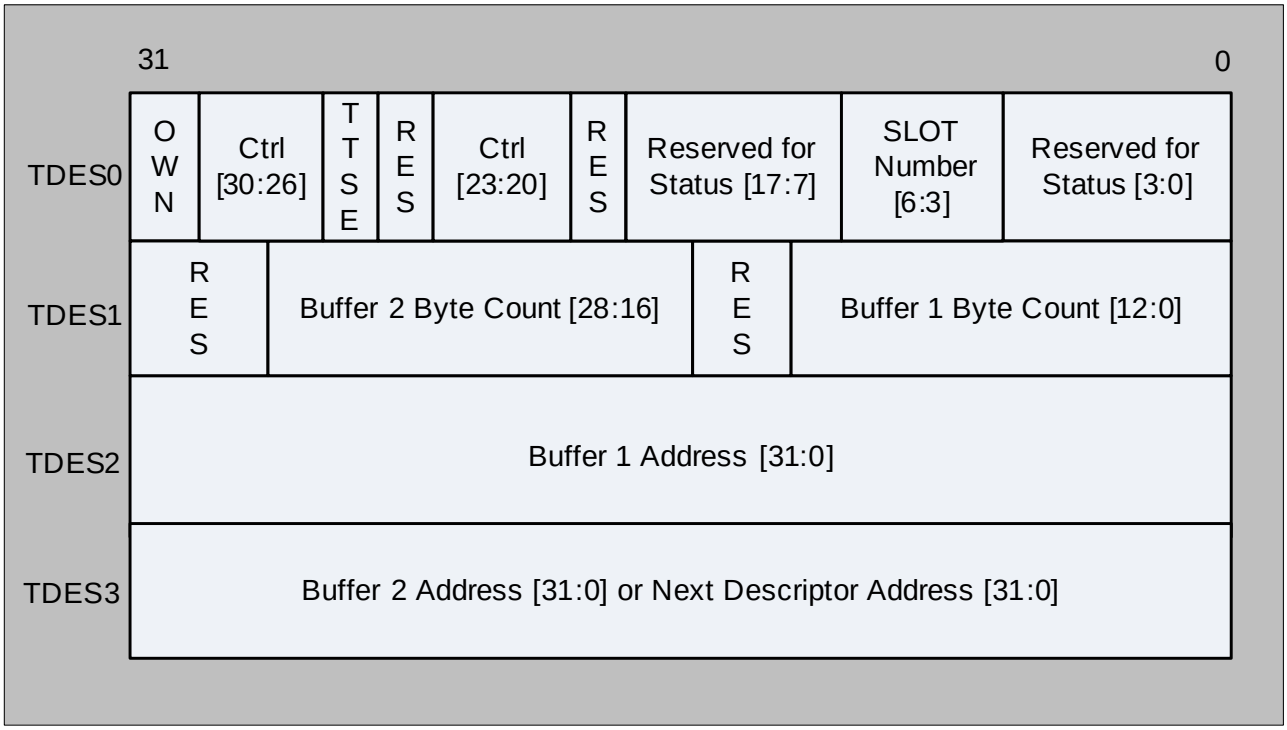


Fig 71. Transmit descriptor fetch (read)

Table 576. Transmit descriptor word 0 (TDES0)

Bit	Symbol	Description
0	DB	Deferred Bit When set, this bit indicates that the MAC defers before transmission because of the presence of carrier. This bit is valid only in Half-Duplex mode.
1	UF	Underflow Error When set, this bit indicates that the MAC aborted the frame because data arrived late from the Host memory. Underflow Error indicates that the DMA encountered an empty transmit buffer while transmitting the frame. The transmission process enters the Suspended state and sets both Transmit Underflow (Register 5[5]) and Transmit Interrupt (Register 5[0]).
2	ED	Excessive Deferral When set, this bit indicates that the transmission has ended because of excessive deferral of over 24,288 bit times (155,680 bits times in 1,000-Mbps mode or if Jumbo Frame is enabled) if the Deferral Check (DC) bit in the MAC Control register is set high.

Table 576. Transmit descriptor word 0 (TDES0)

Bit	Symbol	Description
6:3	CC/ SLOTNUM	CC: Collision Count (Status field) These status bits indicate the number of collisions that occurred before the frame was transmitted. This count is not valid when the Excessive Collisions bit (TDES0[8]) is set. The core updates this status field only in the half-duplex mode. SLOTNUM: Slot Number Control Bits in AV Mode These bits indicate the slot interval in which the data should be fetched from the corresponding buffers addressed by TDES2 or TDES3. When the transmit descriptor is fetched, the DMA compares the slot number value in this field with the slot interval maintained in the core (Register 11xx). It fetches the data from the buffers only if there is a match in values. These bits are valid only for the AV channels (not channel 0).
7	VF	VLAN Frame When set, this bit indicates that the transmitted frame was a VLAN-type frame.
8	EC	Excessive Collision When set, this bit indicates that the transmission was aborted after 16 successive collisions while attempting to transmit the current frame. If the DR (Disable Retry) bit in the MAC Configuration register is set, this bit is set after the first collision, and the transmission of the frame is aborted.
9	LC	Late Collision When set, this bit indicates that frame transmission was aborted due to a collision occurring after the collision window (64 byte-times, including preamble, in MII mode and 512 byte-times, including preamble and carrier extension, in MII mode). This bit is not valid if the Underflow Error bit is set.
10	NC	No Carrier When set, this bit indicates that the Carrier Sense signal form the PHY was not asserted during transmission.
11	LC	Loss of Carrier When set, this bit indicates that a loss of carrier occurred during frame transmission (that is, the gmii_crs_i signal was inactive for one or more transmit clock periods during frame transmission). This is valid only for the frames transmitted without collision when the MAC operates in Half-Duplex mode.
12	IPE	IP Payload Error When set, this bit indicates that MAC transmitter detected an error in the TCP, UDP, or ICMP IP datagram payload. The transmitter checks the payload length received in the IPv4 or IPv6 header against the actual number of TCP, UDP, or ICMP packet bytes received from the application and issues an error status in case of a mismatch.
13	FF	Frame Flushed When set, this bit indicates that the DMA/MTL flushed the frame due to a software Flush command given by the CPU.
14	JT	Jabber Timeout When set, this bit indicates the MAC transmitter has experienced a jabber time-out. This bit is only set when the MAC configuration register's JD bit is not set.

Table 576. Transmit descriptor word 0 (TDES0)

Bit	Symbol	Description
15	ES	Error Summary Indicates the logical OR of the following bits: • TDES0[14]: Jabber Timeout • TDES0[13]: Frame Flush • TDES0[11]: Loss of Carrier • TDES0[10]: No Carrier • TDES0[9]: Late Collision • TDES0[8]: Excessive Collision • TDES0[2]: Excessive Deferral • TDES0[1]: Underflow Error • TDES0[16]: IP Header Error • TDES0[12]: IP Payload Error
16	IHE	IP Header Error When set, this bit indicates that the MAC transmitter detected an error in the IP datagram header. The transmitter checks the header length in the IPv4 packet against the number of header bytes received from the application and indicates an error status if there is a mismatch. For IPv6 frames, a header error is reported if the main header length is not 40 bytes. Furthermore, the Ethernet Length/Type field value for an IPv4 or IPv6 frame must match the IP header version received with the packet. For IPv4 frames, an error status is also indicated if the Header Length field has a value less than 0x5.
17	TTSS	Transmit Timestamp Status This field is used as a status bit to indicate that a timestamp was captured for the described transmit frame. When this bit is set, TDES2 and TDES3 have a timestamp value captured for the transmit frame. This field is only valid when the descriptor's Last Segment control bit (TDES0[29]) is set.
19:18	-	Reserved
20	TCH	Second Address Chained When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When TDES0[20] is set, TBS2 (TDES1[28:16]) is a "don't care" value. TDES0[21] takes precedence over TDES0[20].
21	TER	Transmit End of Ring When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a descriptor ring.
23:22	CIC	Checksum Insertion Control These bits control the checksum calculation and insertion. Bit encodings are as shown below. • 00: Checksum Insertion Disabled. • 01: Only IP header checksum calculation and insertion are enabled. • 10: IP header checksum and payload checksum calculation and insertion are enabled, but pseudo-header checksum is not calculated in hardware. • 11: IP Header checksum and payload checksum calculation and insertion are enabled, and pseudo-header checksum is calculated in hardware. This field is reserved when the IPC_FULL_OFFLOAD configuration parameter is not selected.
24	-	Reserved
25	TTSE	Transmit Timestamp Enable When set, this bit enables IEEE1588 hardware time stamping for the transmit frame referenced by the descriptor. This field is valid only when the First Segment control bit (TDES0[28]) is set.

Table 576. Transmit descriptor word 0 (TDES0)

Bit	Symbol	Description
26	DP	Disable Pad When set, the MAC does not automatically add padding to a frame shorter than 64 bytes. When this bit is reset, the DMA automatically adds padding and CRC to a frame shorter than 64 bytes, and the CRC field is added despite the state of the DC (TDES0[27]) bit. This is valid only when the first segment (TDES0[28]) is set.
27	DC	Disable CRC When this bit is set, the MAC does not append a cyclic redundancy check (CRC) to the end of the transmitted frame. This is valid only when the first segment (TDES0[28]) is set.
28	FS	First Segment When set, this bit indicates that the buffer contains the first segment of a frame.
29	LS	Last Segment When set, this bit indicates that the buffer contains the last segment of the frame. When this bit is set, the TBS1: Transmit Buffer 1 Size or TBS2: Transmit Buffer 2 Size field in TDES1 should have a non-zero value.
30	IC	Interrupt on Completion When set, this bit sets the Transmit Interrupt (Register 5[0]) after the present frame has been transmitted.
31	OWN	Own Bit When set, this bit indicates that the descriptor is owned by the DMA. When this bit is reset, it indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame transmission or when the buffers allocated in the descriptor are read completely. The ownership bit of the frame's first descriptor must be set after all subsequent descriptors belonging to the same frame have been set. This avoids a possible race condition between fetching a descriptor and the driver setting an ownership bit.

Table 577. Transmit descriptor word 1 (TDES1)

Bit	Symbol	Description
12:0	TBS1	Transmit buffer 1 size These bits indicate the first data buffer byte size, in bytes. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or the next descriptor, depending on the value of TCH (TDES0[20]).
15:13	-	Reserved
28:16	TBS2	These bits indicate the second data buffer size in bytes. This field is not valid if TDES0[20] is set. See Section 26.10.1.3.
31:29	-	Reserved

Table 578. Transmit descriptor word 2 (TDES2)

Bit	Symbol	Description
31:0	B1ADD	Buffer 1 Address Pointer These bits indicate the physical address of Buffer 1. There is no limitation on the buffer address alignment. See Section 26.10.1.2 for further detail on buffer address alignment.

Table 579. Transmit descriptor word 3 (TDES3)

Bit	Symbol	Description
31:0	B2ADD	Buffer 2 Address Pointer (Next Descriptor Address) Indicates the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (TDES1[24]) bit is set, this address contains the pointer to the physical memory where the Next Descriptor is present. The buffer address pointer must be aligned to the bus width only when TDES1[24] is set. (LSBs are ignored internally.)

Table 580. Transmit descriptor word 6 (TDES6)

Bit	Symbol	Description
31:0	TTSL	Transmit Frame Timestamp Low This field is updated by DMA with the least significant 32 bits of the timestamp captured for the corresponding transmit frame. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and Timestamp status (TTSS) bit is set.

Table 581. Transmit descriptor word 7 (TDES7)

Bit	Symbol	Description
31:0	TTSH	Transmit Frame Timestamp High This field is updated by DMA with the most significant 32 bits of the timestamp captured for the corresponding receive frame. This field has the timestamp only if the Last Segment bit (LS) in the descriptor is set and Timestamp status (TTSS) bit is set.

26.10.3.2 Receive descriptor

The structure of the received descriptor is shown in Figure 72. This can have 32 bytes of descriptor data (8 DWORDs) when Advanced Timestamp or IPC Full Offload feature is selected.

Remark: When either of these features is enabled, the SW should set the DMA Bus Mode register[7] so that the DMA operates with extended descriptor size. When this control bit is reset, RDES0[7] and RDES0[0] is always cleared and the RDES4-RDES7 descriptor space are not valid.

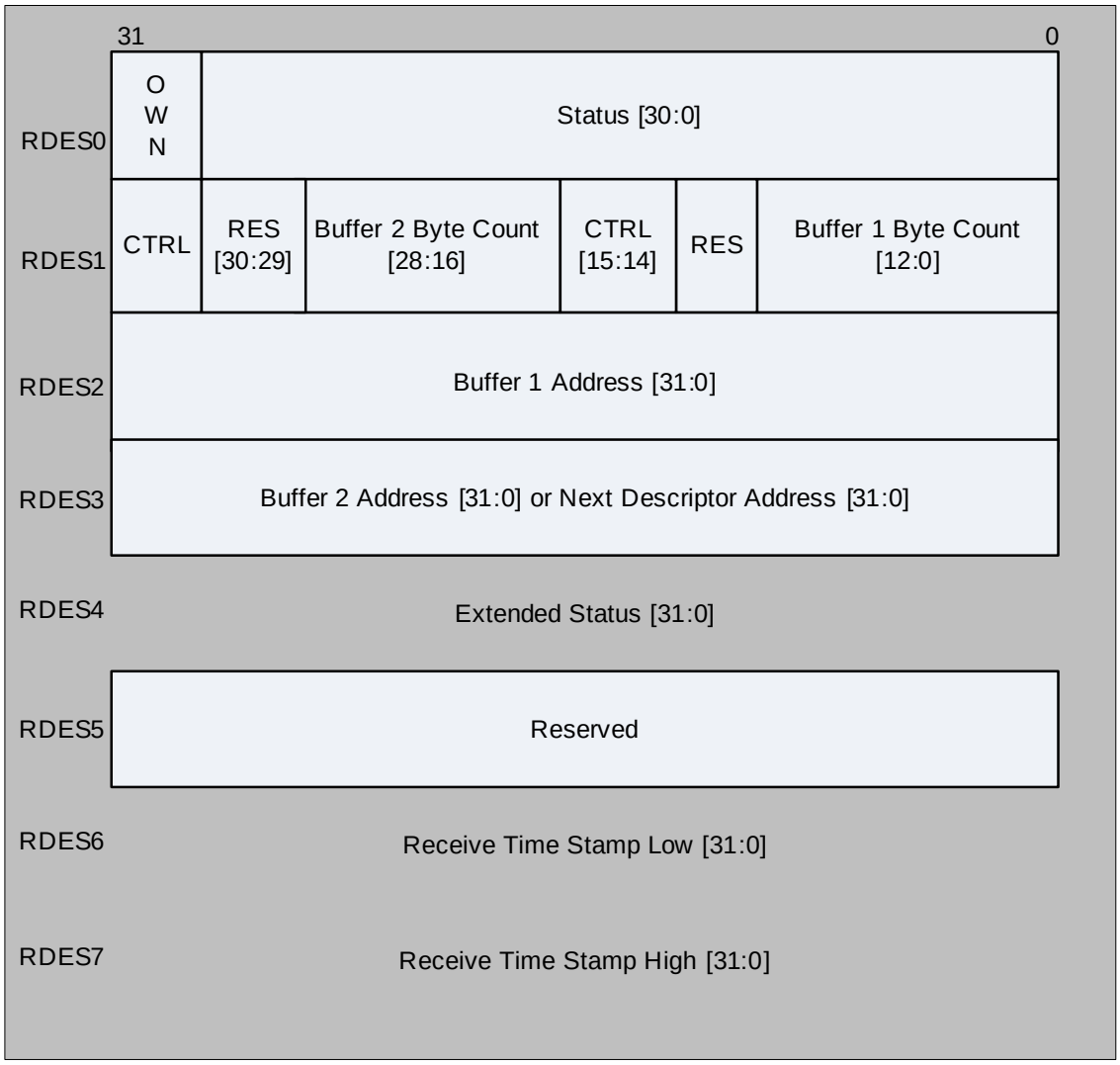


Fig 72. Receive descriptor fields (alternate configuration)

The contents of RDES0 are identified in Table 582. The contents of RDES1 through RDES3 are identified in Table 583 to Table 585.

Table 582. Receive descriptor fields 0 (RDES0)

Bit	Symbol	Description
0	ESA	Extended Status Available/Rx MAC Address When either Advanced Timestamp or IP Checksum Offload (Type 2) is present, this bit, when set, indicates that the extended status is available in descriptor word 4 (RDES4). This is valid only when the Last Descriptor bit (RDES0[8]) is set. When Advance Timestamp Feature or IPC Full Offload is not selected, this bit indicates Rx MAC Address status. When set, this bit indicates that the Rx MAC Address registers value (1 to 31) matched the frame's DA field. When reset, this bit indicates that the Rx MAC Address Register 0 value matched the DA field.
1	CE	CRC Error When set, this bit indicates that a Cyclic Redundancy Check (CRC) Error occurred on the received frame. This field is valid only when the Last Descriptor (RDES0[8]) is set.
2	DE	Dribble Bit Error When set, this bit indicates that the received frame has a non-integer multiple of bytes (odd nibbles). This bit is valid only in MII Mode.
3	RE	Receive Error When set, this bit indicates that the gmii_rxer_i signal is asserted while gmii_rxdv_i is asserted during frame reception. This error also includes carrier extension error in MII and Half-duplex mode. Error can be of less/no extension, or error (rxd $\neq$ 0f) during extension.
4	RWT	Receive Watchdog Timeout When set, this bit indicates that the Receive Watchdog Timer has expired while receiving the current frame and the current frame is truncated after the Watchdog Timeout.
5	FT	Frame Type When set, this bit indicates that the Receive Frame is an Ethernet-type frame (the LT field is greater than or equal to 0x0600). When this bit is reset, it indicates that the received frame is an IEEE802.3 frame. This bit is not valid for Runt frames less than 14 bytes.
6	LC	Late Collision When set, this bit indicates that a late collision has occurred while receiving the frame in Half-Duplex mode.
7	TSA	Timestamp Available/IP Checksum Error (Type1) /Giant Frame When Advanced Timestamp feature is present, when set, this bit indicates that a snapshot of the Timestamp is written in descriptor words 6 (RDES6) and 7 (RDES7). This is valid only when the Last Descriptor bit (RDES0[8]) is set. When IP Checksum Engine (Type 1) is selected, this bit, when set, indicates that the 16-bit IPv4 Header checksum calculated by the core did not match the received checksum bytes. Otherwise, this bit, when set, indicates the Giant Frame Status. Giant frames are larger-than-1,518-byte (or 1,522-byte for VLAN) normal frames and larger-than-9,018-byte (9,022-byte for VLAN) frame when Jumbo Frame processing is enabled.
8	LS	Last Descriptor When set, this bit indicates that the buffers pointed to by this descriptor are the last buffers of the frame
9	FS	First Descriptor When set, this bit indicates that this descriptor contains the first buffer of the frame. If the size of the first buffer is 0, the second buffer contains the beginning of the frame. If the size of the second buffer is also 0, the next Descriptor contains the beginning of the frame.

Table 582. Receive descriptor fields 0 (RDES0)

Bit	Symbol	Description
10	VLAN	VLAN Tag When set, this bit indicates that the frame pointed to by this descriptor is a VLAN frame tagged by the MAC Core.
11	OE	Overflow Error When set, this bit indicates that the received frame was damaged due to buffer overflow in MTL.
12	LE	Length Error When set, this bit indicates that the actual length of the frame received and that the Length/ Type field does not match. This bit is valid only when the Frame Type (RDES0[5]) bit is reset.
13	SAF	Source Address Filter Fail When set, this bit indicates that the SA field of frame failed the SA Filter in the MAC Core.
14	DE	Descriptor Error When set, this bit indicates a frame truncation caused by a frame that does not fit within the current descriptor buffers, and that the DMA does not own the Next Descriptor. The frame is truncated. This field is valid only when the Last Descriptor (RDES0[8]) is set.
15	ES	ES: Error Summary Indicates the logical OR of the following bits: • RDES0[1]: CRC Error • RDES0[3]: Receive Error • RDES0[4]: Watchdog Timeout • RDES0[6]: Late Collision • RDES0[7]: Giant Frame • RDES4[4:3]: IP Header/Payload Error • RDES0[11]: Overflow Error • RDES0[14]: Descriptor Error This field is valid only when the Last Descriptor (RDES0[8]) is set.
29:16	FL	Frame Length These bits indicate the byte length of the received frame that was transferred to host memory (including CRC). This field is valid when Last Descriptor (RDES0[8]) is set and either the Descriptor Error (RDES0[14]) or Overflow Error bits are reset. The frame length also includes the two bytes appended to the Ethernet frame when IP checksum calculation (Type 1) is enabled and the received frame is not a MAC control frame. This field is valid when Last Descriptor (RDES0[8]) is set. When the Last Descriptor and Error Summary bits are not set, this field indicates the accumulated number of bytes that have been transferred for the current frame.
30	AFM	Destination Address Filter Fail When set, this bit indicates a frame that failed in the DA Filter in the MAC Core.
31	OWN	Own Bit When set, this bit indicates that the descriptor is owned by the DMA of the MAC Subsystem. When this bit is reset, this bit indicates that the descriptor is owned by the Host. The DMA clears this bit either when it completes the frame reception or when the buffers that are associated with this descriptor are full.

Table 583. Receive descriptor fields 1 (RDES1)

Bit	Symbol	Description
12:0	RBS1	Receive Buffer 1 Size Indicates the first data buffer size in bytes. The buffer size must be a multiple of 4, 8, or 16, depending upon the bus widths (32, 64, or 128), even if the value of RDES2 (buffer1 address pointer) is not aligned. When the buffer size is not a multiple of 4, 8, or 16, the resulting behavior is undefined. If this field is 0, the DMA ignores this buffer and uses Buffer 2 or next descriptor depending on the value of RCH (Bit 14). See Section 26.10.1.3 for further details on calculating buffer sizes.
13	-	Reserved
14	RCH	Second Address Chained When set, this bit indicates that the second address in the descriptor is the Next Descriptor address rather than the second buffer address. When this bit is set, RBS2 (RDES1[28:16]) is a “don’t care” value. RDES1[15] takes precedence over RDES1[14].
15	RER	Receive End of Ring When set, this bit indicates that the descriptor list reached its final descriptor. The DMA returns to the base address of the list, creating a descriptor ring.
28:16	RBS2	Receive Buffer 2 Size These bits indicate the second data buffer size, in bytes. The buffer size must be a multiple of 4, 8, or 16, depending on the bus widths (32, 64, or 128, respectively), even if the value of RDES3 (buffer2 address pointer) is not aligned to bus width. If the buffer size is not an appropriate multiple of 4, 8, or 16, the resulting behavior is undefined. This field is not valid if RDES1[14] is set. See Section 26.10.1.3 for further details on calculating buffer sizes.

Table 584. Receive descriptor fields 2 (RDES2)

Bit	Symbol	Description
31:0	B1ADD	Address Pointer These bits indicate the physical address of Buffer 1. There are no limitations on the buffer address alignment except for the following condition: The DMA uses the configured value for its address generation when the RDES2 value is used to store the start of frame. Note that the DMA performs a write operation with the RDES2[3/2/1:0] bits as 0 during the transfer of the start of frame but the frame data is shifted as per the actual Buffer address pointer. The DMA ignores RDES2[3/2/1:0] (corresponding to bus width of 128/64/32) if the address pointer is to a buffer where the middle or last part of the frame is stored. See Section 26.10.1.2for further details on buffer address alignment.

Table 585. Receive descriptor fields 3 (RDES3)

Bit	Symbol	Description
31:0	B2ADD	Buffer 2 Address Pointer (Next Descriptor Address) These bits indicate the physical address of Buffer 2 when a descriptor ring structure is used. If the Second Address Chained (RDES1[24]) bit is set, this address contains the pointer to the physical memory where the Next Descriptor is present. If RDES1[24] is set, the buffer (Next Descriptor) address pointer must be bus width-aligned (RDES3[3, 2, or 1:0] = 0, corresponding to a bus width of 128, 64, or 32. LSBs are ignored internally.) However, when RDES1[24] is reset, there are no limitations on the RDES3 value, except for the following condition: The DMA uses the configured value for its buffer address generation when the RDES3 value is used to store the start of frame. The DMA ignores RDES3 [3, 2, or 1:0] (corresponding to a bus width of 128, 64, or 32) if the address pointer is to a buffer where the middle or last part of the frame is stored.

The extended status written is as shown in Table 586. The extended status is written only when there is status related to IPC or timestamp available. The availability of extended status is indicated by bit-0 of RDES0. This status is available only when Advance Timestamp or IPC Full Offload feature is selected.

Table 586. Receive descriptor fields 4 (RDES4)

Bit	Symbol	Description
2:0	IPPL	IP Payload Type These bits indicate the type of payload encapsulated in the IP datagram processed by the Receive Checksum Offload Engine (COE). The COE also sets these bits to 00 if it does not process the IP datagram's payload due to an IP header error or fragmented IP. • 000: Unknown or did not process IP payload • 001: UDP • 010: TCP • 011: ICMP • 1xx: Reserved
3	IPHE	IP Header Error When set, this bit indicates either that the 16-bit IPv4 header checksum calculated by the core does not match the received checksum bytes, or that the IP datagram version is not consistent with the Ethernet Type value.
4	IPPLE	IP Payload Error When set, this bit indicates that the 16-bit IP payload checksum (that is, the TCP, UDP, or ICMP checksum) that the core calculated does not match the corresponding checksum field in the received segment. It is also set when the TCP, UDP, or ICMP segment length does not match the payload length value in the IP Header field.
5	IPCSB	IP Checksum Bypassed When set, this bit indicates that the checksum offload engine is bypassed.

Table 586. Receive descriptor fields 4 (RDES4)

Bit	Symbol	Description
6	IPv4	IPv4 Packet Received When set, this bit indicates that the received packet is an IPv4 packet.
7	IPv6	IPv6 Packet Received When set, this bit indicates that the received packet is an IPv6 packet.
11:8	MT	Message Type These bits are encoded to give the type of the message received. • 0000: No PTP message received • 0001: SYNC (all clock types) • 0010: Follow_Up (all clock types) • 0011: Delay_Req (all clock types) • 0100: Delay_Resp (all clock types) • 0101: Pdelay_Req (in peer-to-peer transparent clock) • 0110: Pdelay_Resp (in peer-to-peer transparent clock) • 0111: Pdelay_Resp_Follow_Up (in peer-to-peer transparent clock) • 1000: Announce • 1001: Management • 1010: Signaling • 1011-1110: Reserved • 1111: PTP packet with Reserved message type These bits are valid only when you select the Advance Timestamp feature.

RDES6 and RDES7 contain the snapshot of the time-stamp. The availability of the snapshot of the time-stamp in RDES6 and RDES7 is indicated by bit-7 in the RDES0 descriptor. The contents of RDES6 and RDES7 are identified in Table 587 and Table 588.

Table 587. Receive descriptor fields 6 (RDES6)

Bit	Symbol	Description
31:0	RTSL	Receive Frame Timestamp Low This field is updated by DMA with the least significant 32 bits of the timestamp captured for the corresponding receive frame. This field is updated by DMA only for the last descriptor of the receive frame which is indicated by Last Descriptor status bit (RDES0[8]).

Table 588. Receive descriptor fields 7 (RDES7)

Bit	Symbol	Description
31:0	RTSH	Receive Frame Timestamp High This field is updated by DMA with the most significant 32 bits of the timestamp captured for the corresponding receive frame. This field is updated by DMA only for the last descriptor of the receive frame which is indicated by Last Descriptor status bit (RDES0[8]).

27.1 How to read this chapter

The LCD controller is available on part LPC4350.

27.2 Basic configuration

The LCD controller is configured as follows:

- See Table 589 for clocking and power control.
- The LCD is reset by the LCD_RST (reset # 16).
- The LCD interrupt is connected to interrupt slot # 7 in the NVIC.

Table 589. LCD clocking and power control

	Base clock	Branch clock	Operating frequency
LCD register interface clock	BASE_M4_CLK	CLK_M4_LCD	up to 204 MHz

27.3 Features

- AHB bus master interface to access frame buffer.
- Setup and control via a separate AHB slave interface.
- Dual 16-deep programmable 64-bit wide FIFOs for buffering incoming display data.
- Supports single and dual-panel monochrome Super Twisted Nematic (STN) displays with 4 or 8-bit interfaces.
- Supports single and dual-panel color STN displays.
- Supports Thin Film Transistor (TFT) color displays.
- Programmable display resolution including, but not limited to: 320x200, 320x240, 640x200, 640x240, 640x480, 800x600, and 1024x768.
- Hardware cursor support for single-panel displays.
- 15 gray-level monochrome, 3375 color STN, and 32K color palettized TFT support.
- 1, 2, or 4 bits-per-pixel (bpp) palettized displays for monochrome STN.
- 1, 2, 4, or 8 bpp palettized color displays for color STN and TFT.
- 16 bpp true-color non-palettized, for color STN and TFT.
- 24 bpp true-color non-palettized, for color TFT.
- Programmable timing for different display panels.
- 256 entry, 16-bit palette RAM, arranged as a 128x32-bit RAM.
- Frame, line, and pixel clock signals.
- AC bias signal for STN, data enable signal for TFT panels.
- Supports little and big-endian, and Windows CE data formats.
- LCD panel clock may be generated from the peripheral clock, or from a clock input pin.

27.4 General description

27.4.1 Programmable parameters

The following key display and controller parameters can be programmed:

- Horizontal front and back porch
- Horizontal synchronization pulse width
- Number of pixels per line
- Vertical front and back porch
- Vertical synchronization pulse width
- Number of lines per panel
- Number of pixel clocks per line
- Hardware cursor control.
- Signal polarity, active HIGH or LOW
- AC panel bias
- Panel clock frequency
- Bits-per-pixel
- Display type: STN monochrome, STN color, or TFT
- STN 4 or 8-bit interface mode
- STN dual or single panel mode
- Little-endian, big-endian, or Windows CE mode
- Interrupt generation event

27.4.2 Hardware cursor support

The hardware cursor feature reduces software overhead associated with maintaining a cursor image in the LCD frame buffer.

Without this feature, software needed to:

- Save an image of the area under the next cursor position.
- Update the area with the cursor image.
- Repair the last cursor position with a previously saved image.

In addition, the LCD driver had to check whether the graphics operation had overwritten the cursor, and correct it. With a cursor size of 64x64 and 24-bit color, each cursor move involved reading and writing approximately 75 kB of data.

The hardware cursor removes the requirement for this management by providing a completely separate image buffer for the cursor, and superimposing the cursor image on the LCD output stream at the current cursor (X,Y) coordinate.

To move the hardware cursor, the software driver supplies a new cursor coordinate. The frame buffer requires no modification. This significantly reduces software overhead.

The cursor image is held in the LCD controller in an internal 256x32-bit buffer memory.

27.4.3 Types of LCD panels supported

The LCD controller supports the following types of LCD panel:

- Active matrix TFT panels with up to 24-bit bus interface.
- Single-panel monochrome STN panels (4-bit and 8-bit bus interface).
- Dual-panel monochrome STN panels (4-bit and 8-bit bus interface per panel).
- Single-panel color STN panels, 8-bit bus interface.
- Dual-panel color STN panels, 8-bit bus interface per panel.

27.4.3.1 TFT panels

TFT panels support one or more of the following color modes:

- 1 bpp, palettized, 2 colors selected from available colors.
- 2 bpp, palettized, 4 colors selected from available colors.
- 4 bpp, palettized, 16 colors selected from available colors.
- 8 bpp, palettized, 256 colors selected from available colors.
- 12 bpp, direct 4:4:4 RGB.
- 16 bpp, direct 5:5:5 RGB, with 1 bpp not normally used. This pixel is still output, and can be used as a brightness bit to connect to the Least Significant Bit (LSB) of RGB components of a 6:6:6 TFT panel.
- 16 bpp, direct 5:6:5 RGB.
- 24 bpp, direct 8:8:8 RGB, providing over 16 million colors.

Each 16-bit palette entry is composed of 5 bpp (RGB), plus a common intensity bit. This provides better memory utilization and performance compared with a full 6 bpp structure. The total number of colors supported can be doubled from 32K to 64K if the intensity bit is used and applied to all three color components simultaneously.

Alternatively, the 16 signals can be used to drive a 5:6:5 panel with the extra bit only applied to the green channel.

27.4.3.2 Color STN panels

Color STN panels support one or more of the following color modes:

- 1 bpp, palettized, 2 colors selected from 3375.
- 2 bpp, palettized, 4 colors selected from 3375.
- 4 bpp, palettized, 16 colors selected from 3375.
- 8 bpp, palettized, 256 colors selected from 3375.
- 16 bpp, direct 4:4:4 RGB, with 4 bpp not being used.

27.4.3.3 Monochrome STN panels

Monochrome STN panels support one or more of the following modes:

- 1 bpp, palettized, 2 gray scales selected from 15.
- 2 bpp, palettized, 4 gray scales selected from 15.
- 4 bpp, palettized, 16 gray scales selected from 15.

More than 4 bpp for monochrome panels can be programmed, but using these modes has no benefit because the maximum number of gray scales supported on the display is 15.

27.5 Pin description

The largest configuration for the LCD controller uses 31 pins. There are many variants using as few as 10 pins for a monochrome STN panel. Pins are allocated in groups based on the selected configuration. All LCD functions are shared with other chip functions. In Table 590, only the LCD related portion of the pin name is shown.

Table 590. LCD controller pins

Pin name	Type	Function
LCDPWR	Output	LCD panel power enable.
LCDCLK	Output	LCD panel clock.
LCDENAB/LCDM (LCDAC)	Output	STN AC bias drive or TFT data enable output.
LCDFP	Output	Frame pulse (STN). Vertical synchronization pulse (TFT)
LCDLE	Output	Line end signal
LCDLP	Output	Line synchronization pulse (STN). Horizontal synchronization pulse (TFT)
LCDVD[23:0]	Output	LCD panel data. Bits used depend on the panel configuration.
GP_CLKIN	Input	General purpose CGU input clock. Can be used as the LCD external clock LCDCLKIN.

27.5.1 Signal usage

The signals that are used for various display types are identified in the following sections.

27.5.1.1 Signals used for single panel STN displays

The signals used for single panel STN displays are shown in Table 591. UD refers to upper panel data.

Table 591. Pins used for single panel STN displays

Pin name	4-bit Monochrome (10 pins)	8-bit Monochrome (14 pins)	Color (14 pins)
LCDPWR	Y	Y	Y
LCDDCLK	Y	Y	Y
LCDENAB/ LCDM	Y	Y	Y
LCDFP	Y	Y	Y
LCDLE	Y	Y	Y
LCDLP	Y	Y	Y
LCDVD[3:0]	UD[3:0]	UD[3:0]	UD[3:0]
LCDVD[7:4]	-	UD[7:4]	UD[7:4]
LCDVD[23:8]	-	-	-

27.5.1.2 Signals used for dual panel STN displays

The signals used for dual panel STN displays are shown in Table 592. UD refers to upper panel data, and LD refers to lower panel data.

Table 592. Pins used for dual panel STN displays

Pin name	4-bit Monochrome (14 pins)	8-bit Monochrome (22 pins)	Color (22 pins)
LCDPWR	Y	Y	Y
LCDDCLK	Y	Y	Y
LCDENAB/ LCDM	Y	Y	Y
LCDFP	Y	Y	Y
LCDLE	Y	Y	Y
LCDLP	Y	Y	Y
LCDVD[3:0]	UD[3:0]	UD[3:0]	UD[3:0]
LCDVD[7:4]	-	UD[7:4]	UD[7:4]
LCDVD[11:8]	LD[3:0]	LD[3:0]	LD[3:0]
LCDVD[15:12]	-	LD[7:4]	LD[7:4]
LCDVD[23:16]	-	-	-

27.5.1.3 Signals used for TFT displays

The signals used for TFT displays are shown in Table 593.

Table 593. Pins used for TFT displays

Pin name	12-bit, 4:4:4 mode (18 pins)	16-bit, 5:6:5 mode (22 pins)	16-bit, 1:5:5:5 mode (24 pins)	24-bit (30 pins)
LCDPWR	Y	Y	Y	Y
LCDDCLK	Y	Y	Y	Y
LCDENAB/ LCDM	Y	Y	Y	Y
LCDFP	Y	Y	Y	Y
LCDLE	Y	Y	Y	Y
LCDLP	Y	Y	Y	Y
LCDVD[1:0]	-	-	-	RED[1:0]
LCDVD[2]	-	-	Intensity	RED[2]
LCDVD[3]	-	RED[0]	RED[0]	RED[3]
LCDVD[7:4]	RED[3:0]	RED[4:1]	RED[4:1]	RED[7:4]
LCDVD[9:8]	-	-	-	GREEN[1:0]
LCDVD[10]	-	GREEN[0]	Intensity	GREEN[2]
LCDVD[11]	-	GREEN[1]	GREEN[0]	GREEN[3]
LCDVD[15:12]	GREEN[3:0]	GREEN[5:2]	GREEN[4:1]	GREEN[7:4]
LCDVD[17:16]	-	-	-	BLUE[1:0]
LCDVD[18]	-	-	Intensity	BLUE[2]
LCDVD[19]	-	BLUE[0]	BLUE[0]	BLUE[3]
LCDVD[23:20]	BLUE[3:0]	BLUE[4:1]	BLUE[4:1]	BLUE[7:4]

27.6 Register description

Table 594 shows the registers associated with the LCD controller and a summary of their functions. Following the table are details for each register.

Table 594. Register overview: LCD controller (base address: 0x4000 8000)

Name	Access	Address offset	Description	Reset value [1]	Reference
TIMH	R/W	0x000	Horizontal Timing Control register	0x0	Table 595
TIMV	R/W	0x004	Vertical Timing Control register	0x0	Table 596
POL	R/W	0x008	Clock and Signal Polarity Control register	0x0	Table 597
LE	R/W	0x00C	Line End Control register	0x0	Table 598
UPBASE	R/W	0x010	Upper Panel Frame Base Address register	0x0	Table 599
LPBASE	R/W	0x014	Lower Panel Frame Base Address register	0x0	Table 600
CTRL	R/W	0x018	LCD Control register	0x0	Table 601
INTMSK	R/W	0x01C	Interrupt Mask register	0x0	Table 602
INTRAW	RO	0x020	Raw Interrupt Status register	0x0	Table 603
INTSTAT	RO	0x024	Masked Interrupt Status register	0x0	Table 604
INTCLR	WO	0x028	Interrupt Clear register	0x0	Table 605
UPCURR	RO	0x02C	Upper Panel Current Address Value register	0x0	Table 606
LPCURR	RO	0x030	Lower Panel Current Address Value register	0x0	Table 607
-	-	0x034 to 0x1FC	Reserved	-	-
PAL	R/W	0x200 to 0x3FC	256x16-bit Color Palette registers	0x0	Table 608
-	-	0x400 to 0x7FC	Reserved	-	-
CRSR_IMG	R/W	0x800 to 0xBFC	Cursor Image registers	0x0	Table 609
CRSR_CTRL	R/W	0xC00	Cursor Control register	0x0	Table 610
CRSR_CFG	R/W	0xC04	Cursor Configuration register	0x0	Table 611
CRSR_PAL0	R/W	0xC08	Cursor Palette register 0	0x0	Table 612
CRSR_PAL1	R/W	0xC0C	Cursor Palette register 1	0x0	Table 613
CRSR_XY	R/W	0xC10	Cursor XY Position register	0x0	Table 614
CRSR_CLIP	R/W	0xC14	Cursor Clip Position register	0x0	Table 615
CRSR_INTMSK	R/W	0xC20	Cursor Interrupt Mask register	0x0	Table 616
CRSR_INTCLR	WO	0xC24	Cursor Interrupt Clear register	0x0	Table 617
CRSR_INTRAW	RO	0xC28	Cursor Raw Interrupt Status register	0x0	Table 618
CRSR_INTSTAT	RO	0xC2C	Cursor Masked Interrupt Status register	0x0	Table 619

[1] Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

27.6.1 Horizontal Timing register

The TIMH register controls the Horizontal Synchronization pulse Width (HSW), the Horizontal Front Porch (HFP) period, the Horizontal Back Porch (HBP) period, and the Pixels-Per-Line (PPL).

Table 595. Horizontal Timing register (TIMH, address 0x4000 8000) bit description

Bits	Symbol	Description	Reset value
1:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
7:2	PPL	Pixels-per-line. The PPL bit field specifies the number of pixels in each line or row of the screen. PPL is a 6-bit value that represents between 16 and 1024 pixels per line. PPL counts the number of pixel clocks that occur before the HFP is applied. Program the value required divided by 16, minus 1. Actual pixels-per-line = 16 * (PPL + 1). For example, to obtain 320 pixels per line, program PPL as (320/16) -1 = 19.	0x0
15:8	HSW	Horizontal synchronization pulse width. The 8-bit HSW field specifies the pulse width of the line clock in passive mode, or the horizontal synchronization pulse in active mode. Program with desired value minus 1.	0x0
23:16	HFP	Horizontal front porch. The 8-bit HFP field sets the number of pixel clock intervals at the end of each line or row of pixels, before the LCD line clock is pulsed. When a complete line of pixels is transmitted to the LCD driver, the value in HFP counts the number of pixel clocks to wait before asserting the line clock. HFP can generate a period of 1-256 pixel clock cycles. Program with desired value minus 1.	0x0
31:24	HBP	Horizontal back porch. The 8-bit HBP field is used to specify the number of pixel clock periods inserted at the beginning of each line or row of pixels. After the line clock for the previous line has been deasserted, the value in HBP counts the number of pixel clocks to wait before starting the next display line. HBP can generate a delay of 1-256 pixel clock cycles. Program with desired value minus 1.	0x0

27.6.1.1 Horizontal timing restrictions

DMA requests new data at the start of a horizontal display line. Some time must be allowed for the DMA transfer and for data to propagate down the FIFO path in the LCD interface. The data path latency forces some restrictions on the usable minimum values for horizontal porch width in STN mode. The minimum values are HSW = 2 and HBP = 2.

Single panel mode:

- HSW = 3 pixel clock cycles
- HBP = 5 pixel clock cycles
- HFP = 5 pixel clock cycles
- Panel Clock Divisor (PCD) = 1 (LCDCLK / 3)

Dual panel mode:

- HSW = 3 pixel clock cycles
- HBP = 5 pixel clock cycles
- HFP = 5 pixel clock cycles

- PCD = 5 (LCDCLK / 7)

If enough time is given at the start of the line, for example, setting HSW = 6, HBP = 10, data does not corrupt for PCD = 4, the minimum value.

27.6.2 Vertical Timing register

The TIMV register controls the Vertical Synchronization pulse Width (VSW), the Vertical Front Porch (VFP) period, the Vertical Back Porch (VBP) period, and the Lines-Per-Panel (LPP).

Table 596. Vertical Timing register (TIMV, address 0x4000 8004) bit description

Bits	Symbol	Description	Reset value
9:0	LPP	Lines per panel. This is the number of active lines per screen. The LPP field specifies the total number of lines or rows on the LCD panel being controlled. LPP is a 10-bit value allowing between 1 and 1024 lines. Program the register with the number of lines per LCD panel, minus 1. For dual panel displays, program the register with the number of lines on each of the upper and lower panels.	0x0
15:10	VSW	Vertical synchronization pulse width. This is the number of horizontal synchronization lines. The 6-bit VSW field specifies the pulse width of the vertical synchronization pulse. Program the register with the number of lines required, minus one. The number of horizontal synchronization lines must be small (for example, program to zero) for passive STN LCDs. The higher the value the worse the contrast on STN LCDs.	0x0
23:16	VFP	Vertical front porch. This is the number of inactive lines at the end of a frame, before the vertical synchronization period. The 8-bit VFP field specifies the number of line clocks to insert at the end of each frame. When a complete frame of pixels is transmitted to the LCD display, the value in VFP is used to count the number of line clock periods to wait. After the count has elapsed, the vertical synchronization signal, LCDFP, is asserted in active mode, or extra line clocks are inserted as specified by the VSW bit-field in passive mode. VFP generates 0–255 line clock cycles. Program to zero on passive displays for improved contrast.	0x0
31:24	VBP	Vertical back porch. This is the number of inactive lines at the start of a frame, after the vertical synchronization period. The 8-bit VBP field specifies the number of line clocks inserted at the beginning of each frame. The VBP count starts immediately after the vertical synchronization signal for the previous frame has been negated for active mode, or the extra line clocks have been inserted as specified by the VSW bit field in passive mode. After this has occurred, the count value in VBP sets the number of line clock periods inserted before the next frame. VBP generates 0–255 extra line clock cycles. Program to zero on passive displays for improved contrast.	0x0

27.6.3 Clock and Signal Polarity register

The POL register controls various details of clock timing and signal polarity.

Table 597. Clock and Signal Polarity register (POL, address 0x4000 8008) bit description

Bits	Symbol	Description	Reset value
4:0	PCD_LO	Lower five bits of panel clock divisor. The ten-bit PCD field, comprising PCD_HI (bits 31:27 of this register) and PCD_LO, is used to derive the LCD panel clock frequency LCDDCLK from the input clock, LCDDCLK = LCDCLK/(PCD+2). For monochrome STN displays with a 4 or 8-bit interface, the panel clock is a factor of four and eight down from the actual individual pixel clock rate. For color STN displays, 22/3 pixels are output per LCDDCLK cycle, so the panel clock is 0.375 times the pixel rate. For TFT displays, the pixel clock divider can be bypassed by setting the BCD bit in this register. **Note:** data path latency forces some restrictions on the usable minimum values for the panel clock divider in STN modes: Single panel color mode, PCD = 1 (LCDDCLK = LCDCLK/3). Dual panel color mode, PCD = 4 (LCDDCLK = LCDCLK/6). Single panel monochrome 4-bit interface mode, PCD = 2(LCDDCLK = LCDCLK/4). Dual panel monochrome 4-bit interface mode and single panel monochrome 8-bit interface mode, PCD = 6(LCDDCLK = LCDCLK/8). Dual panel monochrome 8-bit interface mode, PCD = 14(LCDDCLK = LCDCLK/16).	0x0
5	CLKSEL	Clock Select. This bit controls the selection of the source for LCDCLK. 0 = the clock source for the LCD block is CCLK. 1 = the clock source for the LCD block is LCDCLKIN (external clock input for the LVD).	0x0
10:6	ACB	AC bias pin frequency. The AC bias pin frequency is only applicable to STN displays. These require the pixel voltage polarity to periodically reverse to prevent damage caused by DC charge accumulation. Program this field with the required value minus one to apply the number of line clocks between each toggle of the AC bias pin, LCDENAB. This field has no effect if the LCD is operating in TFT mode, when the LCDENAB pin is used as a data enable signal.	0x0
11	IVS	Invert vertical synchronization. The IVS bit inverts the polarity of the LCDFP signal. 0 = LCDFP pin is active HIGH and inactive LOW. 1 = LCDFP pin is active LOW and inactive HIGH.	0x0
12	IHS	Invert horizontal synchronization. The IHS bit inverts the polarity of the LCDLP signal. 0 = LCDLP pin is active HIGH and inactive LOW. 1 = LCDLP pin is active LOW and inactive HIGH.	0x0

Table 597. Clock and Signal Polarity register (POL, address 0x4000 8008) bit description

Bits	Symbol	Description	Reset value
13	IPC	Invert panel clock. The IPC bit selects the edge of the panel clock on which pixel data is driven out onto the LCD data lines. 0 = Data is driven on the LCD data lines on the rising edge of LCDDCLK. 1 = Data is driven on the LCD data lines on the falling edge of LCDDCLK.	0x0
14	IOE	Invert output enable. This bit selects the active polarity of the output enable signal in TFT mode. In this mode, the LCDENAB pin is used as an enable that indicates to the LCD panel when valid display data is available. In active display mode, data is driven onto the LCD data lines at the programmed edge of LCDDCLK when LCDENAB is in its active state. 0 = LCDENAB output pin is active HIGH in TFT mode. 1 = LCDENAB output pin is active LOW in TFT mode.	0x0
15	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
25:16	CPL	Clocks per line. This field specifies the number of actual LCDDCLK clocks to the LCD panel on each line. This is the number of PPL divided by either 1 (for TFT), 4 or 8 (for monochrome passive), 2 2/3 (for color passive), minus one. This must be correctly programmed in addition to the PPL bit in the TIMH register for the LCD display to work correctly.	0x0
26	BCD	Bypass pixel clock divider. Setting this to 1 bypasses the pixel clock divider logic. This is mainly used for TFT displays.	0x0
31:27	PCD_HI	Upper five bits of panel clock divisor. See description for PCD_LO, in bits [4:0] of this register.	0x0

27.6.4 Line End Control register

The LE register controls the enabling of line-end signal LCDLE. When enabled, a positive pulse, four LCDCLK periods wide, is output on LCDLE after a programmable delay, LED, from the last pixel of each display line. If the line-end signal is disabled it is held permanently LOW.

Table 598. Line End Control register (LE, address 0x4000 800C) bit description

Bits	Symbol	Description	Reset value
6:0	LED	Line-end delay. Controls Line-end signal delay from the rising-edge of the last panel clock, LCDDCLK. Program with number of LCDCLK clock periods minus 1.	0x0
15:7	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
16	LEE	LCD Line end enable. 0 = LCDLE disabled (held LOW). 1 = LCDLE signal active.	0x0
31:17	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.5 Upper Panel Frame Base Address register

The UPBASE register is the color LCD upper panel DMA base address register, and is used to program the base address of the frame buffer for the upper panel. LCDUPBase (and LCDLPBase for dual panels) must be initialized before enabling the LCD controller. The base address must be doubleword aligned.

Optionally, the value may be changed mid-frame to create double-buffered video displays. These registers are copied to the corresponding current registers at each LCD vertical synchronization. This event causes the LNBU bit and an optional interrupt to be generated. The interrupt can be used to reprogram the base address when generating double-buffered video.

Table 599. Upper Panel Frame Base register (UPBASE, address 0x4000 8010) bit description

Bits	Symbol	Description	Reset value
2:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
31:3	LCDUPBASE	LCD upper panel base address. This is the start address of the upper panel frame data in memory and is doubleword aligned.	0x0

27.6.6 Lower Panel Frame Base Address register

The LPBASE register is the color LCD lower panel DMA base address register, and is used to program the base address of the frame buffer for the lower panel. LCDLPBase must be initialized before enabling the LCD controller. The base address must be doubleword aligned.

Optionally, the value may be changed mid-frame to create double-buffered video displays. These registers are copied to the corresponding current registers at each LCD vertical synchronization. This event causes the LNBU bit and an optional interrupt to be generated. The interrupt can be used to reprogram the base address when generating double-buffered video.

Table 600. Lower Panel Frame Base register (LPBASE, address 0x4000 8014) bit description

Bits	Symbol	Description	Reset value
2:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
31:3	LCDLPBASE	LCD lower panel base address. This is the start address of the lower panel frame data in memory and is doubleword aligned.	0x0

27.6.7 LCD Control register

The CTRL register controls the LCD operating mode and the panel pixel parameters.

Table 601. LCD Control register (CTRL, address 0x4000 8018) bit description

Bits	Symbol	Description	Reset value
0	LCDEN	LCD enable control bit. 0 = LCD disabled. Signals LCDLP, LCDDCLK, LCDFP, LCDENAB, and LCDLE are low. 1 = LCD enabled. Signals LCDLP, LCDDCLK, LCDFP, LCDENAB, and LCDLE are high. See LCD power-up and power-down sequence for details on LCD power sequencing.	0x0
3:1	LCDBPP	LCD bits per pixel: Selects the number of bits per LCD pixel: 000 = 1 bpp. 001 = 2 bpp. 010 = 4 bpp. 011 = 8 bpp. 100 = 16 bpp. 101 = 24 bpp (TFT panel only). 110 = 16 bpp, 5:6:5 mode. 111 = 12 bpp, 4:4:4 mode.	0x0
4	LCDBW	STN LCD monochrome/color selection. 0 = STN LCD is color. 1 = STN LCD is monochrome. This bit has no meaning in TFT mode.	0x0
5	LCDTFT	LCD panel TFT type selection. 0 = LCD is an STN display. Use gray scaler. 1 = LCD is a TFT display. Do not use gray scaler.	0x0
6	LCDMONO8	Monochrome LCD interface width. This bit controls whether a monochrome STN LCD uses a 4 or 8-bit parallel interface. It has no meaning in other modes and must be programmed to zero. 0 = monochrome LCD uses a 4-bit interface. 1 = monochrome LCD uses a 8-bit interface.	0x0

Table 601. LCD Control register (CTRL, address 0x4000 8018) bit description *…continued*

Bits	Symbol	Description	Reset value
7	LCDDUAL	Single or Dual LCD panel selection. STN LCD interface is: 0 = single-panel. 1 = dual-panel.	0x0
8	BGR	Color format selection. 0 = RGB: normal output. 1 = BGR: red and blue swapped.	0x0
9	BEBO	Big-endian Byte Order. Controls byte ordering in memory: 0 = little-endian byte order. 1 = big-endian byte order.	0x0
10	BEPO	Big-Endian Pixel Ordering. Controls pixel ordering within a byte: 0 = little-endian ordering within a byte. 1 = big-endian pixel ordering within a byte. The BEPO bit selects between little and big-endian pixel packing for 1, 2, and 4 bpp display modes, it has no effect on 8 or 16 bpp pixel formats. See Pixel serializer for more information on the data format.	0x0
11	LCDPWR	LCD power enable. 0 = power not gated through to LCD panel and LCDV[23:0] signals disabled, (held LOW). 1 = power gated through to LCD panel and LCDV[23:0] signals enabled, (active). See LCD power-up and power-down sequence for details on LCD power sequencing.	0x0
13:12	LCDVCOMP	LCD Vertical Compare Interrupt. Generate VComp interrupt at: 00 = start of vertical synchronization. 01 = start of back porch. 10 = start of active video. 11 = start of front porch.	0x0
15:14	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
16	WATERMARK	LCD DMA FIFO watermark level. Controls when DMA requests are generated: 0 = An LCD DMA request is generated when either of the DMA FIFOs have four or more empty locations. 1 = An LCD DMA request is generated when either of the DMA FIFOs have eight or more empty locations.	0x0
31:17	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.8 Interrupt Mask register

The INTMSK register controls whether various LCD interrupts occur.Setting bits in this register enables the corresponding raw interrupt INTRAW status bit values to be passed to the INTSTAT register for processing as interrupts.

Table 602. Interrupt Mask register (INTMSK, address 0x4000 801C) bit description

Bits	Function	Description	Reset value
0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
1	FUFIM	FIFO underflow interrupt enable. 0: The FIFO underflow interrupt is disabled. 1: Interrupt will be generated when the FIFO underflows.	0x0
2	LNBUIM	LCD next base address update interrupt enable. 0: The base address update interrupt is disabled. 1: Interrupt will be generated when the LCD base address registers have been updated from the next address registers.	0x0
3	VCOMPIM	Vertical compare interrupt enable. 0: The vertical compare time interrupt is disabled. 1: Interrupt will be generated when the vertical compare time (as defined by LcdVComp field in the CTRL register) is reached.	0x0
4	BERIM	AHB master error interrupt enable. 0: The AHB Master error interrupt is disabled. 1: Interrupt will be generated when an AHB Master error occurs.	0x0
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.9 Raw Interrupt Status register

The INTRAW register contains status flags for various LCD controller events. These flags can generate an interrupts if enabled by mask bits in the INTMSK register.

Table 603. Raw Interrupt Status register (INTRAW, address 0x4000 8020) bit description

Bits	Function	Description	Reset value
0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
1	FUFRIS	FIFO underflow raw interrupt status. Set when either the upper or lower DMA FIFOs have been read accessed when empty causing an underflow condition to occur. Generates an interrupt if the FUFIM bit in the INTMSK register is set.	
2	LNBURIS	LCD next address base update raw interrupt status. Mode dependent. Set when the current base address registers have been successfully updated by the next address registers. Signifies that a new next address can be loaded if double buffering is in use. Generates an interrupt if the LNBUIM bit in the INTMSK register is set.	0x0

Table 603. Raw Interrupt Status register (INTRAW, address 0x4000 8020) bit description

Bits	Function	Description	Reset value
3	VCOMPRIS	Vertical compare raw interrupt status. Set when one of the four vertical regions is reached, as selected by the LcdVComp bits in the CTRL register. Generates an interrupt if the VCompIM bit in the INTMSK register is set.	0x0
4	BERRAW	AHB master bus error raw interrupt status. Set when the AHB master interface receives a bus error response from a slave. Generates an interrupt if the BERIM bit in the INTMSK register is set.	0x0
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.10 Masked Interrupt Status register

The INTSTAT register is Read-Only, and contains a bit-by-bit logical AND of the INTRAW register and the INTMASK register. A logical OR of all interrupts is provided to the system interrupt controller.

Table 604. Masked Interrupt Status register (INTSTAT, address 0x4000 8024) bit description

Bits	Function	Description	Reset value
0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
1	FUFMIS	FIFO underflow masked interrupt status. Set when the both the FUFRIS bit in the INTRAW register and the FUFIM bit in the INTMSK register are set.	0x0
2	LNBUMIS	LCD next address base update masked interrupt status. Set when the both the LNBURIS bit in the INTRAW register and the LNBUIM bit in the INTMSK register are set.	0x0
3	VCOMPMIS	Vertical compare masked interrupt status. Set when the both the VCompRIS bit in the INTRAW register and the VCompIM bit in the INTMSK register are set.	0x0
4	BERMIS	AHB master bus error masked interrupt status. Set when the both the BERRAW bit in the INTRAW register and the BERIM bit in the INTMSK register are set.	0x0
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.11 Interrupt Clear register

The INTCLR register is Write-Only. Writing a logic 1 to the relevant bit clears the corresponding interrupt.

Table 605. Interrupt Clear register (INTCLR, address 0x4000 8028) bit description

Bits	Function	Description	Reset value
0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
1	FUFIC	FIFO underflow interrupt clear. Writing a 1 to this bit clears the FIFO underflow interrupt.	0x0
2	LNBUIC	LCD next address base update interrupt clear. Writing a 1 to this bit clears the LCD next address base update interrupt.	0x0
3	VCOMPIC	Vertical compare interrupt clear. Writing a 1 to this bit clears the vertical compare interrupt.	0x0
4	BERIC	AHB master error interrupt clear. Writing a 1 to this bit clears the AHB master error interrupt.	0x0
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.12 Upper Panel Current Address register

The UPCURR register is Read-Only, and contains an approximate value of the upper panel data DMA address when read.

Note: This register can change at any time and therefore can only be used as a rough indication of display position.

The contents of the UPCURR register are described in Table 606.

Table 606. Upper Panel Current Address register (UPCURR, address 0x4000 802C) bit description

Bits	Function	Description	Reset value
31:0	LCDUPCURR	LCD Upper Panel Current Address. Contains the current LCD upper panel data DMA address.	0x0

27.6.13 Lower Panel Current Address register

The LPCURR register is Read-Only, and contains an approximate value of the lower panel data DMA address when read.

Note: This register can change at any time and therefore can only be used as a rough indication of display position.

Table 607. Lower Panel Current Address register (LPCURR, address 0x4000 8030) bit description

Bits	Function	Description	Reset value
31:0	LCDLPCURR	LCD Lower Panel Current Address. Contains the current LCD lower panel data DMA address.	0x0

27.6.14 Color Palette registers

The PAL register contain 256 palette entries organized as 128 locations of two entries per word.

Each word location contains two palette entries. This means that 128 word locations are used for the palette. When configured for little-endian byte ordering, bits [15:0] are the lower numbered palette entry and [31:16] are the higher numbered palette entry. When configured for big-endian byte ordering this is reversed, because bits [31:16] are the low numbered palette entry and [15:0] are the high numbered entry.

Note: Only TFT displays use all of the palette entry bits.

The contents of the PAL register are described in Table 608.

Table 608. Color Palette registers (PAL, address 0x4000 8200 (PAL0) to 0x4000 83FC (PAL255)) bit description

Bits	Function	Description	Reset value
4:0	R04_0	Red palette data. For STN displays, only the four MSBs, bits [4:1], are used. For monochrome displays only the red palette data is used. All of the palette registers have the same bit fields.	0x0
9:5	G04_0	Green palette data.	0x0
14:10	B04_0	Blue palette data.	0x0
15	I0	Intensity / unused bit. Can be used as the LSB of the R, G, and B inputs to a 6:6:6 TFT display, doubling the number of colors to 64K, where each color has two different intensities.	0x0
20:16	R14_0	Red palette data. For STN displays, only the four MSBs, bits [4:1], are used. For monochrome displays only the red palette data is used. All of the palette registers have the same bit fields.	0x0
25:21	G14_0	Green palette data.	0x0
30:26	B14_0	Blue palette data.	0x0
31	I1	Intensity / unused bit. Can be used as the LSB of the R, G, and B inputs to a 6:6:6 TFT display, doubling the number of colors to 64K, where each color has two different intensities.	0x0

27.6.15 Cursor Image registers

The CRSR_IMG register area contains 256-word wide values which are used to define the image or images overlaid on the display by the hardware cursor mechanism. The image must always be stored in LBBP mode (little-endian byte, big-endian pixel) mode, as described in Section 27.7.5.6. Two bits are used to encode color and transparency for each pixel in the cursor.

Depending on the state of bit 0 in the CRSR_CFG register (see Cursor Configuration register description), the cursor image RAM contains either four 32x32 cursor images, or a single 64x64 cursor image.

The two colors defined for the cursor are mapped onto values from the CRSR_PAL0 and CRSR_PAL0 registers (see Cursor Palette register descriptions).

The contents of the CRSR_IMG register are described in Table 609.

Table 609. Cursor Image registers (CRSR_IMG, address 0x4000 8800 (CRSR_IMG0) to 0x4000 8BFC (CRSR_IMG1)) bit description

Bits	Function	Description	Reset value
31:0	CRSR_IMG	Cursor Image data. The 256 words of the cursor image registers define the appearance of either one 64x64 cursor or 4 32x32 cursors.	0x0

27.6.16 Cursor Control register

The CRSR_CTRL register provides access to frequently used cursor functions, such as the display on/off control for the cursor, and the cursor number.

If a 32x32 cursor is selected, one of four 32x32 cursors can be enabled. The images each occupy one quarter of the image memory, with Cursor0 from location 0, followed by Cursor1 from address 0x100, Cursor2 from 0x200 and Cursor3 from 0x300. If a 64x64 cursor is selected only one cursor fits in the image buffer, and no selection is possible.

Similar frame synchronization rules apply to the cursor number as apply to the cursor coordinates. If CrsrFramesync is 1, the displayed cursor image is only changed during the vertical frame blanking period. If CrsrFrameSync is 0, the cursor image index is changed immediately, even if the cursor is currently being scanned.

The contents of the CRSR_CTRL register are described in Table 610.

Table 610. Cursor Control register (CRSR_CTRL, address 0x4000 8C00) bit description

Bits	Function	Description	Reset value
0	CrsrOn	Cursor enable. 0 = Cursor is not displayed. 1 = Cursor is displayed.	0x0
3:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0x0
5:4	CRSRNUM1_0	Cursor image number. If the selected cursor size is 6x64, this field has no effect. If the selected cursor size is 32x32: 00 = Cursor0. 01 = Cursor1. 10 = Cursor2. 11 = Cursor3.	0x0
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0x0

27.6.17 Cursor Configuration register

The CRSR_CFG register provides overall configuration information for the hardware cursor.

The contents of the CRSR_CFG register are described in Table 611.

Table 611. Cursor Configuration register (CRSR_CFG, address 0x4000 8C04) bit description

Bits	Function	Description	Reset value
0	CrsrSize	Cursor size selection. 0 = 32x32 pixel cursor. Allows for 4 defined cursors. 1 = 64x64 pixel cursor.	0x0
1	FRAMESYNC	Cursor frame synchronization type. 0 = Cursor coordinates are asynchronous. 1 = Cursor coordinates are synchronized to the frame synchronization pulse.	0x0
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.18 Cursor Palette register 0

The cursor palette registers provide color palette information for the visible colors of the cursor. Color0 maps through CRSR_PAL0.

The register provides 24-bit RGB values that are displayed according to the abilities of the LCD panel in the same way as the frame-buffers palette output is displayed.

In monochrome STN mode, only the upper 4 bits of the Red field are used. In STN color mode, the upper 4 bits of the Red, Blue, and Green fields are used. In 24 bits per pixel mode, all 24 bits of the palette registers are significant.

The contents of the CRSR_PAL0 register are described in Table 612.

Table 612. Cursor Palette register 0 (CRSR_PAL0, address 0x4000 8C08) bit description

Bits	Function	Description	Reset value
7:0	RED	Red color component	0x0
15:8	GREEN	Green color component	0x0
23:16	BLUE	Blue color component.	0x0
31:24	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.19 Cursor Palette register 1

The cursor palette registers provide color palette information for the visible colors of the cursor. Color1 maps through CRSR_PAL1.

The register provides 24-bit RGB values that are displayed according to the abilities of the LCD panel in the same way as the frame-buffers palette output is displayed.

In monochrome STN mode, only the upper 4 bits of the Red field are used. In STN color mode, the upper 4 bits of the Red, Blue, and Green fields are used. In 24 bits per pixel mode, all 24 bits of the palette registers are significant.

The contents of the CRSR_PAL1 register are described in Table 613.

Table 613. Cursor Palette register 1 (CRSR_PAL1, address 0x4000 8C0C) bit description

Bits	Function	Description	Reset value
7:0	RED	Red color component	0x0
15:8	GREEN	Green color component	0x0
23:16	BLUE	Blue color component.	0x0
31:24	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.20 Cursor XY Position register

The CRSR_XY register defines the distance of the top-left edge of the cursor from the top-left side of the cursor overlay. Refer to the section on Cursor Clipping for more details.

If the FrameSync bit in the CRSR_CFG register is 0, the cursor position changes immediately, even if the cursor is currently being scanned. If Framesync is 1, the cursor position is only changed during the next vertical frame blanking period.

The contents of the CRSR_XY register are described in Table 614.

Table 614. Cursor XY Position register (CRSR_XY, address 0x4000 8C10) bit description

Bits	Function	Description	Reset value
9:0	CRSRX	X ordinate of the cursor origin measured in pixels. When 0, the left edge of the cursor is at the left of the display.	0x0
15:10	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
25:16	CRSRY	Y ordinate of the cursor origin measured in pixels. When 0, the top edge of the cursor is at the top of the display.	0x0
31:26	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.21 Cursor Clip Position register

The CRSR_CLIP register defines the distance from the top-left edge of the cursor image, to the first displayed pixel in the cursor image.

Different synchronization rules apply to the Cursor Clip registers than apply to the cursor coordinates. If the FrameSync bit in the CRSR_CFG register is 0, the cursor clip point is changed immediately, even if the cursor is currently being scanned.

If the Framesync bit in the CRSR_CFG register is 1, the displayed cursor image is only changed during the vertical frame blanking period, providing that the cursor position has been updated since the Clip register was programmed. When programming, the Clip register must be written before the Position register (ClcdCrsrXY) to ensure that in a given frame, the clip and position information is coherent.

The contents of the CRSR_CLIP register are described in Table 615.

Table 615. Cursor Clip Position register (CRSR_CLIP, address 0x4000 8C14) bit description

Bits	Function	Description	Reset value
5:0	CRSRCLIPX	Cursor clip position for X direction. Distance from the left edge of the cursor image to the first displayed pixel in the cursor. When 0, the first pixel of the cursor line is displayed.	0x0
7:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
13:8	CRSRCLIPY	Cursor clip position for Y direction. Distance from the top of the cursor image to the first displayed pixel in the cursor. When 0, the first displayed pixel is from the top line of the cursor image.	0x0
31:14	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.22 Cursor Interrupt Mask register

The CRSR_INTMSK register is used to enable or disable the cursor from interrupting the processor.

The contents of the CRSR_INTMSK register are described in Table 616.

Table 616. Cursor Interrupt Mask register (CRSR_INTMSK, address 0x4000 8C20) bit description

Bits	Function	Description	Reset value
0	CRSRIM	Cursor interrupt mask. When clear, the cursor never interrupts the processor. When set, the cursor interrupts the processor immediately after reading of the last word of cursor image.	0x0
31:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.23 Cursor Interrupt Clear register

The CRSR_INTCLR register is used by software to clear the cursor interrupt status and the cursor interrupt signal to the processor.

The contents of the CRSR_INTCLR register are described in Table 617.

Table 617. Cursor Interrupt Clear register (CRSR_INTCLR, address 0x4000 8C24) bit description

Bits	Function	Description	Reset value
0	CRSRIC	Cursor interrupt clear. Writing a 0 to this bit has no effect. Writing a 1 to this bit causes the cursor interrupt status to be cleared.	0x0
31:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.24 Cursor Raw Interrupt Status register

The CRSR_INTRAW register is set to indicate a cursor interrupt. When enabled via the CrsrIM bit in the CRSR_INTMSK register, provides the interrupt to the system interrupt controller.

The contents of the CRSR_INTRAW register are described in Table 618.

Table 618. Cursor Raw Interrupt Status register (CRSR_INTRAW, address 0x4000 8C28) bit description

Bits	Function	Description	Reset value
0	CRSRRIS	Cursor raw interrupt status. The cursor interrupt status is set immediately after the last data is read from the cursor image for the current frame. This bit is cleared by writing to the CrsrIC bit in the CRSR_INTCLR register.	0x0
31:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.6.25 Cursor Masked Interrupt Status register

The CRSR_INTSTAT register is set to indicate a cursor interrupt providing that the interrupt is not masked in the CRSR_INTMSK register.

The contents of the CRSR_INTSTAT register are described in Table 619.

Table 619. Cursor Masked Interrupt Status register (CRSR_INTSTAT, address 0x4000 8C2C) bit description

Bits	Function	Description	Reset value
0	CRSRMIS	Cursor masked interrupt status. The cursor interrupt status is set immediately after the last data read from the cursor image for the current frame, providing that the corresponding bit in the CRSR_INTMSK register is set. The bit remains clear if the CRSR_INTMSK register is clear. This bit is cleared by writing to the CRSR_INTCLR register.	0x0
31:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

27.7 LCD controller functional description

The LCD controller performs translation of pixel-coded data into the required formats and timings to drive a variety of single or dual panel monochrome and color LCDs.

Packets of pixel coded data are fed using the AHB interface, to two independent, programmable, 32-bit wide, DMA FIFOs that act as input data flow buffers.

The buffered pixel coded data is then unpacked using a pixel serializer.

Depending on the LCD type and mode, the unpacked data can represent:

- An actual true display gray or color value.
- An address to a 256x16 bit wide palette RAM gray or color value.

In the case of STN displays, either a value obtained from the addressed palette location, or the true value is passed to the gray scaling generators. The hardware-coded gray scale algorithm logic sequences the activity of the addressed pixels over a programmed number of frames to provide the effective display appearance.

For TFT displays, either an addressed palette value or true color value is passed directly to the output display drivers, bypassing the gray scaling algorithmic logic.

In addition to data formatting, the LCD controller provides a set of programmable display control signals, including:

- LCD panel power enable
- Pixel clock
- Horizontal and vertical synchronization pulses
- Display bias

The LCD controller generates individual interrupts for:

- Upper or lower panel DMA FIFO underflow
- Base address update signification
- Vertical compare
- Bus error

There is also a single combined interrupt that is asserted when any of the individual interrupts become active.

Figure 73 shows a simplified block diagram of the LCD controller.

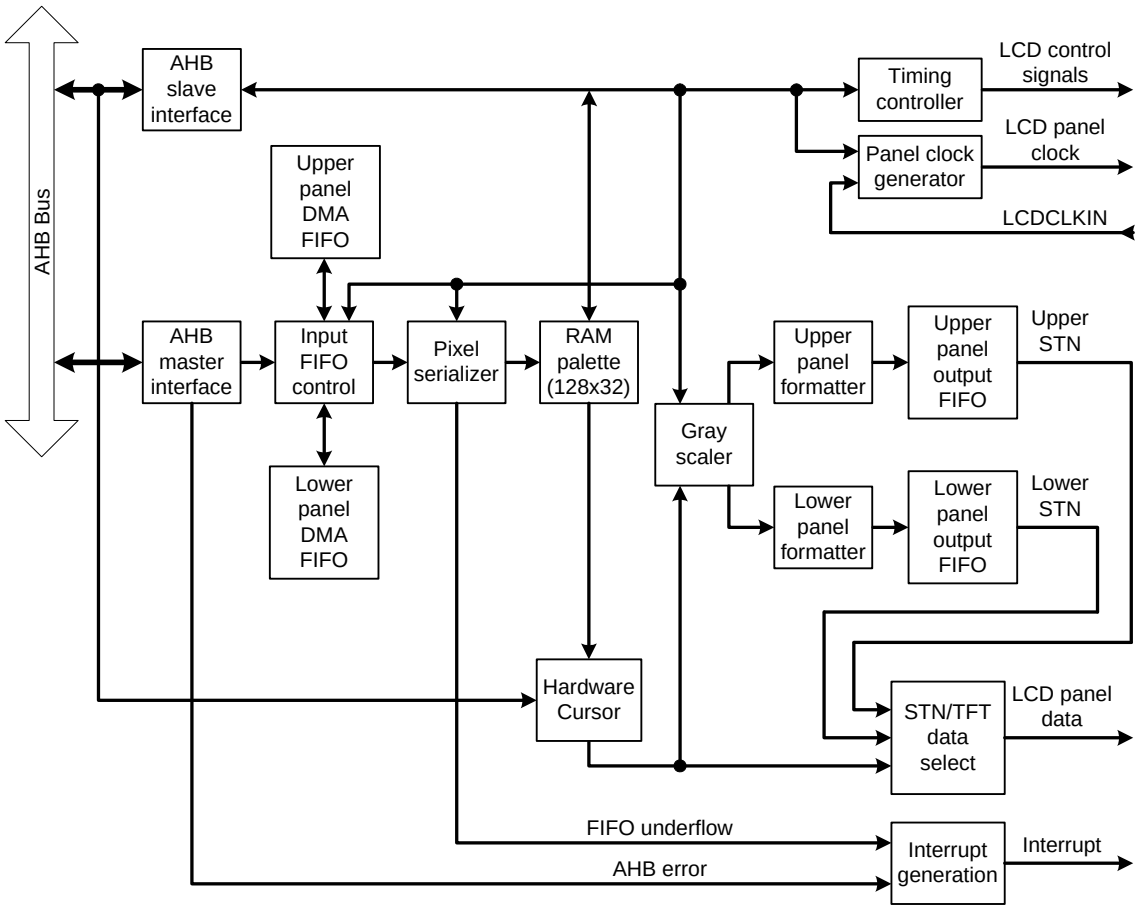


Fig 73. LCD controller block diagram

27.7.1 AHB interfaces

The LCD controller includes two separate AHB interfaces. The first, an AHB slave interface, is used primarily by the CPU to access control and data registers within the LCD controller. The second, an AHB master interface, is used by the LCD controller for DMA access to display data stored in memory elsewhere in the system. The LCD DMA controller can access any SRAM on AHB and the external memory.

27.7.1.1 AMBA AHB slave interface

The AHB slave interface connects the LCD controller to the AHB bus and provides CPU accesses to the registers and palette RAM.

27.7.1.2 AMBA AHB master interface

The AHB master interface transfers display data from a selected slave (memory) to the LCD controller DMA FIFOs. It can be configured to obtain data from any on-chip SRAM on AHB, various types of off-chip static memory, or off-chip SDRAM.

In dual panel mode, the DMA FIFOs are filled up in an alternating fashion via a single DMA request. In single panel mode, the DMA FIFOs are filled up in a sequential fashion from a single DMA request.

The inherent AHB master interface state machine performs the following functions:

- Loads the upper panel base address into the AHB address incrementer on recognition of a new frame.
- Monitors both the upper and lower DMA FIFO levels and asserts a DMA request to request display data from memory, filling them to above the programmed watermark. the DMA request is reasserted when there are at least four locations available in either FIFO (dual panel mode).
- Checks for 1 kB boundaries during fixed-length bursts, appropriately adjusting the address in such occurrences.
- Generates the address sequences for fixed-length and undefined bursts.
- Controls the handshaking between the memory and DMA FIFOs. It inserts busy cycles if the FIFOs have not completed their synchronization and updating sequence.
- Fills up the DMA FIFOs, in dual panel mode, in an alternating fashion from a single DMA request.
- Asserts the a bus error interrupt if an error occurs during an active burst.
- Responds to retry commands by restarting the failed access. This introduces some busy cycles while it re-synchronizes.

27.7.2 Dual DMA FIFOs and associated control logic

The pixel data accessed from memory is buffered by two DMA FIFOs that can be independently controlled to cover single and dual-panel LCD types. Each FIFO is 16 words deep by 64 bits wide and can be cascaded to form an effective 32-Dword deep FIFO in single panel mode.

Synchronization logic transfers the pixel data from the AHB clock domain to the LCD controller clock domain. The water level marks in each FIFO are set such that each FIFO requests data when at least four locations become available.

An interrupt signal is asserted if an attempt is made to read either of the two DMA FIFOs when they are empty (an underflow condition has occurred).

27.7.3 Pixel serializer

This block reads the 32-bit wide LCD data from the output port of the DMA FIFO and extracts 24, 16, 8, 4, 2, or 1 bpp data, depending on the current mode of operation. The LCD controller supports big-endian, little-endian, and Windows CE data formats.

Depending on the mode of operation, the extracted data can be used to point to a color or gray scale value in the palette RAM or can actually be a true color value that can be directly applied to an LCD panel input.

Table 620 through Table 622 show the structure of the data in each DMA FIFO word corresponding to the endianness and bpp combinations. For each of the three supported data formats, the required data for each panel display pixel must be extracted from the data word.

Table 620. FIFO bits for Little-endian Byte, Little-endian Pixel order

FIFO bit	1 bpp	2 bpp	4 bpp	8 bpp	16 bpp	24 bpp
31	p31	p15	p7	p3	p1	p0
30	p30					
29	p29	p14				
28	p28					
27	p27	p13	p6			
26	p26					
25	p25	p12				
24	p24					
23	p23	p11	p5	p2		
22	p22					
21	p21	p10				
20	p20					
19	p19	p9	p4			
18	p18					
17	p17	p8				
16	p16					
15	p15	p7	p3	p1	p0	
14	p14					
13	p13	p6				
12	p12					
11	p11	p5	p2			
10	p10					
9	p9	p4				
8	p8					
7	p7	p3	p1	p0		
6	p6					
5	p5	p2				
4	p4					
3	p3	p1	p0			
2	p2					
1	p1	p0				
0	p0					

Table 621. FIFO bits for Big-endian Byte, Big-endian Pixel order

FIFO bit	1 bpp	2 bpp	4 bpp	8 bpp	16 bpp	24 bpp
31	p0	p0	p0	p0	p0	p0
30	p1					
29	p2	p1				
28	p3					
27	p4	p2	p1			
26	p5					
25	p6	p3				
24	p7					
23	p8	p4	p2	p1		
22	p9					
21	p10	p5				
20	p11					
19	p12	p6	p3			
18	p13					
17	p14	p7				
16	p15					
15	p16	p8	p4	p2	p1	
14	p17					
13	p18	p9				
12	p19					
11	p20	p10	p5			
10	p21					
9	p22	p11				
8	p23					
7	p24	p12	p6	p3		
6	p25					
5	p26	p13				
4	p27					
3	p28	p14	p7			
2	p29					
1	p30	p15				
0	p31					

Table 622. FIFO bits for Little-endian Byte, Big-endian Pixel order

FIFO bit	1 bpp	2 bpp	4 bpp	8 bpp	16 bpp	24 bpp
31	p24	p12	p6	p3	p1	
30	p25					
29	p26	p13				
28	p27					
27	p28	p14	p7			
26	p29					
25	p30	p15				
24	p31					
23	p16	p8	p4	p2		p0
22	p17					
21	p18	p9				
20	p19					
19	p20	p10	p5			
18	p21					
17	p22	p11				
16	p23					
15	p8	p4	p2	p1	p0	
14	p9					
13	p10	p5				
12	p11					
11	p12	p6	p3			
10	p13					
9	p14	p7				
8	p15					
7	p0	p0	p0	p0		
6	p1					
5	p2	p1				
4	p3					
3	p4	p2	p1			
2	p5					
1	p6	p3				
0	p7					

Table 623 shows the structure of the data in each DMA FIFO word in RGB mode.

Table 623. RGB mode data formats

FIFO data	24-bit RGB	16-bit (1:5:5:5 RGB)	16-bit (5:6:5 RGB)	16-bit (4:4:4 RGB)
31	-	p1 intensity bit	p1, Blue 4	-
30	-	p1, Blue 4	p1, Blue 3	-
29	-	p1, Blue 3	p1, Blue 2	-
28	-	p1, Blue 2	p1, Blue 1	-
27	-	p1, Blue 1	p1, Blue 0	p1, Blue 3
26	-	p1, Blue 0	p1, Green 5	p1, Blue 2
25	-	p1, Green 4	p1, Green 4	p1, Blue 1
24	-	p1, Green 3	p1, Green 3	p1, Blue 0
23	p0, Blue 7	p1, Green 2	p1, Green 2	p1, Green 3
22	p0, Blue 6	p1, Green 1	p1, Green 1	p1, Green 2
21	p0, Blue 5	p1, Green 0	p1, Green 0	p1, Green 1
20	p0, Blue 4	p1, Red 4	p1, Red 4	p1, Green 0
19	p0, Blue 3	p1, Red 3	p1, Red 3	p1, Red 3
18	p0, Blue 2	p1, Red 2	p1, Red 2	p1, Red 2
17	p0, Blue 1	p1, Red 1	p1, Red 1	p1, Red 1
16	p0, Blue 0	p1, Red 0	p1, Red 0	p1, Red 0
15	p0, Green 7	p0 intensity bit	p0, Blue 4	-
14	p0, Green 6	p0, Blue 4	p0, Blue 3	-
13	p0, Green 5	p0, Blue 3	p0, Blue 2	-
12	p0, Green 4	p0, Blue 2	p0, Blue 1	-
11	p0, Green 3	p0, Blue 1	p0, Blue 0	p0, Blue 3
10	p0, Green 2	p0, Blue 0	p0, Green 5	p0, Blue 2
9	p0, Green 1	p0, Green 4	p0, Green 4	p0, Blue 1
8	p0, Green 0	p0, Green 3	p0, Green 3	p0, Blue 0
7	p0, Red 7	p0, Green 2	p0, Green 2	p0, Green 3
6	p0, Red 6	p0, Green 1	p0, Green 1	p0, Green 2
5	p0, Red 5	p0, Green 0	p0, Green 0	p0, Green 1
4	p0, Red 4	p0, Red 4	p0, Red 4	p0, Green 0
3	p0, Red 3	p0, Red 3	p0, Red 3	p0, Red 3
2	p0, Red 2	p0, Red 2	p0, Red 2	p0, Red 2
1	p0, Red 1	p0, Red 1	p0, Red 1	p0, Red 1
0	p0, Red 0	p0, Red 0	p0, Red 0	p0, Red 0

27.7.4 RAM palette

The RAM-based palette is a 256 x 16 bit dual-port RAM physically structured as 128 x 32 bits. Two entries can be written into the palette from a single word write access. The Least Significant Bit (LSB) of the serialized pixel data selects between upper and lower halves of the palette RAM. The half that is selected depends on the byte ordering mode. In little-endian mode, setting the LSB selects the upper half, but in big-endian mode, the lower half of the palette is selected.

Pixel data values can be written and verified through the AHB slave interface. For information on the supported colors, refer to the section on the related panel type earlier in this chapter.

The palette RAM is a dual port RAM with independent controls and addresses for each port. Port1 is used as a read/write port and is connected to the AHB slave interface. The palette entries can be written and verified through this port. Port2 is used as a read-only port and is connected to the unpacker and gray scaler. For color modes of less than 16 bpp, the palette enables each pixel value to be mapped to a 16-bit color:

- For TFT displays, the 16-bit value is passed directly to the pixel serializer.
- For STN displays, the 16-bit value is first converted by the gray scaler.

Table 624 shows the bit representation of the palette data. The palette 16-bit output uses the TFT 1:5:5:5 data format. In 16 and 24 bpp TFT mode, the palette is bypassed and the output of the pixel serializer is used as the TFT panel data.

Table 624. Palette data storage for TFT modes.

Bit(s)	Name (RGB format)	Description (RGB format)	Name (BGR format)	Description (BGR format)
31	I	Intensity / unused	I	Intensity / unused
30:26	B[4:0]	Blue palette data	R[4:0]	Red palette data
25:21	G[4:0]	Green palette data	G[4:0]	Green palette data
20:16	R[4:0]	Red palette data	B[4:0]	Blue palette data
15	I	Intensity / unused	I	Intensity / unused
14:10	B[4:0]	Blue palette data	R[4:0]	Red palette data
9:5	G[4:0]	Green palette data	G[4:0]	Green palette data
4:0	R[4:0]	Red palette data	B[4:0]	Blue palette data

The red and blue pixel data can be swapped to support BGR data format using a control register bit (bit 8 = BGR). See the CTRL register description for more information. Table 625 shows the bit representation of the palette data for the STN color modes.

Table 625. Palette data storage for STN color modes.

Bit(s)	Name (RGB format)	Description (RGB format)	Name (BGR format)	Description (BGR format)
31	-	Unused	-	Unused
30:27	B[3:0]	Blue palette data	R[3:0]	Red palette data
26	-	Unused	-	Unused
25:22	G[3:0]	Green palette data	G[3:0]	Green palette data
21	-	Unused	-	Unused
20:17	R[3:0]	Red palette data	B[3:0]	Blue palette data
16	-	Unused	-	Unused
15	I	Unused	I	Unused
14:11	B[4:1]	Blue palette data	R[4:1]	Red palette data
10	B[0]	Unused	R[0]	Unused
9:6	G[4:1]	Green palette data	G[4:1]	Green palette data

Table 625. Palette data storage for STN color modes.

Bit(s)	Name (RGB format)	Description (RGB format)	Name (BGR format)	Description (BGR format)
5	G[0]	Unused	G[0]	Unused
4:1	R[4:1]	Red palette data	B[4:1]	Blue palette data
0	R[0]	Unused	B[0]	Unused

For monochrome STN mode, only the red palette field bits [4:1] are used. However, in STN color mode the green and blue [4:1] are also used. Only 4 bits per color are used, because the gray scaler only supports 16 different shades per color.

Table 626 shows the bit representation of the palette data for the STN monochrome mode.

Table 626. Palette data storage for STN monochrome mode.

Bit(s)	Name	Description
31	-	Unused
30:27	-	Unused
26	-	Unused
25:22	-	Unused
21	-	Unused
20:17	Y[3:0]	Intensity data
16	-	Unused
15	-	Unused
14:11	-	Unused
10	-	Unused
9:6	-	Unused
5	-	Unused
4:1	Y[3:0]	Intensity data
0	-	Unused

27.7.5 Hardware cursor

The hardware cursor is an integral part of the LCD controller. It uses the LCD timing module to provide an indication of the current scan position coordinate, and intercepts the pixel stream between the palette logic and the gray scale/output multiplexer.

All cursor programming registers are accessed through the LCD slave interface. This also provides a read/write port to the cursor image RAM.

27.7.5.1 Cursor operation

The hardware cursor is contained in a dual port RAM. It is programmed by software through the AHB slave interface. The AHB slave interface also provides access to the hardware cursor control registers. These registers enable you to modify the cursor position and perform various other functions.

When enabled, the hardware cursor uses the horizontal and vertical synchronization signals, along with a pixel clock enable and various display parameters to calculate the current scan coordinate.

When the display point is inside the bounds of the cursor image, the cursor replaces frame buffer pixels with cursor pixels.

When the last cursor pixel is displayed, an interrupt is generated that software can use as an indication that it is safe to modify the cursor image. This enables software controlled animations to be performed without flickering for frame synchronized cursors.

27.7.5.2 Cursor sizes

Two cursor sizes are supported, as shown in Table 627.

Table 627. Palette data storage for STN monochrome mode.

X Pixels	Y Pixels	Bits per pixel	Words per line	Words in cursor image
32	32	2	2	64
64	64	2	4	256

27.7.5.3 Cursor movement

The following descriptions assume that both the screen and cursor origins are at the top left of the visible screen (the first visible pixel scanned each frame). Figure 74 shows how each pixel coordinate is assumed to be the top left corner of the pixel.

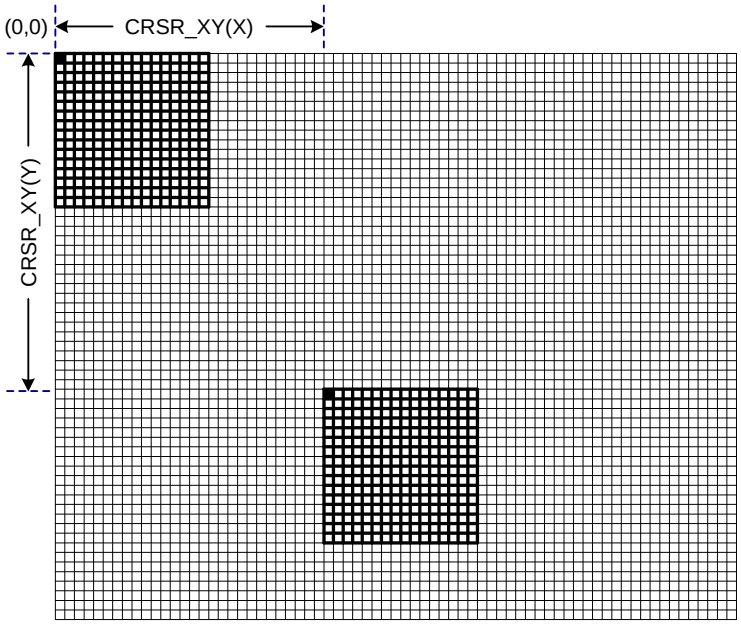


Fig 74. Cursor movement

27.7.5.4 Cursor XY positioning

The CRSR_XY register controls the cursor position on the cursor overlay (see Cursor XY Position register). This provides separate fields for X and Y ordinates.

The CRSR_CFG register (see Cursor Configuration register) provides a FrameSync bit controlling the visible behavior of the cursor.

With FrameSync inactive, the cursor responds immediately to any change in the programmed CRSR_XY value. Some transient smearing effects may be visible if the cursor is moved across the LCD scan line.

With FrameSync active, the cursor only updates its position after a vertical synchronization has occurred. This provides clean cursor movement, but the cursor position only updates once a frame.

27.7.5.5 Cursor clipping

The CRSR_XY register (see Cursor XY Position register) is programmed with positive binary values that enable the cursor image to be located anywhere on the visible screen image. The cursor image is clipped automatically at the screen limits when it extends beyond the screen image to the right or bottom (see X1,Y1 in Figure 75). The checked pattern shows the visible portion of the cursor.

Because the CRSR_XY register values are positive integers, to emulate cursor clipping on the left and top of screen, a Clip Position register, CRSR_CLIP, is provided. This controls which point of the cursor image is positioned at the CRSR_CLIP coordinate. For clipping functions on the Y axis, CRSR_XY(X) is zero, and Clip(X) is programmed to provide the offset into the cursor image (X2 and X3). The equivalent function is provided to clip on the X axis at the top of the display (Y2).

For cursors that are not clipped at the X=0 or Y=0 lines, program the Clip Position register X and Y fields with zero to display the cursor correctly. See Clip(X4,Y4) for the effect of incorrect programming.

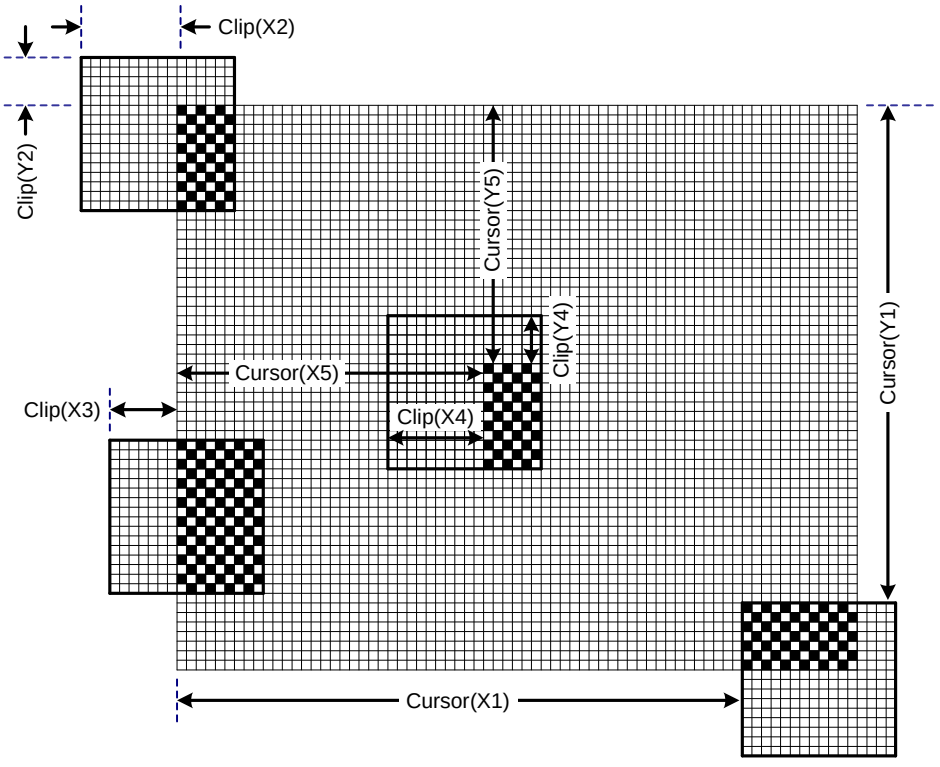


Fig 75. Cursor clipping

27.7.5.6 Cursor image format

The LCD frame buffer supports three packing formats, but the hardware cursor image requirement has been simplified to support only LBBP. This is little-endian byte, big-endian pixel for Windows CE mode.

The Image RAM start address is offset by 0x800 from the LCD base address, as shown in the register description in this chapter.

The displayed cursor coordinate system is expressed in terms of (X,Y). 64 x 64 is an extension of the 32 x 32 format shown in Figure 76.

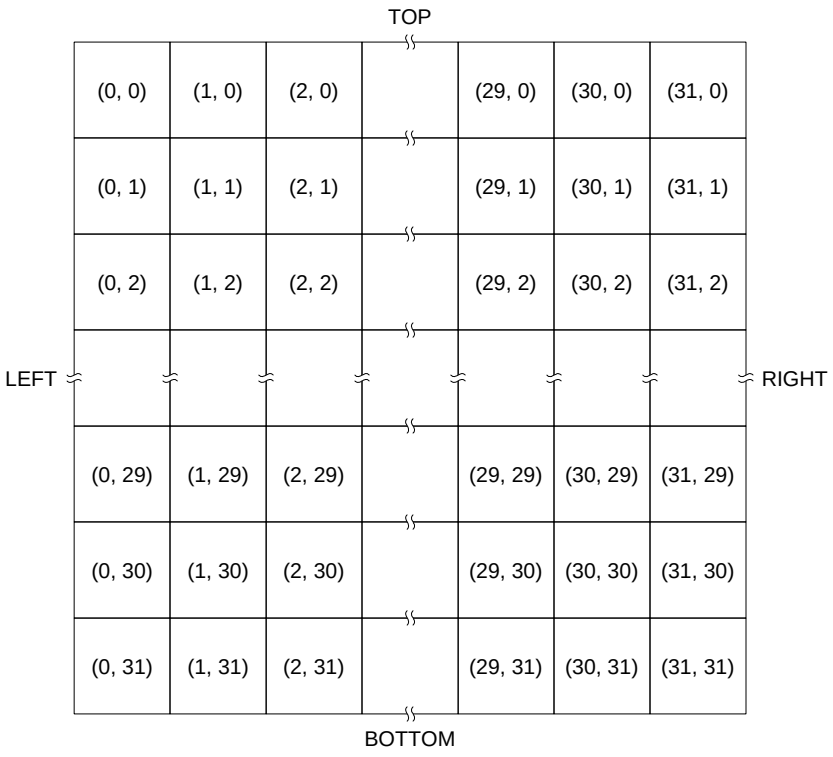


Fig 76. Cursor image format

32 by 32 pixel format

Four cursors are held in memory, each with the same pixel format. Table 628 lists the base addresses for the four cursors.

Table 628. Addresses for 32 x 32 cursors

Address	Description
0x4000 8800	Cursor 0 start address.
0x4000 8900	Cursor 1 start address.
0x4000 8A00	Cursor 2 start address.
0x4000 8B00	Cursor 3 start address.

Table 629 shows the buffer to pixel mapping for Cursor 0.

Table 629. Buffer to pixel mapping for 32 x 32 pixel cursor format

	Offset into cursor memory					
Data bits	0	4	(8 * y)	(8 * y) +4	F8	FC
31:30	(12, 0)	(28, 0)	(12, y)	(28, y)	(12, 31)	(28,31)
29:28	(13, 0)	(29, 0)	(13, y)	(29, y)	(13, 31)	(29, 31)
27:26	(14, 0)	(30, 0)	(14, y)	(30, y)	(14, 31)	(30, 31)
25:24	(15, 0)	(31, 0)	(15, y)	(31, y)	(15, 31)	(31, 31)
23:22	(8, 0)	(24, 0)	(8, y)	(24, y)	(8, 31)	(24, 31)
21:20	(9, 0)	(25, 0)	(9, y)	(25, y)	(9, 31)	(25, 31)
19:18	(10, 0)	(26, 0)	(10, y)	(26, y)	(10, 31)	(26, 31)
17:16	(11, 0)	(27, 0)	(11, y)	(27, y)	(11, 31)	(27, 31)
15:14	(4, 0)	(20, 0)	(4, y)	(20, y)	(4, 31)	(20, 31)
13:12	(5, 0)	(21, 0)	(5, y)	(21, y)	(5, 31)	(21, 31)
11:10	(6, 0)	(22, 0)	(6, y)	(22, y)	(6, 31)	(22, 31)
9:8	(7, 0)	(23, 0)	(7, y)	(23, y)	(7, 31)	(23, 31)
7:6	(0, 0)	(16, 0)	(0, y)	(16, y)	(0, 31)	(16, 31)
5:4	(1, 0)	(17, 0)	(1, y)	(17, y)	(1, 31)	(17, 31)
3:2	(2, 0)	(18, 0)	(2, y)	(18, y)	(2, 31)	(18, 31)
1:0	(3, 0)	(19, 0)	(3, y)	(19, y)	(3, 31)	(19, 31)

64 by 64 pixel format

Only one cursor fits in the memory space in 64 x 64 mode. Table 630 shows the 64 x 64 cursor format.

Table 630. Buffer to pixel mapping for 64 x 64 pixel cursor format

	Offset into cursor memory								
Data bits	0	4	8	12	(16 * y)	(16 * y) +4	(16 * y) + 8	(16 * y) + 12	FC
31:30	(12, 0)	(28, 0)	(44, 0)	(60, 0)	(12, y)	(28, y)	(44, y)	(60, y)	(60, 63)
29:28	(13, 0)	(29, 0)	(45, 0)	(61, 0)	(13, y)	(29, y)	(45, y)	(61, y)	(61, 63)
27:26	(14, 0)	(30, 0)	(46, 0)	(62, 0)	(14, y)	(30, y)	(46, y)	(62, y)	(62, 63)
25:24	(15, 0)	(31, 0)	(47, 0)	(63, 0)	(15, y)	(31, y)	(47, y)	(63, y)	(63, 63)
23:22	(8, 0)	(24, 0)	(40, 0)	(56, 0)	(8, y)	(24, y)	(40, y)	(56, y)	(56, 63)
21:20	(9, 0)	(25, 0)	(41, 0)	(57, 0)	(9, y)	(25, y)	(41, y)	(57, y)	(57, 63)
19:18	(10, 0)	(26, 0)	(42, 0)	(58, 0)	(10, y)	(26, y)	(42, y)	(58, y)	(58, 63)
17:16	(11, 0)	(27, 0)	(43, 0)	(59, 0)	(11, y)	(27, y)	(43, y)	(59, y)	(59, 63)
15:14	(4, 0)	(20, 0)	(36, 0)	(52, 0)	(4, y)	(20, y)	(36, y)	(52, y)	(52, 63)
13:12	(5, 0)	(21, 0)	(37, 0)	(53, 0)	(5, y)	(21, y)	(37, y)	(53, y)	(53, 63)
11:10	(6, 0)	(22, 0)	(38, 0)	(54, 0)	(6, y)	(22, y)	(38, y)	(54, y)	(54, 63)
9:8	(7, 0)	(23, 0)	(39, 0)	(55, 0)	(7, y)	(23, y)	(39, y)	(55, y)	(55, 63)
7:6	(0, 0)	(16, 0)	(32, 0)	(48, 0)	(0, y)	(16, y)	(32, y)	(48, y)	(48, 63)

Table 630. Buffer to pixel mapping for 64 x 64 pixel cursor format

	Offset into cursor memory								
Data bits	0	4	8	12	(16 * y)	(16 * y) +4	(16 * y) + 8	(16 * y) + 12	FC
5:4	(1, 0)	(17, 0)	(33, 0)	(49, 0)	(1, y)	(17, y)	(33, y)	(49, y)	(49, 63)
3:2	(2, 0)	(18, 0)	(34, 0)	(50, 0)	(2, y)	(18, y)	(34, y)	(50, y)	(50, 63)
1:0	(3, 0)	(19, 0)	(35, 0)	(51, 0)	(3, y)	(19, y)	(35, y)	(51, y)	(51, 63)

Cursor pixel encoding

Each pixel of the cursor requires two bits of information. These are interpreted as Color0, Color1, Transparent, and Transparent inverted.

In the coding scheme, bit 1 selects between color and transparent (AND mask) and bit 0 selects variant (XOR mask).

Table 631 shows the pixel encoding bit assignments.

Table 631. Pixel encoding

Value	Description
00	Color0. The cursor color is displayed according to the Red-Green-Blue (RGB) value programmed into the CRSR_PAL0 register.
01	Color1. The cursor color is displayed according to the RGB value programmed into the CRSR_PAL1 register.
10	Transparent. The cursor pixel is transparent, so is displayed unchanged. This enables the visible cursor to assume shapes that are not square.
11	Transparent inverted. The cursor pixel assumes the complementary color of the frame pixel that is displayed. This can be used to ensure that the cursor is visible regardless of the color of the frame buffer image.

27.7.6 Gray scaler

A patented gray scale algorithm drives monochrome and color STN panels. This provides 15 gray scales for monochrome displays. For STN color displays, the three color components (RGB) are gray scaled simultaneously. This results in 3375 (15x15x15) colors being available. The gray scaler transforms each 4-bit gray value into a sequence of activity-per-pixel over several frames, relying to some degree on the display characteristics, to give the representation of gray scales and color.

27.7.7 Upper and lower panel formatters

Formatters are used in STN mode to convert the gray scaler output to a parallel format as required by the display. For monochrome displays, this is either 4 or 8 bits wide, and for color displays, it is 8 bits wide. Table 632 shows a color display driven with 2 2/3 pixels worth of data in a repeating sequence.

Table 632. Color display driven with 2 2/3 pixel data

Byte	CLD[7]	CLD[6]	CLD[5]	CLD[4]	CLD[3]	CLD[2]	CLD[1]	CLD[0]
0	P2[Green]	P2[Red]	P1[Blue]	P1[Green]	P1[Red]	P0[Blue]	P0[Green]	P0[Red]
1	P5[Red]	P4q[Blue]	P4[Green]	P4[Red]	P3[Blue]	P3[Green]	P3[Red]	P2[Blue]
2	P7[Blue]	P7[Green]	P7[Red]	P6[Blue]	P6[Green]	P6[Red]	P5[Blue]	P5[Green]

Each formatter consists of three 3-bit (RGB) shift left registers. RGB pixel data bit values from the gray scaler are concurrently shifted into the respective registers. When enough data is available, a byte is constructed by multiplexing the registered data to the correct bit position to satisfy the RGB data pattern of LCD panel. The byte is transferred to the 3-byte FIFO, which has enough space to store eight color pixels.

27.7.8 Panel clock generator

The output of the panel clock generator block is the panel clock, pin LCDDCLK. The panel clock can be based on either the peripheral clock for the LCD block or the external clock input for the LCD, pin LCDCLKIN. Whichever source is selected can be divided down in order to produce the internal LCD clock, LCDCLK.

The panel clock generator can be programmed to output the LCD panel clock in the range of LCDCLK/2 to LCDCLK/1025 to match the bpp data rate of the LCD panel being used.

The CLKSEL bit in the POL register determines whether the base clock used is CCLK or the LCDCLKIN pin.

27.7.9 Timing controller

The primary function of the timing controller block is to generate the horizontal and vertical timing panel signals. It also provides the panel bias and enable signals. These timings are all register-programmable.

27.7.10 STN and TFT data select

Support is provided for passive Super Twisted Nematic (STN) and active Thin Film Transistor (TFT) LCD display types:

27.7.10.1 STN displays

STN display panels require algorithmic pixel pattern generation to provide pseudo gray scaling on monochrome displays, or color creation on color displays.

27.7.10.2 TFT displays

TFT display panels require the digital color value of each pixel to be applied to the display data inputs.

27.7.11 Interrupt generation

Four interrupts are generated by the LCD controller, and a single combined interrupt. The four interrupts are:

- Master bus error interrupt.
- Vertical compare interrupt.

- Next base address update interrupt.
- FIFO underflow interrupt.

Each of the four individual maskable interrupts is enabled or disabled by changing the mask bits in the INT_MSK register. These interrupts are also combined into a single overall interrupt, which is asserted if any of the individual interrupts are both asserted and unmasked. Provision of individual outputs in addition to a combined interrupt output enables use of either a global interrupt service routine, or modular device drivers to handle interrupts.

The status of the individual interrupt sources can be read from the INTRAW register.

27.7.11.1 Master bus error interrupt

The master bus error interrupt is asserted when an ERROR response is received by the master interface during a transaction with a slave. When such an error is encountered, the master interface enters an error state and remains in this state until clearance of the error has been signaled to it. When the respective interrupt service routine is complete, the master bus error interrupt may be cleared by writing a 1 to the BERIC bit in the INTCLR register. This action releases the master interface from its ERROR state to the start of FRAME state, and enables fresh frame of data display to be initiated.

27.7.11.2 Vertical compare interrupt

The vertical compare interrupt asserts when one of four vertical display regions, selected using the CTRL register, is reached. The interrupt can be made to occur at the start of:

- Vertical synchronization.
- Back porch.
- Active video.
- Front porch.

The interrupt may be cleared by writing a 1 to the VcompIC bit in the INTCLR register.

27.7.11.2.1 Next base address update interrupt

The LCD next base address update interrupt asserts when either the LCDUPBASE or LCDLPBASE values have been transferred to the LCDUPCURR or LCDLPCURR incrementers respectively. This signals to the system that it is safe to update the LCDUPBASE or the LCDLPBASE registers with new frame base addresses if required.

The interrupt can be cleared by writing a 1 to the LNBUIC bit in the INTCLR register

27.7.11.2.2 FIFO underflow interrupt

The FIFO underflow interrupt asserts when internal data is requested from an empty DMA FIFO. Internally, upper and lower panel DMA FIFO underflow interrupt signals are generated.

The interrupt can be cleared by writing a 1 to the FUFIC bit in the INTCLR register.

27.7.12 LCD power-up and power-down sequence

The LCD controller requires the following power-up sequence to be performed:

1. When power is applied, the following signals are held LOW:

- LCDLP
- LCDDCLK
- LCDFP
- LCDENAB/ LCDM
- LCDVD[23:0]
- LCDLE

2. When LCD power is stabilized, a 1 is written to the LcdEn bit in the CTRL register. This enables the following signals into their active states:

- LCDLP
- LCDDCLK
- LCDFP
- LCDENAB/ LCDM
- LCDLE

The LCDV[23:0] signals remain in an inactive state.

3. When the signals in step 2 have stabilized, the contrast voltage (not controlled or supplied by the LCD controller) is applied to the LCD panel.

4. If required, a software or hardware timer can be used to provide the minimum display specific delay time between application of the control signals and power to the panel display. On completion of the time interval, power is applied to the panel by writing a 1 to the LcdPwr bit within the CTRL register that, in turn, sets the LCDPWR signal high and enables the LCDV[23:0] signals into their active states. The LCDPWR signal is intended to be used to gate the power to the LCD panel.

The power-down sequence is the reverse of the above four steps and must be strictly followed, this time, writing the respective register bits with 0.

Figure 77 shows the power-up and power-down sequences.

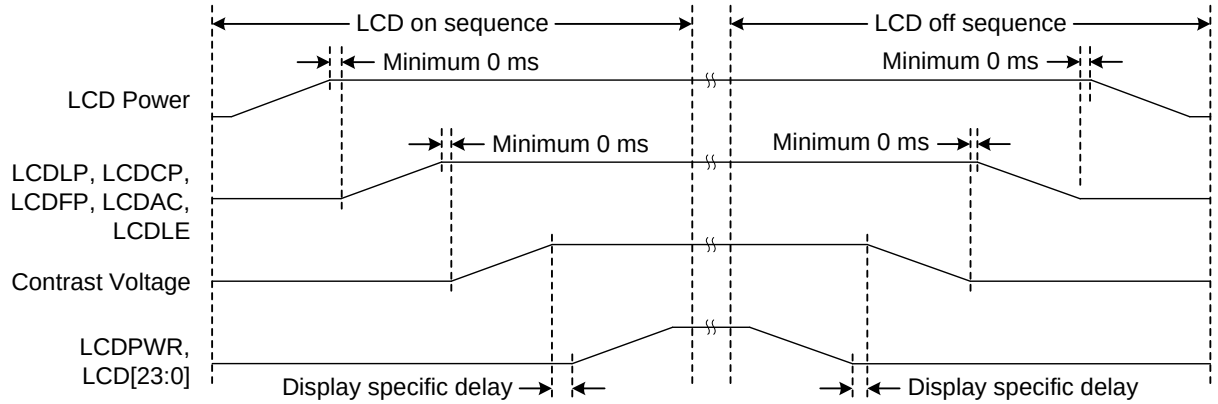
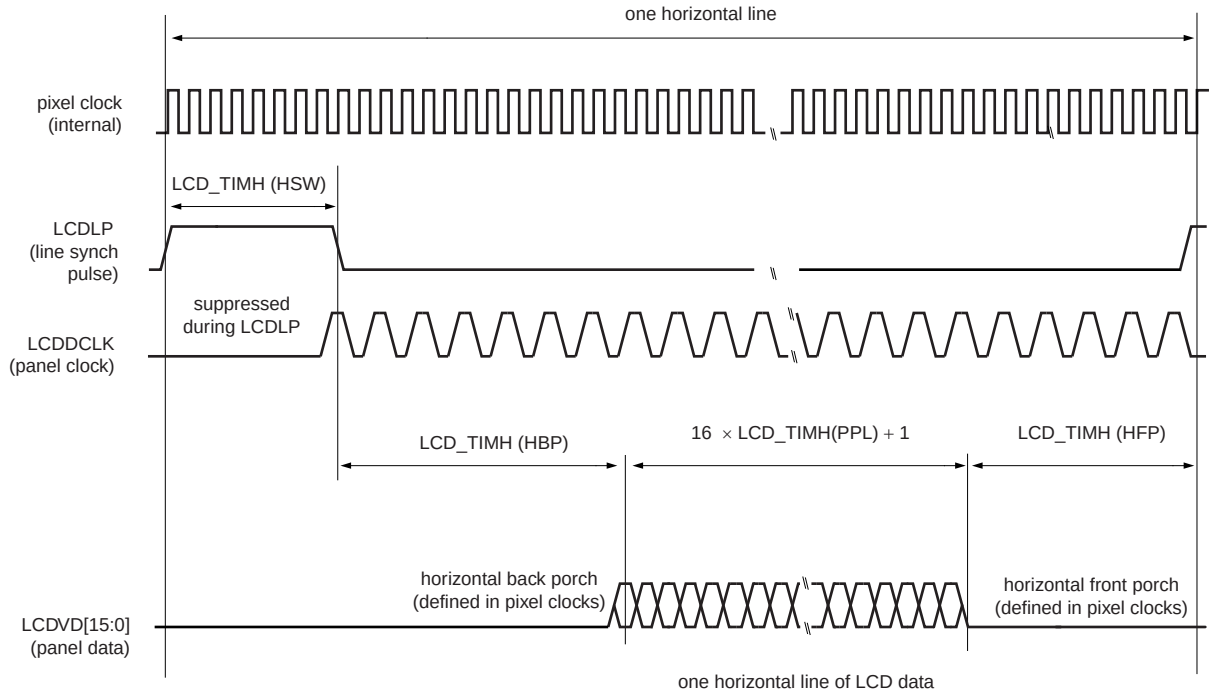


Fig 77. Power-up and power-down sequences

27.8 LCD timing diagrams



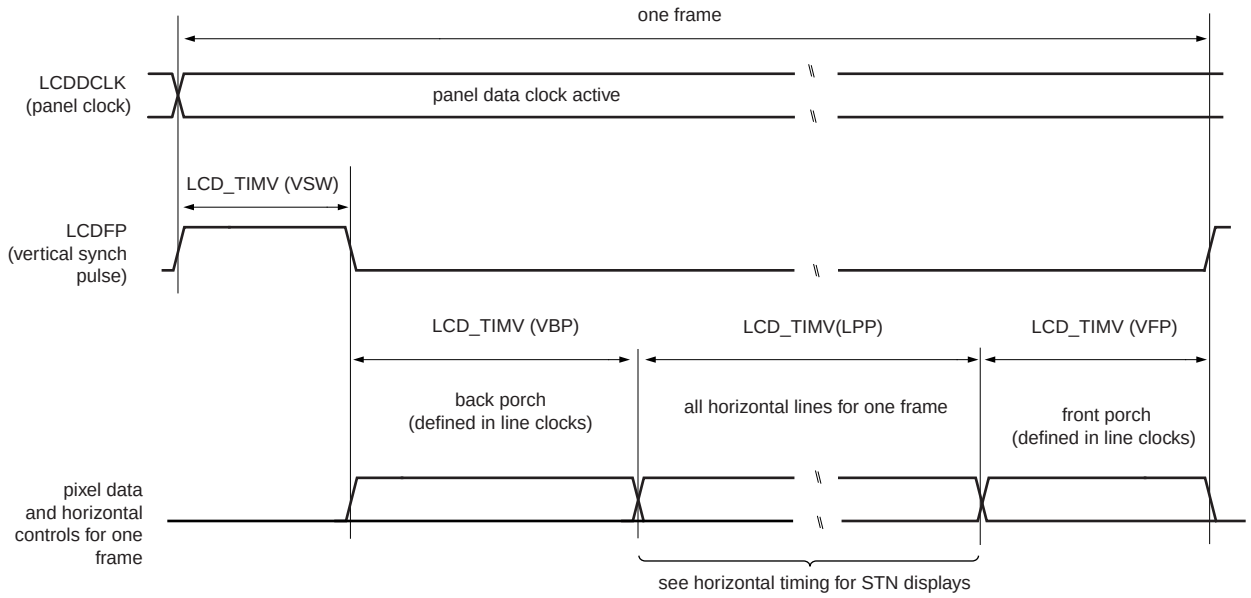
(1) The active data lines will vary with the type of STN panel (4-bit, 8-bit, color, mono) and with single or dual frames.

(2) The LCD panel clock is selected and scaled by the LCD controller and used to produce LCDCLK.

(3) The duration of the LCDLP signal is controlled by the HSW field in the TIMH register.

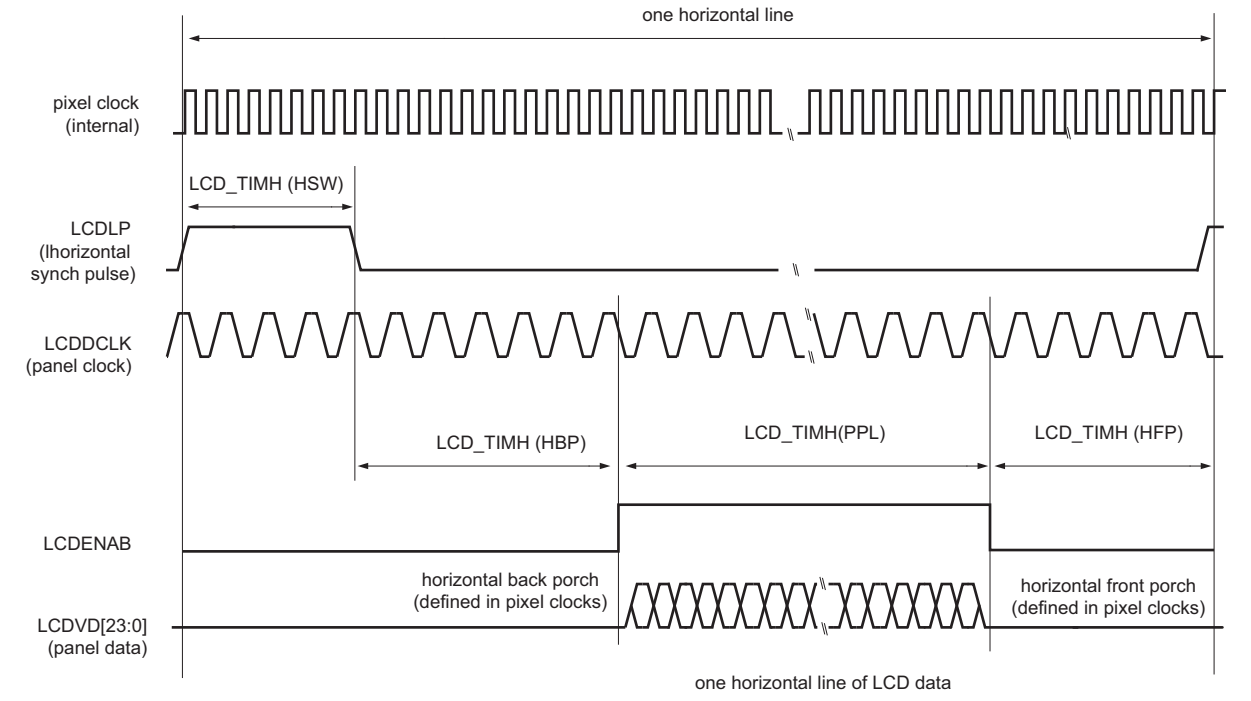
(4) The Polarity of the LCDLP signal is determined by the IHS bit in the POL register.

Fig 78. Horizontal timing for STN displays



(1) Signal polarities may vary for some displays.

Fig 79. Vertical timing for STN displays



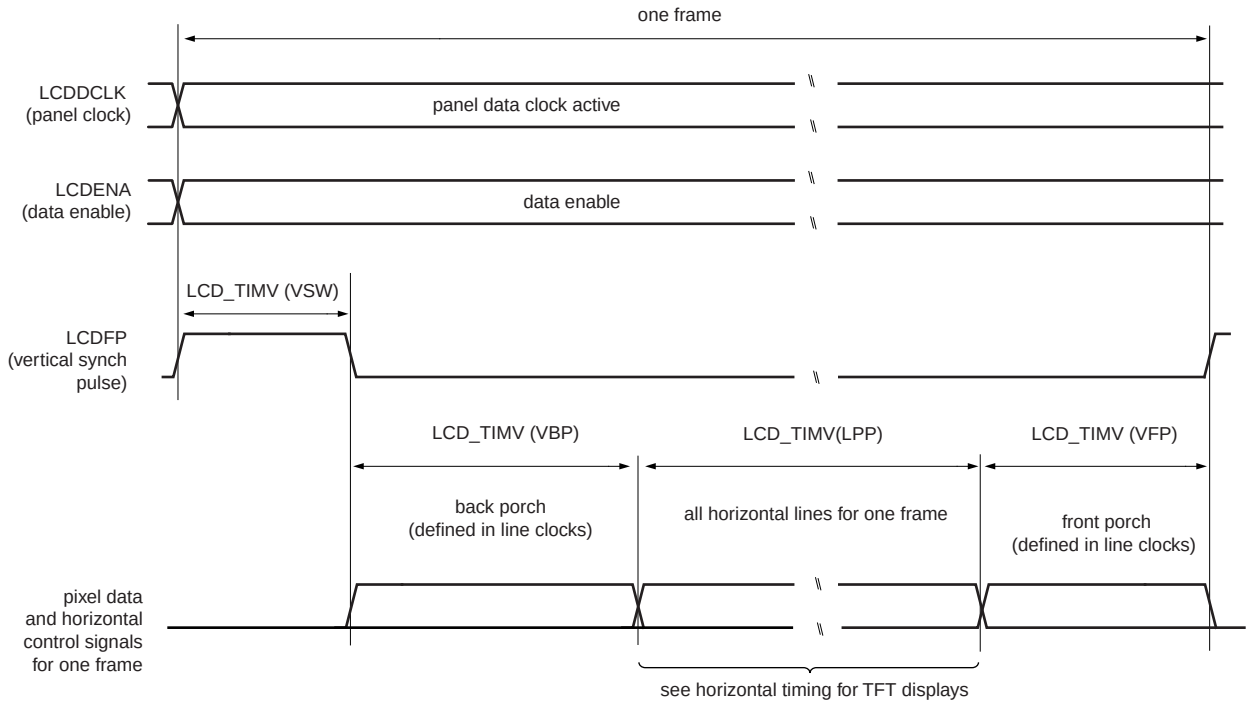
(1) The active data lines will vary with the type of TFT panel.

(2) The LCD panel clock is selected and scaled by the LCD controller and used to produce LCDCLK.

(3) The duration of the LCDLP is controlled by the HSW field in the TIMH register.

(4) The polarity of the LCDLP signal is determined by the IHS bit in the POL register.

Fig 80. Horizontal timing for TFT displays



(1) Polarities may vary for some displays.

Fig 81. Vertical timing for TFT displays

27.9 LCD panel signal usage

Table 633. LCD panel connections for STN single panel mode

External pin	4-bit mono STN single panel		8-bit mono STN single panel		Color STN single panel	
	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function
LCDVD23	-	-	-	-	-	-
LCDVD22	-	-	-	-	-	-
LCDVD21	-	-	-	-	-	-
LCDVD20	-	-	-	-	-	-
LCDVD19	-	-	-	-	-	-
LCDVD18	-	-	-	-	-	-
LCDVD17	-	-	-	-	-	-
LCDVD16	-	-	-	-	-	-
LCDVD15	-	-	-	-	-	-
LCDVD14	-	-	-	-	-	-
LCDVD13	-	-	-	-	-	-
LCDVD12	-	-	-	-	-	-
LCDVD11	-	-	-	-	-	-
LCDVD10	-	-	-	-	-	-
LCDVD9	-	-	-	-	-	-

Table 633. LCD panel connections for STN single panel mode

External pin	4-bit mono STN single panel		8-bit mono STN single panel		Color STN single panel	
	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function
LCDVD8	-	-	-	-	-	-
LCDVD7	-	-	P8_4	UD[7]	P8_4	UD[7]
LCDVD6	-	-	P8_5	UD[6]	P8_5	UD[6]
LCDVD5	-	-	P8_6	UD[5]	P8_6	UD[5]
LCDVD4	-	-	P8_7	UD[4]	P8_7	UD[4]
LCDVD3	P4_2	UD[3]	P4_2	UD[3]	P4_2	UD[3]
LCDVD2	P4_3	UD[2]	P4_3	UD[2]	P4_3	UD[2]
LCDVD1	P4_4	UD[1]	P4_4	UD[1]	P4_4	UD[1]
LCDVD0	P4_1	UD[0]	P4_1	UD[0]	P4_1	UD[0]
LCDLP	P7_6	LCDLP	P7_6	LCDLP	P7_6	LCDLP
LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM
LCDFP	P4_5	LCDFP	P4_5	LCDFP	P4_5	LCDFP
LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK
LCDLE	P7_0	LCDLE	P7_0	LCDLE	P7_0	LCDLE
LCDPWR	P7_7	CDPWR	P7_7	LCDPWR	P7_7	LCDPWR
GP_CLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN

Table 634. LCD panel connections for STN dual panel mode

External pin	4-bit mono STN dual panel		8-bit mono STN dual panel		Color STN dual panel	
	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function
LCDVD23	-	-	-	-	-	-
LCDVD22	-	-	-	-	-	-
LCDVD21	-	-	-	-	-	-
LCDVD20	-	-	-	-	-	-
LCDVD19	-	-	-	-	-	-
LCDVD18	-	-	-	-	-	-
LCDVD17	-	-	-	-	-	-
LCDVD16	-	-	-	-	-	-
LCDVD15	-	-	PB_4	LD[7]	PB_4	LD[7]
LCDVD14	-	-	PB_5	LD[6]	PB_5	LD[6]
LCDVD13	-	-	PB_6	LD[5]	PB_6	LD[5]
LCDVD12	-	-	P8_3	LD[4]	P8_3	LD[4]
LCDVD11	P4_9	LD[3]	P4_9	LD[3]	P4_9	LD[3]
LCDVD10	P4_10	LD[2]	P4_10	LD[2]	P4_10	LD[2]
LCDVD9	P4_8	LD[1]	P4_8	LD[1]	P4_8	LD[1]
LCDVD8	P7_5	LD[0]	P7_5	LD[0]	P7_5	LD[0]
LCDVD7	-	-		UD[7]	P8_4	UD[7]
LCDVD6	-	-	P8_5	UD[6]	P8_5	UD[6]

Table 634. LCD panel connections for STN dual panel mode

External pin	4-bit mono STN dual panel		8-bit mono STN dual panel		Color STN dual panel	
	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function
LCDVD5	-	-	P8_6	UD[5]	P8_6	UD[5]
LCDVD4	-	-	P8_7	UD[4]	P8_7	UD[4]
LCDVD3	P4_2	UD[3]	P4_2	UD[3]	P4_2	UD[3]
LCDVD2	P4_3	UD[2]	P4_3	UD[2]	P4_3	UD[2]
LCDVD1	P4_4	UD[1]	P4_4	UD[1]	P4_4	UD[1]
LCDVD0	P4_1	UD[0]	P4_1	UD[0]	P4_1	UD[0]
LCDLP	P7_6	LCDLP	P7_6	LCDLP	P7_6	LCDLP
LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM
LCDFP	P4_5	LCDFP	P4_5	LCDFP	P4_5	LCDFP
LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK
LCDLE	P7_0	LCDLE	P7_0	LCDLE	P7_0	LCDLE
LCDPWR	P7_7	LCDPWR	P7_7	LCDPWR	P7_7	LCDPWR
GP_CLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN

Table 635. LCD panel connections for TFT panels

External pin	TFT 12 bit (4:4:4 mode)		TFT 16 bit (5:6:5 mode)		TFT 16 bit (1:5:5:5 mode)		TFT 24 bit	
	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function
LCDVD23	PB_0	BLUE3	PB_0	BLUE4	PB_0	BLUE4		BLUE7
LCDVD22	PB_1	BLUE2	PB_1	BLUE3	PB_1	BLUE3		BLUE6
LCDVD21	PB_2	BLUE1	PB_2	BLUE2	PB_2	BLUE2		BLUE5
LCDVD20	PB_3	BLUE0	PB_3	BLUE1	PB_3	BLUE1		BLUE4
LCDVD19	-	-	P7_1	BLUE0	P7_1	BLUE0		BLUE3
LCDVD18	-	-	-	-	P7_2	intensity		BLUE2
LCDVD17	-	-	-	-	-	-	P7_3	BLUE1
LCDVD16	-	-	-	-	-	-	P7_4	BLUE0
LCDVD15	PB_4	GREEN3	PB_4	GREEN5	PB_4	GREEN4	PB_4	GREEN7
LCDVD14	PB_5	GREEN2	PB_5	GREEN4	PB_5	GREEN3	PB_5	GREEN6
LCDVD13	PB_6	GREEN1	PB_6	GREEN3	PB_6	GREEN2	PB_6	GREEN5
LCDVD12	P8_3	GREEN0	P8_3	GREEN2	P8_3	GREEN1	P8_3	GREEN4
LCDVD11	-	-	P4_9	GREEN1	P4_9	GREEN0	P4_9	GREEN3
LCDVD10	-	-	P4_10	GREEN0	P4_10	intensity	P4_10	GREEN2
LCDVD9	-	-	-	-	-	-	P4_8	GREEN1
LCDVD8	-	-	-	-	-	-	P7_5	GREEN0
LCDVD7	P8_4	RED3	P8_4	RED4	P8_4	RED4	P8_4	RED7
LCDVD6	P8_5	RED2	P8_5	RED3	P8_5	RED3	P8_5	RED6
LCDVD5	P8_6	RED1	P8_6	RED2	P8_6	RED2	P8_6	RED5
LCDVD4	P8_7	RED0	P8_7	RED1	P8_7	RED1	P8_7	RED4

Table 635. LCD panel connections for TFT panels

External pin	TFT 12 bit (4:4:4 mode)		TFT 16 bit (5:6:5 mode)		TFT 16 bit (1:5:5:5 mode)		TFT 24 bit	
	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function	LPC43xx pin used	LCD function
LCDVD3	-	-	P4_2	RED0	P4_2	RED0	P4_2	RED3
LCDVD2	-	-	-	-	P4_3	intensity	P4_3	RED2
LCDVD1	-	-	-	-	-	-	P4_4	RED1
LCDVD0	-	-	-	-	-	-	P4_1	RED0
LCDLP	P7_6	LCDLP	P7_6	LCDLP	P7_6	LCDLP	P7_6	LCDLP
LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/ LCDM	P4_6	LCDENAB/L CDM
LCDFP	P4_5	LCDFP	P4_5	LCDFP	P4_5	LCDFP	P4_5	LCDFP
LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK	P4_7	LCDDCLK
LCDLE	P7_0	LCDLE	P7_0	LCDLE	P7_0	LCDLE	P7_0	LCDLE
LCDPWR	P7_7	LCDPWR	P7_7	LCDPWR	P7_7	LCDPWR	P7_7	LCDPWR
GP_CLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN	PF_4	LCDCLKIN

28.1 How to read this chapter

The SCT is available on all LPC43xx parts.

28.2 Basic configuration

The SCT is configured as follows:

- See Table 636 for clocking and power control.
- The SCT is reset by the SCT_RST (reset #37).
- Connect inputs and outputs of the SCT through the GIMA (see Chapter 16).
- The SCT combined interrupt is connected to slot # 10 in the NVIC. SCT outputs 2, 6, 14 are ORed with timer match channels and connected to slots # 13, 14, 16 in the Event router (see Table 24).
- For connecting the SCT outputs 0 and 1 to the GPDMA, use the DMAMUX register in the CREG block (see Table 41) and enable the GPDMA channel in the DMA Channel Configuration registers Section 19.6.20.

Table 636. SCT clocking and power control

	Base clock	Branch clock	Operating frequency
SCT	BASE_M4_CLK	CLK_M4_SCT	up to 204 MHz

28.3 Features

- Two 16-bit counters or one 32-bit counter.
- Counter(s) clocked by bus clock or selected input.
- Up counter(s) or up-down counter(s).
- State variable allows sequencing across multiple counter cycles.
- Event can be defined by a counter match condition, an input (or output) condition, a combination of a match and/or and input/output condition in a specified state.
- Events control outputs, interrupts, and DMA requests.
- Selected event(s) can limit, halt, start, or stop a counter.
- Supports:
  - 8 inputs
  - 16 outputs
  - 16 match/capture registers
  - 16 events
  - 32 states

28.4 General description

The State Configurable Timer (SCT) allows a wide variety of timing, counting, output modulation, and input capture operations.

The most basic user-programmable option is whether a SCT operates as two 16-bit counters or a unified 32-bit counter. In the two-counter case, in addition to the counter value the following operational elements are independent for each half:

- State variable
- Limit, halt, stop, and start conditions
- Values of Match/Capture registers, plus reload or capture control values

In the two-counter case, the following operational elements are global to the SCT, but events, outputs, interrupts, and DMA requests can use match conditions from either counter:

- Clock selection
- Inputs
- Events
- Outputs
- Interrupts
- DMA requests

Remark: This document uses the term “bus error” to indicate a SCT response that makes the processor take an exception.

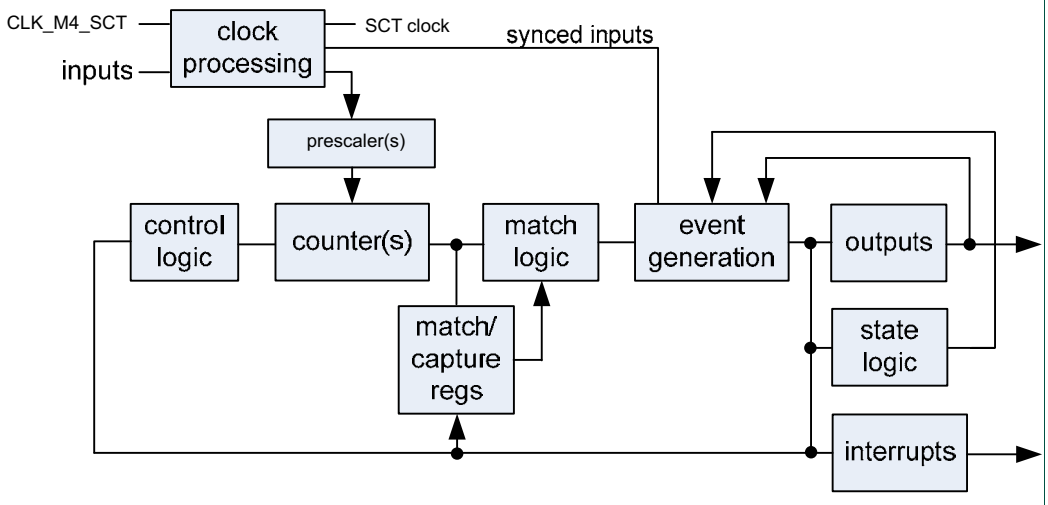


Fig 82. SCT block diagram

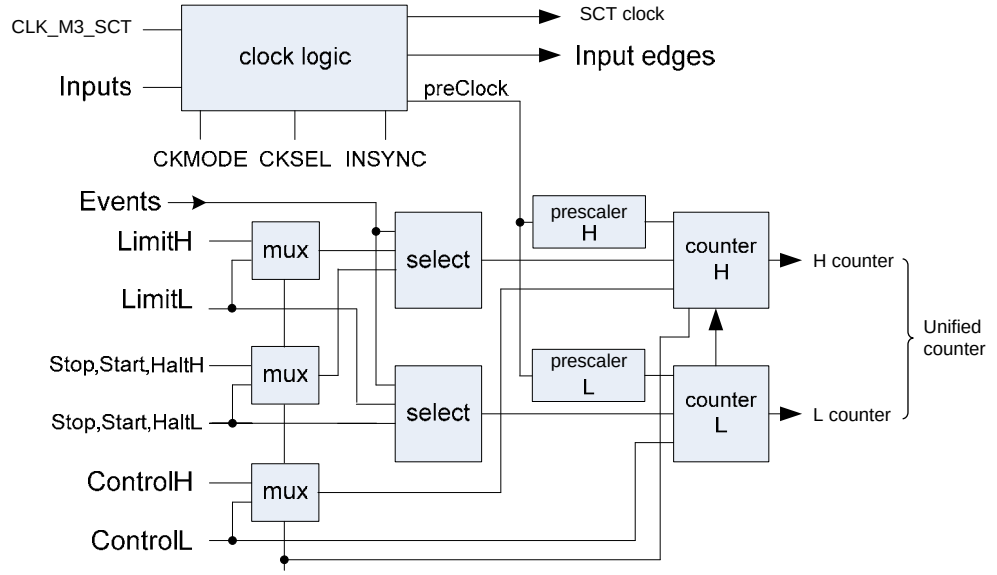


Fig 83. SCT counter and select logic

28.5 Pin description

The SCT inputs can originate from the external pins or from several internal sources. Each SCT input is connected to one GIMA register which defines the input source.

SCT outputs are connected to the CTOUT_n pins and are ORed with timer match outputs when the CTOUCTRL bit is set to 0 in CREG6 (see Table 43). Some SCT outputs are connected to multiple destinations at once, for example to an external pin and the event router.

Table 637. SCT inputs and outputs

Description	Pin function	Internal signal	Default (see GIMA, Table 138)	CTOUTCTRL bit (see Table 43)
SCT inputs				
SCT input 0	CTIN_0	-	yes	-
SCT input 1	CTIN_1	-	yes	-
	-	USART2 TX active	no	-
SCT input 2	CTIN_2	-	yes	-
SCT input 3	CTIN_3	-	yes	-
	-	USART0 TX active	no	-
	-	I2S1_RX_MWS	no	-
	-	I2S1_TX_MWS	no	-
SCT input 4	CTIN4	-	yes	-
		USART0 RX active	no	-
		I2S1_RX_MWS	no	-
		I2S1_TX_MWS	no	-

Table 637. SCT inputs and outputs *...continued*

Description	Pin function	Internal signal	Default (see GIMA, Table 138)	CTOUTCTRL bit (see Table 43)
SCT input 5	CTIN_5	-	yes	-
	-	USART2 TX active	no	-
SCT input 6	CTIN_6	-	yes	-
	-	USART3 TX active	no	-
	-	I2S0_RX_MWS	no	-
	-	I2S0_TX_MWS	no	-
SCT input 7	CTIN_7	-	yes	-
	-	USART3 RX active	no	-
	-	SOF0	no	-
	-	SOF1	no	-
SCT outputs				
SCT output 0 ORed with Timer0 match output 0	CTOUT_0	-	-	0
SCT output 0	CTOUT_0	-	-	1
SCT output 1 ORed with Timer0 match output 1	CTOUT_1	-	-	0
SCT output 1	CTOUT_1	-	-	1
SCT output 2 ORed with Timer0 match output 2	CTOUT_2	Event router input 13	-	0
SCT output 2	CTOUT_2	Event router input 13	-	1
SCT output 3 ORed with Timer0 match output 3	CTOUT_3	T1 capture channel 3	-	0
SCT output 3	CTOUT_3	T1 capture channel 3	-	1
SCT output 4 ORed with Timer1 match output 0	CTOUT_4	-	-	0
SCT output 4	CTOUT_4	-		1
SCT output 5 ORed with Timer1 match output 1	CTOUT_5	-	-	0
SCT output 5	CTOUT_5	-		1
SCT output 6 ORed with Timer1 match output 2	CTOUT_6	Event router input 14	-	0
SCT output 6	CYOUT_6	Event router input 14		1
SCT output 7 ORed with Timer1 match output 3	CTOUT_7	T2 capture channel 3	-	0
SCT output 7	CTOUT_7	T2 capture channel 3		1
SCT output 8 ORed with Timer2 match output 0	CTOUT_8	ADC start1 input (ADC CR register bit START = 0x3)	-	0
SCT output 8	CTOUT_8	ADC start1 input (ADC CR register bit START = 0x3)		1
SCT output 9 ORed with Timer2 match output 1	CTOUT_9	-	-	0
SCT output 9	CTOUT_9	-		1
SCT output 10 ORed with Timer2 match output 2	CTOUT_10	-	-	0
SCT output 10	CTOUT_10	-		1
SCT output 11 ORed with Timer2 match output 3	CTOUT_11	T3 capture channel 3	-	0
SCT output 11	CTOUT_11	T3 capture channel 3	-	1
SCT output 12 ORed with Timer3 match output 0	CTOUT_12	-	-	0

Table 637. SCT inputs and outputs *...continued*

Description	Pin function	Internal signal	Default (see GIMA, Table 138)	CTOUTCTRL bit (see Table 43)
SCT output 12	CTOUT_12	-	-	1
SCT output 13 ORed with Timer3 match output 1	CTOUT_13	-	-	0
SCT output 13	CTOUT_13	-	-	1
SCT output 14 ORed with Timer3 match output 2	CTOUT_14	Event router input 16	-	0
SCT output 14	CTOUT_14	Event router input 16	-	1
SCT output 15 ORed with Timer3 match output 3	CTOUT_15	T0 capture channel 3/ADC start0 input (ADC CR register START bits = 0x2)	-	0
SCT output 15	CTOUT_15	T0 capture channel 3/ADC start0 input (ADC CR register START bits = 0x2)	-	1

28.6 Register description

The register addresses of the State Configurable Timer are shown in Table 638. For most of the SCT registers, the register function depends on the setting of certain other register bits:

1. The UNIFY bit in the CONFIG register determines whether the SCT is used as one 32-bit register (for operation as one 32-bit counter/timer) or as two 16-bit counter/timers named L and H. The setting of the UNIFY bit is reflected in the register map:
   - UNIFY = 1: Only one register is used (for operation as one 32-bit counter/timer).
   - UNIFY = 0: Access the L and H registers by a 32-bit read or write operation or can be read or written to individually (for operation as two 16-bit counter/timers).

   Typically, the UNIFY bit is configured by writing to the CONFIG register before any other registers are accessed.
2. The REGMODEn bits in the REGMODE register determine whether each set of Match/Capture registers uses the match or capture functionality:
   - REGMODEn = 1: Registers operate as match and reload registers.
   - REGMODEn = 0: Registers operate as capture and capture control registers.

Table 638. Register overview: State Configurable Timer (base address 0x4000 0000)

Name	Access	Address offset	Description	Reset value	Reference
CONFIG	R/W	0x000	SCT configuration register	0x0000 7E00	Table 639
CTRL	R/W	0x004	SCT control register	0x0004 0004	Table 640
CTRL_L	R/W	0x004	SCT control register low counter 16-bit	0x0004 0004	Table 640
CTRL_H	R/W	0x006	SCT control register high counter 16-bit	0x0004 0004	Table 640
LIMIT	R/W	0x008	SCT limit register	0x0000 0000	Table 641
LIMIT_L	R/W	0x008	SCT limit register low counter 16-bit	0x0000 0000	Table 641

Table 638. Register overview: State Configurable Timer (base address 0x4000 0000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
LIMIT_H	R/W	0x00A	SCT limit register high counter 16-bit	0x0000 0000	Table 641
HALT	R/W	0x00C	SCT halt condition register	0x0000 0000	Table 642
HALT_L	R/W	0x00C	SCT halt condition register low counter 16-bit	0x0000 0000	Table 642
HALT_H	R/W	0x00E	SCT halt condition register high counter 16-bit	0x0000 0000	Table 642
STOP	R/W	0x010	SCT stop condition register	0x0000 0000	Table 643
STOP_L	R/W	0x010	SCT stop condition register low counter 16-bit	0x0000 0000	Table 643
STOP_H	R/W	0x012	SCT stop condition register high counter 16-bit	0x0000 0000	Table 643
START	R/W	0x014	SCT start condition register	0x0000 0000	Table 644
START_L	R/W	0x014	SCT start condition register low counter 16-bit	0x0000 0000	Table 644
START_H	R/W	0x016	SCT start condition register high counter 16-bit	0x0000 0000	Table 644
-	-	0x018 - 0x03C	Reserved		-
COUNT	R/W	0x040	SCT counter register	0x0000 0000	Table 645
COUNT_L	R/W	0x040	SCT counter register low counter 16-bit	0x0000 0000	Table 645
COUNT_H	R/W	0x042	SCT counter register high counter 16-bit	0x0000 0000	Table 645
STATE	R/W	0x044	SCT state register	0x0000 0000	Table 646
STATE_L	R/W	0x044	SCT state register low counter 16-bit	0x0000 0000	Table 646
STATE_H	R/W	0x046	SCT state register high counter 16-bit	0x0000 0000	Table 646
INPUT	RO	0x048	SCT input register	0x0000 0000	Table 647
REGMODE	R/W	0x04C	SCT match/capture registers mode register	0x0000 0000	Table 648
REGMODE_L	R/W	0x04C	SCT match/capture registers mode register low counter 16-bit	0x0000 0000	Table 648
REGMODE_H	R/W	0x04E	SCT match/capture registers mode register high counter 16-bit	0x0000 0000	Table 648
OUTPUT	R/W	0x050	SCT output register	0x0000 0000	Table 649
OUTPUTDIRCTRL	R/W	0x054	SCT output counter direction control register	0x0000 0000	Table 650
RES	R/W	0x058	SCT conflict resolution register	0x0000 0000	Table 651
DMAREQ0	R/W	0x05C	SCT DMA request 0 register	0x0000 0000	Table 652
DMAREQ1	R/W	0x060	SCT DMA request 1 register	0x0000 0000	Table 653
-	-	0x064 - 0x0EC	Reserved	-	-
EVEN	R/W	0x0F0	SCT event enable register	0x0000 0000	Table 654
EVFLAG	R/W	0x0F4	SCT event flag register	0x0000 0000	Table 655
CONEN	R/W	0x0F8	SCT conflict enable register	0x0000 0000	Table 656
CONFLAG	R/W	0x0FC	SCT conflict flag register	0x0000 0000	Table 657
MATCH0 to MATCH15	R/W	0x100 to 0x13C	SCT match value register of match channels 0 to 15; REGMOD0 to REGMODE15 = 0	0x0000 0000	Table 657
MATCH0_L to MATCH15_L	R/W	0x100 to 0x13C	SCT match value register of match channels 0 to 15; low counter 16-bit; REGMOD0_L to REGMODE15_L = 0	0x0000 0000	Table 657

Table 638. Register overview: State Configurable Timer (base address 0x4000 0000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
MATCH0_H to MATCH15_H	R/W	0x102 to 0x13E	SCT match value register of match channels 0 to 15; high counter 16-bit; REGMOD0_H to REGMODE15_H = 0	0x0000 0000	Table 657
CAP0 to CAP15		0x100 to 0x13C	SCT capture register of capture channel 0 to 15; REGMOD0 to REGMODE15 = 1	0x0000 0000	Table 659
CAP0_L to CAP15_L		0x100 to 0x13C	SCT capture register of capture channel 0 to 15; low counter 16-bit; REGMOD0_L to REGMODE15_L = 1	0x0000 0000	Table 659
CAP0_H to CAP15_H		0x102 to 0x13E	SCT capture register of capture channel 0 to 15; high counter 16-bit; REGMOD0_H to REGMODE15_H = 1	0x0000 0000	Table 659
MATCHREL0 to MATCHREL15	R/W	0x200 to 0x23C	SCT match reload value register 0 to 15; REGMOD0 = 0 to REGMODE15 = 0	0x0000 0000	Table 660
MATCHREL0_L to MATCHREL15_L	R/W	0x200 to 0x23C	SCT match reload value register 0 to 15; low counter 16-bit; REGMOD0_L = 0 to REGMODE15_L = 0	0x0000 0000	Table 660
MATCHREL0_H to MATCHREL15_H	R/W	0x202 to 0x23E	SCT match reload value register 0 to 15; high counter 16-bit; REGMOD0_H = 0 to REGMODE15_H = 0	0x0000 0000	Table 660
CAPCTRL0 to CAPCTRL15		0x200 to 0x23C	SCT capture control register 0 to 15; REGMOD0 = 1 to REGMODE15 = 1	0x0000 0000	Table 661
CAPCTRL0_L to CAPCTRL15_L		0x200 to 0x23C	SCT capture control register 0 to 15; low counter 16-bit; REGMOD0_L = 1 to REGMODE15_L = 1	0x0000 0000	Table 661
CAPCTRL0 to CAPCTRL15		0x202 to 0x23E	SCT capture control register 0 to 15; high counter 16-bit; REGMOD0 = 1 to REGMODE15 = 1	0x0000 0000	Table 661
EVSTATEMSK0	R/W	0x300	SCT event state register 0	0x0000 0000	Table 662
EVCTRL0	R/W	0x304	SCT event control register 0	0x0000 0000	Table 663
EVSTATEMSK1	R/W	0x308	SCT event state register 1	0x0000 0000	Table 662
EVCTRL1	R/W	0x30C	SCT event control register 1	0x0000 0000	Table 663
EVSTATEMSK2	R/W	0x310	SCT event state register 2	0x0000 0000	Table 662
EVCTRL2	R/W	0x314	SCT event control register 2	0x0000 0000	Table 663
EVSTATEMSK3	R/W	0x318	SCT event state register 3	0x0000 0000	Table 662
EVCTRL3	R/W	0x31C	SCT event control register 3	0x0000 0000	Table 663
EVSTATEMSK4	R/W	0x320	SCT event state register 4	0x0000 0000	Table 662
EVCTRL4	R/W	0x324	SCT event control register4	0x0000 0000	Table 663
EVSTATEMSK5	R/W	0x328	SCT event state register 5	0x0000 0000	Table 662
EVCTRL5	R/W	0x32C	SCT event control register 5	0x0000 0000	Table 663
EVSTATEMSK6	R/W	0x330	SCT event state register 6	0x0000 0000	Table 662
EVCTRL6	R/W	0x334	SCT event control register 6	0x0000 0000	Table 663
EVSTATEMSK7	R/W	0x338	SCT event state register 7	0x0000 0000	Table 662
EVCTRL7	R/W	0x33C	SCT event control register 7	0x0000 0000	Table 663
EVSTATEMSK8	R/W	0x340	SCT event state register 8	0x0000 0000	Table 662
EVCTRL8	R/W	0x344	SCT event control register 8	0x0000 0000	Table 663

Table 638. Register overview: State Configurable Timer (base address 0x4000 0000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
EVSTATEMSK9	R/W	0x348	SCT event state register 9	0x0000 0000	Table 662
EVCTRL9	R/W	0x34C	SCT event control register 9	0x0000 0000	Table 663
EVSTATEMSK10	R/W	0x350	SCT event state register 10	0x0000 0000	Table 662
EVCTRL10	R/W	0x354	SCT event control register 10	0x0000 0000	Table 663
EVSTATEMSK11	R/W	0x358	SCT event state register 11	0x0000 0000	Table 662
EVCTRL11	R/W	0x35C	SCT event control register 11	0x0000 0000	Table 663
EVSTATEMSK12	R/W	0x360	SCT event state register 12	0x0000 0000	Table 662
EVCTRL12	R/W	0x364	SCT event control register 12	0x0000 0000	Table 663
EVSTATEMSK13	R/W	0x368	SCT event state register 13	0x0000 0000	Table 662
EVCTRL13	R/W	0x36C	SCT event control register 13	0x0000 0000	Table 663
EVSTATEMSK14	R/W	0x370	SCT event state register 14	0x0000 0000	Table 662
EVCTRL14	R/W	0x374	SCT event control register 14	0x0000 0000	Table 663
EVSTATEMSK15	R/W	0x378	SCT event state register 15	0x0000 0000	Table 662
EVCTRL15	R/W	0x37C	SCT event control register 15	0x0000 0000	Table 663
OUTPUTSET0	R/W	0x500	SCT output 0 set register	0x0000 0000	Table 664
OUTPUTCL0	R/W	0x504	SCT output 0 clear register	0x0000 0000	Table 665
OUTPUTSET1	R/W	0x508	SCT output 1 set register	0x0000 0000	Table 664
OUTPUTCL1	R/W	0x50C	SCT output 1 clear register	0x0000 0000	Table 665
OUTPUTSET2	R/W	0x510	SCT output 2 set register	0x0000 0000	Table 664
OUTPUTCL2	R/W	0x514	SCT output 2 clear register	0x0000 0000	Table 665
OUTPUTSET3	R/W	0x518	SCT output 3 set register	0x0000 0000	Table 664
OUTPUTCL3	R/W	0x51C	SCT output 3 clear register	0x0000 0000	Table 665
OUTPUTSET4	R/W	0x520	SCT output 4 set register	0x0000 0000	Table 664
OUTPUTCL4	R/W	0x524	SCT output 4 clear register	0x0000 0000	Table 665
OUTPUTSET5	R/W	0x528	SCT output 5 set register	0x0000 0000	Table 664
OUTPUTCL5	R/W	0x52C	SCT output 5 clear register	0x0000 0000	Table 665
OUTPUTSET6	R/W	0x530	SCT output 6 set register	0x0000 0000	Table 664
OUTPUTCL6	R/W	0x534	SCT output 6 clear register	0x0000 0000	Table 665
OUTPUTSET7	R/W	0x538	SCT output 7 set register	0x0000 0000	Table 664
OUTPUTCL7	R/W	0x53C	SCT output 7 clear register	0x0000 0000	Table 665
OUTPUTSET8	R/W	0x540	SCT output 8 set register	0x0000 0000	Table 664
OUTPUTCL8	R/W	0x544	SCT output 8 clear register	0x0000 0000	Table 665
OUTPUTSET9	R/W	0x548	SCT output 9 set register	0x0000 0000	Table 664
OUTPUTCL9	R/W	0x54C	SCT output 9 clear register	0x0000 0000	Table 665
OUTPUTSET10	R/W	0x550	SCT output 10 set register	0x0000 0000	Table 664
OUTPUTCL10	R/W	0x554	SCT output 10 clear register	0x0000 0000	Table 665
OUTPUTSET11	R/W	0x558	SCT output 11 set register	0x0000 0000	Table 664
OUTPUTCL11	R/W	0x55C	SCT output 11 clear register	0x0000 0000	Table 665
OUTPUTSET12	R/W	0x560	SCT output 12 set register	0x0000 0000	Table 664

Table 638. Register overview: State Configurable Timer (base address 0x4000 0000) *…continued*

Name	Access	Address offset	Description	Reset value	Reference
OUTPUTCL12	R/W	0x564	SCT output 12 clear register	0x0000 0000	Table 665
OUTPUTSET13	R/W	0x568	SCT output 13 set register	0x0000 0000	Table 664
OUTPUTCL13	R/W	0x56C	SCT output 13 clear register	0x0000 0000	Table 665
OUTPUTSET14	R/W	0x570	SCT output 14 set register	0x0000 0000	Table 664
OUTPUTCL14	R/W	0x574	SCT output 14 clear register	0x0000 0000	Table 665
OUTPUTSET15	R/W	0x578	SCT output 15 set register	0x0000 0000	Table 664
OUTPUTCL15	R/W	0x57C	SCT output 15 clear register	0x0000 0000	Table 665

28.6.1 SCT configuration register

This register configures the overall operation of the SCT. Write to this register before any other registers.

Table 639. SCT configuration register (CONFIG - address 0x4000 0000) bit description

Bit	Symbol	Value	Description	Reset value
0	UNIFY		SCT operation	0
		0	The SCT operates as two 16-bit counters named L and H.	
		1	The SCT operates as a unified 32-bit counter.	
2:1	CLKMODE		SCT clock mode	00
		0x0	The bus clock clocks the SCT and prescalers.	
		0x1	The SCT clock is the bus clock, but the prescalers are enabled to count only when sampling of the input selected by the CKSEL field finds the selected edge. The minimum pulse width on the clock input is 1 bus clock period. This mode is the high-performance sampled-clock mode.	
		0x2	The input selected by CKSEL clocks the SCT and prescalers. The input is synchronized to the bus clock and possibly inverted. The minimum pulse width on the clock input is 1 bus clock period. This mode is the low-power sampled-clock mode.	
		0x3	The input edge selected by the CKSEL field clocks the SCT and prescalers. In this mode, the following applies: Most of the SCT is clocked by the (selected polarity of the) input. Outputs are switched synchronously to the input clock. The input clock rate must be at least half the bus clock rate and can be faster than the bus clock.	

Table 639. SCT configuration register (CONFIG - address 0x4000 0000) bit description *...continued*

Bit	Symbol	Value	Description	Reset value
6:3	CLKSEL		SCT clock select	0000
		0x0	Rising edges on input 0.	
		0x1	Falling edges on input 0.	
		0x2	Rising edges on input 1.	
		0x3	Falling edges on input 1.	
		0x4	Rising edges on input 2.	
		0x5	Falling edges on input 2.	
		0x6	Rising edges on input 3.	
		0x7	Falling edges on input 3.	
		0x8	Rising edges on input 4.	
		0x9	Falling edges on input 4.	
		0xA	Rising edges on input 5.	
		0xB	Falling edges on input 5.	
		0xC	Rising edges on input 6.	
		0xD	Falling edges on input 6.	
		0xE	Rising edges on input 7.	
		0xF	Falling edges on input 7.	
7	NORELAODL_ NORELOADU	-	A 1 in this bit prevents the lower match registers from being reloaded from their respective reload registers. Software can write to set or clear this bit at any time. This bit applies to both the higher and lower registers when the UNIFY bit is set.	0
8	NORELOADH	-	A 1 in this bit prevents the higher match registers from being reloaded from their respective reload registers. Software can write to set or clear this bit at any time. This bit is not used when the UNIFY bit is set.	0
16:9	INSYNCn	-	Synchronization for input n (bit 9 = input 0, bit 10 = input 1,..., bit 16 = input 7). A 1 in one of these bits subjects the corresponding input to synchronization to the SCT clock, before it is used to create an event. If an input is synchronous to the SCT clock, keep its bit 0 for faster response. When the CKMODE field is 1x, the bit in this field, corresponding to the input selected by the CKSEL field, is not used.	1
31:17	-		Reserved	-

28.6.2 SCT control register

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers CTRL_L (address 0x4000 4004) and CTRL_H (address 0x4000 4006). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

All bits in this register can be written to when the counter is stopped or halted. When the counter is running, the only bits that can be written are STOP or HALT.(Other bits can be written in a subsequent write after HALT is set to 1.)

Table 640. SCT control register (CTRL - address 0x4000 0004) bit description

Bit	Symbol	Value	Description	Reset value
0	DOWN_L	-	This bit is 1 when the L or unified counter is counting down. Hardware sets this bit when the counter limit is reached and BIDIR is 1. Hardware clears this bit when the counter reaches 0.	0
1	STOP_L	-	When this bit is 1 and HALT is 0, the L or unified counter does not run but I/O events related to the counter can occur. If such an event matches the mask in the Start register, this bit is cleared and counting resumes.	0
2	HALT_L	-	When this bit is 1, the L or unified counter does not run and no events can occur. A reset sets this bit. **Remark:** Once set, only software can clear this bit to restore counter operation.	1
3	CLRCTR_L	-	Writing a 1 to this bit clears the L or unified counter. This bit always reads as 0.	0
4	BIDIR_L		L or unified counter direction select	0
		0	The counter counts up to its limit condition, then is cleared to zero.	
		1	The counter counts up to its limit, then counts down to 0.	
12:5	PRE_L	-	Specifies the factor by which the SCT clock is prescaled to produce the L or unified counter clock. The counter clock is clocked at the rate of the SCT clock divided by PRE_L+1. **Remark:** Clear the counter (by writing a 1 to the CLRCTR bit) whenever changing the PRE value.	0
15:13	-		Reserved	
16	DOWN_H	-	This bit is 1 when the H counter is counting down. Hardware sets this bit when the counter limit is reached and BIDIR is 1. Hardware clears this bit when the counter reaches 0.	0
17	STOP_H	-	When this bit is 1 and HALT is 0, the H counter does not run but I/O events related to the counter can occur. If such an event matches the mask in the Start register, this bit is cleared and counting resumes.	0
18	HALT_H	-	When this bit is 1, the H counter does not run and no events can occur. A reset sets this bit. **Remark:** Once set, this bit can only be cleared by software to restore counter operation.	1
19	CLRCTR_H	-	Writing a 1 to this bit clears the H counter. This bit always reads as 0.	0
20	BIDIR_H		Direction select	0
		0	The H counter counts up to its limit condition, then is cleared to zero.	
		1	The H counter counts up to its limit, then counts down to 0.	
28:21	PRE_H	-	Specifies the factor by which the SCT clock is prescaled to produce the H counter clock. The counter clock is clocked at the rate of the SCT clock divided by PRELH+1. **Remark:** Clear the counter (by writing a 1 to the CLRCTR bit) whenever changing the PRE value.	0
31:29	-		Reserved	

28.6.3 SCT limit register

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers LIMIT_L (address 0x4000 4008) and LIMIT_H (address 0x4000 400A). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

The bits in this register set which events act as counter limits. When a limit event occurs, the counter is cleared to zero in unidirectional mode or begins counting down in bidirectional mode. When the counter reaches all ones, this state is always treated as a limit event, and the counter is cleared in unidirectional mode or, in bidirectional mode, begins counting down on the next clock edge - even if no limit event as defined by the SCT limit register has occurred.

Table 641. SCT limit register (LIMIT - address 0x4000 0008) bit description

Bit	Symbol	Description	Reset value
15:0	LIMMSK_L	If bit n is one, event n is used as a counter limit for the L or unified counter (event 0 = bit 0, event 1 = bit 1, event 15 = bit 15).	0
31:16	LIMMSK_H	If bit n is one, event n is used as a counter limit for the H counter (event 0 = bit 16, event 1 = bit 17, event 15 = bit 31).	0

28.6.4 SCT halt condition register

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers HALT_L (address 0x4000 400C) and HALT_H (address 0x4000 400E). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Remark: Any event halting the counter disables its operation until software clears the HALT bit (or bits) in the CTRL register (Table 640).

Table 642. SCT halt condition register (HALT - address 0x4000 000C) bit description

Bit	Symbol	Description	Reset value
15:0	HALTMSK_L	If bit n is one, event n sets the HALT_L bit in the CTRL register (event 0 = bit 0, event 1 = bit 1, event 15 = bit 15).	0
31:16	HALTMSK_H	If bit n is one, event n sets the HALT_H bit in the CTRL register (event 0 = bit 16, event 1 = bit 17, event 15 = bit 31).	0

28.6.5 SCT stop condition register

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers STOPT_L (address 0x4000 4010) and STOP_H (address 0x4000 4012). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Table 643. SCT stop condition register (STOP - address 0x4000 0010) bit description

Bit	Symbol	Description	Reset value
15:0	STOPMSK_L	If bit n is one, event n sets the STOP_L bit in the CTRL register (event 0 = bit 0, event 1 = bit 1, event 15 = bit 15).	0
31:16	STOPMSK_H	If bit n is one, event n sets the STOP_H bit in the CTRL register (event 0 = bit 16, event 1 = bit 17, event 15 = bit 31).	0

28.6.6 SCT start condition register

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers START_L (address 0x4000 4014) and START_H (address 0x4000 4016). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

The bits in this register select which events, if any, clear the STOP bit in the Control register. (Since no events can occur when HALT is 1, only software can clear the HALT bit by writing the Control register.)

Table 644. SCT start condition register (START - address 0x4000 0014) bit description

Bit	Symbol	Description	Reset value
15:0	STARTMSK_L	If bit n is one, event n clears the STOP_L bit in the CTRL register (event 0 = bit 0, event 1 = bit 1, event 15 = bit 15).	0
31:16	STARTMSK_H	If bit n is one, event n clears the STOP_H bit in the CTRL register (event 0 = bit 16, event 1 = bit 17, event 15 = bit 31).	0

28.6.7 SCT counter register

If UNIFY = 1 in the CONFIG register, the counter is a unified 32-bit register and both the _L and _H bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers COUNT_L (address 0x4000 4040) and COUNT_H (address 0x4000 4042). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation. In this case, the L and H registers count independently under the control of the other registers.

Attempting to write a counter while it is running does not affect the counter but produces a bus error. Software can read the counter registers at any time.

Table 645. SCT counter register (COUNT - address 0x4000 0040) bit description

Bit	Symbol	Description	Reset value
15:0	CTR_L	When UNIFY = 0, read or write the 16-bit L counter value. When UNIFY = 1, read or write the lower 16 bits of the 32-bit unified counter.	0
31:16	CTR_H	When UNIFY = 0, read or write the 16-bit H counter value. When UNIFY = 1, read or write the upper 16 bits of the 32-bit unified counter.	0

28.6.8 SCT state register

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers STATE_L (address 0x4000 4044) and STATE_H (address 0x4000 4046). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Software can read the state associated with a counter at any time. Writing the state is only allowed when the counter HALT bit is 1; when HALT is 0, a write attempt does not change the state and results in a bus error.

The state variable is the main feature that distinguishes the SCT from other counter/timer/PWM blocks. Events can be made to occur only in certain states. Events, in turn, can perform the following actions:

- set and clear outputs
- limit, stop, and start the counter
- cause interrupts and DMA requests
- modify the state variable

The value of a state variable is completely under the control of the application. If an application does not use states, the value of the state variable remains zero, which is the default value.

A state variable can be used to track and control multiple cycles of the associated counter in any desired operational sequence. The state variable is logically associated with a state machine diagram which represents the SCT configuration. See Section 28.6.23 and 28.6.24 for more about the relationship between states and events.

The STATELD/STADEV fields in the event control registers of all defined events set all possible values for the state variable. The change of the state variable during multiple counter cycles reflects how the associated state machine moves from one state to the next.

Table 646. SCT state register (STATE - address 0x4000 0044) bit description

Bit	Symbol	Description	Reset value
4:0	STATE_L	State variable.	0
15:5	-	Reserved.	-
20:16	STATE_H	State variable.	0
31:21	-	Reserved.	

28.6.9 SCT input register

Software can read the state of the SCT inputs in this read-only register in two slightly different forms. The only situation in which these values are different is if CLKMODE = 2 in the CONFIG register.

Table 647. SCT input register (INPUT - address 0x4000 0048) bit description

Bit	Symbol	Description	Reset value
0	AIN0	Real-time status of input 0.	pin
1	AIN1	Real-time status of input 1.	pin
2	AIN2	Real-time status of input 2.	pin
3	AIN3	Real-time status of input 3.	pin
4	AIN4	Real-time status of input 4.	pin
5	AIN5	Real-time status of input 5.	pin
6	AIN6	Real-time status of input 6.	pin
7	AIN7	Real-time status of input 7.	pin
15:8	-	Reserved.	-

Table 647. SCT input register (INPUT - address 0x4000 0048) bit description

Bit	Symbol	Description	Reset value
16	SIN0	Input 0 state synchronized to the SCT clock.	-
17	SIN1	Input 1 state synchronized to the SCT clock.	-
18	SIN2	Input 2 state synchronized to the SCT clock.	-
19	SIN3	Input 3 state synchronized to the SCT clock.	-
20	SIN4	Input 4 state synchronized to the SCT clock.	-
21	SIN5	Input 5 state synchronized to the SCT clock.	-
22	SIN6	Input 6 state synchronized to the SCT clock.	-
23	SIN7	Input 7 state synchronized to the SCT clock.	-
31:24	-	Reserved	-

28.6.10 SCT match/capture registers mode register

If UNIFY = 1 in the CONFIG register, only the _L bits of this register are used. The L bits control whether each set of match/capture registers operates as unified 32-bit capture/match registers.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers REGMODE_L (address 0x4000 404C) and REGMODE_H (address 0x4000 404E). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.The _L bits/registers control the L match/capture registers, and the _H bits/registers control the H match/capture registers.

The SCT contains 16 Match/Capture register pairs. The Register Mode register selects whether each register pair acts as a Match register (see Section 28.6.19) or as a Capture register (see Section 28.6.20). Each Match/Capture register has an accompanying register which serves as a Reload register when the register is used as a Match register (Section 28.6.21) or as a Capture-Control register when the register is used as a capture register (Section 28.6.22). REGMODE_H is used only when the UNIFY bit is 0.

An alternate addressing mode is available for all of the Match/Capture and Reload/Capture-Control registers, for DMA access to halfword registers when UNIFY=0. This mode is described in Section 28.7.9.

Table 648. SCT match/capture registers mode register (REGMODE - address 0x4000 004C) bit description

Bit	Symbol	Description	Reset value
15:0	REGMOD_L	Each bit controls one pair of match/capture registers (register 0 = bit 0, register 1 = bit 1,..., register 15 = bit 15). 0 = registers operate as match registers. 1 = registers operate as capture registers.	0
31:16	REGMOD_H	Each bit controls one pair of match/capture registers (register 0 = bit 16, register 1 = bit 17,..., register 15 = bit 31). 0 = registers operate as match registers. 1 = registers operate as capture registers.	0

28.6.11 SCT output register

The SCT supports 16 outputs, each of which has a corresponding bit in this register.

Software can write to any of the output registers when both counters are halted to control the outputs directly. Writing to this register when either counter is stopped or running does not affect the outputs and results in an bus error.

Software can read this register at any time to sense the state of the outputs.

Table 649. SCT output register (OUTPUT - address 0x4000 0050) bit description

Bit	Symbol	Description	Reset value
15:0	OUT	Writing a 1 to bit n makes the corresponding output HIGH. 0 makes the corresponding output LOW (output 0 = bit 0, output 1 = bit 1,..., output 15 = bit 15).	0
31:16	-	Reserved	

28.6.12 SCT bidirectional output control register

This register specifies (for each output) the impact of the counting direction on the meaning of set and clear operations on the output (see Section 28.6.25 and Section 28.6.26).

Table 650. SCT bidirectional output control register (OUTPUTDIRCTRL - address 0x4000 0054) bit description

Bit	Symbol	Value	Description	Reset value
1:0	SETCLR0		Set/clear operation on output 0. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
3:2	SETCLR1		Set/clear operation on output 1. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
5:4	SETCLR2		Set/clear operation on output 2. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
7:6	SETCLR3		Set/clear operation on output 3. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
9:8	SETCLR4		Set/clear operation on output 4. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	

Table 650. SCT bidirectional output control register (OUTPUTDIRCTRL - address 0x4000 0054) bit description

Bit	Symbol	Value	Description	Reset value
11:10	SETCLR5		Set/clear operation on output 5. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
13:12	SETCLR6		Set/clear operation on output 6. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
15:14	SETCLR7		Set/clear operation on output 7. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
17:16	SETCLR8		Set/clear operation on output 8. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
19:18	SETCLR9		Set/clear operation on output 9. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
21:20	SETCLR10		Set/clear operation on output 5. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
23:22	SETCLR11		Set/clear operation on output 11. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
25:24	SETCLR12		Set/clear operation on output 12. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
27:26	SETCLR13		Set/clear operation on output 13. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	
29:28	SETCLR14		Set/clear operation on output 14. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	

Table 650. SCT bidirectional output control register (OUTPUTDIRCTRL - address 0x4000 0054) bit description

Bit	Symbol	Value	Description	Reset value
31:30	SETCLR15		Set/clear operation on output 15. Value 0x3 is reserved. Do not program this value.	0
		0x0	Set and clear do not depend on any counter.	
		0x1	Set and clear are reversed when counter L or the unified counter is counting down.	
		0x2	Set and clear are reversed when counter H is counting down. Do not use if UNIFY = 1.	

28.6.13 SCT conflict resolution register

The registers OUTPUTSETn (Section 28.6.25) and OUTPUTCLn (Section 28.6.26) allow both setting and clearing to be indicated for an output in the same clock cycle, even for the same event. This SCT conflict resolution register resolves this conflict.

To enable an event to toggle an output, set the OnRES value to 0x3 in this register, and set the event bits in both the Set and Clear registers.

Table 651. SCT conflict resolution register (RES - address 0x4000 0058) bit description

Bit	Symbol	Value	Description	Reset value
1:0	O0RES		Effect of simultaneous set and clear on output 0.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR0 field).	
		0x2	Clear output (or set based on the SETCLR0 field).	
		0x3	Toggle output.	
3:2	O1RES		Effect of simultaneous set and clear on output 1.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR1 field).	
		0x2	Clear output (or set based on the SETCLR1 field).	
		0x3	Toggle output.	
5:4	O2RES		Effect of simultaneous set and clear on output 2.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR2 field).	
		0x2	Clear output n (or set based on the SETCLR2 field).	
		0x3	Toggle output.	
7:6	O3RES		Effect of simultaneous set and clear on output 3.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR3 field).	
		0x2	Clear output (or set based on the SETCLR3 field).	
		0x3	Toggle output.	
9:8	O4RES		Effect of simultaneous set and clear on output 4.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR4 field).	
		0x2	Clear output (or set based on the SETCLR4 field).	
		0x3	Toggle output.	

Table 651. SCT conflict resolution register (RES - address 0x4000 0058) bit description

Bit	Symbol	Value	Description	Reset value
11:10	O5RES		Effect of simultaneous set and clear on output 5.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR5 field).	
		0x2	Clear output (or set based on the SETCLR5 field).	
		0x3	Toggle output.	
13:12	O6RES		Effect of simultaneous set and clear on output 6.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR6 field).	
		0x2	Clear output (or set based on the SETCLR6 field).	
		0x3	Toggle output.	
15:14	O7RES		Effect of simultaneous set and clear on output 7.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR7 field).	
		0x2	Clear output (or set based on the SETCLR7 field).	
		0x3	Toggle output.	
17:16	O8RES		Effect of simultaneous set and clear on output 8.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR8 field).	
		0x2	Clear output (or set based on the SETCLR8 field).	
		0x3	Toggle output.	
19:18	O9RES		Effect of simultaneous set and clear on output 9.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR9 field).	
		0x2	Clear output (or set based on the SETCLR9 field).	
		0x3	Toggle output.	
21:20	O10RES		Effect of simultaneous set and clear on output 10.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR10 field).	
		0x2	Clear output (or set based on the SETCLR10 field).	
		0x3	Toggle output.	
23:22	O11RES		Effect of simultaneous set and clear on output 11.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR11 field).	
		0x2	Clear output (or set based on the SETCLR11 field).	
		0x3	Toggle output.	
25:24	O12RES		Effect of simultaneous set and clear on output 12.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR12 field).	
		0x2	Clear output (or set based on the SETCLR12 field).	
		0x3	Toggle output.	

Table 651. SCT conflict resolution register (RES - address 0x4000 0058) bit description

Bit	Symbol	Value	Description	Reset value
27:26	O13RES		Effect of simultaneous set and clear on output 13.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR13 field).	
		0x2	Clear output (or set based on the SETCLR13 field).	
		0x3	Toggle output.	
29:28	O14RES		Effect of simultaneous set and clear on output 14.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR14 field).	
		0x2	Clear output (or set based on the SETCLR14 field).	
		0x3	Toggle output.	
31:30	O15RES		Effect of simultaneous set and clear on output 15.	0
		0x0	No change.	
		0x1	Set output (or clear based on the SETCLR15 field).	
		0x2	Clear output (or set based on the SETCLR15 field).	
		0x3	Toggle output.	

28.6.14 SCT DMA request 0 and 1 registers

The SCT includes two DMA request outputs. These registers enable the DMA requests to be triggered when a particular event occurs or when counter Match registers are loaded from its Reload registers.

Table 652. SCT DMA 0 request register (DMAREQ0 - address 0x4000 005C) bit description

Bit	Symbol	Description	Reset value
15:0	DEV_0	If bit n is one, event n sets DMA request 0 (event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15).	0
29:16	-	Reserved	-
30	DRL0	A 1 in this bit makes the SCT set DMA request 0 when it loads the Match_L/Unified registers from the Reload_L/Unified registers.	
31	DRQ0	This read-only bit indicates the state of DMA Request 0	

Table 653. SCT DMA 1 request register (DMAREQ1 - address 0x4000 0060) bit description

Bit	Symbol	Description	Reset value
15:0	DEV_1	If bit n is one, event n sets DMA request 1 (event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15).	0
29:16	-	Reserved	-
30	DRL1	A 1 in this bit makes the SCT set DMA request 1 when it loads the Match L/Unified registers from the Reload L/Unified registers.	
31	DRQ1	This read-only bit indicates the state of DMA Request 1.	

28.6.15 SCT flag enable register

This register enables flags to request an interrupt if the FLAGn bit in the SCT event flag register (Section 28.6.16) is also set.

Table 654. SCT flag enable register (EVEN - address 0x4000 00F0) bit description

Bit	Symbol	Description	Reset value
15:0	IEN	The SCT requests interrupt when bit n of this register and the event flag register are both one (event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15).	0
31:16	-	Reserved	

28.6.16 SCT event flag register

This register records events. Writing ones to this register clears the corresponding flags and negates the SCT interrupt request if all enabled Flag bits are zero.

Table 655. SCT event flag register (EVFLAG - address 0x4000 00F4) bit description

Bit	Symbol	Description	Reset value
15:0	FLAG	Bit n is one if event n has occurred since reset or a 1 was last written to this bit (event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15).	0
31:16	-	Reserved	-

28.6.17 SCT conflict enable register

This register enables the “no change conflict” events specified in the SCT conflict resolution register to request an IRQ.

Table 656. SCT conflict enable register (CONEN - address 0x4000 00F8) bit description

Bit	Symbol	Description	Reset value
15:0	NCEN	The SCT requests interrupt when bit n of this register and the SCT conflict flag register are both one (output 0 = bit 0, output 1 = bit 1,..., output 15 = bit 15).	0
31:16	-	Reserved	

28.6.18 SCT conflict flag register

This register records interrupt-enabled no-change conflict events and provides details of a bus error. Writing ones to the NCFLAG bits clears the corresponding read bits and negates the SCT interrupt request if all enabled Flag bits are zero.

Table 657. SCT conflict flag register (CONFLAG - address 0x4000 00FC) bit description

Bit	Symbol	Description	Reset value
15:0	NCFLAG	Bit n is one if a no-change conflict event occurred on output n since reset or a 1 was last written to this bit (output 0 = bit 0, output 1 = bit 1,..., output 15 = bit 15).	0
29:16	-	Reserved.	-
30	BUSERRL	The most recent bus error from this SCT involved writing CTR L/Unified, STATE L/Unified, MATCH L/Unified, or the Output register when the L/U counter was not halted. A word write to certain L and H registers can be half successful and half unsuccessful.	0
31	BUSERRH	The most recent bus error from this SCT involved writing CTR H, STATE H, MATCH H, or the Output register when the H counter was not halted.	0

28.6.19 SCT match registers 0 to 15 (REGMODEn bit = 0)

Match registers are compared to the counters to help create events. When the UNIFY bit is 0, the L and H registers are independently compared to the L and H counters. When UNIFY is 1, the L and H registers hold a 32-bit value that is compared to the unified counter. A Match can only occur in a clock in which the counter is running (STOP and HALT are both 0).

Match registers can be read at any time. Writing to a Match register while the associated counter is running does not affect the Match register and results in a bus error. Match events occur in the SCT clock in which the counter is (or would be) incremented to the next value. When a Match event limits its counter as described in Section 28.6.3, the value in the Match register is the last value of the counter before it is cleared to zero (or decremented if BIDIR is 1).

There is no “write-through” from Reload registers to Match registers. Before starting a counter, software can write one value to the Match register used in the first cycle of the counter and a different value to the corresponding Match Reload register used in the second cycle.

Table 658. SCT match registers 0 to 15 (MATCH - address 0x4000 0100 (MATCH0) to 0x4000 4013C (MATCH15)) bit description (REGMODEn bit = 0)

Bit	Symbol	Description	Reset value
15:0	MATCHn_L	When UNIFY = 0, read or write the 16-bit value to be compared to the L counter. When UNIFY = 1, read or write the lower 16 bits of the 32-bit value to be compared to the unified counter.	0
31:16	MATCHn_H	When UNIFY = 0, read or write the 16-bit value to be compared to the H counter. When UNIFY = 1, read or write the upper 16 bits of the 32-bit value to be compared to the unified counter.	0

28.6.20 SCT capture registers 0 to 15 (REGMODEn bit = 1)

These registers allow software to read the counter values at which the event selected by the corresponding Capture Control registers occurred.

Table 659. SCT capture registers 0 to 15 (CAP - address 0x4000 0100 (CAP0) to 0x4000 013C (CAP15)) bit description (REGMODEn bit = 1)

Bit	Symbol	Description	Reset value
15:0	CAPn_L	When UNIFY = 0, read the 16-bit counter value at which this register was last captured. When UNIFY = 1, read the lower 16 bits of the 32-bit value at which this register was last captured.	0
31:16	CAPn_H	When UNIFY = 0, read the 16-bit counter value at which this register was last captured. When UNIFY = 1, read the upper 16 bits of the 32-bit value at which this register was last captured.	0

28.6.21 SCT match reload registers 0 to 15 (REGMODEn bit = 0)

A Match register (L, H, or unified 32-bit) is loaded from the corresponding Reload register when BIDIR is 0 and the counter reaches its limit condition, or when BIDIR is 1 and the counter reaches 0.

Table 660. SCT match reload registers 0 to 15 (MATCHREL- address 0x4000 0200 (MATCHRELOAD0) to 0x4000 023C (MATCHRELOAD15) bit description (REGMODEn bit = 0)

Bit	Symbol	Description	Reset value
15:0	RELOADn_L	When UNIFY = 0, read or write the 16-bit value to be loaded into the SCTMATCHn_L register. When UNIFY = 1, read or write the lower 16 bits of the 32-bit value to be loaded into the MATCHn register.	0
31:16	RELOADn_H	When UNIFY = 0, read or write the 16-bit to be loaded into the MATCHn_H register. When UNIFY = 1, read or write the upper 16 bits of the 32-bit value to be loaded into the MATCHn register.	0

28.6.22 SCT capture control registers 0 to 15 (REGMODEn bit = 1)

If UNIFY = 1 in the CONFIG register, only the _L bits are used.

If UNIFY = 0 in the CONFIG register, this register can be written to as two registers CAPCTRLn_L (address 0x4000 4100 to 0x4000 413C) and CAPCTRLn_H (address 0x4000 4102 to 0x4000 413E). Both the L and H registers can be read or written individually or in a single 32-bit read or write operation.

Each Capture Control register (L, H, or unified 32-bit) controls which events load the corresponding Capture register from the counter.

Table 661. SCT capture control registers 0 to 15 (CAPCTRL- address 0x4000 0200 (CAPCTRL0) to 0x4000 023C (CAPCTRL15)) bit description (REGMODEn bit = 1)

Bit	Symbol	Description	Reset value
15:0	CAPCONn_L	If bit m is one, event m causes the CAPn_L (UNIFY = 0) or the CAPn (UNIFY = 1) register to be loaded (event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15).	0
31:16	CAPCONn_H	If bit m is one, event m causes the CAPn_H (UNIFY = 0) register to be loaded (event 0 = bit 16, event 1 = bit 17,..., event 15 = bit 31).	0

28.6.23 SCT event state mask registers 0 to 15

Each event has one associated SCT event state mask register that allow this event to happen in one or more states of the counter selected by the HEVENT bit in the corresponding EVCTRLn register.

An event n is disabled when its EVSTATEMSKn register contains all zeros, since it is masked regardless of the current state.

In simple applications that do not use states, write 0x01 to this register to enable an event. Since the state always remains at its reset value of 0, writing 0x01 effectively permanently state-enables this event.

Table 662. SCT event state mask registers 0 to 15 (EVSTATEMSK - addresses 0x4000 0300 (EVSTATEMSK0) to 0x4000 0378 (EVSTATEMSK15)) bit description

Bit	Symbol	Description	Reset value
31:0	STATEMSKn	If bit m is one, event n (n= 0 to 15) happens in state m of the counter selected by the HEVENT bit (m = state number; state 0 = bit 0, state 1= bit 1,..., state 31 = bit 31).	0

28.6.24 SCT event control registers 0 to 15

This register defines the conditions for event n to occur, other than the state variable which is defined by the state mask register. Most events are associated with a particular counter (high, low, or unified), in which case the event can depend on a match to that register. The other possible ingredient of an event is a selected input or output signal.

When the UNIFY bit is 0, each event is associated with a particular counter by the HEVENT bit in its event control register. An event cannot occur when its related counter is halted nor when the current state is not enabled to cause the event as specified in its event mask register. An event is permanently disabled when its event state mask register contains all 0s.

An enabled event can be programmed to occur based on a selected input or output edge or level and/or based on its counter value matching a selected match register.

Each event can modify its counter STATE value. If more than one event associated with the same counter occurs in a given clock cycle, only the state change specified for the highest-numbered event among them takes place. Other actions dictated by any simultaneously occurring events all take place.

Table 663. SCT event control register 0 to 15 (EVCTRL - address 0x4000 0304 (EVCTRL0) to 0x4000 037C (EVCTRL15)) bit description

Bit	Symbol	Value	Description	Reset value
3:0	MATCHSEL	-	Selects the Match register associated with this event (if any). A match can occur only when the counter selected by the HEVENT bit is running.	0
4	HEVENT		Select L/H counter. Do not set this bit if UNIFY = 1.	0
		0	Selects the L state and the L match register selected by MATCHSEL.	
		1	Selects the H state and the H match register selected by MATCHSEL.	

Table 663. SCT event control register 0 to 15 (EVCTRL - address 0x4000 0304 (EVCTRL0) to 0x4000 037C (EVCTRL15)) bit description

Bit	Symbol	Value	Description	Reset value
5	OUTSEL		Input/output select	0
		0	Selects the output selected by IOSEL.	
		1	Selects the input selected by IOSEL.	
9:6	IOSEL	-	Selects the input or output signal associated with this event (if any). Do not select an input in this register, if CKMODE is 1x. In this case the clock input is an implicit ingredient of every event.	0
11:10	IOCOND		Selects the I/O condition for event n. (The detection of edges on outputs lag the conditions that switch the outputs by one SCT clock). In order to guarantee proper edge/state detection, an input must have a minimum pulse width of at least one SCT clock period .	0
		0x0	LOW	
		0x1	Rise	
		0x2	Fall	
		0x3	HIGH	
13:12	COMBMODE		Selects how the specified match and I/O condition are used and combined.	
		0x0	OR. The event occurs when either the specified match or I/O condition occurs.	
		0x1	MATCH. Uses the specified match only.	
		0x2	IO. Uses the specified I/O condition only.	
		0x3	AND. The event occurs when the specified match and I/O condition occur simultaneously.	
14	STATELD		This bit controls how the STATEV value modifies the state selected by HEVENT when this event is the highest-numbered event occurring for that state.	
		0	STATEV value is added into STATE (the carry-out is ignored).	
		1	STATEV value is loaded into STATE.	
19:15	STATEV		This value is loaded into or added to the state selected by HEVENT, depending on STATELD, when this event is the highest-numbered event occurring for that state. If STATELD and STATEV are both zero, there is no change to the STATE value.	
31:20	-		Reserved	

28.6.25 SCT output set registers 0 to 15

Each output n has one set register that controls how events affect each output. Whether outputs are set or cleared depends on the setting of the SETCLRn field in the SCTOUTPUTDIRCTRL register.

Table 664. SCT output set register 0 to 15 (OUTPUTSET - address 0x4000 0500 (OUTPUTSET0) to 0x4000 0578 (OUTPUTSET15)) bit description

Bit	Symbol	Description	Reset value
15:0	SET	A 1 in bit m selects event m to set output n (or clear it if SETCLRn = 0x1 or 0x2) event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15.	0
31:16	-	Reserved	

28.6.26 SCT output clear registers 0 to 15

Each output n has one clear register that controls how events affect each output. Whether outputs are set or cleared depends on the setting of the SETCLRn field in the OUTPUTDIRCTRL register.

Table 665. SCT output clear register 0 to 15 (OUTPUTCL - address 0x4000 0504 (OUTPUTCL0) to 0x4000 057C (OUTPUTCL15)) bit description

Bit	Symbol	Description	Reset value
15:0	CLR	A 1 in bit m selects event m to clear output n (or set it if SETCLRn = 0x1 or 0x2) event 0 = bit 0, event 1 = bit 1,..., event 15 = bit 15.	0
31:16	-	Reserved	

28.7 Functional description

28.7.1 Match logic

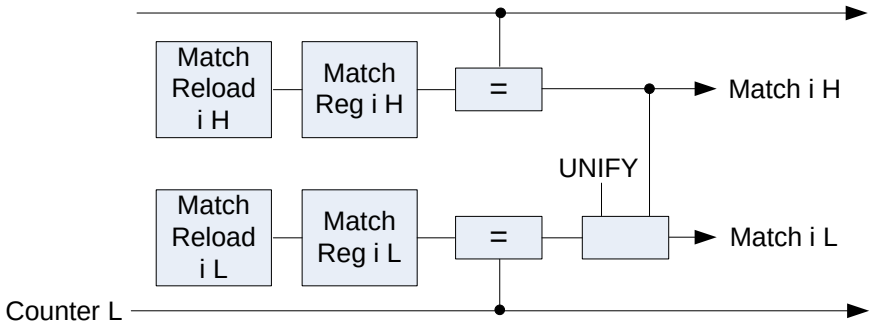


Fig 84. Match logic

28.7.2 Capture logic

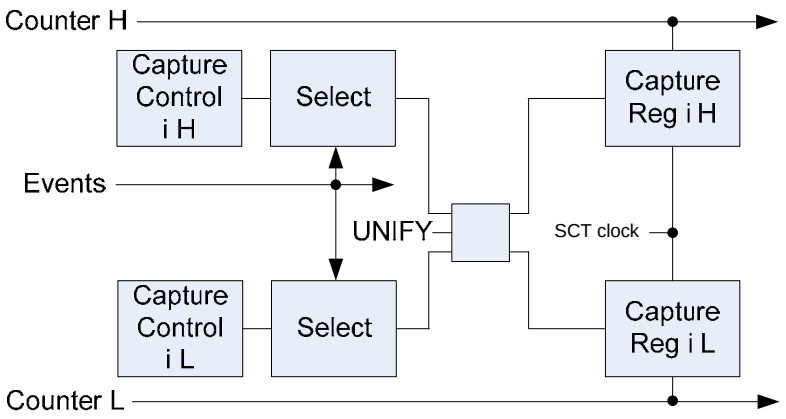


Fig 85. Capture logic

28.7.3 Event selection

State variables allow control of the SCT across more than one cycle of the counter. Counter matches, input/output edges, and state values are combined into a set of general-purpose events that can switch outputs, request interrupts, and change state values.

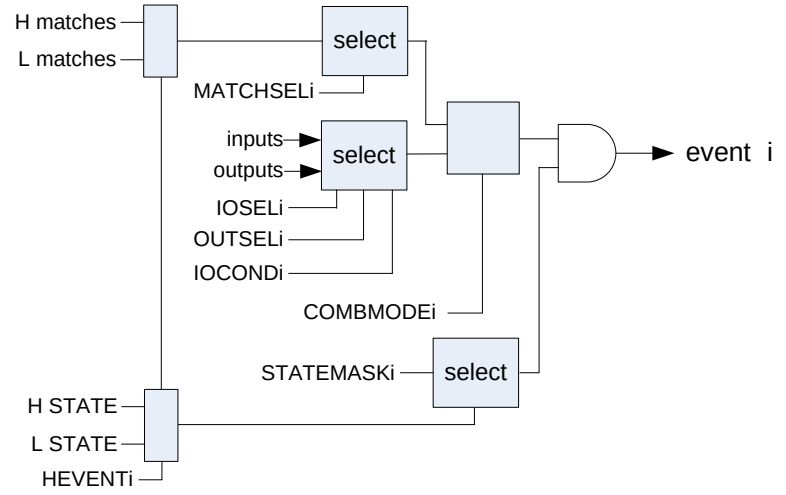


Fig 86. Event selection

28.7.4 Output generation

Figure 87 shows one output slice of the SCT.

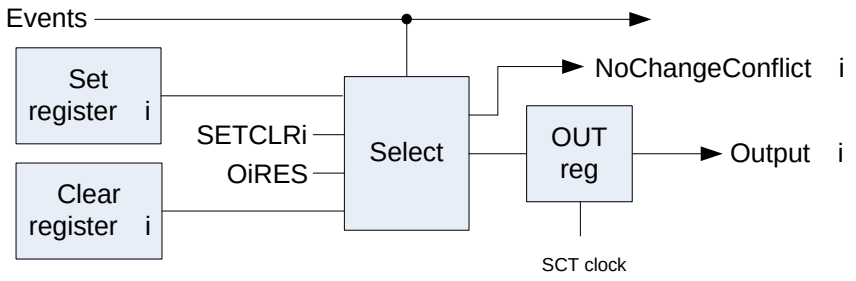


Fig 87. Output slice i

28.7.5 Interrupt generation

The SCT generates one interrupt to the NVIC.

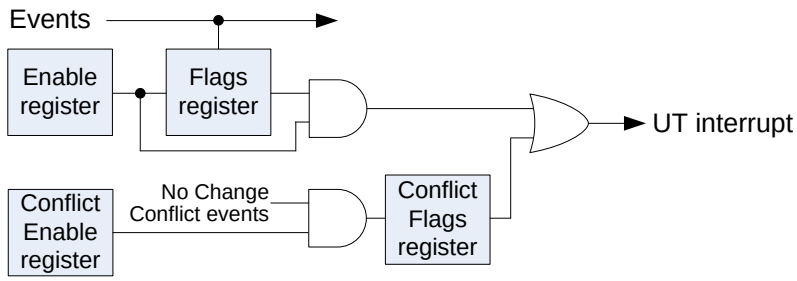


Fig 88. SCT interrupt generation

28.7.6 Clearing the prescaler

When enabled by a non-zero PRE field in the Control register, the prescaler acts as a clock divider for the counter, like a fractional part of the counter value. The prescaler is cleared whenever the counter is cleared or loaded for any of the following reasons:

- Hardware reset
- Software writing to the counter register
- Software writing a 1 to the CLRCTR bit in the control register
- an event selected by a 1 in the counter limit register when BIDIR = 0

When BIDIR is 0, a limit event caused by an I/O signal can clear a non-zero prescaler. However, a limit event caused by a Match only clears a non-zero prescaler in one special case as described Section 28.7.7.

A limit event when BIDIR is 1 does not clear the prescaler. Rather it clears the DOWN bit in the Control register, and decrements the counter on the same clock if the counter is enabled in that clock.

28.7.7 Match vs. I/O events

Counter operation is complicated by the prescaler and by clock mode 01 in which the SCT clock is the bus clock. However, the prescaler and counter are enabled to count only when a selected edge is detected on a clock input.

- The prescaler is enabled when the clock mode is not 01, or when the input edge selected by the CLKSEL field is detected.
- The counter is enabled when the prescaler is enabled, and (PRELIM=0 or the prescaler is equal to the value in PRELIM).

An I/O component of an event can occur in any SCT clock when its counter HALT bit is 0. In general, a Match component of an event can only occur in a UT clock when its counter HALT and STOP bits are both 0 and the counter is enabled.

Table 666 shows when the various kinds of events can occur.

Table 666. Event conditions

COMBMODE	IOMODE	Event can occur on clock:
IO	Any	Event can occur whenever HALT = 0 (type A).
MATCH	Any	Event can occur when HALT = 0 and STOP = 0 and the counter is enabled (type C).
OR	Any	From the IO component: Event can occur whenever HALT = 0 (A). From the match component: Event can occur when HALT = 0 and STOP = 0 and the counter is enabled (C).
AND	LOW or HIGH	Event can occur when HALT = 0 and STOP = 0 and the counter is enabled (C).
AND	RISE or FALL	Event can occur whenever HALT = 0 (A).

28.7.8 DMA operation

A DMA controller can be used to write one or more Reload registers, or read one or more Capture registers, typically at the start of a counter cycle. DMA access to more than one Reload or Capture register requires that they be consecutive registers. (Nothing else in the SCT constrains how these registers are assigned and used.)

An event can set a DMA request or set when a counter Match registers are loaded from its Reload registers, as described in Section 28.6.14. The two requests of the SCT can be used to do the same register access for both counters when UNIFY is 0. Alternatively, one request can be used for writing Reload registers and the other for reading Capture registers.

The SCT does not know how many transfers are done for each request, so it cannot control its DMA requests accordingly.

The two DMA requests are connected to DMABREQ7 and DMABREQ8. Write the number of registers to be transferred for each request to the TransferSize field in the Channel Control Register of the DMA channel to which the request is connected. If the Linked List feature is used, there is a TransferSize value in each Linked List entry. The GPDMA asserts the DMACCLR signal when that number of transfers has been completed, which makes the SCT clear the request.

28.7.9 Alternate addressing for match/capture registers

The Match, Reload, Capture, and Capture Control registers are arranged as consecutive words, with the standard division of each word into two halfwords. When the UNIFY bit is zero, these two halfwords are related to the L and H counters. Software has the option of writing words initially to set up both halves of a SCT simultaneously, or writing halfwords to set up each half separately.

Applications can use a DMA controller to write Reload registers or to read Capture registers. However, when UNIFY is 0, the addressing of the halfword registers is not compatible with the requirement of many DMA controllers to use consecutive addresses for sequential address operation. Table 667 shows how the second half of the range occupied by each type of register contains an alternate address map for halfword accesses to the same registers, which is compatible with DMA that use sequential address operation. When UNIFY is 1, perform DMA word accesses using standard offsets.

Table 667. Alternate address map for DMA halfword access

Match register	Capture register	Standard offset	DMA halfword offset
MATCH0_L	CAP0_L	0x100	0x180
MATCH0_H	CAP0_H	0x102	0x1C0
MATCH1_L	CAP1_L	0x104	0x182
MATCH1_H	CAP1_H	0x106	0x1C2
...	...	...	...
MATCHREL0_L	CAPCTRL0_L	0x200	0x280
MATCHREL0_H	CAPCTRL0_H	0x202	0x2C0
MATCHREL1_L	CAPCTRL1_L	0x204	0x282
MATCHREL1_H	CAPCTRL1_H	0x206	0x2C2
...	...	...	...

28.7.10 SCT operation

In its simplest, single-state configuration, the SCT operates as an event controlled one- or bidirectional counter. Events can be configured to be counter match events, an input or output level, transitions on an input or output pin, or a combination of match and input/output behavior. In response to an event, the SCT output or outputs can transition, or the SCT can perform other actions such as creating an interrupt or starting, stopping, or resetting the counter. Multiple simultaneous actions are allowed for each event. Furthermore, any number of events can trigger one specific action of the SCT.

An action or multiple actions of the SCT uniquely define an event. A state is defined by which events are enabled to trigger an SCT action or actions in any stage of the counter. Events not selected for this state are ignored.

In a multi-state configuration, states change in response to events. A state change is an additional action that the SCT can perform when the event occurs. When an event is configured to change the state, the new state defines a new set of events resulting in different actions of the SCT. Through multiple cycles of the counter, events can change the state multiple times and thus create a large variety of event controlled transitions on the SCT outputs and/or interrupts.

Once configured, the SCT can run continuously without software intervention and can generate multiple output patterns entirely under the control of events.

- To configure the SCT, see Section 28.7.10.1.
- To start, run, and stop the SCT, see Section 28.7.10.2.
- To configure the SCT as simple event controlled counter/timer, see Section 28.7.10.3.

28.7.10.1 Configure the SCT

To set up the SCT for multiple events and states, perform the following configuration steps:

28.7.10.1.1 Configure the counter

1. Configure the L and H counters in the CONFIG register by selecting two independent 16-bit counters (L counter and H counter) or one combined 32-bit counter in the UNIFY field.
2. Select the SCT clock source in the CONFIG register (fields CLKMODE and CLKSEL) from any of the inputs or an internal clock.

28.7.10.1.2 Configure the match and capture registers

1. Select how many match and capture registers the application uses (total of up to 16):
   - In the REGMODE register, select for each of the 16 match/capture register pairs whether the register is used as a match register or capture register.
2. Define match conditions for each match register selected:
   - Each match register MATCH sets one match value, if a 32-bit counter is used, or two match values, if the L and H 16-bit counters are used.
   - Each match reload register MATCHRELOAD sets a reload value that is loaded into the match register when the counter reaches a limit condition or the value 0.

28.7.10.1.3 Configure events and event responses

1. Define when each event can occur in the following way in the EVCTRL registers (up to 16, one register per event):
   - Select whether the event occurs on an input or output changing, on an input or output level, a match condition of the counter, or a combination of match and input/output conditions in field COMBMODE.
   - For a match condition:

     Select the match register that contains the match condition for the event to occur. Enter the number of the selected match register in field MATCHSEL.

     If using L and H counters, define whether the event occurs on matching the L or the H counter in field HEVENT.
   - For an SCT input or output level or transition:

     Select the input number or the output number that is associated with this event in fields IOSEL and OUTSEL.

     Define how the selected input or output triggers the event (edge or level sensitive) in field IOCOND.
2. Define what the effect of each event is on the SCT outputs in the OUTPUTSET or OUTPUTCLR registers (up to 16 outputs, one register per output):
   - For each SCT output, select which events set or clear this output. More than one event can change the output, and each event can change multiple outputs.
3. Define how each event affects the counter:
   - Set the corresponding event bit in the LIMIT register for the event to set an upper limit for the counter.

     When a limit event occurs in unidirectional mode, the counter is cleared to zero and begins counting up on the next clock edge.

     When a limit event occurs in bidirectional mode, the counter begins to count down from the current value on the next clock edge.
   - Set the corresponding event bit in the HALT register for the event to halt the counter. If the counter is halted, it stops counting and no new events can occur. The counter operation can only be restored by clearing the HALT_L and/or the HALT_H bits in the CTRL register.
   - Set the corresponding event bit in the STOP register for the event to stop the counter. If the counter is stopped, it stops counting. However, an event that is configured as a transition on an input/output can restart the counter.
   - Set the corresponding event bit in the START register for the event to restart the counting. Only events that are defined by an input changing can be used to restart the counter.
4. Define which events contribute to the SCT interrupt:
   - Set the corresponding event bit in the EVEN and the EVFLAG registers to enable the event to contribute to the SCT interrupt.
5. Define whether an event triggers a DMA request.
   - Set the corresponding event bit in the DMAREQ0/1 registers for the event to trigger DMA requests 0 or 1.

28.7.10.1.4 Configure multiple states

1. In the EVSTATEMASK register for each event (up to 16 events, one register per event), select the state or states (up to 31) in which this event is allowed to occur. Each state can be selected for more than one event.
2. Determine how the event affects the system state:

   In the EVCTRL registers (up to 16 events, one register per event), set the new state value in the STATEV field for this event. If the event is the highest numbered in the current state, this value is either added to the existing state value or replaces the existing state value, depending on the field STATELD.

   Remark: If there are higher numbered events in the current state, this event cannot change the state.

   If the STATEV and STATELD values are set to zero, the state does not change.

28.7.10.1.5 Miscellaneous options

- There are a certain (selectable) number of capture registers. Each capture register can be programmed to capture the counter contents when one or more events occur.
- If the counter is in bidirectional mode, the effect of set and clear of an output can be made to depend on whether the counter is counting up or down by writing to the OUTPUTDIRCTRL register.
- <tbd>

28.7.10.2 Operate the SCT

1. Configure the SCT (see Section 28.7.10.1 “Configure the SCT”).
   a. Configure the counter (see Section 28.7.10.1.1).
   b. Configure the match and capture registers (see Section 28.7.10.1.2).
   c. Configure the events and event responses (see Section 28.7.10.1.3).
   d. Configure multiple states (Section 28.7.10.1.4).
2. Write to the STATE register to define the initial state. By default the initial state is state 0.
3. To start the SCT, write to the CTRL register:
   – Clear the counters.
   – Clear or set the STOP_L and/or STOP_H bits.

     Remark: The counter starts counting once the STOP bit is cleared as well. If the STOP bit is set, the SCT waits instead for an event to occur that is configured to start the counter.
   – For each counter, select unidirectional or bidirectional counting mode (field BIDIR_L and/or BIDIR_H).
   – Select the prescale factor for the counter clock (CTRL register).
   – Clear the HALT_L and/or HALT_H bit. By default, the counters are halted and no events can occur.
4. To stop the counters by software at any time, stop or halt the counter (write to STOP_L and/or STOP_H bits or HALT_L and/or HALT_H bits in the CTRL register).

- When the counters are stopped, both an event configured to clear the STOP bit or software writing a zero to the STOP bit can start the counter again.
- When the counter are halted, only a software write to clear the HALT bit can start the counter again. No events can occur.
- When the counters are halted, software can set any SCT output HIGH or LOW directly by writing to the OUT register.

The current state can be read at any time by reading the STATE register.

To change the current state by software (that is independently of any event occurring), set the HALT bit and write to the STATE register to change the state value. Writing to the STATE register is only allowed when the counter is halted (the HALT_L and/or HALT_H bits are set) and no events can occur.

28.7.10.3 Configure the SCT without using states

The SCT can be used as standard counter/timer with external capture inputs and match outputs without using the state logic. To operate the SCT without states, configure the SCT as follows:

- Write zero to the STATE register (zero is the default).
- Write zero to the STATELD and STATEV fields in the EVCTRL registers for each event.
- Write 0x1 to the EVSTATEMASK register of each event. Writing 0x1 enables the event.

  In effect, the event is allowed to occur in a single state which never changes while the counter is running.

28.7.10.4 Example

Figure 89 shows a simple application of the SCT using two sets of match events (EV0/1 and EV3/4) to set/clear SCT output 0. A third match event (EV2) is used to reset the counter regardless of the current state.

In the initial state 0, match event EV0 sets output 0 to HIGH and match event EV1 clears output 0. The SCT input 0 is monitored: If the input transitions from HIGH to LOW (EV2), the state is changed to state 1, and EV3/4 are enabled, which create the same output but triggered by different match values. If input 0 transitions from LOW to HIGH, the associated event (EV5) causes the state to change back to state 0. In state 0, the events EV0 and EV1 are enabled.

The example uses the following SCT configuration:

- 1 input
- 1 output
- 5 match registers
- 7 events
- 2 states

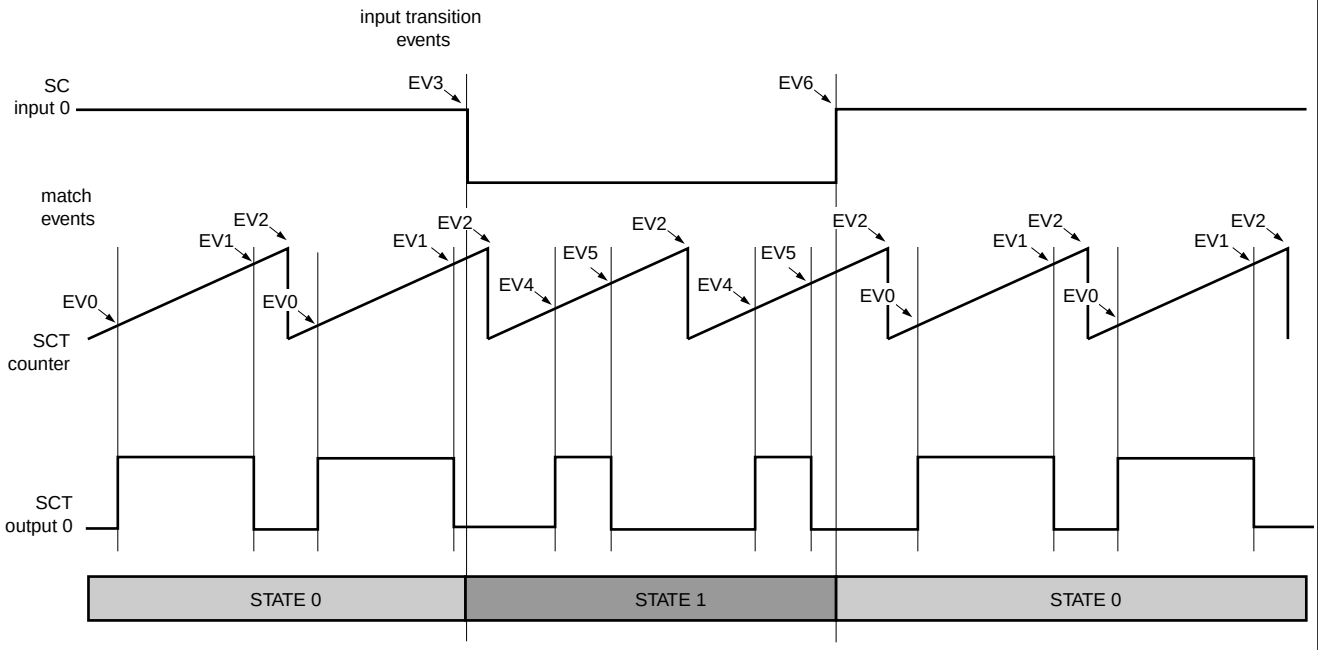


Fig 89. SCT configuration example

This application of the SCT uses the following configuration (all register values not listed in Table 668 are set to their default values):

Table 668. SCT configuration example

Configuration	Registers	Setting
Counter	CONFIG	Uses one counter (UNIFY = 1).
	CTRL	Uses unidirectional counter (BIDIR_L = 0).
Clock base	CONFIG	Uses default values for clock configuration.
Match/Capture registers	REGMODE	Configure one match register for each match event by setting REGMODE_L bits 0,1, 2, 4, 5 to 0. This is the default.
Define match values	MATCH0/1/2/4/5	Set a match value MATCH0/1/2/4/5_L in each register.
Define match reload values	MATCHREL0/1/2/4/5	Set a match reload value RELOAD0/1/2/4/5_L in each register (same as the match value in this example).
Define when event 0 occurs	EVCTRL0	• Set COMBMODE = 0x1. Event 0 uses match condition only. • Set MATCHSEL = 0. Select match value of match register 0.
Define when event 1 occurs	EVCTRL1	• Set COMBMODE = 0x1. Event 1 uses match condition only. • Set MATCHSEL = 1. Select match value of match register 1.
Define when event 2 occurs	EVCTRL2	• Set COMBMODE = 0x1. Event 2 uses match condition only. • Set MATCHSEL = 2. Select match value of match register 2.
Define when event 3 occurs	EVCTRL3	• Set COMBMODE = 0x2. Event 3 uses I/O condition only. • Set IOSEL = 0. Select input 0. • Set IOCOND = 0x2. Input 0 goes LOW.
Define how event 3 changes the state	EVCTRL3	Set STATEV bits to 1 and the STATED bit to 1. Event 3 changes the state to state 1.

Table 668. SCT configuration example

Configuration	Registers	Setting
Define when event 4 occurs	EVCTRL4	• Set COMBMODE = 0x1. Event 4 uses match condition only. • Set MATCHSEL = 0x3. Select match value of match register 4.
Define when event 5 occurs	EVCTRL5	• Set COMBMODE = 0x1. Event 5 uses match condition only. • Set MATCHSEL = 0x3. Select match value of match register 5.
Define when event 6 occurs	EVCTRL6	• Set COMBMODE = 0x2. Event 6 uses I/O condition only. • Set IOSEL = 0. Select input 0. • Set IOCOND = 0x1. Input 0 goes HIGH.
Define how event 6 changes the state	EVCTRL6	Set STATEV bits to 0 and the STATED bit to 1. Event 6 changes the state to state 0.
Define by which events output 0 is set	OUTPUTSET0	Set SET0 bits 0 (for event 0) and 4 (for event 4) to one to set the output when these events 0 and 4 occur.
Define by which events output 0 is cleared	OUTPUTCLR0	Set CLR0 bits 1 (for events 1) and 5 (for event 5) to one to clear the output when events 1 and 5 occur.
Define which event resets the counter	LIMIT	Set LIMMASK_L bit 2 to 1 (for event 2 to limit the counter). Set all other bits to zero.
Configure states event 0 is enabled	EVSTATEMSK0	Set STATEMSK0 bit 0 to 1. Set all other bits to 0. Event 0 is enabled in state 0.
Configure states event 1 is enabled	EVSTATEMSK1	Set STATEMSK1 bit 0 to 1. Set all other bits to 0. Event 1 is enabled in state 0.
Configure states event 2 is enabled	EVSTATEMSK2	Set STATEMSK2 bit 0 to 1 and bit 1 to 1. Set all other bits to 0. Event 2 is enabled in state 0 and state 1.
Configure states event 3 is enabled	EVSTATEMSK3	Set STATEMSK3 bit 0 to 1. Set all other bits to 0. Event 3 is enabled in state 0.
Configure states event 4 is enabled	EVSTATEMSK4	Set STATEMSK4 bit 1 to 1. Set all other bits to 0. Event 4 is enabled in state 1.
Configure states event 5 is enabled	EVSTATEMSK5	Set STATEMSK5 bit 1 to 1. Set all other bits to 0. Event 5 is enabled in state 1.
Configure states event 6 is enabled	EVSTATEMSK6	Set STATEMSK6 bit 1 to 1. Set all other bits to 0. Event 6 is enabled in state 1.

29.1 How to read this chapter

The timers are available on all LPC43xx parts.

29.2 Basic configuration

The Timers are configured as follows:

- See Table 669 for clocking and power control.
- The Timer0/1/2/3 are reset by the TIMER0/1/2/3_RST (reset #32/33/34/35).
- The Timer0/1/2/3 interrupts are connected to slot # 12/13/14/15 in the NVIC. Match channels 2 of Timer0/1/3 are connected to slots # 13, 14, 16 in the Event router. (These outputs are ORed with SCT outputs 2, 6, 14.)
- For connecting the match channels 0 and 1 of Timer0/1/2/3 to the GPDMA, use the DMAMUX register in the CREG block (see Table 41) and enable the GPDMA channel in the DMA Channel Configuration registers (Section 19.6.20).
- The timer registers can be also accessed by the GPDMA as memory-to-memory transfer.
- The timer capture inputs and match outputs are configured through the GIMA (see Section 16.3).
- All timer capture inputs are also connected to dedicated external pins (see Section 16.3 and Section 15.3.11).

Table 669. Timer0/1/2/3 clocking and power control

	Base clock	Branch clock	Operating frequenct
Clock to the timer0 register interface and timer0 peripheral clock PCLK.	BASE_M4_CLK	CLK_M4_TIMER0	up to 204 MHz
Clock to the timer1 register interface and timer1 peripheral clock PCLK.	BASE_M4_CLK	CLK_M4_TIMER1	up to 204 MHz
Clock to the timer2 register interface and timer2 peripheral clock PCLK.	BASE_M4_CLK	CLK_M4_TIMER2	up to 204 MHz
Clock to the timer3 register interface and timer3 peripheral clock PCLK.	BASE_M4_CLK	CLK_M4_TIMER3	up to 204 MHz

29.3 Features

- A 32 bit Timer/Counter with a programmable 32 bit Prescaler.
- Counter or Timer operation
- Up to four 32 bit capture channels per timer, that can take a snapshot of the timer value when an input signal transitions. A capture event may also optionally generate an interrupt.
- Four 32 bit match registers that allow:

  - Continuous operation with optional interrupt generation on match.
  - Stop timer on match with optional interrupt generation.
  - Reset timer on match with optional interrupt generation.
- Up to four external outputs corresponding to match registers, with the following capabilities:
  - Set low on match.
  - Set high on match.
  - Toggle on match.
  - Do nothing on match.

29.4 General description

The Timer/Counter is designed to count cycles of the peripheral clock (PCLK) or an externally-supplied clock, and can optionally generate interrupts or perform other actions at specified timer values, based on four match registers. It also includes four capture inputs to trap the timer value when an input signal transitions, optionally generating an interrupt.

Table 670 gives a brief summary of each of the Timer/Counter related functions.

Table 670. Timer/Counter function description

Pin	Type	Description
CAP0_[3:0] CAP1_[3:0] CAP2_[3:0] CAP3_[3:0]	Input	Capture Signals- A transition on a capture input can be configured to load one of the Capture Registers with the value in the Timer Counter and optionally generate an interrupt. Capture functionality can be selected from a number of pins. Timer/Counter block can select a capture signal as a clock source instead of the PCLK derived clock. For more details see Section 29.7.11.
MAT0_[3:0] MAT1_[3:0] MAT2_[3:0] MAT3_[3:0]	Output	External Match Output - When a match register (MR3:0) equals the timer counter (TC) this output can either toggle, go LOW, go HIGH, or do nothing. The External Match Register (EMR) controls the functionality of this output. Match Output functionality can be selected on a number of pins in parallel.

29.5 Pin description

Input signals to each timer capture channel can originate from the external pins or from several other internal sources. The GIMA (see Table 138) and (for capture channel 3 of each timer) the CTOUTCTRL bit of CREG6 determine which signal is captured by the timer.

The match outputs are connected to the Tn_MATm pin functions. In addition, the match outputs ORed with the SCT outputs can be monitored on the CTOUT pins provided that the CTOUTCTRL bit is set to 0 (default) in the CREG6 register (see Table 43).

Table 671. Timer0/1/2/3 inputs and outputs

Description	Pin function	Internal signal	Default (see GIMA, Table 138)	CTOUTCTRL bit (see Table 43)
Timer0 inputs				
Timer 0, input to capture channel 0	CTIN_0	-	yes	-
	T0_CAP0	-	no	-
Timer 0, input to capture channel 1	CTIN_1	-	yes	-
	T0_CAP1	-	no	-
	-	USART2 TX active	no	-
Timer 0, input to capture channel 2	CTIN_2	-	yes	-
	T0_CAP2	-	no	-
Timer 0, input to capture channel 3	-	SCT output 15 OR T3 match channel 3	yes	0
	-	SCT output 15	yes	1
	T0_CAP3	-	no	-
	-	T3 match channel 3	no	-
Timer0 outputs				
Timer 0; match output channel 3 to 0	T0_MAT[3:0]	-	-	-
Timer 0; match output channel 0	-	ADC start0 input (ADC CR register START bits = 0x2)		1
Timer 0; match output channel 2	-	Event router input 13	no	1
Timer 0; match output channel 3 to 0 ORed with SCT outputs 3 to 0.	CTOUT_[3:0]	-		0
Timer1 inputs				
Timer 1, input to capture channel 0	CTIN_0	-	yes	-
	T1_CAP0	-	no	-
Timer 1, input to capture channel 1	CTIN_3	-	yes	-
	T1_CAP1	-	no	-
	-	USART0 TX active	no	-
Timer 1, input to capture channel 2	CTIN_4	-	yes	-
	T1_CAP2	-	no	-
	-	USART0 RX active	no	-
Timer 1, input to capture channel 3	-	SCT output 3 OR T0 match channel 3	yes	0
	-	SCT output 3	yes	1
	-	T0 match channel 3	no	-
	T1_CAP3	-	no	-
Timer1 outputs				
Timer 1; match output channel 3 to 0	T1_MAT[3:0]	-	-	-
Timer 1; match output channel 2	-	Event router input 14	no	1
Timer 1; match output channel 3 to 0 ORed with SCT outputs 7 to 4.	CTOUT_[7:4]	-	-	0

Table 671. Timer0/1/2/3 inputs and outputs *...continued*

Description	Pin function	Internal signal	Default (see GIMA, Table 138)	CTOUTCTRL bit (see Table 43)
Timer 1; match output channel 2 ORed with SCT output 6	-	Event router input 14	no	0
Timer2 inputs				
Timer 2, input to capture channel 0	CTIN_0	-	yes	-
	T2_CAP0	-	no	-
Timer 2, input to capture channel 1	CTIN_1	-	yes	-
	T2_CAP1	-	no	-
	-	USART2 TX active	no	-
	-	I2S1_RX_MWS	no	-
Timer 2, input to capture channel 2	CTIN_5	-	yes	-
	T2_CAP2	-	no	-
	-	USART2 RX active	no	-
	-	I2S1_TX_MWS	no	-
Timer 2, input to capture channel 3	-	SCT output 7 OR T1 match channel 3	yes	0
Timer 2, input to capture channel 3	-	SCT output 7	yes	1
	-	T1 match channel 3	no	-
	T2_CAP3	-	no	-
Timer2 outputs				
Timer 2; match output channel 3 to 0	T2_MAT[3:0]	-	-	-
Timer 2; match output channel 0	-	ADC start1 input (ADC CR register bit START = 0x3)	no	1
Timer 2; match output channel 3 to 0 ORed with SCT outputs 11 to 8.	CTOUT_[11:8]	-	-	0
Timer 2; match output channel 0 ORed with SCT output 8	-	ADC start1 input (ADC CR register bit START = 0x3)	no	0
Timer3 inputs				
Timer 3, input to capture channel 0	CTIN_0		yes	-
	-	I2S0_RX_MWS	no	-
	T3_CAP0	-	no	-
Timer 3, input to capture channel 1	CTIN_6	-	yes	-
	T3_CAP1	-	no	-
	-	USART3 TX active	no	-
	-	I2S0_TX_MWS	no	-
Timer 3, input to capture channel 2	CTIN_7	-	yes	-
	T3_CAP2	-	no	-
	-	USART3 RX active	no	-
	-	SOF0	no	-

Table 671. Timer0/1/2/3 inputs and outputs *...continued*

Description	Pin function	Internal signal	Default (see GIMA, Table 138)	CTOUTCTRL bit (see Table 43)
Timer 3, input to capture channel 3	T3_CAP3		no	-
	-	SCT output 11 OR T2 match channel 3	yes	0
	-	SCT output 11	yes	1
	-	T2 match channel 3	no	-
	-	SOF1	no	-
Timer3 outputs				
Timer 3; match output channel 3 to 0	T3_MAT[3:0]	-	-	-
Timer 3; match output channel 2	-	Event router input 16	no	1
Timer 3; match output channel 3 to 0 ORed with SCT outputs 15 to 12.	CTOUT_[15:12]	-	-	0
Timer 3; match output channel 2 ORed with SCT output 14	-	Event router input 16	no	0

29.6 DMA connections

<tbd>

29.7 Register description

Each Timer/Counter contains the registers shown in Table 672.

Table 672. Register overview: Timer0/1/2/3 (register base addresses 0x4008 4000 (TIMER0), 0x4008 5000 (TIMER1), 0x400C 3000 (TIMER2), 0x400C 4000 (TIMER3))

Name	Access	Address offset	Description	Reset value[1]	Reference
IR	R/W	0x000	Interrupt Register. The IR can be written to clear interrupts. The IR can be read to identify which of eight possible interrupt sources are pending.	0	Table 673
TCR	R/W	0x004	Timer Control Register. The TCR is used to control the Timer Counter functions. The Timer Counter can be disabled or reset through the TCR.	0	Table 674
TC	R/W	0x008	Timer Counter. The 32 bit TC is incremented every PR+1 cycles of PCLK. The TC is controlled through the TCR.	0	Table 675
PR	R/W	0x00C	Prescale Register. When the Prescale Counter (PC) is equal to this value, the next clock increments the TC and clears the PC.	0	Table 676
PC	R/W	0x010	Prescale Counter. The 32 bit PC is a counter which is incremented to the value stored in PR. When the value in PR is reached, the TC is incremented and the PC is cleared. The PC is observable and controllable through the bus interface.	0	Table 677
MCR	R/W	0x014	Match Control Register. The MCR is used to control if an interrupt is generated and if the TC is reset when a Match occurs.	0	Table 678

Table 672. Register overview: Timer0/1/2/3 (register base addresses 0x4008 4000 (TIMER0), 0x4008 5000 (TIMER1), 0x400C 3000 (TIMER2), 0x400C 4000 (TIMER3))

Name	Access	Address offset	Description	Reset value[1]	Reference
MR0	R/W	0x018	Match Register 0. MR0 can be enabled through the MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt every time MR0 matches the TC.	0	Table 679
MR1	R/W	0x01C	Match Register 1. See MR0 description.	0	Table 679
MR2	R/W	0x020	Match Register 2. See MR0 description.	0	Table 679
MR3	R/W	0x024	Match Register 3. See MR0 description.	0	Table 679
CCR	R/W	0x028	Capture Control Register. The CCR controls which edges of the capture inputs are used to load the Capture Registers and whether or not an interrupt is generated when a capture takes place.	0	Table 680
CR0	RO	0x02C	Capture Register 0. CR0 is loaded with the value of TC when there is an event on the CAPn.0(CAP0.0 or CAP1.0 respectively) input.	0	Table 681
CR1	RO	0x030	Capture Register 1. See CR0 description.	0	Table 681
CR2	RO	0x034	Capture Register 2. See CR0 description.	0	Table 681
CR3	RO	0x038	Capture Register 3. See CR0 description.	0	Table 681
EMR	R/W	0x03C	External Match Register. The EMR controls the external match pins MATn.0-3 (MAT0.0-3 and MAT1.0-3 respectively).	0	Table 682
CTCR	R/W	0x070	Count Control Register. The CTCR selects between Timer and Counter mode, and in Counter mode selects the signal and edge(s) for counting.	0	Table 684

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

29.7.1 Timer interrupt registers

The Interrupt Register consists of four bits for the match interrupts and four bits for the capture interrupts. If an interrupt is generated then the corresponding bit in the IR will be high. Otherwise, the bit will be low. Writing a logic one to the corresponding IR bit will reset the interrupt. Writing a zero has no effect. The act of clearing an interrupt for a timer match also clears any corresponding DMA request.

Table 673. Timer interrupt registers IR(IR - addresses 0x4008 4000 (TIMER0), 0x4008 5000 (TIMER1), 0x400C 3000 (TIMER3), 0x400C 4000 (TIMER4)) bit description

Bit	Symbol	Description	Reset value
0	MR0INT	Interrupt flag for match channel 0.	0
1	MR1INT	Interrupt flag for match channel 1.	0
2	MR2INT	Interrupt flag for match channel 2.	0
3	MR3INT	Interrupt flag for match channel 3.	0
4	CR0INT	Interrupt flag for capture channel 0 event.	0
5	CR1INT	Interrupt flag for capture channel 1 event.	0
6	CR2INT	Interrupt flag for capture channel 2 event.	0
7	CR3INT	Interrupt flag for capture channel 3 event.	0
31:8	-	Reserved.	-

29.7.2 Timer control registers

The Timer Control Register (TCR) is used to control the operation of the Timer/Counter.

Table 674. Timer control register TCR (TCR - addresses 0x4008 4004 (TIMER0), 0x4008 5004 (TIMER1), 0x400C 3003 (TIMER2), 0x400C 4004 (TIMER3)) bit description

Bit	Symbol	Description	Reset value
0	CEN	When one, the Timer Counter and Prescale Counter are enabled for counting. When zero, the counters are disabled.	0
1	CRST	When one, the Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of PCLK. The counters remain reset until TCR[1] is returned to zero.	0
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

29.7.3 Timer counter registers

The 32-bit Timer Counter register is incremented when the prescale counter reaches its terminal count. Unless it is reset before reaching its upper limit, the Timer Counter will count up through the value 0xFFFF FFFF and then wrap back to the value 0x0000 0000. This event does not cause an interrupt, but a match register can be used to detect an overflow if needed.

Table 675. Timer counter registers TC (TC - addresses 0x4008 4008 (TIMER0), 0x4008 5008 (TIMER1), 0x400C 3008 (TIMER2), 0x400C 4008 (TIMER3)) bit description

Bit	Symbol	Description	Reset value
31:0	TC	Timer counter value.	0

29.7.4 Timer prescale registers

The 32-bit Timer prescale register specifies the maximum value for the Prescale Counter.

Table 676. Timer prescale registers PR (PR - addresses 0x4008 400C (TIMER0), 0x4008 500C (TIMER1), 0x400C 300C (TIMER2), 0x400C 400C (TIMER3)) bit description

Bit	Symbol	Description	Reset value
31:0	PM	Prescale counter maximum value.	0

29.7.5 Timer prescale counter registers

The 32-bit Prescale Counter controls division of PCLK by some constant value before it is applied to the Timer Counter. This allows control of the relationship of the resolution of the timer versus the maximum time before the timer overflows. The Prescale Counter is incremented on every PCLK. When it reaches the value stored in the Prescale register, the Timer Counter is incremented and the Prescale Counter is reset on the next PCLK. This causes the Timer Counter to increment on every PCLK when PR = 0, every 2 PCLKs when PR = 1, etc.

Table 677. Timer prescale counter registers PC(PC - addresses 0x4008 4010 (TIMER0), 0x4008 5010 (TIMER1), 0x400C 3010 (TIMER2), 0x400C 4010 (TIMER3)) bit description

Bit	Symbol	Description	Reset value
31:0	PC	Prescale counter value.	0

29.7.6 Timer match control registers

The Match Control Register is used to control what operations are performed when one of the Match Registers matches the Timer Counter. The function of each of the bits is shown in Table 678.

Table 678. Timer match control registers MCR (MCR - addresses 0x4008 4014 (TIMER0), 0x4008 5014 (TIMER1), 0x400C 3014 (TIMER2), 0x400C 4014 (TIMER3)) bit description

Bit	Symbol	Value	Description	Reset value
0	MR0I		Interrupt on MR0	0
		1	Interrupt is generated when MR0 matches the value in the TC.	
		0	Interrupt is disabled	
1	MR0R		Reset on MR0	0
		1	TC will be reset if MR0 matches it.	
		0	Feature disabled.	
2	MR0S	1	Stop on MR0	0
		1	TC and PC will be stopped and TCR[0] will be set to 0 if MR0 matches the TC.	
		0	Feature disabled.	
3	MR1I		Interrupt on MR1	0
		1	Interrupt is generated when MR1 matches the value in the TC.	
		0	Interrupt is disabled.	
4	MR1R		Reset on MR1	0
		1	TC will be reset if MR1 matches it.	
		0	Feature disabled.	
5	MR1S		Stop on MR1	0
		1	TC and PC will be stopped and TCR[0] will be set to 0 if MR1 matches the TC.	
		0	Feature disabled.	
6	MR2I		Interrupt on MR2	0
		1	Interrupt is generated when MR2 matches the value in the TC.	
		0	Interrupt is disabled	
7	MR2R		Reset on MR2	0
		1	TC will be reset if MR2 matches it.	
		0	Feature disabled.	

Table 678. Timer match control registers MCR (MCR - addresses 0x4008 4014 (TIMER0), 0x4008 5014 (TIMER1), 0x400C 3014 (TIMER2), 0x400C 4014 (TIMER3)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
8	MR2S		Stop on MR2.	0
		1	TC and PC will be stopped and TCR[0] will be set to 0 if MR2 matches the TC	
		0	Feature disabled.	
9	MR3I		Interrupt on MR3	0
		1	Interrupt is generated when MR3 matches the value in the TC.	
		0	This interrupt is disabled	
10	MR3R		Reset on MR3	0
		1	TC will be reset if MR3 matches it.	
		0	Feature disabled.	
11	MR3S		Stop on MR3	0
		1	TC and PC will be stopped and TCR[0] will be set to 0 if MR3 matches the TC.	
		0	Feature disabled.	
31:12	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

29.7.7 Timer match registers (MR0 - MR3)

The Match register values are continuously compared to the Timer Counter value. When the two values are equal, actions can be triggered automatically. The action possibilities are to generate an interrupt, reset the Timer Counter, or stop the timer. Actions are controlled by the settings in the MCR register.

Table 679. Timer match registers MR0 to 3 (MR, addresses 0x4008 4018 (MR0) to 0x4008 4024 (M3) (TIMER0), 0x4008 5018 (MR0) to 0x4008 5024 (MR3)(TIMER1), 0x400C 3018 (MR0) to 0x400C 8024 (MR3) (TIMER2), 0x400C 4018 (MR0) to 0x400C 4024 (MR3)(TIMER3)) bit description

Bit	Symbol	Description	Reset value
31:0	MATCH	Timer counter match value.	0

29.7.8 Timer capture control registers

The Capture Control Register is used to control whether one of the four Capture Registers is loaded with the value in the Timer Counter when the capture event occurs, and whether an interrupt is generated by the capture event. Setting both the rising and falling bits at the same time is a valid configuration, resulting in a capture event for both edges. In the description below, n represents the Timer number.

Remark: If Counter mode is selected for a particular CAP input in the CTCR, the 3 bits for that input in this register should be programmed as 000, but capture and/or interrupt can be selected for the other 3 CAP inputs.

Table 680. Timer capture control registers (CCR - addresses 0x4008 4028 (TIMER0), 0x4008 5020 (TIMER1), 0x400C 3028 (TIMER2), 0x400C 4028 (TIMER3)) bit description

Bit	Symbol	Value	Description	Reset value
0	CAP0RE		Capture on CAPn.0 rising edge	0
		1	A sequence of 0 then 1 on CAPn.0 will cause CR0 to be loaded with the contents of TC.	
		0	This feature is disabled.	
1	CAP0FE		Capture on CAPn.0 falling edge	0
		1	A sequence of 1 then 0 on CAPn.0 will cause CR0 to be loaded with the contents of TC.	
		0	This feature is disabled.	
2	CAP0I		Interrupt on CAPn.0 event	0
		1	A CR0 load due to a CAPn.0 event will generate an interrupt.	
		0	This feature is disabled.	
3	CAP1RE		Capture on CAPn.1 rising edge	0
		1	A sequence of 0 then 1 on CAPn.1 will cause CR1 to be loaded with the contents of TC.	
		0	This feature is disabled.	
4	CAP1FE		Capture on CAPn.1 falling edge	0
		1	A sequence of 1 then 0 on CAPn.1 will cause CR1 to be loaded with the contents of TC.	
		0	This feature is disabled.	
5	CAP1I		Interrupt on CAPn.1 event	0
		1	A CR1 load due to a CAPn.1 event will generate an interrupt.	
		0	This feature is disabled.	
6	CAP2RE		Capture on CAPn.2 rising edge	0
		1	A sequence of 0 then 1 on CAPn.2 will cause CR2 to be loaded with the contents of TC.	
		0	This feature is disabled.	
7	CAP2FE		Capture on CAPn.2 falling edge:	0
		1	A sequence of 1 then 0 on CAPn.2 will cause CR2 to be loaded with the contents of TC.	
		0	This feature is disabled.	
8	CAP2I		Interrupt on CAPn.2 event	0
		1	A CR2 load due to a CAPn.2 event will generate an interrupt.	
		0	This feature is disabled.	
9	CAP3RE		Capture on CAPn.3 rising edge	0
		1	A sequence of 0 then 1 on CAPn.3 will cause CR3 to be loaded with the contents of TC.	
		0	This feature is disabled.	
10	CAP3FE		Capture on CAPn.3 falling edge	0
		1	A sequence of 1 then 0 on CAPn.3 will cause CR3 to be loaded with the contents of TC.	
		0	This feature is disabled.	

Table 680. Timer capture control registers (CCR - addresses 0x4008 4028 (TIMER0), 0x4008 5020 (TIMER1), 0x400C 3028 (TIMER2), 0x400C 4028 (TIMER3)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
11	CAP3I		Interrupt on CAPn.3 event:	0
		1	A CR3 load due to a CAPn.3 event will generate an interrupt.	
		0	This feature is disabled.	
31:12	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

29.7.9 Timer capture registers (CR0 - CR3)

Each Capture register is associated with a device pin and may be loaded with the Timer Counter value when a specified event occurs on that pin. The settings in the Capture Control Register register determine whether the capture function is enabled, and whether a capture event happens on the rising edge of the associated pin, the falling edge, or on both edges.

Table 681. Timer capture registers CR0 to 3 (CR, address 0x4008 402C (CR0) to 0x4008 4038 (CR3) (TIMER0), 0x4008 502C (CR0) to 0x4008 5038 (CR3) (TIMER1), 0x400C 302C (CR0) to 0x400C 3038 (CR3) (TIMER2), 0x400C 402C (CR0) to 0x400C 4038 (CR3) (TIMER3)) bit description

Bit	Symbol	Description	Reset value
31:0	CAP	Timer counter capture value.	0

29.7.10 Timer external match registers

The External Match Register provides both control and status of the external match pins. In the descriptions below, “n” represents the Timer number, 0 or 1, and “m” represent a Match number, 0 through 3.

Match events for Match 0 and Match 1 in each timer can cause a DMA request, see Section 29.7.12.

Table 682. Timer external match registers (EMR - addresses 0x4008 403C (TIMER0), 0x4008 503C (TIMER1), 0x400C 303C (TIMER2), 0x400C 403C (TIMER3)) bit description

Bit	Symbol	Value	Description	Reset value
0	EM0		External Match 0. When a match occurs between the TC and MR0, this bit can either toggle, go low, go high, or do nothing, depending on bits 5:4 of this register. This bit can be driven onto a MATn.0 pin, in a positive-logic manner (0 = low, 1 = high).	0
1	EM1		External Match 1. When a match occurs between the TC and MR1, this bit can either toggle, go low, go high, or do nothing, depending on bits 7:6 of this register. This bit can be driven onto a MATn.1 pin, in a positive-logic manner (0 = low, 1 = high).	0
2	EM2		External Match 2. When a match occurs between the TC and MR2, this bit can either toggle, go low, go high, or do nothing, depending on bits 9:8 of this register. This bit can be driven onto a MATn.0 pin, in a positive-logic manner (0 = low, 1 = high).	0
3	EM3		External Match 3. When a match occurs between the TC and MR3, this bit can either toggle, go low, go high, or do nothing, depending on bits 11:10 of this register. This bit can be driven onto a MATn.0 pin, in a positive-logic manner (0 = low, 1 = high).	0
5:4	EMC0		External Match Control 0. Determines the functionality of External Match 0.	00
		0x0	Do Nothing.	
		0x1	Clear the corresponding External Match bit/output to 0 (MATn.m pin is LOW if pinned out).	
		0x2	Set the corresponding External Match bit/output to 1 (MATn.m pin is HIGH if pinned out).	
		0x3	Toggle the corresponding External Match bit/output.	
7:6	EMC1		External Match Control 1. Determines the functionality of External Match 1.	00
		0x0	Do Nothing.	
		0x1	Clear the corresponding External Match bit/output to 0 (MATn.m pin is LOW if pinned out).	
		0x2	Set the corresponding External Match bit/output to 1 (MATn.m pin is HIGH if pinned out).	
		0x3	Toggle the corresponding External Match bit/output.	
9:8	EMC2		External Match Control 2. Determines the functionality of External Match 2.	00
		0x0	Do Nothing.	
		0x1	Clear the corresponding External Match bit/output to 0 (MATn.m pin is LOW if pinned out).	
		0x2	Set the corresponding External Match bit/output to 1 (MATn.m pin is HIGH if pinned out).	
		0x3	Toggle the corresponding External Match bit/output.	

Table 682. Timer external match registers (EMR - addresses 0x4008 403C (TIMER0), 0x4008 503C (TIMER1), 0x400C 303C (TIMER2), 0x400C 403C (TIMER3)) bit description

Bit	Symbol	Value	Description	Reset value
11:10	EMC3		External Match Control 3. Determines the functionality of External Match 3.	00
		0x0	Do Nothing.	
		0x1	Clear the corresponding External Match bit/output to 0 (MATn.m pin is LOW if pinned out).	
		0x2	Set the corresponding External Match bit/output to 1 (MATn.m pin is HIGH if pinned out).	
		0x3	Toggle the corresponding External Match bit/output.	
15:12	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Table 683. External Match Control

EMR[11:10], EMR[9:8], EMR[7:6], or EMR[5:4]	Function
00	Do Nothing.
01	Clear the corresponding External Match bit/output to 0 (MATn.m pin is LOW if pinned out).
10	Set the corresponding External Match bit/output to 1 (MATn.m pin is HIGH if pinned out).
11	Toggle the corresponding External Match bit/output.

29.7.11 Timer count control registers

The Count Control Register (CTCR) is used to select between Timer and Counter mode, and in Counter mode to select the pin and edge(s) for counting.

When Counter Mode is chosen as a mode of operation, the CAP input (selected by the CTCR bits 3:2) is sampled on every rising edge of the PCLK clock. After comparing two consecutive samples of this CAP input, one of the following four events is recognized: rising edge, falling edge, either of edges or no changes in the level of the selected CAP input. Only if the identified event corresponds to the one selected by bits 1:0 in the CTCR register, the Timer Counter register will be incremented.

Effective processing of the externally supplied clock to the counter has some limitations. Since two successive rising edges of the PCLK clock are used to identify only one edge on the CAP selected input, the frequency of the CAP input can not exceed one quarter of the PCLK clock. Consequently, duration of the high/low levels on the same CAP input in this case can not be shorter than 1/(2 PCLK).

Table 684. Timer count control register CTCR(CTCR - addresses 0x4008 4070 (TIMER0), 0x4008 5070 (TIMER1), 0x400C 3070 (TIMER2), 0x400C 4070 (TIMER3)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	CTMODE		Counter/Timer Mode This field selects which rising PCLK edges can increment Timer's Prescale Counter (PC), or clear PC and increment Timer Counter (TC). Timer Mode: the TC is incremented when the Prescale Counter matches the Prescale Register.	00
		0x0	Timer Mode: every rising PCLK edge	
		0x1	Counter Mode: TC is incremented on rising edges on the CAP input selected by bits 3:2.	
		0x2	Counter Mode: TC is incremented on falling edges on the CAP input selected by bits 3:2.	
		0x3	Counter Mode: TC is incremented on both edges on the CAP input selected by bits 3:2.	
3:2	CINSEL		Count Input Select When bits 1:0 in this register are not 00, these bits select which CAP pin is sampled for clocking:	00
		0x0	CAPn.0 for TIMERn	
		0x1	CAPn.1 for TIMERn	
		0x2	CAPn.2 for TIMERn	
		0x3	CAPn.3 for TIMERn **Note:** If Counter mode is selected for a particular CAPn input in the TnCTCR, the 3 bits for that input in the Capture Control Register (TnCCR) must be programmed as 000. However, capture and/or interrupt can be selected for the other 3 CAPn inputs in the same timer.	
31:4	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

29.7.12 DMA operation

DMA requests are generated by 0 to 1 transitions of the External Match 0 and 1 bits of each timer. In order to have an effect, the GPDMA must be configured and the relevant timer DMA request selected as a DMA source via the CREG block, see Table 41.

When a timer is initially set up to generate a DMA request, the request may already be asserted before a match condition occurs. An initial DMA request may be avoided by having software by write a one to the interrupt flag location, as if clearing a timer interrupt. See Section 29.7.1. A DMA request will be cleared automatically when it is acted upon by the GPDMA controller.

29.8 Example timer operation

Figure 90 shows a timer configured to reset the count and generate an interrupt on match. The prescaler is set to 2 and the match register set to 6. At the end of the timer cycle where the match occurs, the timer count is reset. This gives a full length cycle to the match value. The interrupt indicating that a match occurred is generated in the next clock after the timer reached the match value.

Figure 91 shows a timer configured to stop and generate an interrupt on match. The prescaler is again set to 2 and the match register set to 6. In the next clock after the timer reaches the match value, the timer enable bit in TCR is cleared, and the interrupt indicating that a match occurred is generated.

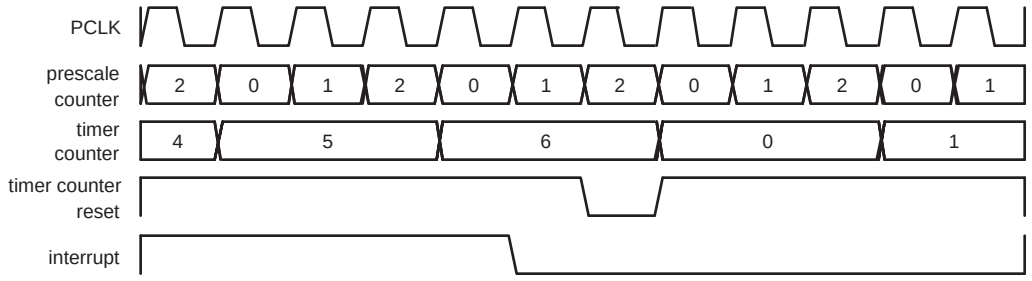


Fig 90. A timer cycle in which PR=2, MRx=6, and both interrupt and reset on match are enabled.

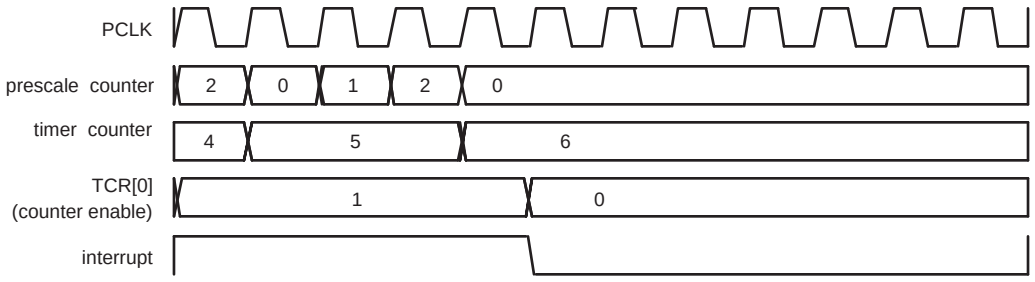


Fig 91. A timer Cycle in Which PR=2, MRx=6, and both interrupt and stop on match are enabled

29.9 Architecture

The block diagram for TIMER/COUNTER0 and TIMER/COUNTER1 is shown in Figure 92.

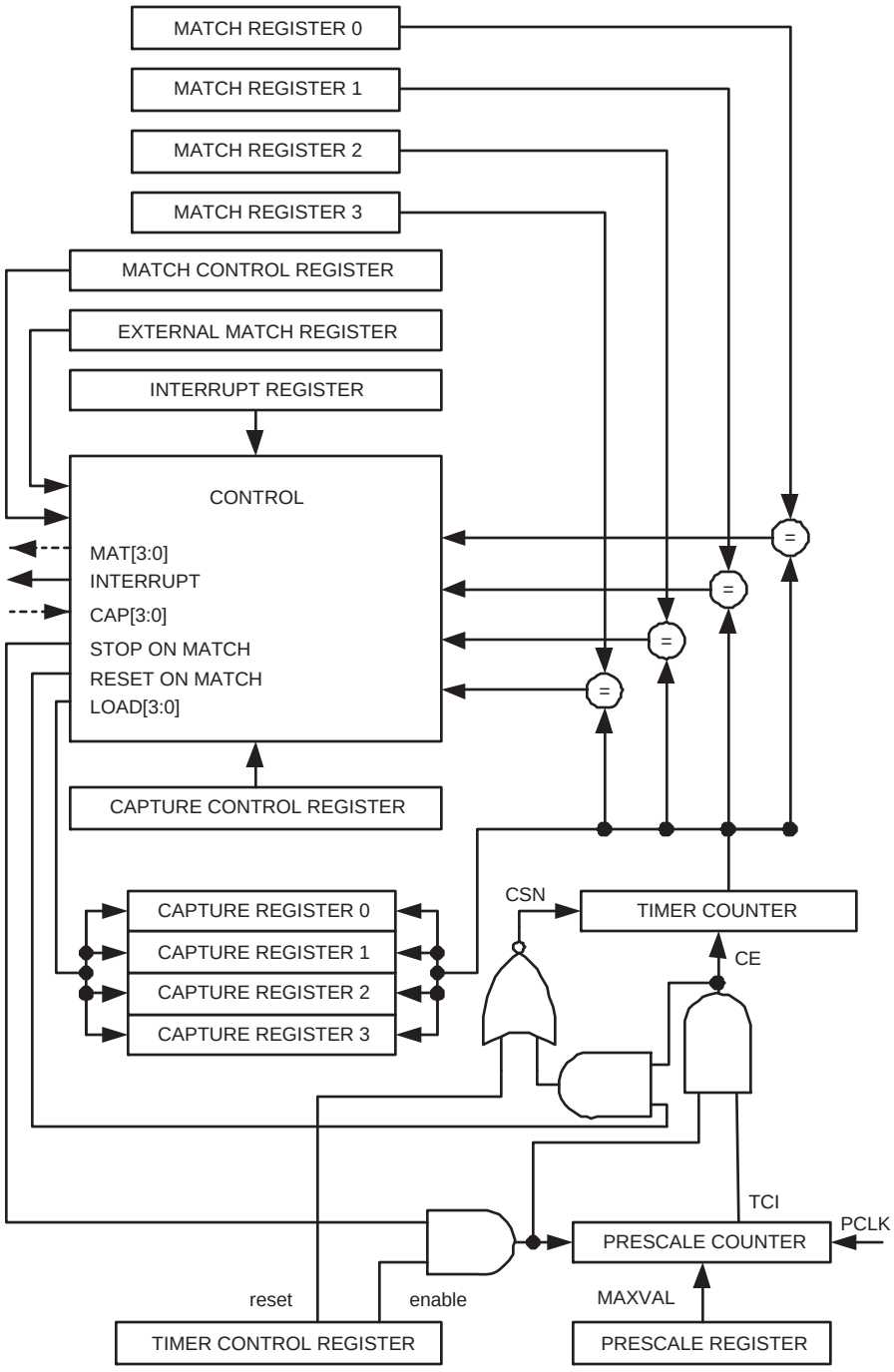


Fig 92. Timer block diagram

30.1 How to read this chapter

The Motor control PWM is available on all LPC43xx parts.

30.2 Basic configuration

The PWM is configured as follows:

- See Table 685 for clocking and power control.
- The PWM is reset by the MOTOCONPWM_RST (reset #38).
- The PWM interrupt is connected to slot #16 in the NVIC.

Table 685. PWM clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the PWM Motor control block and PWM Motocon peripheral clock.	BASE_APB1_CLK	CLK_APB1_MOTOCON	up to 204 MHz

30.3 Introduction

The Motor Control PWM (MCPWM) is optimized for three-phase AC and DC motor control applications, but can be used in many other applications that need timing, counting, capture, and comparison.

30.4 Features

The MCPWM contains three independent channels, each including:

- a 32-bit Timer/Counter (TC)
- a 32-bit Limit register (LIM)
- a 32-bit Match register (MAT)
- a 10-bit dead-time register (DT) and an associated 10-bit dead-time counter
- a 32-bit capture register (CAP)
- two modulated outputs (MCOA and MCOB) with opposite polarities
- a period interrupt, a pulse-width interrupt, and a capture interrupt

Input pins MCI0-2 can trigger TC capture or increment a channel's TC. A global Abort input can force all of the channels into "A passive" state and cause an interrupt.

30.5 General description

Section 30.8 includes detailed descriptions of the various modes of MCPWM operation, but a quick preview here will provide background for the register descriptions below.

The MCPWM includes 3 channels, each of which controls a pair of outputs that in turn can control something off-chip, like one set of coils in a motor. Each channel includes a Timer/Counter (TC) register that is incremented by a processor clock (timer mode) or by an input pin (counter mode).

Each channel has a Limit register that is compared to the TC value, and when a match occurs the TC is “recycled” in one of two ways. In “edge-aligned mode” the TC is reset to 0, while in “centered mode” a match switches the TC into a state in which it decrements on each processor clock or input pin transition until it reaches 0, at which time it starts counting up again.

Each channel also includes a Match register that holds a smaller value than the Limit register. In edge-aligned mode the channel’s outputs are switched whenever the TC matches either the Match or Limit register, while in center-aligned mode they are switched only when it matches the Match register.

So the Limit register controls the period of the outputs, while the Match register controls how much of each period the outputs spend in each state. Having a small value in the Limit register minimizes “ripple” if the output is integrated into a voltage, and allows the MCPWM to control devices that operate at high speed.

The “downside” of small values in the Limit register is that they reduce the resolution of the duty cycle controlled by the Match register. If you have 8 in the Limit register, the Match register can only select the duty cycle among 0%, 12.5%, 25%, …, 87.5%, or 100%. In general, the resolution of each step in the Match value is 1 divided by the Limit value.

This trade-off between resolution and period/frequency is inherent in the design of pulse width modulators.

30.5.1 Block Diagram

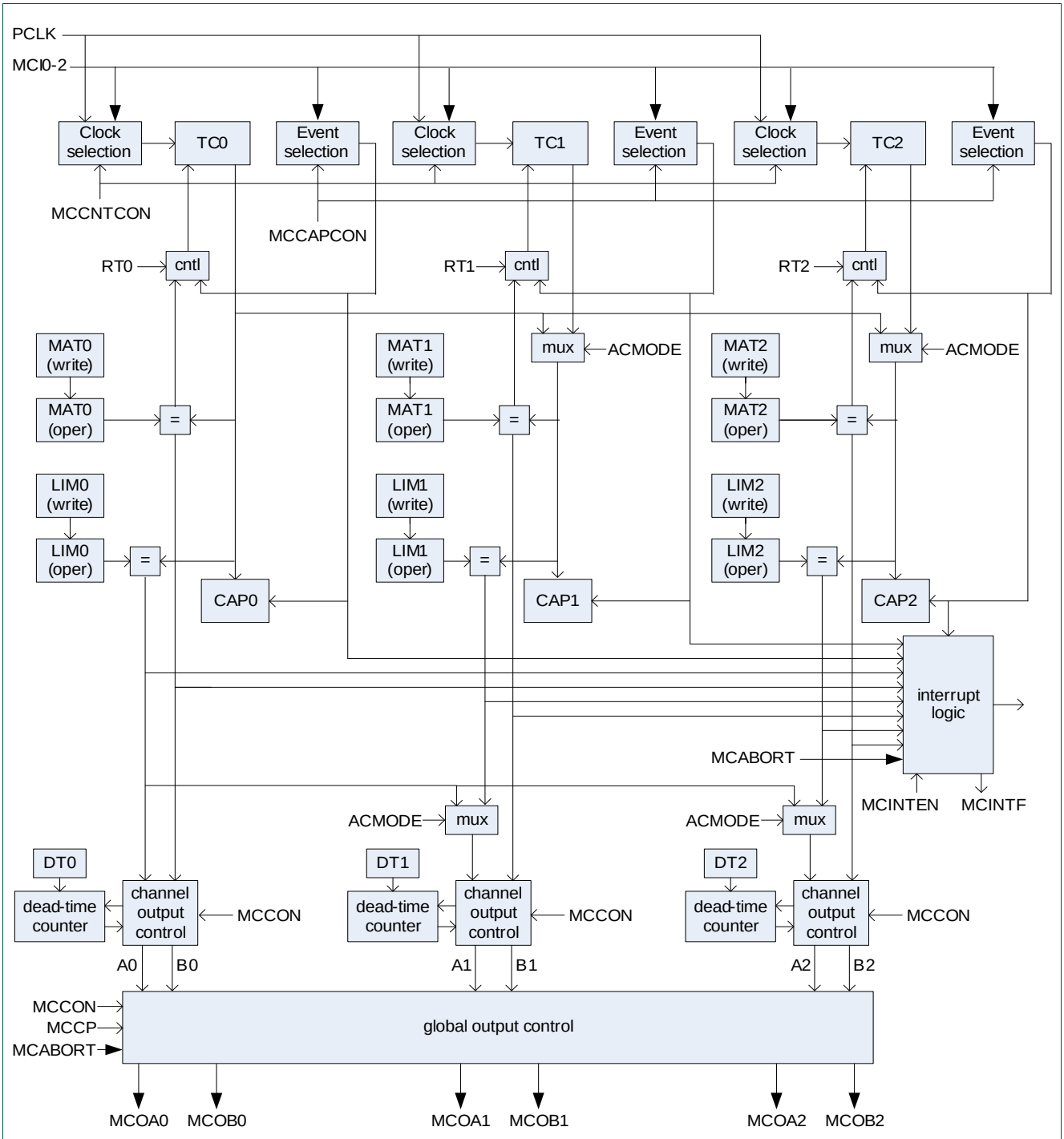


Fig 93. MCPWM Block Diagram

30.6 Pin description

Table 686 lists the MCPWM pins.

Table 686. MOTOCON PWM pin description

Function name	Type	Description
MCOA0/1/2	O	Output A for channels 0, 1, 2
MCOB0/1/2	O	Output B for channels 0, 1, 2
$\overline{\text{MCABORT}}$	I	Low-active Fast Abort
MCI0/1/2	I	Input for channels 0, 1, 2

30.7 Register description

"Control" registers and "interrupt" registers have separate read, set, and clear addresses. Reading such a register's read address (e.g. MCCON) yields the state of the register bits. Writing ones to the set address (e.g. MCCON_SET) sets register bit(s), and writing ones to the clear address (e.g. MCCON_CLR) clears register bit(s).

The Capture registers (MCCAP) are read-only, and the write-only MCCAP_CLR address can be used to clear one or more of them. All the other MCPWM registers (MCTIM, MCPER, MCPW, MCDEADTIME, and MCCP) are normal read-write registers.

Table 687. Register overview: Motor Control Pulse Width Modulator (MCPWM) (base address 0x400A 0000)

Name	Access	Address offset	Description	Reset value	Reference
CON	RO	0x000	PWM Control read address	0	Table 688
CON_SET	WO	0x004	PWM Control set address	-	Table 689
CON_CLR	WO	0x008	PWM Control clear address	-	Table 690
CAPCON	RO	0x00C	Capture Control read address	0	Table 691
CAPCON_SET	WO	0x010	Capture Control set address	-	Table 692
CAPCON_CLR	WO	0x014	Event Control clear address	-	Table 693
TC0	R/W	0x018	Timer Counter register, channel 0	0	Table 694
TC1	R/W	0x01C	Timer Counter register, channel 1	0	Table 694
TC2	R/W	0x020	Timer Counter register, channel 2	0	Table 694
LIM0	R/W	0x024	Limit register, channel 0	0xFFFF FFFF	Table 695
LIM1	R/W	0x028	Limit register, channel 1	0xFFFF FFFF	Table 695
LIM2	R/W	0x02C	Limit register, channel 2	0xFFFF FFFF	Table 695
MAT0	R/W	0x030	Match register, channel 0	0xFFFF FFFF	Table 696
MAT1	R/W	0x034	Match register, channel 1	0xFFFF FFFF	Table 696
MAT2	R/W	0x038	Match register, channel 2	0xFFFF FFFF	Table 696
DT	R/W	0x03C	Dead time register	0x3FFF FFFF	Table 697
MCCP	R/W	0x040	Communication Pattern register	0	Table 698
CAP0	RO	0x044	Capture register, channel 0	0	Table 699
CAP1	RO	0x048	Capture register, channel 1	0	Table 699
CAP2	RO	0x04C	Capture register, channel 2	0	Table 699
INTEN	RO	0x050	Interrupt Enable read address	0	Table 701
INTEN_SET	WO	0x054	Interrupt Enable set address	-	Table 702
INTEN_CLR	WO	0x058	Interrupt Enable clear address	-	Table 703
CNTCON	RO	0x05C	Count Control read address	0	Table 704

Table 687. Register overview: Motor Control Pulse Width Modulator (MCPWM) (base address 0x400A 0000)

Name	Access	Address offset	Description	Reset value	Reference
CNTCON_SET	WO	0x060	Count Control set address	-	Table 705
CNTCON_CLR	WO	0x064	Count Control clear address	-	Table 706
INTF	RO	0x068	Interrupt flags read address	0	Table 707
INTF_SET	WO	0x06C	Interrupt flags set address	-	Table 708
INTF_CLR	WO	0x070	Interrupt flags clear address	-	Table 709
CAP_CLR	WO	0x074	Capture clear address	-	Table 710

30.7.1 MCPWM Control register

30.7.1.1 MCPWM Control read address

The CON register controls the operation of all channels of the PWM. This address is read-only, but the underlying register can be modified by writing to addresses CON_SET and CON_CLR.

Table 688. MCPWM Control read address (CON - 0x400A 0000) bit description

Bit	Symbol	Value	Description	Reset value
0	RUN0		Stops/starts timer channel 0.	0
		0	Stop.	
		1	Run.	
1	CENTER0		Edge/center aligned operation for channel 0.	0
		0	Edge-aligned.	
		1	Center-aligned.	
2	POLA0		Selects polarity of the MCOA0 and MCOB0 pins.	0
		0	Passive state is LOW, active state is HIGH.	
		1	Passive state is HIGH, active state is LOW.	
3	DTE0		Controls the dead-time feature for channel 0.	0
		0	Dead-time disabled.	
		1	Dead-time enabled.	
4	DISUP0		Enable/disable updates of functional registers for channel 0 (see Section 30.8.2).	0
		0	Functional registers are updated from the write registers at the end of each PWM cycle.	
		1	Functional registers remain the same as long as the timer is running.	
7:5	-	-	Reserved.	
8	RUN1		Stops/starts timer channel 1.	0
		0	Stop.	
		1	Run.	
9	CENTER1		Edge/center aligned operation for channel 1.	0
		0	Edge-aligned.	
		1	Center-aligned.	

Table 688. MCPWM Control read address (CON - 0x400A 0000) bit description

Bit	Symbol	Value	Description	Reset value
10	POLA1		Selects polarity of the MCOA1 and MCOB1 pins.	0
		0	Passive state is LOW, active state is HIGH.	
		1	Passive state is HIGH, active state is LOW.	
11	DTE1		Controls the dead-time feature for channel 1.	0
		0	Dead-time disabled.	
		1	Dead-time enabled.	
12	DISUP1		Enable/disable updates of functional registers for channel 1 (see Section 30.8.2).	0
		0	Functional registers are updated from the write registers at the end of each PWM cycle.	
		1	Functional registers remain the same as long as the timer is running.	
15:13	-	-	Reserved.	0
16	RUN2		Stops/starts timer channel 2.	0
		0	Stop.	
		1	Run.	
17	CENTER2		Edge/center aligned operation for channel 2.	0
		0	Edge-aligned.	
		1	Center-aligned.	
18	POLA2		Selects polarity of the MCOA2 and MCOB2 pins.	0
		0	Passive state is LOW, active state is HIGH.	
		1	Passive state is HIGH, active state is LOW.	
19	DTE2		Controls the dead-time feature for channel 1.	0
		0	Dead-time disabled.	
		1	Dead-time enabled.	
20	DISUP2		Enable/disable updates of functional registers for channel 2 (see Section 30.8.2).	0
		0	Functional registers are updated from the write registers at the end of each PWM cycle.	
		1	Functional registers remain the same as long as the timer is running.	
28:21	-	-	Reserved.	
29	INVBDC		Controls the polarity of the MCOB outputs for all 3 channels. This bit is typically set to 1 only in 3-phase DC mode.	
		0	The MCOB outputs have opposite polarity from the MCOA outputs (aside from dead time).	
		1	The MCOB outputs have the same basic polarity as the MCOA outputs. (see Section 30.8.6)	
30	ACMODE		3-phase AC mode select (see Section 30.8.7).	0
		0	3-phase AC-mode off: Each PWM channel uses its own timer-counter and period register.	
		1	3-phase AC-mode on: All PWM channels use the timer-counter and period register of channel 0.	

Table 688. MCPWM Control read address (CON - 0x400A 0000) bit description

Bit	Symbol	Value	Description	Reset value
31	DCMODE		3-phase DC mode select (see Section 30.8.6).	0
		0	3-phase DC mode off: PWM channels are independent (unless bit ACMODE = 1)	
		1	3-phase DC mode on: The internal MCOA0 output is routed through the CP register (i.e. a mask) register to all six PWM outputs.	

30.7.1.2 MCPWM Control set address

Writing ones to this write-only address sets the corresponding bits in MCCON.

Table 689. MCPWM Control set address (CON_SET - 0x400A 0004) bit description

Bit	Symbol	Description	Reset value
0	RUN0_SET	Writing a one sets the corresponding bit in the CON register.	-
1	CENTER0_SET	Writing a one sets the corresponding bit in the CON register.	-
2	POLA0_SET	Writing a one sets the corresponding bit in the CON register.	-
3	DTE0_SET	Writing a one sets the corresponding bit in the CON register.	-
4	DISUP0_SET	Writing a one sets the corresponding bit in the CON register.	-
7:5	-	Writing a one sets the corresponding bit in the CON register.	-
8	RUN1_SET	Writing a one sets the corresponding bit in the CON register.	-
9	CENTER1_SET	Writing a one sets the corresponding bit in the CON register.	-
10	POLA1_SET	Writing a one sets the corresponding bit in the CON register.	-
11	DTE1_SET	Writing a one sets the corresponding bit in the CON register.	-
12	DISUP1_SET	Writing a one sets the corresponding bit in the CON register.	-
15:13	-	Writing a one sets the corresponding bit in the CON register.	-
16	RUN2_SET	Writing a one sets the corresponding bit in the CON register.	-
17	CENTER2_SET	Writing a one sets the corresponding bit in the CON register.	-
18	POLA2_SET	Writing a one sets the corresponding bit in the CON register.	-
19	DTE2_SET	Writing a one sets the corresponding bit in the CON register.	-
20	DISUP2_SET	Writing a one sets the corresponding bit in the CON register.	-
28:21	-	Writing a one sets the corresponding bit in the CON register.	-
29	INVBDC_SET	Writing a one sets the corresponding bit in the CON register.	-
30	ACMODE_SET	Writing a one sets the corresponding bit in the CON register.	-
31	DCMODE_SET	Writing a one sets the corresponding bit in the CON register.	-

30.7.1.3 MCPWM Control clear address

Writing ones to this write-only address clears the corresponding bits in CON.

Table 690. MCPWM Control clear address (CON_CLR - 0x400A 0008) bit description

Bit	Symbol	Description	Reset value
0	RUN0_CLR	Writing a one clears the corresponding bit in the CON register.	-
1	CENTER0_CLR	Writing a one clears the corresponding bit in the CON register.	-
2	POLA0_CLR	Writing a one clears the corresponding bit in the CON register.	-
3	DTE0_CLR	Writing a one clears the corresponding bit in the CON register.	-

Table 690. MCPWM Control clear address (CON_CLR - 0x400A 0008) bit description

Bit	Symbol	Description	Reset value
4	DISUP0_CLR	Writing a one clears the corresponding bit in the CON register.	-
7:5	-	Writing a one clears the corresponding bit in the CON register.	-
8	RUN1_CLR	Writing a one clears the corresponding bit in the CON register.	-
9	CENTER1_CLR	Writing a one clears the corresponding bit in the CON register.	-
10	POLA1_CLR	Writing a one clears the corresponding bit in the CON register.	-
11	DTE1_CLR	Writing a one clears the corresponding bit in the CON register.	-
12	DISUP1_CLR	Writing a one clears the corresponding bit in the CON register.	-
15:13	-	Writing a one clears the corresponding bit in the CON register.	-
16	RUN2_CLR	Writing a one clears the corresponding bit in the CON register.	-
17	CENTER2_CLR	Writing a one clears the corresponding bit in the CON register.	-
18	POLA2_CLR	Writing a one clears the corresponding bit in the CON register.	-
19	DTE2_CLR	Writing a one clears the corresponding bit in the CON register.	-
20	DISUP2_CLR	Writing a one clears the corresponding bit in the CON register.	-
28:21	-	Writing a one clears the corresponding bit in the CON register.	-
29	INVBDC_CLR	Writing a one clears the corresponding bit in the CON register.	-
30	ACMOD_CLR	Writing a one clears the corresponding bit in the CON register.	-
31	DCMODE_CLR	Writing a one clears the corresponding bit in the CON register.	

30.7.2 PWM Capture Control register

30.7.2.1 MCPWM Capture Control read address

The MCCAPCON register controls detection of events on the MCI0-2 inputs for all MCPWM channels. Any of the three MCI inputs can be used to trigger a capture event on any or all of the three channels. This address is read-only, but the underlying register can be modified by writing to addresses CAPCON_SET and CAPCON_CLR.

Table 691. MCPWM Capture Control read address (CAPCON - 0x400A 000C) bit description

Bit	Symbol	Description	Reset value
0	CAP0MCI0_RE	A 1 in this bit enables a channel 0 capture event on a rising edge on MCI0.	0
1	CAP0MCI0_FE	A 1 in this bit enables a channel 0 capture event on a falling edge on MCI0.	0
2	CAP0MCI1_RE	A 1 in this bit enables a channel 0 capture event on a rising edge on MCI1.	0
3	CAP0MCI1_FE	A 1 in this bit enables a channel 0 capture event on a falling edge on MCI1.	0
4	CAP0MCI2_RE	A 1 in this bit enables a channel 0 capture event on a rising edge on MCI2.	0
5	CAP0MCI2_FE	A 1 in this bit enables a channel 0 capture event on a falling edge on MCI2.	0
6	CAP1MCI0_RE	A 1 in this bit enables a channel 1 capture event on a rising edge on MCI0.	0
7	CAP1MCI0_FE	A 1 in this bit enables a channel 1 capture event on a falling edge on MCI0.	0
8	CAP1MCI1_RE	A 1 in this bit enables a channel 1 capture event on a rising edge on MCI1.	0
9	CAP1MCI1_FE	A 1 in this bit enables a channel 1 capture event on a falling edge on MCI1.	0
10	CAP1MCI2_RE	A 1 in this bit enables a channel 1 capture event on a rising edge on MCI2.	0

Table 691. MCPWM Capture Control read address (CAPCON - 0x400A 000C) bit description

Bit	Symbol	Description	Reset value
11	CAP1MCI2_FE	A 1 in this bit enables a channel 1 capture event on a falling edge on MCI2.	0
12	CAP2MCI0_RE	A 1 in this bit enables a channel 2 capture event on a rising edge on MCI0.	0
13	CAP2MCI0_FE	A 1 in this bit enables a channel 2 capture event on a falling edge on MCI0.	0
14	CAP2MCI1_RE	A 1 in this bit enables a channel 2 capture event on a rising edge on MCI1.	0
15	CAP2MCI1_FE	A 1 in this bit enables a channel 2 capture event on a falling edge on MCI1.	0
16	CAP2MCI2_RE	A 1 in this bit enables a channel 2 capture event on a rising edge on MCI2.	0
17	CAP2MCI2_FE	A 1 in this bit enables a channel 2 capture event on a falling edge on MCI2.	0
18	RT0	If this bit is 1, TC0 is reset by a channel 0 capture event.	0
19	RT1	If this bit is 1, TC1 is reset by a channel 1 capture event.	0
20	RT2	If this bit is 1, TC2 is reset by a channel 2 capture event.	0
21	HNFCAP0	Hardware noise filter: if this bit is 1, channel 0 capture events are delayed as described in Section 30.8.4.	0
22	HNFCAP1	Hardware noise filter: if this bit is 1, channel 1 capture events are delayed as described in Section 30.8.4.	0
23	HNFCAP2	Hardware noise filter: if this bit is 1, channel 2 capture events are delayed as described in Section 30.8.4.	0
31:24	-	Reserved.	-

30.7.2.2 MCPWM Capture Control set address

Writing ones to this write-only address sets the corresponding bits in CAPCON.

Table 692. MCPWM Capture Control set address (CAPCON_SET - 0x400A 0010) bit description

Bit	Symbol	Description	Reset value
0	CAP0MCI0_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
1	CAP0MCI0_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
2	CAP0MCI1_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
3	CAP0MCI1_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
4	CAP0MCI2_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
5	CAP0MCI2_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
6	CAP1MCI0_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
7	CAP1MCI0_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
8	CAP1MCI1_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
9	CAP1MCI1_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-

Table 692. MCPWM Capture Control set address (CAPCON_SET - 0x400A 0010) bit description

Bit	Symbol	Description	Reset value
10	CAP1MCI2_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
11	CAP1MCI2_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
12	CAP2MCI0_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
13	CAP2MCI0_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
14	CAP2MCI1_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
15	CAP2MCI1_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
16	CAP2MCI2_RE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
17	CAP2MCI2_FE_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
18	RT0_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
19	RT1_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
20	RT2_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
21	HNFCAP0_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
22	HNFCAP1_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
23	HNFCAP2_SET	Writing a one sets the corresponding bits in the CAPCON register.	-
31:24	-	Reserved.	-

30.7.2.3 MCPWM Capture control clear address

Writing ones to this write-only address clears the corresponding bits in MCCAPCON.

Table 693. MCPWM Capture control clear register (CAPCON_CLR - address 0x400A 0014) bit description

Bit	Symbol	Description	Reset value
0	CAP0MCI0_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
1	CAP0MCI0_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
2	CAP0MCI1_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
3	CAP0MCI1_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-

Table 693. MCPWM Capture control clear register (CAPCON_CLR - address 0x400A 0014) bit description

Bit	Symbol	Description	Reset value
4	CAP0MCI2_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
5	CAP0MCI2_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
6	CAP1MCI0_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
7	CAP1MCI0_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
8	CAP1MCI1_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
9	CAP1MCI1_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
10	CAP1MCI2_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
11	CAP1MCI2_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
12	CAP2MCI0_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
13	CAP2MCI0_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
14	CAP2MCI1_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
15	CAP2MCI1_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
16	CAP2MCI2_RE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
17	CAP2MCI2_FE_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
18	RT0_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
19	RT1_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
20	RT2_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
21	HNFCAP0_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
22	HNFCAP1_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
23	HNFCAP2_CLR	Writing a one clears the corresponding bits in the CAPCON register.	-
31:24	-	Reserved.	-

30.7.3 MCPWM Timer/Counter 0-2 registers

These registers hold the current values of the 32-bit counter/timers for channels 0-2. Each value is incremented on every PCLK, or by edges on the MCI0-2 pins, as selected by CNTCON. The timer/counter counts up from 0 until it reaches the value in its corresponding PER register (or is stopped by writing to CON_CLR).

A TC register can be read at any time. In order to write to the TC register, its channel must be stopped. If not, the write will not take place, no exception is generated.

Table 694. MCPWM Timer/Counter 0 to 2 registers (TC - 0x400A 0018 (TC0), 0x400A 001C (TC1), 0x400A 0020) (TC2)bit description

Bit	Symbol	Description	Reset value
31:0	MCTC	Timer/Counter value.	0

30.7.4 MCPWM Limit 0-2 registers

These registers hold the limiting values for timer/counters 0-2. When a timer/counter reaches its corresponding limiting value: 1) in edge-aligned mode, it is reset and starts over at 0; 2) in center-aligned mode, it begins counting down until it reaches 0, at which time it begins counting up again.

If the channel's CENTER bit in CON is 0 selecting edge-aligned mode, the match between TC and LIM switches the channel's A output from "active" to "passive" state. If the channel's CENTER and DTE bits in CON are both 0, the match simultaneously switches the channel's B output from "passive" to "active" state.

If the channel's CENTER bit is 0 but the DTE bit is 1, the match triggers the channel's deadtime counter to begin counting -- when the deadtime counter expires, the channel's B output switches from "passive" to "active" state.

In center-aligned mode, matches between a channel's TC and LIM registers have no effect on its A and B outputs.

Writing to either a Limit or a Match (30.7.5) register loads a "write" register, and if the channel is stopped it also loads an "operating" register that is compared to the TC. If the channel is running and its "disable update" bit in CON is 0, the operating registers are loaded from the write registers: 1) in edge-aligned mode, when the TC matches the operating Limit register; 2) in center-aligned mode, when the TC counts back down to 0. If the channel is running and the "disable update" bit is 1, the operating registers are not loaded from the write registers until software stops the channel.

Reading an LIM address always returns the operating value.

Table 695. MCPWM Limit 0 to 2 registers (LIM - 0x400A 0024 (LIM0), 0x400A 0028 (LIM1), 0x400A 002C (LIM2)) bit description

Bit	Symbol	Description	Reset value
31:0	MCLIM	Limit value.	0xFFFF FFFF

Remark: In timer mode, the period of a channel's modulated MCO outputs is determined by its Limit register, and the pulse width at the start of the period is determined by its Match register. If it suits your way of thinking, consider the Limit register to be the "Period register" and the Match register to be the "Pulse Width register".

30.7.5 MCPWM Match 0-2 registers

These registers also have “write” and “operating” versions as described above for the Limit registers, and the operating registers are also compared to the channels’ TCs. See 30.7.4 above for details of reading and writing both Limit and Match registers.

The Match and Limit registers control the MCO0-2 outputs. If a Match register is to have any effect on its channel’s operation, it must contain a smaller value than the corresponding Limit register.

Table 696. MCPWM Match 0 to 2 registers (MAT - addresses 0x400A 0030 (MAT0), 0x400A 0034 (MAT1), 0x400A 0038 (MAT2)) bit description

Bit	Symbol	Description	Reset value
31:0	MCMAT	Match value.	0xFFFF FFFF

30.7.5.1 Match register in Edge-Aligned mode

If the channel’s CENTER bit in CON is 0 selecting edge-aligned mode, a match between TC and MAT switches the channel’s B output from “active” to “passive” state. If the channel’s CENTER and DTE bits in CON are both 0, the match simultaneously switches the channel’s A output from “passive” to “active” state.

If the channel’s CENTER bit is 0 but the DTE bit is 1, the match triggers the channel’s deadtime counter to begin counting -- when the deadtime counter expires, the channel’s A output switches from “passive” to “active” state.

30.7.5.2 Match register in Center-Aligned mode

If the channel’s CENTER bit in CON is 1 selecting center-aligned mode, a match between TC and MAT while the TC is incrementing switches the channel’s B output from “active” to “passive” state, and a match while the TC is decrementing switches the A output from “active” to “passive”. If the channel’s CENTER bit in CON is 1 but the DTE bit is 0, a match simultaneously switches the channel’s other output in the opposite direction.

If the channel’s CENTER and DTE bits are both 1, a match between TC and MAT triggers the channel’s deadtime counter to begin counting -- when the deadtime counter expires, the channel’s B output switches from “passive” to “active” if the TC was counting up at the time of the match, and the channel’s A output switches from “passive” to “active” if the TC was counting down at the time of the match.

30.7.5.3 0 and 100% duty cycle

To lock a channel’s MCO outputs at the state “B active, A passive”, write its Match register with a higher value than you write to its Limit register. The match never occurs.

To lock a channel’s MCO outputs at the opposite state, “A active, B passive”, simply write 0 to its Match register.

30.7.6 MCPWM Dead-time register

This register holds the dead-time values for the three channels. If a channel’s DTE bit in CON is 1 to enable its dead-time counter, the counter counts down from this value whenever one its channel’s outputs changes from “active” to “passive” state. When the dead-time counter reaches 0, the channel changes its other output from “passive” to “active” state.

The motivation for the dead-time feature is that power transistors, like those driven by the A and B outputs in a motor-control application, take longer to fully turn off than they take to start to turn on. If the A and B transistors are ever turned on at the same time, a wasteful and damaging current will flow between the power rails through the transistors. In such applications, the dead-time register should be programmed with the number of PCLK periods that is greater than or equal to the transistors' maximum turn-off time minus their minimum turn-on time.

Table 697. MCPWM Dead-time register (DT - address 0x400A 003C) bit description

Bit	Symbol	Description	Reset value
9:0	DT0	Dead time for channel 0.[1]	0x3FF
19:10	DT1	Dead time for channel 1.[2]	0x3FF
29:20	DT2	Dead time for channel 2.[2]	0x3FF
31:30	-	reserved	

[1] If ACMODE is 1 selecting AC-mode, this field controls the dead time for all three channels.

[2] If ACMODE is 0.

30.7.7 MCPWM Communication Pattern register

This register is used in DC mode only. The internal MCOA0 signal is routed to any or all of the six output pins under the control of the bits in this register. Like the Match and Limit registers, this register has "write" and "operational" versions. See 30.7.4 and 30.8.2 for more about this subject.

Table 698. MCPWM Communication Pattern register (CP - address 0x400A 0040) bit description

Bit	Symbol	Value	Description	Reset value
0	CCPA0		Communication pattern output A, channel 0.	0
		0	MCOA0 passive.	
		1	internal MCOA0.	
1	CCPB0		Communication pattern output B, channel 0.	0
		0	MCOB0 passive.	
		1	MCOB0 tracks internal MCOA0.	
2	CCPA1		Communication pattern output A, channel 1.	0
		0	MCOA1 passive.	
		1	MCOA1 tracks internal MCOA0.	
3	CCPB1		Communication pattern output B, channel 1.	0
		0	MCOB1 passive.	
		1	MCOB1 tracks internal MCOA0.	
4	CCPA2		Communication pattern output A, channel 2.	0
		0	MCOA2 passive.	
		1	MCOA2 tracks internal MCOA0.	
5	CCPB2		Communication pattern output B, channel 2.	0
		0	MCOB2 passive.	
		1	MCOB2 tracks internal MCOA0.	
31:6	-		Reserved.	

30.7.8 MCPWM Capture read addresses

The CAPCON register (Table 691) allows software to select any edge(s) on any of the MCI0-2 inputs as a capture event for each channel. When a channel's capture event occurs, the current TC value for that channel is stored in its read-only Capture register. These addresses are read-only, but the underlying registers can be cleared by writing to the CAP_CLR address

Table 699. MCPWM Capture read addresses (CAP - 0x400A 0044 (CAP0), 0x400A 0048 (CAP1), 0x400A 004C 9CAP2)) bit description

Bit	Symbol	Description	Reset value
31:0	CAP	Current TC value at a capture event.	0x0000 0000

30.7.9 MCPWM Interrupt registers

The Motor Control PWM module includes the following interrupt sources:

Table 700. Motor Control PWM interrupts

Symbol	Description
ILIM0/1/2	Limit interrupts for channels 0, 1, 2.
IMAT0/1/2	Match interrupts for channels 0, 1, 2.
ICAP0/1/2	Capture interrupts for channels 0, 1, 2.
ABORT	Fast abort interrupt

7.9.1 MCPWM Interrupt Enable read address

The INTEN register controls which of the MCPWM interrupts are enabled. This address is read-only, but the underlying register can be modified by writing to addresses INTEN_SET and INTEN_CLR.

Table 701. MCPWM Interrupt Enable read address (INTEN - 0x400A 0050) bit description

Bit	Symbol	Value	Description	Reset value
0	ILIM0		Limit interrupt for channel 0.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
1	IMAT0		Match interrupt for channel 0.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
2	ICAP0		Capture interrupt for channel 0.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
3	-		Reserved.	-
4	ILIM1		Limit interrupt for channel 1.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	

Table 701. MCPWM Interrupt Enable read address (INTEN - 0x400A 0050) bit description

Bit	Symbol	Value	Description	Reset value
5	IMAT1		Match interrupt for channel 1.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
6	ICAP1		Capture interrupt for channel 1.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
7	-		Reserved.	-
8	ILIM2		Limit interrupt for channel 2.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
9	IMAT2		Match interrupt for channel 2.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
10	ICAP2		Capture interrupt for channel 2.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
14:11	-		Reserved.	-
15	ABORT		Fast abort interrupt.	0
		0	Interrupt disabled.	
		1	Interrupt enabled.	
31:16	-		Reserved.	-

30.7.9.2 MCPWM Interrupt Enable set address

Writing ones to this write-only address sets the corresponding bits in INTEN, thus enabling interrupts.

Table 702. MCPWM interrupt enable set register (INTEN_SET - address 0x400A 0054) bit description

Bit	Symbol	Description	Reset value
0	ILIM0_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
1	IMAT0_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
2	ICAP0_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
3	-	Reserved.	-
4	ILIM1_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
5	IMAT1_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
6	ICAP1_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-

Table 702. MCPWM interrupt enable set register (INTEN_SET - address 0x400A 0054) bit description

Bit	Symbol	Description	Reset value
7	-	Reserved.	-
9	ILIM2_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
10	IMAT2_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
11	ICAP2_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
14:12	-	Reserved.	-
15	ABORT_SET	Writing a one sets the corresponding bit in INTEN, thus enabling the interrupt.	-
31:16	-	Reserved.	-

30.7.9.3 MCPWM Interrupt Enable clear address

Writing ones to this write-only address clears the corresponding bits in INTEN, thus disabling interrupts.

Table 703. PWM interrupt enable clear register (INTEN_CLR - address 0x400A 0058) bit description

Bit	Symbol	Description	Reset value
0	ILIM0_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
1	IMAT0_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
2	ICAP0_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
3	-	Reserved.	-
4	ILIM1_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
5	IMAT1_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
6	ICAP1_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
7	-	Reserved.	-
8	ILIM2_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
9	IMAT2_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
10	ICAP2_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
14:11	-	Reserved.	-
15	ABORT_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
31:16	-	Reserved.	-

30.7.10 MCPWM Count Control register

30.7.10.1 MCPWM Count Control read address

The CNTCON register controls whether the MCPWM channels are in timer or counter mode, and in counter mode whether the counter advances on rising and/or falling edges on any or all of the three MCI inputs. If timer mode is selected, the counter advances based on the PCLK clock.

This address is read-only. To set or clear the register bits, write ones to the CNTCON_SET or CNTCON_CLR address.

Table 704. MCPWM Count Control read address (CNTCON - 0x400A 005C) bit description

Bit	Symbol	Value	Description	Reset value
0	TC0MCI0_RE		Counter 0 rising edge mode, channel 0.	0
		0	A rising edge on MCI0 does not affect counter 0.	
		1	If MODE0 is 1, counter 0 advances on a rising edge on MCI0.	
1	TC0MCI0_FE		Counter 0 falling edge mode, channel 0.	0
		0	A falling edge on MCI0 does not affect counter 0.	
		1	If MODE0 is 1, counter 0 advances on a falling edge on MCI0.	
2	TC0MCI1_RE		Counter 0 rising edge mode, channel 1.	0
		0	A rising edge on MCI1 does not affect counter 0.	
		1	If MODE0 is 1, counter 0 advances on a rising edge on MCI1.	
3	TC0MCI1_FE		Counter 0 falling edge mode, channel 1.	0
		0	A falling edge on MCI1 does not affect counter 0.	
		1	If MODE0 is 1, counter 0 advances on a falling edge on MCI1.	
4	TC0MCI2_RE		Counter 0 rising edge mode, channel 2.	0
		0	A rising edge on MCI0 does not affect counter 0.	
		1	If MODE0 is 1, counter 0 advances on a rising edge on MCI2.	
5	TC0MCI2_FE		Counter 0 falling edge mode, channel 2.	0
		0	A falling edge on MCI0 does not affect counter 0.	
		1	If MODE0 is 1, counter 0 advances on a falling edge on MCI2.	
6	TC1MCI0_RE		Counter 1 rising edge mode, channel 0.	0
		0	A rising edge on MCI0 does not affect counter 1.	
		1	If MODE1 is 1, counter 1 advances on a rising edge on MCI0.	
7	TC1MCI0_FE		Counter 1 falling edge mode, channel 0.	0
		0	A falling edge on MCI0 does not affect counter 1.	
		1	If MODE1 is 1, counter 1 advances on a falling edge on MCI0.	
8	TC1MCI1_RE		Counter 1 rising edge mode, channel 1.	0
		0	A rising edge on MCI1 does not affect counter 1.	
		1	If MODE1 is 1, counter 1 advances on a rising edge on MCI1.	
9	TC1MCI1_FE		Counter 1 falling edge mode, channel 1.	0
		0	A falling edge on MCI0 does not affect counter 1.	
		1	If MODE1 is 1, counter 1 advances on a falling edge on MCI1.	

Table 704. MCPWM Count Control read address (CNTCON - 0x400A 005C) bit description

Bit	Symbol	Value	Description	Reset value
10	TC1MCI2_RE		Counter 1 rising edge mode, channel 2.	0
		0	A rising edge on MCI2 does not affect counter 1.	
		1	If MODE1 is 1, counter 1 advances on a rising edge on MCI2.	
11	TC1MCI2_FE		Counter 1 falling edge mode, channel 2.	0
		0	A falling edge on MCI2 does not affect counter 1.	
		1	If MODE1 is 1, counter 1 advances on a falling edge on MCI2.	
12	TC2MCI0_RE		Counter 2 rising edge mode, channel 0.	0
		0	A rising edge on MCI0 does not affect counter 2.	
		1	If MODE2 is 1, counter 2 advances on a rising edge on MCI0.	
13	TC2MCI0_FE		Counter 2 falling edge mode, channel 0.	0
		0	A falling edge on MCI0 does not affect counter 2.	
		1	If MODE2 is 1, counter 2 advances on a falling edge on MCI0.	
14	TC2MCI1_RE		Counter 2 rising edge mode, channel 1.	0
		0	A rising edge on MCI1 does not affect counter 2.	
		1	If MODE2 is 1, counter 2 advances on a rising edge on MCI1.	
15	TC2MCI1_FE		Counter 2 falling edge mode, channel 1.	0
		0	A falling edge on MCI1 does not affect counter 2.	
		1	If MODE2 is 1, counter 2 advances on a falling edge on MCI1.	
16	TC2MCI2_RE		Counter 2 rising edge mode, channel 2.	0
		0	A rising edge on MCI2 does not affect counter 2.	
		1	If MODE2 is 1, counter 2 advances on a rising edge on MCI2.	
17	TC2MCI2_FE		Counter 2 falling edge mode, channel 2.	0
		0	A falling edge on MCI2 does not affect counter 2.	
		1	If MODE2 is 1, counter 2 advances on a falling edge on MCI2.	
28:18	-	-	Reserved.	-
29	CNTR0		Channel 0 counter/timer mode.	0
		0	Channel 0 is in timer mode.	
		1	Channel 0 is in counter mode.	
30	CNTR1		Channel 1 counter/timer mode.	0
		0	Channel 1 is in timer mode.	
		1	Channel 1 is in counter mode.	
31	CNTR2		Channel 2 counter/timer mode.	0
		0	Channel 2 is in timer mode.	
		1	Channel 2 is in counter mode.	

30.7.10.2 MCPWM Count Control set address

Writing one(s) to this write-only address sets the corresponding bit(s) in CNTCON.

Table 705. MCPWM Count Control set address (CNTCON_SET - 0x400A 0060) bit description

Bit	Symbol	Description	Reset value
0	TC0MCI0_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
1	TC0MCI0_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
2	TC0MCI1_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
3	TC0MCI1_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
4	TC0MCI2_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
5	TC0MCI2_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
6	TC1MCI0_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
7	TC1MCI0_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
8	TC1MCI1_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
9	TC1MCI1_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
10	TC1MCI2_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
11	TC1MCI2_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
12	TC2MCI0_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
13	TC2MCI0_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
14	TC2MCI1_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
15	TC2MCI1_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
16	TC2MCI2_RE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
17	TC2MCI2_FE_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
28:18	-	Reserved.	
29	CNTR0_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
30	CNTR1_SET	Writing a one sets the corresponding bit in the CNTCON register.	-
31	CNTR2_SET	Writing a one sets the corresponding bit in the CNTCON register.	-

30.7.10.3 MCPWM Count Control clear address

Writing one(s) to this write-only address clears the corresponding bit(s) in CNTCON.

Table 706. MCPWM Count Control clear address (CNTCON_CLR - 0x400A 0064) bit description

Bit	Symbol	Description	Reset value
0	TC0MCI0_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
1	TC0MCI0_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
2	TC0MCI1_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
3	TC0MCI1_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
4	TC0MCI2_RE	Writing a one clears the corresponding bit in the CNTCON register.	-
5	TC0MCI2_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
6	TC1MCI0_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
7	TC1MCI0_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
8	TC1MCI1_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
9	TC1MCI1_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
10	TC1MCI2_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
11	TC1MCI2_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
12	TC2MCI0_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
13	TC2MCI0_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
14	TC2MCI1_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
15	TC2MCI1_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
16	TC2MCI2_RE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
17	TC2MCI2_FE_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
28:18	-	Reserved.	
29	CNTR0_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
30	CNTR1_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-
31	CNTR2_CLR	Writing a one clears the corresponding bit in the CNTCON register.	-

30.7.11 MCPWM Interrupt flag registers

30.7.11.1 MCPWM Interrupt Flags read address

The INTF register includes all MCPWM interrupt flags, which are set when the corresponding hardware event occurs, or when ones are written to the INTF_SET address. When corresponding bits in this register and INTEN are both 1, the MCPWM asserts its interrupt request to the Interrupt Controller module. This address is read-only, but the bits in the underlying register can be modified by writing ones to addresses INTF_SET and INTF_CLR.

Table 707. MCPWM Interrupt flags read address (INTF - 0x400A 0068) bit description

Bit	Symbol	Value	Description	Reset value
0	ILIM0_F		Limit interrupt flag for channel 0.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
1	IMAT0_F		Match interrupt flag for channel 0.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
2	ICAP0_F		Capture interrupt flag for channel 0.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
3	-		Reserved.	-
4	ILIM1_F		Limit interrupt flag for channel 1.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
5	IMAT1_F		Match interrupt flag for channel 1.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
6	ICAP1_F		Capture interrupt flag for channel 1.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
7	-		Reserved.	-

Table 707. MCPWM Interrupt flags read address (INTF - 0x400A 0068) bit description

Bit	Symbol	Value	Description	Reset value
8	ILIM2_F		Limit interrupt flag for channel 2.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
9	IMAT2_F		Match interrupt flag for channel 2.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
10	ICAP2_F		Capture interrupt flag for channel 2.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
14:11	-		Reserved.	-
15	ABORT_F		Fast abort interrupt flag.	0
		0	This interrupt source is not contributing to the MCPWM interrupt request.	
		1	If the corresponding bit in INTEN is 1, the MCPWM module is asserting its interrupt request to the Interrupt Controller.	
31:16	-		Reserved.	-

30.7.11.2 MCPWM Interrupt Flags set address

Writing one(s) to this write-only address sets the corresponding bit(s) in INTF, thus possibly simulating hardware interrupt(s).

Table 708. MCPWM Interrupt Flags set address (INTF_SET - 0x400A 006C) bit description

Bit	Symbol	Description	Reset value
0	ILIM0_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
1	IMAT0_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
2	ICAP0_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
3	-	Reserved.	-
4	ILIM1_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
5	IMAT1_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
6	ICAP1_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
7	-	Reserved.	-

Table 708. MCPWM Interrupt Flags set address (INTF_SET - 0x400A 006C) bit description

Bit	Symbol	Description	Reset value
8	ILIM2_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
9	IMAT2_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
10	ICAP2_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
14:11	-	Reserved.	-
15	ABORT_F_SET	Writing a one sets the corresponding bit in the INTF register, thus possibly simulating hardware interrupt.	-
31:16	-	Reserved.	-

30.7.11.3 MCPWM Interrupt Flags clear address

Writing one(s) to this write-only address sets the corresponding bit(s) in INTF, thus clearing the corresponding interrupt request(s). This is typically done in interrupt service routines.

Table 709. MCPWM Interrupt Flags clear address (INTF_CLR - 0x400A 0070) bit description

Bit	Symbol	Description	Reset value
0	ILIM0_F_CLR	Writing a one clears the corresponding bit in the INTF register, thus clearing the corresponding interrupt request.	-
1	IMAT0_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
2	ICAP0_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
3	-	Reserved.	-
4	ILIM1_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
5	IMAT1_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
6	ICAP1_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
7	-	Reserved.	-
8	ILIM2_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
9	IMAT2_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
10	ICAP2_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
14:11	-	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
15	ABORT_F_CLR	Writing a one clears the corresponding bit in INTEN, thus disabling the interrupt.	-
31:16	-	Reserved.	-

30.7.12 MCPWM Capture clear address

Writing ones to this write-only address clears the selected CAP register(s).

Table 710. MCPWM Capture clear address (CAP_CLR - 0x400A 0074) bit description

Bit	Symbol	Description
0	CAP_CLR0	Writing a 1 to this bit clears the CAP0 register.
1	CAP_CLR1	Writing a 1 to this bit clears the CAP1 register.
2	CAP_CLR2	Writing a 1 to this bit clears the CAP2 register.
31:3	-	Reserved

30.8 Functional description

30.8.1 Pulse-width modulation

Each channel of the MCPWM has two outputs, A and B, that can drive a pair of transistors to switch a controlled point between two power rails. Most of the time the two outputs have opposite polarity, but a dead-time feature can be enabled (on a per-channel basis) to delay both signals' transitions from "passive" to "active" state so that the transistors are never both turned on simultaneously. In a more general view, the states of each output pair can be thought of "high", "low", and "floating" or "up", "down", and "center-off".

Each channel's mapping from "active" and "passive" to "high" and "low" is programmable. After Reset, the three A outputs are passive/low, and the B outputs are active/high.

The MCPWM can perform edge-aligned and center-aligned pulse-width modulation.

Remark: In timer mode, the period of a channel's modulated MCO outputs is determined by its Limit register, and the pulse width at the start of the period is determined by its Match register. If it suits your way of thinking, consider the Limit register to be the "Period register" and the Match register to be the "Pulse Width register".

Edge-aligned PWM without dead-time

In this mode the timer TC counts up from 0 to the value in the LIM register. As shown in Figure 94, the MCO state is "A passive" until the TC matches the Match register, at which point it changes to "A active". When the TC matches the Limit register, the MCO state changes back to "A passive", and the TC is reset and starts counting up again.

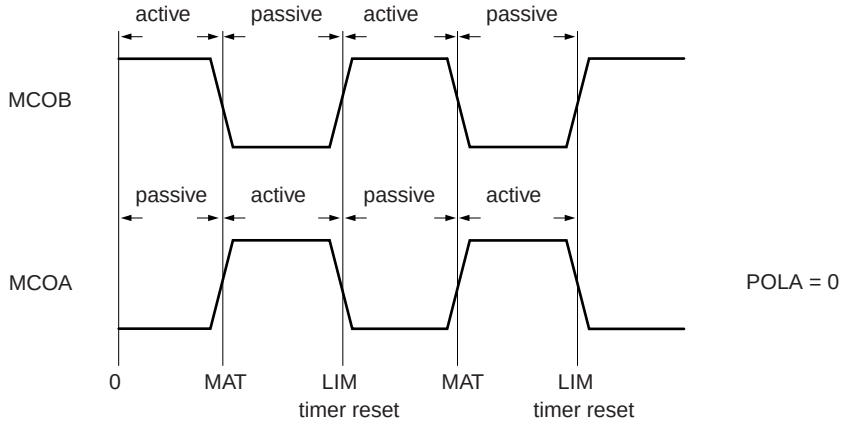


Fig 94. Edge-aligned PWM waveform without dead time, POLA = 0

Center-aligned PWM without dead-time

In this mode the timer TC counts up from 0 to the value in the LIM register, then counts back down to 0 and repeats. As shown in Figure 95, while the timer counts up, the MCO state is "A passive" until the TC matches the Match register, at which point it changes to "A active". When the TC matches the Limit register it starts counting down. When the TC matches the Match register on the way down, the MCO state changes back to "A passive".

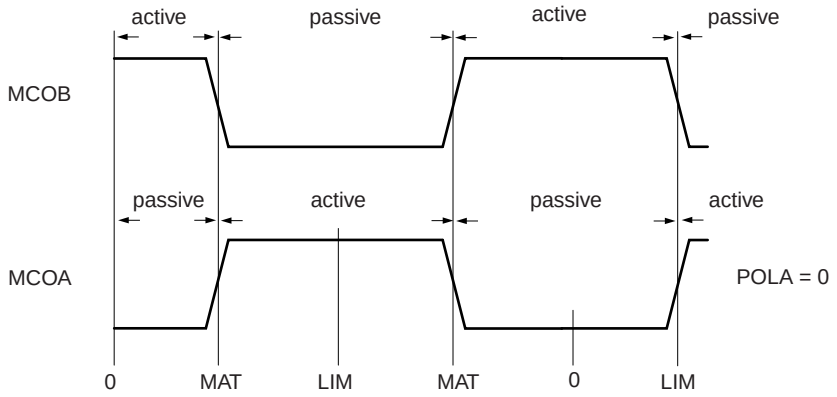


Fig 95. Center-aligned PWM waveform without dead time, POLA = 0

Dead-time counter

When the a channel's DTE bit is set in CON, the dead-time counter delays the passive-to-active transitions of both MCO outputs. The dead-time counter starts counting down, from the channel's DT value (in the DT register) to 0, whenever the channel's A or B output changes from active to passive. The transition of the other output from passive to active is delayed until the dead-time counter reaches 0. During the dead time, the MCOA and MCOB output levels are both passive. Figure 96 shows operation in edge aligned mode with dead time, and Figure 97 shows center-aligned operation with dead time.

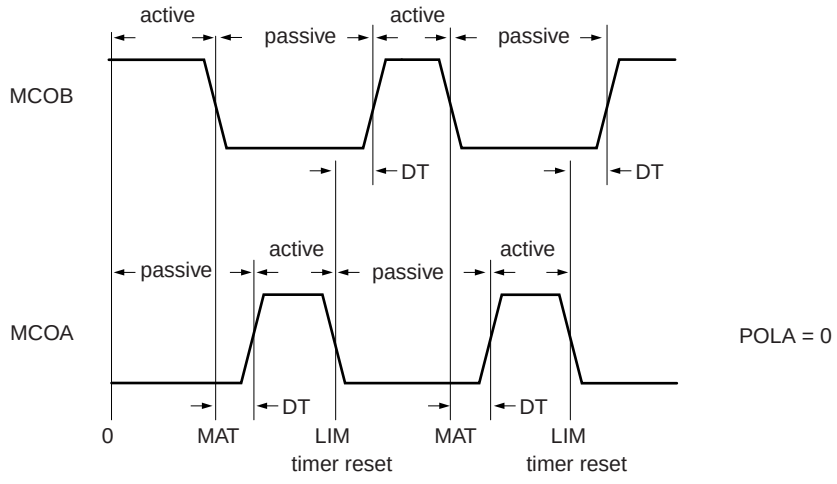


Fig 96. Edge-aligned PWM waveform with dead time, POLA = 0

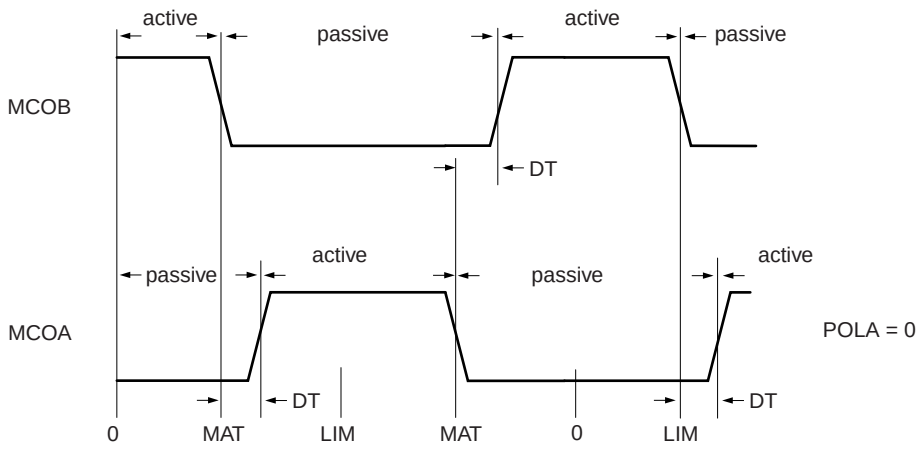


Fig 97. Center-aligned waveform with dead time, POLA = 0

30.8.2 Shadow registers and simultaneous updates

The Limit, Match, and Communication Pattern registers (LIM, MAT, and CP) are implemented as register pairs, each consisting of a write register and an operational register. Software writes into the write registers. The operational registers control the actual operation of each channel and are loaded with the current value in the write registers when the TC starts counting up from 0.

Updating of the functional registers can be disabled by setting a channel's DISUP bit in the CON register. If the DISUP bits are set, the functional registers are not updated until software stops the channel.

If a channel is not running when software writes to its LIM or MAT register, the functional register is updated immediately.

Software can write to a TC register only when its channel is stopped.

30.8.3 Fast Abort (ABORT)

The MCPWM has an external input $\overline{\text{MCABORT}}$. When this input goes low, all six MCO outputs assume their "A passive" states, and the Abort interrupt is generated if enabled. The outputs remain locked in "A passive" state until the ABORT interrupt flag is cleared or the Abort interrupt is disabled. The ABORT flag may not be cleared before the $\overline{\text{MCABORT}}$ input goes high.

In order to clear an ABORT flag, a 1 must be written to bit 15 of the INTF_CLR register. This will remove the interrupt request. The interrupt can also be disabled by writing a 1 to bit 15 of the INTEN_CLR register.

30.8.4 Capture events

Each PWM channel can take a snapshot of its TC when an input signal transitions. Any channel may use any combination of rising and/or falling edges on any or all of the MCI0-2 inputs as a capture event, under control of the CAPCON register. Rising or falling edges on the inputs are detected synchronously with respect to PCLK.

If a channel's HNF bit in the CAPCON register is set to enable "noise filtering", a selected edge on an MCI pin starts the dead-time counter for that channel, and the capture event actions described below are delayed until the dead-time counter reaches 0. This function is targeted specifically for performing three-phase brushless DC motor control with Hall sensors.

A capture event on a channel (possibly delayed by HNF) causes the following:

- The current value of the TC is stored in the Capture register (CAP).
- If the channel's capture event interrupt is enabled (see Table 701), the capture event interrupt flag is set.
- If the channel's RT bit is set in the CAPCON register, enabling reset on a capture event, the input event has the same effect as matching the channel's TC to its LIM register. This includes resetting the TC and switching the MCO pin(s) in edge-aligned mode as described in 30.7.4 and 30.8.1.

30.8.5 External event counting (Counter mode)

If a channel's MODE bit is 1 in CNTCON, its TC is incremented by rising and/or falling edge(s) (synchronously detected) on the MCI0-2 input(s), rather than by PCLK. The PWM functions and capture functions are unaffected.

30.8.6 Three-phase DC mode

The three-phase DC mode is selected by setting the DCMODE bit in the CON register.

In this mode, the internal MCOA0 signal can be routed to any or all of the MCO outputs. Each MCO output is masked by a bit in the current commutation pattern register CP. If a bit in the CP register is 0, its output pin has the logic level for the passive state of output MCOA0. The polarity of the off state is determined by the POLA0 bit.

All MCO outputs that have 1 bits in the CP register are controlled by the internal MCOA0 signal.

The three MCOB output pins are inverted when the INVBDC bit is 1 in the CON register. This feature accommodates bridge-drivers that have active-low inputs for the low-side switches.

The CP register is implemented as a shadow register pair, so that changes to the active communication pattern occur at the beginning of a new PWM cycle. See 30.7.4 and 30.8.2 for more about writing and reading such registers.

Figure 98 shows sample waveforms of the MCO outputs in three-phase DC mode. Bits 1 and 3 in the CP register (corresponding to outputs MCOB1 and MCOB0) are set to 0 so that these outputs are masked and in the off state. Their logic level is determined by the POLA0 bit (here, POLA0 = 0 so the passive state is logic LOW). The INVBDC bit is set to 0 (logic level not inverted) so that the B output have the same polarity as the A outputs. Note that this mode differs from other modes in that the MCOB outputs are **not** the opposite of the MCOA outputs.

In the situation shown in Figure 98, bits 0, 2, 4, and 5 in the CP register are set to 1. That means that MCOA1 and both MCO outputs for channel 2 follow the MCOA0 signal.

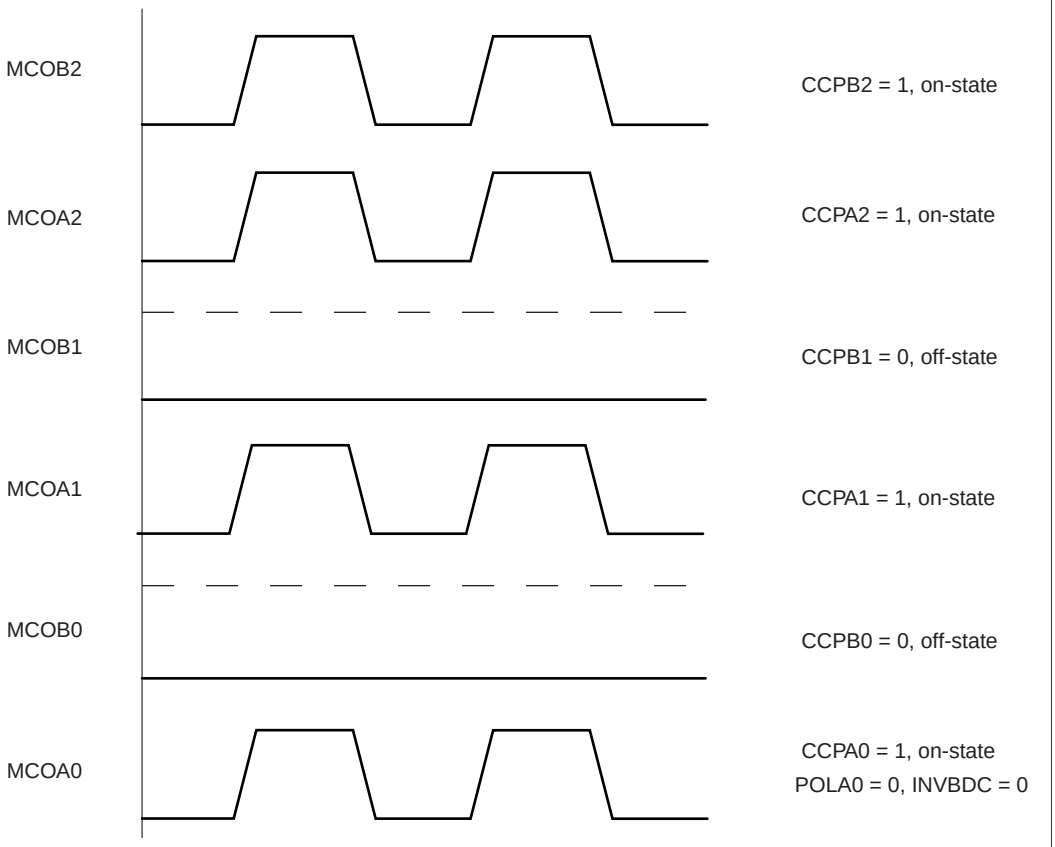


Fig 98. Three-phase DC mode sample waveforms

30.8.7 Three phase AC mode

The three-phase AC-mode is selected by setting the ACMODE bit in the CON register.

In this mode, the value of channel 0's TC is routed to all channels for comparison with their MAT registers. (The LIM1-2 registers are not used.)

Each channel controls its MCO output by comparing its MAT value to TC0.

Figure 99 shows sample waveforms for the six MCO outputs in three-phase AC mode. The POLA bits are set to 0 for all three channels, so that for all MCO outputs the active levels are high and the passive levels are low. Each channel has a different MAT value which is compared to the TC0 value. In this mode the period value is identical for all three channels and is determined by LIM0. The dead-time mode is disabled.

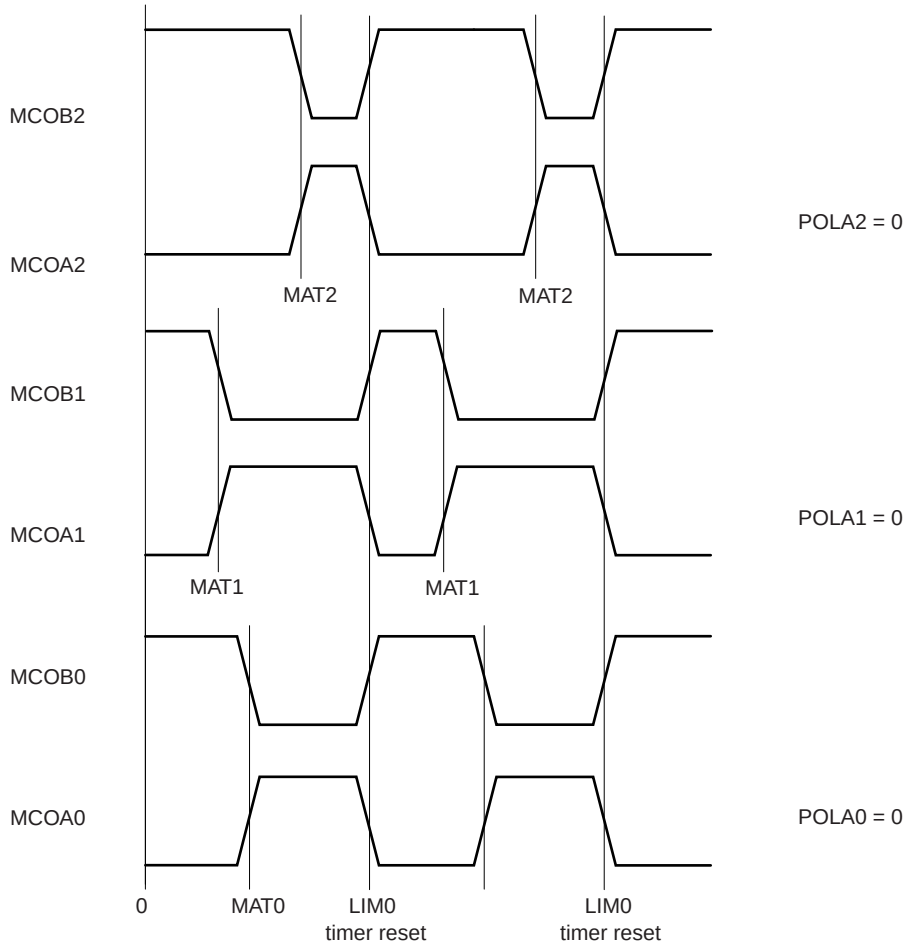


Fig 99. Three-phase AC mode sample waveforms, edge aligned PWM mode

30.8.8 Interrupts

The MCPWM includes 10 possible interrupt sources:

- When any channel's TC matches its Match register.
- When any channel's TC matches its Limit register.
- When any channel captures the value of its TC into its Capture register, because a selected edge occurs on any of MCI0-2.
- When all three channels' outputs are forced to "A passive" state because the MCABORT pin goes low.

Section 30.7.9 "MCPWM Interrupt registers" explains how to enable these interrupts, and Section 30.7.2 "PWM Capture Control register" describes how to map edges on the MCI0-2 inputs to "capture events" on the three channels.

31.1 How to read this chapter

The QEI is available on all LPC43xx parts.

31.2 Basic configuration

The QEI is configured as follows:

- See Table 711 for clocking and power control.
- The QEI is reset by the QEI_RST (reset #39).
- The QEI interrupt is connected to slot # 15 in the Event router.

Table 711. QEI clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the QEI register interface and QEI peripheral clock.	BASE_M4_CLK	CLK_M4_QEI	up to 204 MHz

31.3 Features

This Quadrature Encoder Interface (QEI) has the following features:

- Tracks encoder position.
- Increments/ decrements depending on direction.
- Programmable for 2X or 4X position counting.
- Velocity capture using built-in timer.
- Velocity compare function with less than interrupt.
- Uses 32-bit registers for position and velocity.
- Three position compare registers with interrupts.
- Index counter for revolution counting.
- Index compare register with interrupts.
- Can combine index and position interrupts to produce an interrupt for whole and partial revolution displacement.
- Digital filter with programmable delays for encoder input signals.
- Can accept decoded signal inputs (clock and direction).

31.4 Introduction

A quadrature encoder, also known as a 2-channel incremental encoder, converts angular displacement into two pulse signals. By monitoring both the number of pulses and the relative phase of the two signals, you can track the position, direction of rotation, and velocity. In addition, a third channel, or index signal, can be used to reset the position counter. This quadrature encoder interface module decodes the digital pulses from a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture the velocity of the encoder wheel.

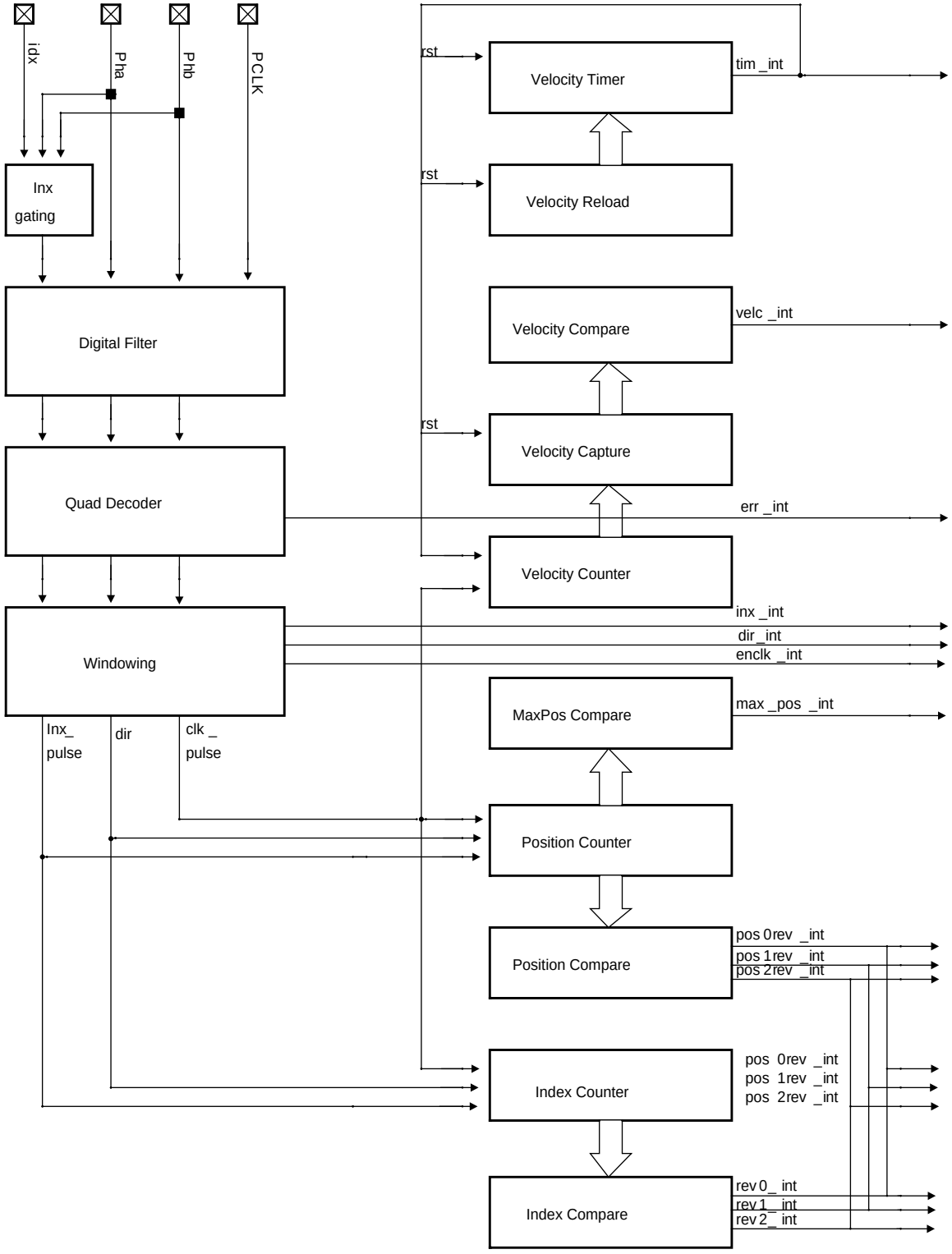


Fig 100.Encoder interface block diagram

31.5 Pin description

Table 712. QEI pin description

Function name	I/O	Description
QEI_A	I	Used as the Phase A (PhA) input to the Quadrature Encoder Interface.
QEI_B	I	Used as the Phase B (PhB) input to the Quadrature Encoder Interface.
QEI_IDX	I	Used as the Index (IDX) input to the Quadrature Encoder Interface.

31.6 Register description

Table 713. Register overview: QEI (base address 0x400C 6000)

Name	Access	Address offset	Description	Reset value	Reference
Control registers					
CON	WO	0x000	Control register	0	Table 714
STAT	RO	0x004	Encoder status register	0	Table 715
CONF	R/W	0x008	Configuration register	0x000F 0000	Table 716
Position, index, and timer registers					
POS	RO	0x00C	Position register	0	Table 717
MAXPOS	R/W	0x010	Maximum position register	0	Table 718
CMPOS0	R/W	0x014	position compare register 0	0xFFFF FFFF	Table 719
CMPOS1	R/W	0x018	position compare register 1	0xFFFF FFFF	Table 720
CMPOS2	R/W	0x01C	position compare register 2	0xFFFF FFFF	Table 721
INXCNT	RO	0x020	Index count register	0	Table 722
INXCMP0	R/W	0x024	Index compare register 0	0xFFFF FFFF	Table 723
LOAD	R/W	0x028	Velocity timer reload register	0xFFFF FFFF	Table 724
TIME	RO	0x02C	Velocity timer register	0xFFFF FFFF	Table 725
VEL	RO	0x030	Velocity counter register	0	Table 726
CAP	RO	0x034	Velocity capture register	0xFFFF FFFF	Table 727
VELCOMP	R/W	0x038	Velocity compare register	0	Table 728
FILTERPHA	R/W	0x03C	Digital filter register on input phase A (QEI_A)	0	Table 729
FILTERPHB	R/W	0x040	Digital filter register on input phase B (QEI_B)	0	Table 730
FILTERINX	R/W	0x044	Digital filter register on input index (QEI_IDX)	0	Table 731
WINDOW	R/W	0x048	Index acceptance window register	0x0000 0000	Table 732
INXCMP1	R/W	0x04C	Index compare register 1	0xFFFF FFFF	Table 733
INXCMP2	R/W	0x050	Index compare register 2	0xFFFF FFFF	Table 734
Interrupt registers					
IEC	WO	0xFD8	Interrupt enable clear register	0	Table 735
IES	WO	0xFDC	Interrupt enable set register	0	Table 736

Table 713. Register overview: QEI (base address 0x400C 6000)

Name	Access	Address offset	Description	Reset value	Reference
INTSTAT	RO	0xFE0	Interrupt status register	0	Table 737
IE	RO	0xFE4	Interrupt enable register	0	Table 738
CLR	WO	0xFE8	Interrupt status clear register	0	Table 739
SET	WO	0xFEC	Interrupt status set register	0	Table 740

31.6.1 Control registers

31.6.1.1 QEI Control register

This register contains bits which control the operation of the position and velocity counters of the QEI module.

Table 714: QEI Control register (CON - address 0x400C 6000) bit description

Bit	Symbol	Description	Reset value
0	RESP	Reset position counter. When set = 1, resets the position counter to all zeros. Autoclears when the position counter is cleared.	0
1	RESPI	Reset position counter on index. When set = 1, resets the position counter to all zeros when an index pulse occurs. Autoclears when the position counter is cleared.	0
2	RESV	Reset velocity. When set = 1, resets the velocity counter to all zeros and reloads the velocity timer. Autoclears when the velocity counter is cleared.	0
3	RESI	Reset index counter. When set = 1, resets the index counter to all zeros. Autoclears when the index counter is cleared.	0
31:4	-	reserved	0

31.6.1.2 QEI Status register

This register provides the status of the encoder interface.

Table 715: QEI Interrupt Status register (STAT - address 0x400C 6004) bit description

Bit	Symbol	Description	Reset value
0	DIR	Direction bit. In combination with DIRINV bit indicates forward or reverse direction. See Table 743.	
31:1	-	reserved	0

31.6.1.3 QEI Configuration register

This register contains the configuration of the QEI module.

Table 716: QEI Configuration register (CONF - address 0x400C 6008) bit description

Bit	Symbol	Description	Reset value
0	DIRINV	Direction invert. When = 1, complements the DIR bit.	0
1	SIGMODE	Signal Mode. When = 0, PhA and PhB function as quadrature encoder inputs. When = 1, PhA functions as the direction signal and PhB functions as the clock signal.	0
2	CAPMODE	Capture Mode. When = 0, only PhA edges are counted (2X). When = 1, BOTH PhA and PhB edges are counted (4X), increasing resolution but decreasing range.	0
3	INVINX	Invert Index. When set, inverts the sense of the index input.	0
4	CRESPI	Continuously reset position counter on index. When set = 1, resets the position counter to all zeros when an index pulse occurs at the next position increase (recalibration). Auto-clears when the position counter is cleared.	0
15:5	-	Reserved	0
19:16	INXGATE	Index gating configuration: when INXGATE(19)=1, pass the index when Pha=0 and Phb=0, else block. when INXGATE(18)=1, pass the index when Pha=0 and Phb=1, else block. when INXGATE(17)=1, pass the index when Pha=1 and Phb=1, else block. when INXGATE(16)=1, pass the index when Pha=1 and Phb=0, else block.	1111
31:20	-	reserved	0

31.6.2 Position, index and timer registers

31.6.2.1 QEI Position register

This register contains the current value of the encoder position. Increments or decrements when encoder counts occur, depending on the direction of rotation.

Table 717. QEI Position register (POS - address 0x400C 600C) bit description

Bit	Symbol	Description	Reset value
31:0	POS	Current position value.	0

31.6.2.2 QEI Maximum Position register

This register contains the maximum value of the encoder position. In forward rotation the position register resets to zero when the position register exceeds this value. In reverse rotation the position register resets to this value when the position register decrements from zero.

Table 718. QEI Maximum Position register (MAXPOS - address 0x400C 6010) bit description

Bit	Symbol	Description	Reset value
31:0	MAXPOS	Maximum position value.	0

31.6.2.3 QEI Position Compare register 0

This register contains a position compare value. This value is compared against the current value of the position register. Interrupts can be enabled to interrupt when the compare value is less than, equal to, or greater than the current value of the position register.

Table 719. QEI Position Compare register 0 (CMPOS0 - address 0x400C 6014) bit description

Bit	Symbol	Description	Reset value
31:0	PCMP0	Position compare value 0.	0xFFFF FFFF

31.6.2.4 QEI Position Compare register 1

This register contains a position compare value. This value is compared against the current value of the position register. Interrupts can be enabled to interrupt when the compare value is less than, equal to, or greater than the current value of the position register.

Table 720. QEI Position Compare register 1 (CMPOS1 - address 0x400C 6018) bit description

Bit	Symbol	Description	Reset value
31:0	PCMP1	Position compare value 1.	0xFFFF FFFF

31.6.2.5 QEI Position Compare register 2

This register contains a position compare value. This value is compared against the current value of the position register. Interrupts can be enabled to interrupt when the compare value is less than, equal to, or greater than the current value of the position register.

Table 721. QEI Position Compare register 2 (CMPOS2 - address 0x400C 601C) bit description

Bit	Symbol	Description	Reset value
31:0	PCMP2	Position compare value 2.	0xFFFF FFFF

31.6.2.6 QEI Index Count register

This register contains the current value of the encoder position. Increments or decrements when encoder counts occur, depending on the direction of rotation.

Table 722. QEI Index Count register (INXCNT- address 0x400C 6020) bit description

Bit	Symbol	Description	Reset value
31:0	ENCPOS	Current encoder position value.	0

31.6.2.7 QEI Index Compare register 0

This register contains an index compare value. This value is compared against the current value of the index count register. Interrupts can be enabled to interrupt when the compare value is less than, equal to, or greater than the current value of the index count register.

Table 723. QEI Index Compare register 0(INXCMP0 - address 0x400C 6024) bit description

Bit	Symbol	Description	Reset value
31:0	ICMP0	Index compare value.	0xFFFF FFFF

31.6.2.8 QEI Timer Reload register

This register contains the reload value of the velocity timer. When the timer (QEITIME) overflows or the RESV bit is asserted, this value is loaded into the timer (QEITIME).

Table 724. QEI Timer Load register (LOAD - address 0x400C 6028) bit description

Bit	Symbol	Description	Reset value
31:0	VELLOAD	Current velocity timer load value.	0xFFFF FFFF

31.6.2.9 QEI Timer register

This register contains the current value of the velocity timer. When this timer overflows the value of velocity counter (QEIVEL) is stored in the velocity capture register (QEICAP), the velocity counter is reset to zero, the timer is reloaded with the value stored in the velocity reload register (QEILOAD), and the velocity interrupt (TIM_Int) is asserted.

Table 725. QEI Timer register (TIME - address 0x400C 602C) bit description

Bit	Symbol	Description	Reset value
31:0	VELVAL	Current velocity timer value.	0xFFFF FFFF

31.6.2.10 QEI Velocity register

This register contains the running count of velocity pulses for the current time period. When the velocity timer (QEITIME) overflows the contents of this register is captured in the velocity capture register (QEICAP). After capture, this register is set to zero. This register is also reset when the velocity reset bit (RESV) is asserted.

Table 726. QEI Velocity register (VEL - address 0x400C 6030) bit description

Bit	Symbol	Description	Reset value
31:0	VELPC	Current velocity pulse count.	0x0

31.6.2.11 QEI Velocity Capture register

This register contains the most recently measured velocity of the encoder. This corresponds to the number of velocity pulses counted in the previous velocity timer period.The current velocity count is latched into this register when the velocity timer overflows.

Table 727. QEI Velocity Capture register (CAP - address 0x400C 6034) bit description

Bit	Symbol	Description	Reset value
31:0	VELCAP	Velocity capture value.	0xFFFF FFFF

31.6.2.12 QEI Velocity Compare register

This register contains a velocity compare value. This value is compared against the captured velocity in the velocity capture register. If the capture velocity is less than the value in this compare register, a velocity compare interrupt (VELC_Int) will be asserted, if enabled.

Table 728. QEI Velocity Compare register (VELCOMP - address 0x400C 6038) bit description

Bit	Symbol	Description	Reset value
31:0	VELCMP	Velocity compare value.	0x0

31.6.2.13 QEI Digital filter on phase A input register

This register contains the sampling count for the digital filter. A sampling count of zero bypasses the filter.

Table 729. QEI Digital filter on phase A input register (FILTERPHA - 0x400C 603C) bit description

Bit	Symbol	Description	Reset value
31:0	FILTA	Digital filter sampling delay	0x0

31.6.2.14 QEI Digital filter on phase B input register

This register contains the sampling count for the digital filter. A sampling count of zero bypasses the filter.

Table 730. QEI Digital filter on phase B input register (FILTERPHB - 0x400C 6040) bit description

Bit	Symbol	Description	Reset value
31:0	FILTB	Digital filter sampling delay	0x0

31.6.2.15 QEI Digital filter on index input register

This register contains the sampling count for the digital filter. A sampling count of zero bypasses the filter.

Table 731. QEI Digital filter on index input register (FILTERINX - 0x400C 6044) bit description

Bit	Symbol	Description	Reset value
31:0	FITLINX	Digital filter sampling delay	0x0

31.6.2.16 QEI Index acceptance window register

This register contains the width of the index acceptance window, when the index and the phase / clock edges fall nearly together. If the activating phase / clock edge falls before the Index, but within the window, the (re)calibration will be activated on that clock/phase edge.

Table 732. QEI Index acceptance window register (WINDOW - 0x400C 6048) bit description

Bit	Symbol	Description	Reset value
31:0	WINDOW	Index acceptance window width	0x0

31.6.2.17 QEI Index Compare register 1

This register contains an index compare value. This value is compared against the current value of the index count register. Interrupts can be enabled to interrupt when the compare value is less than, equal to, or greater than the current value of the index count register.

Table 733. QEI Index Compare register 1 (INXCMP1 - address 0x400C 604C) bit description

Bit	Symbol	Description	Reset value
31:0	ICMP1	Index compare value 1.	0xFFFF FFFF

31.6.2.18 QEI Index Compare register 2

This register contains an index compare value. This value is compared against the current value of the index count register. Interrupts can be enabled to interrupt when the compare value is less than, equal to, or greater than the current value of the index count register.

Table 734. QEI Index Compare register 2 (INXCMP2 - address 0x400C 6050) bit description

Bit	Symbol	Description	Reset value
31:0	ICMP2	Index compare value 2.	0xFFFF FFFF

31.6.3 Interrupt registers

31.6.3.1 QEI Interrupt Enable Clear register

Writing a 1 to a bit in this register clears the corresponding bit in the QEI Interrupt Enable register (QEIIE).

Table 735: QEI Interrupt Enable Clear register (IEC - address 0x400C 6FD8) bit description

Bit	Symbol	Description	Reset value
0	INX_EN	Indicates that an index pulse was detected.	0
1	TIM_EN	Indicates that a velocity timer overflow occurred	0
2	VELC_EN	Indicates that captured velocity is less than compare velocity.	0
3	DIR_EN	Indicates that a change of direction was detected.	0
4	ERR_EN	Indicates that an encoder phase error was detected.	0
5	ENCLK_EN	Indicates that and encoder clock pulse was detected.	0
6	POS0_INT	Indicates that the position 0 compare value is equal to the current position.	0
7	POS1_INT	Indicates that the position 1compare value is equal to the current position.	0
8	POS2_INT	Indicates that the position 2 compare value is equal to the current position.	0
9	REV0_INT	Indicates that the index 0 compare value is equal to the current index count.	0
10	POS0REV_INT	Combined position 0 and revolution count interrupt. Set when both the POS0_Int bit is set and the REV_0 is set.	0
11	POS1REV_INT	Combined position 1 and revolution count interrupt. Set when both the POS1_Int bit is set and the REV_1 is set.	0
12	POS2REV_Int	Combined position 2 and revolution count interrupt. Set when both the POS2_Int bit is set and the REV_2 is set.	0
13	REV1_INT	Indicates that the index 1 compare value is equal to the current index count.	0
14	REV2_INT	Indicates that the index 2 compare value is equal to the current index count.	0
15	MAXPOS_INT	Indicates that the current position count goes through the MAXPOS value to zero in forward direction, or through zero to MAXPOS in backward direction.	0
31:16	-	Reserved	0

31.6.3.2 QEI Interrupt Enable Set register

Writing a 1 to a bit in this register sets the corresponding bit in the QEI Interrupt Enable register (QEIIE).

Table 736: QEI Interrupt Enable Set register (IES - address 0x400C 6FDC) bit description

Bit	Symbol	Description	Reset value
0	INX_EN	Indicates that an index pulse was detected.	0
1	TIM_EN	Indicates that a velocity timer overflow occurred	0
2	VELC_EN	Indicates that captured velocity is less than compare velocity.	0
3	DIR_EN	Indicates that a change of direction was detected.	0
4	ERR_EN	Indicates that an encoder phase error was detected.	0
5	ENCLK_EN	Indicates that and encoder clock pulse was detected.	0
6	POS0_Int	Indicates that the position 0 compare value is equal to the current position.	0
7	POS1_Int	Indicates that the position 1compare value is equal to the current position.	0

Table 736: QEI Interrupt Enable Set register (IES - address 0x400C 6FDC) bit description

Bit	Symbol	Description	Reset value
8	POS2_Int	Indicates that the position 2 compare value is equal to the current position.	0
9	REV_Int	Indicates that the index compare value is equal to the current index count.	0
10	POS0REV_Int	Combined position 0 and revolution count interrupt. Set when both the POS0_Int bit is set and the REV_Int is set.	0
11	POS1REV_Int	Combined position 1 and revolution count interrupt. Set when both the POS1_Int bit is set and the REV_Int is set.	0
12	POS2REV_Int	Combined position 2 and revolution count interrupt. Set when both the POS2_Int bit is set and the REV_Int is set.	0
13	REV1_Int	Indicates that the index 1 compare value is equal to the current index count.	0
14	REV2_Int	Indicates that the index 2 compare value is equal to the current index count.	0
15	MAXPOS_Int	Indicates that the current position count goes through the MAXPOS value to zero in forward direction, or through zero to MAXPOS in backward direction.	0
31:16	-	Reserved	0

31.6.3.3 QEI Interrupt Status register

This register provides the status of the encoder interface and the current set of interrupt sources that are asserted to the controller. Bits set to 1 indicate the latched events that have occurred; a zero bit indicates that the event in question has not occurred. Writing a 0 to a bit position clears the corresponding interrupt.

Table 737: QEI Interrupt Status register (INTSTAT - address 0x400C 6FE0) bit description

Bit	Symbol	Description	Reset value
0	INX_Int	Indicates that an index pulse was detected.	0
1	TIM_Int	Indicates that a velocity timer overflow occurred	0
2	VELC_Int	Indicates that captured velocity is less than compare velocity.	0
3	DIR_Int	Indicates that a change of direction was detected.	0
4	ERR_Int	Indicates that an encoder phase error was detected.	0
5	ENCLK_Int	Indicates that and encoder clock pulse was detected.	0
6	POS0_Int	Indicates that the position 0 compare value is equal to the current position.	0
7	POS1_Int	Indicates that the position 1compare value is equal to the current position.	0
8	POS2_Int	Indicates that the position 2 compare value is equal to the current position.	0
9	REV_Int	Indicates that the index compare value is equal to the current index count.	0
10	POS0REV_Int	Combined position 0 and revolution count interrupt. Set when both the POS0_Int bit is set and the REV_Int is set.	0
11	POS1REV_Int	Combined position 1 and revolution count interrupt. Set when both the POS1_Int bit is set and the REV_Int is set.	0
12	POS2REV_Int	Combined position 2 and revolution count interrupt. Set when both the POS2_Int bit is set and the REV_Int is set.	0
13	REV1_Int	Indicates that the index 1 compare value is equal to the current index count.	0
14	REV2_Int	Indicates that the index 2 compare value is equal to the current index count.	0
15	MAXPOS_Int	Indicates that the current position count goes through the MAXPOS value to zero in forward direction, or through zero to MAXPOS in backward direction.	0
31:16	-	Reserved	0

31.6.3.4 QEI Interrupt Enable register

This register enables interrupt sources. Bits set to 1 enable the corresponding interrupt; a 0 bit disables the corresponding interrupt.

Table 738: QEI Interrupt Enable register (IE - address 0x400C 6FE4) bit description

Bit	Symbol	Description	Reset value
0	INX_Int	Indicates that an index pulse was detected.	0
1	TIM_Int	Indicates that a velocity timer overflow occurred	0
2	VELC_Int	Indicates that captured velocity is less than compare velocity.	0
3	DIR_Int	Indicates that a change of direction was detected.	0
4	ERR_Int	Indicates that an encoder phase error was detected.	0
5	ENCLK_Int	Indicates that and encoder clock pulse was detected.	0
6	POS0_Int	Indicates that the position 0 compare value is equal to the current position.	0
7	POS1_Int	Indicates that the position 1compare value is equal to the current position.	0
8	POS2_Int	Indicates that the position 2 compare value is equal to the current position.	0
9	REV_Int	Indicates that the index compare value is equal to the current index count.	0
10	POS0REV_Int	Combined position 0 and revolution count interrupt. Set when both the POS0_Int bit is set and the REV_Int is set.	0
11	POS1REV_Int	Combined position 1 and revolution count interrupt. Set when both the POS1_Int bit is set and the REV_Int is set.	0
12	POS2REV_Int	Combined position 2 and revolution count interrupt. Set when both the POS2_Int bit is set and the REV_Int is set.	0
13	REV1_Int	Indicates that the index 1 compare value is equal to the current index count.	0
14	REV2_Int	Indicates that the index 2 compare value is equal to the current index count.	0
15	MAXPOS_Int	Indicates that the current position count goes through the MAXPOS value to zero in forward direction, or through zero to MAXPOS in backward direction.	0
31:16	-	Reserved	0

31.6.3.5 QEI Interrupt Status Clear register

Writing a 1 to a bit in this register clears the corresponding bit in the QEI Interrupt Status register (STAT).

Table 739: QEI Interrupt Status Clear register (CLR - 0x400C 6FE8) bit description

Bit	Symbol	Description	Reset value
0	INX_Int	Indicates that an index pulse was detected.	0
1	TIM_Int	Indicates that a velocity timer overflow occurred	0
2	VELC_Int	Indicates that captured velocity is less than compare velocity.	0
3	DIR_Int	Indicates that a change of direction was detected.	0
4	ERR_Int	Indicates that an encoder phase error was detected.	0
5	ENCLK_Int	Indicates that and encoder clock pulse was detected.	0
6	POS0_Int	Indicates that the position 0 compare value is equal to the current position.	0
7	POS1_Int	Indicates that the position 1compare value is equal to the current position.	0
8	POS2_Int	Indicates that the position 2 compare value is equal to the current position.	0
9	REV_Int	Indicates that the index compare value is equal to the current index count.	0

Table 739: QEI Interrupt Status Clear register (CLR - 0x400C 6FE8) bit description

Bit	Symbol	Description	Reset value
10	POS0REV_Int	Combined position 0 and revolution count interrupt. Set when both the POS0_Int bit is set and the REV_Int is set.	0
11	POS1REV_Int	Combined position 1 and revolution count interrupt. Set when both the POS1_Int bit is set and the REV_Int is set.	0
13	REV1_Int	Indicates that the index 1 compare value is equal to the current index count.	0
14	REV2_Int	Indicates that the index 2 compare value is equal to the current index count.	0
15	MAXPOS_Int	Indicates that the current position count goes through the MAXPOS value to zero in forward direction, or through zero to MAXPOS in backward direction.	0
31:16	-	Reserved	0

31.6.3.6 QEI Interrupt Status Set register

Writing a one to a bit in this register sets the corresponding bit in the QEI Interrupt Status register (STAT).

Table 740: QEI Interrupt Status Set register (SET - address 0x400C 6FEC) bit description

Bit	Symbol	Description	Reset value
0	INX_Int	Indicates that an index pulse was detected.	0
1	TIM_Int	Indicates that a velocity timer overflow occurred	0
2	VELC_Int	Indicates that captured velocity is less than compare velocity.	0
3	DIR_Int	Indicates that a change of direction was detected.	0
4	ERR_Int	Indicates that an encoder phase error was detected.	0
5	ENCLK_Int	Indicates that and encoder clock pulse was detected.	
6	POS0_Int	Indicates that the position 0 compare value is equal to the current position.	0
7	POS1_Int	Indicates that the position 1compare value is equal to the current position.	0
8	POS2_Int	Indicates that the position 2 compare value is equal to the current position.	0
9	REV_Int	Indicates that the index compare value is equal to the current index count.	0
10	POS0REV_Int	Combined position 0 and revolution count interrupt. Set when both the POS0_Int bit is set and the REV_Int is set.	0
11	POS1REV_Int	Combined position 1 and revolution count interrupt. Set when both the POS1_Int bit is set and the REV_Int is set.	0
12	POS2REV_Int	Combined position 2 and revolution count interrupt. Set when both the POS2_Int bit is set and the REV_Int is set.	0
13	REV1_Int	Indicates that the index 1 compare value is equal to the current index count.	0
14	REV2_Int	Indicates that the index 2 compare value is equal to the current index count.	0
15	MAXPOS_Int	Indicates that the current position count goes through the MAXPOS value to zero in forward direction, or through zero to MAXPOS in backward direction.	0
31:16	-	Reserved	0

31.7 Functional description

The QEI module interprets the two-bit gray code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture the velocity of the encoder wheel.

31.7.1 Input signals

The QEI module supports two modes of signal operation: quadrature phase mode and clock/direction mode. In quadrature phase mode, the encoder produces two clocks that are 90 degrees out of phase; the edge relationship is used to determine the direction of rotation. In clock/direction mode, the encoder produces a clock signal to indicate steps and a direction signal to indicate the direction of rotation.).

This mode is determined by the SigMode bit of the QEI Control (CON) register (See Table 714). When the SigMode bit = 1, the quadrature decoder is bypassed and the PhA pin functions as the direction signal and PhB pin functions as the clock signal for the counters, etc. When the SigMode bit = 0, the PhA pin and PhB pins are decoded by the quadrature decoder. In this mode the quadrature decoder produces the direction and clock signals for the counters, etc. In both modes the direction signal is subject to the effects of the direction invert (DIRINV) bit.

31.7.1.1 Quadrature input signals

When edges on PhA lead edges on PhB, the position counter is incremented. When edges on PhB lead edges on PhA, the position counter is decremented. When a rising and falling edge pair is seen on one of the phases without any edges on the other, the direction of rotation has changed.

Table 741. Encoder states

Phase A	Phase B	state
1	0	1
1	1	2
0	1	3
0	0	4

Table 742. Encoder state transitions[1]

from state	to state	Direction
1	2	positive
2	3	
3	4	
4	1	
4	3	negative
3	2	
2	1	
1	4	

[1] All other state transitions are illegal and should set the ERR bit.

Interchanging of the PhA and PhB input signals are compensated by complementing the DIR bit. When set = 1, the direction inversion bit (DIRINV) complements the DIR bit.

Table 743. Encoder direction

DIR bit	DIRINV bit	direction
0	0	forward
1	0	reverse
0	1	reverse
1	1	forward

Figure 101 shows how quadrature encoder signals equate to direction and count.

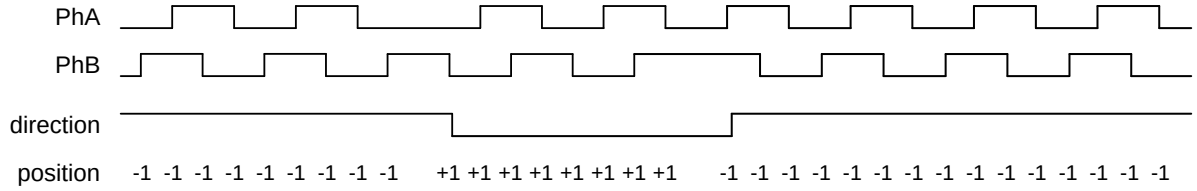


Fig 101. Quadrature Encoder Basic Operation

31.7.1.2 Digital input filtering

All three encoder inputs (PhA, PhB, and index) require digital filtering. The number of sample clocks is user programmable from 1 to 4,294,967,295 (0xFFFF FFFF). In order for a transition to be accepted, the input signal must remain in new state for the programmed number of sample clocks.

31.7.2 Position capture

The capture mode for the position integrator can be set to update the position counter on every edge of the PhA signal or to update on every edge of both PhA and PhB. Updating the position counter on every PhA and PhB provides more positional resolution at the cost of less range in the positional counter.

The position integrator and velocity capture can be independently enabled. Alternatively, the phase signals can be interpreted as a clock and direction signal as output by some encoders.

The position counter is automatically reset on one of three conditions. Incrementing past the maximum position value (MAXPOS) will reset the position counter to zero. If the reset on index bit (RESPI) is set, sensing the index pulse for the first time will once reset the position counter to zero after the next positional increase (calibrate). If the continuously reset on index bit (CRESPI) is set, sensing the index pulse will continuously reset the position counter to zero after the next positional increase (recalibrate).

31.7.3 Velocity capture

The velocity capture has a programmable timer and a capture register. It counts the number of phase edges (using the same configuration as for the position integrator) in a given time period. When the velocity timer (TIME) overflows the contents of the velocity counter (VEL) are transferred to the capture (CAP) register. The velocity counter is then

cleared. The velocity timer is loaded with the contents of the velocity reload register (LOAD). Finally, the velocity interrupt (TIM_Int) is asserted. The number of edges counted in a given time period is directly proportional to the velocity of the encoder.

Setting the reset velocity bit (RESV) will clear the velocity counter, reset the velocity capture register to 0xFFFF FFFF, and load the velocity timer with the contents of the velocity reload register (LOAD).

The following equation converts the velocity counter value into an RPM value:

RPM = (PCLK × Speed × 60) / Load × PPR × Edges)

where:

- PCLK is the QEI controller clock.
- PPR is the number of pulses per revolution of the physical encoder.
- Edges is 2 or 4, based on the capture mode set in the CON register (2 for CapMode set to 0 and 4 for CapMode set to 1)

For example, consider a motor running at 600 rpm. A 2048 pulse per revolution quadrature encoder is attached to the motor, producing 8192 phase edges per revolution. With clocking on both PhA and PhB edges, this results in 81,920 pulses per second (the motor turns 10 times per second). If the timer were clocked at 10,000 Hz, and the load value was 2,500 (¼ of a second), it would count 20,480 pulses per update. Using the above equation:

RPM = (10000 × 20480 × 60) / (2500 × 2048 × 4) = 600 RPM

Now, consider that the motor is sped up to 3000 RPM. This results in 409,600 pulses per second, or 102,400 every ¼ of a second. Again, the above equation gives:

RPM = (10000 × 102400 × 60) / (2500 × 2048 × 4) = 3000 RPM

31.7.4 Velocity compare

In addition to velocity capture, the velocity measurement system includes a programmable velocity compare register. After every velocity capture event the contents of the velocity capture register (CAP) is compared with the contents of the velocity compare register (VELCOMP). If the captured velocity is less than the compare value an interrupt is asserted provided that the velocity compare interrupt enable bit is set. This can be used to determine if a motor shaft is either stalled or moving too slow.

32.1 How to read this chapter

The RIT is available on all LPC43xx parts.

32.2 Basic configuration

The RIT is configured as follows:

- See Table 744 for clocking and power control.
- The RIT is reset by the RITIMER_RST (reset #36).
- The RIT interrupt is connected to slot # 11 in the NVIC.

Table 744. RIT clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the RI timer register interface and RI timer peripheral clock (PCLK).	BASE_M4_CLK	CLK_M4_RITIMER	up to 204 MHz

32.3 Features

- 32-bit counter running from BASE_M4_CLK. Counter can be free-running, or be reset by a generated interrupt.
- 32-bit compare value.
- 32-bit compare mask. An interrupt is generated when the counter value equals the compare value after masking. This allows for combinations not possible with a simple compare.

32.4 General description

The Repetitive Interrupt Timer (RIT) provides a versatile means of generating interrupts at specified time intervals, without using a standard timer. It is intended for repeating interrupts that aren't related to Operating System interrupts. The RIT could also be used as an alternative to the Cortex-M4 System Tick Timer if there are different system requirements.

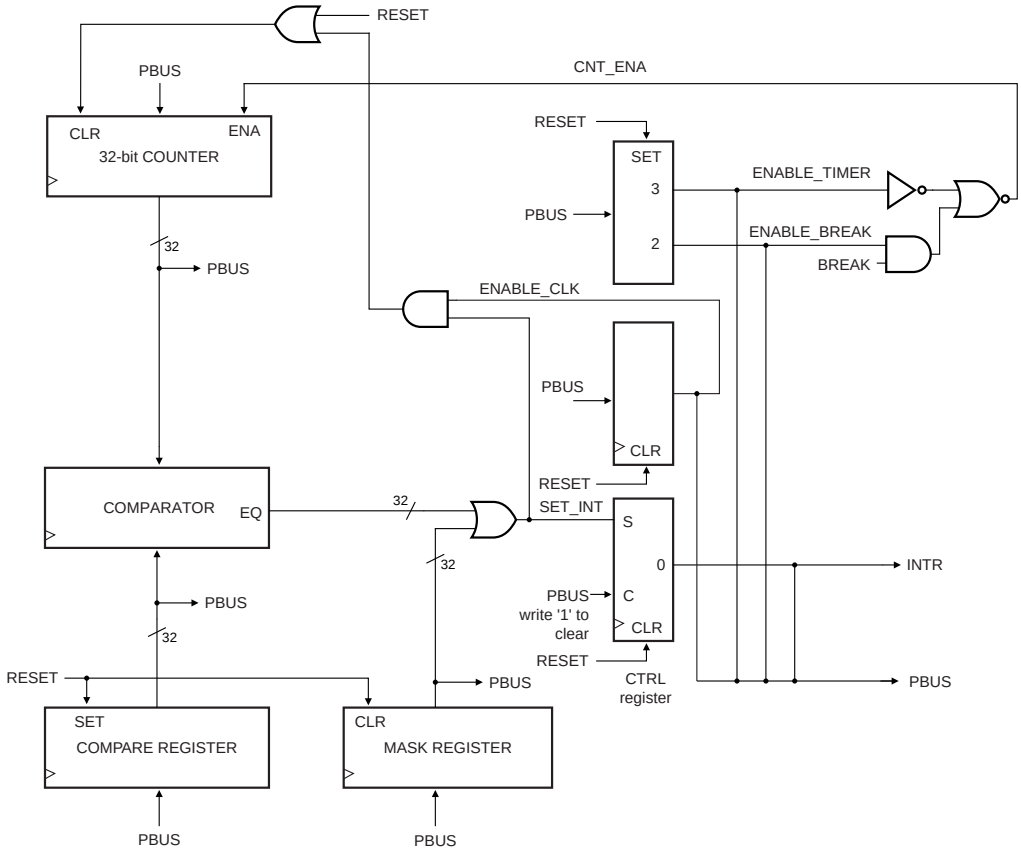


Fig 102. Repetitive Interrupt Timer (RIT) block diagram

32.5 Register description

Table 745. Register overview: Repetitive Interrupt Timer (RIT) (base address 0x400C 0000)

Name	Access	Address	Description	Reset value[1]	Reference
COMPVAL	R/W	0x000	Compare register	0xFFFF FFFF	Table 746
MASK	R/W	0x004	Mask register. This register holds the 32-bit mask value. A '1' written to any bit will force a compare on the corresponding bit of the counter and compare register.	0	Table 747
CTRL	R/W	0x008	Control register.	0xC	Table 748
COUNTER	R/W	0x00C	32-bit counter	0	Table 749

[1] Reset Value reflects the data stored in used bits only. It does not include content of reserved bits.

32.5.1 RI Compare Value register

Table 746. RI Compare Value register (COMPVAL - address 0x400C 0000) bit description

Bit	Symbol	Description	Reset value
31:0	RICOMP	Compare register. Holds the compare value which is compared to the counter.	0xFFFF FFFF

32.5.2 RI Mask register

Table 747. RI Mask register (MASK - address 0x400C 0004) bit description

Bit	Symbol	Description	Reset value
31:0	RIMASK	Mask register. This register holds the 32-bit mask value. A one written to any bit overrides the result of the comparison for the corresponding bit of the counter and compare register (causes the comparison of the register bits to be always true).	0

32.5.3 RI Control register

Table 748. RI Control register (CTRL - address 0x400C 0008) bit description

Bit	Symbol	Value	Description	Reset value
0	RITINT		Interrupt flag	0
		1	This bit is set to 1 by hardware whenever the counter value equals the masked compare value specified by the contents of RICOMPVAL and RIMASK registers. Writing a 1 to this bit will clear it to 0. Writing a 0 has no effect.	
		0	The counter value does not equal the masked compare value.	
1	RITENCLR		Timer enable clear	
		1	The timer will be cleared to 0 whenever the counter value equals the masked compare value specified by the contents of RICOMPVAL and RIMASK registers. This will occur on the same clock that sets the interrupt flag.	0
		0	The timer will not be cleared to 0.	
2	RITENBR		Timer enable for debug	1
		1	The timer is halted when the processor is halted for debugging.	
		0	Debug has no effect on the timer operation.	
3	RITEN		Timer enable.	1
		1	Timer enabled. **Remark:** This can be overruled by a debug halt if enabled in bit 2.	
		0	Timer disabled.	
31:4	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

32.5.4 RI Counter register

Table 749. RI Counter register (COUNTER - address 0x400C 000C) bit description

Bit	Symbol	Description	Reset value
31:0	RICOUNTER	32-bit up counter. Counts continuously unless RITEN bit in RICTRL register is cleared or debug mode is entered (if enabled by the RITNEBR bit in RICTRL). Can be loaded to any value in software.	0

32.6 RI timer operation

Following reset, the counter begins counting up from 0x00000000. Whenever the counter value equals the value programmed into the RICOMPVAL register the interrupt flag will be set. Any bit or combination of bits can be removed from this comparison (i.e. forced to compare) by writing a ‘1’ to the corresponding bit(s) in the RIMASK register. If the enable_clr bit is low (default state), a valid comparison ONLY causes the interrupt flag to be set. It has no effect on the count sequence. Counting continues as usual. When the counter reaches 0xFFFFFFFF it rolls-over to 0x00000000 on the next clock and continues counting. If the enable_clr bit is set to ‘1’ a valid comparison will also cause the counter to be reset to zero. Counting will resume from there on the next clock edge.

Counting can be halted in software by writing a ‘0’ to the Enable_Timer bit - RICTRL(2). Counting will also be halted when the processor is halted for debugging provided the Enable_Break bit – RICTRL(1) is set. Both the Enable_Timer and Enable_Break bits are set on reset.

The interrupt flag can be cleared in software by writing a ‘1’ to the Interrupt bit – RICTRL(0).

Software can load the counter to any value at any time by writing to RICOUNTER.

The counter (RICOUNTER), RICOMPVAL register, RIMASK register and RICTRL register can all be read by software at any time.

33.1 How to read this chapter

The Alarm timer is identical on all LPC43xx parts.

33.2 Basic configuration

The Alarm timer is configured as follows:

- See Table 750 for clocking and power control. The 32 kHz output of the 32 kHz oscillator must be enabled in the CREG0 register in the CREG block (see Table 38).
-
- The Alarm timer interrupt is connected to slot # 4 in the Event router.

Table 750. Alarm timer clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to alarm timer register interface	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz
32 kHz crystal oscillator output for the counter/timer clock	-	-	1024 Hz (fixed frequency)

33.3 General description

The alarm timer is a 16-bit timer and counts down from a preset value. The counter triggers a status bit when it reaches 0x00 and asserts an interrupt if enabled.

The alarm timer operates in the RTC power domain. It consists of a 16-bit counter (DOWNCOUNTER) running at a 1024 Hz clock. The 1024 Hz clock is derived from the 32 kHz crystal clock. The alarm timer is inactive when this clock is not active.

The alarm timer counts down from an initial value PRESET. When it reaches 0x0 and the interrupt is enabled (via SET_EN), bit STATUS is triggered. The counter continues counting down starting from PRESET.

STATUS is propagated to the interrupt output. The interrupt is connected to the Event router and can be used to wake up the device from a low power mode.

33.4 Register description

Table 751. Register overview: Alarm timer (base address 0x4004 0000)

Name	Access	Address offset	Description	Reset value	Reference
DOWNCOUNTER	R/W	0x000	Downcounter register	0x000	Table 752
PRESET	R/W	0x004	Preset value register	0x000	Table 753
-	-	0x008 - 0xFD4	Reserved	-	-
CLR_EN	W	0xFD8	Interrupt clear enable register	0x0	Table 754
SET_EN	W	0xFDC	Interrupt set enable register	0x0	Table 755
STATUS	R	0xFE0	Status register	0x0	Table 756
ENABLE	R	0xFE4	Enable register	0x0	Table 757
CLR_STAT	W	0xFE8	Clear register	0x0	Table 758
SET_STAT	W	0xFEC	Set register	0x0	Table 759

33.4.1 Downcounter register

Table 752. Downcounter register (DOWNCOUNTER - 0x4004 0000) bit description

Bit	Symbol	Description	Reset value
15:0	CVAL	When equal to zero an interrupt is raised. When equal to zero PRESET is loaded and counting continues.	0x0
31:16	-	Reserved.	-

33.4.2 Preset value register

Table 753. Preset value register (PRESET - 0x4004 0004) bit description

Bit	Symbol	Description	Reset value
15:0	PRESETVAL	Value loaded in DOWNCOUNTER when DOWNCOUNTER equals zero	
31:16	-	Reserved.	-

33.4.3 Interrupt clear enable register

Table 754. Interrupt clear enable register (CLR_EN - 0x4004 0FD8) bit description

Bit	Symbol	Description	Reset value
0	CLR_EN	Writing a 1 to this bit clears the interrupt enable bit in the ENABLE register.	-
31:1	-	Reserved.	-

33.4.4 Interrupt set enable register

Table 755. Interrupt set enable register (SET_EN - 0x4004 0FDC) bit description

Bit	Symbol	Description	Reset value
0	SET_EN	Writing a 1 to this bit sets the interrupt enable bit in the ENABLE register.	0
31:1	-	Reserved.	-

33.4.5 Interrupt status register

Table 756. Interrupt status register (STATUS - 0x4004 0FE0) bit description

Bit	Symbol	Description	Reset value
0	STAT	A 1 in this bit shows that the STATUS interrupt has been raised.	0
31:1	-	Reserved.	-

33.4.6 Interrupt enable register

Table 757. Interrupt enable register (ENABLE - 0x4004 0FE4) bit description

Bit	Symbol	Description	Reset value
0	EN	A 1 in this bit shows that the STATUS interrupt has been enabled and that the STATUS interrupt request signal is asserted when STAT = 1 in the STATUS register.	0
31:1	-	Reserved.	-

33.4.7 Clear status register

Table 758. Interrupt clear status register (CLR_STAT - 0x4004 0FE8) bit description

Bit	Symbol	Description	Reset value
0	CSTAT	Writing a 1 to this bit clears the STATUS interrupt bit in the STATUS register.	0
31:1	-	Reserved.	-

33.4.8 Set status register

Table 759. Interrupt set status register (SET_STAT - 0x4004 0FEC) bit description

Bit	Symbol	Description	Reset value
0	SSTAT	Writing a 1 to this bit sets the STATUS interrupt bit in the STATUS register.	0
31:1	-	Reserved.	-

34.1 How to read this chapter

The WWDT is identical for all LPC43xx parts.

34.2 Basic configuration

The WWDT is configured as follows:

- See Table 760 for clocking and power control. The only clock source for the WWDT clock (WDCLK) is the IRC.
- The WWDT cannot be reset by software.
- The WWDT interrupt is connected to slot # 7 in the Event router and slot #49 in the M4 NVIC.
- The WWDT registers can be accessed by the GPDMA as memory-to-memory transfer.

Table 760. WWDT clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to WWDT register interface (PCLK)	BASE_M4_CLK	CLK_M4_WWDT	up to 204 MHz
Watchdog clock (WDCLK)	BASE_SAFE_CLK	-	12 MHz (fixed frequency)

34.3 Features

- Internally resets chip if not reloaded during the programmable time-out period.
- Optional windowed operation requires reload to occur between a minimum and maximum time-out period, both programmable.
- Optional warning interrupt can be generated at a programmable time prior to watchdog time-out.
- Programmable 24 bit timer with internal fixed pre-scaler.
- Selectable time period from 1,024 watchdog clocks ($T_{WDCLK} \times 256 \times 4$) to over 67 million watchdog clocks ($T_{WDCLK} \times 2^{24} \times 4$) in increments of 4 watchdog clocks.
- Safe watchdog operation. Once enabled, requires a hardware reset or a Watchdog reset to be disabled.
- Incorrect feed sequence causes immediate watchdog reset if enabled.
- The watchdog reload value can optionally be protected such that it can only be changed after the “warning interrupt” time is reached.
- Flag to indicate Watchdog reset.
- The WWDT uses the IRC as a fixed clock source.

34.4 Applications

The purpose of the Watchdog Timer is to reset the microcontroller within a reasonable amount of time if it enters an erroneous state. When enabled, a watchdog event will be generated if the user program fails to feed (or reload) the Watchdog within a predetermined amount of time. The Watchdog event will cause a chip reset if configured to do so.

When a watchdog window is programmed, an early watchdog feed is also treated as a watchdog event. This allows preventing situations where a system failure may still feed the watchdog. For example, application code could be stuck in an interrupt service that contains a watchdog feed. Setting the window such that this would result in an early feed will generate a watchdog event, allowing for system recovery.

34.5 Description

The Watchdog consists of a fixed divide-by-4 pre-scaler and a 24-bit counter which decrements on every clock cycle. The minimum value from which the counter decrements is 0xFF. Setting a value lower than 0xFF causes 0xFF to be loaded in the counter. Hence the minimum Watchdog interval is ($T_{WDCLK} \times 256 \times 4$) and the maximum Watchdog interval is ($T_{WDCLK} \times 2^{24} \times 4$) in multiples of ($T_{WDCLK} \times 4$). The Watchdog should be used in the following manner:

- Set the Watchdog time-out value in TC register.
- Setup the Watchdog timer operating mode in MOD register.
- Set a value for the watchdog window time in WINDOW register if windowed operation is required.
- Set a compare value for the watchdog warning interrupt in the WARNINT register if a warning interrupt is required.
- Enable the Watchdog by writing 0xAA followed by 0x55 to the FEED register.
- The Watchdog must be fed again before the Watchdog counter reaches zero in order to prevent a watchdog event. If a window value is programmed, the feed must also occur after the watchdog counter passes that value.

When the Watchdog Timer is configured so that a watchdog event will cause a reset and the counter reaches zero, the CPU will be reset, loading the stack pointer and program counter from the vector table as in the case of external reset. The Watchdog time-out flag (WDTOF) can be examined to determine if the Watchdog has caused the reset condition. The WDTOF flag must be cleared by software.

When the Watchdog Timer is configured to generate a warning interrupt, the interrupt will occur when the counter matches the compare value defined by the WARNINT register.

34.5.1 WWDT behavior in Debug mode

If code execution is halted in Debug mode, the WWDT stops counting until code execution resumes.

34.5.2 WWDT behavior in the power-down modes

The WWDT is running in Sleep mode. A watchdog triggered reset in Sleep mode resets and wakes up the chip. Likewise, a watchdog triggered interrupt wakes up the chip from Sleep mode if the interrupt is enabled in the NVIC.

The WWDT is not operating in Deep-sleep, Power-down, and Deep power-down modes.

34.6 Clocking

The watchdog timer block uses two clocks: PCLK and WDCLK. PCLK is used for the APB accesses to the watchdog registers and is derived from the BASE_M4_CLK. The WDCLK is used for the watchdog timer counting and is derived from the IRC. The clock source (the IRC) is fixed to ensure that the WDT always has a valid clock.

There is some synchronization logic between these two clock domains. When the MOD and TC registers are updated by APB operations, the new value will take effect in three WDCLK cycles on the logic in the WDCLK clock domain. When the watchdog timer is counting the WDCLK clock cycles, the synchronization logic will first lock the value of the counter on WDCLK and then synchronize it with the PCLK for reading when the TV register by the CPU.

34.7 Register description

The Watchdog registers are shown in Table 761.

Table 761. Register overview: Watchdog timer (base address 0x4008 0000)

Name	Access	Address offset	Description	Reset value[1]	Reference
MOD	R/W	0x000	Watchdog mode register. This register contains the basic mode and status of the Watchdog Timer.	0	Table 762
TC	R/W	0x004	Watchdog timer constant register. This register determines the time-out value.	0xFF	Table 764
FEED	WO	0x008	Watchdog feed sequence register. Writing 0xAA followed by 0x55 to this register reloads the Watchdog timer with the value contained in WDTC.	NA	Table 765
TV	RO	0x00C	Watchdog timer value register. This register reads out the current value of the Watchdog timer.	0xFF	Table 766
-	-	0x010	Reserved	-	-
WARNINT	R/W	0x014	Watchdog warning interrupt register. This register contains the Watchdog warning interrupt compare value.	0	Table 767
WINDOW	R/W	0x018	Watchdog timer window register. This register contains the Watchdog window value.	0xFF FFFF	Table 768

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

34.7.1 Watchdog mode register

The WDMOD register controls the operation of the Watchdog as per the combination of WDEN and RESET bits. Note that a watchdog feed must be performed before any changes to the WDMOD register take effect.

Table 762. Watchdog Mode register (MOD - 0x4008 0000) bit description

Bit	Symbol	Value	Description	Reset value
0	WDEN		Watchdog enable bit. This bit is Set Only.	0
		0	The watchdog timer is stopped.	
		1	The watchdog timer is running.	
1	WDRESET		Watchdog reset enable bit. This bit is Set Only.	0
		0	A watchdog time-out will not cause a chip reset.	
		1	A watchdog time-out will cause a chip reset.	
2	WDTOF		Watchdog time-out flag. Set when the watchdog timer times out, by a feed error, or by events associated with WDPROTECT, cleared by software. Causes a chip reset if WDRESET = 1. This flag is cleared by software writing a 0 to this bit.	0 (Only after external reset)
3	WDINT		Watchdog interrupt flag. Set when the timer reaches the value in the WARNINT register. Cleared by software by writing a 1 to this bit.	0
4	WDPROTECT		Watchdog update mode. This bit is Set Only.	0
		0	The watchdog time-out value (WDTC) can be changed at any time.	
		1	The watchdog time-out value (WDTC) can be changed only after the counter is below the value of WDWARNINT and WDWINDOW.	
7:5	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Once the **WDEN**, **WDPROTECT**, or **WDRESET** bits are set they can not be cleared by software. Both flags are cleared by an external reset or a Watchdog timer reset.

WDTOF The Watchdog time-out flag is set when the Watchdog times out, when a feed error occurs, or when WDPROTECT =1 and an attempt is made to write to the TC register. This flag is cleared by software writing a 0 to this bit.

WDINT The Watchdog interrupt flag is set when the Watchdog counter reaches the value specified by WDWARNINT. This flag is cleared when any reset occurs, and is cleared by software by writing a 1 to this bit.

Watchdog reset or interrupt will occur any time the watchdog is running and has an operating clock source. Any clock source works in Sleep mode, and the IRC works in Deep-sleep mode. If a watchdog interrupt occurs in Sleep or Deep-sleep mode, it will wake up the device.

Table 763. Watchdog operating modes selection

WDEN	WDRESET	Mode of Operation
0	X (0 or 1)	Debug/Operate without the Watchdog running.
1	0	Watchdog interrupt mode: the watchdog warning interrupt will be generated but watchdog reset will not. When this mode is selected, the watchdog counter reaching the value specified by WDWARNINT will set the WDINT flag and the Watchdog interrupt request will be generated.
1	1	Watchdog reset mode: Both the watchdog interrupt and watchdog reset are enabled. When this mode is selected, the watchdog counter reaching the value specified by WDWARNINT will set the WDINT flag and the Watchdog interrupt request will be generated. The watchdog counter reaching zero will reset the microcontroller. **Remark:** Other causes for a watchdog reset are: A watchdog feed or changing the WDTC value (if the WDPROTECT bit is set in the MOD register) before reaching the value of WDWINDOW.

34.7.2 Watchdog timer constant register

The TC register determines the time-out value. Every time a feed sequence occurs, the TC register content is reloaded into the Watchdog timer. This is pre-loaded with the value 0x00 00FF upon reset. Writing values below 0xFF will cause 0x00 00FF to be loaded into the TC register. Thus the minimum time-out interval is $T_{WDCLK} \times 256 \times 4$.

If the WDPROTECT bit in MOD register is set to one, an attempt to change the value of TC before the watchdog counter is below the values of WDWARNINT and WDWINDOW will cause a watchdog reset and set the WDTOF flag.

Table 764. Watchdog Timer Constant register (TC - 0x4008 0004) bit description

Bit	Symbol	Description	Reset value
23:0	WDTC	Watchdog time-out value.	0x00 00FF
31:24	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

34.7.3 Watchdog feed register

Writing 0xAA followed by 0x55 to this register will reload the Watchdog timer with the time-out value in the TC register. This operation will also start the Watchdog if it is enabled via the MOD register. Setting the WDEN bit in the WDMOD register is not sufficient to enable the Watchdog. A valid feed sequence must be completed after setting WDEN before the Watchdog is capable of generating a reset. Until then, the Watchdog will ignore feed errors. After writing 0xAA to FEED register, access to any Watchdog register other than writing 0x55 to FEED register causes an immediate reset/interrupt when the Watchdog is enabled, and sets the WDTOF flag. The reset will be generated during the second PCLK following an incorrect access to a Watchdog register during a feed sequence.

Interrupts should be disabled during the feed sequence. An abort condition will occur if an interrupt happens during the feed sequence.

Table 765. Watchdog Feed register (FEED - 0x4008 0008) bit description

Bit	Symbol	Description	Reset value
7:0	Feed	Feed value should be 0xAA followed by 0x55.	NA

34.7.4 Watchdog timer value register

The WDTV register is used to read the current value of Watchdog timer counter.

When reading the value of the 24 bit counter, the lock and synchronization procedure takes up to 6 WDCLK cycles plus 6 PCLK cycles, so the value of WDTV is older than the actual value of the timer when it's being read by the CPU.

Table 766. Watchdog Timer Value register (TV - 0x4008 000C) bit description

Bit	Symbol	Description	Reset value
23:0	Count	Counter timer value.	0x00 00FF
31:24	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

34.7.5 Watchdog timer warning interrupt register

The WDWARNINT register determines the watchdog timer counter value that will generate a watchdog interrupt. When the watchdog timer counter matches the value defined by WDWARNINT, an interrupt will be generated after the subsequent WDCLK.

A match of the watchdog timer counter to WDWARNINT occurs when the bottom 10 bits of the counter have the same value as the 10 bits of WARNINT, and the remaining upper bits of the counter are all 0. This gives a maximum time of 1,023 watchdog timer counts (4,096 watchdog clocks) for the interrupt to occur prior to a watchdog event. If WDWARNINT is set to 0, the interrupt will occur at the same time as the watchdog event.

Table 767. Watchdog Timer Warning Interrupt register (WARNINT - 0x4008 0014) bit description

Bit	Symbol	Description	Reset value
9:0	WDWARNINT	Watchdog warning interrupt compare value.	0
31:10	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

34.7.6 Watchdog timer window register

The WDWINDOW register determines the highest WDTV value allowed when a watchdog feed is performed. If a feed valid sequence completes prior to WDTV reaching the value in WDWINDOW, a watchdog event will occur.

WDWINDOW resets to the maximum possible WDTV value, so windowing is not in effect. Values of WDWINDOW below 0x100 will make it impossible to ever feed the watchdog successfully.

Table 768. Watchdog Timer Window register (WINDOW - 0x4008 0018) bit description

Bit	Symbol	Description	Reset value
23:0	WDWINDOW	Watchdog window value.	0xFF FFFF
31:24	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

34.8 Block diagram

The block diagram of the Watchdog is shown below in the Figure 103. The synchronization logic (PCLK - WDCLK) is not shown in the block diagram.

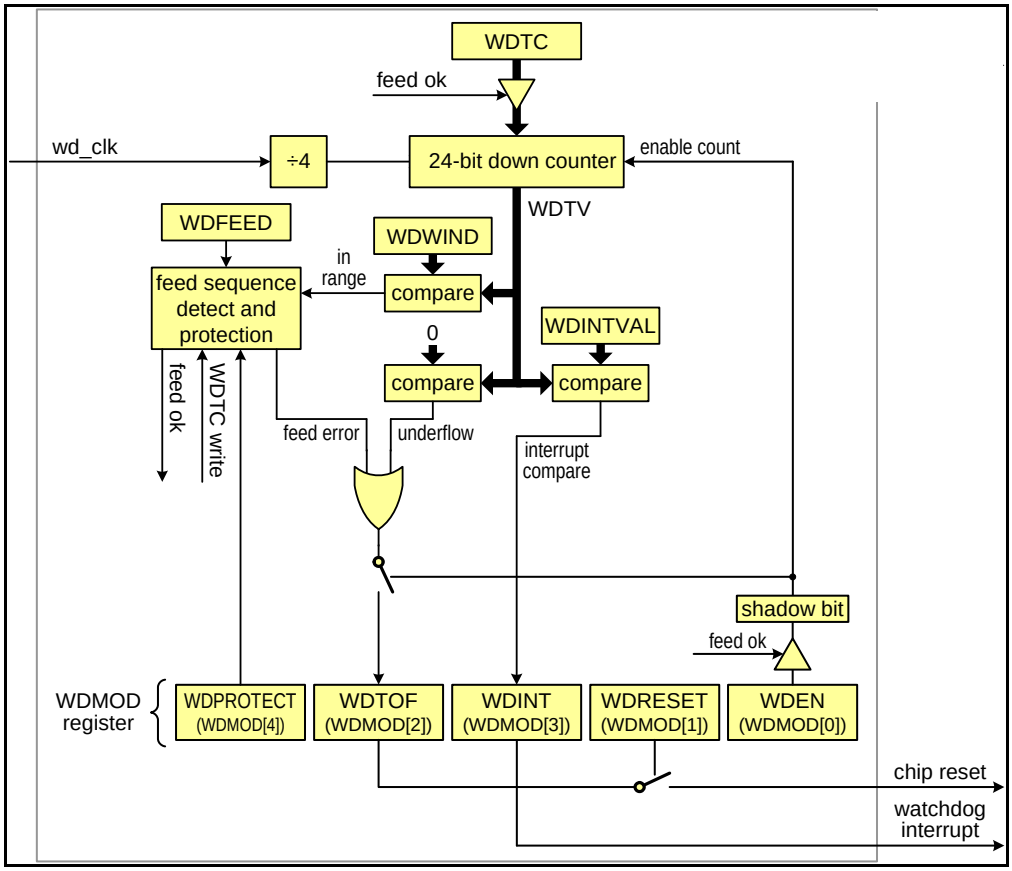


Fig 103. Watchdog block diagram

34.9 Watchdog timing examples

The following figures illustrate several aspects of Watchdog Timer operation is shown below in Figure 104.

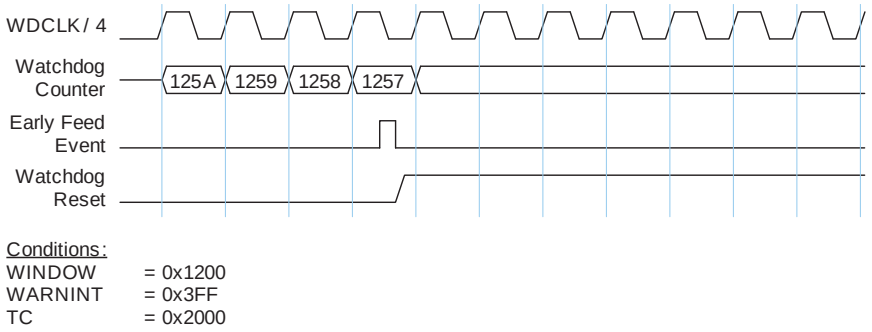


Fig 104. Early Watchdog Feed with Windowed Mode Enabled

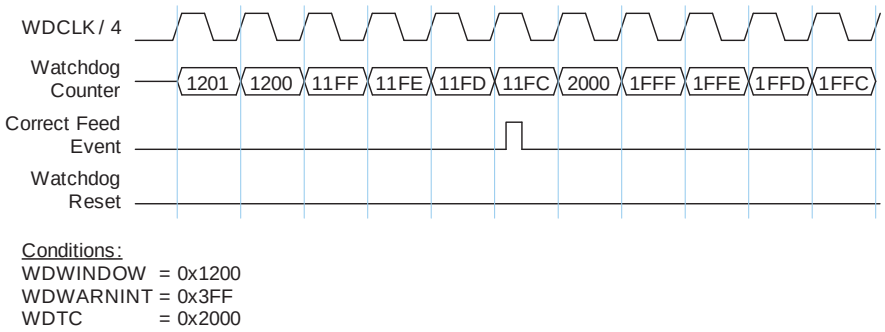


Fig 105. Correct Watchdog Feed with Windowed Mode Enabled

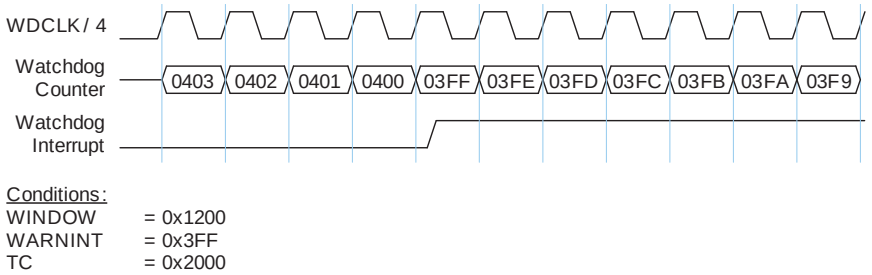


Fig 106. Watchdog Warning Interrupt

35.1 How to read this chapter

The RTC is available on all LPC43xx parts.

35.2 Basic configuration

The RTC is configured as follows:

- See Table 769 for clocking and power control. The 1 kHz output of the 32 kHz oscillator must be enabled in the CREG0 register in the CREG block (see Table 38).
- The RTC interrupt is connected to slot # 5 in the Event router.
- The functionality of the RTC_ALARM pin is controlled by the CREG0 register (see Table 38). By default, the RTC Alarm interrupt can be monitored on this pin.
-

Remark: After initializing the 32 kHz oscillator, wait for 2 sec before writing to the RTC registers.

Table 769. RTC clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the RTC register interface	BASE_M4_CLK	CLK_M4_BUS	up to 204 MHz
32 kHz crystal oscillator output for the RTC counter/timer clock	-	-	1024 Hz (fixed frequency); the RTC receives an internal 1 Hz clock.

35.3 Features

- Measures the passage of time to maintain a calendar and clock. Provides seconds, minutes, hours, day of month, month, year, day of week, and day of year.
- Ultra-low power design to support battery powered systems. Uses power from the CPU power supply when it is present.
- Dedicated battery power supply pin.
- RTC power supply is isolated from the rest of the chip.
- Calibration counter allows adjustment to better than ±1 sec/day with 1 sec resolution.
- Periodic interrupts can be generated from increments of any field of the time registers and selected fractional second values.
- Alarm interrupt can be generated for a specific date/time.

35.4 General description

The Real Time Clock (RTC) is a set of counters for measuring time when system power is on, and optionally when it is off. It uses very little power when its registers are not being accessed by the CPU, especially in reduced power modes. On the LPC43xx, the RTC is clocked by a separate 32 kHz oscillator that produces a 1 Hz internal time reference. The RTC is powered by its own power supply pin, VBAT.

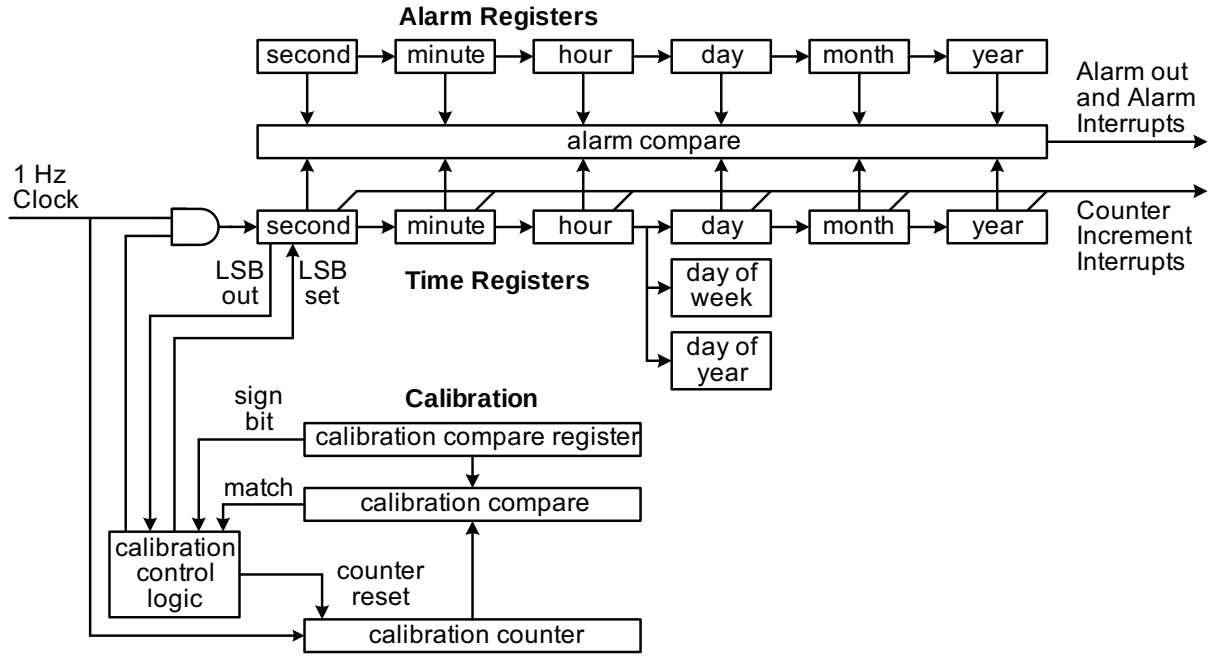


Fig 107. RTC functional block diagram

35.5 Pin description

Table 770. RTC pin description

Function name	Direction	Description
RTC_ALARM	O	RTC controlled output. This is a 1.8 V pin. It goes HIGH when an RTC alarm is generated. CREG0 bits ALARMCTRL must be set to 0.(see Table 38). **Remark:** This pin must be LOW for JTAG programming.

35.6 Register description

Table 771. Register overview: RTC (base address 0x4004 6000)

Name	Access	Address offset	Description	Reset value	Reference
ILR	W	0x000	Interrupt Location Register	0x0	Table 773
-	-	0x004	Reserved	0x00	-
CCR	R/W	0x008	Clock Control Register	0x00	Table 774
CIIR	R/W	0x00C	Counter Increment Interrupt Register	0x00	Table 775
AMR	R/W	0x010	Alarm Mask Register	-[1]	Table 776
CTIME0	R	0x014	Consolidated Time Register 0	-[1]	Table 777
CTIME1	R	0x018	Consolidated Time Register 1	-[1]	Table 778
CTIME2	R	0x01C	Consolidated Time Register 2	-[1]	Table 779
SEC	R/W	0x020	Seconds Register	-[1]	Table 782
MIN	R/W	0x024	Minutes Register	-[1]	Table 783
HRS	R/W	0x028	Hours Register	-[1]	Table 784
DOM	R/W	0x02C	Day of Month Register	-[1]	Table 785
DOW	R/W	0x030	Day of Week Register	-[1]	Table 786
DOY	R/W	0x034	Day of Year Register	-[1]	Table 787
MONTH	R/W	0x038	Months Register	-[1]	Table 788
YEAR	R/W	0x03C	Years Register	-[1]	Table 789
CALIBRATION	R/W	0x040	Calibration Value Register	-[1]	Table 790
-	-	0x044 - 0x05C		-	-
ASEC	R/W	0x060	Alarm Seconds Register	-[1]	Table 792
AMIN	R/W	0x064	Alarm Minutes Register	-[1]	Table 793
AHRS	R/W	0x068	Alarm Hours Register	-[1]	Table 794
ADOM	R/W	0x6C	Alarm Day of Month Register	-[1]	Table 795
ADOW	R/W	0x070	Alarm Day of Week Register	-[1]	Table 796
ADOY	R/W	0x074	Alarm Day of Year Register	-[1]	Table 797
AMON	R/W	0x078	Alarm Month Register	-[1]	Table 798
AYRS	R/W	0x07C	Alarm Year Register	-[1]	Table 799

[1] This register value is not changed by reset.

In addition to the RTC registers, 64 general purpose registers are available to store data when the main power supply is switched off. The general purpose registers reside in the RTC power domain and can be battery powered.

Table 772. Register overview: REGFILE (base address 0x4004 1000)

Name	Access	Address offset	Description	Reset Value
REGFILE0	R/W	0x000	General purpose storage register	0x0
to				
REGFILE63	R/W	0x0FC	General purpose storage register	0x0

35.6.1 Interrupt Location Register

The Interrupt Location Register is a 2-bit register that specifies which blocks are generating an interrupt (see Table 773). Writing a one to the appropriate bit clears the corresponding interrupt. Writing a zero has no effect. This allows the programmer to read this register and write back the same value to clear only the interrupt that is detected by the read.

Table 773. Interrupt Location Register (ILR - address 0x4004 6000) bit description

Bit	Symbol	Description	Reset value
0	RTCCIF	When one, the Counter Increment Interrupt block generated an interrupt. Writing a one to this bit location clears the counter increment interrupt.	0
1	RTCALF	When one, the alarm registers generated an interrupt. Writing a one to this bit location clears the alarm interrupt.	0
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

35.6.2 Clock Control Register

The clock register is a 4-bit register that controls the operation of the clock divide circuit. Each bit of the clock register is described in Table 774. Bits 0, 1, and 4 in this register should be initialized when the RTC is first turned on.

Table 774. Clock Control Register (CCR - address 0x4004 6008) bit description

Bit	Symbol	Value	Description	Reset value
0	CLKEN		Clock Enable.	-[1]
		0	The time counters are disabled so that they may be initialized.	
		1	The time counters are enabled.	
1	CTCRST		CTC Reset.	0
		0	No effect.	
		1	When one, the elements in the internal oscillator divider are reset, and remain reset until CCR[1] is changed to zero. This is the divider that generates the 1 Hz clock from the 32.768 kHz crystal. The state of the divider is not visible to software.	
3:2	-		Internal test mode controls. These bits must be 0 for normal RTC operation.	-[1]
4	CCALEN		Calibration counter enable.	-[1]
		0	The calibration counter is enabled and counting, using the 1 Hz clock. When the calibration counter is equal to the value of the CALIBRATION register, the counter resets and repeats counting up to the value of the CALIBRATION register. See Section 35.6.6.2 and Section 35.7.1.	
		1	The calibration counter is disabled and reset to zero.	
31:5	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

[1] This register value is not changed by reset.

35.6.3 Counter Increment Interrupt Register

The Counter Increment Interrupt Register (CIIR) gives the ability to generate an interrupt every time a counter is incremented. This interrupt remains valid until cleared by writing a 1 to bit 0 of the Interrupt Location Register (ILR[0]).

Table 775. Counter Increment Interrupt Register (CIIR - address 0x4004 600C) bit description

Bit	Symbol	Description	Reset value
0	IMSEC	When 1, an increment of the Second value generates an interrupt.	0
1	IMMIN	When 1, an increment of the Minute value generates an interrupt.	0
2	IMHOUR	When 1, an increment of the Hour value generates an interrupt.	0
3	IMDOM	When 1, an increment of the Day of Month value generates an interrupt.	0
4	IMDOW	When 1, an increment of the Day of Week value generates an interrupt.	0
5	IMDOY	When 1, an increment of the Day of Year value generates an interrupt.	0
6	IMMON	When 1, an increment of the Month value generates an interrupt.	0
7	IMYEAR	When 1, an increment of the Year value generates an interrupt.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

35.6.4 Alarm Mask Register

The Alarm Mask Register (AMR) allows the user to mask any of the alarm registers. Table 776 shows the relationship between the bits in the AMR and the alarms. For the alarm function, every non-masked alarm register must match the corresponding time counter for an interrupt to be generated. The interrupt is generated only when the counter comparison first changes from no match to match. The interrupt is removed when a one is written to the appropriate bit of the Interrupt Location Register (ILR). If all mask bits are set, then the alarm is disabled.

Table 776. Alarm Mask Register (AMR - address 0x4004 6010) bit description

Bit	Symbol	Description	Reset value
0	AMRSEC	When 1, the Second value is not compared for the alarm.	0
1	AMRMIN	When 1, the Minutes value is not compared for the alarm.	0
2	AMRHOUR	When 1, the Hour value is not compared for the alarm.	0
3	AMRDOM	When 1, the Day of Month value is not compared for the alarm.	0
4	AMRDOW	When 1, the Day of Week value is not compared for the alarm.	0
5	AMRDOY	When 1, the Day of Year value is not compared for the alarm.	0
6	AMRMON	When 1, the Month value is not compared for the alarm.	0
7	AMRYEAR	When 1, the Year value is not compared for the alarm.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

35.6.5 Consolidated time registers

The values of the Time Counters can optionally be read in a consolidated format which allows the programmer to read all time counters with only three read operations. The various registers are packed into 32-bit values as shown in Table 777, Table 778, and Table 779. The least significant bit of each register is read back at bit 0, 8, 16, or 24.

The Consolidated Time Registers are read-only. To write new values to the Time Counters, the Time Counter addresses should be used.

35.6.5.1 Consolidated Time Register 0

The Consolidated Time Register 0 contains the low order time values: Seconds, Minutes, Hours, and Day of Week.

Table 777. Consolidated Time register 0 (CTIME0 - address 0x4004 6014) bit description

Bit	Symbol	Description	Reset value
5:0	SECONDS	Seconds value in the range of 0 to 59	-[1]
7:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
13:8	MINUTES	Minutes value in the range of 0 to 59	-[1]
15:14	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
20:16	HOURS	Hours value in the range of 0 to 23	-[1]
23:21	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-[1]
26:24	DOW	Day of week value in the range of 0 to 6	-[1]
31:27	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

35.6.5.2 Consolidated Time Register 1

The Consolidate Time Register 1 contains the Day of Month, Month, and Year values.

Table 778. Consolidated Time register 1 (CTIME1 - address 0x4004 6018) bit description

Bit	Symbol	Description	Reset value
4:0	DOM	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year).	-[1]
7:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
11:8	MONTH	Month value in the range of 1 to 12.	-[1]
15:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
27:16	YEAR	Year value in the range of 0 to 4095.	-[1]
31:28	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

35.6.5.3 Consolidated Time Register 2

The Consolidate Time Register 2 contains just the Day of Year value.

Table 779. Consolidated Time register 2 (CTIME2 - address 0x4004 601C) bit description

Bit	Symbol	Description	Reset value
11:0	DOY	Day of year value in the range of 1 to 365 (366 for leap years).	-[1]
31:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

35.6.6 Time Counter Group

The time value consists of the eight counters shown in Table 780 and Table 781. These counters can be read or written at the locations shown in Table 781.

Table 780. Time Counter relationships and values

Counter	Size	Enabled by	Minimum value	Maximum value
Second	6	1 Hz Clock	0	59
Minute	6	Second	0	59
Hour	5	Minute	0	23
Day of Month	5	Hour	1	28, 29, 30 or 31
Day of Week	3	Hour	0	6
Day of Year	9	Hour	1	365 or 366 (for leap year)
Month	4	Day of Month	1	12
Year	12	Month or day of Year	0	4095

Table 781. Time Counter registers

Name	Size	Description	Access	Address
SEC	6	Seconds value in the range of 0 to 59	R/W	0x4004 6020
MIN	6	Minutes value in the range of 0 to 59	R/W	0x4004 6024
HRS	5	Hours value in the range of 0 to 23	R/W	0x4004 6028
DOM	5	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year).[1]	R/W	0x4004 602C
DOW	3	Day of week value in the range of 0 to 6[1]	R/W	0x4004 6030
DOY	9	Day of year value in the range of 1 to 365 (366 for leap years)[1]	R/W	0x4004 6034
MONTH	4	Month value in the range of 1 to 12	R/W	0x4004 6038
YEAR	12	Year value in the range of 0 to 4095	R/W	0x4004 603C

[1] These values are simply incremented at the appropriate intervals and reset at the defined overflow point. They are not calculated and must be correctly initialized in order to be meaningful.

Table 782. Seconds register (SEC - address 0x4004 6020) bit description

Bit	Symbol	Description	Reset value
5:0	SECONDS	Seconds value in the range of 0 to 59	-[1]
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 783. Minutes register (MIN - address 0x4004 6024) bit description

Bit	Symbol	Description	Reset value
5:0	MINUTES	Minutes value in the range of 0 to 59	-[1]
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 784. Hours register (HRS - address 0x4004 6028) bit description

Bit	Symbol	Description	Reset value
4:0	HOURS	Hours value in the range of 0 to 23	-[1]
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 785. Day of month register (DOM - address 0x4004 602C) bit description

Bit	Symbol	Description	Reset value
4:0	DOM	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year).	-[1]
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 786. Day of week register (DOW - address 0x4004 6030) bit description

Bit	Symbol	Description	Reset value
2:0	DOW	Day of week value in the range of 0 to 6.	-[1]
31:3	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 787. Day of year register (DOY - address 0x4004 6034) bit description

Bit	Symbol	Description	Reset value
8:0	DOY	Day of year value in the range of 1 to 365 (366 for leap years).	-[1]
31:9	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 788. Month register (MONTH - address 0x4004 6038) bit description

Bit	Symbol	Description	Reset value
3:0	MONTH	Month value in the range of 1 to 12.	-[1]
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 789. Year register (YEAR - address 0x4004 603C) bit description

Bit	Symbol	Description	Reset value
11:0	YEAR	Year value in the range of 0 to 4095.	-[1]
31:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

35.6.6.1 Leap year calculation

The RTC does a simple bit comparison to see if the two lowest order bits of the year counter are zero. If true, then the RTC considers that year a leap year. The RTC considers all years evenly divisible by 4 as leap years. This algorithm is accurate from the year 1901 through the year 2099, but fails for the year 2100, which is not a leap year. The only effect of leap year on the RTC is to alter the length of the month of February for the month, day of month, and year counters.

35.6.6.2 Calibration register

The following register is used to calibrate the time counter. The bits in this register are not changed by reset.

Table 790. Calibration register (CALIBRATION - address 0x4004 6040) bit description

Bit	Symbol	Value	Description	Reset value
16:0	CALVAL	-	If enabled, the calibration counter counts up to this value. The maximum value is 131072 corresponding to about 36.4 hours. Calibration is disabled if CALVAL = 0.	-[1]

Table 790. Calibration register (CALIBRATION - address 0x4004 6040) bit description

Bit	Symbol	Value	Description	Reset value
17	CALDIR		Calibration direction	-[1]
		0	Forward calibration. When CALVAL is equal to the calibration counter, the RTC timers will jump by 2 seconds.	
		1	Backward calibration. When CALVAL is equal to the calibration counter, the RTC timers will stop incrementing for 1 second.	
31:18	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

35.6.7 Alarm register group

The alarm registers are shown in Table 791. The values in these registers are compared with the time counters. If all the unmasked (See Section 35.6.4 “Alarm Mask Register” on page 858) alarm registers match their corresponding time counters then an interrupt is generated. The interrupt is cleared when a 1 is written to bit 1 of the Interrupt Location Register (ILR[1]).

Table 791. Alarm registers

Name	Size	Description	Access	Address
ALSEC	6	Alarm value for Seconds	R/W	0x4004 6060
ALMIN	6	Alarm value for Minutes	R/W	0x4004 6064
ALHRS	5	Alarm value for Hours	R/W	0x4004 6068
ALDOM	5	Alarm value for Day of Month	R/W	0x4004 606C
ALDOW	3	Alarm value for Day of Week	R/W	0x4004 6070
ALDOY	9	Alarm value for Day of Year	R/W	0x4004 6074
ALMON	4	Alarm value for Months	R/W	0x4004 6078
ALYEAR	12	Alarm value for Years	R/W	0x4004 607C

Table 792. Alarm Seconds register (ASEC - address 0x4004 6060) bit description

Bit	Symbol	Description	Reset value
5:0	SECONDS	Seconds value in the range of 0 to 59	-[1]
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 793. Alarm Minutes register (AMIN - address 0x4004 6064) bit description

Bit	Symbol	Description	Reset value
5:0	MINUTES	Minutes value in the range of 0 to 59	-[1]
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 794. Alarm Hours register (AHRS - address 0x4004 6068) bit description

Bit	Symbol	Description	Reset value
4:0	HOURS	Hours value in the range of 0 to 23	-[1]
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 795. Alarm Day of month register (ADOM - address 0x4004 606C) bit description

Bit	Symbol	Description	Reset value
4:0	DOM	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year).	-[1]
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 796. Alarm Day of week register (ADOW - address 0x4004 6070) bit description

Bit	Symbol	Description	Reset value
2:0	DOW	Day of week value in the range of 0 to 6.	-[1]
31:3	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 797. Alarm Day of year register (ADOY - address 0x4004 6074) bit description

Bit	Symbol	Description	Reset value
8:0	DOY	Day of year value in the range of 1 to 365 (366 for leap years).	-[1]
31:9	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 798. Alarm Month register (AMON - address 0x4004 6078) bit description

Bit	Symbol	Description	Reset value
3:0	MONTH	Month value in the range of 1 to 12.	-[1]
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

Table 799. Alarm Year register (AYRS - address 0x4004 607C) bit description

Bit	Symbol	Description	Reset value
11:0	YEAR	Year value in the range of 0 to 4095.	-[1]
31:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

[1] This register value is not changed by reset.

35.7 Functional description

35.7.1 Calibration procedure

The calibration logic can periodically adjust the time counter either by not incrementing the counter, or by incrementing the counter by 2 instead of 1. This allows calibrating the RTC oscillator under some typical voltage and temperature conditions without the need to externally trim the RTC oscillator.

A recommended method for determining the calibration value is to use the CLKOUT feature to unintrusively observe the RTC oscillator frequency under the conditions it is to be trimmed for, and calculating the number of clocks that will be seen before the time is off by one second. That value is used to determine CALVAL.

The exact method of calibration depends on whether CALVAL is even or odd. For even values, the hardware performs a two calibrations sequentially multiple times (one calibration at CALVAL+1 and one calibration at CALVAL - 1) and averages both calibration values. For odd values of CALVAL, the calibration time is accurate.

If the RTC oscillator is trimmed externally, the same method of unintrusively observing the RTC oscillator frequency may be helpful in that process.

Backward calibration

Enable the RTC timer and calibration in the CCR register (set bits CLKEN = 1 and CCALEN = 0). In the CALIBRATION register, set the calibration value CALVAL $\geq$ 1 and select CALDIR = 1.

- The SEC timer and the calibration counter count up for every 1 Hz clock cycle.
- When the calibration counter reaches CALVAL, a calibration match occurs and all RTC timers will be stopped for one clock cycle so that the timers will not increment in the next cycle.
- If an alarm match event occurs in the same cycle as the calibration match, the alarm interrupt will be delayed by one cycle to avoid a double alarm interrupt.

Forward calibration

Enable the RTC timer and calibration in the CCR register (set bits CLKEN = 1 and CCALEN = 0). In the CALIBRATION register, set the calibration value CALVAL $\geq$ 1 and select CALDIR = 0.

- The SEC timer and the calibration counter count up for every 1 Hz clock cycle.
- When the calibration counter reaches CALVAL, a calibration match occurs and the RTC timers are incremented by 2.

- When the calibration event occurs, the LSB of the ALSEC register is forced to be one so that the alarm interrupt will not be missed when skipping a second.

36.1 How to read this chapter

The USART0/2/3 controllers are available on all LPC43xx parts.

36.2 Basic configuration

The USART0/2/3 are configured as follows:

- See Table 800 for clocking and power control.
- The USART0/2/3 are reset by the UART0/2/3_RST (reset #44/46/47).
- The USART0/2/3 interrupts are connected to slots # 24/26/27 in the NVIC.
- For connecting the USART0/2/3 receive and transmit lines to the GPDMA, use the DMAMUX register in the CREG block (see Table 41) and enable the GPDMA channel in the DMA Channel Configuration registers (Section 19.6.20).
- The internal USART transmit/receive active signals (USARTx TX/RX) are connected to the GIMA (see Table 138). These signals are driven when data are transmitted or received and can be used to trigger any of the timers or the SCT.

Table 800. USART0/2/3 clocking and power control

	Base clock	Branch clock	Operating frequency
USART0 clock to register interface	BASE_M4_CLK	CLK_M4_UART0	up to 204 MHz
USART0 peripheral clock (PCLK)	BASE_UART0_CLK	CLK_APB0_UART0	up to 204 MHz
USART2 clock to register interface	BASE_M4_CLK	CLK_M4_UART2	up to 204 MHz
USART2 peripheral clock (PCLK)	BASE_UART2_CLK	CLK_APB2_UART2	up to 204 MHz
USART3 clock to register interface	BASE_M4_CLK	CLK_M4_UART3	up to 204 MHz
USART3 peripheral clock (PCLK)	BASE_UART3_CLK	CLK_APB2_UART3	up to 204 MHz

36.3 Features

- 16-byte receive and transmit FIFOs.
- Register locations conform to ‘550 industry standard.
- Receiver FIFO trigger points at 1, 4, 8, and 14 bytes.
- Built-in baud rate generator.
- USART in UART mode allows for implementation of either software or hardware flow control.
- RS-485/EIA-485 9-bit mode support with output enable.
- Support for synchronous mode UART (USART).
- IrDA interface (USART3 only).
- DMA support.
- Smart Card interface.

36.4 General description

The architecture of the USART is shown below in the block diagram.

The APB interface provides a communications link between the CPU or host and the USART.

The USART receiver block, RX, monitors the serial input line, RXD, for valid input. The USART RX Shift Register (RSR) accepts valid characters via RXD. After a valid character is assembled in the RSR, it is passed to the USART RX Buffer Register FIFO to await access by the CPU or host via the generic host interface.

The USART transmitter block, TX, accepts data written by the CPU or host and buffers the data in the USART TX Holding Register FIFO (THR). The USART TX Shift Register (TSR) reads the data stored in the THR and assembles the data to transmit via the serial output pin, TXD1.

The USART Baud Rate Generator block, BRG, generates the timing enables used by the USART TX block. The BRG clock input source is PCLK. The main clock is divided down per the divisor specified in the DLL and DLM registers. This divided down clock is a 16x oversample clock, NBAUDOUT.

The interrupt interface contains registers IER and IIR. The interrupt interface receives several one clock wide enables from the TX and RX blocks.

Status information from the TX and RX is stored in the LSR. Control information for the TX and RX is stored in the LCR.

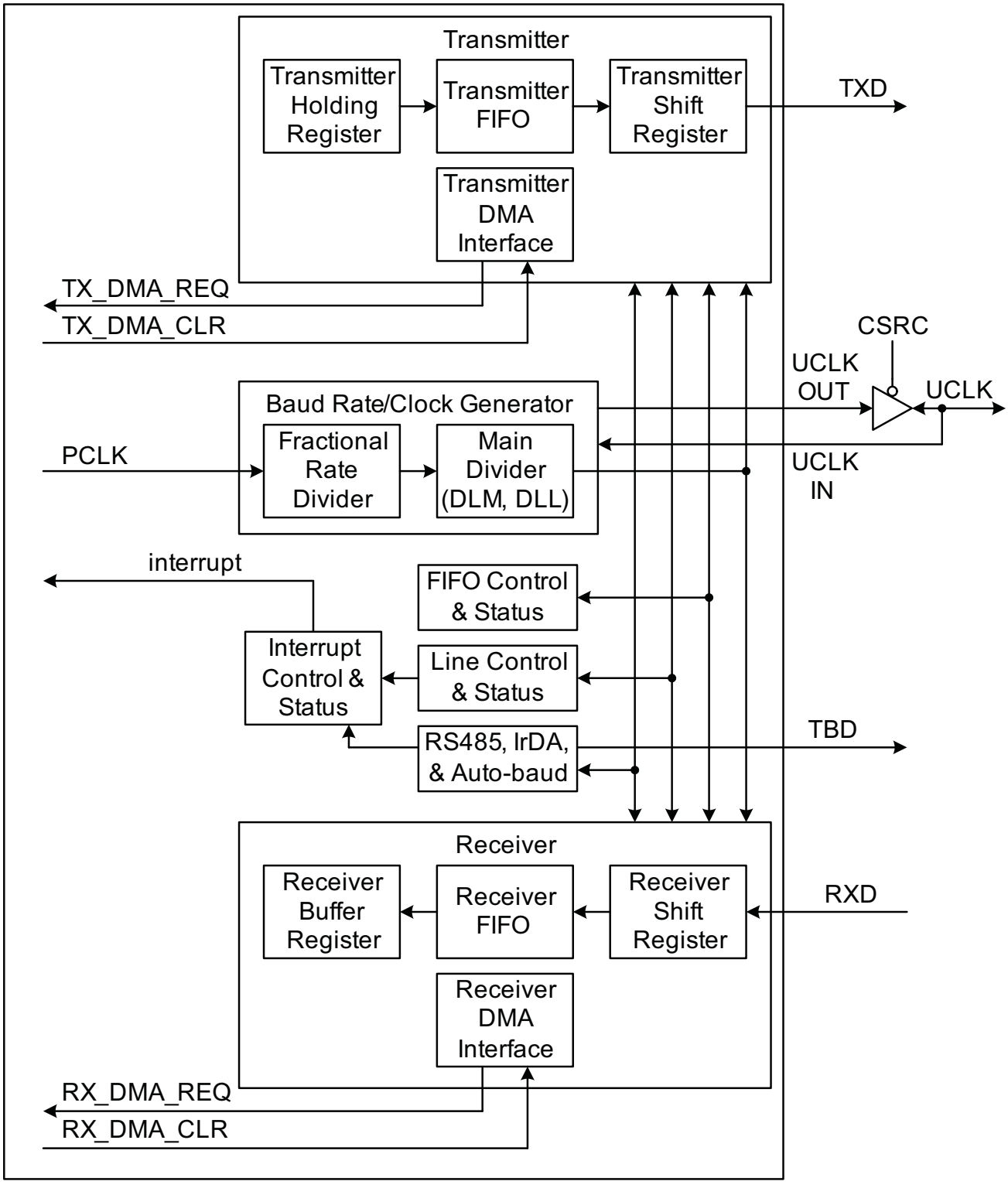


Fig 108. USART block diagram

36.5 Pin description

Table 801. USART0/2/3 pin description

Function name	Direction	Description
USART0		
U0_RXD	I	**Serial Input.** Serial receive data.
U0_TXD	O	**Serial Output.** Serial transmit data.
U0_DIR	I/O	RS-485/EIA-485 output enable/direction control.
U0_UCLK	I/O	Serial clock input/output for USART0 in synchronous mode.
USART2		
U2_RXD	I	**Serial Input.** Serial receive data.
U2_TXD	O	**Serial Output.** Serial transmit data.
U2_DIR	I/O	RS-485/EIA-485 output enable/direction control.
U2_UCLK	I/O	Serial clock input/output for USART2 in synchronous mode.
USART3		
U3_RXD	I	**Serial Input.** Serial receive data.
U3_TXD	O	**Serial Output.** Serial transmit data.
U3_DIR	I/O	RS-485/EIA-485 output enable/direction control.
U3_UCLK	I/O	Serial clock input/output for USART3 in synchronous mode.
U3_BAUD	O	USART3 baud output. U3_BAUD is an active LOW signal of the single clock cycle and is generated at each rising edge of a 16x clock signal for the transmitter section of the UART. The clock rate is established by the USART3 clock frequency divided by the fractional divider and the divisor specified in the baud generator divisor latches. U3_BAUD can be used as an input to an external IrDA module.

36.6 Register description

The USART contains registers organized as shown in Table 802. The Divisor Latch Access Bit (DLAB) is contained in LCR[7] and enables access to the Divisor Latches.

Reset value reflects the data stored in used bits only. It does not include the content of reserved bits.

Table 802. Register overview: USART0/2/3 (base address: 0x4008 1000, 0x400C 1000, 0x400C 2000)

Name	Access	Address offset	Description	Reset value	Reference
RBR	RO	0x000	Receiver Buffer Register. Contains the next received character to be read (DLAB = 0).	NA	Table 803
THR	WO	0x000	Transmit Holding Register. The next character to be transmitted is written here (DLAB = 0).	NA	Table 804
DLL	R/W	0x000	Divisor Latch LSB. Least significant byte of the baud rate divisor value. The full divisor is used to generate a baud rate from the fractional rate divider (DLAB = 1).	0x01	Table 805
DLM	R/W	0x004	Divisor Latch MSB. Most significant byte of the baud rate divisor value. The full divisor is used to generate a baud rate from the fractional rate divider (DLAB = 1).	0x00	Table 806

Table 802. Register overview: USART0/2/3 (base address: 0x4008 1000, 0x400C 1000, 0x400C 2000)

Name	Access	Address offset	Description	Reset value	Reference
IER	R/W	0x004	Interrupt Enable Register. Contains individual interrupt enable bits for the 7 potential USART interrupts (DLAB = 0).	0x00	Table 807
IIR	RO	0x008	Interrupt ID Register. Identifies which interrupt(s) are pending.	0x01	Table 808
FCR	WO	0x008	FIFO Control Register. Controls USART FIFO usage and modes.	0x00	Table 810
LCR	R/W	0x00C	Line Control Register. Contains controls for frame formatting and break generation.	0x00	Table 811
-	-	0x010	Reserved	-	-
LSR	RO	0x014	Line Status Register. Contains flags for transmit and receive status, including line errors.	0x60	Table 812
-	-	0x018	Reserved	-	-
SCR	R/W	0x01C	Scratch Pad Register. Eight-bit temporary storage for software.	0x00	Table 813
ACR	R/W	0x020	Auto-baud Control Register. Contains controls for the auto-baud feature.	0x00	Table 814
ICR	R/W	0x024	IrDA control register (USART3 only)	0x00	Table 815
FDR	R/W	0x028	Fractional Divider Register. Generates a clock input for the baud rate divider.	0x10	Table 817
OSR	R/W	0x02C	Oversampling Register. Controls the degree of oversampling during each bit time.	0xF0	Table 819
-	-	0x030 - 0x03C	Reserved	-	-
HDEN	R/W	0x040	Half-duplex enable Register		Table 820
-	-	0x044	Reserved	-	-
SCICTRL	R/W	0x048	Smart card interface control register	0x00	Table 821
RS485CTRL	R/W	0x04C	RS-485/EIA-485 Control. Contains controls to configure various aspects of RS-485/EIA-485 modes.	0x00	Table 822
RS485ADRMATCH	R/W	0x050	RS-485/EIA-485 address match. Contains the address match value for RS-485/EIA-485 mode.	0x00	Table 823
RS485DLY	R/W	0x054	RS-485/EIA-485 direction control delay.	0x00	Table 824
SYNCCTRL	R/W	0x058	Synchronous mode control register.	0x00	Table 825
TER	R/W	0x05C	Transmit Enable Register. Turns off USART transmitter for use with software flow control.	0x01	Table 826

36.6.1 USART Receiver Buffer Register

The RBR is the top byte of the USART RX FIFO. The top byte of the RX FIFO contains the oldest character received and can be read via the bus interface. The LSB (bit 0) represents the “oldest” received data bit. If the character received is less than 8 bits, the unused MSBs are padded with zeroes.

The Divisor Latch Access Bit (DLAB) in LCR must be zero in order to access the RBR. The RBR is always Read Only.

Since PE, FE and BI bits (see Table 812) correspond to the byte sitting on the top of the RBR FIFO (i.e. the one that will be read in the next read from the RBR), the right approach for fetching the valid pair of received byte and its status bits is first to read the content of the LSR register, and then to read a byte from the RBR.

Table 803. USART Receiver Buffer Registers when DLAB = 0, Read Only (RBR - addresses 0x4008 1000 (USART0), 0x400C 1000 (USART2), 0x400C 2000 (USART3)) bit description

Bit	Symbol	Description	Reset value
7:0	RBR	Receiver buffer. The USART Receiver Buffer Register contains the oldest received byte in the USART RX FIFO.	undefined
31:8	-	Reserved	-

36.6.2 USART Transmitter Holding Register

The THR is the top byte of the USART TX FIFO. The top byte is the newest character in the TX FIFO and can be written via the bus interface. The LSB represents the first bit to transmit.

The Divisor Latch Access Bit (DLAB) in LCR must be zero in order to access the THR. The THR is always Write Only.

Table 804. USART Transmitter Holding Register when DLAB = 0, Write Only (THR - addresses 0x4008 1000 (USART0), 0x400C 1000 (USART2), 0x400C 2000 (USART3)) bit description

Bit	Symbol	Description	Reset value
7:0	THR	Transmit Holding Register. Writing to the USART Transmit Holding Register causes the data to be stored in the USART transmit FIFO. The byte will be sent when it reaches the bottom of the FIFO and the transmitter is available.	NA
31:8	-	Reserved	-

36.6.3 USART Divisor Latch LSB and MSB Registers

The USART Divisor Latch is part of the USART Baud Rate Generator and holds the value used, along with the Fractional Divider, to divide the USART_PCLK clock in order to produce the baud rate clock, which must be 16x the desired baud rate. The DLL and DLM registers together form a 16-bit divisor where DLL contains the lower 8 bits of the divisor and DLM contains the higher 8 bits of the divisor. A 0x0000 value is treated like a 0x0001 value as division by zero is not allowed.The Divisor Latch Access Bit (DLAB) in LCR must be one in order to access the USART Divisor Latches. Details on how to select the right value for DLL and DLM can be found in Section 36.6.12.

Table 805. USART Divisor Latch LSB Register when DLAB = 1 (DLL - addresses 0x4008 1000 (USART0), 0x400C 1000 (USART2), 0x400C 2000 (USART3)) bit description

Bit	Symbol	Description	Reset value
7:0	DLLSB	Divisor latch LSB. The USART Divisor Latch LSB Register, along with the DLM register, determines the baud rate of the USART.	0x01
31:8	-	Reserved	-

Table 806. USART Divisor Latch MSB Register when DLAB = 1 (DLM - addresses 0x4008 1004 (USART0), 0x400C 1004 (USART2), 0x400C 2004 (USART3)) bit description

Bit	Symbol	Description	Reset value
7:0	DLMSB	Divisor latch MSB. The USART Divisor Latch MSB Register, along with the DLL register, determines the baud rate of the USART.	0x00
31:8	-	Reserved	-

36.6.4 USART Interrupt Enable Register

The IER is used to enable the four USART interrupt sources.

Table 807. USART Interrupt Enable Register when DLAB = 0 (IER - addresses 0x4008 1004 (USART0), 0x400C 1004 (USART2), 0x400C 2004 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	RBRIE		RBR Interrupt Enable. Enables the Receive Data Available interrupt for USART. It also controls the Character Receive Time-out interrupt.	0
		0	Disable the RDA interrupt.	
		1	Enable the RDA interrupt.	
1	THREIE		THRE Interrupt Enable. Enables the THRE interrupt for USART. The status of this interrupt can be read from LSR[5].	0
		0	Disable the THRE interrupt.	
		1	Enable the THRE interrupt.	
2	RXIE		RX Line Interrupt Enable. Enables the USART RX line status interrupts. The status of this interrupt can be read from LSR[4:1].	0
		0	Disable the RX line status interrupts.	
		1	Enable the RX line status interrupts.	
3	-	-	Reserved	-
6:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7	-	-	Reserved	0
8	ABEOINTEN		Enables the end of auto-baud interrupt.	0
		0	Disable end of auto-baud Interrupt.	
		1	Enable end of auto-baud Interrupt.	

Table 807. USART Interrupt Enable Register when DLAB = 0 (IER - addresses 0x4008 1004 (USART0), 0x400C 1004 (USART2), 0x400C 2004 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
9	ABTOINTEN		Enables the auto-baud time-out interrupt.	0
		0	Disable auto-baud time-out Interrupt.	
		1	Enable auto-baud time-out Interrupt.	
31:10	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

36.6.5 USART Interrupt Identification Register

IIR provides a status code that denotes the priority and source of a pending interrupt. The interrupts are frozen during a IIR access. If an interrupt occurs during a IIR access, the interrupt is recorded for the next IIR access.

Table 808. USART Interrupt Identification Register, read only (IIR - addresses 0x4008 1008 (USART0), 0x400C 1008 (USART2), 0x400C 2008 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	INTSTATUS		Interrupt status. Note that IIR[0] is active low. The pending interrupt can be determined by evaluating IIR[3:1].	1
		0	At least one interrupt is pending.	
		1	No interrupt is pending.	
3:1	INTID		Interrupt identification. IER[3:1] identifies an interrupt corresponding to the USART Rx FIFO. All other combinations of IER[3:1] not listed below are reserved (100,101,111).	0
		0x3	Priority 1 (highest) - Receive Line Status (RLS).	
		0x2	Priority 2 - Receive Data Available (RDA).	
		0x6	Priority 2 - Character Time-out Indicator (CTI).	
		0x1	Priority 3 - THRE Interrupt.	
		0x0	Priority 4 (lowest) - Reserved.	
5:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	FIFOENABLE		Copies of FCR[0].	0
8	ABEOINT		End of auto-baud interrupt. True if auto-baud has finished successfully and interrupt is enabled.	0
9	ABTOINT		Auto-baud time-out interrupt. True if auto-baud has timed out and interrupt is enabled.	0
31:10	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Bits IIR[9:8] are set by the auto-baud function and signal a time-out or end of auto-baud condition. The auto-baud interrupt conditions are cleared by setting the corresponding Clear bits in the Auto-baud Control Register.

If the IntStatus bit is one and no interrupt is pending and the IntId bits will be zero. If the IntStatus is 0, a non auto-baud interrupt is pending in which case the IntId bits identify the type of interrupt and handling as described in Table 809. Given the status of IIR[3:0], an interrupt handler routine can determine the cause of the interrupt and how to clear the active interrupt. The IIR must be read in order to clear the interrupt prior to exiting the Interrupt Service Routine.

The USART RLS interrupt (IIR[3:1] = 011) is the highest priority interrupt and is set whenever any one of four error conditions occur on the USART RX input: overrun error (OE), parity error (PE), framing error (FE) and break interrupt (BI). The USART Rx error condition that set the interrupt can be observed via LSR[4:1]. The interrupt is cleared upon a LSR read.

The USART RDA interrupt (IIR[3:1] = 010) shares the second level priority with the CTI interrupt (IIR[3:1] = 110). The RDA is activated when the USART Rx FIFO reaches the trigger level defined in FCR7:6 and is reset when the USART Rx FIFO depth falls below the trigger level. When the RDA interrupt goes active, the CPU can read a block of data defined by the trigger level.

The CTI interrupt (IIR[3:1] = 110) is a second level interrupt and is set when the USART Rx FIFO contains at least one character and no USART Rx FIFO activity has occurred in 3.5 to 4.5 character times. Any USART Rx FIFO activity (read or write of USART RSR) will clear the interrupt. This interrupt is intended to flush the USART RBR after a message has been received that is not a multiple of the trigger level size. For example, if a peripheral wished to send a 105 character message and the trigger level was 10 characters, the CPU would receive 10 RDA interrupts resulting in the transfer of 100 characters and 1 to 5 CTI interrupts (depending on the service routine) resulting in the transfer of the remaining 5 characters.

Table 809. USART Interrupt Handling

IIR[3:0] value[1]	Priority	Interrupt type	Interrupt source	Interrupt reset
0001	-	None	None	-
0110	Highest	RX Line Status / Error	OE[2] or PE[2] or FE[2] or BI[2]	LSR Read[2]

Table 809. USART Interrupt Handling

IIR[3:0] value[1]	Priority	Interrupt type	Interrupt source	Interrupt reset
0100	Second	RX Data Available	Rx data available or trigger level reached in FIFO (FCR0=1)	RBR Read[3] or USART FIFO drops below trigger level
1100	Second	Character Time-out indication	Minimum of one character in the RX FIFO and no character input or removed during a time period depending on how many characters are in FIFO and what the trigger level is set at (3.5 to 4.5 character times). The exact time will be: [(word length) × 7 - 2] × 8 + [(trigger level - number of characters) × 8 + 1] RCLKs	RBR Read[3]
0010	Third	THRE	THRE[2]	IIR Read[4] (if source of interrupt) or THR write

[1] Values 0000, 0011, 010, 0111, 1000, 1001, 1010, 1011,1101, 1110,1111 are reserved.

[2] For details see Section 36.6.8 “USART Line Status Register”

[3] For details see Section 36.6.1 “USART Receiver Buffer Register”

[4] For details see Section 36.6.5 “USART Interrupt Identification Register” and Section 36.6.2 “USART Transmitter Holding Register”

The USART THRE interrupt (IIR[3:1] = 001) is a third level interrupt and is activated when the USART THR FIFO is empty provided certain initialization conditions have been met. These initialization conditions are intended to give the USART THR FIFO a chance to fill up with data to eliminate many THRE interrupts from occurring at system start-up. The initialization conditions implement a one character delay minus the stop bit whenever THRE = 1 and there have not been at least two characters in the THR at one time since the last THRE = 1 event. This delay is provided to give the CPU time to write data to THR without a THRE interrupt to decode and service. A THRE interrupt is set immediately if the USART THR FIFO has held two or more characters at one time and currently, the THR is empty. The THRE interrupt is reset when a THR write occurs or a read of the IIR occurs and the THRE is the highest interrupt (IIR[3:1] = 001).

36.6.6 USART FIFO Control Register

The FCR controls the operation of the USART RX and TX FIFOs.

Table 810. USART FIFO Control Register Write Only (FCR - addresses 0x4008 1008 (USART0), 0x400C 1008 (USART2), 0x400C 2008 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	FIFOEN		FIFO Enable.	0
		0	USART FIFOs are disabled. Must not be used in the application.	
		1	Active high enable for both USART Rx and TX FIFOs and FCR[7:1] access. This bit must be set for proper USART operation. Any transition on this bit will automatically clear the USART FIFOs.	
1	RXFIFO RES		RX FIFO Reset.	0
		0	No impact on either of USART FIFOs.	
		1	Writing a logic 1 to FCR[1] will clear all bytes in USART Rx FIFO, reset the pointer logic. This bit is self-clearing.	
2	TXFIFO RES		TX FIFO Reset.	0
		0	No impact on either of USART FIFOs.	
		1	Writing a logic 1 to FCR[2] will clear all bytes in USART TX FIFO, reset the pointer logic. This bit is self-clearing.	
3	DMAMO DE		DMA Mode Select. When the FIFO enable bit (bit 0 of this register) is set, this bit selects the DMA mode.	0
5:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	RXTRIG LVL		RX Trigger Level. These two bits determine how many receiver USART FIFO characters must be written before an interrupt is activated.	0
		0x0	Trigger level 0 (1 character or 0x01).	
		0x1	Trigger level 1 (4 characters or 0x04).	
		0x2	Trigger level 2 (8 characters or 0x08).	
		0x3	Trigger level 3 (14 characters or 0x0E).	
31:8	-	-	Reserved	-

36.6.6.1 DMA Operation

The user can optionally operate the USART transmit and/or receive using DMA. The DMA mode is determined by the DMA Mode Select bit in the FCR register. Note that for DMA operation as for any operation of the USART, the FIFOs must be enabled via the FIFO Enable bit in the FCR register.

USART receiver DMA

In DMA mode, the receiver DMA request is asserted when the receiver FIFO level becomes equal to or greater than trigger level, or if a character time-out occurs. See the description of the RX Trigger Level above. The receiver DMA request is cleared by the DMA controller.

USART transmitter DMA

In DMA mode, the transmitter DMA request is asserted when the transmitter FIFO transitions to not full. The transmitter DMA request is cleared by the DMA controller.

36.6.7 USART Line Control Register

The LCR determines the format of the data character that is to be transmitted or received.

Table 811. USART Line Control Register (LCR - addresses 0x4008 100C (USART0), 0x400C 100C (USART2), 0x400C 200C (USART3)) bit description

Bit	Symbol	Value	Description	Reset Value
1:0	WLS		Word Length Select.	0
		0x0	5-bit character length.	
		0x1	6-bit character length.	
		0x2	7-bit character length.	
		0x3	8-bit character length.	
2	SBS		Stop Bit Select.	0
		0	1 stop bit.	
		1	2 stop bits (1.5 if LCR[1:0]=00).	
3	PE		Parity Enable	0
		0	Disable parity generation and checking.	
		1	Enable parity generation and checking.	
5:4	PS		Parity Select.	0
		0x0	Odd parity. Number of 1s in the transmitted character and the attached parity bit will be odd.	
		0x1	Even Parity. Number of 1s in the transmitted character and the attached parity bit will be even.	
		0x2	Forced "1" stick parity.	
		0x3	Forced "0" stick parity.	
6	BC		Break Control.	0
		0	Disable break transmission.	
		1	Enable break transmission. Output pin USART TXD is forced to logic 0 when LCR[6] is active high.	
7	DLAB		Divisor Latch Access Bit.	0
		0	Disable access to Divisor Latches.	
		1	Enable access to Divisor Latches.	
31:8	-	-	Reserved	-

36.6.8 USART Line Status Register

The LSR is a Read Only register that provides status information on the USART TX and RX blocks.

Table 812. USART Line Status Register Read Only (LSR - addresses 0x4008 1014 (USART0), 0x400C 1014 (USART2), 0x400C 2014 (USART3)) bit description

Bit	Symbol	Value	Description	Reset Value
0	RDR		Receiver Data Ready. LSR[0] is set when the RBR holds an unread character and is cleared when the USART RBR FIFO is empty.	0
		0	RBR is empty.	
		1	RBR contains valid data.	
1	OE		Overrun Error. The overrun error condition is set as soon as it occurs. A LSR read clears LSR[1]. LSR[1] is set when USART RSR has a new character assembled and the USART RBR FIFO is full. In this case, the USART RBR FIFO will not be overwritten and the character in the USART RSR will be lost.	0
		0	Overrun error status is inactive.	
		1	Overrun error status is active.	
2	PE		Parity Error. When the parity bit of a received character is in the wrong state, a parity error occurs. A LSR read clears LSR[2]. Time of parity error detection is dependent on FCR[0]. **Note:** A parity error is associated with the character at the top of the USART RBR FIFO.	0
		0	Parity error status is inactive.	
		1	Parity error status is active.	
3	FE		Framing Error. When the stop bit of a received character is a logic 0, a framing error occurs. A LSR read clears LSR[3]. The time of the framing error detection is dependent on FCR0. Upon detection of a framing error, the RX will attempt to re-synchronize to the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no Framing Error. **Note:** A framing error is associated with the character at the top of the USART RBR FIFO.	0
		0	Framing error status is inactive.	
		1	Framing error status is active.	
4	BI		Break Interrupt. When RXD1 is held in the spacing state (all zeros) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until RXD1 goes to marking state (all ones). A LSR read clears this status bit. The time of break detection is dependent on FCR[0]. **Note:** The break interrupt is associated with the character at the top of the USART RBR FIFO.	0
		0	Break interrupt status is inactive.	
		1	Break interrupt status is active.	

Table 812. USART Line Status Register Read Only (LSR - addresses 0x4008 1014 (USART0), 0x400C 1014 (USART2), 0x400C 2014 (USART3)) bit description *…continued*

Bit	Symbol	Value	Description	Reset Value
5	THRE		Transmitter Holding Register Empty. THRE is set immediately upon detection of an empty USART THR and is cleared on a THR write.	1
		0	THR contains valid data.	
		1	THR is empty.	
6	TEMT		Transmitter Empty. TEMT is set when both THR and TSR are empty; TEMT is cleared when either the TSR or the THR contain valid data.	1
		0	THR and/or the TSR contains valid data.	
		1	THR and the TSR are empty.	
7	RXFE		Error in RX FIFO. LSR[7] is set when a character with a RX error such as framing error, parity error or break interrupt, is loaded into the RBR. This bit is cleared when the LSR register is read and there are no subsequent errors in the USART FIFO.	0
		0	RBR contains no USART RX errors or FCR[0]=0.	
		1	USART RBR contains at least one USART RX error.	
8	TXERR		Error in transmitted character. A NACK response is given by the receiver in Smart card T=0 mode. This bit is cleared when the LSR register is read.	0
		0	No error (normal default condition).	
		1	A NACK response is received during Smart card T=0 operation.	
31:9	-	-	Reserved	-

36.6.9 USART Scratch Pad Register

The SCR has no effect on the USART operation. This register can be written and/or read at user's discretion. There is no provision in the interrupt interface that would indicate to the host that a read or write of the SCR has occurred.

Table 813. USART Scratch Pad Register (SCR - addresses 0x4008 101C (USART0), 0x400C 101C (USART2), 0x400C 201C (USART3)) bit description

Bit	Symbol	Description	Reset Value
7:0	PAD	Scratch pad. A readable, writable byte.	0x00
31:8	-	Reserved	-

36.6.10 USART Auto-baud Control Register

The USART Auto-baud Control Register (ACR) controls the process of measuring the incoming clock/data rate for the baud rate generation and can be read and written at user's discretion.

Table 814. Autobaud Control Register (ACR - addresses 0x4008 1020 (USART0), 0x400C 1020 (USART2), 0x400C 2020 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	START		Start bit. This bit is automatically cleared after auto-baud completion.	0
		0	Auto-baud stop (auto-baud is not running).	
		1	Auto-baud start (auto-baud is running). Auto-baud run bit. This bit is automatically cleared after auto-baud completion.	
1	MODE		Auto-baud mode select bit.	0
		0	Mode 0.	
		1	Mode 1.	
2	AUTORESTART		Restart bit.	0
		0	No restart	
		1	Restart in case of time-out (counter restarts at next USART Rx falling edge)	
7:3	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
8	ABEOINTCLR		End of auto-baud interrupt clear bit (write-only).	0
		0	Writing a 0 has no impact.	
		1	Writing a 1 will clear the corresponding interrupt in the IIR.	
9	ABTOINTCLR		Auto-baud time-out interrupt clear bit (write-only).	0
		0	Writing a 0 has no impact.	
		1	Writing a 1 will clear the corresponding interrupt in the IIR.	
31:10	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

36.6.10.1 Auto-baud

The USART auto-baud function can be used to measure the incoming baud rate based on the ”AT" protocol (Hayes command). If enabled the auto-baud feature will measure the bit time of the receive data stream and set the divisor latch registers DLM and DLL accordingly.

Auto-baud is started by setting the ACR Start bit. Auto-baud can be stopped by clearing the ACR Start bit. The Start bit will clear once auto-baud has finished and reading the bit will return the status of auto-baud (pending/finished).

Two auto-baud measuring modes are available which can be selected by the ACR Mode bit. In Mode 0 the baud rate is measured on two subsequent falling edges of the USART Rx pin (the falling edge of the start bit and the falling edge of the least significant bit). In Mode 1 the baud rate is measured between the falling edge and the subsequent rising edge of the USART Rx pin (the length of the start bit).

The ACR AutoRestart bit can be used to automatically restart baud rate measurement if a time-out occurs (the rate measurement counter overflows). If this bit is set, the rate measurement will restart at the next falling edge of the USART Rx pin.

The auto-baud function can generate two interrupts.

- The IIR ABTOInt interrupt will get set if the interrupt is enabled (IER ABToIntEn is set and the auto-baud rate measurement counter overflows).
- The IIR ABEOInt interrupt will get set if the interrupt is enabled (IER ABEOIntEn is set and the auto-baud has completed successfully).

The auto-baud interrupts have to be cleared by setting the corresponding ACR ABTOIntClr and ABEOIntEn bits.

The fractional baud rate generator must be disabled (DIVADDVAL = 0) during auto-baud. Also, when auto-baud is used, any write to DLM and DLL registers should be done before ACR register write. The minimum and the maximum baud rates supported by USART are function of USART_PCLK, number of data bits, stop bits and parity bits.

(6)

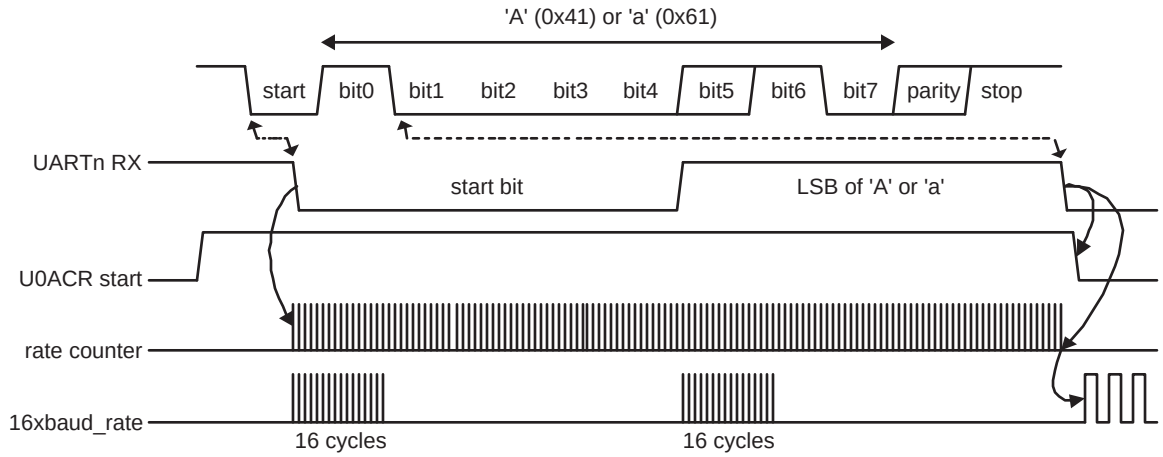
$$ratemin = \frac{2 \times PCLK}{16 \times 2^{15}} \leq UART_{baudrate} \leq \frac{PCLK}{16 \times (2 + databits + paritybits + stopbits)} = ratemax$$

36.6.10.2 Auto-baud modes

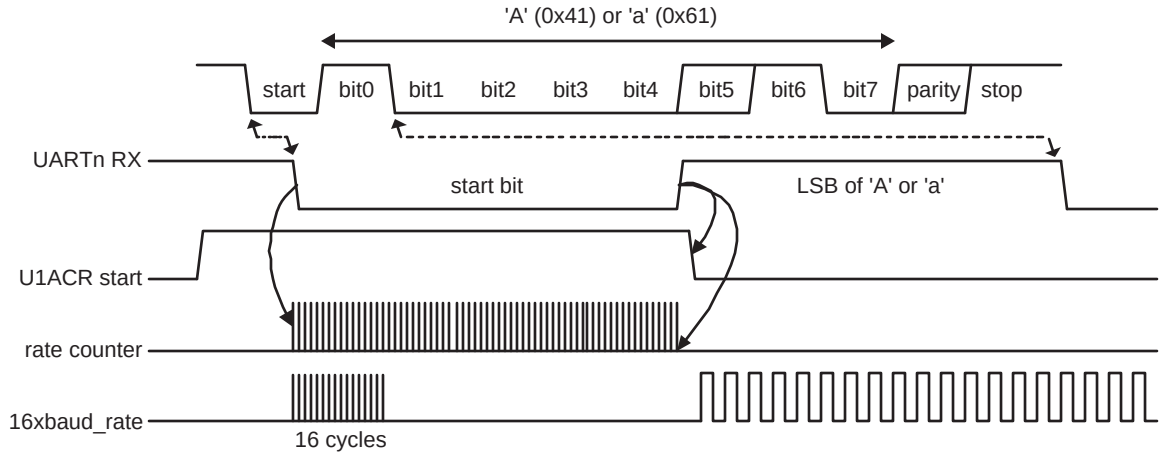
When the software is expecting an ”AT" command, it configures the USART with the expected character format and sets the ACR Start bit. The initial values in the divisor latches DLM and DLM don‘t care. Because of the ”A" or ”a" ASCII coding (”A" = 0x41, ”a" = 0x61), the USART Rx pin sensed start bit and the LSB of the expected character are delimited by two falling edges. When the ACR Start bit is set, the auto-baud protocol will execute the following phases:

1. On ACR Start bit setting, the baud rate measurement counter is reset and the USART RSR is reset. The RSR baud rate is switched to the highest rate.
2. A falling edge on USART Rx pin triggers the beginning of the start bit. The rate measuring counter will start counting USART_PCLK cycles.
3. During the receipt of the start bit, 16 pulses are generated on the RSR baud input with the frequency of the USART input clock, guaranteeing the start bit is stored in the RSR.
4. During the receipt of the start bit (and the character LSB for Mode = 0), the rate counter will continue incrementing with the pre-scaled USART input clock (USART_PCLK).
5. If Mode = 0, the rate counter will stop on next falling edge of the USART Rx pin. If Mode = 1, the rate counter will stop on the next rising edge of the USART Rx pin.

6. The rate counter is loaded into DLM/DLL and the baud rate will be switched to normal operation. After setting the DLM/DLL, the end of auto-baud interrupt IIR ABEOInt will be set, if enabled. The RSR will now continue receiving the remaining bits of the "A/a" character.



a. Mode 0 (start bit and LSB are used for auto-baud)



b. Mode 1 (only start bit is used for auto-baud)

Fig 109. Auto-baud a) mode 0 and b) mode 1 waveform

36.6.11 IrDA Control Register (USART3)

The IrDA Control Register enables and configures the IrDA mode for USART3 only. The value of U3ICR should not be changed while transmitting or receiving data, or data loss or corruption may occur.

Remark: IrDA is available on USART3 only.

Table 815. IrDA Control Register (ICR - address 0x4000 8024) bit description

Bit	Symbol	Value	Description	Reset value
0	IRDAEN		IrDA mode enable.	0
		0	IrDA mode on USART3 is disabled, USART3 acts as a standard USART.	
		1	IrDA mode on USART3 is enabled.	
1	IRDAINV		Serial input direction.	0
		0	The serial input is not inverted.	
		1	The serial input is inverted. This has no effect on the serial output.	
2	FIXPULSEEN		IrDA fixed pulse width mode.	0
		0	IrDA fixed pulse width mode disabled.	
		1	IrDA fixed pulse width mode enabled.	
5:3	PULSEDIV		Configures the pulse when FixPulseEn = 1. See Table 816 for details.	0
31:6	-	NA	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

The PulseDiv bits in U3ICR are used to select the pulse width when the fixed pulse width mode is used in IrDA mode (IrDAEn = 1 and FixPulseEn = 1). The value of these bits should be set so that the resulting pulse width is at least 1.63 µs. Table 816 shows the possible pulse widths.

Table 816. IrDA Pulse Width

FixPulseEn	PulseDiv	IrDA Transmitter Pulse width (µs)
0	x	3 / (16 × baud rate)
1	0	$2 \times T_{PCLK}$
1	1	$4 \times T_{PCLK}$
1	2	$8 \times T_{PCLK}$
1	3	$16 \times T_{PCLK}$
1	4	$32 \times T_{PCLK}$
1	5	$64 \times T_{PCLK}$
1	6	$128 \times T_{PCLK}$
1	7	$256 \times T_{PCLK}$

36.6.12 USART Fractional Divider Register (U0FDR - 0x4000 8028)

The USART Fractional Divider Register (FDR) controls the clock pre-scaler for the baud rate generation and can be read and written at the user's discretion. This pre-scaler takes the APB clock and generates an output clock according to the specified fractional requirements.

Important: If the fractional divider is active (DIVADDVAL > 0) and DLM = 0, the value of the DLL register must be 3 or greater.

Table 817. USART Fractional Divider Register (FDR - addresses 0x4008 1028 (USART0), 0x400C 1028 (USART2), 0x400C 2028 (USART3)) bit description

Bit	Function	Description	Reset value
3:0	DIVADDVAL	Baud rate generation pre-scaler divisor value. If this field is 0, fractional baud rate generator will not impact the USART baud rate.	0
7:4	MULVAL	Baud rate pre-scaler multiplier value. This field must be greater or equal 1 for USART to operate properly, regardless of whether the fractional baud rate generator is used or not.	1
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

This register controls the clock pre-scaler for the baud rate generation. The reset value of the register keeps the fractional capabilities of USART disabled making sure that USART is fully software and hardware compatible with USARTs not equipped with this feature.

The USART baud rate can be calculated as:

(7)

$$UART_{baudrate} = \frac{PCLK}{16 \times (256 \times DLM + DLL) \times \left(1 + \frac{DivAddVal}{MulVal}\right)}$$

Where USART_PCLK is the peripheral clock, DLM and DLL are the standard USART baud rate divider registers, and DIVADDVAL and MULVAL are USART fractional baud rate generator specific parameters.

The value of MULVAL and DIVADDVAL should comply to the following conditions:

1. $1 \leq MULVAL \leq 15$
2. $0 \leq DIVADDVAL \leq 14$
3. DIVADDVAL< MULVAL

The value of the FDR should not be modified while transmitting/receiving data or data may be lost or corrupted.

If the FDR register value does not comply to these two requests, then the fractional divider output is undefined. If DIVADDVAL is zero then the fractional divider is disabled, and the clock will not be divided.

36.6.12.1 Baud rate calculation

USART can operate with or without using the Fractional Divider. In real-life applications it is likely that the desired baud rate can be achieved using several different Fractional Divider settings. The following algorithm illustrates one way of finding a set of DLM, DLL, MULVAL, and DIVADDVAL values. Such set of parameters yields a baud rate with a relative error of less than 1.1% from the desired one.

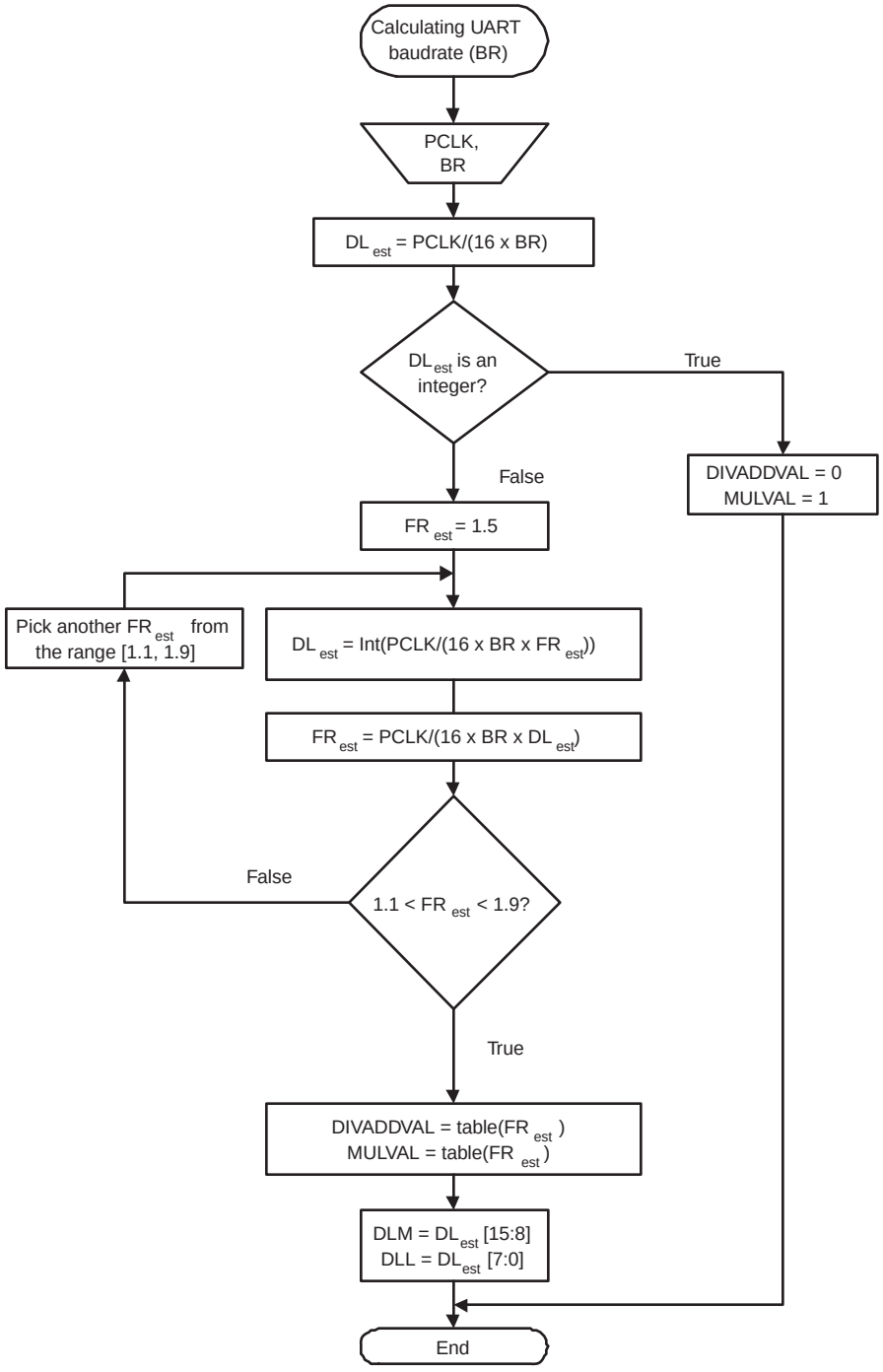


Fig 110. Algorithm for setting USART dividers

Table 818. Fractional Divider setting look-up table

FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal
1.000	0/1	1.250	1/4	1.500	1/2	1.750	3/4
1.067	1/15	1.267	4/15	1.533	8/15	1.769	10/13
1.071	1/14	1.273	3/11	1.538	7/13	1.778	7/9
1.077	1/13	1.286	2/7	1.545	6/11	1.786	11/14
1.083	1/12	1.300	3/10	1.556	5/9	1.800	4/5
1.091	1/11	1.308	4/13	1.571	4/7	1.818	9/11
1.100	1/10	1.333	1/3	1.583	7/12	1.833	5/6
1.111	1/9	1.357	5/14	1.600	3/5	1.846	11/13
1.125	1/8	1.364	4/11	1.615	8/13	1.857	6/7
1.133	2/15	1.375	3/8	1.625	5/8	1.867	13/15
1.143	1/7	1.385	5/13	1.636	7/11	1.875	7/8
1.154	2/13	1.400	2/5	1.643	9/14	1.889	8/9
1.167	1/6	1.417	5/12	1.667	2/3	1.900	9/10
1.182	2/11	1.429	3/7	1.692	9/13	1.909	10/11
1.200	1/5	1.444	4/9	1.700	7/10	1.917	11/12
1.214	3/14	1.455	5/11	1.714	5/7	1.923	12/13
1.222	2/9	1.462	6/13	1.727	8/11	1.929	13/14
1.231	3/13	1.467	7/15	1.733	11/15	1.933	14/15

36.6.12.1.1 Example 1: USART_PCLK = 14.7456 MHz, BR = 9600

According to the provided algorithm DL_{est} = PCLK/(16 x BR) = 14.7456 MHz / (16 x 9600) = 96. Since this DL_{est} is an integer number, DIVADDVAL = 0, MULVAL = 1, DLM = 0, and DLL = 96.

36.6.12.1.2 Example 2: USART_PCLK = 12 MHz, BR = 115200

According to the provided algorithm DL_{est} = PCLK/(16 x BR) = 12 MHz / (16 x 115200) = 6.51. This DL_{est} is not an integer number and the next step is to estimate the FR parameter. Using an initial estimate of FR_{est} = 1.5 a new DL_{est} = 4 is calculated and FR_{est} is recalculated as FR_{est} = 1.628. Since FRest = 1.628 is within the specified range of 1.1 and 1.9, DIVADDVAL and MULVAL values can be obtained from the attached look-up table.

The closest value for FRest = 1.628 in the look-up Table 818 is FR = 1.625. It is equivalent to DIVADDVAL = 5 and MULVAL = 8.

Based on these findings, the suggested USART setup would be: DLM = 0, DLL = 4, DIVADDVAL = 5, and MULVAL = 8. According to Equation 7, the USART's baud rate is 115384. This rate has a relative error of 0.16% from the originally specified 115200.

36.6.13 USART Oversampling Register

In most applications, the USART samples received data 16 times in each nominal bit time and sends bits that are 16 input clocks wide. This register allows software to control the ratio between the input clock and bit clock. Oversampling is required for smart card mode and provides an alternative to fractional division for other modes.

Table 819. USART Oversampling Register (OSR - addresses 0x4008 102C (USART0), 0x400C 102C (USART2), 0x400C 20402C (USART3)) bit description

Bit	Symbol	Description	Reset value
0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3:1	OSFRAC	Fractional part of the oversampling ratio, in units of 1/8th of an input clock period. (001 = 0.125, ..., 111 = 0.875)	0
7:4	OSINT	Integer part of the oversampling ratio, minus 1. The reset values equate to the normal operating mode of 16 input clocks per bit time.	0xF
14:8	FDINT	In Smart Card mode, these bits act as a more-significant extension of the OSint field, allowing an oversampling ratio up to 2048 as required by ISO7816-3. In Smart Card mode, bits 14:4 should initially be set to 371, yielding an oversampling ratio of 372.	0
31:15	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Example: For a baud rate of 3.25 Mbps with a 24 MHz USART clock frequency, the ideal oversampling ratio is 24/3.25 or 7.3846. Setting OSINT to 0110 (binary) for 7 clocks/bit and OSFrac to 011 (binary) for 0.375 clocks/bit, results in an oversampling ratio of 7.375.

In Smart card mode, OSInt is extended by FDINT. This extends the possible oversampling to 2048, as required to support ISO 7816-3. Note that this value can be exceeded when D<0, but this is not supported by the USART. When Smart card mode is enabled, the initial value of OSINT and FDINT should be programmed as 00101110011 (binary) (372 minus one).

36.6.14 USART Half-duplex enable register

Remark: The HDEN register should be disabled when in smart card mode (smart card by default runs in half-duplex mode).

After reset the USART will be in full-duplex mode, meaning that both TX and RX work independently. After setting the HDEN bit, the USART will be in half-duplex mode. In this mode, the USART ensures that the receiver is locked when idle, or will enter a locked state after having received a complete ongoing character reception. Line conflicts must be handled in software. The behavior of the USART is unpredictable when data is presented for reception while data is being transmitted.

For this reason, the value of the HDEN register should not be modified while sending or receiving data, or data may be lost or corrupted.

Table 820. USART Half duplex enable register (HDEN - addresses 0x4008 1040 (USART0), 0x400C 1040 (USART2), 0x400C 2040 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	HDEN		Half-duplex mode enable	0
		0	Disable half-duplex mode.	
		1	Enable half-duplex mode.	
31:1	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

36.6.15 USART Smart card interface control register

Table 821. USART Smart card interface control register (SCICTRL - addresses 0x4008 1048 (USART0), 0x400C 1048 (USART2), 0x400C 2048 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	SCIEN		Smart Card Interface Enable.	0
		0	Smart card interface disabled.	
		1	Asynchronous half duplex smart card interface is enabled.	
1	NACKDIS		NACK response disable. Only applicable in T=0.	0
		0	A NACK response is enabled.	
		1	A NACK response is inhibited.	
2	PROTSEL		Protocol selection as defined in the ISO7816-3 standard.	0
		0	T = 0	
		1	T = 1	
7:5	TXRETRY		Maximum number of retransmissions in case of a negative acknowledge (protocol T=0). When the retry counter is exceeded, the USART will be locked until the FIFO is cleared. A TX error interrupt is generated when enabled.	-
15:8	GUARDTIME		Extra guard time. No extra guard time (0x0) results in a standard guard time as defined in ISO 7816-3, depending on the protocol type. A guard time of 0xFF indicates a minimal guard time as defined for the selected protocol.	
31:16	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

After reset the USART smart card interface will be disabled. After setting the SCIEN bit the USART will be in ISO 7816-3 compliant asynchronous smart card mode T=0.

The NACKDIS bit is used to inhibit a nack response during T=0 (the I/O line is not pulled low during the guard time to indicate an erroneous reception). The received character will be stored in the RX FIFO but a parity error will be generated. It is up to the software to handle the incorrect received character.

The PROTSEL bit is used to selected between the two supported smart card protocols T=0 and T=1. More information on these protocols can be found in the ISO 7816-3 standard.

The retry bit field indicates the number of retransmission when receiving a NACK response, which can be up to 7 trails. When the number is exceeded, an interrupt is generated and the USART is locked until the FIFO is empty. This can be done by flushing the FIFO. When no FIFO is available, or the FIFO is already empty, the interrupt can be used by the software to determine the next action.

The guard time bit file is used to program the extra number of guard time cycles to allow the smart card to process the information before sending a response. The extra guard time can be programmed from 0 to 255, where 255 indicates the minimum possible character length. This value is depending on the selected protocol and can be either 11 etu for protocol T=1 or 12 etu for protocol T=0.

Waiting times as defined in the standard cannot be programmed directly, but are implemented using the CAP1 and CAP2 inputs of the timers. <tbd>

Remark: The SCICTRL register should not be modified while sending or receiving data, or data may be lost or corrupted.

Remark: The SCICTRL register should not be enabled in combination with the SYNCCTRL register, as only asynchronous smart card is supported.

36.6.16 USART RS485 Control register

The RS485CTRL register controls the configuration of the USART in RS-485/EIA-485 mode.

Table 822. USART RS485 Control register (RS485CTRL - addresses 0x4008 104C (USART0), 0x400C 104C (USART2), 0x400C 204C (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	NMMEN		NMM enable.	0
		0	RS-485/EIA-485 Normal Multidrop Mode (NMM) is disabled.	
		1	RS-485/EIA-485 Normal Multidrop Mode (NMM) is enabled. In this mode, an address is detected when a received byte causes the USART to set the parity error and generate an interrupt.	
1	RXDIS		Receiver enable.	0
		0	The receiver is enabled.	
		1	The receiver is disabled.	
2	AADEN		AAD enable	0
		0	Auto Address Detect (AAD) is disabled.	
		1	Auto Address Detect (AAD) is enabled.	
3	-	-	Reserved.	-
4	DCTRL		Direction control for DIR pin.	0
		0	Disable Auto Direction Control.	
		1	Enable Auto Direction Control.	

Table 822. USART RS485 Control register (RS485CTRL - addresses 0x4008 104C (USART0), 0x400C 104C (USART2), 0x400C 204C (USART3)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
5	OINV		Direction control pin polarity. This bit reverses the polarity of the direction control signal on the DIR pin.	0
		0	The direction control pin will be driven to logic ‘0’ when the transmitter has data to be sent. It will be driven to logic ‘1’ after the last bit of data has been transmitted.	
		1	The direction control pin will be driven to logic ‘1’ when the transmitter has data to be sent. It will be driven to logic ‘0’ after the last bit of data has been transmitted.	
31:6	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

After reset RS485 mode will be disabled. The RS485 feature allows the USART to be configured as one of multiple addressable slave receivers controlled by a single USART. In RS485 mode the USART differentiates between an address character and a data character by means of a ninth bit. The parity bit is used to implement this bit, and when set to ‘1’ indicates an address and when set to ‘0’ indicates data. RS485 mode is enabled by setting the NMMEN bit. The USART slave receiver can be assigned a unique address and, manually or automatically, reject or accept data based on a received address. See section Section 36.7.2 for details.

36.6.17 USART RS485 Address Match register

The RS485ADRMATCH register contains the address match value for RS-485/EIA-485 mode.

Table 823. USART RS485 Address Match register (RS485ADRMATCH - addresses 0x4008 1050 (USART0), 0x400C 1050 (USART2), 0x400C 2050 (USART3)) bit description

Bit	Symbol	Description	Reset value
7:0	ADRMATCH	Contains the address match value.	0x00
31:8	-	Reserved	-

The ADRMATCH bit field contains the slave address match value that is used to compare a received address value to. During automatic address detection, this value is used to accept or reject serial input data.

36.6.18 USART1 RS485 Delay value register

The user may program the 8-bit RS485DLY register with a delay between the last stop bit leaving the TXFIFO and the de-assertion of the DIR pin. This delay time is in periods of the baud clock. Any delay time from 0 to 255 bit times may be programmed.

Table 824. USART RS485 Delay value register (RS485DLY - addresses 0x4008 1054 (USART0), 0x400C 1054 (USART2), 0x400C 2054 (USART3)) bit description

Bit	Symbol	Description	Reset value
7:0	DLY	Contains the direction control delay value. This register works in conjunction with an 8-bit counter.	0x00
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

36.6.19 USART Synchronous mode control register

SYNCCTRL register is a Read/write register that controls the synchronous mode. The synchronous mode control module generates or receives the synchronous clock with the serial input/ output data and distributes the edge detect samples to the transmit and receive shift registers.

Table 825. USART Synchronous mode control registers (SYNCCTRL - address addresses 0x4008 1058 (USART0), 0x400C 1058 (USART2), 0x400C 2058 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
0	SYNC		Enables synchronous mode.	0
		0	Disabled	
		1	Enabled	
1	CSRC		Clock source select.	0
		0	Synchronous slave mode (SCLK in)	
		1	Synchronous master mode (SCLK out)	
2	FES		Falling edge sampling.	0
		0	RxD is sampled on the rising edge of SCLK.	
		1	RxD is sampled on the falling edge of SCLK.	
3	TSBYPASS		Transmit synchronization bypass in synchronous slave mode.	0
		0	The input clock is synchronized prior to being used in clock edge detection logic.	
		1	The input clock is not synchronized prior to being used in clock edge detection logic. This allows for a high er input clock rate at the expense of potential metastability.	
4	CSCEN		Continuous master clock enable (used only when CSRC is 1)	0
		0	SCLK cycles only when characters are being sent on TxD.	
		1	SCLK runs continuously (characters can be received on RxD independently from transmission on TxD).	
5	SSSDIS		Start/stop bits	0
		0	Send start and stop bits as in other modes.	
		1	Do not send start/stop bits.	

Table 825. USART Synchronous mode control registers (SYNCCTRL - address addresses 0x4008 1058 (USART0), 0x400C 1058 (USART2), 0x400C 2058 (USART3)) bit description

Bit	Symbol	Value	Description	Reset value
6	CCCLR		Continuous clock clear	0
		0	CSCEN is under software control.	
		1	Hardware clears CSCEN after each character is received.	
31:7	-		Reserved. The value read from a reserved bit is not defined.	NA

After reset, synchronous mode is disabled. Synchronous mode allows the user to send (synchronous master mode) or receive (synchronous slave mode) a clock with the serial input and output data. Synchronous mode is enabled by setting the SYNC bit. The CSRC bit can be used to switch between synchronous slave mode (logic 0) and synchronous master mode (logic 1). The serial data can either be sampled on the rising edge (default) or the falling edge of the serial clock. When the STARTSTOPDISABLE bit is set, the FES bit is hardware overwritten to sample on the falling edge.

A master clock is only required to generate a clock when transmitting data. In this case, data can only be received when data is transmitted. When the CSCEN bit is set, the clock will always be running (during synchronous master mode only), allowing data to be received continuously.

Note that this option should not be used in combination with STARTSTOPDISABLE (during full-duplex communication). The continuous clock can be automatically stopped by hardware after having received a complete character. This can be done by asserting the CCCLR bit. This is useful in half-duplex mode, where the clock cannot be generated by sending a character. After the reception of one character, the CSCEN bit is automatically cleared by hardware. When another character needs to be received, the CSCEN should be enabled again.

By default data transmission and reception performs the same in asynchronous mode and synchronous mode. When the STARTSTOPDISABLE bit is set, no start and stop bits are transmitted (nor are they received). This means that all bits that are send or received (a clock is running) are data bits.

Remark: The value of the SYNCCTRL register should not be modified while transmitting/receiving, data or data might get lost or corrupted.

Remark: The SYNCCTRL register should not be enabled in combination with the SCICTRL register, as only asynchronous smart card is supported.

36.6.20 USART Transmit Enable Register

In addition to being equipped with full hardware flow control (auto-cts and auto-rts mechanisms described above), TER enables implementation of software flow control. When TxEn = 1, USART transmitter will keep sending data as long as they are available. As soon as TxEn becomes 0, USART transmission will stop.

Table 826 describes how to use TXEN bit in order to achieve software flow control.

Table 826. USART Transmit Enable Register (TER - addresses 0x4008 1030 (USART0), 0x400C 1030 (USART2), 0x400C 205C (USART3)) bit description

Bit	Symbol	Description	Reset value
0	TXEN	Transmit enable. After reset transmission is enabled. When the TXEN bit is de-asserted, no data will be transmitted although data may be pending in the TSR or THR.	1
31:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

36.7 Functional description

36.7.1 Synchronous mode

When the synchronous receiver/ transmitter feature is configured (USART), the serial interface is extended with a serial input and output clock and an output enable for controlling the clock pad.

<tbd>

Fig 111. USART serial interface protocol

By default transmission and reception in synchronous mode operates uses the same protocol as in asynchronous mode. Synchronous mode can be configured using the Synchronous Mode Control Register. This register allows to control:

- The direction of the serial clock, i.e. synchronous slave or master mode
- The sampling edge of the serial clock
- Two-stage or one stage synchronization of the input serial clock during transmission
- During synchronous master mode, the clock can be continuous or disabled when in idle or break mode
- The transmission of start and stop bits can be omitted. Valid data is identified by a running clock. Sampling is always done on the falling edge of the serial clock

Data is shifted in the receive shift register at the sampling edge of the serial clock.

36.7.1.1 Synchronous slave mode

This mode is enabled by setting the CSRC bit of the control register to ‘0’. During synchronous slave mode, an external clock is required that clocks the serial input and output data. Note that internally, the serial clock is treated as a data signal. Edge detection on the serial clock is performed to synchronize the serial clock with the USART clock domain, hence no registers are clocked with the serial clock.

Reception

By default the received character is similar to the character in asynchronous mode. The serial data stream is kept HIGH when no data is available. During this time it is not required for the external serial clock to be running. The first bit that will be received is the

start bit. During this time, the external serial clock must be running. The beginning of the start bit can either be aligned with the rising edge of the serial clock (sampling on the falling edge) or the falling edge (sampling on the rising edge), see the FES bit in Table 825. When sampling on the rising edge, it is not required that the beginning of the start bit is aligned with a clock edge (the clock may not have been running before). In this case, the edge on the serial input data due to the start bit (logic 1 to 0) is used to determine the start of the character (see<tbd>).

Fig 112. Transmission of data in synchronous slave mode

The NOSTARTSTOPBITS bit of the Synchronous Mode Control register allows the user to disable the transmission/ reception of the start and stop bits, improving the efficiency of the USART. As a character is no longer identified by the start and stop bits, the serial clock is used to determine the data bits. When the serial clock is running, all data that is sampled is regarded as valid data.

In order to be able to identify the start of a character, the beginning of the character must be aligned with the rising edge of the serial clock. For this reason, the FES bit of the Synchronous Mode Control register is forced in hardware to ‘1’.

Directly after sampling the last bit, the character is stored in the receive FIFO.

Transmission

During synchronous slave mode, data can only be transmitted when the external serial clock is running. Hence, when no start and stop bits are sent, transmission can only take place when data is received from the master. When the start and stop bits are transmitted, the external clock may only be detected after the first half of the received start bit (sampling at the rising edge of the external serial clock). By using the edge created by the received start bit (logic 1 to 0), it is made sure that the start bit of the character that is to be transmitted by the slave is stable before this rising edge the external slave clock. In this way it is ensured, that the master receives as many bits as it has transmitted.

When the first sample edge of the incoming serial clock samples a ‘1’ on the serial input data (and start-stop bits are transmitted, thus the master has not initiated a transaction yet), it is assumed that the master is running a continuous clock (instead of only running the clock when sending data characters). The USART will not wait for a start bit from the master, but will immediately start transmitting data when available. Note that in this situation, the number of bits transmitted by the master and the number of bits transmitted by the slave (received by the master) may not be aligned. It is assumed that a higher level protocol ensures that complete characters are received when the master stops the clock.

Transmission of data during synchronous slave mode is most time-critical. First the external serial input clock must be detected using edge detection logic. Then, data needs to be shifted out and be stable before the sampling edge of the external serial clock.

Remark: In this mode the u_clk period is allowed to be 4x the serial clock period.

36.7.1.2 Synchronous master mode

Synchronous master mode is enabled by setting the CSRC register bit to ‘1’. In this mode, the external clock is generated internally by the baud-rate generation logic and is used to clock the input and output serial data. The functionality of the baud-rate generation is described in Section 36.6.12.1. Auto-baud is not supported during synchronous mode. The 1x baud rate clock is used to shift out the serial output data and to sample the serial input data.

Synchronous master mode behaves similar to the slave mode, except that the serial input data is not registered at the interface but is clocked in the USART clock domain at the sampling edge of the serial clock.

During synchronous master mode, when start and stop bits are transmitted, the user can enable the external clock continuously using cscen bit of the Synchronous Mode Control register. This allows the connected slave to transmit data even when no data is transmitted by the master itself.

36.7.2 RS-485/EIA-485 modes of operation

The RS-485/EIA-485 feature allows the USART to be configured as an addressable slave. The addressable slave is one of multiple slaves controlled by a single master.

The USART master transmitter will identify an address character by setting the parity (9th) bit to ‘1’. For data characters, the parity bit is set to ‘0’.

Each USART slave receiver can be assigned a unique address. The slave can be programmed to either manually or automatically reject data following an address which is not theirs.

RS-485/EIA-485 Normal Multidrop Mode (NMM)

Setting the RS485CTRL bit 0 enables this mode. In this mode, an address is detected when a received byte causes the USART to set the parity error and generate an interrupt.

If the receiver is disabled (RS485CTRL bit 1 = ‘1’), any received data bytes will be ignored and will not be stored in the RXFIFO. When an address byte is detected (parity bit = ‘1’) it will be placed into the RXFIFO and an Rx Data Ready Interrupt will be generated. The processor can then read the address byte and decide whether or not to enable the receiver to accept the following data.

While the receiver is enabled (RS485CTRL bit 1 =’0’), all received bytes will be accepted and stored in the RXFIFO regardless of whether they are data or address. When an address character is received a parity error interrupt will be generated and the processor can decide whether or not to disable the receiver.

RS-485/EIA-485 Auto Address Detection (AAD) mode

When both RS485CTRL register bits 0 (9-bit mode enable) and 2 (AAD mode enable) are set, the USART is in auto address detect mode.

In this mode, the receiver will compare any address byte received (parity = ‘1’) to the 8-bit value programmed into the RS485ADRMATCH register.

If the receiver is disabled (RS485CTRL bit 1 = ‘1’), any received byte will be discarded if it is either a data byte OR an address byte which fails to match the RS485ADRMATCH value.

When a matching address character is detected it will be pushed onto the RXFIFO along with the parity bit, and the receiver will be automatically enabled (RS485CTRL bit 1 will be cleared by hardware). The receiver will also generate an Rx Data Ready Interrupt.

While the receiver is enabled (RS485CTRL bit 1 = ‘0’), all bytes received will be accepted and stored in the RXFIFO until an address byte which does not match the RS485ADRMATCH value is received. When this occurs, the receiver will be automatically disabled in hardware (RS485CTRL bit 1 will be set), The received non-matching address character will not be stored in the RXFIFO.

RS-485/EIA-485 Auto Direction Control

RS485/EIA-485 mode includes the option of allowing the transmitter to automatically control the state of the DIR pin as a direction control output signal.

Setting RS485CTRL bit 4 = ‘1’ enables this feature.

When Auto Direction Control is enabled, the selected pin will be asserted (driven LOW) when the CPU writes data into the TXFIFO. The pin will be de-asserted (driven HIGH) once the last bit of data has been transmitted. See bits 4 and 5 in the RS485CTRL register.

The RS485CTRL bit 4 takes precedence over all other mechanisms controlling the direction control pin.

RS485/EIA-485 driver delay time

The driver delay time is the delay between the last stop bit leaving the TXFIFO and the de-assertion of the DIR pin. This delay time can be programmed in the 8-bit RS485DLY register. The delay time is in periods of the baud clock. Any delay time from 0 to 255 bit times may be used.

RS485/EIA-485 output inversion

The polarity of the direction control signal on the DIR pin can be reversed by programming bit 5 in the RS485CTRL register. When this bit is set, the direction control pin will be driven to logic 1 when the transmitter has data waiting to be sent. The direction control pin will be driven to logic 0 after the last bit of data has been transmitted.

36.7.3 Smart card mode

Figure 113 shows a typical asynchronous smart card application.

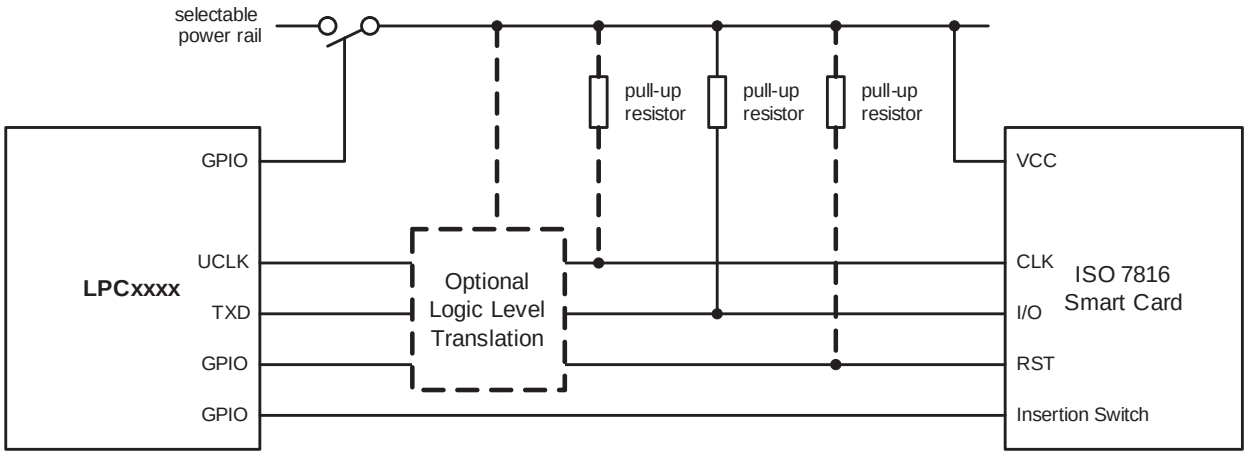


Fig 113. Typical smart card application

When the SCIEN bit in the SCICTRL register (Table 821) is set as described above, the USART provides bidirectional serial data on the open-drain TXD pin. No RXD pin is used when SCIEN is 1. The USART UCLK pin will output synchronously with the data at the data bit rate. Software must use timers to implement character and block waiting times (no hardware support via trigger signals is provided on this part). GPIO pins can be used to control the smart card reset and power pins. Any power supplied to the card must be externally switched as card power supply requirements often exceed source currents possible on this part. As the specific application may accommodate any of the available ISO 7816 class A, B, or C power requirements, be aware of the logic level tolerances and requirements when communicating or powering cards that use different power rails than this part.

36.7.3.1 Smart card set-up procedure

A T = 0 protocol transfer consists of 8-bits of data, an even parity bit, and two guard bits that allow for the receiver of the particular transfer to flag parity errors through the NACK response (see Figure 114). Extra guard bits may be added according to card requirements. If no NACK is sent (provided the interface accepts them in SCICTRL), the next byte may be transmitted immediately after the last guard bit. If the NACK is sent, the transmitter will retry sending the byte until successfully received or until the SCICTRL retry limit has been met.

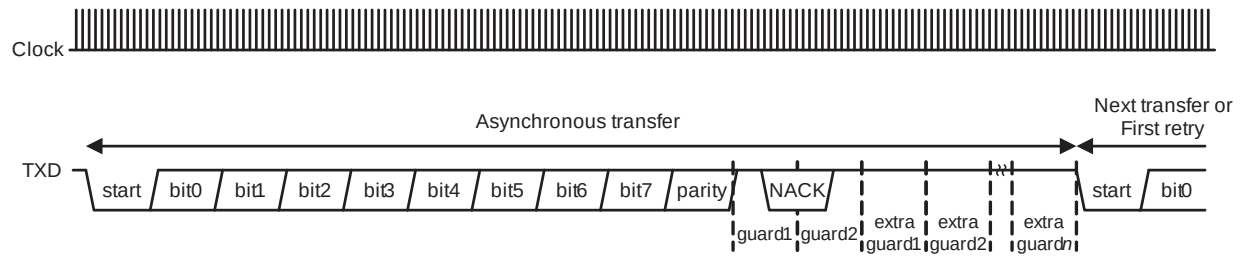


Fig 114. Smart card T = 0 waveform

The smart card must be set up with the following considerations:

1. If necessary, bring the USART out of reset and enable clocking to the peripheral.
2. Setup an available USART TXD pin for the bidirectional transfers.
3. Set up UCLK pin as clock source in the pin configuration register. The default clock requirement for most asynchronous cards is 372 times the bit rate.
4. Configure DLL and DLM for baud rate. It may not be necessary to target a specific standard baud rate but rather to maintain a fraction of the previously mentioned clock rate. For example if the clock rate is set to 4 MHz the baud rate would be 10753. A clock rate of 3.5712 MHz would need a baud rate of 9600. An ISO 7816 PPS exchange may require the baud rate to be changed later.
5. Configure LCR for character size and parity (typically 8-bit and even parity).
6. Configure SCICTRL with the desired NACK response, extra guard bits, and protocol type.
7. Place the GPIO output signals into an inactive state where card power is off, RST is low, and CLK is low and unchanging.

Thereafter, software should monitor card insertion, handle activation, wait for answer to reset as described in ISO7816-3.

37.1 How to read this chapter

The UART1 controller is available on all LPC43xx parts.

37.2 Basic configuration

The UART1 is configured as follows:

- See Table 827 for clocking and power control.
- The UART1 is reset by the UART1_RST (reset #45).
- The UART1 interrupt is connected to slot # 25 in the NVIC.
- For connecting the UART1 receive and transmit lines to the GPDMA, use the DMAMUX register in the CREG block (see Table 41) and enable the GPDMA channel in the DMA Channel Configuration registers (Section 19.6.20).

Table 827. UART1 clocking and power control

	Base clock	Branch clock	Operating frequency
UART1 clock to register interface	BASE_M4_CLK	CLK_M4_UART0	up to 204 MHz
UART1 peripheral clock (PCLK)	BASE_UART1_CLK	CLK_APB0_UART1	up to 204 MHz

37.3 Features

- Full modem control handshaking available.
- Data sizes of 5, 6, 7, and 8 bits.
- Parity generation and checking: odd, even mark, space or none.
- One or two stop bits.
- 16 byte Receive and Transmit FIFOs.
- Built-in baud rate generator, including a fractional rate divider for great versatility.
- Supports DMA for both transmit and receive.
- Auto-baud capability.
- Break generation and detection.
- Multiprocessor addressing mode.
- RS-485 support.

37.4 Pin description

Table 828: UART1 Pin description

Function name	Direction	Description
RXD1	Input	**Serial Input.** Serial receive data.
TXD1	Output	**Serial Output.** Serial transmit data.
CTS1	Input	**Clear To Send.** Active low signal indicates if the external modem is ready to accept transmitted data via TXD1 from the UART1. In normal operation of the modem interface (U1MCR[4] = 0), the complement value of this signal is stored in U1MSR[4]. State change information is stored in U1MSR[0] and is a source for a priority level 4 interrupt, if enabled (U1IER[3] = 1). Clear to send. CTS1 is an asynchronous, active low modem status signal. Its condition can be checked by reading bit 4 (CTS) of the modem status register. Bit 0 (DCTS) of the Modem Status Register (MSR) indicates that CTS1 has changed states since the last read from the MSR. If the modem status interrupt is enabled when CTS1 changes levels and the auto-cts mode is not enabled, an interrupt is generated. CTS1 is also used in the auto-cts mode to control the transmitter.
DCD1	Input	**Data Carrier Detect.** Active low signal indicates if the external modem has established a communication link with the UART1 and data may be exchanged. In normal operation of the modem interface (U1MCR[4]=0), the complement value of this signal is stored in U1MSR[7]. State change information is stored in U1MSR3 and is a source for a priority level 4 interrupt, if enabled (U1IER[3] = 1).
DSR1	Input	**Data Set Ready.** Active low signal indicates if the external modem is ready to establish a communications link with the UART1. In normal operation of the modem interface (U1MCR[4] = 0), the complement value of this signal is stored in U1MSR[5]. State change information is stored in U1MSR[1] and is a source for a priority level 4 interrupt, if enabled (U1IER[3] = 1).
DTR1	Output	Data Terminal Ready. Active low signal indicates that the UART1 is ready to establish connection with external modem. The complement value of this signal is stored in U1MCR[0]. The DTR pin can also be used as an RS-485/EIA-485 output enable signal.
RI1	Input	**Ring Indicator.** Active low signal indicates that a telephone ringing signal has been detected by the modem. In normal operation of the modem interface (U1MCR[4] = 0), the complement value of this signal is stored in U1MSR[6]. State change information is stored in U1MSR[2] and is a source for a priority level 4 interrupt, if enabled (U1IER[3] = 1).
RTS1	Output	**Request To Send.** Active low signal indicates that the UART1 would like to transmit data to the external modem. The complement value of this signal is stored in U1MCR[1]. In auto-rts mode, RTS1 is used to control the transmitter FIFO threshold logic. Request to send. RTS1 is an active low signal informing the modem or data set that the UART is ready to receive data. RTS1 is set to the active (low) level by setting the RTS modem control register bit and is set to the inactive (high) level either as a result of a system reset or during loop-back mode operations or by clearing bit 1 (RTS) of the MCR. In the auto-rts mode, RTS1 is controlled by the transmitter FIFO threshold logic. The RTS pin can also be used as an RS-485/EIA-485 output enable signal.

37.5 Register description

UART1 contains registers organized as shown in Table 829. The Divisor Latch Access Bit (DLAB) is contained in U1LCR[7] and enables access to the Divisor Latches.

Reset value reflects the data stored in used bits only. It does not include the content of reserved bits.

Table 829: Register overview: UART1 (base address 0x4008 2000)

Name	Access	Address offset	Description	Reset value	Reference
RBR	RO	0x000	Receiver Buffer Register. Contains the next received character to be read. (DLAB=0)	NA	Table 830
THR	WO	0x000	Transmit Holding Register. The next character to be transmitted is written here. (DLAB=0)	NA	Table 831
DLL	R/W	0x000	Divisor Latch LSB. Least significant byte of the baud rate divisor value. The full divisor is used to generate a baud rate from the fractional rate divider. (DLAB=1)	0x01	Table 832
DLM	R/W	0x004	Divisor Latch MSB. Most significant byte of the baud rate divisor value. The full divisor is used to generate a baud rate from the fractional rate divider.(DLAB=1)	0x00	Table 833
IER	R/W	0x004	Interrupt Enable Register. Contains individual interrupt enable bits for the 7 potential UART1 interrupts. (DLAB=0)	0x00	Table 834
IIR	RO	0x008	Interrupt ID Register. Identifies which interrupt(s) are pending.	0x01	Table 835
FCR	WO	0x008	FIFO Control Register. Controls UART1 FIFO usage and modes.	0x00	Table 837
LCR	R/W	0x00C	Line Control Register. Contains controls for frame formatting and break generation.	0x00	Table 838
MCR	R/W	0x010	Modem Control Register. Contains controls for flow control handshaking and loopback mode.	0x00	Table 839
LSR	RO	0x014	Line Status Register. Contains flags for transmit and receive status, including line errors.	0x60	Table 841
MSR	RO	0x018	Modem Status Register. Contains handshake signal status flags.	0x00	Table 842
SCR	R/W	0x01C	Scratch Pad Register. 8-bit temporary storage for software.	0x00	Table 843
ACR	R/W	0x020	Auto-baud Control Register. Contains controls for the auto-baud feature.	0x00	Table 844
FDR	R/W	0x028	Fractional Divider Register. Generates a clock input for the baud rate divider.	0x10	Table 845
TER	R/W	0x030	Transmit Enable Register. Turns off UART transmitter for use with software flow control.	0x80	Table 847
RS485CTRL	R/W	0x04C	RS-485/EIA-485 Control. Contains controls to configure various aspects of RS-485/EIA-485 modes.	0x00	Table 848
RS485ADRMATCH	R/W	0x050	RS-485/EIA-485 address match. Contains the address match value for RS-485/EIA-485 mode.	0x00	Table 849
RS485DLY	R/W	0x054	RS-485/EIA-485 direction control delay.	0x00	Table 850
FIFOLVL	RO	0x058	FIFO Level register. Provides the current fill levels of the transmit and receive FIFOs.	0x00	Table 851

37.5.1 UART1 Receiver Buffer Register (when DLAB = 0)

The U1RBR is the top byte of the UART1 RX FIFO. The top byte of the RX FIFO contains the oldest character received and can be read via the bus interface. The LSB (bit 0) represents the “oldest” received data bit. If the character received is less than 8 bits, the unused MSBs are padded with zeroes.

The Divisor Latch Access Bit (DLAB) in U1LCR must be zero in order to access the U1RBR. The U1RBR is always read-only.

Since PE, FE and BI bits correspond to the byte sitting on the top of the RBR FIFO (i.e. the one that will be read in the next read from the RBR), the right approach for fetching the valid pair of received byte and its status bits is first to read the content of the U1LSR register, and then to read a byte from the U1RBR.

Table 830: UART1 Receiver Buffer Register when DLAB = 0 (RBR - address 0x4008 2000) bit description

Bit	Symbol	Description	Reset value
7:0	RBR	Receiver Buffer. Contains the oldest received byte in the UART1 RX FIFO.	undefined
31:8	-	Reserved, the value read from a reserved bit is not defined.	NA

37.5.2 UART1 Transmitter Holding Register (when DLAB = 0)

The write-only U1THR is the top byte of the UART1 TX FIFO. The top byte is the newest character in the TX FIFO and can be written via the bus interface. The LSB represents the first bit to transmit.

The Divisor Latch Access Bit (DLAB) in U1LCR must be zero in order to access the U1THR. The U1THR is write-only.

Table 831: UART1 Transmitter Holding Register when DLAB = 0 (THR - address 0x4008 2000) bit description

Bit	Symbol	Description	Reset value
7:0	THR	Transmit Holding Register. Writing to the UART1 Transmit Holding Register causes the data to be stored in the UART1 transmit FIFO. The byte will be sent when it reaches the bottom of the FIFO and the transmitter is available.	NA
31:8	-	Reserved, user software should not write ones to reserved bits.	NA

37.5.3 UART1 Divisor Latch LSB and MSB Registers (when DLAB = 1)

The UART1 Divisor Latch is part of the UART1 Baud Rate Generator and holds the value used, along with the Fractional Divider, to divide the APB clock (PCLK) in order to produce the baud rate clock, which must be 16x the desired baud rate. The U1DLL and U1DLM registers together form a 16-bit divisor where U1DLL contains the lower 8 bits of the divisor and U1DLM contains the higher 8 bits of the divisor. A 0x0000 value is treated like a 0x0001 value as division by zero is not allowed.The Divisor Latch Access Bit (DLAB) in U1LCR must be one in order to access the UART1 Divisor Latches. Details on how to select the right value for U1DLL and U1DLM can be found later in this chapter, see Section 37.5.16.

Table 832: UART1 Divisor Latch LSB Register when DLAB = 1 (DLL - address 0x4008 2000) bit description

Bit	Symbol	Description	Reset value
7:0	DLLSB	Divisor Latch LSB. The UART1 Divisor Latch LSB Register, along with the U1DLM register, determines the baud rate of the UART1.	0x01
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Table 833: UART1 Divisor Latch MSB Register when DLAB = 1 (DLM - address 0x4008 2004) bit description

Bit	Symbol	Description	Reset value
7:0	DLMSB	Divisor Latch MSB. The UART1 Divisor Latch MSB Register, along with the U1DLL register, determines the baud rate of the UART1.	0x00
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.4 UART1 Interrupt Enable Register (when DLAB = 0)

The U1IER is used to enable the four UART1 interrupt sources.

Table 834: UART1 Interrupt Enable Register when DLAB = 0 (IER - address 0x4008 2004) bit description

Bit	Symbol	Value	Description	Reset value
0	RBRIE		RBR Interrupt Enable. Enables the Receive Data Available interrupt for UART1. It also controls the Character Receive Time-out interrupt.	0
		0	Disable the RDA interrupts.	
		1	Enable the RDA interrupts.	
1	THREIE		THRE Interrupt Enable. Enables the THRE interrupt for UART1. The status of this interrupt can be read from U1LSR[5].	0
		0	Disable the THRE interrupts.	
		1	Enable the THRE interrupts.	
2	RXIE		RX Line Interrupt Enable. Enables the UART1 RX line status interrupts. The status of this interrupt can be read from U1LSR[4:1].	0
		0	Disable the RX line status interrupts.	
		1	Enable the RX line status interrupts.	
3	MSIE		Modem Status Interrupt Enable. Enables the modem interrupt. The status of this interrupt can be read from U1MSR[3:0].	0
		0	Disable the modem interrupt.	
		1	Enable the modem interrupt.	
6:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Table 834: UART1 Interrupt Enable Register when DLAB = 0 (IER - address 0x4008 2004) bit description

Bit	Symbol	Value	Description	Reset value
7	CTSIE		CTS Interrupt Enable. If auto-cts mode is enabled this bit enables/disables the modem status interrupt generation on a CTS1 signal transition. If auto-cts mode is disabled a CTS1 transition will generate an interrupt if Modem Status Interrupt Enable (U1IER[3]) is set. In normal operation a CTS1 signal transition will generate a Modem Status Interrupt unless the interrupt has been disabled by clearing the U1IER[3] bit in the U1IER register. In auto-cts mode a transition on the CTS1 bit will trigger an interrupt only if both the U1IER[3] and U1IER[7] bits are set.	0
		0	Disable the CTS interrupt.	
		1	Enable the CTS interrupt.	
8	ABEOIE		Enables the end of auto-baud interrupt.	0
		0	Disable end of auto-baud Interrupt.	
		1	Enable end of auto-baud Interrupt.	
9	ABTOIE		Enables the auto-baud time-out interrupt.	0
		0	Disable auto-baud time-out Interrupt.	
		1	Enable auto-baud time-out Interrupt.	
31:10	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.5 UART1 Interrupt Identification Register

The U1IIR provides a status code that denotes the priority and source of a pending interrupt. The interrupts are frozen during an U1IIR access. If an interrupt occurs during an U1IIR access, the interrupt is recorded for the next U1IIR access.

Table 835: UART1 Interrupt Identification Register (IIR - address 0x4008 2008) bit description

Bit	Symbol	Value	Description	Reset value
0	INTSTATUS		Interrupt status. Note that U1IIR[0] is active low. The pending interrupt can be determined by evaluating U1IIR[3:1].	1
		0	At least one interrupt is pending.	
		1	No interrupt is pending.	
3:1	INTID		Interrupt identification. U1IER[3:1] identifies an interrupt corresponding to the UART1 Rx or TX FIFO. All other combinations of U1IER[3:1] not listed below are reserved (100,101,111).	0
		0x3	1 - Receive Line Status (RLS).	
		0x2	2a - Receive Data Available (RDA).	
		0x6	2b - Character Time-out Indicator (CTI).	
		0x1	3 - THRE Interrupt.	
		0x0	4 - Modem Interrupt.	
5:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	FIFOENABLE		Copies of U1FCR[0].	0

Table 835: UART1 Interrupt Identification Register (IIR - address 0x4008 2008) bit description

Bit	Symbol	Value	Description	Reset value
8	ABEOINT		End of auto-baud interrupt. True if auto-baud has finished successfully and interrupt is enabled.	0
9	ABTOINT		Auto-baud time-out interrupt. True if auto-baud has timed out and interrupt is enabled.	0
31:10	-		Reserved, the value read from a reserved bit is not defined.	NA

Bit U1IIR[9:8] are set by the auto-baud function and signal a time-out or end of auto-baud condition. The auto-baud interrupt conditions are cleared by setting the corresponding Clear bits in the Auto-baud Control Register.

If the IntStatus bit is 1 no interrupt is pending and the IntId bits will be zero. If the IntStatus is 0, a non auto-baud interrupt is pending in which case the IntId bits identify the type of interrupt and handling as described in Table 836. Given the status of U1IIR[3:0], an interrupt handler routine can determine the cause of the interrupt and how to clear the active interrupt. The U1IIR must be read in order to clear the interrupt prior to exiting the Interrupt Service Routine.

The UART1 RLS interrupt (U1IIR[3:1] = 011) is the highest priority interrupt and is set whenever any one of four error conditions occur on the UART1RX input: overrun error (OE), parity error (PE), framing error (FE) and break interrupt (BI). The UART1 Rx error condition that set the interrupt can be observed via U1LSR[4:1]. The interrupt is cleared upon an U1LSR read.

The UART1 RDA interrupt (U1IIR[3:1] = 010) shares the second level priority with the CTI interrupt (U1IIR[3:1] = 110). The RDA is activated when the UART1 Rx FIFO reaches the trigger level defined in U1FCR7:6 and is reset when the UART1 Rx FIFO depth falls below the trigger level. When the RDA interrupt goes active, the CPU can read a block of data defined by the trigger level.

The CTI interrupt (U1IIR[3:1] = 110) is a second level interrupt and is set when the UART1 Rx FIFO contains at least one character and no UART1 Rx FIFO activity has occurred in 3.5 to 4.5 character times. Any UART1 Rx FIFO activity (read or write of UART1 RSR) will clear the interrupt. This interrupt is intended to flush the UART1 RBR after a message has been received that is not a multiple of the trigger level size. For example, if a peripheral wished to send a 105 character message and the trigger level was 10 characters, the CPU would receive 10 RDA interrupts resulting in the transfer of 100 characters and 1 to 5 CTI interrupts (depending on the service routine) resulting in the transfer of the remaining 5 characters.

Table 836: UART1 Interrupt Handling

U1IIR[3:0] value[1]	Priority	Interrupt Type	Interrupt Source	Interrupt Reset
0001	-	None	None	-
0110	Highest	RX Line Status / Error	OE[2] or PE[2] or FE[2] or BI[2]	U1LSR Read[2]
0100	Second	RX Data Available	Rx data available or trigger level reached in FIFO (U1FCR0=1)	U1RBR Read[3] or UART1 FIFO drops below trigger level

Table 836: UART1 Interrupt Handling

U1IIR[3:0] value[1]	Priority	Interrupt Type	Interrupt Source	Interrupt Reset
1100	Second	Character Time-out indication	Minimum of one character in the RX FIFO and no character input or removed during a time period depending on how many characters are in FIFO and what the trigger level is set at (3.5 to 4.5 character times). The exact time will be: [(word length) × 7 - 2] × 8 + [(trigger level - number of characters) × 8 + 1] RCLKs	U1RBR Read[3]
0010	Third	THRE	THRE[2]	U1IIR Read[4] (if source of interrupt) or THR write
0000	Fourth	Modem Status	CTS or DSR or RI or DCD	MSR Read

[1] Values "0000", “0011”, “0101”, “0111”, “1000”, “1001”, “1010”, “1011”,”1101”,”1110”,”1111” are reserved.

[2] For details see Section 37.5.10 “UART1 Line Status Register”

[3] For details see Section 37.5.1 “UART1 Receiver Buffer Register (when DLAB = 0)”

[4] For details see Section 37.5.5 “UART1 Interrupt Identification Register” and Section 37.5.2 “UART1 Transmitter Holding Register (when DLAB = 0)”

The UART1 THRE interrupt (U1IIR[3:1] = 001) is a third level interrupt and is activated when the UART1 THR FIFO is empty provided certain initialization conditions have been met. These initialization conditions are intended to give the UART1 THR FIFO a chance to fill up with data to eliminate many THRE interrupts from occurring at system start-up. The initialization conditions implement a one character delay minus the stop bit whenever THRE = 1 and there have not been at least two characters in the U1THR at one time since the last THRE = 1 event. This delay is provided to give the CPU time to write data to U1THR without a THRE interrupt to decode and service. A THRE interrupt is set immediately if the UART1 THR FIFO has held two or more characters at one time and currently, the U1THR is empty. The THRE interrupt is reset when a U1THR write occurs or a read of the U1IIR occurs and the THRE is the highest interrupt (U1IIR[3:1] = 001).

It is the lowest priority interrupt and is activated whenever there is any state change on modem inputs pins, DCD, DSR or CTS. In addition, a low to high transition on modem input RI will generate a modem interrupt. The source of the modem interrupt can be determined by examining U1MSR[3:0]. A U1MSR read will clear the modem interrupt.

37.5.6 UART1 FIFO Control Register

The write-only U1FCR controls the operation of the UART1 RX and TX FIFOs.

Table 837: UART1 FIFO Control Register (FCR - address 0x4008 2008) bit description

Bit	Symbol	Value	Description	Reset value
0	FIFOEN		FIFO enable.	0
		0	Must not be used in the application.	
		1	Active high enable for both UART1 Rx and TX FIFOs and U1FCR[7:1] access. This bit must be set for proper UART1 operation. Any transition on this bit will automatically clear the UART1 FIFOs.	

Table 837: UART1 FIFO Control Register (FCR - address 0x4008 2008) bit description

Bit	Symbol	Value	Description	Reset value
1	RXFIFORES		RX FIFO Reset.	0
		0	No impact on either of UART1 FIFOs.	
		1	Writing a logic 1 to U1FCR[1] will clear all bytes in UART1 Rx FIFO, reset the pointer logic. This bit is self-clearing.	
2	TXFIFORES		TX FIFO Reset.	0
		0	No impact on either of UART1 FIFOs.	
		1	Writing a logic 1 to U1FCR[2] will clear all bytes in UART1 TX FIFO, reset the pointer logic. This bit is self-clearing.	
3	DMAMODE		DMA Mode Select. When the FIFO enable bit (bit 0 of this register) is set, this bit selects the DMA mode. See Section 37.5.6.1.	0
5:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:6	RXTRIGLVL		RX Trigger Level. These two bits determine how many receiver UART1 FIFO characters must be written before an interrupt is activated.	0
		0x0	Trigger level 0 (1 character or 0x01).	
		0x1	Trigger level 1 (4 characters or 0x04).	
		0x2	Trigger level 2 (8 characters or 0x08).	
		0x3	Trigger level 3 (14 characters or 0x0E).	
31:8	-		Reserved, user software should not write ones to reserved bits.	NA

37.5.6.1 DMA Operation

The user can optionally operate the UART transmit and/or receive using DMA. The DMA mode is determined by the DMA Mode Select bit in the FCR register. Note that for DMA operation as for any operation of the UART, the FIFOs must be enabled via the FIFO Enable bit in the FCR register.

UART receiver DMA

In DMA mode, the receiver DMA request is asserted on the event of the receiver FIFO level becoming equal to or greater than trigger level, or if a character time-out occurs. See the description of the RX Trigger Level above. The receiver DMA request is cleared by the DMA controller.

UART transmitter DMA

In DMA mode, the transmitter DMA request is asserted on the event of the transmitter FIFO transitioning to not full. The transmitter DMA request is cleared by the DMA controller.

37.5.7 UART1 Line Control Register

The U1LCR determines the format of the data character that is to be transmitted or received.

Table 838: UART1 Line Control Register (LCR - address 0x4008 200C) bit description

Bit	Symbol	Value	Description	Reset value
1:0	WLS		Word Length Select.	0
		0x0	5-bit character length.	
		0x1	6-bit character length.	
		0x2	7-bit character length.	
		0x3	8-bit character length.	
2	SBS		Stop Bit Select.	0
		0	1 stop bit.	
		1	2 stop bits (1.5 if U1LCR[1:0]=00).	
3	PE		Parity Enable.	0
		0	Disable parity generation and checking.	
		1	Enable parity generation and checking.	
5:4	PS		Parity Select.	0
		0x0	Odd parity. Number of 1s in the transmitted character and the attached parity bit will be odd.	
		0x1	Even Parity. Number of 1s in the transmitted character and the attached parity bit will be even.	
		0x2	Forced "1" stick parity.	
		0x3	Forced "0" stick parity.	
6	BC		Break Control.	0
		0	Disable break transmission.	
		1	Enable break transmission. Output pin UART1 TXD is forced to logic 0 when U1LCR[6] is active high.	
7	DLAB		Divisor Latch Access Bit (DLAB)	0
		0	Disable access to Divisor Latches.	
		1	Enable access to Divisor Latches.	
31:8	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.8 UART1 Modem Control Register

The U1MCR enables the modem loopback mode and controls the modem output signals.

Table 839: UART1 Modem Control Register (MCR - address 0x4008 2010) bit description

Bit	Symbol	Value	Description	Reset value
0	DTRCTRL	-	DTR Control. Source for modem output pin, DTR. This bit reads as 0 when modem loopback mode is active.	0
1	RTSCTRL	-	RTS Control. Source for modem output pin RTS. This bit reads as 0 when modem loopback mode is active.	0
3:2	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

Table 839: UART1 Modem Control Register (MCR - address 0x4008 2010) bit description

Bit	Symbol	Value	Description	Reset value
4	LMS		Loopback Mode Select. The modem loopback mode provides a mechanism to perform diagnostic loopback testing. Serial data from the transmitter is connected internally to serial input of the receiver. Input pin, RXD1, has no effect on loopback and output pin, TXD1 is held in marking state. The 4 modem inputs (CTS, DSR, RI and DCD) are disconnected externally. Externally, the modem outputs (RTS, DTR) are set inactive. Internally, the 4 modem outputs are connected to the 4 modem inputs. As a result of these connections, the upper 4 bits of the U1MSR will be driven by the lower 4 bits of the U1MCR rather than the 4 modem inputs in normal mode. This permits modem status interrupts to be generated in loopback mode by writing the lower 4 bits of U1MCR.	0
		0	Disable modem loopback mode.	
		1	Enable modem loopback mode.	
5	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
6	RTSEN		RTS enable.	0
		0	Disable auto-rts flow control.	
		1	Enable auto-rts flow control.	
7	CTSEN		CTS enable.	0
		0	Disable auto-cts flow control.	
		1	Enable auto-cts flow control.	
31:8	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.9 Auto-flow control

If auto-RTS mode is enabled the UART1‘s receiver FIFO hardware controls the RTS1 output of the UART1. If the auto-CTS mode is enabled the UART1‘s U1TSR hardware will only start transmitting if the CTS1 input signal is asserted.

37.5.9.1 Auto-RTS

The auto-RTS function is enabled by setting the RTSen bit. Auto-RTS data flow control originates in the U1RBR module and is linked to the programmed receiver FIFO trigger level. If auto-RTS is enabled, the data-flow is controlled as follows:

When the receiver FIFO level reaches the programmed trigger level, RTS1 is de-asserted (to a high value). It is possible that the sending UART sends an additional byte after the trigger level is reached (assuming the sending UART has another byte to send) because it might not recognize the de-assertion of RTS1 until after it has begun sending the additional byte. RTS1 is automatically reasserted (to a low value) once the receiver FIFO has reached the previous trigger level. The re-assertion of RTS1 signals to the sending UART to continue transmitting data.

If Auto-RTS mode is disabled, the RTSen bit controls the RTS1 output of the UART1. If Auto-RTS mode is enabled, hardware controls the RTS1 output, and the actual value of RTS1 will be copied in the RTS Control bit of the UART1. As long as Auto-RTS is enabled, the value of the RTS Control bit is read-only for software.

Example: Suppose the UART1 operating in '550 mode has trigger level in U1FCR set to 0x2 then if Auto-RTS is enabled the UART1 will de-assert the RTS1 output as soon as the receive FIFO contains 8 bytes (Table 837 on page 907). The RTS1 output will be reasserted as soon as the receive FIFO hits the previous trigger level: 4 bytes.

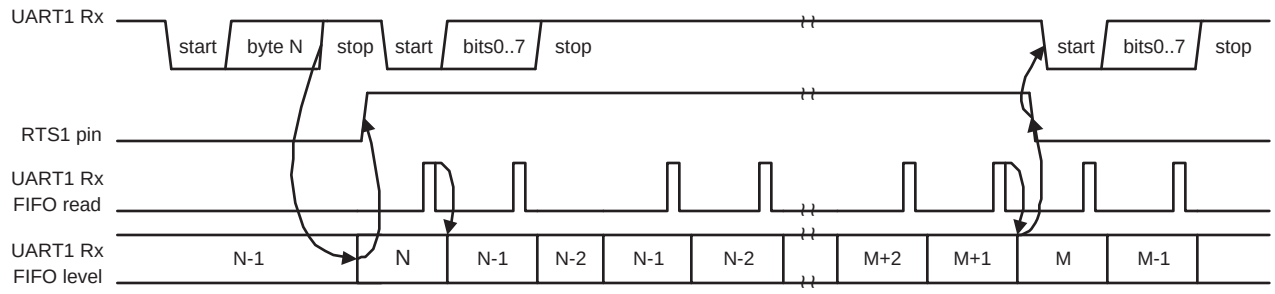


Fig 115.Auto-RTS Functional Timing

37.5.9.2 Auto-CTS

The Auto-CTS function is enabled by setting the CTSen bit. If Auto-CTS is enabled the transmitter circuitry in the U1TSR module checks CTS1 input before sending the next data byte. When CTS1 is active (low), the transmitter sends the next byte. To stop the transmitter from sending the following byte, CTS1 must be released before the middle of the last stop bit that is currently being sent. In Auto-CTS mode a change of the CTS1 signal does not trigger a modem status interrupt unless the CTS Interrupt Enable bit is set, Delta CTS bit in the U1MSR will be set though. Table 840 lists the conditions for generating a Modem Status interrupt.

Table 840: Modem status interrupt generation

Enable Modem Status Interrupt (U1ER[3])	CTSen (U1MCR[7])	CTS Interrupt Enable (U1IER[7])	Delta CTS (U1MSR[0])	Delta DCD or Trailing Edge RI or Delta DSR (U1MSR[3] or U1MSR[2] or U1MSR[1])	Modem Status Interrupt
0	x	x	x	x	No
1	0	x	0	0	No
1	0	x	1	x	Yes
1	0	x	x	1	Yes
1	1	0	x	0	No
1	1	0	x	1	Yes
1	1	1	0	0	No
1	1	1	1	x	Yes
1	1	1	x	1	Yes

The auto-CTS function reduces interrupts to the host system. When flow control is enabled, a CTS1 state change does not trigger host interrupts because the device automatically controls its own transmitter. Without Auto-CTS, the transmitter sends any data present in the transmit FIFO and a receiver overrun error can result. Figure 116 illustrates the Auto-CTS functional timing.

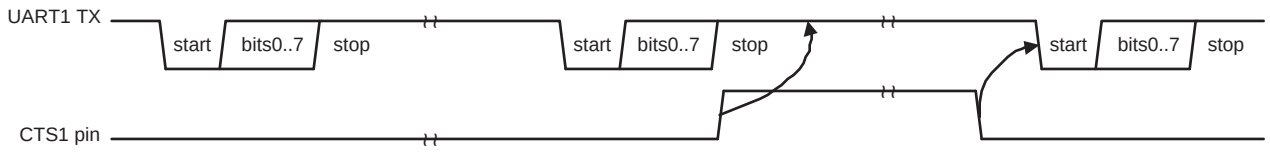


Fig 116.Auto-CTS Functional Timing

While starting transmission of the initial character the CTS1 signal is asserted. Transmission will stall as soon as the pending transmission has completed. The UART will continue transmitting a 1 bit as long as CTS1 is de-asserted (high). As soon as CTS1 gets de-asserted transmission resumes and a start bit is sent followed by the data bits of the next character.

37.5.10 UART1 Line Status Register

The U1LSR is a read-only register that provides status information on the UART1 TX and RX blocks.

Table 841: UART1 Line Status Register (LSR - address 0x4008 2014) bit description

Bit	Symbol	Value	Description	Reset value
0	RDR		Receiver Data Ready. U1LSR[0] is set when the U1RBR holds an unread character and is cleared when the UART1 RBR FIFO is empty.	0
		0	The UART1 receiver FIFO is empty.	
		1	The UART1 receiver FIFO is not empty.	
1	OE		Overrun Error. The overrun error condition is set as soon as it occurs. An U1LSR read clears U1LSR[1]. U1LSR[1] is set when UART1 RSR has a new character assembled and the UART1 RBR FIFO is full. In this case, the UART1 RBR FIFO will not be overwritten and the character in the UART1 RSR will be lost.	0
		0	Overrun error status is inactive.	
		1	Overrun error status is active.	
2	PE		Parity Error. When the parity bit of a received character is in the wrong state, a parity error occurs. An U1LSR read clears U1LSR[2]. Time of parity error detection is dependent on U1FCR[0]. **Note:** A parity error is associated with the character at the top of the UART1 RBR FIFO.	0
		0	Parity error status is inactive.	
		1	Parity error status is active.	

Table 841: UART1 Line Status Register (LSR - address 0x4008 2014) bit description

Bit	Symbol	Value	Description	Reset value
3	FE		Framing Error. When the stop bit of a received character is a logic 0, a framing error occurs. An U1LSR read clears U1LSR[3]. The time of the framing error detection is dependent on U1FCR0. Upon detection of a framing error, the RX will attempt to resynchronize to the data and assume that the bad stop bit is actually an early start bit. However, it cannot be assumed that the next received byte will be correct even if there is no Framing Error. **Note:** A framing error is associated with the character at the top of the UART1 RBR FIFO.	0
		0	Framing error status is inactive.	
		1	Framing error status is active.	
4	BI		Break Interrupt. When RXD1 is held in the spacing state (all zeroes) for one full character transmission (start, data, parity, stop), a break interrupt occurs. Once the break condition has been detected, the receiver goes idle until RXD1 goes to marking state (all ones). An U1LSR read clears this status bit. The time of break detection is dependent on U1FCR[0]. **Note:** The break interrupt is associated with the character at the top of the UART1 RBR FIFO.	0
		0	Break interrupt status is inactive.	
		1	Break interrupt status is active.	
5	THRE		Transmitter Holding Register Empty. THRE is set immediately upon detection of an empty UART1 THR and is cleared on a U1THR write.	1
		0	U1THR contains valid data.	
		1	U1THR is empty.	
6	TEMT		Transmitter Empty. TEMT is set when both U1THR and U1TSR are empty; TEMT is cleared when either the U1TSR or the U1THR contain valid data.	1
		0	U1THR and/or the U1TSR contains valid data.	
		1	U1THR and the U1TSR are empty.	
7	RXFE		Error in RX FIFO. U1LSR[7] is set when a character with a RX error such as framing error, parity error or break interrupt, is loaded into the U1RBR. This bit is cleared when the U1LSR register is read and there are no subsequent errors in the UART1 FIFO.	0
		0	U1RBR contains no UART1 RX errors or U1FCR[0]=0.	
		1	UART1 RBR contains at least one UART1 RX error.	
31:8	-		Reserved, the value read from a reserved bit is not defined.	NA

37.5.11 UART1 Modem Status Register

The U1MSR is a read-only register that provides status information on the modem input signals. U1MSR[3:0] is cleared on U1MSR read. Note that modem signals have no direct effect on UART1 operation, they facilitate software implementation of modem signal operations.

Table 842: UART1 Modem Status Register (MSR - address 0x4008 2018) bit description

Bit	Symbol	Value	Description	Reset value
0	DCTS		Delta CTS. Set upon state change of input CTS. Cleared on an U1MSR read.	0
		0	No change detected on modem input, CTS.	
		1	State change detected on modem input, CTS.	
1	DDSR		Delta DSR. Set upon state change of input DSR. Cleared on an U1MSR read.	0
		0	No change detected on modem input, DSR.	
		1	State change detected on modem input, DSR.	
2	TERI		Trailing Edge RI. Set upon low to high transition of input RI. Cleared on an U1MSR read.	0
		0	No change detected on modem input, RI.	
		1	Low-to-high transition detected on RI.	
3	DDCD		Delta DCD. Set upon state change of input DCD. Cleared on an U1MSR read.	0
		0	No change detected on modem input, DCD.	
		1	State change detected on modem input, DCD.	
4	CTS	-	Clear To Send State. Complement of input signal CTS. This bit is connected to U1MCR[1] in modem loopback mode.	0
5	DSR	-	Data Set Ready State. Complement of input signal DSR. This bit is connected to U1MCR[0] in modem loopback mode.	0
6	RI	-	Ring Indicator State. Complement of input RI. This bit is connected to U1MCR[2] in modem loopback mode.	0
7	DCD	-	Data Carrier Detect State. Complement of input DCD. This bit is connected to U1MCR[3] in modem loopback mode.	0
31:8	-	-	Reserved, the value read from a reserved bit is not defined.	NA

37.5.12 UART1 Scratch Pad Register

The U1SCR has no effect on the UART1 operation. This register can be written and/or read at user's discretion. There is no provision in the interrupt interface that would indicate to the host that a read or write of the U1SCR has occurred.

Table 843: UART1 Scratch Pad Register (SCR - address 0x4008 2014) bit description

Bit	Symbol	Description	Reset value
7:0	Pad	Scratch pad. A readable, writable byte.	0x00
31:8	-	Reserved, the value read from a reserved bit is not defined.	NA

37.5.13 UART1 Auto-baud Control Register

The UART1 Auto-baud Control Register (U1ACR) controls the process of measuring the incoming clock/data rate for the baud rate generation and can be read and written at user's discretion.

Table 844: Autobaud Control Register (ACR - address 0x4008 2020) bit description

Bit	Symbol	Value	Description	Reset value
0	START		Auto-baud start bit. This bit is automatically cleared after auto-baud completion.	0
		0	Auto-baud stop (auto-baud is not running).	
		1	Auto-baud start (auto-baud is running). Auto-baud run bit. This bit is automatically cleared after auto-baud completion.	
1	MODE		Auto-baud mode select bit.	0
		0	Mode 0.	
		1	Mode 1.	
2	AUTORESTART		Auto-baud restart bit.	0
		0	No restart	
		1	Restart in case of time-out (counter restarts at next UART1 Rx falling edge)	
7:3	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
8	ABEOINTCLR		End of auto-baud interrupt clear bit (write-only).	0
		0	Writing a 0 has no impact.	
		1	Writing a 1 will clear the corresponding interrupt in the U1IIR.	
9	ABTOINTCLR		Auto-baud time-out interrupt clear bit (write-only).	0
		0	Writing a 0 has no impact.	
		1	Writing a 1 will clear the corresponding interrupt in the U1IIR.	
31:10	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

37.5.14 Auto-baud

The UART1 auto-baud function can be used to measure the incoming baud-rate based on the “AT” protocol (Hayes command). If enabled the auto-baud feature will measure the bit time of the receive data stream and set the divisor latch registers U1DLM and U1DLL accordingly.

Remark: the fractional rate divider is not connected during auto-baud operations, and therefore should not be used when the auto-baud feature is needed.

Auto-baud is started by setting the U1ACR Start bit. Auto-baud can be stopped by clearing the U1ACR Start bit. The Start bit will clear once auto-baud has finished and reading the bit will return the status of auto-baud (pending/finished).

Two auto-baud measuring modes are available which can be selected by the U1ACR Mode bit. In mode 0 the baud-rate is measured on two subsequent falling edges of the UART1 Rx pin (the falling edge of the start bit and the falling edge of the least significant bit). In mode 1 the baud-rate is measured between the falling edge and the subsequent rising edge of the UART1 Rx pin (the length of the start bit).

The U1ACR AutoRestart bit can be used to automatically restart baud-rate measurement if a time-out occurs (the rate measurement counter overflows). If this bit is set the rate measurement will restart at the next falling edge of the UART1 Rx pin.

The auto-baud function can generate two interrupts.

- The U1IIR ABTOInt interrupt will get set if the interrupt is enabled (U1IER ABToIntEn is set and the auto-baud rate measurement counter overflows).
- The U1IIR ABEOInt interrupt will get set if the interrupt is enabled (U1IER ABEOIntEn is set and the auto-baud has completed successfully).

The auto-baud interrupts have to be cleared by setting the corresponding U1ACR ABTOIntClr and ABEOIntEn bits.

Typically the fractional baud-rate generator is disabled (DIVADDVAL = 0) during auto-baud. However, if the fractional baud-rate generator is enabled (DIVADDVAL > 0), it is going to impact the measuring of UART1 Rx pin baud-rate, but the value of the U1FDR register is not going to be modified after rate measurement. Also, when auto-baud is used, any write to U1DLM and U1DLL registers should be done before U1ACR register write. The minimum and the maximum baud rates supported by UART1 are function of pclk, number of data bits, stop bits and parity bits.

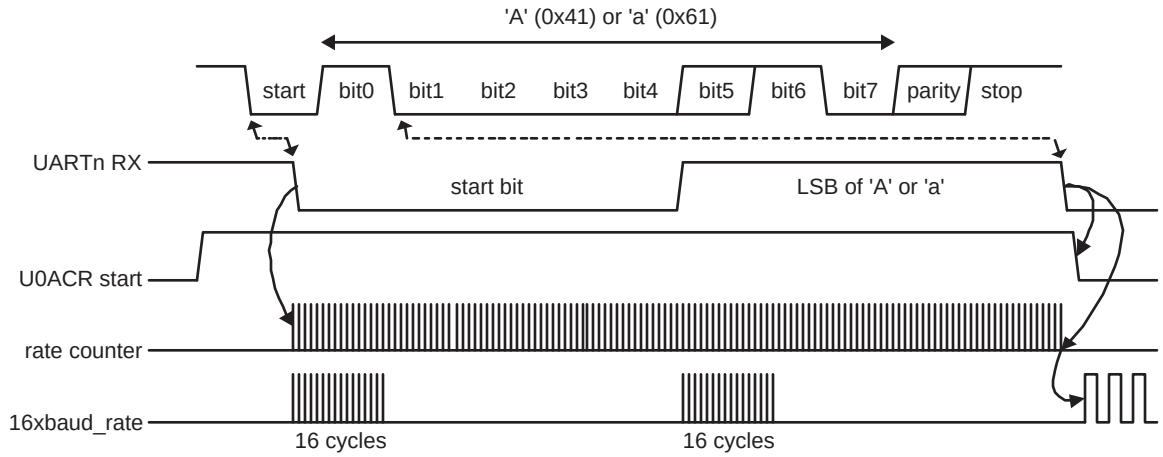
(8)

$$ratemin = \frac{2 \times PCLK}{16 \times 2^{15}} \leq UART1_{baudrate} \leq \frac{PCLK}{16 \times (2 + databits + paritybits + stopbits)} = ratemax$$

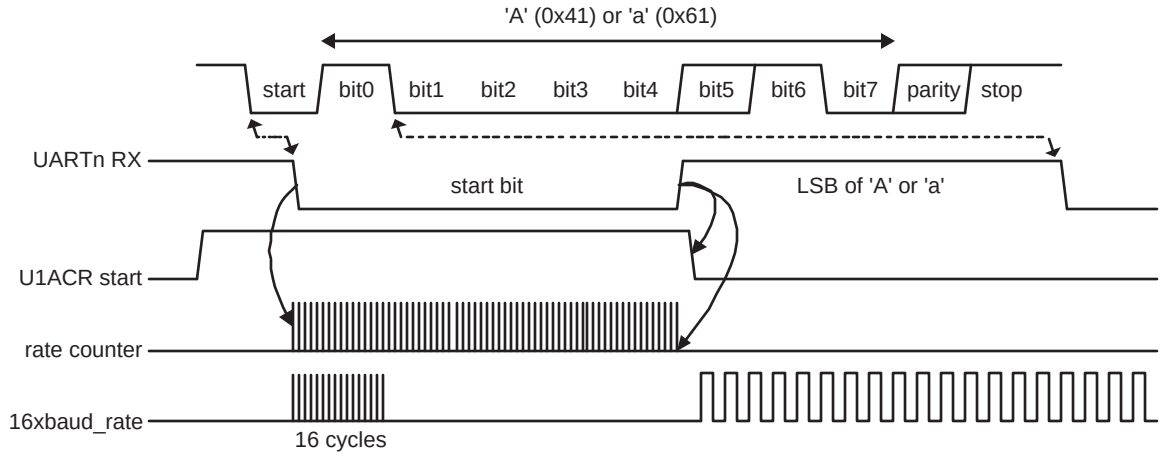
37.5.15 Auto-baud modes

When the software is expecting an “AT” command, it configures the UART1 with the expected character format and sets the U1ACR Start bit. The initial values in the divisor latches U1DLM and U1DLM don‘t care. Because of the “A” or “a” ASCII coding (”A" = 0x41, “a” = 0x61), the UART1 Rx pin sensed start bit and the LSB of the expected character are delimited by two falling edges. When the U1ACR Start bit is set, the auto-baud protocol will execute the following phases:

1. On U1ACR Start bit setting, the baud-rate measurement counter is reset and the UART1 U1RSR is reset. The U1RSR baud rate is switch to the highest rate.
2. A falling edge on UART1 Rx pin triggers the beginning of the start bit. The rate measuring counter will start counting pclk cycles optionally pre-scaled by the fractional baud-rate generator.
3. During the receipt of the start bit, 16 pulses are generated on the RSR baud input with the frequency of the (fractional baud-rate pre-scaled) UART1 input clock, guaranteeing the start bit is stored in the U1RSR.
4. During the receipt of the start bit (and the character LSB for mode = 0) the rate counter will continue incrementing with the pre-scaled UART1 input clock (pclk).
5. If Mode = 0 then the rate counter will stop on next falling edge of the UART1 Rx pin. If Mode = 1 then the rate counter will stop on the next rising edge of the UART1 Rx pin.
6. The rate counter is loaded into U1DLM/U1DLL and the baud-rate will be switched to normal operation. After setting the U1DLM/U1DLL the end of auto-baud interrupt U1IIR ABEOInt will be set, if enabled. The U1RSR will now continue receiving the remaining bits of the “A/a” character.



a. Mode 0 (start bit and LSB are used for auto-baud)



b. Mode 1 (only start bit is used for auto-baud)

Fig 117.Auto-baud a) mode 0 and b) mode 1 waveform

37.5.16 UART1 Fractional Divider Register

The UART1 Fractional Divider Register (U1FDR) controls the clock pre-scaler for the baud rate generation and can be read and written at the user's discretion. This pre-scaler takes the APB clock and generates an output clock according to the specified fractional requirements.

Important: If the fractional divider is active (DIVADDVAL > 0) and DLM = 0, the value of the DLL register must be greater than 2.

Table 845: UART1 Fractional Divider Register (FDR - address 0x4008 2028) bit description

Bit	Function	Description	Reset value
3:0	DIVADDVAL	Baud-rate generation pre-scaler divisor value. If this field is 0, fractional baud-rate generator will not impact the UARTn baudrate.	0
7:4	MULVAL	Baud-rate pre-scaler multiplier value. This field must be greater or equal 1 for UARTn to operate properly, regardless of whether the fractional baud-rate generator is used or not.	1
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0

This register controls the clock pre-scaler for the baud rate generation. The reset value of the register keeps the fractional capabilities of UART1 disabled making sure that UART1 is fully software and hardware compatible with UARTs not equipped with this feature.

UART1 baud rate can be calculated as (n = 1):

(9)

$$UART1_{baudrate} = \frac{PCLK}{16 \times (256 \times U1DLM + U1DLL) \times \left(1 + \frac{DivAddVal}{MulVal}\right)}$$

Where PCLK is the peripheral clock, U1DLM and U1DLL are the standard UART1 baud rate divider registers, and DIVADDVAL and MULVAL are UART1 fractional baud rate generator specific parameters.

The value of MULVAL and DIVADDVAL should comply to the following conditions:

1. $1 \leq MULVAL \leq 15$
2. $0 \leq DIVADDVAL \leq 14$
3. $DIVADDVAL < MULVAL$

The value of the U1FDR should not be modified while transmitting/receiving data or data may be lost or corrupted.

If the U1FDR register value does not comply to these two requests, then the fractional divider output is undefined. If DIVADDVAL is zero then the fractional divider is disabled, and the clock will not be divided.

37.5.16.1 Baud rate calculation

UART1 can operate with or without using the Fractional Divider. In real-life applications it is likely that the desired baud rate can be achieved using several different Fractional Divider settings. The following algorithm illustrates one way of finding a set of DLM, DLL, MULVAL, and DIVADDVAL values. Such set of parameters yields a baud rate with a relative error of less than 1.1% from the desired one.

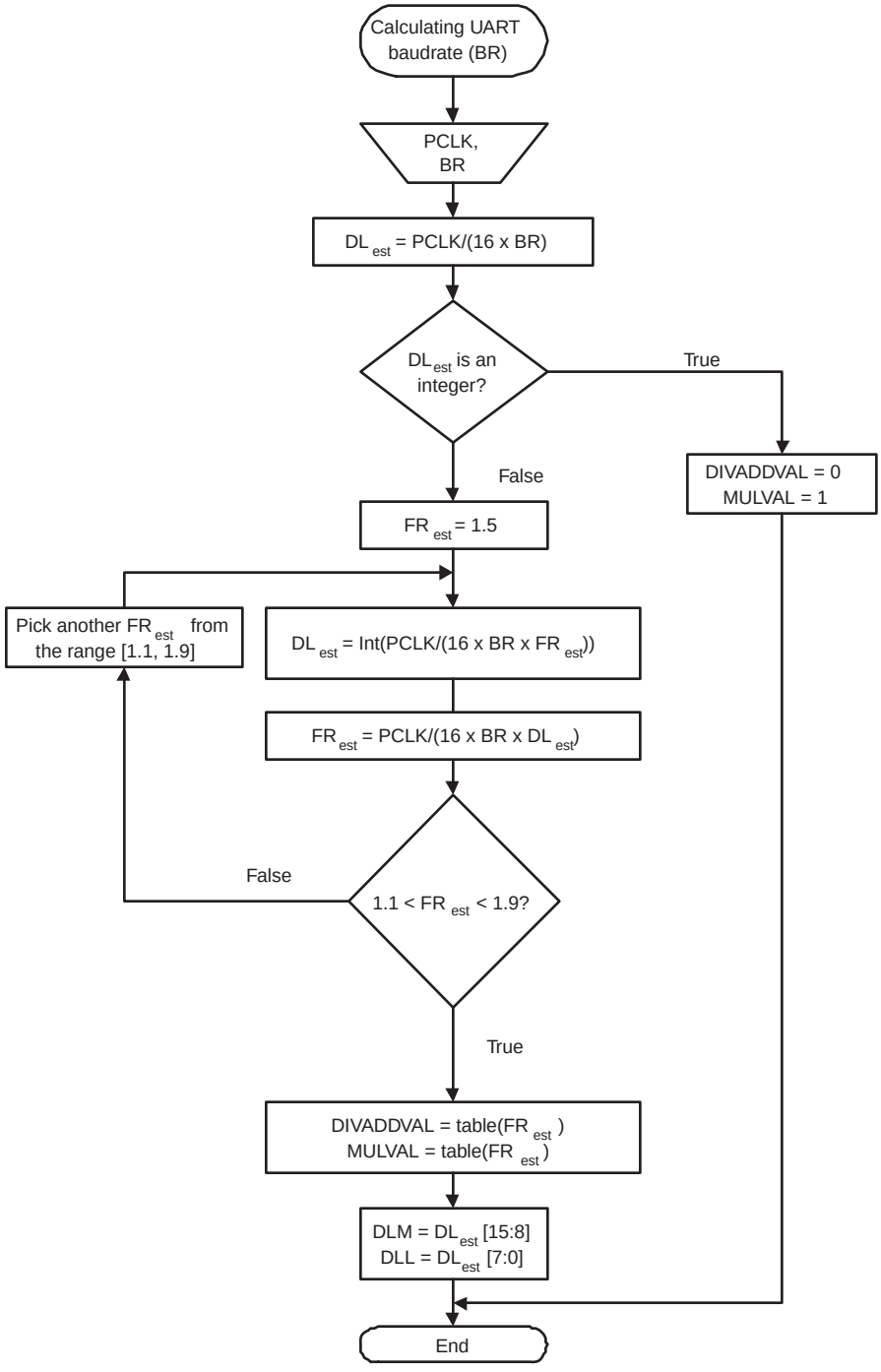


Fig 118.Algorithm for setting UART dividers

Table 846. Fractional Divider setting look-up table

FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal	FR	DivAddVal/ MulVal
1.000	0/1	1.250	1/4	1.500	1/2	1.750	3/4
1.067	1/15	1.267	4/15	1.533	8/15	1.769	10/13
1.071	1/14	1.273	3/11	1.538	7/13	1.778	7/9
1.077	1/13	1.286	2/7	1.545	6/11	1.786	11/14
1.083	1/12	1.300	3/10	1.556	5/9	1.800	4/5
1.091	1/11	1.308	4/13	1.571	4/7	1.818	9/11
1.100	1/10	1.333	1/3	1.583	7/12	1.833	5/6
1.111	1/9	1.357	5/14	1.600	3/5	1.846	11/13
1.125	1/8	1.364	4/11	1.615	8/13	1.857	6/7
1.133	2/15	1.375	3/8	1.625	5/8	1.867	13/15
1.143	1/7	1.385	5/13	1.636	7/11	1.875	7/8
1.154	2/13	1.400	2/5	1.643	9/14	1.889	8/9
1.167	1/6	1.417	5/12	1.667	2/3	1.900	9/10
1.182	2/11	1.429	3/7	1.692	9/13	1.909	10/11
1.200	1/5	1.444	4/9	1.700	7/10	1.917	11/12
1.214	3/14	1.455	5/11	1.714	5/7	1.923	12/13
1.222	2/9	1.462	6/13	1.727	8/11	1.929	13/14
1.231	3/13	1.467	7/15	1.733	11/15	1.933	14/15

37.5.16.1.1 Example 1: PCLK = 14.7456 MHz, BR = 9600

According to the provided algorithm DL_{est} = PCLK/(16 x BR) = 14.7456 MHz / (16 x 9600) = 96. Since this DL_{est} is an integer number, DIVADDVAL = 0, MULVAL = 1, DLM = 0, and DLL = 96.

37.5.16.1.2 Example 2: PCLK = 12 MHz, BR = 115200

According to the provided algorithm DL_{est} = PCLK/(16 x BR) = 12 MHz / (16 x 115200) = 6.51. This DL_{est} is not an integer number and the next step is to estimate the FR parameter. Using an initial estimate of FR_{est} = 1.5 a new DL_{est} = 4 is calculated and FR_{est} is recalculated as FR_{est} = 1.628. Since FRest = 1.628 is within the specified range of 1.1 and 1.9, DIVADDVAL and MULVAL values can be obtained from the attached look-up table.

The closest value for FRest = 1.628 in the look-up Table 846 is FR = 1.625. It is equivalent to DIVADDVAL = 5 and MULVAL = 8.

Based on these findings, the suggested UART setup would be: DLM = 0, DLL = 4, DIVADDVAL = 5, and MULVAL = 8. According to Equation 9 the UART rate is 115384. This rate has a relative error of 0.16% from the originally specified 115200.

37.5.17 UART1 Transmit Enable Register

In addition to being equipped with full hardware flow control (auto-cts and auto-rts mechanisms described above), U1TER enables implementation of software flow control, too. When TxEn=1, UART1 transmitter will keep sending data as long as they are available. As soon as TxEn becomes 0, UART1 transmission will stop.

Although Table 847 describes how to use TxEn bit in order to achieve hardware flow control, it is strongly suggested to let UART1 hardware implemented auto flow control features take care of this, and limit the scope of TxEn to software flow control.

U1TER enables implementation of software and hardware flow control. When TXEn=1, UART1 transmitter will keep sending data as long as they are available. As soon as TXEn becomes 0, UART1 transmission will stop.

Table 847 describes how to use TXEN bit in order to achieve software flow control.

Table 847: UART1 Transmit Enable Register (TER - address 0x4008 2030) bit description

Bit	Symbol	Description	Reset value
6:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7	TXEN	Transmit enable bit. When this bit is 1, as it is after a Reset, data written to the THR is output on the TXD pin as soon as any preceding data has been sent. If this bit cleared to 0 while a character is being sent, the transmission of that character is completed, but no further characters are sent until this bit is set again. In other words, a 0 in this bit blocks the transfer of characters from the THR or TX FIFO into the transmit shift register. Software can clear this bit when it detects that the a hardware-handshaking TX-permit signal (CTS) has gone false, or with software handshaking, when it receives an XOFF character (DC3). Software can set this bit again when it detects that the TX-permit signal has gone true, or when it receives an XON (DC1) character.	1
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.18 UART1 RS485 Control register

The U1RS485CTRL register controls the configuration of the UART in RS-485/EIA-485 mode.

Table 848: UART1 RS485 Control register (RS485CTRL - address 0x4008 204C) bit description

Bit	Symbol	Value	Description	Reset value
0	NMMEN		Multidrop mode select.	0
		0	RS-485/EIA-485 Normal Multidrop Mode (NMM) is disabled.	
		1	RS-485/EIA-485 Normal Multidrop Mode (NMM) is enabled. In this mode, an address is detected when a received byte causes the UART to set the parity error and generate an interrupt.	
1	RXDIS		Receive enable.	0
		0	The receiver is enabled.	
		1	The receiver is disabled.	
2	AADEN		Auto Address Detect enable.	0
		0	Auto Address Detect (AAD) is disabled.	
		1	Auto Address Detect (AAD) is enabled.	
3	SEL		Direction control.	0
		0	If direction control is enabled (bit DCTRL = 1), pin $\overline{\text{RTS}}$ is used for direction control.	
		1	If direction control is enabled (bit DCTRL = 1), pin DTR is used for direction control.	

Table 848: UART1 RS485 Control register (RS485CTRL - address 0x4008 204C) bit description

Bit	Symbol	Value	Description	Reset value
4	DCTRL		Direction control enable.	0
		0	Disable Auto Direction Control.	
		1	Enable Auto Direction Control.	
5	OINV		Polarity. This bit reverses the polarity of the direction control signal on the RTS (or DTR) pin.	0
		0	The direction control pin will be driven to logic ‘0’ when the transmitter has data to be sent. It will be driven to logic ‘1’ after the last bit of data has been transmitted.	
		1	The direction control pin will be driven to logic ‘1’ when the transmitter has data to be sent. It will be driven to logic ‘0’ after the last bit of data has been transmitted.	
31:6	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.19 UART1 RS-485 Address Match register

The U1RS485ADRMATCH register contains the address match value for RS-485/EIA-485 mode.

Table 849. UART1 RS485 Address Match register (RS485ADRMATCH - address 0x4008 2050) bit description

Bit	Symbol	Description	Reset value
7:0	ADRMATCH	Contains the address match value.	0x00
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.20 UART1 RS-485 Delay value register

The user may program the 8-bit RS485DLY register with a delay between the last stop bit leaving the TXFIFO and the de-assertion of RTS (or DTR). This delay time is in periods of the baud clock. Any delay time from 0 to 255 bit times may be programmed.

Table 850. UART1 RS485 Delay value register (RS485DLY - address 0x4008 2054) bit description

Bit	Symbol	Description	Reset value
7:0	DLY	Contains the direction control (RTS or DTR) delay value. This register works in conjunction with an 8-bit counter.	0x00
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

37.5.21 RS-485/EIA-485 modes of operation

The RS-485/EIA-485 feature allows the UART to be configured as an addressable slave. The addressable slave is one of multiple slaves controlled by a single master.

The UART master transmitter will identify an address character by setting the parity (9th) bit to ‘1’. For data characters, the parity bit is set to ‘0’.

Each UART slave receiver can be assigned a unique address. The slave can be programmed to either manually or automatically reject data following an address which is not theirs.

RS-485/EIA-485 Normal Multidrop Mode (NMM)

Setting the RS485CTRL bit 0 enables this mode. In this mode, an address is detected when a received byte causes the UART to set the parity error and generate an interrupt.

If the receiver is DISABLED (RS485CTRL bit 1 = ‘1’) any received data bytes will be ignored and will not be stored in the RXFIFO. When an address byte is detected (parity bit = ‘1’) it will be placed into the RXFIFO and an Rx Data Ready Interrupt will be generated. The processor can then read the address byte and decide whether or not to enable the receiver to accept the following data.

While the receiver is ENABLED (RS485CTRL bit 1 =’0’) all received bytes will be accepted and stored in the RXFIFO regardless of whether they are data or address. When an address character is received a parity error interrupt will be generated and the processor can decide whether or not to disable the receiver.

RS-485/EIA-485 Auto Address Detection (AAD) mode

When both RS485CTRL register bits 0 (9-bit mode enable) and 2 (AAD mode enable) are set, the UART is in auto address detect mode.

In this mode, the receiver will compare any address byte received (parity = ‘1’) to the 8-bit value programmed into the RS485ADRMATCH register.

If the receiver is DISABLED (RS485CTRL bit 1 = ‘1’) any received byte will be discarded if it is either a data byte OR an address byte which fails to match the RS485ADRMATCH value.

When a matching address character is detected it will be pushed onto the RXFIFO along with the parity bit, and the receiver will be automatically enabled (RS485CTRL bit 1 will be cleared by hardware). The receiver will also generate n Rx Data Ready Interrupt.

While the receiver is ENABLED (RS485CTRL bit 1 = ‘0’) all bytes received will be accepted and stored in the RXFIFO until an address byte which does not match the RS485ADRMATCH value is received. When this occurs, the receiver will be automatically disabled in hardware (RS485CTRL bit 1 will be set), The received non-matching address character will not be stored in the RXFIFO.

RS-485/EIA-485 Auto Direction Control

RS485/EIA-485 Mode includes the option of allowing the transmitter to automatically control the state of either the $\overline{\text{RTS}}$ pin or the $\overline{\text{DTR}}$ pin as a direction control output signal.

Setting RS485CTRL bit 4 = ‘1’ enables this feature.

Direction control, if enabled, will use the $\overline{\text{RTS}}$ pin when RS485CTRL bit 3 = ‘0’. It will use the $\overline{\text{DTR}}$ pin when RS485CTRL bit 3 = ‘1’.

When Auto Direction Control is enabled, the selected pin will be asserted (driven low) when the CPU writes data into the TXFIFO. The pin will be de-asserted (driven high) once the last bit of data has been transmitted. See bits 4 and 5 in the RS485CTRL register.

The RS485CTRL bit 4 takes precedence over all other mechanisms controlling $\overline{\text{RTS}}$ (or $\overline{\text{DTR}}$) with the exception of loopback mode.

RS485/EIA-485 driver delay time

The driver delay time is the delay between the last stop bit leaving the TXFIFO and the de-assertion of RTS (or DTR). This delay time can be programmed in the 8-bit RS485DLY register. The delay time is in periods of the baud clock. Any delay time from 0 to 255 bit times may be programmed.

RS485/EIA-485 output inversion

The polarity of the direction control signal on the $\overline{\text{RTS}}$ (or $\overline{\text{DTR}}$) pins can be reversed by programming bit 5 in the U1RS485CTRL register. When this bit is set, the direction control pin will be driven to logic 1 when the transmitter has data waiting to be sent. The direction control pin will be driven to logic 0 after the last bit of data has been transmitted.

37.5.22 UART1 FIFO Level register

U1FIFOLVL register is a read-only register that allows software to read the current FIFO level status. Both the transmit and receive FIFO levels are present in this register.

Table 851. UART1 FIFO Level register (FIFOLVL - address 0x4008 2058) bit description

Bit	Symbol	Description	Reset value
3:0	RXFIFILVL	Reflects the current level of the UART1 receiver FIFO. 0 = empty, 0xF = FIFO full.	0x00
7:4	-	Reserved. The value read from a reserved bit is not defined.	NA
11:8	TXFIFOLVL	Reflects the current level of the UART1 transmitter FIFO. 0 = empty, 0xF = FIFO full.	0x00
31:12	-	Reserved, the value read from a reserved bit is not defined.	NA

37.6 Architecture

The architecture of the UART1 is shown below in the block diagram.

The APB interface provides a communications link between the CPU or host and the UART1.

The UART1 receiver block, U1RX, monitors the serial input line, RXD1, for valid input. The UART1 RX Shift Register (U1RSR) accepts valid characters via RXD1. After a valid character is assembled in the U1RSR, it is passed to the UART1 RX Buffer Register FIFO to await access by the CPU or host via the generic host interface.

The UART1 transmitter block, U1TX, accepts data written by the CPU or host and buffers the data in the UART1 TX Holding Register FIFO (U1THR). The UART1 TX Shift Register (U1TSR) reads the data stored in the U1THR and assembles the data to transmit via the serial output pin, TXD1.

The UART1 Baud Rate Generator block, U1BRG, generates the timing enables used by the UART1 TX block. The U1BRG clock input source is the APB clock (PCLK). The main clock is divided down per the divisor specified in the U1DLL and U1DLM registers. This divided down clock is a 16x oversample clock, NBAUDOUT.

The modem interface contains registers U1MCR and U1MSR. This interface is responsible for handshaking between a modem peripheral and the UART1.

The interrupt interface contains registers U1IER and U1IIR. The interrupt interface receives several one clock wide enables from the U1TX and U1RX blocks.

Status information from the U1TX and U1RX is stored in the U1LSR. Control information for the U1TX and U1RX is stored in the U1LCR.

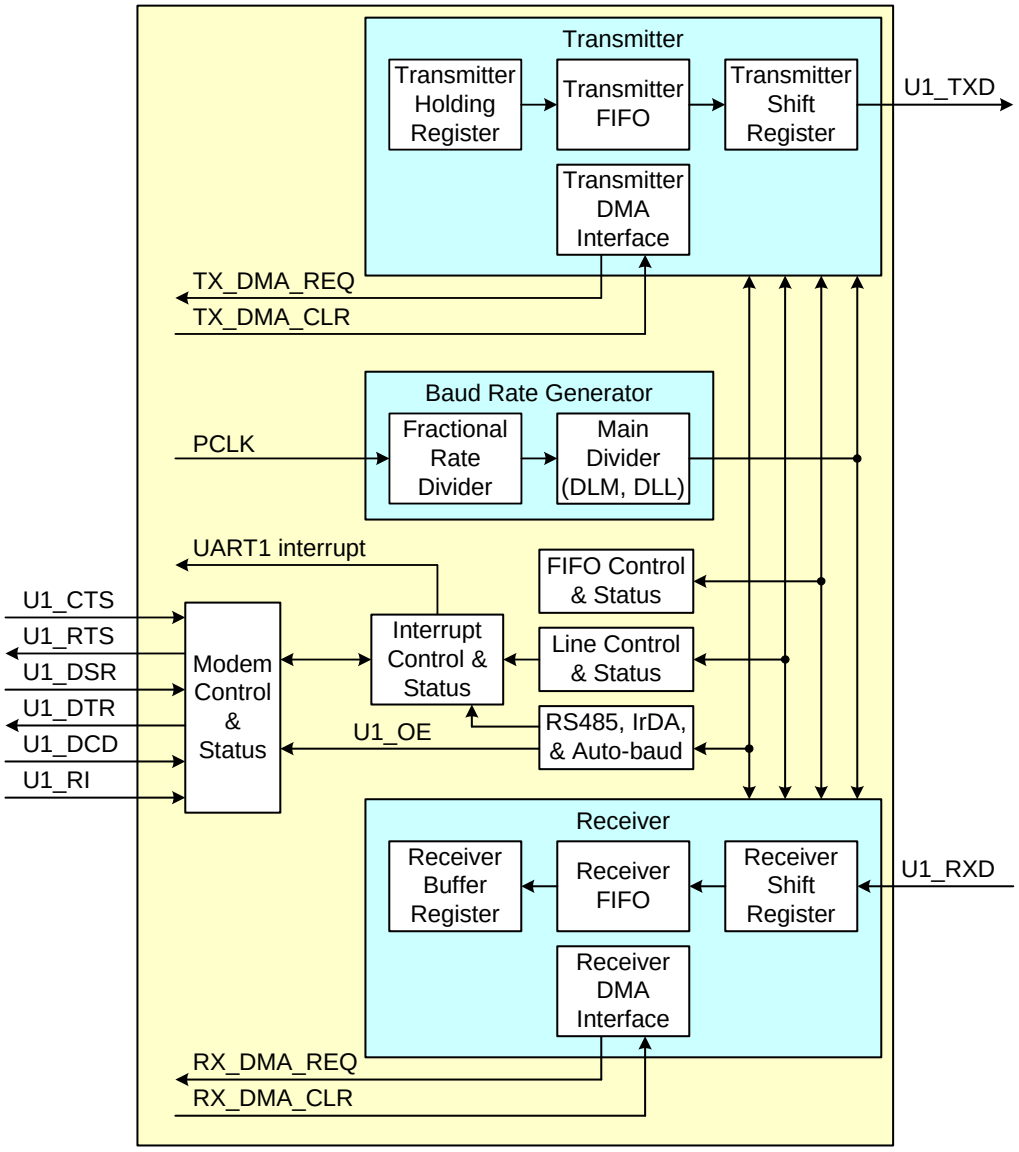


Fig 119.UART1 block diagram

38.1 How to read this chapter

The SSP0/1 controllers are available on all LPC43xx parts.

38.2 Basic configuration

The SSP0/1 are configured as follows:

- See Table 852 for clocking and power control.
- The SSP0/1 are reset by the SSP0/1_RST (reset #50/51).
- The SSP0/1 interrupts are connected to slots # 22/23 in the NVIC.
- For connecting the SSP0/1 receive and transmit lines to the GPDMA, use the DMAMUX register in the CREG block (see Table 41) and enable the GPDMA channel in the DMA Channel Configuration registers (Section 19.6.20).

Table 852. SSP0/1 clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to SSP0 register interface	BASE_M4_CLK	CLK_M4_SSP0	up to 204 MHz
SSP0 peripheral clock (PCLK)	BASE_SSP0_CLK	CLK_APB0_SSP0	up to 204 MHz
Clock to SSP1 register interface	BASE_M4_CLK	CLK_M4_SSP1	up to 204 MHz
SSP1 peripheral clock (PCLK)	BASE_SSP1_CLK	CLK_APB2_SSP1	up to 204 MHz

38.3 Features

- Compatible with Motorola SPI, 4-wire TI SSI, and National Semiconductor Microwire buses.
- Synchronous Serial Communication.
- Supports master or slave operation.
- Eight-frame FIFOs for both transmit and receive.
- 4-bit to 16-bit frame.

38.4 General description

The SSP is a Synchronous Serial Port (SSP) controller capable of operation on a SPI, 4-wire SSI, or Microwire bus. It can interact with multiple masters and slaves on the bus. Only a single master and a single slave can communicate on the bus during a given data transfer. Data transfers are in principle full duplex, with frames of 4 to 16 bits of data flowing from the master to the slave and from the slave to the master. In practice it is often the case that only one of these data flows carries meaningful data.

The LPC43xx has two Synchronous Serial Port controllers -- SSP0 and SSP1.

38.5 Pin description

Table 853. SSP pin description

Pin Name	Direction	Interface pin name/function			Pin description
		SPI	SSI	Microwire	
SCK0/1	I/O	SCK	CLK	SK	**Serial Clock.** SCK/CLK/SK is a clock signal used to synchronize the transfer of data. It is driven by the master and received by the slave. When the SPI interface is used, the clock is programmable to be active-high or active-low, otherwise it is always active-high. SCK1 only switches during a data transfer. Any other time, the SSPn interface either holds it in its inactive state, or does not drive it (leaves it in high-impedance state).
SSEL0/1	I/O	SSEL	FS	CS	**Frame Sync/Slave Select.** When the SSPn interface is a bus master, it drives this signal to an active state before the start of serial data, and then releases it to an inactive state after the serial data has been sent. The active state of this signal can be high or low depending upon the selected bus and mode. When the SSPn is a bus slave, this signal qualifies the presence of data from the Master, according to the protocol in use. When there is just one bus master and one bus slave, the Frame Sync or Slave Select signal from the Master can be connected directly to the slave's corresponding input. When there is more than one slave on the bus, further qualification of their Frame Select/Slave Select inputs will typically be necessary to prevent more than one slave from responding to a transfer.
MISO0/1	I/O	MISO	DR(M) DX(S)	SI(M) SO(S)	**Master In Slave Out.** The MISO signal transfers serial data from the slave to the master. When the SSPn is a slave, serial data is output on this signal. When the SSPn is a master, it clocks in serial data from this signal. When the SSPn is a slave and is not selected by FS/SSEL, it does not drive this signal (leaves it in high-impedance state).
MOSI0/1	I/O	MOSI	DX(M) DR(S)	SO(M) SI(S)	**Master Out Slave In.** The MOSI signal transfers serial data from the master to the slave. When the SSPn is a master, it outputs serial data on this signal. When the SSPn is a slave, it clocks in serial data from this signal.

38.6 Register description

The register addresses of the SSP controllers are shown in Table 854 and Table 855.

Table 854. Register overview: SSP0 (base address 0x4008 3000)

Name	Access	Address offset	Description	Reset value[1]	Reference
CR0	R/W	0x000	Control Register 0. Selects the serial clock rate, bus type, and data size.	0	Table 856
CR1	R/W	0x004	Control Register 1. Selects master/slave and other modes.	0	Table 857
DR	R/W	0x008	Data Register. Writes fill the transmit FIFO, and reads empty the receive FIFO.	0	Table 858
SR	RO	0x00C	Status Register	0x0000 0003	Table 859
CPSR	R/W	0x010	Clock Prescale Register	0	Table 860

Table 854. Register overview: SSP0 (base address 0x4008 3000)

Name	Access	Address offset	Description	Reset value[1]	Reference
IMSC	R/W	0x014	Interrupt Mask Set and Clear Register	0	Table 861
RIS	RO	0x018	Raw Interrupt Status Register	0x0000 0008	Table 862
MIS	RO	0x01C	Masked Interrupt Status Register	0	Table 863
ICR	WO	0x020	SSPICR Interrupt Clear Register	-	Table 864
DMACR	R/W	0x024	SSP0 DMA control register	0	Table 865

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

Table 855. Register overview: SSP1 (base address 0x400C 5000)

Name	Access	Address offset	Description	Reset value[1]	Reference
CR0	R/W	0x000	Control Register 0. Selects the serial clock rate, bus type, and data size.	0	Table 856
CR1	R/W	0x004	Control Register 1. Selects master/slave and other modes.	0	Table 857
DR	R/W	0x008	Data Register. Writes fill the transmit FIFO, and reads empty the receive FIFO.	0	Table 858
SR	RO	0x00C	Status Register	0x0000 0003	Table 859
CPSR	R/W	0x010	Clock Prescale Register	0	Table 860
IMSC	R/W	0x014	Interrupt Mask Set and Clear Register	0	Table 861
RIS	RO	0x018	Raw Interrupt Status Register	0x0000 0008	Table 862
MIS	RO	0x01C	Masked Interrupt Status Register	0	Table 863
ICR	R/W	0x020	SSPICR Interrupt Clear Register	-	Table 864
DMACR	R/W	0x024	SSP1 DMA control register	0	Table 865

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

38.6.1 SSP Control Register 0

This register controls the basic operation of the SSP controller.

Table 856: SSP Control Register 0 (CR0 - address 0x4008 3000 (SSP0), 0x400C 5000 (SSP1)) bit description

Bit	Symbol	Value	Description	Reset value
3:0	DSS		Data Size Select. This field controls the number of bits transferred in each frame. Values 0000-0010 are not supported and should not be used.	0000
		0x3	4-bit transfer	
		0x4	5-bit transfer	
		0x5	6-bit transfer	
		0x6	7-bit transfer	
		0x7	8-bit transfer	
		0x8	9-bit transfer	
		0x9	10-bit transfer	
		0xA	11-bit transfer	
		0xB	12-bit transfer	
		0xC	13-bit transfer	
		0xD	14-bit transfer	
		0xE	15-bit transfer	
		0xF	16-bit transfer	
5:4	FRF		Frame Format.	00
		0x0	SPI	
		0x1	TI	
		0x2	Microwire	
		0x3	This combination is not supported and should not be used.	
6	CPOL		Clock Out Polarity. This bit is only used in SPI mode.	0
		0	SSP controller maintains the bus clock low between frames.	
		1	SSP controller maintains the bus clock high between frames.	
7	CPHA		Clock Out Phase. This bit is only used in SPI mode.	0
		0	SSP controller captures serial data on the first clock transition of the frame, that is, the transition **away from** the inter-frame state of the clock line.	
		1	SSP controller captures serial data on the second clock transition of the frame, that is, the transition **back to** the inter-frame state of the clock line.	
15:8	SCR		Serial Clock Rate. The number of prescaler-output clocks per bit on the bus, minus one. Given that CPSDVSR is the prescale divider, and the APB clock PCLK clocks the prescaler, the bit frequency is PCLK / (CPSDVSR $\times$ [SCR+1]).	0x00
31:16	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.2 SSP Control Register 1

This register controls certain aspects of the operation of the SSP controller.

Table 857: SSP Control Register 1 (CR1 - address 0x4008 3004 (SSP0), 0x400C 5004 (SSP1)) bit description

Bit	Symbol	Value	Description	Reset value
0	LBM		Loop Back Mode.	0
		0	Normal operation.	
		1	Loop back mode. Serial input is taken from the serial output (MOSI or MISO) rather than the serial input pin (MISO or MOSI respectively).	
1	SSE		SSP Enable.	0
		0	The SSP controller is disabled.	
		1	The SSP controller will interact with other devices on the serial bus. Software should write the appropriate control information to the other SSP registers and interrupt controller registers, before setting this bit.	
2	MS		Master/Slave Mode.This bit can only be written when the SSE bit is 0.	0
		0	The SSP controller acts as a master on the bus, driving the SCLK, MOSI, and SSEL lines and receiving the MISO line.	
		1	The SSP controller acts as a slave on the bus, driving MISO line and receiving SCLK, MOSI, and SSEL lines.	
3	SOD		Slave Output Disable. This bit is relevant only in slave mode (MS = 1). If it is 1, this blocks this SSP controller from driving the transmit data line (MISO).	0
31:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.3 SSP Data Register

Software can write data to be transmitted to this register, and read data that has been received.

Table 858: SSP Data Register (DR - address 0x4008 3008 (SSP0), 0x400C 5008 (SSP1)) bit description

Bit	Symbol	Description	Reset value
15:0	DATA	**Write:** software can write data to be sent in a future frame to this register whenever the TNF bit in the Status register is 1, indicating that the Tx FIFO is not full. If the Tx FIFO was previously empty and the SSP controller is not busy on the bus, transmission of the data will begin immediately. Otherwise the data written to this register will be sent as soon as all previous data has been sent (and received). If the data length is less than 16 bits, software must right-justify the data written to this register. **Read:** software can read data from this register whenever the RNE bit in the Status register is 1, indicating that the Rx FIFO is not empty. When software reads this register, the SSP controller returns data from the least recent frame in the Rx FIFO. If the data length is less than 16 bits, the data is right-justified in this field with higher order bits filled with 0s.	0x0000
31:16	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.4 SSP Status Register

This read-only register reflects the current status of the SSP controller.

Table 859: SSP Status Register (SR - address 0x4008 300C (SSP0), 0x400C 500C (SSP1)) bit description

Bit	Symbol	Description	Reset value
0	TFE	Transmit FIFO Empty. This bit is 1 is the Transmit FIFO is empty, 0 if not.	1
1	TNF	Transmit FIFO Not Full. This bit is 0 if the Tx FIFO is full, 1 if not.	1
2	RNE	Receive FIFO Not Empty. This bit is 0 if the Receive FIFO is empty, 1 if not.	0
3	RFF	Receive FIFO Full. This bit is 1 if the Receive FIFO is full, 0 if not.	0
4	BSY	Busy. This bit is 0 if the SSPn controller is idle, or 1 if it is currently sending/receiving a frame and/or the Tx FIFO is not empty.	0
31:5	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.5 SSP Clock Prescale Register

This register controls the factor by which the Prescaler divides the SSP peripheral clock PCLK to yield the prescaler clock that is, in turn, divided by the SCR factor in SSPnCR0, to determine the bit clock.

Table 860: SSP Clock Prescale Register (CPSR - address 0x4008 3010 (SSP0), 0x400C 5010 (SSP1)) bit description

Bit	Symbol	Description	Reset value
7:0	CPSDVSR	This even value between 2 and 254, by which PCLK is divided to yield the prescaler output clock. Bit 0 always reads as 0.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Important: the SSPnCPSR value must be properly initialized or the SSP controller will not be able to transmit data correctly.

In Slave mode, the SSP clock rate provided by the master must not exceed 1/12 of the SSP peripheral clock. The content of the SSPnCPSR register is not relevant.

In master mode, $CPSDVSR_{min}$ = 2 or larger (even numbers only).

38.6.6 SSP Interrupt Mask Set/Clear Register

This register controls whether each of the four possible interrupt conditions in the SSP controller are enabled. Note that ARM uses the word “masked” in the opposite sense from classic computer terminology, in which “masked” meant “disabled”. ARM uses the word “masked” to mean “enabled”. To avoid confusion we will not use the word “masked”.

Table 861: SSP Interrupt Mask Set/Clear register (IMSC - address 0x4008 3014 (SSP0), 0x400C 5014 (SSP1)) bit description

Bit	Symbol	Description	Reset value
0	RORIM	Software should set this bit to enable interrupt when a Receive Overrun occurs, that is, when the Rx FIFO is full and another frame is completely received. The ARM spec implies that the preceding frame data is overwritten by the new frame data when this occurs.	0
1	RTIM	Software should set this bit to enable interrupt when a Receive Time-out condition occurs. A Receive Time-out occurs when the Rx FIFO is not empty, and no has not been read for a time-out period. The time-out period is the same for master and slave modes and is determined by the SSP bit rate: 32 bits at PCLK / (CPSDVSR × [SCR+1]).	0
2	RXIM	Software should set this bit to enable interrupt when the Rx FIFO is at least half full.	0
3	TXIM	Software should set this bit to enable interrupt when the Tx FIFO is at least half empty.	0
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.7 SSP Raw Interrupt Status Register

This read-only register contains a 1 for each interrupt condition that is asserted, regardless of whether or not the interrupt is enabled in the SSPnIMSC.

Table 862: SSP Raw Interrupt Status register (RIS - address 0x4008 3018 (SSP0), RIS - 0x400C 5018 (SSP1)) bit description

Bit	Symbol	Description	Reset value
0	RORRIS	This bit is 1 if another frame was completely received while the RxFIFO was full. The ARM spec implies that the preceding frame data is overwritten by the new frame data when this occurs.	0
1	RTRIS	This bit is 1 if the Rx FIFO is not empty, and has not been read for a time-out period. The time-out period is the same for master and slave modes and is determined by the SSP bit rate: 32 bits at PCLK / (CPSDVSR × [SCR+1]).	0
2	RXRIS	This bit is 1 if the Rx FIFO is at least half full.	0
3	TXRIS	This bit is 1 if the Tx FIFO is at least half empty.	1
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.8 SSP Masked Interrupt Status Register

This read-only register contains a 1 for each interrupt condition that is asserted and enabled in the SSPnIMSC. When an SSP interrupt occurs, the interrupt service routine should read this register to determine the cause(s) of the interrupt.

Table 863: SSP Masked Interrupt Status register (MIS -address 0x4008 301C (SSP0), 0x400C 501C (SSP1)) bit description

Bit	Symbol	Description	Reset value
0	RORMIS	This bit is 1 if another frame was completely received while the RxFIFO was full, and this interrupt is enabled.	0
1	RTMIS	This bit is 1 if the Rx FIFO is not empty, has not been read for a time-out period, and this interrupt is enabled. The time-out period is the same for master and slave modes and is determined by the SSP bit rate: 32 bits at PCLK / (CPSDVSR × [SCR+1]).	0
2	RXMIS	This bit is 1 if the Rx FIFO is at least half full, and this interrupt is enabled.	0
3	TXMIS	This bit is 1 if the Tx FIFO is at least half empty, and this interrupt is enabled.	0
31:4	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.9 SSP Interrupt Clear Register

Software can write one or more one(s) to this write-only register, to clear the corresponding interrupt condition(s) in the SSP controller. Note that the other two interrupt conditions can be cleared by writing or reading the appropriate FIFO, or disabled by clearing the corresponding bit in SSPnIMSC.

Table 864: SSP interrupt Clear Register (ICR - address 0x4008 3020 (SSP0), ICR - 0x400C 5020 (SSP1)) bit description

Bit	Symbol	Description	Reset value
0	RORIC	Writing a 1 to this bit clears the “frame was received when RxFIFO was full” interrupt.	NA
1	RTIC	Writing a 1 to this bit clears the Rx FIFO was not empty and has not been read for a time-out period interrupt. The time-out period is the same for master and slave modes and is determined by the SSP bit rate: 32 bits at PCLK / (CPSDVSR × [SCR+1]).	NA
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

38.6.10 SSP DMA Control Register

The SSPnDMACR register is the DMA control register. It is a read/write register.

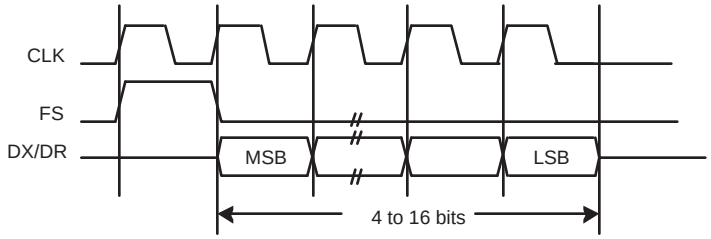
Table 865: SSP DMA Control Register (DMACR - address 0x4008 3024 (SSP0), 0x400C 5024 (SSP1)) bit description

Bit	Symbol	Description	Reset value
0	RXDMAE	Receive DMA Enable. When this bit is set to one 1, DMA for the receive FIFO is enabled, otherwise receive DMA is disabled.	0
1	TXDMAE	Transmit DMA Enable. When this bit is set to one 1, DMA for the transmit FIFO is enabled, otherwise transmit DMA is disabled	0
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

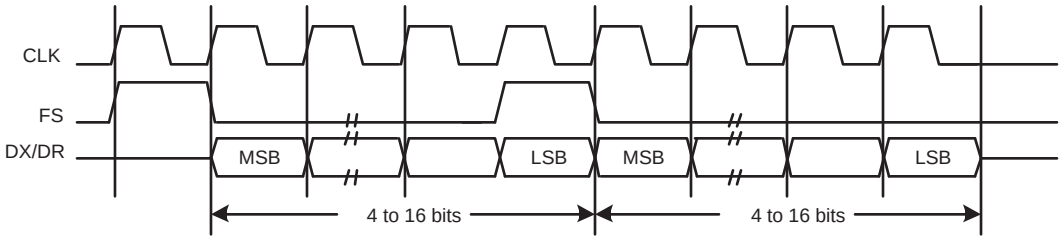
38.7 Functional description

38.7.1 Texas Instruments synchronous serial frame format

Figure 120 shows the 4-wire Texas Instruments synchronous serial frame format supported by the SSP module.



a. Single frame transfer



b. Continuous/back-to-back frames transfer

Fig 120. Texas Instruments Synchronous Serial Frame Format: a) Single and b) Continuous/back-to-back Two Frames Transfer

For device configured as a master in this mode, CLK and FS are forced LOW, and the transmit data line DX is tri-stated whenever the SSP is idle. Once the bottom entry of the transmit FIFO contains data, FS is pulsed HIGH for one CLK period. The value to be transmitted is also transferred from the transmit FIFO to the serial shift register of the transmit logic. On the next rising edge of CLK, the MSB of the 4-bit to 16-bit data frame is shifted out on the DX pin. Likewise, the MSB of the received data is shifted onto the DR pin by the off-chip serial slave device.

Both the SSP and the off-chip serial slave device then clock each data bit into their serial shifter on the falling edge of each CLK. The received data is transferred from the serial shifter to the receive FIFO on the first rising edge of CLK after the LSB has been latched.

38.7.2 SPI frame format

The SPI interface is a four-wire interface where the SSEL signal behaves as a slave select. The main feature of the SPI format is that the inactive state and phase of the SCK signal are programmable through the CPOL and CPHA bits within the SSPCR0 control register.

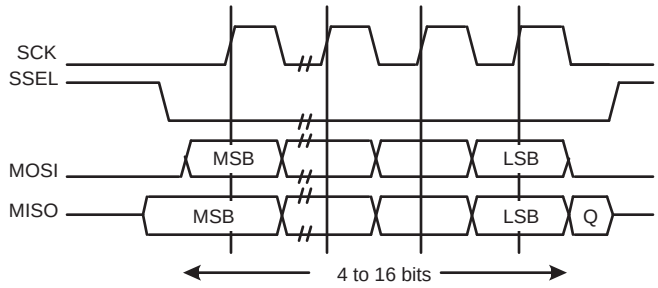
38.7.2.1 Clock Polarity (CPOL) and Phase (CPHA) control

When the CPOL clock polarity control bit is 0, it produces a steady state low value on the SCK pin. If the CPOL clock polarity control bit is 1, a steady state high value is placed on the CLK pin when data is not being transferred.

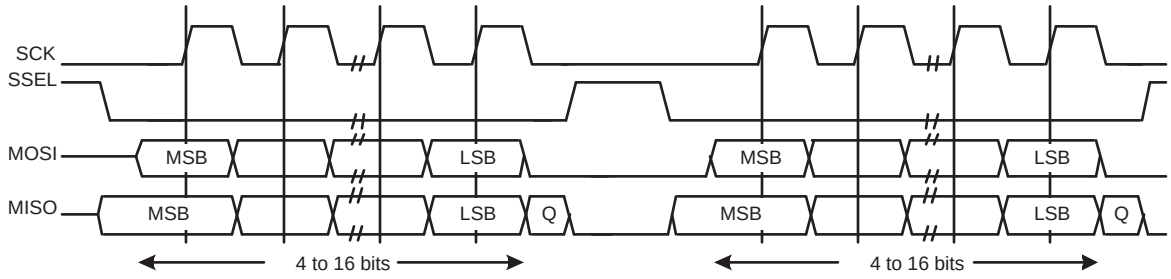
The CPHA control bit selects the clock edge that captures data and allows it to change state. It has the most impact on the first bit transmitted by either allowing or not allowing a clock transition before the first data capture edge. When the CPHA phase control bit is 0, data is captured on the first clock edge transition. If the CPHA clock phase control bit is 1, data is captured on the second clock edge transition.

38.7.2.2 SPI format with CPOL=0,CPHA=0

Single and continuous transmission signal sequences for SPI format with CPOL = 0, CPHA = 0 are shown in Figure 121.



a. Single transfer with CPOL=0 and CPHA=0



b. Continuous transfer with CPOL=0 and CPHA=0

Fig 121. SPI frame format with CPOL=0 and CPHA=0 (a) Single and b) Continuous Transfer)

In this configuration, during idle periods:

- The CLK signal is forced LOW.
- SSEL is forced HIGH.
- The transmit MOSI/MISO pad is in high impedance.

If the SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSEL master signal being driven LOW. This causes slave data to be enabled onto the MISO input line of the master. Master's MOSI is enabled.

One half SCK period later, valid master data is transferred to the MOSI pin. Now that both the master and slave data have been set, the SCK master clock pin goes HIGH after one further half SCK period.

The data is now captured on the rising and propagated on the falling edges of the SCK signal.

In the case of a single word transmission, after all bits of the data word have been transferred, the SSEL line is returned to its idle HIGH state one SCK period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSEL signal must be pulsed HIGH between each data word transfer. This is because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the CPHA bit is logic zero. Therefore the master device must raise the SSEL pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSEL pin is returned to its idle state one SCK period after the last bit has been captured.

38.7.2.3 SPI format with CPOL=0,CPHA=1

The transfer signal sequence for SPI format with CPOL = 0, CPHA = 1 is shown in Figure 122, which covers both single and continuous transfers.

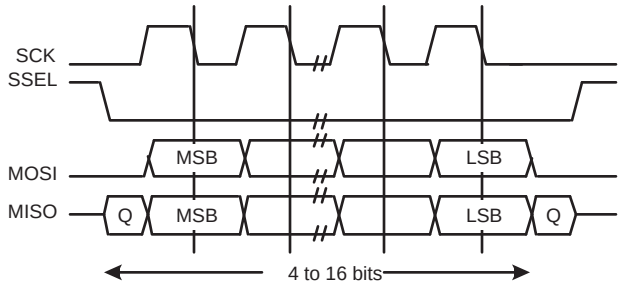


Fig 122. SPI frame format with CPOL=0 and CPHA=1

In this configuration, during idle periods:

- The CLK signal is forced LOW.
- SSEL is forced HIGH.
- The transmit MOSI/MISO pad is in high impedance.

If the SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSEL master signal being driven LOW. Master's MOSI pin is enabled. After a further one half SCK period, both master and slave valid data is enabled onto their respective transmission lines. At the same time, the SCK is enabled with a rising edge transition.

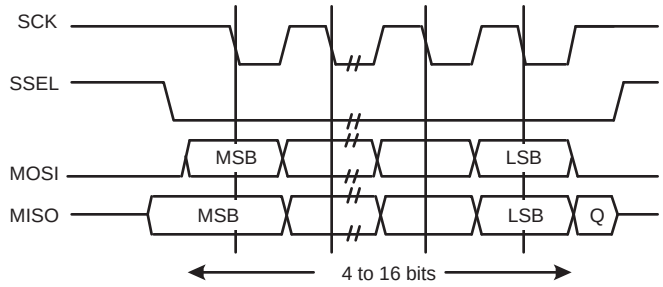
Data is then captured on the falling edges and propagated on the rising edges of the SCK signal.

In the case of a single word transfer, after all bits have been transferred, the SSEL line is returned to its idle HIGH state one SCK period after the last bit has been captured.

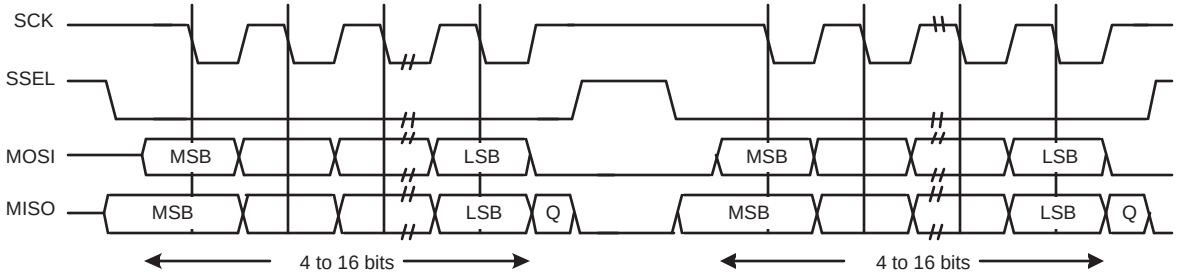
For continuous back-to-back transfers, the SSEL pin is held LOW between successive data words and termination is the same as that of the single word transfer.

38.7.2.4 SPI format with CPOL = 1,CPHA = 0

Single and continuous transmission signal sequences for SPI format with CPOL=1, CPHA=0 are shown in Figure 123.



a. Single transfer with CPOL=1 and CPHA=0



b. Continuous transfer with CPOL=1 and CPHA=0

Fig 123. SPI frame format with CPOL = 1 and CPHA = 0 (a) Single and b) Continuous Transfer)

In this configuration, during idle periods:

- The CLK signal is forced HIGH.
- SSEL is forced HIGH.
- The transmit MOSI/MISO pad is in high impedance.

If the SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSEL master signal being driven LOW, which causes slave data to be immediately transferred onto the MISO line of the master. Master's MOSI pin is enabled.

One half period later, valid master data is transferred to the MOSI line. Now that both the master and slave data have been set, the SCK master clock pin becomes LOW after one further half SCK period. This means that data is captured on the falling edges and be propagated on the rising edges of the SCK signal.

In the case of a single word transmission, after all bits of the data word are transferred, the SSEL line is returned to its idle HIGH state one SCK period after the last bit has been captured.

However, in the case of continuous back-to-back transmissions, the SSEL signal must be pulsed HIGH between each data word transfer. This is because the slave select pin freezes the data in its serial peripheral register and does not allow it to be altered if the CPHA bit is logic zero. Therefore the master device must raise the SSEL pin of the slave device between each data transfer to enable the serial peripheral data write. On completion of the continuous transfer, the SSEL pin is returned to its idle state one SCK period after the last bit has been captured.

38.7.2.5 SPI format with CPOL = 1,CPHA = 1

The transfer signal sequence for SPI format with CPOL = 1, CPHA = 1 is shown in Figure 124, which covers both single and continuous transfers.

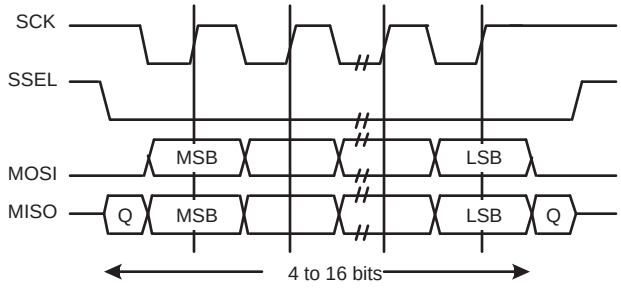


Fig 124. SPI Frame Format with CPOL = 1 and CPHA = 1

In this configuration, during idle periods:

- The CLK signal is forced HIGH.
- SSEL is forced HIGH.
- The transmit MOSI/MISO pad is in high impedance.

If the SSP is enabled and there is valid data within the transmit FIFO, the start of transmission is signified by the SSEL master signal being driven LOW. Master's MOSI is enabled. After a further one half SCK period, both master and slave data are enabled onto their respective transmission lines. At the same time, the SCK is enabled with a falling edge transition. Data is then captured on the rising edges and propagated on the falling edges of the SCK signal.

After all bits have been transferred, in the case of a single word transmission, the SSEL line is returned to its idle HIGH state one SCK period after the last bit has been captured. For continuous back-to-back transmissions, the SSEL pins remains in its active LOW state, until the final bit of the last word has been captured, and then returns to its idle state as described above. In general, for continuous back-to-back transfers the SSEL pin is held LOW between successive data words and termination is the same as that of the single word transfer.

38.7.3 National Semiconductor Microwire frame format

Figure 125 shows the Microwire frame format for a single frame. Figure 126 shows the same format when back-to-back frames are transmitted.

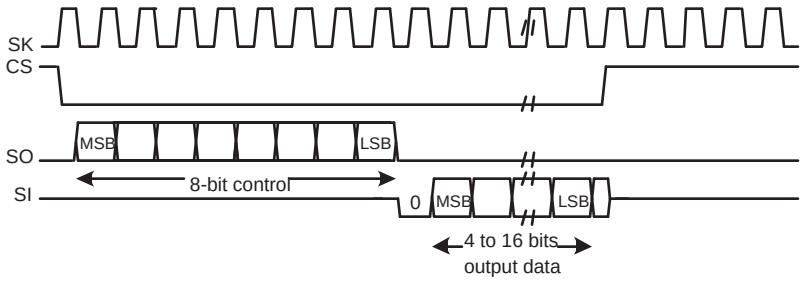


Fig 125. Microwire frame format (single transfer)

Microwire format is very similar to SPI format, except that transmission is half-duplex instead of full-duplex, using a master-slave message passing technique. Each serial transmission begins with an 8-bit control word that is transmitted from the SSP to the off-chip slave device. During this transmission, no incoming data is received by the SSP. After the message has been sent, the off-chip slave decodes it and, after waiting one serial clock after the last bit of the 8-bit control message has been sent, responds with the required data. The returned data is 4 to 16 bits in length, making the total frame length anywhere from 13 to 25 bits.

In this configuration, during idle periods:

- The SK signal is forced LOW.
- CS is forced HIGH.
- The transmit data line SO is arbitrarily forced LOW.

A transmission is triggered by writing a control byte to the transmit FIFO.The falling edge of CS causes the value contained in the bottom entry of the transmit FIFO to be transferred to the serial shift register of the transmit logic, and the MSB of the 8-bit control frame to be shifted out onto the SO pin. CS remains LOW for the duration of the frame transmission. The SI pin remains tristated during this transmission.

The off-chip serial slave device latches each control bit into its serial shifter on the rising edge of each SK. After the last bit is latched by the slave device, the control byte is decoded during a one clock wait-state, and the slave responds by transmitting data back to the SSP. Each bit is driven onto SI line on the falling edge of SK. The SSP in turn

latches each bit on the rising edge of SK. At the end of the frame, for single transfers, the CS signal is pulled HIGH one clock period after the last bit has been latched in the receive serial shifter, that causes the data to be transferred to the receive FIFO.

Note: The off-chip slave device can tristate the receive line either on the falling edge of SK after the LSB has been latched by the receive shiftier, or when the CS pin goes HIGH.

For continuous transfers, data transmission begins and ends in the same manner as a single transfer. However, the CS line is continuously asserted (held LOW) and transmission of data occurs back to back. The control byte of the next frame follows directly after the LSB of the received data from the current frame. Each of the received values is transferred from the receive shifter on the falling edge SK, after the LSB of the frame has been latched into the SSP.

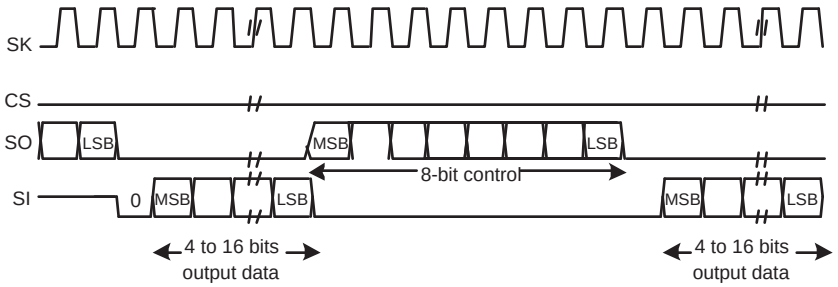


Fig 126. Microwire frame format (continuous transfers)

38.7.3.1 Setup and hold time requirements on CS with respect to SK in Microwire mode

In the Microwire mode, the SSP slave samples the first bit of receive data on the rising edge of SK after CS has gone LOW. Masters that drive a free-running SK must ensure that the CS signal has sufficient setup and hold margins with respect to the rising edge of SK.

Figure 127 illustrates these setup and hold time requirements. With respect to the SK rising edge on which the first bit of receive data is to be sampled by the SSP slave, CS must have a setup of at least two times the period of SK on which the SSP operates. With respect to the SK rising edge previous to this edge, CS must have a hold of at least one SK period.

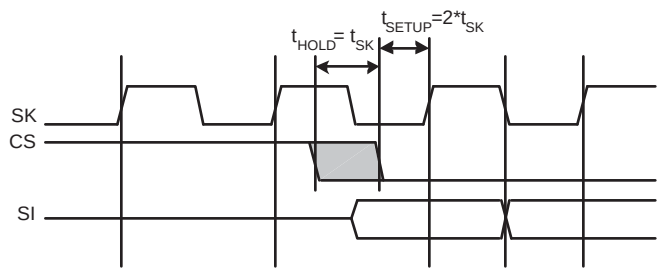


Fig 127. Microwire frame format setup and hold details

39.1 How to read this chapter

The SPI controller is available on all LPC43xx parts.

39.2 Basic configuration

The SPI is configured as follows:

- See Table 866 for clocking and power control.
- The SPI is reset by the SPI_RST (reset # 58).
- The SPI interrupt is connected to NVIV slot # 20 in the Cortex-M0 NVIC.

Table 866. SPI clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to SPI; peripheral SPI clock	BASE_SPI_CLK	CLK_SPI	up to 204 MHz
Clock to the peripheral bus controller	BASE_PERIPH_CLK	CLK_PERIPH_BUS	up to 204 MHz
Clock to the peripheral core controller	BASE_PERIPH_CLK	BASE_PERIPH_CORE	up to 204 MHz

39.3 Features

- Compliant with Serial Peripheral Interface (SPI) specification.
- Synchronous, Serial, Full Duplex Communication.
- SPI master or slave.
- Maximum data bit rate of one eighth of the peripheral clock rate.
- 8 to 16 bits per transfer.

39.4 General description

SPI is a full duplex serial interface. It can handle multiple masters and slaves being connected to a given bus. Only a single master and a single slave can communicate on the interface during a given data transfer. During a data transfer the master always sends 8 to 16 bits of data to the slave, and the slave always sends a byte of data to the master.

The block diagram of the SPI solution implemented in SPI interface is shown in the Figure 128.

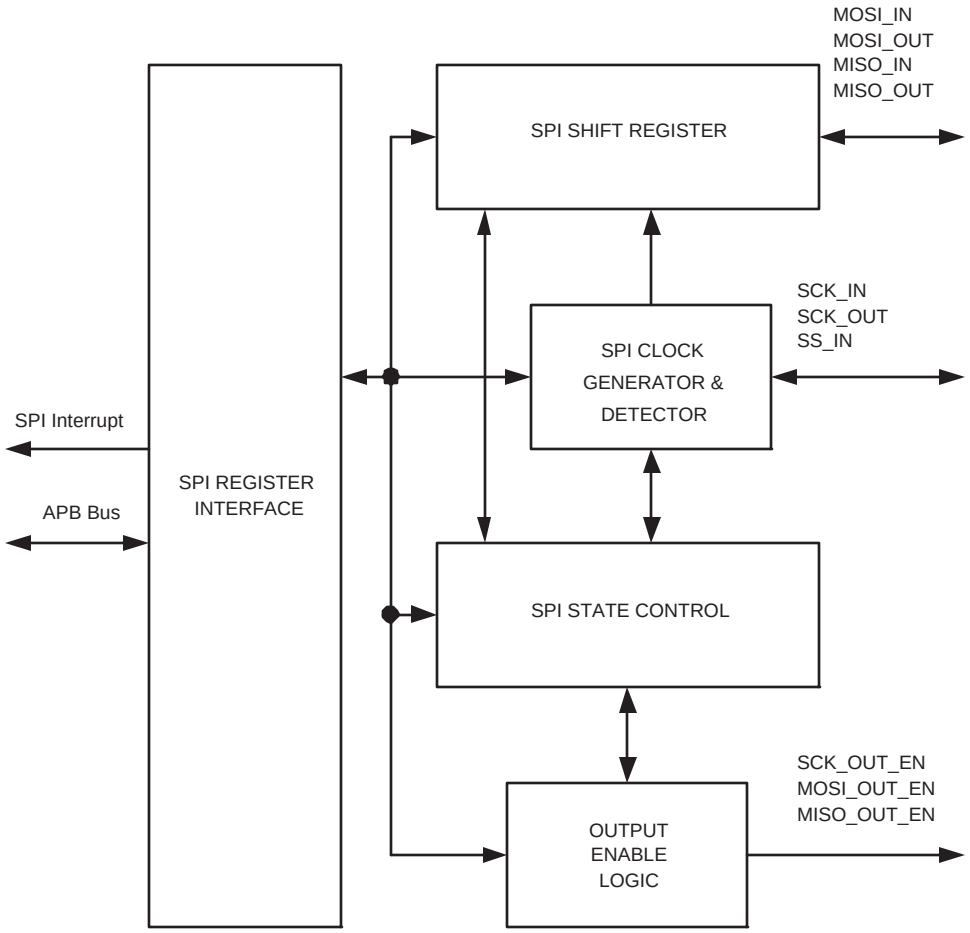


Fig 128. SPI block diagram

39.5 Pin description

Table 867. SPI pin description

Pin Name	Type	Pin Description
SCK	Input/ Output	**Serial Clock.** The SPI clock signal (SCK) is used to synchronize the transfer of data across the SPI interface. The SPI is always driven by the master and received by the slave. The clock is programmable to be active high or active low. The SPI is only active during a data transfer. Any other time, it is either in its inactive state, or tri-stated.
SSEL	Input	**Slave Select.** The SPI slave select signal (SSEL) is an active low signal that indicates which slave is currently selected to participate in a data transfer. Each slave has its own unique slave select signal input. The SSEL must be low before data transactions begin and normally stays low for the duration of the transaction. If the SSEL signal goes high any time during a data transfer, the transfer is considered to be aborted. In this event, the slave returns to idle, and any data that was received is thrown away. There are no other indications of this exception. This signal is not directly driven by the master. It could be driven by a simple general purpose I/O under software control. **Remark:** Note that this pin in an input pin only. The SPI in master mode cannot drive the CS input on the slave. Any GPIO pin can be used for SPI chip select in master mode.
MISO	Input/ Output	**Master In Slave Out.** The SPI Master In Slave Out signal (MISO) is a unidirectional signal used to transfer serial data from an SPI slave to an SPI master. When a device is a slave, serial data is output on this pin. When a device is a master, serial data is input on this pin. When a slave device is not selected, the slave drives the signal high-impedance.
MOSI	Input/ Output	**Master Out Slave In.** The SPI Master Out Slave In signal (MOSI) is a unidirectional signal used to transfer serial data from an SPI master to an SPI slave. When a device is a master, serial data is output on this pin. When a device is a slave, serial data is input on this pin.

39.6 Register description

The SPI contains registers as shown in Table 868. All registers are byte, half word and word accessible.

Table 868. Register overview: SPI (base address 0x4010 0000)

Name	Access	Address offset	Description	Reset value[1]
CR	R/W	0x000	SPI Control Register. This register controls the operation of the SPI.	0x00
SR	RO	0x004	SPI Status Register. This register shows the status of the SPI.	0x00
DR	R/W	0x008	SPI Data Register. This bi-directional register provides the transmit and receive data for the SPI. Transmit data is provided to the SPI0 by writing to this register. Data received by the SPI0 can be read from this register.	0x00
CCR	R/W	0x00C	SPI Clock Counter Register. This register controls the frequency of a master's SCK0.	0x00

Table 868. Register overview: SPI (base address 0x4010 0000)

Name	Access	Address offset	Description	Reset value[1]
TCR	R/W	0x010	SPI Test Control register. For functional testing only.	0x00
TSR	R/W	0x014	SPI Test Status register. For functional testing only.	0x00
-	R/W	0x018	Reserved.	-
INT	R/W	0x01C	SPI Interrupt Flag. This register contains the interrupt flag for the SPI interface.	0x00

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

39.6.1 SPI Control Register

The SPCR register controls the operation of SPI through the configuration bits setting shown in Table 869.

Table 869: SPI Control Register (CR - address 0x4010 0000) bit description

Bit	Symbol	Value	Description	Reset value
1:0	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
2	BITENABLE	0	The SPI controller sends and receives 8 bits of data per transfer.	0
		1	The SPI controller sends and receives the number of bits selected by bits 11:8.	
3	CPHA		Clock phase control determines the relationship between the data and the clock on SPI transfers, and controls when a slave transfer is defined as starting and ending.	0
		0	Data is sampled on the first clock edge of SCK. A transfer starts and ends with activation and deactivation of the SSEL signal.	
		1	Data is sampled on the second clock edge of the SCK. A transfer starts with the first clock edge, and ends with the last sampling edge when the SSEL signal is active.	
4	CPOL		Clock polarity control.	0
		0	SCK is active high.	
		1	SCK is active low.	
5	MSTR		Master mode select.	0
		0	The SPI operates in Slave mode.	
		1	The SPI operates in Master mode.	
6	LSBF		LSB First controls which direction each byte is shifted when transferred.	0
		0	SPI data is transferred MSB (bit 7) first.	
		1	SPI data is transferred LSB (bit 0) first.	

Table 869: SPI Control Register (CR - address 0x4010 0000) bit description *…continued*

Bit	Symbol	Value	Description	Reset value
7	SPIE		Serial peripheral interrupt enable.	0
		0	SPI interrupts are inhibited.	
		1	A hardware interrupt is generated each time the SPIF or MODF bits are activated.	
11:8	BITS		When bit 2 of this register is 1, this field controls the number of bits per transfer:	0000
		0x8	8 bits per transfer	
		0x9	9 bits per transfer	
		0xA	10 bits per transfer	
		0xB	11 bits per transfer	
		0xC	12 bits per transfer	
		0xD	13 bits per transfer	
		0xE	14 bits per transfer	
		0xF	15 bits per transfer	
		0x0	16 bits per transfer	
31:12	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.6.2 SPI Status Register

The SPSR register controls the operation of SPI as per the configuration bits setting shown in Table 870.

Table 870: SPI Status Register (SR - address 0x4010 0004) bit description

Bit	Symbol	Description	Reset value
2:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	ABRT	Slave abort. When 1, this bit indicates that a slave abort has occurred. This bit is cleared by reading this register.	0
4	MODF	Mode fault. when 1, this bit indicates that a Mode fault error has occurred. This bit is cleared by reading this register, then writing the SPI0 control register.	0
5	ROVR	Read overrun. When 1, this bit indicates that a read overrun has occurred. This bit is cleared by reading this register.	0

Table 870: SPI Status Register (SR - address 0x4010 0004) bit description

Bit	Symbol	Description	Reset value
6	WCOL	Write collision. When 1, this bit indicates that a write collision has occurred. This bit is cleared by reading this register, then accessing the SPI Data Register.	0
7	SPIF	SPI transfer complete flag. When 1, this bit indicates when a SPI data transfer is complete. When a master, this bit is set at the end of the last cycle of the transfer. When a slave, this bit is set on the last data sampling edge of the SCK. This bit is cleared by first reading this register, then accessing the SPI Data Register. **Note:** this is not the SPI interrupt flag. This flag is found in the SPINT register.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.6.3 SPI Data Register

This bi-directional data register provides the transmit and receive data for the SPI. Transmit data is provided to the SPI by writing to this register. Data received by the SPI can be read from this register. When used as a master, a write to this register will start an SPI data transfer. Writes to this register will be blocked when a data transfer starts, or when the SPIF status bit is set, and the SPI Status Register has not been read.

Table 871: SPI Data Register (DR - address 0x4010 0008) bit description

Bit	Symbol	Description	Reset value
7:0	DATALOW	SPI Bi-directional data port.	0x00
15:8	DATAHIGH	If bit 2 of the SPCR is 1 and bits 11:8 are other than 1000, some or all of these bits contain the additional transmit and receive bits. When less than 16 bits are selected, the more significant among these bits read as zeroes.	0x00
31:16	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.6.4 SPI Clock Counter Register

This register controls the frequency of a master's SCK. The register indicates the number of SPI peripheral clock cycles that make up an SPI clock.

In Master mode, this register must be an even number greater than or equal to 8. Violations of this can result in unpredictable behavior. The SPI SCK rate may be calculated as: PCLK_SPI / SPCCR value.

In Slave mode, the SPI clock rate provided by the master must not exceed 1/8 of the SPI peripheral clock. The content of the SPCCR register is not relevant.

Table 872: SPI Clock Counter Register (CCR - address 0x4010 0010) bit description

Bit	Symbol	Description	Reset value
7:0	COUNTER	SPI0 Clock counter setting.	0x00
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.6.5 SPI Test Control Register

Note that the bits in this register are intended for functional verification only. This register should not be used for normal operation.

Table 873: SPI Test Control Register (TCR - address 0x4010 0010) bit description

Bit	Symbol	Description	Reset value
0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
7:1	TEST	SPI test mode. When 0, the SPI operates normally. When 1, SCK will always be on, independent of master mode select and data availability setting.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.6.6 SPI Test Status Register

Note: The bits in this register are intended for functional verification only. This register should not be used for normal operation.

This register is a replication of the SPI Status Register. The difference between the registers is that a read of this register will not start the sequence of events required to clear these status bits. A write to this register will set an interrupt if the write data for the respective bit is a 1.

Table 874: SPI Test Status Register (TSR - address 0x4010 0014) bit description

Bit	Symbol	Description	Reset value
2:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
3	ABRT	Slave abort.	0
4	MODF	Mode fault.	0
5	ROVR	Read overrun.	0
6	WCOL	Write collision.	0
7	SPIF	SPI transfer complete flag.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.6.7 SPI Interrupt Register

This register contains the interrupt flag for the SPI0 interface.

Table 875: SPI Interrupt Register (INT - address 0x4010 001C) bit description

Bit	Symbol	Description	Reset value
0	SPIF	SPI interrupt flag. Set by the SPI interface to generate an interrupt. Cleared by writing a 1 to this bit. **Note:** this bit will be set once when SPIE = 1 and at least one of SPIF and WCOL bits is 1. However, only when the SPI Interrupt bit is set and SPI0 Interrupt is enabled in the NVIC, SPI based interrupt can be processed by interrupt handling software.	0
7:1	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

39.7 Functional description

39.7.1 SPI data transfers

Figure 129 is a timing diagram that illustrates the four different data transfer formats that are available with the SPI port. This timing diagram illustrates a single 8-bit data transfer. The first thing you should notice in this timing diagram is that it is divided into three horizontal parts. The first part describes the SCK and SSEL signals. The second part describes the MOSI and MISO signals when the Clock Phase control bit (CPHA) in the SPI Control Register is 0. The third part describes the MOSI and MISO signals when the CPHA variable is 1.

In the first part of the timing diagram, note two points. First, the SPI is illustrated with the Clock Polarity control bit (CPOL) in the SPI Control Register set to both 0 and 1. The second point to note is the activation and de-activation of the SSEL signal. When CPHA = 0, the SSEL signal will always go inactive between data transfers. This is not guaranteed when CPHA = 1 (the signal can remain active).

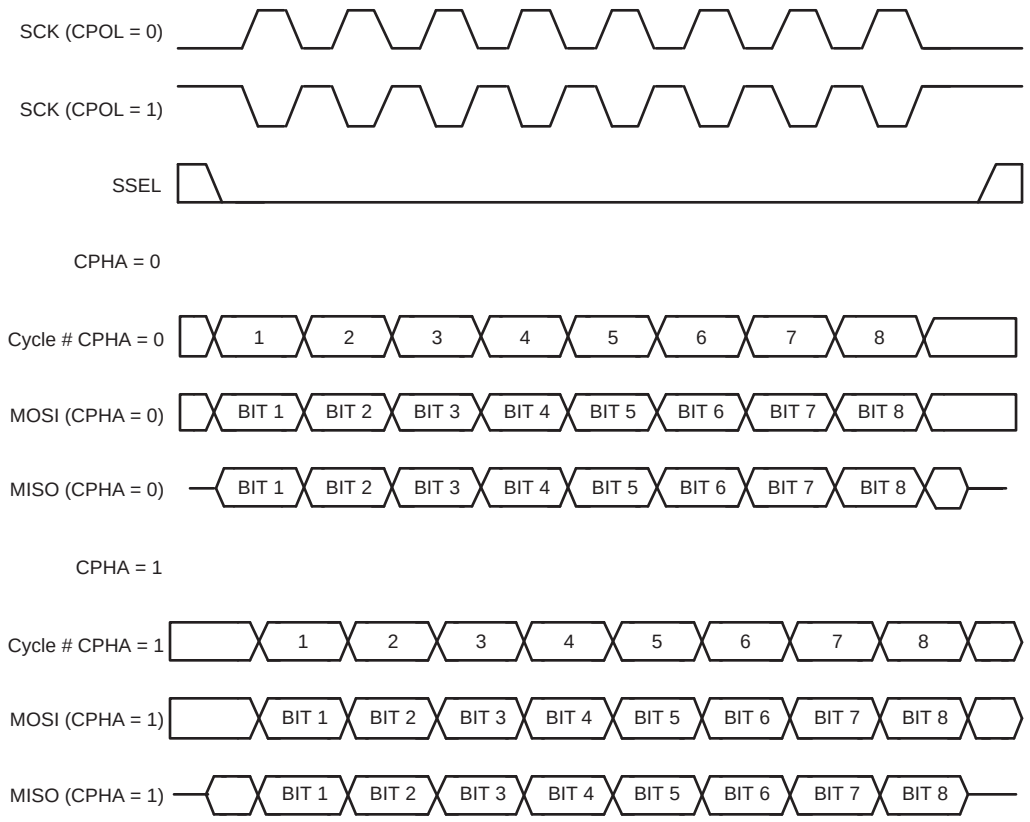


Fig 129. SPI data transfer format (CPHA = 0 and CPHA = 1)

The data and clock phase relationships are summarized in Table 876.

Table 876. SPI Data To Clock Phase Relationship

CPOL and CPHA settings	When the first data bit is driven	When all other data bits are driven	When data is sampled
CPOL = 0, CPHA = 0	Prior to first SCK rising edge	SCK falling edge	SCK rising edge
CPOL = 0, CPHA = 1	First SCK rising edge	SCK rising edge	SCK falling edge
CPOL = 1, CPHA = 0	Prior to first SCK falling edge	SCK rising edge	SCK falling edge
CPOL = 1, CPHA = 1	First SCK falling edge	SCK falling edge	SCK rising edge

The definition of when a transfer starts and stops is dependent on whether a device is a master or a slave, and the setting of the CPHA variable.

When a device is a master, the start of a transfer is indicated by the master having a byte of data that is ready to be transmitted. At this point, the master can activate the clock, and begin the transfer. The transfer ends when the last clock cycle of the transfer is complete.

When a device is a slave and CPHA is set to 0, the transfer starts when the SSEL signal goes active, and ends when SSEL goes inactive. When a device is a slave, and CPHA is set to 1, the transfer starts on the first clock edge when the slave is selected, and ends on the last clock edge where data is sampled.

39.7.2 General information

There are five control and status registers for the SPI port. They are described in detail in Section 39.6 “Register description” on page 943.

The SPI Control Register (S0SPCR) contains a number of programmable bits used to control the function of the SPI block. The settings for this register must be set up prior to a given data transfer taking place.

The SPI Status Register (S0SPSR) contains read-only bits that are used to monitor the status of the SPI interface, including normal functions, and exception conditions. The primary purpose of this register is to detect completion of a data transfer. This is indicated by the SPI Interrupt Flag (SPIF) in the S0SPINT register. The remaining bits in the register are exception condition indicators. These exceptions will be described later in this section.

The SPI Data Register (S0SPDR) is used to provide the transmit and receive data bytes. An internal shift register in the SPI block logic is used for the actual transmission and reception of the serial data. Data is written to the SPI Data Register for the transmit case. There is no buffer between the data register and the internal shift register. A write to the data register goes directly into the internal shift register. Therefore, data should only be written to this register when a transmit is not currently in progress. Read data is buffered. When a transfer is complete, the receive data is transferred to a single byte data buffer, where it is later read. A read of the SPI Data Register returns the value of the read data buffer.

The SPI Clock Counter Register (S0SPCCR) controls the clock rate when the SPI block is in master mode. This needs to be set prior to a transfer taking place, when the SPI block is a master. This register has no function when the SPI block is a slave.

Prior to use, SPI configurations such as the master/slave settings, clock polarity, clock rate, etc. must be set up in the SPI Control Register and SPI Clock Counter Register.

The I/Os for this implementation of SPI are standard CMOS I/Os. The open drain SPI option is not implemented in this design. When a device is set up to be a slave, its I/Os are only active when it is selected by the SSEL signal being active.

39.7.3 Master operation

The following sequence can be followed to set up the SPI prior to its first use as a master. This is typically done during program initialization.

1. Set the SPI Clock Counter Register to the desired clock rate.
2. Set the SPI Control Register to the desired settings for master mode.

The following sequence describes how one should process a data transfer with the SPI block when it is set up to be the master. This process assumes that any prior data transfer has already completed.

1. Optionally, verify the SPI setup before starting the transfer.
2. Write the data to transmitted to the SPI Data Register. This write starts the SPI data transfer.
3. Wait for the SPIF bit in the SPI Status Register to be set to 1. The SPIF bit will be set after the last cycle of the SPI data transfer.

4. Read the SPI Status Register.
5. Read the received data from the SPI Data Register (optional).
6. Go to step 2 if more data is to be transmitted.

Note: A read or write of the SPI Data Register is required in order to clear the SPIF status bit. Therefore, if the optional read of the SPI Data Register does not take place, a write to this register is required in order to clear the SPIF status bit.

39.7.4 Slave operation

The following sequence can be followed to set up the SPI prior to its first use as a slave. This is typically done during program initialization.

1. Set the SPI Control Register to the desired settings for slave mode.

The following sequence describes how one should process a data transfer with the SPI block when it is set up to be a slave. This process assumes that any prior data transfer has already completed. It is required that the system clock driving the SPI logic be at least 8X faster than the SPI.

1. Optionally, verify the SPI setup before starting the transfer.
2. Write the data to transmitted to the SPI Data Register (optional). Note that this can only be done when a slave SPI transfer is not in progress.
3. Wait for the SPIF bit in the SPI Status Register to be set to 1. The SPIF bit will be set after the last sampling clock edge of the SPI data transfer.
4. Read the SPI Status Register.
5. Read the received data from the SPI Data Register (optional).
6. Go to step 2 if more data is to be transferred.

Note: A read or write of the SPI Data Register is required in order to clear the SPIF status bit. Therefore, at least one of the optional reads or writes of the SPI Data Register must take place, in order to clear the SPIF status bit.

39.7.5 Exception conditions

Read Overrun

A read overrun occurs when the SPI block internal read buffer contains data that has not been read by the processor, and a new transfer has completed. The read buffer containing valid data is indicated by the SPIF bit in the SPI Interrupt Register being active. When a transfer completes, the SPI block needs to move the received data to the read buffer. If the SPIF bit is active (the read buffer is full), the new receive data will be lost, and the read overrun (ROVR) bit in the SPI Status Register will be activated.

Write Collision

As stated previously, there is no write buffer between the SPI block bus interface, and the internal shift register. As a result, data must not be written to the SPI Data Register when a SPI data transfer is currently in progress. The time frame where data cannot be written to the SPI Data Register is from when the transfer starts, until after the SPI Status

Register has been read when the SPIF status is active. If the SPI Data Register is written in this time frame, the write data will be lost, and the write collision (WCOL) bit in the SPI Status Register will be activated.

Mode Fault

If the SSEL signal goes active when the SPI block is a master, this indicates another master has selected the device to be a slave. This condition is known as a mode fault. When a mode fault is detected, the mode fault (MODF) bit in the SPI Status Register will be activated, the SPI signal drivers will be de-activated, and the SPI mode will be changed to be a slave.

If the SSEL function is assigned to its related pin in the relevant Pin Function Select Register, the SSEL signal must always be inactive when the SPI controller is a master.

Slave Abort

A slave transfer is considered to be aborted if the SSEL signal goes inactive before the transfer is complete. In the event of a slave abort, the transmit and receive data for the transfer that was in progress are lost, and the slave abort (ABRT) bit in the SPI Status Register will be activated.

40.1 How to read this chapter

The I^2S0/1 interfaces are available on all LPC43xx parts.

40.2 Basic configuration

The I^2S interface is configured as follows:

- See Table 877 for clocking and power control.
- The I2S0 is reset by the I2S0_RST (reset # 52).
- The I2S1 is reset by the I2S1_RST (reset # 53).
- The I2S0 interrupt is connected to slot # 28 in the NVIC.
- The I2S1 interrupt is connected to slot # 29 in the NVIC.
- For connecting the I2S receive and transmit lines to the GPDMA, use the DMAMUX register in the CREG block (see Table 41) and enable the GPDMA channel in the DMA Channel Configuration registers (Section 19.6.20).
- See Table 43 for configuring the I2S clock inputs for the audio PLL.
- The I2S0/1 MWS signals (I2S0_RX_MWS/I2S0_TX_MWS/IS1_RX_MWS/I2S1_TX_MWS) can be connected to timer3 or the SCT through the GIMA (see Table 138).

Table 877. I2S clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the I2S0 and I2S1 register interface and I2S0/1 peripheral clock.	BASE_APB1_CLK	CLK_APB1_I2S	up to 204 MHz

40.3 Features

The I2S bus provides a standard communication interface for digital audio applications.

The I2S bus specification defines a 3-wire serial bus, having one data, one clock, and one word select signal. The basic I2S connection has one master, which is always the master, and one slave. The I2S interface provides a separate transmit and receive channel, each of which can operate as either a master or a slave.

- The I2S input can operate in both master and slave mode, independently of the I2S output.
- The I2S output can operate in both master and slave mode, independently of the I2S input.
- Capable of handling 8-bit, 16-bit, and 32-bit word sizes.
- Mono and stereo audio data supported.
- Versatile clocking includes independent transmit and receive fractional rate generators, and an ability to use a single clock input or output for a 4-wire mode.

- The sampling frequency (fs) can range (in practice) from 16 to 192 kHz (16, 22.05, 32, 44.1, 48, 96, or 192 kHz) for audio applications.
- Separate Master Clock outputs for both transmit and receive channels support a clock up to 512 times the I^2S sampling frequency.
- Word Select period in master mode is configurable (separately for I^2S input and I^2S output).
- Two 8 word (32 byte) FIFO data buffers are provided, one for transmit and one for receive.
- Generates interrupt requests when buffer levels cross a programmable boundary.
- Two DMA requests, controlled by programmable buffer levels. These are connected to the General Purpose DMA block.
- Controls include reset, stop and mute options separately for I2S input and I2S output.

40.4 General description

The I2S performs serial data out via the transmit channel and serial data in via the receive channel. These support the Inter IC Audio format for 8-bit, 16-bit and 32-bit audio data, both for stereo and mono modes. Configuration, data access and control is performed by a APB register set. Data streams are buffered by FIFOs with a depth of 8 words.

The I2S receive and transmit stage can operate independently in either slave or master mode. In master mode, the I2S module supplies the SCK and WS signals. In slave mode, the SCK and WS signals are provided by the external master.

- In master mode, word select is generated internally with a 9-bit counter. The ratio of the SCK and WS signals can be programmed in the control register.
- In slave mode, word select is input from the relevant bus pin.
- When an I2S bus is active, the word select, receive clock and transmit clock signals are sent continuously by the bus master, while data is sent continuously by the transmitter.
- Disabling the I2S can be done with the stop or mute control bits separately for the transmit and receive.
- The stop bit will disable accesses by the transmit channel or the receive channel to the FIFOs and will place the transmit channel in mute mode.
- The mute control bit will place the transmit channel in mute mode. In mute mode, the transmit channel FIFO operates normally, but the output is discarded and replaced by zeroes. This bit does not affect the receive channel, data reception can occur normally.

40.4.1 I2S connection schemes

I2S1 is automatically a slave to I2S0 if no external pins are selected for the I2S1 clock and data lines.

MCLK can be provided by a master or used by the master to create the I2S SCK. MCLK can also be generated internally by the audio PLL through the CREG block (see Table 40).

40.4.2 I2S connections to the GIMA

The Word Select (WS) signal is generated by the I2S blocks in slave and master mode to capture or generate data.The WS signal is routed to the SCU and to the GIMA where it can be selected as input to Timer3 or the SCT. The GIMA input is divided by 128.

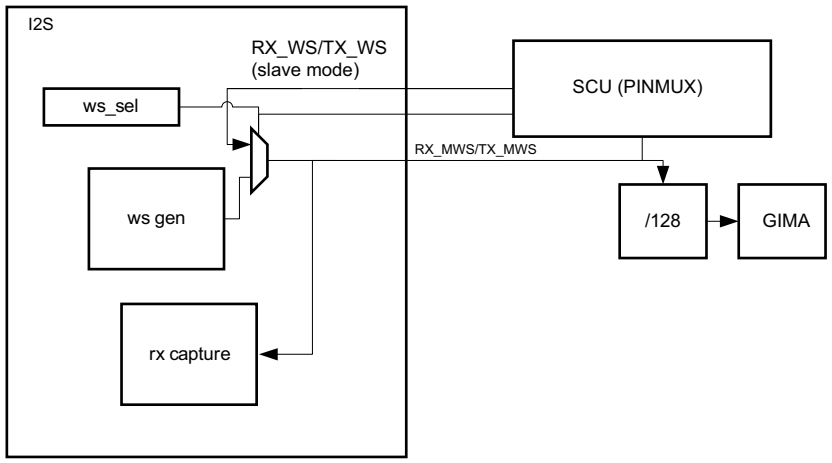


Fig 130. WS signal connections

40.5 Pin description

Table 878. Pin description

Pin function	Direction	Description
I2S0/1_RX_SCK	Input/ Output	Receive Clock. A clock signal used to synchronize the transfer of data on the receive channel. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I2S bus specification.
I2S0/1_RX_WS	Input/ Output	Receive Word Select. Selects the channel from which data is to be received. It is driven by the master and received by the slave. Corresponds to the signal WS in the I2S bus specification. WS = 0 indicates that data is being received by channel 1 (left channel). WS = 1 indicates that data is being received by channel 2 (right channel).
I2S0/1_RX_SDA	Input/ Output	Receive Data. Serial data, received MSB first. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the I2S bus specification.
I2S0/1_RX_MCLK	Output	Optional master clock output for the I2S receive function. This output is <tbd> when the I2S block is in slave mode.
I2S0/1_TX_SCK	Input/ Output	Transmit Clock. A clock signal used to synchronize the transfer of data on the transmit channel. It is driven by the master and received by the slave. Corresponds to the signal SCK in the I2S bus specification.
I2S0/1_TX_WS	Input/ Output	Transmit Word Select. Selects the channel to which data is being sent. It is driven by the master and received by the slave. Corresponds to the signal WS in the I2S bus specification. WS = 0 indicates that data is being sent to channel 1 (left channel). WS = 1 indicates that data is being sent to channel 2 (right channel).
I2S0/1_TX_SDA	Input/ Output	Transmit Data. Serial data, sent MSB first. It is driven by the transmitter and read by the receiver. Corresponds to the signal SD in the I2S bus specification.
IS0/1_TX_MCLK	Output	Optional master clock output for the I2S transmit function. This output is <tbd> when the I2S block is in slave mode.

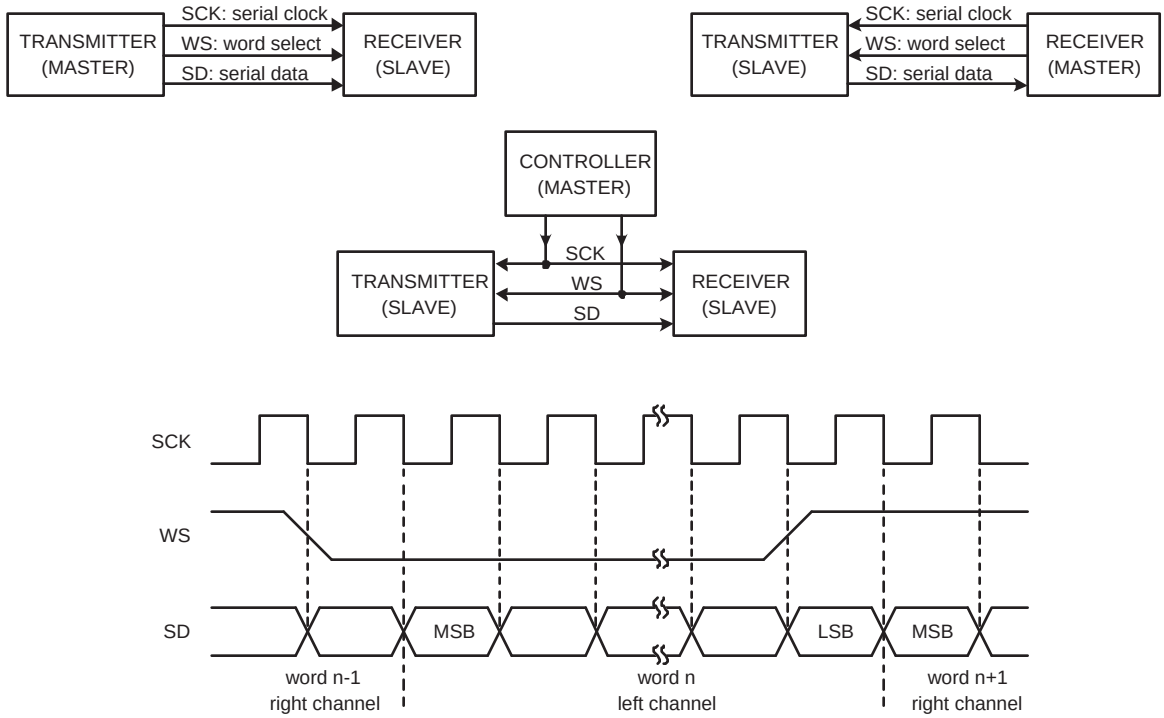


Fig 131. Simple I2S configurations and bus timing

40.6 Register description

Table 879 shows the registers associated with the I2S interface and a summary of their functions. Following the table are details for each register.

Reset value reflects the data stored in used bits only. It does not include reserved bits content.

Table 879. Register overview: I2S0 (base address 0x400A 2000)

Name	Access	Address offset	Description	Reset value	Reference
DAO	R/W	0x000	I2S Digital Audio Output Register. Contains control bits for the I2S transmit channel.	0x87E1	Table 881
DAI	R/W	0x004	I2S Digital Audio Input Register. Contains control bits for the I2S receive channel.	0x07E1	Table 882
TXFIFO	WO	0x008	I2S Transmit FIFO. Access register for the 8 x 32-bit transmitter FIFO.	0	Table 883
RXFIFO	RO	0x00C	I2S Receive FIFO. Access register for the 8 x 32-bit receiver FIFO.	0	Table 884
STATE	RO	0x010	I2S Status Feedback Register. Contains status information about the I2S interface.	0x7	Table 885
DMA1	R/W	0x014	I2S DMA Configuration Register 1. Contains control information for DMA request 1.	0	Table 886
DMA2	R/W	0x018	I2S DMA Configuration Register 2. Contains control information for DMA request 2.	0	Table 887
IRQ	R/W	0x01C	I2S Interrupt Request Control Register. Contains bits that control how the I2S interrupt request is generated.	0	Table 888
TXRATE	R/W	0x020	I2S Transmit MCLK divider. This register determines the I2S TX MCLK rate by specifying the value to divide PCLK by in order to produce MCLK.	0	Table 889
RXRATE	R/W	0x024	I2S Receive MCLK divider. This register determines the I2S RX MCLK rate by specifying the value to divide PCLK by in order to produce MCLK.	0	Table 890
TXBITRATE	R/W	0x028	I2S Transmit bit rate divider. This register determines the I2S transmit bit rate by specifying the value to divide TX_MCLK by in order to produce the transmit bit clock.	0	Table 891
RXBITRATE	R/W	0x02C	I2S Receive bit rate divider. This register determines the I2S receive bit rate by specifying the value to divide RX_MCLK by in order to produce the receive bit clock.	0	Table 892
TXMODE	R/W	0x030	I2S Transmit mode control.	0	Table 893
RXMODE	R/W	0x034	I2S Receive mode control.	0	Table 894

Table 880. Register overview: I2S1 (base address 0x400A 3000)

Name	Access	Address offset	Description	Reset value	Reference
DAO	R/W	0x000	I2S Digital Audio Output Register. Contains control bits for the I2S transmit channel.	0x87E1	Table 881
DAI	R/W	0x004	I2S Digital Audio Input Register. Contains control bits for the I2S receive channel.	0x07E1	Table 882
TXFIFO	WO	0x008	I2S Transmit FIFO. Access register for the 8 x 32-bit transmitter FIFO.	0	Table 883
RXFIFO	RO	0x00C	I2S Receive FIFO. Access register for the 8 x 32-bit receiver FIFO.	0	Table 884
STATE	RO	0x010	I2S Status Feedback Register. Contains status information about the I2S interface.	0x7	Table 885
DMA1	R/W	0x014	I2S DMA Configuration Register 1. Contains control information for DMA request 1.	0	Table 886
DMA2	R/W	0x018	I2S DMA Configuration Register 2. Contains control information for DMA request 2.	0	Table 887
IRQ	R/W	0x01C	I2S Interrupt Request Control Register. Contains bits that control how the I2S interrupt request is generated.	0	Table 888
TXRATE	R/W	0x020	I2S Transmit MCLK divider. This register determines the I2S TX MCLK rate by specifying the value to divide PCLK by in order to produce MCLK.	0	Table 889
RXRATE	R/W	0x024	I2S Receive MCLK divider. This register determines the I2S RX MCLK rate by specifying the value to divide PCLK by in order to produce MCLK.	0	Table 890
TXBITRATE	R/W	0x028	I2S Transmit bit rate divider. This register determines the I2S transmit bit rate by specifying the value to divide TX_MCLK by in order to produce the transmit bit clock.	0	Table 891
RXBITRATE	R/W	0x02C	I2S Receive bit rate divider. This register determines the I2S receive bit rate by specifying the value to divide RX_MCLK by in order to produce the receive bit clock.	0	Table 892
TXMODE	R/W	0x030	I2S Transmit mode control.	0	Table 893
RXMODE	R/W	0x034	I2S Receive mode control.	0	Table 894

40.6.1 I2S Digital Audio Output register

The DAO register controls the operation of the I2S transmit channel. The function of bits in DAO are shown in Table 881.

Table 881. I2S Digital Audio Output register (DAO - address 0x400A 2000 (I2S0) and 0x400A 3000 (I2S1)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	WORDWIDTH		Selects the number of bytes in data as follows:	01
		0x0	8-bit data	
		0x1	16-bit data	
		0x2	Reserved, do not use this setting	
		0x3	32-bit data	
2	MONO		When 1, data is of monaural format. When 0, the data is in stereo format.	0

Table 881. I2S Digital Audio Output register (DAO - address 0x400A 2000 (I2S0) and 0x400A 3000 (I2S1)) bit description

Bit	Symbol	Value	Description	Reset value
3	STOP		When 1, disables accesses on FIFOs, places the transmit channel in mute mode.	0
4	RESET		When 1, asynchronously resets the transmit channel and FIFO.	0
5	WS_SEL		When 0, the interface is in master mode. When 1, the interface is in slave mode. See Section 40.7.2 for a summary of useful combinations for this bit with TXMODE.	1
14:6	WS_HALFPERIOD		Word select half period minus 1, i.e. WS 64clk period -> ws_halfperiod = 31.	0x1F
15	MUTE		When 1, the transmit channel sends only zeroes.	1
31:16	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.2 I2S Digital Audio Input register

The DAI register controls the operation of the I2S receive channel. The function of bits in DAI are shown in Table 882.

Table 882. I2S Digital Audio Input register (DAI - address 0x400A 2004 (I2S0) and 0x400A 3004 (I2S1)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	WORDWIDTH		Selects the number of bytes in data as follows:	01
		0x0	8-bit data	
		0x1	16-bit data	
		0x2	Reserved, do not use this setting	
		0x3	32-bit data	
2	MONO		When 1, data is of monaural format. When 0, the data is in stereo format.	0
3	STOP		When 1, disables accesses on FIFOs, places the transmit channel in mute mode.	0
4	RESET		When 1, asynchronously reset the transmit channel and FIFO.	0
5	WS_SEL		When 0, the interface is in master mode. When 1, the interface is in slave mode. See Section 40.7.2 for a summary of useful combinations for this bit with RXMODE.	1
14:6	WS_HALFPERIOD		Word select half period minus 1, i.e. WS 64clk period -> ws_halfperiod = 31.	0x1F
31:15	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.3 I2S Transmit FIFO register

The TXFIFO register provides access to the transmit FIFO.

Table 883. Transmit FIFO register (TXFIFO - address 0x400A 2008 (I2S0) and 0x400A 3008 (I2S1)) bit description

Bit	Symbol	Description	Reset value
31:0	I2STXFIFO	8 x 32-bit transmit FIFO.	0

40.6.4 Receive FIFO register

The I2SRXFIFO register provides access to the receive FIFO.

Table 884. I2S Receive FIFO register (RXFIFO - address 0x400A 200C (I2S0) and 0x400A 300C (I2S1)) bit description

Bit	Symbol	Description	Reset value
31:0	I2SRXFIFO	8 x 32-bit transmit FIFO.	0

40.6.5 I2S Status Feedback register

The STATE register provides status information about the I2S interface.

Table 885. I2S Status Feedback register (STATE - address 0x400A 2010 (I2S0) and 0x400A 3010 (I2S1)) bit description

Bit	Symbol	Description	Reset value
0	IRQ	This bit reflects the presence of Receive Interrupt or Transmit Interrupt. This is determined by comparing the current FIFO levels to the rx_depth_irq and tx_depth_irq fields in the IRQ register.	1
1	DMAREQ1	This bit reflects the presence of Receive or Transmit DMA Request 1. This is determined by comparing the current FIFO levels to the rx_depth_dma1 and tx_depth_dma1 fields in the DMA1 register.	1
2	DMAREQ2	This bit reflects the presence of Receive or Transmit DMA Request 2. This is determined by comparing the current FIFO levels to the rx_depth_dma2 and tx_depth_dma2 fields in the DMA2 register.	1
7:3	-	Reserved.	0
11:8	RX_LEVEL	Reflects the current level of the Receive FIFO.	0
15:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
19:16	TX_LEVEL	Reflects the current level of the Transmit FIFO.	0
31:20	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.6 I2S DMA Configuration Register 1

The DMA1 register controls the operation of DMA request 1. The function of bits in DMA1 are shown in Table 886. Refer to Chapter 19 for details of DMA operation.

This register enables the DMA for the I^2S receive and transmit channels and sets the FIFO level.

Remark: The FIFOs contain eight 32 bit Dwords. Therefore, if the I^2S controller is configured for 32-bit mode (see Table 881 and Table 882), the maximum allowed FIFO level is 4.

Table 886. I2S DMA Configuration register 1 (DMA1 - address 0x400A 2014 (I2S0) and 0x400A 3014 (I2S1)) bit description

Bit	Symbol	Description	Reset value
0	RX_DMA1_ENABLE	When 1, enables DMA1 for I2S receive.	0
1	TX_DMA1_ENABLE	When 1, enables DMA1 for I2S transmit.	0
7:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	0
11:8	RX_DEPTH_DMA1	Set the FIFO level that triggers a receive DMA request on DMA1.	0

Table 886. I2S DMA Configuration register 1 (DMA1 - address 0x400A 2014 (I2S0) and 0x400A 3014 (I2S1)) bit description

Bit	Symbol	Description	Reset value
15:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
19:16	TX_DEPTH_DMA1	Set the FIFO level that triggers a transmit DMA request on DMA1.	0
31:20	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.7 I2S DMA Configuration Register 2

The DMA2 register controls the operation of DMA request 2. The function of bits in DMA2 are shown in Table 881.

This register enables the DMA for the I^2S receive and transmit channels and sets the FIFO level.

Remark: The FIFOs contain eight 32 bit Dwords. Therefore, if the I^2S controller is configured for 32-bit mode (see Table 881 and Table 882), the maximum allowed FIFO level is 4.

Table 887. I2S DMA Configuration register 2 (DMA2 - address 0x400A 2018 (I2S0) and 0x400A 3018 (I2S1)) bit description

Bit	Symbol	Description	Reset value
0	RX_DMA2_ENABLE	When 1, enables DMA1 for I2S receive.	0
1	TX_DMA2_ENABLE	When 1, enables DMA1 for I2S transmit.	0
7:2	-	Reserved.	0
11:8	RX_DEPTH_DMA2	Set the FIFO level that triggers a receive DMA request on DMA2.	0
15:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
19:16	TX_DEPTH_DMA2	Set the FIFO level that triggers a transmit DMA request on DMA2.	0
31:20	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.8 I2S Interrupt Request Control register

The IRQ register controls the operation of the I2S interrupt request. The function of bits in IRQ are shown in Table 881.

Table 888. I2S Interrupt Request Control register (IRQ - address 0x400A 201C (I2S0) and 0x400A 301C (I2S1)) bit description

Bit	Symbol	Description	Reset value
0	RX_IRQ_ENABLE	When 1, enables I2S receive interrupt.	0
1	TX_IRQ_ENABLE	When 1, enables I2S transmit interrupt.	0
7:2	-	Reserved.	0
11:8	RX_DEPTH_IRQ	Set the FIFO level on which to create an irq request.	0

Table 888. I2S Interrupt Request Control register (IRQ - address 0x400A 201C (I2S0) and 0x400A 301C (I2S1)) bit description

Bit	Symbol	Description	Reset value
15:12	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
19:16	TX_DEPTH_IRQ	Set the FIFO level on which to create an irq request.	0
31:20	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.9 I2S Transmit Clock Rate register

The MCLK rate for the I2S transmitter is determined by the values in the TXRATE register. The required TXRATE setting depends on the desired audio sample rate, the format (stereo/mono) used, and the data size.

The transmitter MCLK rate is generated using a fractional rate generator, dividing down the frequency of PCLK = CLK_APB1_I2S. Values of the numerator (X) and the denominator (Y) must be chosen to produce a frequency twice that desired for the transmitter MCLK, which must be an integer multiple of the transmitter bit clock rate. Fractional rate generators have some aspects that the user should be aware of when choosing settings. These are discussed in Section 40.6.9.1. The equation for the fractional rate generator is:

I2S_TX_MCLK = PCLK * (X/Y) /2

Note: If the value of X or Y is 0, then no clock is generated. Also, the value of Y must be greater than or equal to X.

Table 889. I2S Transmit Clock Rate register (TXRATE - address 0x400A 2020 (I2S0) and 0x400A 3020 (I2S1)) bit description

Bit	Symbol	Description	Reset value
7:0	Y_DIVIDER	I2S transmit MCLK rate denominator. This value is used to divide PCLK to produce the transmit MCLK. Eight bits of fractional divide supports a wide range of possibilities. A value of 0 stops the clock.	0
15:8	X_DIVIDER	I2S transmit MCLK rate numerator. This value is used to multiply PCLK by to produce the transmit MCLK. A value of 0 stops the clock. Eight bits of fractional divide supports a wide range of possibilities. Note: the resulting ratio X/Y is divided by 2.	0
31:16	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.9.1 Notes on fractional rate generators

A fractional rate generator can introduce output jitter with some divide settings. This is because the fractional rate generator is a fully digital function, so the output clock transitions are synchronous with the source clock, whereas a theoretical perfect fractional rate may have edges that are not related to the source clock. Therefore the output jitter will not be greater than plus or minus one source clock between consecutive clock edges.

For example, if X = 0x07 and Y = 0x11, the fractional rate generator will output 7 clocks for every 17 (11 hex) input clocks, distributed as evenly as it can. In this example, there is no way to distribute the output clocks in a perfectly even fashion, so some clocks will be longer than others. The output is divided by 2 in order to square it up, which also helps

with the jitter. The frequency averages out to exactly (7/17) / 2, but some clocks will be a slightly different length than their neighbors. It is possible to avoid jitter entirely by choosing fractions such that X divides evenly into Y, such as 2/4, 2/6, 3/9, 1/N, etc.

40.6.10 I2S Receive Clock Rate register

The MCLK rate for the I2S receiver is determined by the values in the RXRATE register. The required RXRATE setting depends on the peripheral clock rate (PCLK_I2S = CLK_APB1_I2S) and the desired MCLK rate (such as 256 fs).

The receiver MCLK rate is generated using a fractional rate generator, dividing down the frequency of PCLK_I2S. Values of the numerator (X) and the denominator (Y) must be chosen to produce a frequency twice that desired for the receiver MCLK, which must be an integer multiple of the receiver bit clock rate. Fractional rate generators have some aspects that the user should be aware of when choosing settings. These are discussed in Section 40.6.9.1. The equation for the fractional rate generator is:

I2S_RX_MCLK = PCLK_I2S * (X/Y) /2

Note: If the value of X or Y is 0, then no clock is generated. Also, the value of Y must be greater than or equal to X.

Table 890. I2S Receive Clock Rate register (RXRATE - address 0x400A 2024 (I2S0) and 0x400A 3024 (I2S1)) bit description

Bit	Symbol	Description	Reset value
7:0	Y_DIVIDER	I2S receive MCLK rate denominator. This value is used to divide PCLK to produce the receive MCLK. Eight bits of fractional divide supports a wide range of possibilities. A value of 0 stops the clock.	0
15:8	X_DIVIDER	I2S receive MCLK rate numerator. This value is used to multiply PCLK by to produce the receive MCLK. A value of 0 stops the clock. Eight bits of fractional divide supports a wide range of possibilities. Note: the resulting ratio X/Y is divided by 2.	0
31:16	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.11 I2S Transmit Clock Bit Rate register

The bit rate for the I2S transmitter is determined by the value of the TXBITRATE register. The value depends on the audio sample rate desired and the data size and format (stereo/mono) used. For example, a 48 kHz sample rate for 16-bit stereo data requires a bit rate of 48 000 x 16 x 2 = 1.536 MHz.

Table 891. I2S Transmit Clock Rate register (TXBITRATE - address 0x400A 2028 (I2S0) and 0x400A 3028 (I2S1)) bit description

Bit	Symbol	Description	Reset value
5:0	TX_BITRATE	I2S transmit bit rate. This value plus one is used to divide TX_MCLK to produce the transmit bit clock.	0
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.12 I2S Receive Clock Bit Rate register

The bit rate for the I2S receiver is determined by the value of the RXBITRATE register. The value depends on the audio sample rate, as well as the data size and format used. The calculation is the same as for TXBITRATE.

Table 892. I2S Receive Clock Rate register (RXBITRATE - address 0x400A 202C (I2S0) and 0x400A 302C (I2S1)) bit description

Bit	Symbol	Description	Reset value
5:0	RX_BITRATE	I2S receive bit rate. This value plus one is used to divide RX_MCLK to produce the receive bit clock.	0
31:6	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

40.6.13 I2S Transmit Mode Control register

The Transmit Mode Control register contains additional controls for the transmit clock source, enabling the 4-pin mode (SCK and WS signals are shared between I2S transmit and receive blocks), and how MCLK is used. See Section 40.7.2 for a summary of useful mode combinations.

Table 893. I2S Transmit Mode Control register (TXMODE - address 0x400A 2030 (I2S0) and 0x400A 3030 (I2S1)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	TXCLKSEL		Clock source selection for the transmit bit clock divider.	0
		0x0	Select the TX fractional rate divider clock output as the source	
		0x1	Reserved	
		0x2	Select the RX_MCLK signal as the TX_MCLK clock source	
		0x3	Reserved	
2	TX4PIN		Transmit 4-pin mode selection (SCK and WS signals are shared between I2S transmit and receive blocks). When 1, enables 4-pin mode.	0
3	TXMCENA		Enable for the TX_MCLK output. When 0, output of TX_MCLK is not enabled. When 1, output of TX_MCLK is enabled.	0
31:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

40.6.14 I2S Receive Mode Control register

The Receive Mode Control register contains additional controls for receive clock source, enabling the 4-pin mode (SCK and WS signals are shared between I2S transmit and receive blocks), and how MCLK is used. See Section 40.7.2 for a summary of useful mode combinations.

Table 894. I2S Receive Mode Control register (RXMODE - address 0x400A 2034 (I2S0) and 0x400A 3034 (I2S1)) bit description

Bit	Symbol	Value	Description	Reset value
1:0	RXCLKSEL		Clock source selection for the receive bit clock divider.	0
		0x0	Select the RX fractional rate divider clock output as the source	
		0x1	Reserved	
		0x2	Select the TX_MCLK signal as the RX_MCLK clock source	
		0x3	Reserved	
2	RX4PIN		Receive 4-pin mode selection (SCK and WS signals are shared between I2S transmit and receive blocks). When 1, enables 4-pin mode.	0
3	RXMCENA		Enable for the RX_MCLK output. When 0, output of RX_MCLK is not enabled. When 1, output of RX_MCLK is enabled.	0
31:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

40.7 Functional description

40.7.1 I²S transmit and receive interfaces

The I2S interface can transmit and receive 8-bit, 16-bit or 32-bit stereo or mono audio information. Some details of I2S implementation are:

- When the transmit FIFO contains insufficient data the transmit channel will repeat transmitting the last data until new data is available. This can occur when the microprocessor or the DMA at some time is unable to provide new data fast enough. Because of this delay in new data there is a need to fill the gap, which is accomplished by continuing to transmit the last sample. The data is not muted as this would produce an noticeable and undesirable effect in the sound.
- When mute is true, the data value 0 is transmitted.
- When mono is false, two successive data words are respectively left and right data.
- Data word length is determined by the wordwidth value in the configuration register. There is a separate wordwidth value for the receive channel and the transmit channel.
  - 0: word is considered to contain four 8-bit data words.
  - 1: word is considered to contain two 16-bit data words.
  - 3: word is considered to contain one 32-bit data word.
- The transmit channel and the receive channel only handle 32-bit aligned words, data chunks must be clipped or extended to a multiple of 32 bits.

When switching between data width or modes the I2S must be reset via the reset bit in the control register in order to ensure correct synchronization. It is advisable to set the stop bit also until sufficient data has been written in the transmit FIFO. Note that when stopped data output is muted.

All data accesses to FIFOs are 32 bits. Figure 146 shows the possible data sequences.

Data is read from the transmit FIFO after the falling edge of WS, it will be transferred to the transmit clock domain after the rising edge of WS. On the next falling edge of WS the left data will be loaded in the shift register and transmitted and on the following rising edge of WS the right data is loaded and transmitted.

The receive channel will start receiving data after a change of WS. When word select becomes low it expects this data to be left data, when WS is high received data is expected to be right data. Reception will stop when the bit counter has reached the limit set by wordwidth. On the next change of WS the received data will be stored in the appropriate hold register. When complete data is available it will be written into the receive FIFO.

40.7.2 I²S operating modes

The clocking and WS usage of the I2S interface is configurable. In addition to master and slave modes, which are independently configurable for the transmitter and the receiver, several different clock sources are possible, including variations that share the clock and/or WS between the transmitter and receiver. This last option allows using I2S with fewer pins, typically four.

Many configurations are possible that are not considered useful, the following tables and figures give details of the configurations that are most likely to be useful.

Table 895. I2S transmit modes

DAO bit 5	TXMODE bits [3:0]	Description
0	0 0 0 0	Typical transmitter master mode. See Figure 132. The I2S transmit function operates as a master. The transmit clock source (TX_MCLK) is derived from PCLK using the fractional divider. The WS used is the internally generated TX_WS. The TX_MCLK pin is not enabled for output.
0	0 0 1 0	Transmitter master mode sharing the receiver reference clock (RX_MCLK). See Figure 134. The I2S transmit function operates as a master. The transmit clock source is RX_MCLK. The WS used is the internally generated TX_WS. The TX_MCLK pin is not enabled for output.
0	0 1 0 0	4-wire transmitter master mode sharing the receiver bit clock and WS (4-pin mode). See Figure 135. The I2S transmit function operates as a master. The transmit clock source is the RX bit clock. The WS used is the internally generated RX_WS. The TX_MCLK pin is not enabled for output.
0	1 0 0 0	Transmitter master mode with TX_MCLK output. See Figure 132. The I2S transmit function operates as a master. The transmit clock source is the fractional rate divider. The WS used is the internally generated TX_WS. The TX_MCLK pin is enabled for output.

Table 895. I2S transmit modes

DAO bit 5	TXMODE bits [3:0]	Description
1	0 0 0 0	Typical transmitter slave mode. See Figure 136. The I2S transmit function operates as a slave. The transmit clock source is provided by the external master on the TX_SCK pin. The WS used is the TX_WS pin.
1	0 0 1 0	Transmitter slave mode sharing the receiver reference clock (RX_MCLK). See Figure 137. The I2S transmit function operates as a slave. The transmit clock source is RX_SCK. The WS used is the TX_WS pin.
1	0 1 0 0	4-wire transmitter slave mode sharing the receiver bit clock and WS. See Figure 138. The I2S transmit function operates as a slave. The transmit clock source is the RX bit clock. The WS used is RX_WS ref.

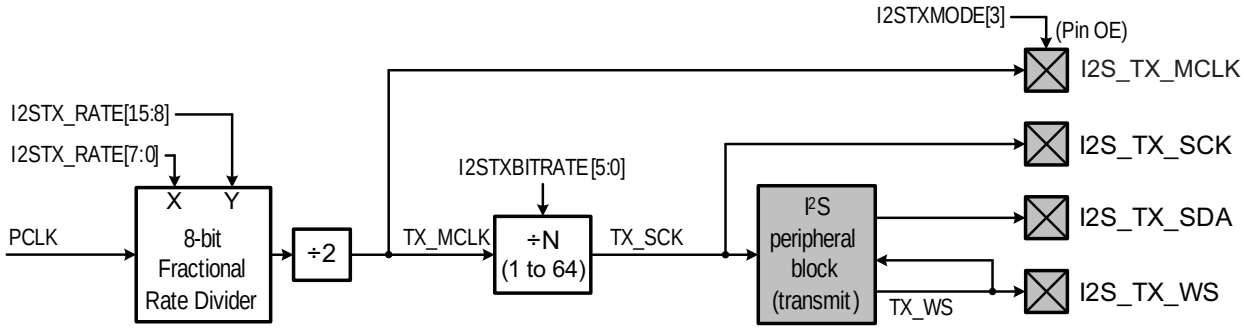


Fig 132. Typical transmitter master mode, with or without MCLK output

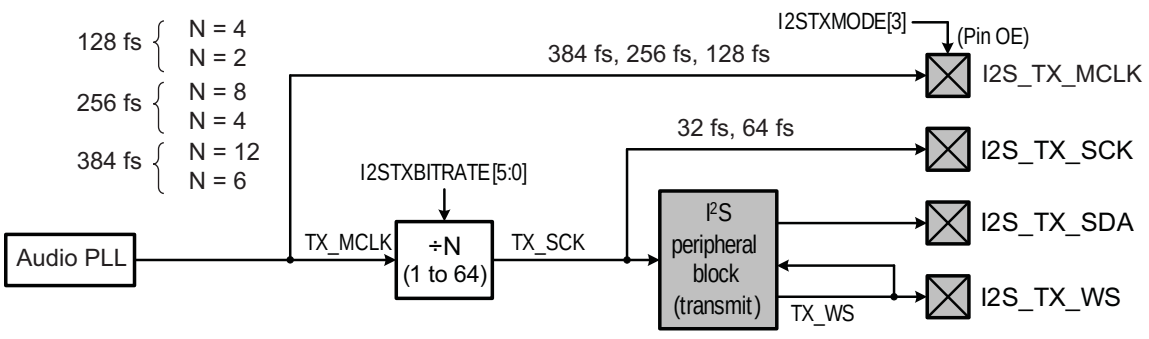


Fig 133. Typical transmitter master mode, with or without MCLK output, using the audio PLL

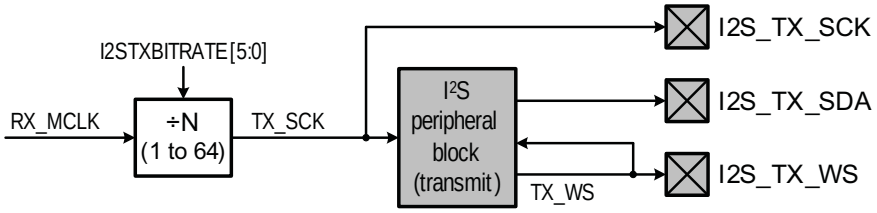


Fig 134. Transmitter master mode sharing the receiver reference clock

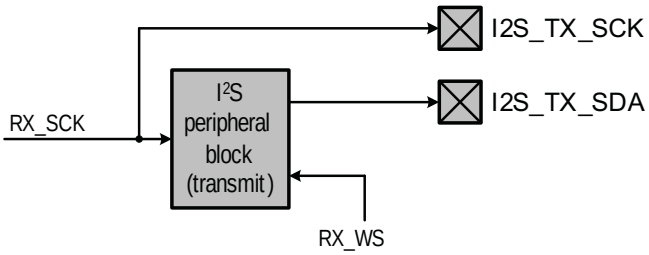


Fig 135. 4-wire transmitter master mode sharing the receiver bit clock and WS

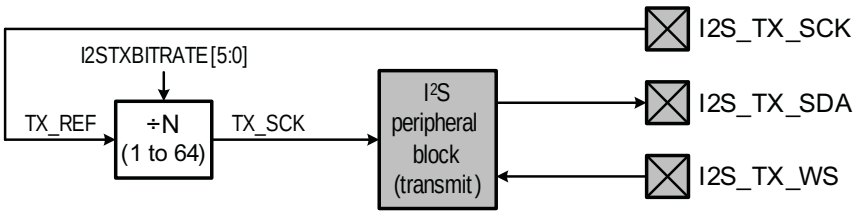


Fig 136. Typical transmitter slave mode

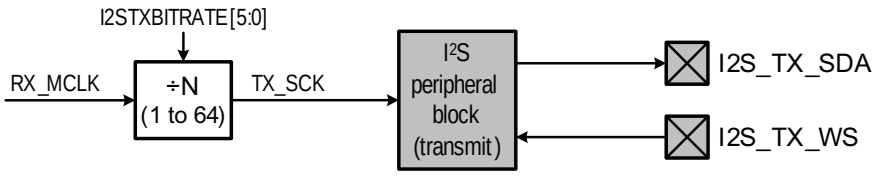


Fig 137. Transmitter slave mode sharing the receiver reference clock

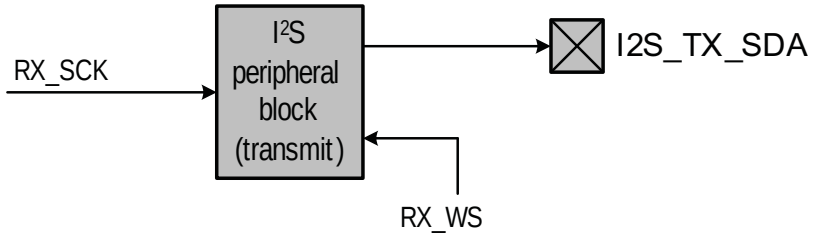


Fig 138. 4-wire transmitter slave mode sharing the receiver bit clock and WS

Table 896. I2S receive modes

DAI bit 5	RXMODE bit [3:0]	Description
0	0 0 0 0	Typical receiver master mode. See Figure 139. The I2S receive function operates as a master. The receive clock source (RX_MCLK) is derived from PCLK using the fractional divider. The WS used is the internally generated RX_WS. The RX_MCLK pin is not enabled for output.
0	0 0 1 0	Receiver master mode sharing the transmitter reference clock (TX_MCLK). See Figure 141. The I2S receive function operates as a master. The receive clock source is TX_MCLK. The WS used is the internally generated RX_WS. The RX_MCLK pin is not enabled for output.
0	0 1 0 0	4-wire receiver master mode sharing the transmitter bit clock (TX_SCK) and WS. See Figure 142. The I2S receive function operates as a master. The receive clock source is the TX bit clock (TX_SCK). The WS used is the internally generated TX_WS. The RX_MCLK pin is not enabled for output.
0	1 0 0 0	Receiver master mode with RX_MCLK output. See Figure 139. The I2S receive function operates as a master. The receive clock source (RX_MCLK) is derived from PCLK using the fractional divider. The WS used is the internally generated RX_WS. The RX_MCLK pin is enabled for output.

Table 896. I2S receive modes

DAI bit 5	RXMODE bit [3:0]	Description
1	0 0 0 0	Typical receiver slave mode. See Figure 143. The I2S receive function operates as a slave. The receive clock source is the RX_SCK pin. The WS used is the RX_WS pin.
1	0 0 1 0	Receiver slave mode sharing the transmitter reference clock (TX_MCLK). See Figure 144. The I2S receive function operates as a slave. The receive clock source is TX_MCLK. The WS used is the RX_WS pin.
1	0 1 0 0	This is a 4-wire receiver slave mode sharing the transmitter bit clock (TX_SCK) and WS. See Figure 145. The I2S receive function operates as a slave. The receive clock source is the TX bit clock (TX_SCK). The WS used is TX_WS ref.

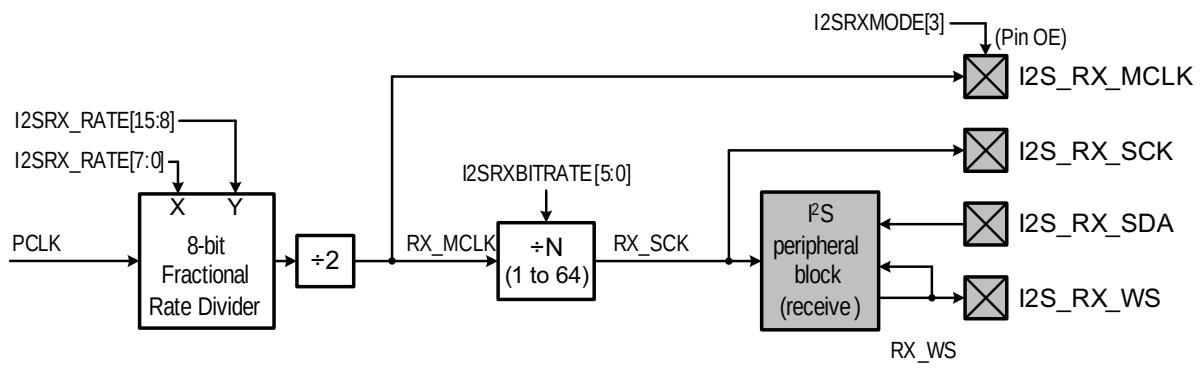


Fig 139. Typical receiver master mode, with or without MCLK output

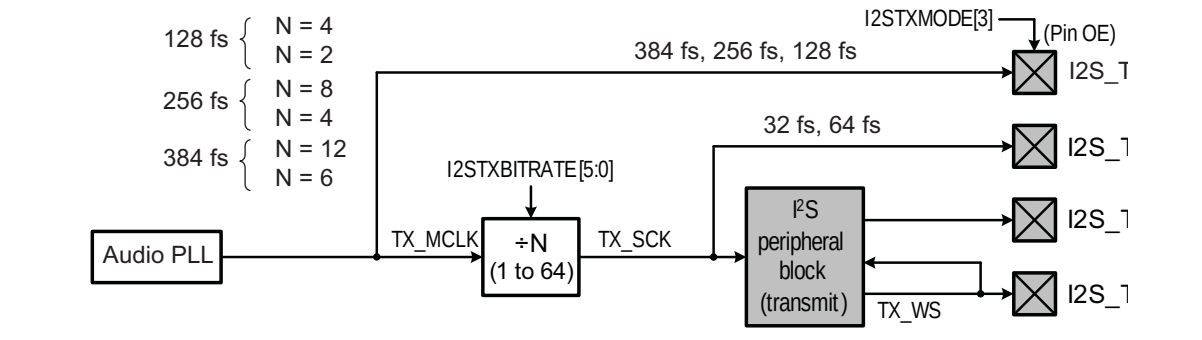


Fig 140. Typical receiver master mode, with or without MCLK output, using the audio PLL

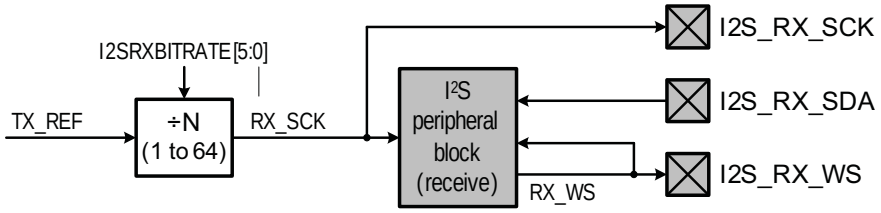


Fig 141. Receiver master mode sharing the transmitter reference clock

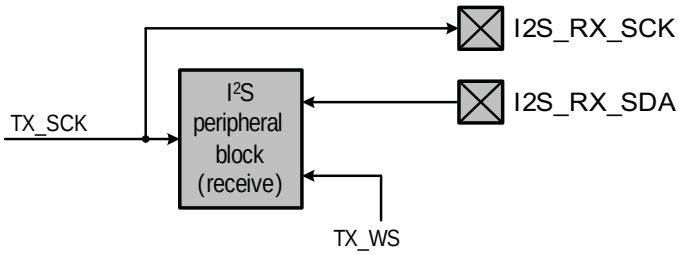


Fig 142. 4-wire receiver master mode sharing the transmitter bit clock and WS

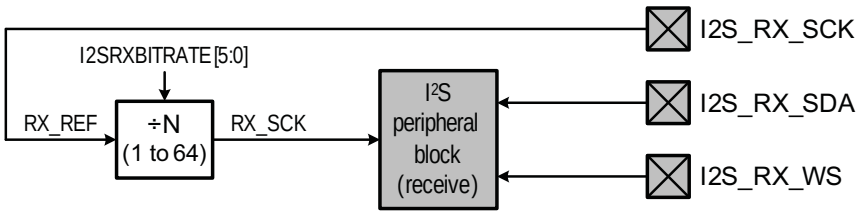


Fig 143. Typical receiver slave mode

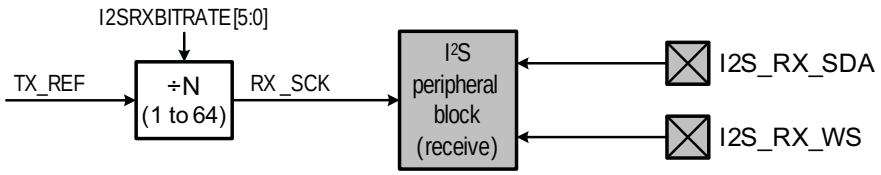


Fig 144. Receiver slave mode sharing the transmitter reference clock

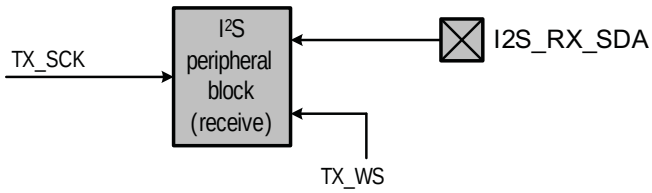


Fig 145. 4-wire receiver slave mode sharing the transmitter bit clock and WS

40.7.3 FIFO controller

Handling of data for transmission and reception is performed via the FIFO controller which can generate two DMA requests and an interrupt request. The controller consists of a set of comparators which compare FIFO levels with depth settings contained in registers. The current status of the level comparators can be seen in the APB status register.

Table 897. Conditions for FIFO level comparison

Level Comparison	Condition
dmareq_tx_1	tx_depth_dma1 >= tx_level
dmareq_rx_1	rx_depth_dma1 <= rx_level
dmareq_tx_2	tx_depth_dma2 >= tx_level
dmareq_rx_2	rx_depth_dma2 <= rx_level
irq_tx	tx_depth_irq >= tx_level
irq_rx	rx_depth_irq <= rx_level

System signaling occurs when a level detection is true and enabled.

Table 898. DMA and interrupt request generation

System Signaling	Condition
irq	(irq_rx & rx_irq_enable) | (irq_tx & tx_irq_enable)
dmareq[0]	(dmareq_tx_1 & tx_dma1_enable) | (dmareq_rx_1 & rx_dma1_enable)
dmareq[1]	(dmareq_tx_2 & tx_dma2_enable) | (dmareq_rx_2 & rx_dma2_enable)

Table 899. Status feedback in the STATE register

Status Feedback	Status
irq	irq_rx | irq_tx
dmareq1	(dmareq_tx_1 | dmareq_rx_1)
dmareq2	(dmareq_rx_2 | dmareq_tx_2)

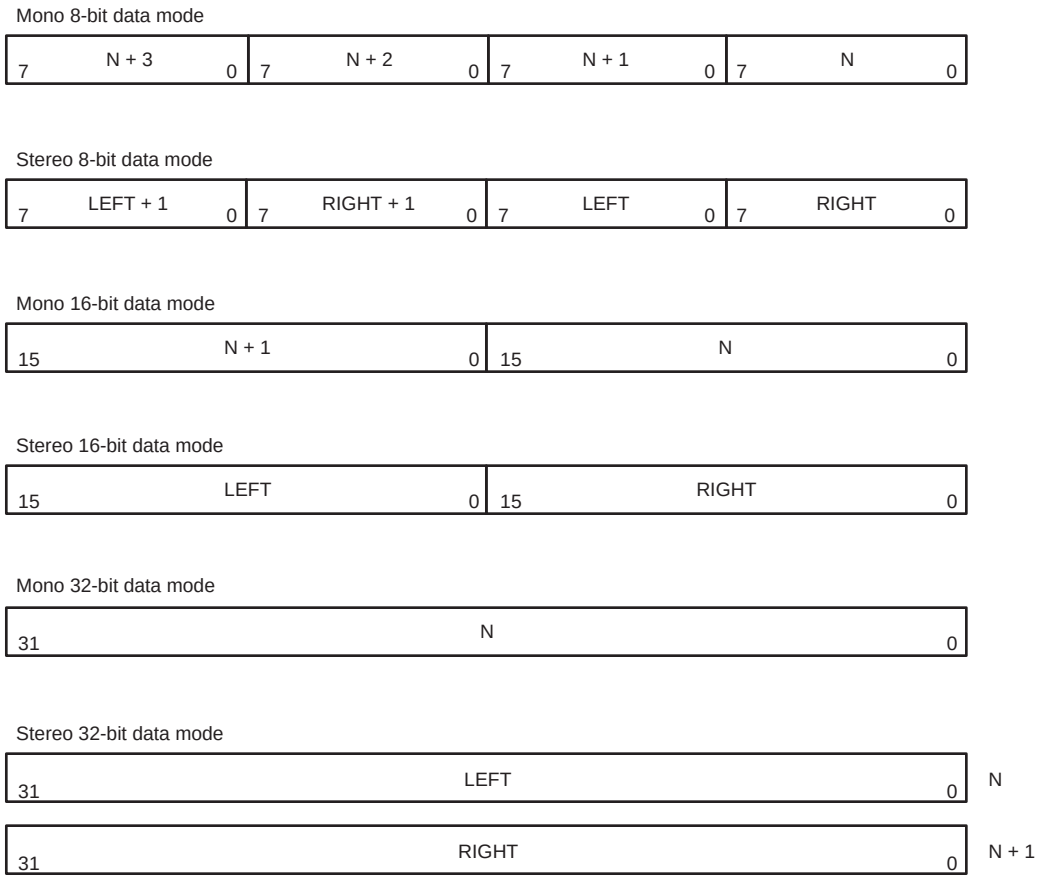


Fig 146. FIFO contents for various I^2S modes

41.1 How to read this chapter

The C_CAN0/1 controllers are available on all LPC43xx parts.

41.2 Basic configuration

The C_CAN is configured as follows:

- See Table 900 for clocking and power control.
- The C_CAN0 is reset by the CAN0_RST (reset # 55).
- The C_CAN1 is reset by the CAN1_RST (reset # 56).
- The ORed C_CAN0 and C_CAN1 interrupt is connected to slot # 12 in the Event router.
- The C_CAN0 interrupt is connected to interrupt #51 in the NVIC.
- The C_CAN1 interrupt is connected to interrupt #43 in the NVIC.

Table 900. C_CAN clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the C_CAN0 register interface and C_CAN0 peripheral clock (PCLK).	BASE_APB3_CLK	CLK_APB3_CAN0	up to 204 MHz
Clock to the C_CAN1 register interface and C_CAN1 peripheral clock (PCLK).	BASE_APB1_CLK	CLK_APB1_CAN1	up to 204 MHz

Remark: The clocks to the C_CAN0 and C_CAN1 interfaces can be set independently of each other.

41.3 Features

- Conforms to protocol version 2.0 parts A and B.
- Supports bit rate of up to 1 Mbit/s.
- Supports 32 Message Objects.
- Each Message Object has its own identifier mask.
- Provides programmable FIFO mode (concatenation of Message Objects).
- Provides maskable interrupts.
- Supports Disabled Automatic Retransmission (DAR) mode for time-triggered CAN applications.
- Provides programmable loop-back mode for self-test operation.

41.4 General description

Controller Area Network (CAN) is the definition of a high performance communication protocol for serial data communication. The C_CAN controller is designed to provide a full implementation of the CAN protocol according to the CAN Specification Version 2.0B. The C_CAN controller allows to build powerful local networks with low-cost multiplex wiring by supporting distributed real-time control with a very high level of security.

The CAN controller consists of a CAN core, message RAM, a message handler, control registers, and the APB interface.

For communication on a CAN network, individual Message Objects are configured. The Message Objects and Identifier Masks for acceptance filtering of received messages are stored in the Message RAM.

All functions concerning the handling of messages are implemented in the Message Handler. Those functions are the acceptance filtering, the transfer of messages between the CAN Core and the Message RAM, and the handling of transmission requests as well as the generation of the module interrupt.

The register set of the CAN controller can be accessed directly by an external CPU via the APB bus. These registers are used to control/configure the CAN Core and the Message Handler and to access the Message RAM.

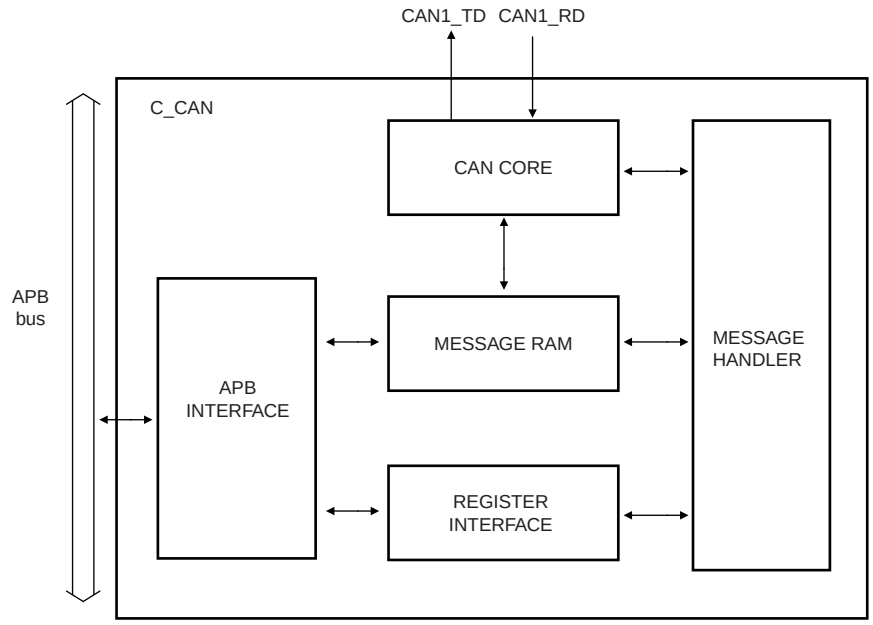


Fig 147. C_CAN block diagram

41.5 Pin description

Table 901. C_CAN pin description

Function name	Direction	Description
CAN0/1_RD	I	C_CAN receive input
CAN0/1_TD	O	C_CAN transmit output

41.6 Register description

Register values at reset

After a hardware reset, the registers hold the values described in Table 902. Additionally, the busoff state is reset and the output TD0,1 is set to recessive (HIGH). The value 0x0001 (INIT = '1') in the CAN Control Register enables the software initialization. The CAN controller does not communicate with the CAN bus until the CPU resets INIT to '0'.

The data stored in the message RAM is not affected by a hardware reset. After power-on, the contents of the message RAM is undefined.

Timing of read/write operations

Remark: Reading any of the CAN registers requires **two** consecutive read operations from the same location. Only the data from the **second** read operation are valid.

Successive read operations to the C_CAN registers must be separated by a minimum of (CLKDIVVAL × 2 + 2) × PCLK, where CLKDIVVAL is the can clock divider value and PCLK is the peripheral clock.

Successive write operations to the C_CAN registers must be separated by a minimum of (CLKDIVVAL × 2) × PCLK, where CLKDIVVAL is the can clock divider value and PCLK is the peripheral clock.

Table 902. Register overview: C_CAN0 (base address 0x400E 2000)

Name	Access	Address offset	Description	Reset value	Reference
CNTL	R/W	0x000	CAN control register	0x0001	Table 904
STAT	R/W	0x004	Status register	0x0000	Table 905
EC	RO	0x008	Error counter register	0x0000	Table 906
BT		0x00C	Bit timing register	0x2301	Table 907
INT	RO	0x010	Interrupt register	0x0000	Table 908
TEST	R/W	0x014	Test register	-	Table 909
BRPE	R/W	0x018	Baud rate prescaler extension register	0x0000	Table 910
-	-	0x01C	Reserved	-	
IF1_CMDREQ	R/W	0x020	Message interface 1 command request	0x0001	Table 913
IF1_CMDMSK_W	R/W	0x024	Message interface 1 command mask (write direction)	0x0000	Table 915
IF1_CMDMSK_R	R/W	0x024	Message interface 1 command mask (read direction)	0x0000	Table 917
IF1_MSK1	R/W	0x028	Message interface 1 mask 1	0xFFFF	Table 919
IF1_MSK2	R/W	0x02C	Message interface 1 mask 2	0xFFFF	Table 921
IF1_ARB1	R/W	0x030	Message interface 1 arbitration 1	0x0000	Table 923
IF1_ARB2	R/W	0x034	Message interface 1 arbitration 2	0x0000	Table 925
IF1_MCTRL	R/W	0x038	Message interface 1 message control	0x0000	Table 927
IF1_DA1	R/W	0x03C	Message interface 1 data A1	0x0000	Table 929
IF1_DA2	R/W	0x040	Message interface 1 data A2	0x0000	Table 930
IF1_DB1	R/W	0x044	Message interface 1 data B1	0x0000	Table 933
IF1_DB2	R/W	0x048	Message interface 1 data B2	0x0000	Table 935

Table 902. Register overview: C_CAN0 (base address 0x400E 2000)

Name	Access	Address offset	Description	Reset value	Reference
-		0x04C - 0x07C	Reserved	-	
IF2_CMDREQ	R/W	0x080	Message interface 2 command request	0x0001	Table 914
IF2_CMDMSK_W	R/W	0x084	Message interface 2 command mask (write direction)	0x0000	Table 916
IF2_CMDMSK_R	R/W	0x084	Message interface 2 command mask (read direction)	0x0000	Table 918
IF2_MSK1	R/W	0x088	Message interface 2 mask 1	0xFFFF	Table 920
IF2_MSK2	R/W	0x08C	Message interface 2 mask 2	0xFFFF	Table 922
IF2_ARB1	R/W	0x090	Message interface 2 arbitration 1	0x0000	Table 924
IF2_ARB2	R/W	0x094	Message interface 2 arbitration 2	0x0000	Table 926
IF2_MCTRL	R/W	0x098	Message interface 2 message control	0x0000	Table 928
IF2_DA1	R/W	0x09C	Message interface 2 data A1	0x0000	Table 930
IF2_DA2	R/W	0x0A0	Message interface 2 data A2	0x0000	Table 931
IF2_DB1	R/W	0x0A4	Message interface 2 data B1	0x0000	Table 934
IF2_DB2	R/W	0x0A8	Message interface 2 data B2	0x0000	Table 936
-	-	0x0AC - 0x0FC			
TXREQ1	RO	0x100	Transmission request 1	0x0000	Table 937
TXREQ2	RO	0x104	Transmission request 2	0x0000	Table 938
-	-	0x108 - 0x11C	Reserved	-	
ND1	RO	0x120	New data 1	0x0000	Table 939
ND2	RO	0x124	New data 2	0x0000	Table 940
-	-	0x128 - 0x13C	Reserved	-	
IR1	RO	0x140	Interrupt pending 1	0x0000	Table 941
IR2	RO	0x144	Interrupt pending 2	0x0000	Table 942
-	-	0x148 - 0x15C	Reserved	-	
MSGV1	RO	0x160	Message valid 1	0x0000	Table 943
MSGV2	RO	0x164	Message valid 2	0x0000	Table 944
-	-	0x168 - 0x17C	Reserved	-	
CLKDIV	R/W	0x180	CAN clock divider register	0x0001	Table 945

Table 903. Register overview: C_CAN1 (base address 0x400A 4000)

Name	Access	Address offset	Description	Reset value	Reference
CNTL	R/W	0x000	CAN control	0x0001	Table 904
STAT	R/W	0x004	Status register	0x0000	Table 905
EC	RO	0x008	Error counter	0x0000	Table 906
BT		0x00C	Bit timing register	0x2301	Table 907

Table 903. Register overview: C_CAN1 (base address 0x400A 4000)

Name	Access	Address offset	Description	Reset value	Reference
INT	RO	0x010	Interrupt register	0x0000	Table 908
TEST	R/W	0x014	Test register	-	Table 909
BRPE	R/W	0x018	Baud rate prescaler extension register	0x0000	Table 910
-	-	0x01C	Reserved	-	
IF1_CMDREQ	R/W	0x020	Message interface 1 command request	0x0001	Table 913
IF1_CMDMSK_W	R/W	0x024	Message interface 1 command mask (write direction)	0x0000	Table 915
IF1_CMDMSK_R	R/W	0x024	Message interface 1 command mask (read direction)	0x0000	Table 917
IF1_MSK1	R/W	0x028	Message interface 1 mask 1	0xFFFF	Table 919
IF1_MSK2	R/W	0x02C	Message interface 1 mask 2	0xFFFF	Table 921
IF1_ARB1	R/W	0x030	Message interface 1 arbitration 1	0x0000	Table 923
IF1_ARB2	R/W	0x034	Message interface 1 arbitration 2	0x0000	Table 925
IF1_MCTRL	R/W	0x038	Message interface 1 message control	0x0000	Table 927
IF1_DA1	R/W	0x03C	Message interface 1 data A1	0x0000	Table 929
IF1_DA2	R/W	0x040	Message interface 1 data A2	0x0000	Table 930
IF1_DB1	R/W	0x044	Message interface 1 data B1	0x0000	Table 933
IF1_DB2	R/W	0x048	Message interface 1 data B2	0x0000	Table 935
-		0x04C - 0x07C	Reserved	-	
IF2_CMDREQ	R/W	0x080	Message interface 2 command request	0x0001	Table 914
IF2_CMDMSK_W	R/W	0x084	Message interface 2 command mask (write direction)	0x0000	Table 916
IF2_CMDMSK_R	R/W	0x084	Message interface 2 command mask (read direction)	0x0000	Table 918
IF2_MSK1	R/W	0x088	Message interface 2 mask 1	0xFFFF	Table 920
IF2_MSK2	R/W	0x08C	Message interface 2 mask 2	0xFFFF	Table 922
IF2_ARB1	R/W	0x090	Message interface 2 arbitration 1	0x0000	Table 924
IF2_ARB2	R/W	0x094	Message interface 2 arbitration 2	0x0000	Table 926
IF2_MCTRL	R/W	0x098	Message interface 2 message control	0x0000	Table 928
IF2_DA1	R/W	0x09C	Message interface 2 data A1	0x0000	Table 930
IF2_DA2	R/W	0x0A0	Message interface 2 data A2	0x0000	Table 931
IF2_DB1	R/W	0x0A4	Message interface 2 data B1	0x0000	Table 934
IF2_DB2	R/W	0x0A8	Message interface 2 data B2	0x0000	Table 936
-	-	0x0AC - 0x0FC			
TXREQ1	RO	0x100	Transmission request 1	0x0000	Table 937
TXREQ2	RO	0x104	Transmission request 2	0x0000	Table 938
-	-	0x108 - 0x11C	Reserved	-	
ND1	RO	0x120	New data 1	0x0000	Table 939
ND2	RO	0x124	New data 2	0x0000	Table 940

Table 903. Register overview: C_CAN1 (base address 0x400A 4000)

Name	Access	Address offset	Description	Reset value	Reference
-	-	0x128 - 0x13C	Reserved	-	
IR1	RO	0x140	Interrupt pending 1	0x0000	Table 941
IR2	RO	0x144	Interrupt pending 2	0x0000	Table 942
-	-	0x148 - 0x15C	Reserved	-	
MSGV1	RO	0x160	Message valid 1	0x0000	Table 943
MSGV2	RO	0x164	Message valid 2	0x0000	Table 944
-	-	0x168 - 0x17C	Reserved	-	
CLKDIV	R/W	0x180	CAN clock divider register	0x0001	Table 945

41.6.1 CAN protocol registers

41.6.1.1 CAN control register

After a hardware reset, the registers of the C_CAN controller hold the values described in Table 902. Additionally, the busoff state is set, and the TD0/1 outputs are set to HIGH. The reset value 0x0001 of the CANCTRL register enables initialization by software (INIT = 1). The C_CAN does not influence the CAN bus until the CPU resets the INIT bit to 0.

Table 904. CAN control registers (CNTL, address 0x400E 2000 (C_CAN0) and 0x400A 4000 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	INIT		Initialization	1	R/W
		0	Normal operation.		
		1	Initialization is started. On reset, software needs to initialize the CAN controller.		
1	IE		Module interrupt enable	0	R/W
		0	Disable CAN interrupts. The interrupt line is always HIGH.		
		1	Enable CAN interrupts. The interrupt line is set to LOW and remains LOW until all pending interrupts are cleared.		
2	SIE		Status change interrupt enable	0	R/W
		0	Disable status change interrupts. No status change interrupt will be generated.		
		1	Enable status change interrupts. A status change interrupt will be generated when a message transfer is successfully completed or a CAN bus error is detected.		

Table 904. CAN control registers (CNTL, address 0x400E 2000 (C_CAN0) and 0x400A 4000 (C_CAN1)) bit description

…continued

Bit	Symbol	Value	Description	Reset value	Access
3	EIE		Error interrupt enable	0	R/W
		0	Disable error interrupt. No error status interrupt will be generated.		
		1	Enable error interrupt. A change in the bits BOFF or EWARN in the CANSTAT registers will generate an interrupt.		
4	-	-	Reserved	0	-
5	DAR		Disable automatic retransmission	0	R/W
		0	Automatic retransmission of disturbed messages enabled.		
		1	Automatic retransmission disabled.		
6	CCE		Configuration change enable	0	R/W
		0	The CPU has no write access to the bit timing register.		
		1	The CPU has write access to the CANBT register while the INIT bit is one.		
7	TEST		Test mode enable	0	R/W
		0	Normal operation.		
		1	Test mode.		
31:8	-		reserved	-	-

Remark: The busoff recovery sequence (see *CAN Specification Rev. 2.0*) cannot be shortened by setting or resetting the INIT bit. If the device goes into busoff state, it will set INIT, stopping all bus activities. Once INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129×11 consecutive HIGH/recessive bits) before resuming normal operations. At the end of the busoff recovery sequence, the Error Management Counters will be reset.

During the waiting time after the resetting of INIT, each time a sequence of 11 HIGH/recessive bits has been monitored, a Bit0Error code is written to the Status Register CANSTAT, enabling the CPU to monitor the proceeding of the busoff recovery sequence and to determine whether the CAN bus is stuck at LOW/dominant or continuously disturbed.

41.6.1.2 CAN status register

Table 905. CAN status register (STAT, address 0x400E 2004 (C_CAN0) and 0x400A 4004 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
2:0	LEC		Last error code Type of the last error to occur on the CAN bus.The LEC field holds a code which indicates the type of the last error to occur on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. The unused code '111' may be written by the CPU to check for updates.	000	R/W
		0x0	**No error**.		
		0x1	**Stuff error**: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.		
		0x2	**Form error**: A fixed format part of a received frame has the wrong format.		
		0x3	**AckError**: The message this CAN core transmitted was not acknowledged.		
		0x4	**Bit1Error**: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a HIGH/recessive level (bit of logical value '1'), but the monitored bus value was LOW/dominant.		
		0x5	**Bit0Error**: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a LOW/dominant level (data or identifier bit logical value '0'), but the monitored Bus value was HIGH/recessive. During busoff recovery this status is set each time a sequence of 11 HIGH/recessive bits has been monitored. This enables the CPU to monitor the proceeding of the busoff recovery sequence (indicating the bus is not stuck at LOW/dominant or continuously disturbed).		
		0x6	**CRCError**: The CRC checksum was incorrect in the message received.		
		0x7	**Unused:** No CAN bus event was detected (written by the CPU).		
3	TXOK		Transmitted a message successfully This bit is reset by the CPU. It is never reset by the CAN controller.	0	R/W
		0	Since this bit was reset by the CPU, no message has been successfully transmitted.		
		1	Since this bit was last reset by the CPU, a message has been successfully transmitted (error free and acknowledged by at least one other node).		
4	RXOK		Received a message successfully This bit is reset by the CPU. It is never reset by the CAN controller.	0	R/W
		0	Since this bit was last reset by the CPU, no message has been successfully transmitted.		
		1	Since this bit was last set to zero by the CPU, a message has been successfully received independent of the result of acceptance filtering.		

Table 905. CAN status register (STAT, address 0x400E 2004 (C_CAN0) and 0x400A 4004 (C_CAN1)) bit description
…continued

Bit	Symbol	Value	Description	Reset value	Access
5	EPASS		Error passive	0	RO
		0	The CAN controller is in the error active state.		
		1	The CAN controller is in the error passive state as defined in the *CAN 2.0 specification*.		
6	EWARN		Warning status	0	RO
		0	Both error counters are below the error warning limit of 96.		
		1	At least one of the error counters in the EML has reached the error warning limit of 96.		
7	BOFF		Busoff status	0	RO
		0	The CAN module is not in busoff state.		
		1	The CAN controller is in busoff state.		
31:8	-	-	reserved		

A status interrupt is generated by bits BOFF, EWARN, RXOK, TXOK, or LEC. BOFF and EWARN generate an error interrupt, and RXOK, TXOK, and LEC generate a status change interrupt if EIE and SIE respectively are set to enabled in the CANCTRL register.

A change of bit EPASS and a write to RXOK, TXOK, or LEC will never create a status interrupt.

Reading the CANSTAT register will clear the Status Interrupt value in the CANIR register.

41.6.1.3 CAN error counter

Table 906. CAN error counter (EC, address 0x400E 2008 (C_CAN0) and 0x400A 4008 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
7:0	TEC_7_0		Transmit error counter Current value of the transmit error counter (maximum value 127)	0	RO
14:8	REC_6_0		Receive error counter Current value of the receive error counter (maximum value 255).	0	RO
15	RP		Receive error passive	0	RO
		0	The receive counter is below the error passive level.		
		1	The receive counter has reached the error passive level as defined in the *CAN2.0 specification*.		
31:16	-	-	Reserved	-	-

41.6.1.4 CAN bit timing register

Table 907. CAN bit timing register (BT, address 0x400E 200C (C_CAN0) and 0x400A 400C (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
5:0	BRP	Baud rate prescaler The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are 0 to 63[1]. Valid programmed values are 0x01 - 0x3F[1].	1	R/W
7:6	SJW	(Re)synchronization jump width Valid programmed values are 0 to 3[1].	0	R/w
11:8	TSEG1	Time segment after the sample point Valid values are 0 to 7[1].	0011	R/W
14:12	TSEG2	Time segment before the sample point Valid values are 1 to 15[1].	010	R/W
31:15	-	Reserved	-	-

[1] Hardware interprets the value programmed into these bits as the bit value + 1.

Remark: With a module clock CAN_CLK of 8 MHz, the reset value of 0x2301 configures the C_CAN for a bit rate of 250 kBit/s. The registers are only writable if a configuration change is enabled in CANCTRL and the controller is initialized by software (bits CCE and INIT in the CAN Control Register are set).

41.6.1.5 CAN interrupt register

Table 908. CAN interrupt register (INT, address 0x400E 2010 (C_CAN0) and 0x400A 4010 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	INTID15_0	0x0000 = No interrupt is pending 0x0001 to 0x0020 = Number of message object which caused the interrupt. 0x0021 to 0x7FFF = Unused 0x8000 = Status interrupt 0x8001 to 0xFFFF = Unused	0	R
31:16	-	Reserved	-	-

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it. If INTID is different from 0x0000 and IE is set, the interrupt line to the CPU is active. The interrupt line remains active until INTID is back to value 0x0000 (the cause of the interrupt is reset) or until IE is reset.

The Status Interrupt has the highest priority. Among the message interrupts, the Message Object' s interrupt priority decreases with increasing message number.

A message interrupt is cleared by clearing the Message Object's INTPND bit. The StatusInterrupt is cleared by reading the Status Register.

41.6.1.6 CAN test register

Write access to the Test Register is enabled by setting bit Test in the CAN Control Register.

The different test functions may be combined, but when TX[1:0] ≠ “00” is selected, the message transfer is disturbed.

Table 909. CAN test register (TEST, address 0x400E 2014 (C_CAN0) and 0x400A 4014 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
1:0	-	-			-
2	BASIC		Basic mode	0	R/W
		0	Basic mode disabled.		
		1	IF1 registers used as TX buffer, IF2 registers used as RX buffer.		
3	SILENT		Silent mode	0	R/W
		0	Normal operation.		
		1	The module is in silent mode.		
4	LBACK		Loop back mode	0	R/W
		0	Loop back mode is disabled.		
		1	Loop back mode is enabled.		
6:5	TX1_0		Control of TD pins	00	R/W
		0x0	Level at the TD pin is controlled by the CAN controller. This is the value at reset.		
		0x1	The sample point can be monitored at the TD pin.		
		0x2	TD pin is driven LOW/dominant.		
		0x3	TD pin is driven HIGH/recessive.		
7	RX		Monitors the actual value of the RD Pin	0	R
		0	The CAN bus is dominant (RD = 0).		
		1	The CAN bus is recessive (RD = 1).		
31:8	-		Reserved		-

41.6.1.7 CAN baud rate prescaler extension register

Table 910. CAN baud rate prescaler extension register (BRPE, address 0x400E 2018 (C_CAN0) and 0x400A 4018 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
3:0	BRPE	Baud rate prescaler extension By programming BRPE the Baud Rate Prescaler can be extended to values up to 1023. Hardware interprets the value as the value of BRPE (MSBs) and BRP (LSBs) plus one. Allowed values are 0x00 to 0x0F	0x0000	R/W
31:4	-	Reserved	-	-

41.6.2 Message interface registers

There are two sets of interface registers which are used to control the CPU access to the Message RAM. The interface registers avoid conflicts between CPU access to the Message RAM and CAN message reception and transmission by buffering the data to be transferred. A complete Message Object (see Section 41.6.2.1) or parts of the Message Object may be transferred between the Message RAM and the IFx Message Buffer registers in one single transfer.

The function of the two interface register sets is identical (except for test mode Basic). One set of registers may be used for data transfer to the Message RAM while the other set of registers may be used for the data transfer from the Message RAM, allowing both processes to be interrupted by each other.

Each set of interface registers consists of message buffer registers controlled by their own command registers. The command mask register specifies the direction of the data transfer and which parts of a message object will be transferred. The command request register is used to select a message object in the message RAM as target or source for the transfer and to start the action specified in the command mask register.

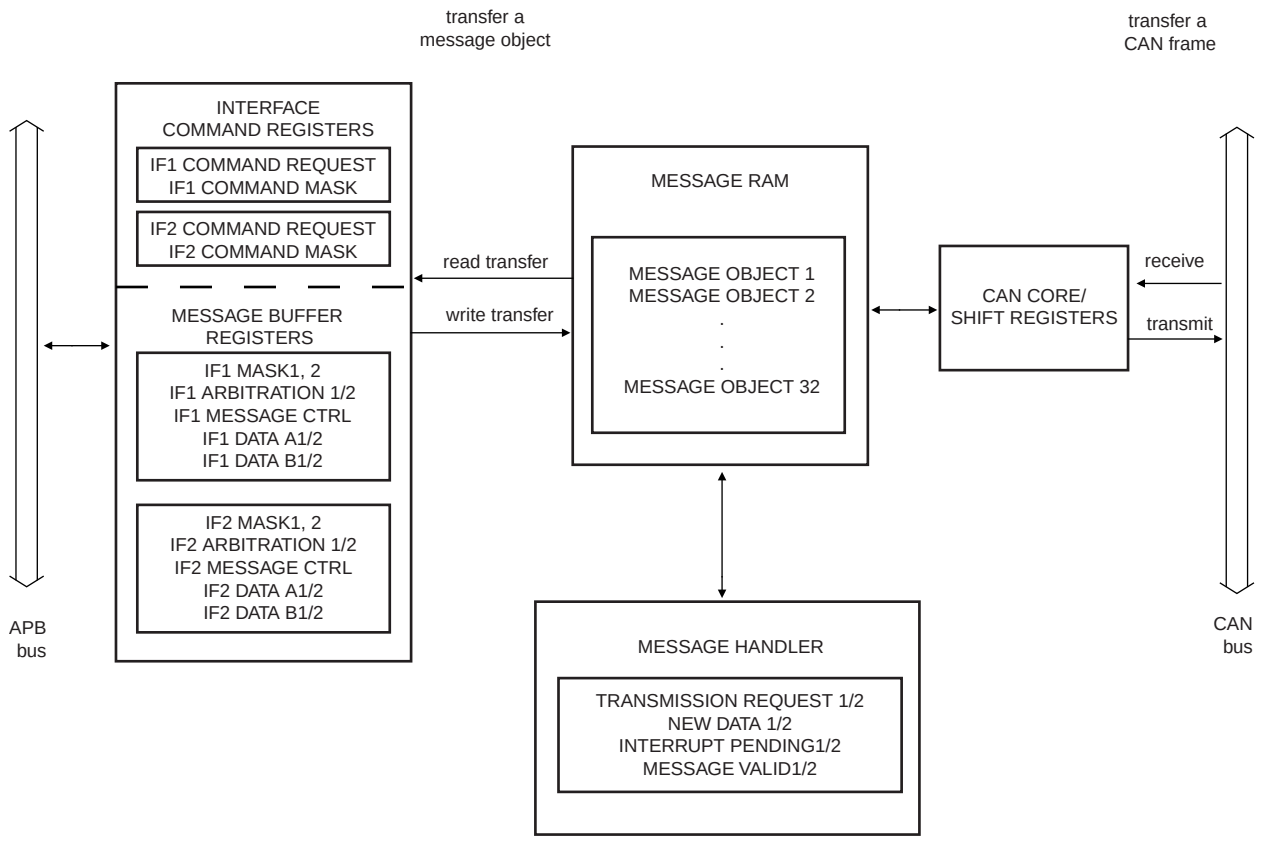


Fig 148. Block diagram of a message object transfer

Table 911. Message interface registers

IF1 register names	IF1 register set	IF2 register names	IF2 register set
IF1_CMDREQ	IF1 command request	IF2_CMDREQ	IF2 command request
IF1_CMDMASK	IF1 command mask	IF2_CMDMASK	IF2 command mask
IF1_MASK1	IF1 mask 1	IF2_MSK1	IF2 mask 1
IF1_MASK2	IF1 mask 2	IF2_MSK2	IF2 mask 2
IF1_ARB1	IF1 arbitration 1	IF2_ARB1	IF2 arbitration 1
IF1_ARB2	IF1 arbitration 2	IF2_ARB2	IF2 arbitration 2
IF1_MCTRL	IF1 message control	IF2_MCTRL	IF2 message control
IF1_DA1	IF1 data A1	IF2_DA1	IF2 data A1
IF1_DA2	IF1 data A2	IF2_DA2	IF2 data A2
CIF1_DB1	IF1 data B1	IF2_DB1	IF2 data B1
IF1_DB2	IF1 data B2	IF2_DB2	IF2 data B2

There are 32 Message Objects in the Message RAM. To avoid conflicts between CPU access to the Message RAM and CAN message reception and transmission, the CPU cannot directly access the Message Objects. The message objects are accessed through the IFx Interface Registers.

41.6.2.1 Message objects

A message object contains the information from the various bits in the message interface registers. Table 912 below shows a schematic representation of the structure of the message object. The bits of a message object and the respective interface register where this bit is set or cleared are shown. For bit functions see the corresponding interface register.

Table 912. Structure of a message object in the message RAM

UMASK	MSK[28:0]	MXTD	MDIR	EOB	NEWDAT	MSGLST	RXIE	TXIE	INTPND
IF1/2_MCTRL	IF1/2_MSK1/2			IF1/2_MCTRL					
RMTEN	**TXRQST**	**MSGVAL**	**ID[28:0]**	**XTD**	**DIR**	**DLC3**	**DLC2**	**DLC1**	**DLC0**
IF1/2_MCTRL		IF1/2_ARB1/2				IF1/2_MCTRL			
DATA0	**DATA1**	**DATA2**	**DATA3**	**DATA4**	**DATA5**	**DATA6**	**DATA7**		
IF1/2_DA1		IF1/2_DA2		IF1/2_DB1		IF1/2_DB2			

41.6.2.2 CAN message interface command request registers

A message transfer is started as soon as the CPU has written the message number to the Command Request Register. With this write operation the BUSY bit is automatically set to ‘1’ and the signal CAN_WAIT_B is pulled LOW) to notify the CPU that a transfer is in progress. After a wait time of 3 to 6 CAN_CLK periods, the transfer between the Interface Register and the Message RAM has completed. The BUSY bit is set back to zero and the signal CAN_WAIT_B is set back).

Table 913. CAN message interface command request registers (IF1_CMDREQ, address 0x400E 2020 (C_CAN0) and 0x400A 4020 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
5:0	Message Number	Message number 0x01 to 0x20 = Valid message numbers The message object in the message RAM is selected for data transfer. 0x00 = Not a valid message number. This value is interpreted as 0x20.[1] 0x21 to 0x3F = Not a valid message number. This value is interpreted as 0x01 - 0x1F.[1]	0x01	R/W
14:6	-	Reserved		
15	BUSY	BUSY flag Set to one by hardware when writing to this Command request register. Set to zero by hardware when read/write action to this Command request register has finished.	0	R
31:16	-	Reserved	-	-

[1] When a message number that is not valid is written into the Command request registers, the message number will be transformed into a valid value and that message object will be transferred.

Table 914. CAN message interface command request registers (IF2_CMDREQ, address 0x400E 2080 (C_CAN0) and 0x400A 4080 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
5:0	Message Number	Message number 0x01 to 0x20 = Valid message numbers The message object in the message RAM is selected for data transfer. 0x00 = Not a valid message number. This value is interpreted as 0x20.[1] 0x21 to 0x3F = Not a valid message number. This value is interpreted as 0x01 - 0x1F.[1]	0x01	R/W
14:6	-	Reserved		
15	BUSY	BUSY flag Set to one by hardware when writing to this Command request register. Set to zero by hardware when read/write action to this Command request register has finished.	0	R
31:16	-	Reserved	-	-

[1] When a message number that is not valid is written into the Command request registers, the message number will be transformed into a valid value and that message object will be transferred.

41.6.2.3 CAN message interface command mask registers

The control bits of the IFx Command Mask Register specify the transfer direction and select which of the IFx Message Buffer Registers are source or target of the data transfer.The functions of the register bits depend on the transfer direction (read or write) which is selected in the WR/RD bit (bit 7) of this Command mask register.

Select the WR/RD to

one for the Write transfer direction (write to message RAM)

zero for the Read transfer direction (read from message RAM)

Transfer direction Write

Table 915. CAN message interface command mask registers write direction (IF1_CMDMSK_W, address 0x400E 2024 (C_CAN0) and 0x400A 4024 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DATA_B		Access data bytes 4-7	0	R/W
		0	Data bytes 4-7 unchanged.		
		1	Transfer data bytes 4-7 to message object.		
1	DATA_A		Access data bytes 0-3	0	R/W
		0	Data bytes 0-3 unchanged.		
		1	Transfer data bytes 0-3 to message object.		
2	TXRQST		Access transmission request bit	0	R/W
		0	No transmission request. TXRQSRT bit unchanged in IF1/2_MCTRL. **Remark:** If a transmission is requested by programming this bit, the TXRQST bit in the CANIFn_MCTRL register is ignored.		
		1	Request a transmission. Set the TXRQST bit IF1/2_MCTRL.		
3	CLRINTPND	-	This bit is ignored in the write direction.	0	R/W
4	CTRL		Access control bits	0	R/W
		0	Control bits unchanged.		
		1	Transfer control bits to message object		
5	ARB		Access arbitration bits	0	R/W
		0	Arbitration bits unchanged.		
		1	Transfer Identifier, DIR, XTD, and MSGVAL bits to message object.		

Table 915. CAN message interface command mask registers write direction (IF1_CMDMSK_W, address 0x400E 2024 (C_CAN0) and 0x400A 4024 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
6	MASK		Access mask bits	0	R/W
		0	Mask bits unchanged.		
		1	Transfer Identifier MASK + MDIR + MXTD to message object.		
7	WR_RD	1	Write transfer Transfer data from the selected message buffer registers to the message object addressed by the command request register CANIFn_CMDREQ.	0	R/W
31:8	-	-	reserved	0	-

Table 916. CAN message interface command mask registers write direction (IF2_CMDMSK_W, address 0x400E 2084 (C_CAN0) and 0x400A 4084 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DATA_B		Access data bytes 4-7	0	R/W
		0	Data bytes 4-7 unchanged.		
		1	Transfer data bytes 4-7 to message object.		
1	DATA_A		Access data bytes 0-3	0	R/W
		0	Data bytes 0-3 unchanged.		
		1	Transfer data bytes 0-3 to message object.		
2	TXRQST		Access transmission request bit	0	R/W
		0	No transmission request. TXRQSRT bit unchanged in IF1/2_MCTRL. **Remark:** If a transmission is requested by programming this bit, the TXRQST bit in the CANIFn_MCTRL register is ignored.		
		1	Request a transmission. Set the TXRQST bit IF1/2_MCTRL.		
3	CLRINTPND	-	This bit is ignored in the write direction.	0	R/W
4	CTRL		Access control bits	0	R/W
		0	Control bits unchanged.		
		1	Transfer control bits to message object		
5	ARB		Access arbitration bits	0	R/W
		0	Arbitration bits unchanged.		
		1	Transfer Identifier, DIR, XTD, and MSGVAL bits to message object.		

Table 916. CAN message interface command mask registers write direction (IF2_CMDMSK_W, address 0x400E 2084 (C_CAN0) and 0x400A 4084 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
6	MASK		Access mask bits	0	R/W
		0	Mask bits unchanged.		
		1	Transfer Identifier MASK + MDIR + MXTD to message object.		
7	WR_RD	1	Write transfer Transfer data from the selected message buffer registers to the message object addressed by the command request register CANIFn_CMDREQ.	0	R/W
31:8	-	-	reserved	0	-

Transfer direction Read

Table 917. CAN message interface command mask registers read direction (IF1_CMDMSK_R, address 0x400E 2024 (C_CAN0) and 0x400A 4024 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DATA_B		Access data bytes 4-7	0	R/W
		0	data bytes 4-7 unchanged.		
		1	Transfer data bytes 4-7 to IFx message buffer register.		
1	DATA_A		Access data bytes 0-3	0	R/W
		0	data bytes 0-3 unchanged.		
		1	Transfer data bytes 0-3 to IFx message buffer.		
2	NEWDAT		Access new data bit	0	R/W
		0	NEWDAT bit remains unchanged. **Remark:** A read access to a message object can be combined with the reset of the control bits INTPND and NEWDAT in IF1/2_MCTRL. The values of these bits transferred to the IFx Message Control Register always reflect the status before resetting these bits.		
		1	Clear NEWDAT bit in the message object.		
3	CLRINTPND		Clear interrupt pending bit.	0	R/W
		0	INTPND bit remains unchanged.		
		1	Clear INTPND bit in the message object.		
4	CTRL		Access control bits	0	R/W
		0	Control bits unchanged.		
		1	Transfer control bits to IFx message buffer.		

Table 917. CAN message interface command mask registers read direction (IF1_CMDMSK_R, address 0x400E 2024 (C_CAN0) and 0x400A 4024 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
5	ARB		Access arbitration bits	0	R/W
		0	Arbitration bits unchanged.		
		1	Transfer Identifier, DIR, XTD, and MSGVAL bits to IFx message buffer register.		
6	MASK		Access mask bits	0	R/W
		0	Mask bits unchanged.		
		1	Transfer Identifier MASK + MDIR + MXTD to IFx message buffer register.		
7	WR_RD	0	Read transfer Transfer data from the message object addressed by the command request register to the selected message buffer registers CANIFn_CMDREQ.	0	R/W
31:8	-	-	reserved	0	-

Table 918. CAN message interface command mask registers read direction (IF2_CMDMSK_R, address 0x400E 2084 (C_CAN0) and 0x400A 4084 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
0	DATA_B		Access data bytes 4-7	0	R/W
		0	data bytes 4-7 unchanged.		
		1	Transfer data bytes 4-7 to IFx message buffer register.		
1	DATA_A		Access data bytes 0-3	0	R/W
		0	data bytes 0-3 unchanged.		
		1	Transfer data bytes 0-3 to IFx message buffer.		
2	NEWDAT		Access new data bit	0	R/W
		0	NEWDAT bit remains unchanged. **Remark:** A read access to a message object can be combined with the reset of the control bits INTPND and NEWDAT in IF1/2_MCTRL. The values of these bits transferred to the IFx Message Control Register always reflect the status before resetting these bits.		
		1	Clear NEWDAT bit in the message object.		
3	CLRINTPND		Clear interrupt pending bit.	0	R/W
		0	INTPND bit remains unchanged.		
		1	Clear INTPND bit in the message object.		
4	CTRL		Access control bits	0	R/W
		0	Control bits unchanged.		
		1	Transfer control bits to IFx message buffer.		

Table 918. CAN message interface command mask registers read direction (IF2_CMDMSK_R, address 0x400E 2084 (C_CAN0) and 0x400A 4084 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
5	ARB		Access arbitration bits	0	R/W
		0	Arbitration bits unchanged.		
		1	Transfer Identifier, DIR, XTD, and MSGVAL bits to IFx message buffer register.		
6	MASK		Access mask bits	0	R/W
		0	Mask bits unchanged.		
		1	Transfer Identifier MASK + MDIR + MXTD to IFx message buffer register.		
7	WR_RD	0	Read transfer Transfer data from the message object addressed by the command request register to the selected message buffer registers CANIFn_CMDREQ.	0	R/W
31:8	-	-	reserved	0	-

41.6.2.4 IF1 and IF2 message buffer registers

The bits of the Message Buffer registers mirror the Message Objects in the Message RAM.

41.6.2.4.1 CAN message interface command mask 1 registers

Table 919. CAN message interface command mask 1 registers (IF1_MSK1, address 0x400E 2028 (C_CAN0) and 0x400A 4028 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	MSK15_0	Identifier mask 0 = The corresponding bit in the identifier of the message can not inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.	0xFFFF	R/W
31:16	-	reserved	0	-

Table 920. CAN message interface command mask 1 registers (IF2_MSK1, address 0x400E 2088 (C_CAN0) and 0x400A 4088 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	MSK15_0	Identifier mask 0 = The corresponding bit in the identifier of the message can not inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.	0xFFFF	R/W
31:16	-	reserved	0	-

41.6.2.4.2 CAN message interface command mask 2 registers

Table 921. CAN message interface command mask 2 registers (IF1_MSK2, address 0x400E 202C (C_CAN0) and 0x400A 402C (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
12:0	MSK28_16		Identifier mask 0 = The corresponding bit in the identifier of the message can not inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.	0xFFF	R/W
13	-		Reserved	1	-
14	MDIR		Mask message direction	1	R/W
		0	The message direction bit (DIR) has no effect on acceptance filtering.		
		1	The message direction bit (DIR) is used for acceptance filtering.		
15	MXTD		Mask extend identifier	1	R/W
		0	The extended identifier bit (IDE) has no effect on acceptance filtering.		
		1	The extended identifier bit (IDE) is used for acceptance filtering.		
31:16	-	-	Reserved	0	-

Table 922. CAN message interface command mask 2 registers (IF2_MSK2, 0x400E 208C (C_CAN0) and 0x400A 408C (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
12:0	MSK28_16		Identifier mask 0 = The corresponding bit in the identifier of the message can not inhibit the match in the acceptance filtering. 1 = The corresponding identifier bit is used for acceptance filtering.	0xFFF	R/W
13	-		Reserved	1	-
14	MDIR		Mask message direction	1	R/W
		0	The message direction bit (DIR) has no effect on acceptance filtering.		
		1	The message direction bit (DIR) is used for acceptance filtering.		
15	MXTD		Mask extend identifier	1	R/W
		0	The extended identifier bit (IDE) has no effect on acceptance filtering.		
		1	The extended identifier bit (IDE) is used for acceptance filtering.		
31:16	-	-	Reserved	0	-

41.6.2.4.3 CAN message interface command arbitration 1 registers

Table 923. CAN message interface command arbitration 1 registers (IF1_ARB1, address 0x400E 2030 (C_CAN0) and 0x400A 4030 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	ID15_0	Message identifier 29-bit identifier (“extended frame”) 11-bit identifier (“standard frame”)	0x00	R/W
31:16	-	Reserved	0	-

Table 924. CAN message interface command arbitration 1 registers (IF2_ARB1, address 0x400E 2090 (C_CAN0) and 0x400A 4090 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	ID15_0	Message identifier 29-bit identifier (“extended frame”) 11-bit identifier (“standard frame”)	0x00	R/W
31:16	-	Reserved	0	-

41.6.2.4.4 CAN message interface command arbitration 2 registers

Table 925. CAN message interface command arbitration 2 registers (IF1_ARB2, address 0x400E 2034 (C_CAN0) and 0x400A 4034 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
12:0	ID28_16		Message identifier 29-bit identifier (“extended frame”) 11-bit identifier (“standard frame”)	0x00	R/W
13	DIR		Message direction	0x00	R/W
		0	Direction = receive. On TXRQST, a Remote Frame with the identifier of this Message Object is transmitted. On reception of a Data Frame with matching identifier, that message is stored in this Message Object.		
		1	Direction = transmit. On TXRQST, the respective Message Object is transmitted as a Data Frame. On reception of a Remote Frame with matching identifier, the TXRQST bit of this Message Object is set (if RMTEN = one).		
14	XTD		Extend identifier	0x00	R/W
		0	The 11-bit standard identifier will be used for this message object.		
		1	The 29-bit extended identifier will be used for this message object.		

Table 925. CAN message interface command arbitration 2 registers (IF1_ARB2, address 0x400E 2034 (C_CAN0) and 0x400A 4034 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
15	MSGVAL		Message valid **Remark:** The MSGVAL bit of all unused Messages Objects is reset during the initialization before bit INIT is reset in the CAN Control Register. This bit must be set to zero before the identifier ID28:0, the control bits XTD, DIR, or the Data Length Code DLC3:0 are modified, or if the Messages Object is no longer required.	0	R/W
		0	The message object is ignored by the message handler.		
		1	The message object is configured and should be considered by the message handler.		
31:16	-	-	Reserved	0	-

Table 926. CAN message interface command arbitration 2 registers (IF2_ARB2, address 0x400E 2094 (C_CAN0) and 0x400A 4094 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
12:0	ID28_16		Message identifier 29-bit identifier (“extended frame”) 11-bit identifier (“standard frame”)	0x00	R/W
13	DIR		Message direction	0x00	R/W
		0	Direction = receive. On TXRQST, a Remote Frame with the identifier of this Message Object is transmitted. On reception of a Data Frame with matching identifier, that message is stored in this Message Object.		
		1	Direction = transmit. On TXRQST, the respective Message Object is transmitted as a Data Frame. On reception of a Remote Frame with matching identifier, the TXRQST bit of this Message Object is set (if RMTEN = one).		
14	XTD		Extend identifier	0x00	R/W
		0	The 11-bit standard identifier will be used for this message object.		
		1	The 29-bit extended identifier will be used for this message object.		

Table 926. CAN message interface command arbitration 2 registers (IF2_ARB2, address 0x400E 2094 (C_CAN0) and 0x400A 4094 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
15	MSGVAL		Message valid **Remark:** The MSGVAL bit of all unused Messages Objects is reset during the initialization before bit INIT is reset in the CAN Control Register. This bit must be set to zero before the identifier ID28:0, the control bits XTD, DIR, or the Data Length Code DLC3:0 are modified, or if the Messages Object is no longer required.	0	R/W
		0	The message object is ignored by the message handler.		
		1	The message object is configured and should be considered by the message handler.		
31:16	-	-	Reserved	0	-

41.6.2.4.5 CAN message interface message control registers

Table 927. CAN message interface message control registers (IF1_MCTRL, address 0x400E 2038 (C_CAN0) and 0x400A 4038 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
3:0	DLC3_0		Data length code **Remark:** The Data Length Code of a Message Object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message. 0000 to 1000 = Data frame has 0 - 8 data bytes. 1001 to 1111 = Data frame has 8 data bytes.	0000	R/W
6:4	-		reserved	-	-
7	EOB		End of buffer	0	R/W
		0	Message object belongs to a FIFO buffer and is not the last message object of that FIFO buffer.		
		1	Single message object or last message object of a FIFO buffer.		
8	TXRQST		Transmit request	0	R/W
		0	This message object is not waiting for transmission.		
		1	The transmission of this message object is requested and is not yet done		
9	RMTEN		Remote enable	0	R/W
		0	At the reception of a remote frame, TXRQST is left unchanged.		
		1	At the reception of a remote frame, TXRQST is set.		

Table 927. CAN message interface message control registers (IF1_MCTRL, address 0x400E 2038 (C_CAN0) and 0x400A 4038 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
10	RXIE		Receive interrupt enable	0	R/W
		0	INTPND will be left unchanged after successful reception of a frame.		
		1	INTPND will be set after successful reception of a frame.		
11	TXIE		Transmit interrupt enable	0	R/W
		0	The INTPND bit will be left unchanged after a successful reception of a frame.		
		1	INTPND will be set after a successful reception of a frame.		
12	UMASK		Use acceptance mask **Remark:** If UMASK is set to 1, the message object's mask bits have to be programmed during initialization of the message object before MAGVAL is set to 1.	0	R/W
		0	Mask ignored.		
		1	Use mask (MSK[28:0], MXTD, and MDIR) for acceptance filtering.		
13	INTPND		Interrupt pending	0	R/W
		0	This message object is not the source of an interrupt.		
		1	This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.		
14	MSGLST		Message lost (only valid for message objects in the direction receive).	0	R/W
		0	No message lost since this bit was reset last by the CPU.		
		1	The Message Handler stored a new message into this object when NEWDAT was still set, the CPU has lost a message.		
15	NEWDAT		New data	0	R/W
		0	No new data has been written into the data portion of this message object by the message handler since this flag was cleared last by the CPU.		
		1	The message handler or the CPU has written new data into the data portion of this message object.		
31:16	-	-	Reserved	0	-

Table 928. CAN message interface message control registers (IF2_MCTRL, address 0x400E 2098 (C_CAN0) and 0x400A 4098 (C_CAN1)) bit description

Bit	Symbol	Value	Description	Reset value	Access
3:0	DLC3_0		Data length code **Remark:** The Data Length Code of a Message Object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message. 0000 to 1000 = Data frame has 0 - 8 data bytes. 1001 to 1111 = Data frame has 8 data bytes.	0000	R/W
6:4	-		reserved	-	-
7	EOB		End of buffer	0	R/W
		0	Message object belongs to a FIFO buffer and is not the last message object of that FIFO buffer.		
		1	Single message object or last message object of a FIFO buffer.		
8	TXRQST		Transmit request	0	R/W
		0	This message object is not waiting for transmission.		
		1	The transmission of this message object is requested and is not yet done		
9	RMTEN		Remote enable	0	R/W
		0	At the reception of a remote frame, TXRQST is left unchanged.		
		1	At the reception of a remote frame, TXRQST is set.		
10	RXIE		Receive interrupt enable	0	R/W
		0	INTPND will be left unchanged after successful reception of a frame.		
		1	INTPND will be set after successful reception of a frame.		
11	TXIE		Transmit interrupt enable	0	R/W
		0	The INTPND bit will be left unchanged after a successful reception of a frame.		
		1	INTPND will be set after a successful reception of a frame.		
12	UMASK		Use acceptance mask **Remark:** If UMASK is set to 1, the message object's mask bits have to be programmed during initialization of the message object before MAGVAL is set to 1.	0	R/W
		0	Mask ignored.		
		1	Use mask (MSK[28:0], MXTD, and MDIR) for acceptance filtering.		

Table 928. CAN message interface message control registers (IF2_MCTRL, address 0x400E 2098 (C_CAN0) and 0x400A 4098 (C_CAN1)) bit description *…continued*

Bit	Symbol	Value	Description	Reset value	Access
13	INTPND		Interrupt pending	0	R/W
		0	This message object is not the source of an interrupt.		
		1	This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.		
14	MSGLST		Message lost (only valid for message objects in the direction receive).	0	R/W
		0	No message lost since this bit was reset last by the CPU.		
		1	The Message Handler stored a new message into this object when NEWDAT was still set, the CPU has lost a message.		
15	NEWDAT		New data	0	R/W
		0	No new data has been written into the data portion of this message object by the message handler since this flag was cleared last by the CPU.		
		1	The message handler or the CPU has written new data into the data portion of this message object.		
31:16	-	-	Reserved	0	-

41.6.2.4.6 CAN message interface data A1 registers

In a CAN Data Frame, DATA0 is the first, DATA7 (in CAN_IF1B2 AND CAN_IF2B2) is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

Remark: Byte DATA0 is the first data byte shifted into the shift register of the CAN Core during a reception, byte DATA7 is the last. When the Message Handler stores a Data Frame, it will write all the eight data bytes into a Message Object. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by non specified values.

Table 929. CAN message interface data A1 registers (IF1_DA1, address 0x400E 203C (C_CAN0) and 0x400A 403C (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA0	Data byte 0	0x00	R/W
15:8	DATA1	Data byte 1	0x00	R/W
31:16	-	Reserved	-	-

Table 930. CAN message interface data A1 registers (IF2_DA1, address 0x400E 209C (C_CAN0) and 0x400A 409C (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA0	Data byte 0	0x00	R/W
15:8	DATA1	Data byte 1	0x00	R/W
31:16	-	Reserved	-	-

41.6.2.4.7 CAN message interface data A2 registers

Table 931. CAN message interface data A2 registers (IF1_DA2, address 0x400E 2040 (C_CAN0) and 0x400A 4040 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA2	Data byte 2	0x00	R/W
15:8	DATA3	Data byte 3	0x00	R/W
31:16	-	Reserved	-	-

Table 932. CAN message interface data A2 registers (IF2_DA2, address 0x400E 20A0 (C_CAN0) and 0x400A 40A0 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA2	Data byte 2	0x00	R/W
15:8	DATA3	Data byte 3	0x00	R/W
31:16	-	Reserved	-	-

41.6.2.4.8 CAN message interface data B1 registers

Table 933. CAN message interface data B1 registers (IF1_DB1, address 0x400E 2044 (C_CAN0) and 0x400A 4044 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA4	Data byte 4	0x00	R/W
15:8	DATA5	Data byte 5	0x00	R/W
31:16	-	Reserved	-	-

Table 934. CAN message interface data B1 registers (IF2_DB1, address 0x400E 20A4 (C_CAN0) and 0x400A 40A4 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA4	Data byte 4	0x00	R/W
15:8	DATA5	Data byte 5	0x00	R/W
31:16	-	Reserved	-	-

41.6.2.4.9 CAN message interface data B2 registers

Table 935. CAN message interface data B2 registers (IF1_DB2, address 0x400E 2048 (C_CAN0) and 0x400A 4048 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA6	Data byte 6	0x00	R/W
15:8	DATA7	Data byte 7	0x00	R/W
31:16	-	Reserved	-	-

Table 936. CAN message interface data B2 registers (IF2_DB2, address 0x400E 20A8 (C_CAN0) and 0x400A 40A8 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
7:0	DATA6	Data byte 6	0x00	R/W
15:8	DATA7	Data byte 7	0x00	R/W
31:16	-	Reserved	-	-

41.6.3 Message handler registers

All Message Handler registers are read-only. Their contents (TXRQST, NEWDAT, INTPND, and MSGVAL bits of each Message Object and the Interrupt Identifier) is status information provided by the Message Handler FSM.

41.6.3.1 CAN transmission request 1 register

This register contains the TXRQST bits of message objects 1 to 16. By reading out the TXRQST bits, the CPU can check for which Message Object a Transmission Request is pending. The TXRQST bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception of a Remote Frame or after a successful transmission.

Table 937. CAN transmission request 1 register (TXREQ1, address 0x400E 2100 (C_CAN0) and 0x400A 4100 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	TXRQST16_1	Transmission request bit of message objects 16 to 1. 0 = This message object is not waiting for transmission. 1 = The transmission of this message object is requested and not yet done.	0x00	R
31:16	-	Reserved	-	-

41.6.3.2 CAN transmission request 2 register

This register contains the TXRQST bits of message objects 32 to 17. By reading out the TXRQST bits, the CPU can check for which Message Object a Transmission Request is pending. The TXRQST bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception of a Remote Frame or after a successful transmission.

Table 938. CAN transmission request 2 register (TXREQ2, address 0x400E 2104 (C_CAN0) and 0x400A 4104 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	TXRQST32_17	Transmission request bit of message objects 32 to 17. 0 = This message object is not waiting for transmission. 1 = The transmission of this message object is requested and not yet done.	0x00	R
31:16	-	Reserved	-	-

41.6.3.3 CAN new data 1 register

This register contains the NEWDAT bits of message objects 16 to 1. By reading out the NEWDAT bits, the CPU can check for which Message Object the data portion was updated. The NEWDAT bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

Table 939. CAN new data 1 register (ND1, address 0x400E 2120 (C_CAN0) and 0x400A 4120 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	NEWDAT16_1	New data bits of message objects 16 to 1. 0 = No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the CPU. 1 = The Message Handler or the CPU has written new data into the data portion of this Message Object.	0x00	R
31:16	-	Reserved	-	-

41.6.3.4 CAN new data 2 register

This register contains the NEWDAT bits of message objects 32 to 17. By reading out the NEWDAT bits, the CPU can check for which Message Object the data portion was updated. The NEWDAT bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

Table 940. CAN new data 2 register (ND2, address 0x400E 2124 (C_CAN0) and 0x400A 4124 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	NEWDAT32_17	New data bits of message objects 32 to 17. 0 = No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the CPU. 1 = The Message Handler or the CPU has written new data into the data portion of this Message Object.	0x00	R
31:16	-	Reserved	-	-

41.6.3.5 CAN interrupt pending 1 register

This register contains the INTPND bits of message objects 16 to 1. By reading out the INTPND bits, the CPU can check for which Message Object an interrupt is pending. The INTPND bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception or after a successful transmission of a frame. This will also affect the value of INTPND in the Interrupt Register.

Table 941. CAN interrupt pending 1 register (IR1, address 0x400E 2140 (C_CAN0) and 0x400A 4140 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	INTPND16_1	Interrupt pending bits of message objects 16 to 1. 0 = This message object is ignored by the message handler. 1 = This message object is the source of an interrupt.	0x00	R
31:16	-	Reserved	-	-

41.6.3.6 CAN interrupt pending 2 register

This register contains the INTPND bits of message objects 32 to 17. By reading out the INTPND bits, the CPU can check for which Message Object an interrupt is pending. The INTPND bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers or by the Message Handler after reception or after a successful transmission of a frame. This will also affect the value of INTPND in the Interrupt Register.

Table 942. CAN interrupt pending 2 register (IR2, addresses 0x400E 2144 (C_CAN0) and 0x400A 4144 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	INTPND32_17	Interrupt pending bits of message objects 32 to 17. 0 = This message object is ignored by the message handler. 1 = This message object is the source of an interrupt.	0x00	R
31:16	-	Reserved	-	-

41.6.3.7 CAN message valid 1 register

This register contains the MSGVAL bits of message objects 16 to 1. By reading out the MSGVAL bits, the CPU can check which Message Object is valid. The MSGVAL bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers.

Table 943. CAN message valid 1 register (MSGV1, addresses 0x400E 2160 (C_CAN0) and 0x400A 4160 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
15:0	MSGVAL16_1	Message valid bits of message objects 16 to 1. 0 = This message object is ignored by the message handler. 1 = This message object is configured and should be considered by the message handler.	0x00	R
31:16	-	Reserved	-	-

41.6.3.8 CAN message valid 2 register

This register contains the MSGVAL bits of message objects 32 to 17. By reading out the MSGVAL bits, the CPU can check which Message Object is valid. The MSGVAL bit of a specific Message Object can be set/reset by the CPU via the IFx Message Interface Registers.

Table 944. CAN message valid 2 register (MSGV2, address 0x400E 2164 (C_CAN0) and 0x400A 4164 (C_CAN1)) bit description

Bit	Symbol	Description	Access	Reset value
15:0	MSGVAL32_17	Message valid bits of message objects 32 to 17. 0 = This message object is ignored by the message handler. 1 = This message object is configured and should be considered by the message handler.	R	0x00
31:16	-	Reserved	-	-

41.6.4 CAN timing register

41.6.4.1 CAN clock divider register

This register determines the CAN clock signal. The CAN_CLK is derived from the peripheral clock PCLK divided by the values in this register.

Table 945. CAN clock divider register (CLKDIV, address 0x400E 2180 (C_CAN0) and 0x400A 4180 (C_CAN1)) bit description

Bit	Symbol	Description	Reset value	Access
3:0	CLKDIVVAL	Clock divider value CAN_CLK = PCLK/($2^{CLKDIVVAL\ -1}$+1) 0000: CAN_CLK = PCLK divided by 1. 0001: CAN_CLK = PCLK divided by 2. 0010: CAN_CLK = PCLK divided by 3. 0010: CAN_CLK = PCLK divided by 4. 0011: CAN_CLK = PCLK divided by 5. 0100: CAN_CLK = PCLK divided by 9. 0101: CAN_CLK = PCLK divided by 17. ... 1111: CAN_CLK = PCLK divided by 16385.	0001	R/W
31:4	-	reserved	-	-

41.7 Functional description

41.7.1 C_CAN controller state after reset

After a hardware reset, the registers hold the values described in Table 902. Additionally, the busoff state is reset and the output CAN_TXD is set to recessive (HIGH). The value 0x0001 (INIT = ‘1’) in the CAN Control Register enables the software initialization. The CAN controller does not communicate with the CAN bus until the CPU resets INIT to ‘0’.

The data stored in the message RAM is not affected by a hardware reset. After power-on, the contents of the message RAM is undefined.

41.7.2 C_CAN operating modes

41.7.2.1 Software initialization

The software initialization is started by setting the bit INIT in the CAN Control Register, either by software or by a hardware reset, or by entering the busoff state.

During software initialization (INIT bit is set), the following conditions are present:

- All message transfer from and to the CAN bus is stopped.
- The status of the CAN output CAN_TXD is recessive (HIGH).
- The EML counters are unchanged.
- The configuration registers are unchanged.
- Access to the bit timing register and the BRP extension register is enabled if the CCE bit in the CAN control register is also set.

To initialize the CAN controller, software has to set up the bit timing register and each message object. If a message object is not needed, it is sufficient to set its MSGVAL bit to not valid. Otherwise, the whole message object has to be initialized.

Resetting the INIT bit finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (Bus Idle) before it can take part in bus activities and starts the message transfer.

Remark: The initialization of the Message Objects is independent of INIT and also can be done on the fly, but the Message Objects should all be configured to particular identifiers or set to not valid during software initialization before the BSP starts the message transfer. To change the configuration of a Message Object during normal operation, the CPU has to start by setting the MSGVAL bit to not valid. When the configuration is completed, MSAGVALis set to valid again.

41.7.2.2 CAN message transfer

Once the CAN controller is initialized and INIT is reset to zero, the CAN core synchronizes itself to the CAN bus and starts the message transfer.

Received messages are stored into their appropriate Message Objects if they pass the Message Handler’s acceptance filtering. The whole message including all arbitration bits, DLC and eight data bytes is stored into the Message Object. If the Identifier Mask is used, the arbitration bits which are masked to “don’t care” may be overwritten in the Message Object.

The CPU may read or write each message any time via the Interface Registers. The Message Handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted are updated by the CPU. If a permanent Message Object (arbitration and control bits set up during configuration) exists for the message, only the data bytes are updated and then TXRQUT bit with NEWDAT bit are set to start the transmission. If several transmit messages are assigned to the same Message Object (when the number of Message Objects is not sufficient), the whole Message Object has to be configured before the transmission of this message is requested.

The transmission of any number of Message Objects may be requested at the same time, and they are transmitted subsequently according to their internal priority. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data will be discarded when a message is updated before its pending transmission has started.

Depending on the configuration of the Message Object, the transmission of a message may be requested autonomously by the reception of a remote frame with a matching identifier.

41.7.2.3 Disabled Automatic Retransmission (DAR)

According to the *CAN Specification (ISO11898, 6.3.3 Recovery Management)*, the CAN controller provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed. By default, the automatic retransmission on lost arbitration or error is enabled. It can be disabled to enable the CAN controller to work within a Time Triggered CAN (TTCAN, see ISO11898-1) environment.

The Disable Automatic Retransmission mode is enabled by programming bit DAR in the CAN Control Register to one. In this operation mode the programmer has to consider the different behavior of bits TXRQST and NEWDAT in the Control Registers of the Message Buffers:

- When a transmission starts, bit TXRQST of the respective Message Buffer is reset while bit NEWDAT remains set.
- When the transmission completed successfully, bit NEWDAT is reset.
- When a transmission failed (lost arbitration or error), bit NEWDAT remains set. To restart the transmission, the CPU has to set TXRQST back to one.

41.7.2.4 Test modes

The Test mode is entered by setting bit TEST in the CAN Control Register to one. In Test mode the bits TX1, TX0, LBACK, SILENT, and BASIC in the Test Register are writable. Bit RX monitors the state of pins RD0,1 and therefore is only readable. All Test register functions are disabled when bit TEST is reset to zero.

41.7.2.4.1 Silent mode

The CAN core can be set in Silent mode by programming the Test register bit SILENT to one.

In Silent Mode, the CAN controller is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus, and it cannot start a transmission. If the CAN Core is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. The Silent mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits (Acknowledge Bits, Error Frames).

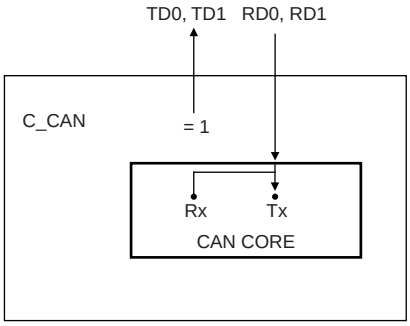


Fig 149. CAN core in Silent mode

41.7.2.4.2 Loop-back mode

The CAN Core can be set in Loop-back mode by programming the Test Register bit LBACK to one. In Loop-back Mode, the CAN Core treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into a Receive Buffer.

This mode is provided for self-test functions. To be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop-back mode. In this mode the CAN core performs an internal feedback from its CAN_TXD output to its CAN_RXD input. The actual value of the CAN_RXD input pin is disregarded by the CAN Core. The transmitted messages can be monitored at the CAN_TXD pin.

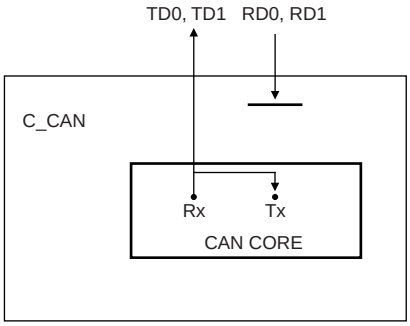


Fig 150. CAN core in Loop-back mode

41.7.2.4.3 Loop-back mode combined with Silent mode

It is also possible to combine Loop-back mode and Silent mode by programming bits LBACK and SILENT to one at the same time. This mode can be used for a “Hot Self test”, meaning the C_CAN can be tested without affecting a running CAN system connected to the pins CAN_TXD and CAN_RXD. In this mode the CAN_RXD pin is disconnected from the CAN Core and the CAN_TXD pin is held recessive.

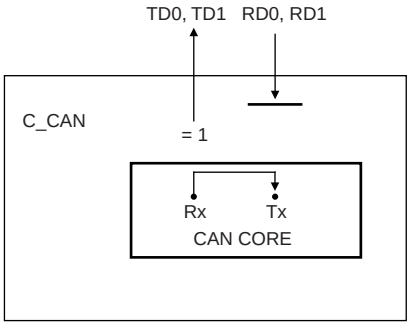


Fig 151. CAN core in Loop-back mode combined with Silent mode

41.7.2.4.4 Basic mode

The CAN Core can be set in Basic mode by programming the Test Register bit BASIC to one. In this mode the CAN controller runs without the Message RAM.

The IF1 Registers are used as Transmit Buffer. The transmission of the contents of the IF1 Registers is requested by writing the BUSY bit of the IF1 Command Request Register to ‘1’. The IF1 Registers are locked while the BUSY bit is set. The BUSY bit indicates that the transmission is pending.

As soon the CAN bus is idle, the IF1 Registers are loaded into the shift register of the CAN Core and the transmission is started. When the transmission has completed, the BUSY bit is reset and the locked IF1 Registers are released.

A pending transmission can be aborted at any time by resetting the BUSY bit in the IF1 Command Request Register while the IF1 Registers are locked. If the CPU has reset the BUSY bit, a possible retransmission in case of lost arbitration or in case of an error is disabled.

The IF2 Registers are used as Receive Buffer. After the reception of a message the contents of the shift register is stored into the IF2 Registers, without any acceptance filtering.

Additionally, the actual contents of the shift register can be monitored during the message transfer. Each time a read Message Object is initiated by writing the BUSY bit of the IF2 Command Request Register to ‘1’, the contents of the shift register is stored into the IF2 Registers.

In Basic mode the evaluation of all Message Object related control and status bits and of the control bits of the IFx Command Mask Registers is turned off. The message number of the Command request registers is not evaluated. The NEWDAT and MSGLST bits of the IF2 Message Control Register retain their function, DLC3-0 will show the received DLC, the other control bits will be read as ‘0’.

In Basic mode the ready output CAN_WAIT_B is disabled (always ‘1’)

41.7.2.4.5 Software control of pin CAN_TXD

Four output functions are available for the CAN transmit pin CAN_TXD:

1. serial data output (default).

2. drives CAN sample point signal to monitor the CAN controller's timing.
3. drives recessive constant value.
4. drives dominant constant value.

The last two functions, combined with the readable CAN receive pin CAN_RXD, can be used to check the CAN bus' physical layer.

The output mode of pin CAN_TXD is selected by programming the Test Register bits TX1 and TX0 as described Section 41.6.1.6.

Remark: The three test functions for pin CAN_TXD interfere with all CAN protocol functions. The CAN_TXD pin must be left in its default function when CAN message transfer or any of the test modes Loo-back mode, Silent mode, or Basic mode are selected.

41.7.3 CAN message handler

The Message handler controls the data transfer between the Rx/Tx Shift Register of the CAN Core, the Message RAM and the IFx Registers, see Figure 148.

The message handler controls the following functions:

- Data Transfer between IFx Registers and the Message RAM
- Data Transfer from Shift Register to the Message RAM
- Data Transfer from Message RAM to Shift Register
- Data Transfer from Shift Register to the Acceptance Filtering unit
- Scanning of Message RAM for a matching Message Object
- Handling of TXRQST flags
- Handling of interrupts

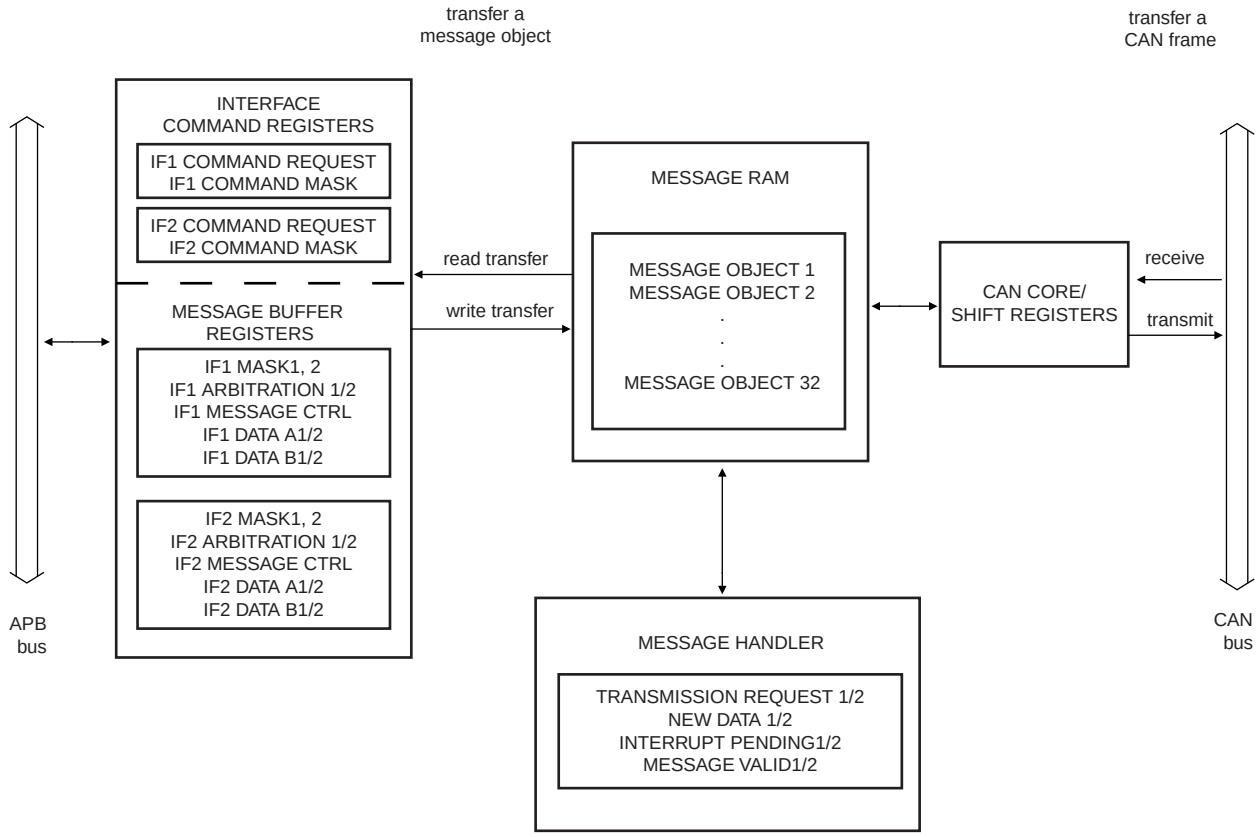


Fig 152. Block diagram of a message object transfer

41.7.3.1 Management of message objects

The configuration of the Message Objects in the Message RAM will (with the exception of the bits MSGVAL, NEWDAT, INTPND, and TXRQST) is not be affected by resetting the chip. All the Message Objects must be initialized by the CPU or they must be set to not valid (MSGVAL = '0').The bit timing must be configured before the CPU clears the INIT bit in the CAN Control Register.

The configuration of a Message Object is done by programming Mask, Arbitration, Control and Data field of one of the two interface register sets to the desired values. By writing to the corresponding IFx Command Request Register, the IFx Message Buffer Registers are loaded into the addressed Message Object in the Message RAM.

When the INIT bit in the CAN Control Register is cleared, the CAN Protocol Controller state machine of the CAN core and the Message Handler State Machine control the CAN controller's internal data flow. Received messages that pass the acceptance filtering are stored into the Message RAM, and messages with pending transmission request are loaded into the CAN core's shift register and are transmitted via the CAN bus.

The CPU reads received messages and updates messages to be transmitted via the IFx Interface Registers. Depending on the configuration, the CPU is interrupted on certain CAN message and CAN error events.

41.7.3.2 Data Transfer between IFx Registers and the Message RAM

When the CPU initiates a data transfer between the IFx Registers and Message RAM, the Message Handler sets the BUSY bit in the respective Command Register to ‘1’. After the transfer has completed, the BUSY bit is set back to ‘0’.

The Command Mask Register specifies whether a complete Message Object or only parts of it will be transferred. Due to the structure of the Message RAM it is not possible to write single bits/bytes of one Message Object. Software must always write a complete Message Object into the Message RAM. Therefore the data transfer from the IFx Registers to the Message RAM requires a read-modify-write cycle:

1. Read the parts of the message object that are not to be changed from the message RAM using the command mask register.
   - After the partial read of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will be left unchanged.
2. Write the complete contents of the message buffer registers into the message object.
   - After the partial write of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will set to the actual contents of the selected Message Object.

41.7.3.3 Transmission of messages between the shift registers in the CAN core and the Message buffer

If the shift register of the CAN Core cell is ready for loading and if there is no data transfer between the IFx Registers and Message RAM, the MSGVAL bits in the Message Valid Register TXRQST bits in the Transmission Request Register are evaluated. The valid Message Object with the highest priority pending transmission request is loaded into the shift register by the Message Handler and the transmission is started. The Message Object’s NEWDAT bit is reset.

After a successful transmission and if no new data was written to the Message Object (NEWDAT = ‘0’) since the start of the transmission, the TXRQST bit will be reset. If TXIE is set, INTPND will be set after a successful transmission. If the CAN controller has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

41.7.3.4 Acceptance filtering of received messages

When the arbitration and control field (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the Rx/Tx Shift Register of the CAN Core, the Message Handler state machine starts the scanning of the Message RAM for a matching valid Message Object.

To scan the Message RAM for a matching Message Object, the Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register. Then the arbitration and mask fields (including MSGVAL, UMASK, NEWDAT, and EOB) of Message Object 1 are loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following Message Object until a matching Message Object is found or until the end of the Message RAM is reached.

If a match occurs, the scanning is stopped and the Message Handler state machine proceeds depending on the type of frame (Data Frame or Remote Frame) received.

41.7.3.4.1 Reception of a data frame

The Message Handler state machine stores the message from the CAN Core shift register into the respective Message Object in the Message RAM. The data bytes, all arbitration bits, and the Data Length Code are stored into the corresponding Message Object. This is implemented to keep the data bytes connected with the identifier even if arbitration mask registers are used.

The NEWDAT bit is set to indicate that new data (not yet seen by the CPU) has been received. The CPU/software should reset NEWDAT when it reads the Message Object. If at the time of the reception the NEWDAT bit was already set, MSGLST is set to indicate that the previous data (supposedly not seen by the CPU) is lost. If the RxIE bit is set, the INTPND bit is also set, causing the Interrupt Register to point to this Message Object.

The TXRQST bit of this Message Object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

41.7.3.4.2 Reception of a remote frame

When a Remote Frame is received, three different configurations of the matching Message Object have to be considered:

1. DIR = ‘1’ (direction = transmit), RMTEN = ‘1’, UMASK = ‘1’ or ’0’

   On the reception of a matching Remote Frame, the TXRQST bit of this Message Object is set. The rest of the Message Object remains unchanged.

2. DIR = ‘1’ (direction = transmit), RMTEN = ‘0’, UMASK = ’0’

   On the reception of a matching Remote Frame, the TXRQST bit of this Message Object remains unchanged; the Remote Frame is ignored.

3. DIR = ‘1’ (direction = transmit), RMTEN = ‘0’, UMASK = ’1’

   On the reception of a matching Remote Frame, the TXRQST bit of this Message Object is reset. The arbitration and control field (Identifier + IDE + RTR + DLC) from the shift register is stored into the Message Object in the Message RAM, and the NEWDAT bit of this Message Object is set. The data field of the Message Object remains unchanged; the Remote Frame is treated similar to a received Data Frame.

41.7.3.5 Receive/transmit priority

The receive/transmit priority for the Message Objects is attached to the message number. Message Object 1 has the highest priority, while Message Object 32 has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding Message Object.

41.7.3.6 Configuration of a transmit object

Table 946 shows how a transmit object should be initialized by software (see also Table 912):

Table 946. Initialization of a transmit object

MSGVAL	Arbitration bits	Data bits	Mask bits	EOB	DIR	NEWDAT
1	application dependent	application dependent	application dependent	1	1	0
MSGLST	**RXIE**	**TXIE**	**INTPND**	**RMTEN**	**TXRQST**	
0	0	application dependent	0	application dependent	0	

The Arbitration Registers (ID28:0 and XTD bit) are given by the application. They define the identifier and the type of the outgoing message. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to ID28. In this case ID18, ID17 to ID0 can be disregarded.

If the TXIE bit is set, the INTPND bit will be set after a successful transmission of the Message Object.

If the RMTEN bit is set, a matching received Remote Frame will cause the TXRQST bit to be set, and the Remote Frame will autonomously be answered by a Data Frame.

The Data Registers (DLC3:0, Data0:7) are given by the application. TXRQST and RMTEN may not be set before the data is valid.

The Mask Registers (Msk28-0, UMASK, MXTD, and MDIR bits) may be used (UMASK=’1’) to allow groups of Remote Frames with similar identifiers to set the TXRQST bit. For details see Section 41.7.3.4.2. The DIR bit should not be masked.

41.7.3.7 Updating a transmit object

The CPU may update the data bytes of a Transmit Object any time via the IFx Interface registers. Neither MSGVAL nor TXRQST have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding IFx Data A Register or IFx Data B Register have to be valid before the content of that register is transferred to the Message Object. Either the CPU has to write all four bytes into the IFx Data Register or the Message Object is transferred to the IFx Data Register before the CPU writes the new data bytes.

When only the (eight) data bytes are updated, first 0x0087 is written to the Command Mask Register. Then the number of the Message Object is written to the Command Request Register, concurrently updating the data bytes and setting TXRQST.

To prevent the reset of TXRQST at the end of a transmission that may already be in progress while the data is updated, NEWDAT has to be set together with TXRQST. For details see Section 41.7.3.3.

When NEWDAT is set together with TXRQST, NEWDAT will be reset as soon as the new transmission has started.

41.7.3.8 Configuration of a receive object

Table 947 shows how a receive object should be initialized by software (see also Table 912)

Table 947. Initialization of a receive object

MSGVAL	Arbitration bits	Data bits	Mask bits	EOB	DIR	NEWDAT
1	application dependent	application dependent	application dependent	1	0	0

MSGLST	RXIE	TXIE	INTPND	RMTEN	TXRQST
0	application dependent	0	0	0	0

The Arbitration Registers (ID28-0 and XTD bit) are given by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (“Standard Frame”) is used, it is programmed to ID28 to ID18. ID17 to ID0 can then be disregarded. When a Data Frame with an 11-bit Identifier is received, ID17 to ID0 will be set to ‘0’.

If the RxIE bit is set, the INTPND bit will be set when a received Data Frame is accepted and stored in the Message Object.

The Data Length Code (DLC[3:0] is given by the application. When the Message Handler stores a Data Frame in the Message Object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by non specified values.

The Mask Registers (Msk[28:0], UMASK, MXTD, and MDIR bits) may be used (UMASK=’1’) to allow groups of Data Frames with similar identifiers to be accepted. For details see section Section 41.7.3.4.1. The DIR bit should not be masked in typical applications.

41.7.3.9 Handling of received messages

The CPU may read a received message any time via the IFx Interface registers. The data consistency is guaranteed by the Message Handler state machine.

To transfer the entire received message from message RAM into the message buffer, software must write first 0x007F to the Command Mask Register and then the number of the Message Object to the Command Request Register. Additionally, the bits NEWDAT and INTPND are cleared in the Message RAM (not in the Message Buffer).

If the Message Object uses masks for acceptance filtering, the arbitration bits show which of the matching messages has been received.

The actual value of NEWDAT shows whether a new message has been received since last time this Message Object was read. The actual value of MSGLST shows whether more than one message has been received since last time this Message Object was read. MSGLST will not be automatically reset.

Using a Remote Frame, the CPU may request another CAN node to provide new data for a receive object. Setting the TXRQST bit of a receive object will cause the transmission of a Remote Frame with the receive object’s identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the TXRQST bit is automatically reset.

41.7.3.10 Configuration of a FIFO buffer

With the exception of the EOB bit, the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a (single) Receive Object, see section Section 41.7.3.8.

To concatenate two or more Message Objects into a FIFO Buffer, the identifiers and masks (if used) of these Message Objects have to be programmed to matching values. Due to the implicit priority of the Message Objects, the Message Object with the lowest number will be the first Message Object of the FIFO Buffer. The EOB bit of all Message Objects of a FIFO Buffer except the last have to be programmed to zero. The EOB bits of the last Message Object of a FIFO Buffer is set to one, configuring it as the End of the Block.

41.7.3.10.1 Reception of messages with FIFO buffers

Received messages with identifiers matching to a FIFO Buffer are stored into a Message Object of this FIFO Buffer starting with the Message Object with the lowest message number.

When a message is stored into a Message Object of a FIFO Buffer the NEWDAT bit of this Message Object is set. By setting NEWDAT while EOB is zero the Message Object is locked for further write accesses by the Message Handler until the CPU has written the NEWDAT bit back to zero.

Messages are stored into a FIFO Buffer until the last Message Object of this FIFO Buffer is reached. If none of the preceding Message Objects is released by writing NEWDAT to zero, all further messages for this FIFO Buffer will be written into the last Message Object of the FIFO Buffer and therefore overwrite previous messages.

41.7.3.10.2 Reading from a FIFO buffer

When the CPU transfers the contents of Message Object to the IFx Message Buffer registers by writing its number to the IFx Command Request Register, bits NEWDAT and INTPND in the corresponding Command Mask Register should be reset to zero (TXRQST/NEWDAT = ‘1’ and ClrINTPND = ‘1’). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

To assure the correct function of a FIFO Buffer, the CPU should read out the Message Objects starting at the FIFO Object with the lowest message number.

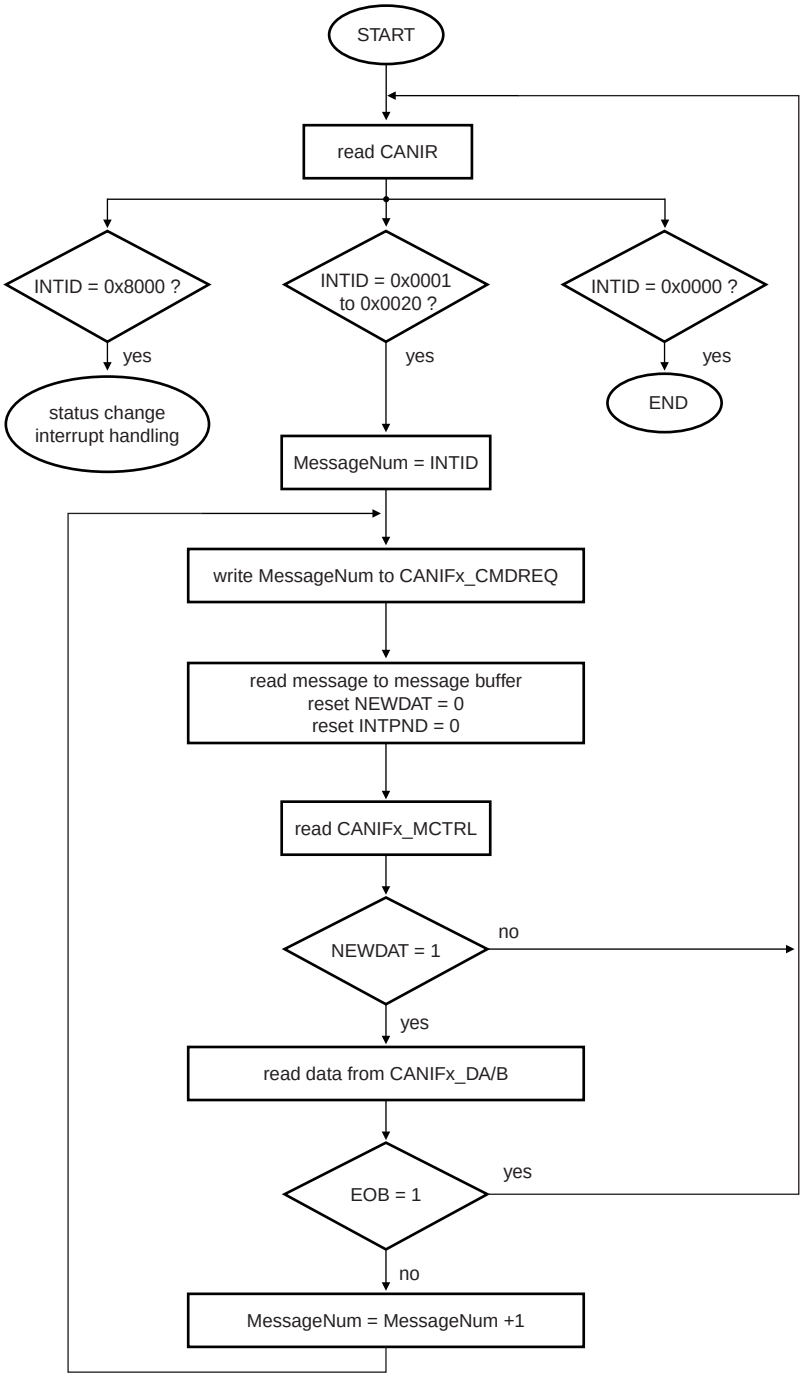


Fig 153. Reading a message from the FIFO buffer to the message buffer

41.7.4 Interrupt handling

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the CPU has cleared it.

The Status Interrupt has the highest priority. Among the message interrupts, the Message Object's interrupt priority decreases with increasing message number.

A message interrupt is cleared by clearing the Message Object's INTPND bit. The Status Interrupt is cleared by reading the Status Register.

The interrupt identifier INTID in the Interrupt Register indicates the cause of the interrupt. When no interrupt is pending, the register will hold the value zero. If the value of the Interrupt Register is different from zero, then there is an interrupt pending and, if IE is set, the interrupt line to the CPU, IRQ_B, is active. The interrupt line remains active until the Interrupt Register is back to value zero (the cause of the interrupt is reset) or until IE is reset.

The value 0x8000 indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The CPU can update (reset) the status bits RXOK, TXOK and LEC, but a write access of the CPU to the Status Register can never generate or reset an interrupt.

All other values indicate that the source of the interrupt is one of the Message Objects where INTID points to the pending message interrupt with the highest interrupt priority.

The CPU controls whether a change of the Status Register may cause an interrupt (bits EIE and SIE in the CAN Control Register) and whether the interrupt line becomes active when the Interrupt Register is different from zero (bit IE in the CAN Control Register). The Interrupt Register will be updated even when IE is reset.

The CPU has two possibilities to follow the source of a message interrupt:

- Software can follow the INTID in the Interrupt Register.
- Software can poll the interrupt pending register.

An interrupt service routine reading the message that is the source of the interrupt may read the message and reset the Message Object's INTPND at the same time (bit ClrINTPND in the Command Mask Register). When INTPND is cleared, the Interrupt Register will point to the next Message Object with a pending interrupt.

41.7.5 Bit timing

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly. In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. In the case of arbitration however, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and of the CAN nodes' interaction on the CAN bus.

41.7.5.1 Bit time and bit rate

CAN supports bit rates in the range of lower than 1 kBit/s up to 1000 kBit/s. Each member of the CAN network has its own clock generator, usually a quartz oscillator. The timing parameter of the bit time (i.e. the reciprocal of the bit rate) can be configured individually for each CAN node, creating a common bit rate even though the CAN nodes' oscillator periods (f_{osc}) may be different.

The frequencies of these oscillators are not absolutely stable, as small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by re-synchronizing to the bit stream.

According to the CAN specification, the bit time is divided into four segments (Figure 154). The Synchronization Segment, the Propagation Time Segment, the Phase Buffer Segment 1, and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta (see Table 948). The length of the time quantum (t_q), which is the basic time unit of the bit time, is defined by the CAN controller's system clock f and the Baud Rate Prescaler (BRP): t_q = BRP / f_{sys}. The C_CAN's system clock f_{sys} is the frequency C_CAN peripheral clock.

The Synchronization Segment Sync_Seg is the part of the bit time where edges of the CAN bus level are expected to occur; the distance between an edge that occurs outside of Sync_Seg and the Sync_Seg is called the phase error of that edge. The Propagation Time Segment Prop_Seg is intended to compensate for the physical delay times within the CAN network. The Phase Buffer Segments Phase_Seg1 and Phase_Seg2 surround the Sample Point. The (Re-)Synchronization Jump Width (SJW) defines how far a re-synchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

Table 948 describes the minimum programmable ranges required by the CAN protocol. Bit time parameters are programmed through the BT register, Table 907. For details on bit timing and examples, see the *C_CAN user's manual, revision 1.2*.

Table 948. Parameters of the C_CAN bit time

Parameter	Range	Function
BRP	(1...32)	Defines the length of the time quantum t_q.
SYNC_SEG	$1t_q$	Synchronization segment. Fixed length. Synchronization of bus input to system clock.
PROP_SEG	(1...8) × t_q	Propagation time segment. Compensates for physical delay times. This parameter is determined by the system delay times in the C_CAN network.
TSEG1	(1...8) × t_q	Phase buffer segment 1. May be lengthened temporarily by synchronization.
TSEG2	(1...8) × t_q	Phase buffer segment 2. May be shortened temporarily by synchronization.
SJW	(1...4) × t_q	(Re-) synchronization jump width. May not be longer than either phase buffer segment.

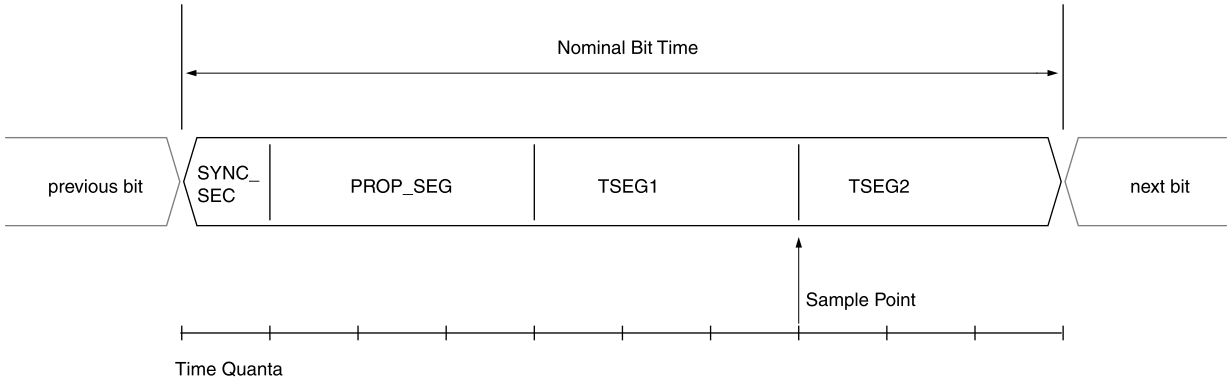


Fig 154. Bit timing

42.1 How to read this chapter

The I2C-bus interfaces I2C0 and I2C1 are available on all LPC43xx parts.

42.2 Basic configuration

The I2C0/1 are configured as follows:

- See Table 949 for clocking and power control.
- The I2C0/1 are reset by the I2C0/1_RST (reset # 48/49).
- The I2C0/1 interrupts are connected to slots # 18/19 in the NVIC.
- Configure the I2C0 pins for Fast-mode Plus, Fast mode, or Standard mode through the SFSI2C0 register in the SYSCON block (see Table 127).

Table 949. I2C0/1 clocking and power control

	Base clock	Branch clock	Operating frequency
Clock to the I2C0 register interface and I2C0 peripheral clock.	BASE_APB1_CLK	CLK_APB1_I2C0	up to 204 MHz
Clock to the I2C1 register interface and I2C1 peripheral clock.	BASE_APB3_CLK	CLK_APB3_I2C1	up to 204 MHz

42.3 Features

- The I2C0-bus interface uses true open-drain pins and supports Fast mode plus with bit rates of up to 1Mbit/s, Fast mode, or Standard mode.
- The I2C1-bus interface uses standard port pins supporting bit rates of up to 400 kbit/s.
- Standard I^2C-compliant bus interfaces may be configured as Master, Slave, or Master/Slave.
- Arbitration is handled between simultaneously transmitting masters without corruption of serial data on the bus.
- Programmable clock allows adjustment of I^2C transfer rates.
- Data transfer is bidirectional between masters and slaves.
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus.
- Serial clock synchronization is used as a handshake mechanism to suspend and resume serial transfer.
- Optional recognition of up to four distinct slave addresses.
- Monitor mode allows observing all I^2C-bus traffic, regardless of slave address.
- I^2C-bus can be used for test and diagnostic purposes.
- The I^2C-bus contains a standard I^2C-compliant bus interface with two pins.

42.4 Applications

Interfaces to external I^{2}C standard parts, such as serial RAMs, LCDs, tone generators, other microcontrollers, etc.

42.5 General description

A typical I^{2}C-bus configuration is shown in Figure 155. Depending on the state of the direction bit (R/W), two types of data transfers are possible on the I^{2}C-bus:

- Data transfer from a master transmitter to a slave receiver. The first byte transmitted by the master is the slave address. Next follows a number of data bytes. The slave returns an acknowledge bit after each received byte.
- Data transfer from a slave transmitter to a master receiver. The first byte (the slave address) is transmitted by the master. The slave then returns an acknowledge bit. Next follows the data bytes transmitted by the slave to the master. The master returns an acknowledge bit after all received bytes other than the last byte. At the end of the last received byte, a “not acknowledge” is returned. The master device generates all of the serial clock pulses and the START and STOP conditions. A transfer is ended with a STOP condition or with a Repeated START condition. Since a Repeated START condition is also the beginning of the next serial transfer, the I^{2}C bus will not be released.

The I^{2}C interface is byte oriented and has four operating modes: master transmitter mode, master receiver mode, slave transmitter mode and slave receiver mode.

The I^{2}C interface complies with the entire I^{2}C specification, supporting the ability to turn power off to the processor without interfering with other devices on the same I^{2}C-bus.

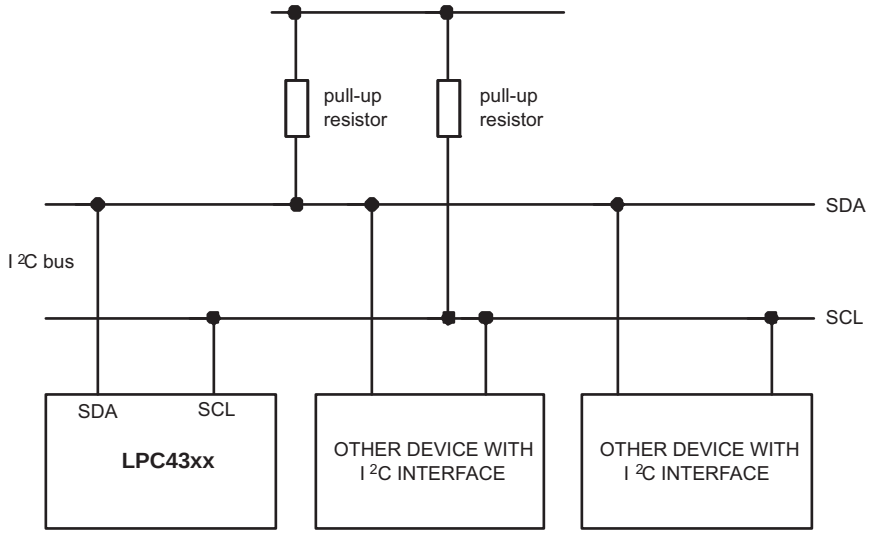


Fig 155. I^{2}C-bus configuration

42.5.1 I²C Fast-mode Plus

Fast-Mode Plus supports a 1 Mbit/sec transfer rate to communicate with the I²C-bus products which NXP Semiconductors is now providing.

In order to use Fast-Mode Plus, the I²C pins must be properly configured in the SFSI2C0 register in the SYSCON block (see Table 127).

42.6 Pin description

Table 950. I²C-bus pin description

Function name	Type	Description
SDA0	Input/Output	I²C data input/output. Open-drain output (for I²C-bus compliance).
SCL0	Input/Output	I²C clock input/output. Open-drain output (for I²C-bus compliance).
SDA1	Input/Output	I²C Serial Data. Uses standard I/O pins (Fast-mode only).
SCL1	Input/Output	I²C Serial Clock. Uses standard I/O pins (Fast-mode only).

The I²C-bus pins must be configured through SYSCON registers for Standard/ Fast-mode or Fast-mode Plus.

42.7 Register description

Table 951. Register overview: I²C0 (base address 0x400A 1000)

Name	Access	Address offset	Description	Reset value[1]	Reference
CONSET	R/W	0x000	**I2C Control Set Register.** When a one is written to a bit of this register, the corresponding bit in the I²C control register is set. Writing a zero has no effect on the corresponding bit in the I²C control register.	0x00	Table 953
STAT	RO	0x004	**I2C Status Register.** During I²C operation, this register provides detailed status codes that allow software to determine the next action needed.	0xF8	Table 954
DAT	R/W	0x008	**I2C Data Register.** During master or slave transmit mode, data to be transmitted is written to this register. During master or slave receive mode, data that has been received may be read from this register.	0x00	Table 955
ADR0	R/W	0x00C	**I2C Slave Address Register 0.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 956
SCLH	R/W	0x010	**SCH Duty Cycle Register High Half Word.** Determines the high time of the I²C clock.	0x04	Table 957
SCLL	R/W	0x014	**SCL Duty Cycle Register Low Half Word.** Determines the low time of the I²C clock. SCLL and SCLH together determine the clock frequency generated by an I²C master and certain times used in slave mode.	0x04	Table 958

Table 951. Register overview: I²C0 (base address 0x400A 1000) *…continued*

Name	Access	Address offset	Description	Reset value[1]	Reference
CONCLR	WO	0x018	**I2C Control Clear Register.** When a one is written to a bit of this register, the corresponding bit in the I²C control register is cleared. Writing a zero has no effect on the corresponding bit in the I²C control register.	-	Table 959
MMCTRL	R/W	0x01C	**Monitor mode control register.**	0x00	Table 960
ADR1	R/W	0x020	**I2C Slave Address Register 1.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 961
ADR2	R/W	0x024	**I2C Slave Address Register 2.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 961
ADR3	R/W	0x028	**I2C Slave Address Register 3.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 961
DATA_BUFFER	RO	0x02C	**Data buffer register.** The contents of the 8 MSBs of the DAT shift register will be transferred to the DATA_BUFFER automatically after every nine bits (8 bits of data plus ACK or NACK) has been received on the bus.	0x00	Table 963
MASK0	R/W	0x030	**I2C Slave address mask register 0**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964
MASK1	R/W	0x034	**I2C Slave address mask register 1**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964
MASK2	R/W	0x038	**I2C Slave address mask register 2**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964
MASK3	R/W	0x03C	**I2C Slave address mask register 3**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

Table 952. Register overview: I²C1 (base address 0x400E 0000)

Name	Access	Address offset	Description	Reset value[1]	Reference
CONSET	R/W	0x000	**I2C Control Set Register.** When a one is written to a bit of this register, the corresponding bit in the I²C control register is set. Writing a zero has no effect on the corresponding bit in the I²C control register.	0x00	Table 953
STAT	RO	0x004	**I2C Status Register.** During I²C operation, this register provides detailed status codes that allow software to determine the next action needed.	0xF8	Table 954
IDAT	R/W	0x008	**I2C Data Register.** During master or slave transmit mode, data to be transmitted is written to this register. During master or slave receive mode, data that has been received may be read from this register.	0x00	Table 955
ADR0	R/W	0x00C	**I2C Slave Address Register 0.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 956
SCLH	R/W	0x010	**SCH Duty Cycle Register High Half Word.** Determines the high time of the I²C clock.	0x04	Table 957
SCLL	R/W	0x014	**SCL Duty Cycle Register Low Half Word.** Determines the low time of the I²C clock. SCLL and SCLH together determine the clock frequency generated by an I²C master and certain times used in slave mode.	0x04	Table 958
CONCLR	WO	0x018	**I2C Control Clear Register.** When a one is written to a bit of this register, the corresponding bit in the I²C control register is cleared. Writing a zero has no effect on the corresponding bit in the I²C control register.	-	Table 959
MMCTRL	R/W	0x01C	**Monitor mode control register.**	0x00	Table 960
ADR1	R/W	0x020	**I2C Slave Address Register 1.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 961
ADR2	R/W	0x024	**I2C Slave Address Register 2.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 961
ADR3	R/W	0x028	**I2C Slave Address Register 3.** Contains the 7-bit slave address for operation of the I²C interface in slave mode, and is not used in master mode. The least significant bit determines whether a slave responds to the General Call address.	0x00	Table 961
DATA_ BUFFER	RO	0x02C	**Data buffer register.** The contents of the 8 MSBs of the DAT shift register will be transferred to the DATA_BUFFER automatically after every nine bits (8 bits of data plus ACK or NACK) has been received on the bus.	0x00	Table 963
MASK0	R/W	0x030	**I2C Slave address mask register 0**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964

Table 952. Register overview: I²C1 (base address 0x400E 0000) *…continued*

Name	Access	Address offset	Description	Reset value[1]	Reference
MASK1	R/W	0x034	**I2C Slave address mask register 1**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964
MASK2	R/W	0x038	**I2C Slave address mask register 2**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964
MASK3	R/W	0x03C	**I2C Slave address mask register 3**. This mask register is associated with ADR0 to determine an address match. The mask register has no effect when comparing to the General Call address (‘0000000’).	0x00	Table 964

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

42.7.1 I²C Control Set register

The CONSET registers control setting of bits in the CON register that controls operation of the I²C interface. Writing a one to a bit of this register causes the corresponding bit in the I²C control register to be set. Writing a zero has no effect.

Table 953. I²C Control Set register (CONSET - address 0x400A 1000 (I2C0) and 0x400E 0000 (I2C1)) bit description

Bit	Symbol	Description	Reset value
1:0	-	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
2	AA	Assert acknowledge flag.	
3	SI	I²C interrupt flag.	0
4	STO	STOP flag.	0
5	STA	START flag.	0
6	I2EN	I²C interface enable.	0
31:7	-	Reserved. The value read from a reserved bit is not defined.	-

I2EN I²C Interface Enable. When I2EN is 1, the I²C interface is enabled. I2EN can be cleared by writing 1 to the I2ENC bit in the CONCLR register. When I2EN is 0, the I²C interface is disabled.

When I2EN is “0”, the SDA and SCL input signals are ignored, the I²C block is in the “not addressed” slave state, and the STO bit is forced to “0”.

I2EN should not be used to temporarily release the I²C-bus since, when I2EN is reset, the I²C-bus status is lost. The AA flag should be used instead.

STA is the START flag. Setting this bit causes the I²C interface to enter master mode and transmit a START condition or transmit a Repeated START condition if it is already in master mode.

When STA is 1 and the I^2C interface is not already in master mode, it enters master mode, checks the bus and generates a START condition if the bus is free. If the bus is not free, it waits for a STOP condition (which will free the bus) and generates a START condition after a delay of a half clock period of the internal clock generator. If the I^2C interface is already in master mode and data has been transmitted or received, it transmits a Repeated START condition. STA may be set at any time, including when the I^2C interface is in an addressed slave mode.

STA can be cleared by writing 1 to the STAC bit in the CONCLR register. When STA is 0, no START condition or Repeated START condition will be generated.

If STA and STO are both set, then a STOP condition is transmitted on the I^2C-bus if it the interface is in master mode, and transmits a START condition thereafter. If the I^2C interface is in slave mode, an internal STOP condition is generated, but is not transmitted on the bus.

STO is the STOP flag. Setting this bit causes the I^2C interface to transmit a STOP condition in master mode, or recover from an error condition in slave mode. When STO is 1 in master mode, a STOP condition is transmitted on the I^2C-bus. When the bus detects the STOP condition, STO is cleared automatically.

In slave mode, setting this bit can recover from an error condition. In this case, no STOP condition is transmitted to the bus. The hardware behaves as if a STOP condition has been received and it switches to “not addressed” slave receiver mode. The STO flag is cleared by hardware automatically.

SI is the I^2C Interrupt Flag. This bit is set when the I^2C state changes. However, entering state F8 does not set SI since there is nothing for an interrupt service routine to do in that case.

While SI is set, the low period of the serial clock on the SCL line is stretched, and the serial transfer is suspended. When SCL is HIGH, it is unaffected by the state of the SI flag. SI must be reset by software, by writing a 1 to the SIC bit in CONCLR register.

AA is the Assert Acknowledge Flag. When set to 1, an acknowledge (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line on the following situations:

1. The address in the Slave Address Register has been received.
2. The General Call address has been received while the General Call bit (GC) in ADR is set.
3. A data byte has been received while the I^2C is in the master receiver mode.
4. A data byte has been received while the I^2C is in the addressed slave receiver mode

The AA bit can be cleared by writing 1 to the AAC bit in the CONCLR register. When AA is 0, a not acknowledge (HIGH level to SDA) will be returned during the acknowledge clock pulse on the SCL line on the following situations:

1. A data byte has been received while the I^2C is in the master receiver mode.
2. A data byte has been received while the I^2C is in the addressed slave receiver mode.

42.7.2 I²C Status register

Each I²C Status register reflects the condition of the corresponding I²C interface. The I²C Status register is Read-Only.

Table 954. I²C Status register (STAT - address 0x400A 1004 (I2C0) and 0x400E 0004 (I2C1)) bit description

Bit	Symbol	Description	Reset value
2:0	-	These bits are unused and are always 0.	0
7:3	Status	These bits give the actual status information about the I²C interface.	0x1F
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

The three least significant bits are always 0. Taken as a byte, the status register contents represent a status code. There are 26 possible status codes. When the status code is 0xF8, there is no relevant information available and the SI bit is not set. All other 25 status codes correspond to defined I²C states. When any of these states entered, the SI bit will be set. For a complete list of status codes, refer to tables from Table 969 to Table 974.

42.7.3 I²C Data register

This register contains the data to be transmitted or the data just received. The CPU can read and write to this register only while it is not in the process of shifting a byte, when the SI bit is set. Data in DAT remains stable as long as the SI bit is set. Data in DAT is always shifted from right to left: the first bit to be transmitted is the MSB (bit 7), and after a byte has been received, the first bit of received data is located at the MSB of DAT.

Table 955. I²C Data register (DAT - 0x400A 1008 (I2C0) and 0x400E 0008 (I2C1)) bit description

Bit	Symbol	Description	Reset value
7:0	Data	This register holds data values that have been received or are to be transmitted.	0
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

42.7.4 I²C Slave Address register 0

This register is readable and writable and are only used when an I²C interface is set to slave mode. In master mode, this register has no effect. The LSB of ADR is the General Call bit. When this bit is set, the General Call address (0x00) is recognized.

If this register contains 0x00, the I²C will not acknowledge any address on the bus. This register will be cleared to this disabled state on reset. See also Table 962.

Table 956. I²C Slave Address register 0 (ADR0 - address 0x400A 100C (I2C0) and 0x400E 000C (I2C1)) bit description

Bit	Symbol	Description	Reset value
0	GC	General Call enable bit.	0
7:1	Address	The I²C device address for slave mode.	0x00
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

42.7.5 I²C SCL HIGH and LOW duty cycle registers

Table 957. I²C SCL HIGH Duty Cycle register (SCLH - address 0x400A 1010 (I2C0) and 0x400E 0010 (I2C1)) bit description

Bit	Symbol	Description	Reset value
15:0	SCLH	Count for SCL HIGH time period selection.	0x0004
31:16	-	Reserved. The value read from a reserved bit is not defined.	-

Table 958. I²C SCL Low duty cycle register (SCLL - address 0x400A 1014 (I2C0) and 0x400E 0014 (I2C1)) bit description

Bit	Symbol	Description	Reset value
15:0	SCLL	Count for SCL low time period selection.	0x0004
31:16	-	Reserved. The value read from a reserved bit is not defined.	-

42.7.5.1 Selecting the appropriate I²C data rate and duty cycle

Software must set values for the registers SCLH and SCLL to select the appropriate data rate and duty cycle. SCLH defines the number of C_PCLK cycles for the SCL HIGH time, SCLL defines the number of I2C_PCLK cycles for the SCL low time. The frequency is determined by the following formula (I2C_PCLK is the frequency of the peripheral I2C clock):

(10)

$$I^2C_{bitfrequency} = \frac{I2CPCLK}{I2CSCLH + I2CSCLL}$$

The values for SCLL and SCLH must ensure that the data rate is in the appropriate I²C data rate range. Each register value must be greater than or equal to 4. Table 959 gives some examples of I²C-bus rates based on I2C_PCLK frequency and SCLL and SCLH values.

Table 959. SCLL + SCLH values for selected I²C clock values

I²C mode	I²C bit frequency	I2C_PCLK (MHz)								
		6	8	10	12	16	20	30	40	50
		SCLH + SCLL								
Standard mode	100 kHz	60	80	100	120	160	200	300	400	500
Fast-mode	400 kHz	15	20	25	30	40	50	75	100	125
Fast-mode Plus	1 MHz	-	8	10	12	16	20	30	40	50

SCLL and SCLH values should not necessarily be the same. Software can set different duty cycles on SCL by setting these two registers. For example, the I²C-bus specification defines the SCL low time and high time at different values for a Fast-mode and Fast-mode Plus I²C.

42.7.6 I²C Control Clear register

The CONCLR registers control clearing of bits in the CON register that controls operation of the I²C interface. Writing a one to a bit of this register causes the corresponding bit in the I²C control register to be cleared. Writing a zero has no effect.

Table 960. I²C Control Clear register (CONCLR - address 0x400A 1018 and 0x400E 0018 (I2C1)) bit description

Bit	Symbol	Description	Reset value
1:0	-	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
2	AAC	Assert acknowledge Clear bit.	
3	SIC	I²C interrupt Clear bit.	0
4	-	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
5	STAC	START flag Clear bit.	0
6	I2ENC	I²C interface Disable bit.	0
7	-	Reserved. User software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

AAC is the Assert Acknowledge Clear bit. Writing a 1 to this bit clears the AA bit in the CONSET register. Writing 0 has no effect.

SIC is the I²C Interrupt Clear bit. Writing a 1 to this bit clears the SI bit in the CONSET register. Writing 0 has no effect.

STAC is the START flag Clear bit. Writing a 1 to this bit clears the STA bit in the CONSET register. Writing 0 has no effect.

I**2ENC** is the I²C Interface Disable bit. Writing a 1 to this bit clears the I2EN bit in the CONSET register. Writing 0 has no effect.

42.7.7 I²C Monitor mode control register

This register controls the Monitor mode which allows the I²C module to monitor traffic on the I²C bus without actually participating in traffic or interfering with the I²C bus.

Table 961. I²C Monitor mode control register (MMCTRL - address 0x400A 101C (I2C0) and 0x400E 001C (I2C1)) bit description

Bit	Symbol	Value	Description	Reset value
0	MM_ENA		Monitor mode enable.	0
		0	Monitor mode disabled.	
		1	The I²C module will enter monitor mode. In this mode the SDA output will be forced high. This will prevent the I²C module from outputting data of any kind (including ACK) onto the I²C data bus. Depending on the state of the ENA_SCL bit, the output may be also forced high, preventing the module from having control over the I²C clock line.	
1	ENA_SCL		SCL output enable.	0
		0	When this bit is cleared to '0', the SCL output will be forced high when the module is in monitor mode. As described above, this will prevent the module from having any control over the I²C clock line.	
		1	When this bit is set, the I²C module may exercise the same control over the clock line that it would in normal operation. This means that, acting as a slave peripheral, the I²C module can "stretch" the clock line (hold it low) until it has had time to respond to an I²C interrupt.[1]	
2	MATCH_ALL		Select interrupt register match.	0
		0	When this bit is cleared, an interrupt will only be generated when a match occurs to one of the (up-to) four address registers described above. That is, the module will respond as a normal slave as far as address-recognition is concerned.	
		1	When this bit is set to '1' and the I2C is in monitor mode, an interrupt will be generated on ANY address received. This will enable the part to monitor all traffic on the bus.	
31:3	-	-	Reserved. The value read from reserved bits is not defined.	-

[1] When the ENA_SCL bit is cleared and the I²C no longer has the ability to stall the bus, interrupt response time becomes important. To give the part more time to respond to an I²C interrupt under these conditions, a DATA _BUFFER register is used (Section 42.7.9) to hold received data for a full 9-bit word transmission time.

Remark: The ENA_SCL and MATCH_ALL bits have no effect if the MM_ENA is '0' (i.e. if the module is NOT in monitor mode).

42.7.7.1 Interrupt in Monitor mode

All interrupts will occur as normal when the module is in monitor mode. This means that the first interrupt will occur when an address-match is detected (any address received if the MATCH_ALL bit is set, otherwise an address matching one of the four address registers).

Subsequent to an address-match detection, interrupts will be generated after each data byte is received for a slave-write transfer, or after each byte that the module “thinks” it has transmitted for a slave-read transfer. In this second case, the data register will actually contain data transmitted by some other slave on the bus which was actually addressed by the master.

Following all of these interrupts, the processor may read the data register to see what was actually transmitted on the bus.

42.7.7.2 Loss of arbitration in Monitor mode

In monitor mode, the I2C module will not be able to respond to a request for information by the bus master or issue an ACK). Some other slave on the bus will respond instead. This will most probably result in a lost-arbitration state as far as our module is concerned.

Software should be aware of the fact that the module is in monitor mode and should not respond to any loss of arbitration state that is detected.

42.7.8 I2C Slave Address registers

These registers are readable and writable and are only used when an I2C interface is set to slave mode. In master mode, this register has no effect. The LSB of ADR is the General Call bit. When this bit is set, the General Call address (0x00) is recognized.

If these registers contain 0x00, the I2C will not acknowledge any address on the bus. All four registers (including ADR0, see Table 956) will be cleared to this disabled state on reset.

Table 962. I2C Slave Address registers (ADR - address 0x400A 1020 (ADR1) to 0x400A 1028 (ADR3) (I2C0) and 0x400E 0020 (ADR1) to 0x400E 0028 (ADR3) (I2C1)) bit description

Bit	Symbol	Description	Reset value
0	GC	General Call enable bit.	0
7:1	Address	The I2C device address for slave mode.	0x00
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

42.7.9 I2C Data buffer register

In monitor mode, the I2C module may lose the ability to stretch the clock (stall the bus) if the ENA_SCL bit is not set. This means that the processor will have a limited amount of time to read the contents of the data received on the bus. If the processor reads the DAT shift register, as it ordinarily would, it could have only one bit-time to respond to the interrupt before the received data is overwritten by new data.

To give the processor more time to respond, a new 8-bit, read-only DATA_BUFFER register will be added. The contents of the 8 MSBs of the DAT shift register will be transferred to the DATA_BUFFER automatically after every nine bits (8 bits of data plus ACK or NACK) has been received on the bus. This means that the processor will have nine bit transmission times to respond to the interrupt and read the data before it is overwritten.

The processor will still have the ability to read DAT directly, as usual, and the behavior of DAT will not be altered in any way.

Although the DATA_BUFFER register is primarily intended for use in monitor mode with the ENA_SCL bit = ‘0’, it will be available for reading at any time under any mode of operation.

Table 963. I²C Data buffer register (DATA_BUFFER - address 0x400A 102C (I2C0) and 0x400E 002C (I2C1)) bit description

Bit	Symbol	Description	Reset value
7:0	Data	This register holds contents of the 8 MSBs of the DAT shift register.	0
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

42.7.10 I²C Mask registers

The four mask registers each contain seven active bits (7:1). Any bit in these registers which is set to ‘1’ will cause an automatic compare on the corresponding bit of the received address when it is compared to the ADRn register associated with that mask register. In other words, bits in an ADRn register which are masked are not taken into account in determining an address match.

On reset, all mask register bits are cleared to ‘0’.

The mask register has no effect on comparison to the General Call address (“0000000”).

Bits(31:8) and bit(0) of the mask registers are unused and should not be written to. These bits will always read back as zeros.

When an address-match interrupt occurs, the processor will have to read the data register (DAT) to determine what the received address was that actually caused the match.

Table 964. I²C Mask registers (MASK - address 0x400A 1030 (MASK0) to 0x400A 103C (MASK3) (I2C0) and 0x400E 0030 (MASK0) to 0x400E 003C (MASK3) (I2C1)) bit description

Bit	Symbol	Description	Reset value
0	-	Reserved. User software should not write ones to reserved bits. This bit reads always back as 0.	0
7:1	MASK	Mask bits.	0x00
31:8	-	Reserved. The value read from a reserved bit is not defined.	-

42.8 I²C operating modes

In a given application, the I²C block may operate as a master, a slave, or both. In the slave mode, the I²C hardware looks for any one of its four slave addresses and the General Call address. If one of these addresses is detected, an interrupt is requested. If the processor wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave operation is not interrupted. If bus arbitration is lost in the master mode, the I²C block switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

42.8.1 Master Transmitter mode

In this mode data is transmitted from master to slave. Before the master transmitter mode can be entered, the CONSET register must be initialized as shown in Table 965. I2EN must be set to 1 to enable the I^2C function. If the AA bit is 0, the I^2C interface will not acknowledge any address when another device is master of the bus, so it can not enter slave mode. The STA, STO and SI bits must be 0. The SI Bit is cleared by writing 1 to the SIC bit in the CONCLR register. The STA bit should be cleared after writing the slave address.

Table 965. CONSET used to configure Master mode

Bit	7	6	5	4	3	2	1	0
Symbol	-	I2EN	STA	STO	SI	AA	-	-
Value	-	1	0	0	0	0	-	-

The first byte transmitted contains the slave address of the receiving device (7 bits) and the data direction bit. In this mode the data direction bit (R/W) should be 0 which means Write. The first byte transmitted contains the slave address and Write bit. Data is transmitted 8 bits at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

The I^2C interface will enter master transmitter mode when software sets the STA bit. The I^2C logic will send the START condition as soon as the bus is free. After the START condition is transmitted, the SI bit is set, and the status code in the STAT register is 0x08. This status code is used to vector to a state service routine which will load the slave address and Write bit to the DAT register, and then clear the SI bit. SI is cleared by writing a 1 to the SIC bit in the CONCLR register.

When the slave address and R/W bit have been transmitted and an acknowledgment bit has been received, the SI bit is set again, and the possible status codes now are 0x18, 0x20, or 0x38 for the master mode, or 0x68, 0x78, or 0xB0 if the slave mode was enabled (by setting AA to 1). The appropriate actions to be taken for each of these status codes are shown in Table 969 to Table 974.

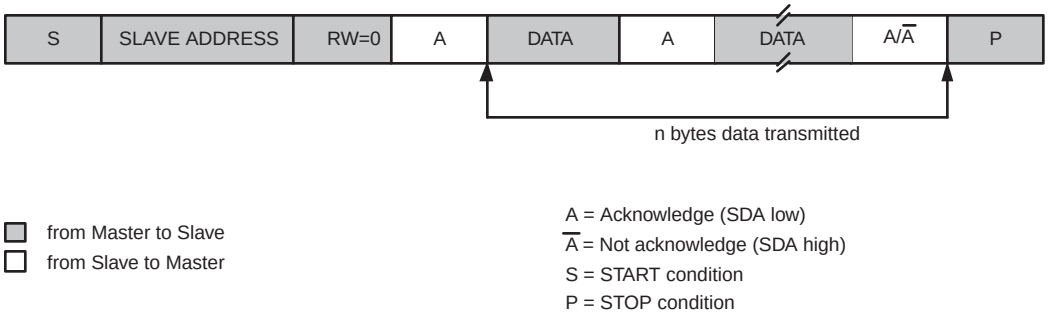


Fig 156. Format in the Master Transmitter mode

42.8.2 Master Receiver mode

In the master receiver mode, data is received from a slave transmitter. The transfer is initiated in the same way as in the master transmitter mode. When the START condition has been transmitted, the interrupt service routine must load the slave address and the data direction bit to the I^2C Data register (DAT), and then clear the SI bit. In this case, the data direction bit (R/W) should be 1 to indicate a read.

When the slave address and data direction bit have been transmitted and an acknowledge bit has been received, the SI bit is set, and the Status Register will show the status code. For master mode, the possible status codes are 0x40, 0x48, or 0x38. For slave mode, the possible status codes are 0x68, 0x78, or 0xB0. For details, refer to Table 970.

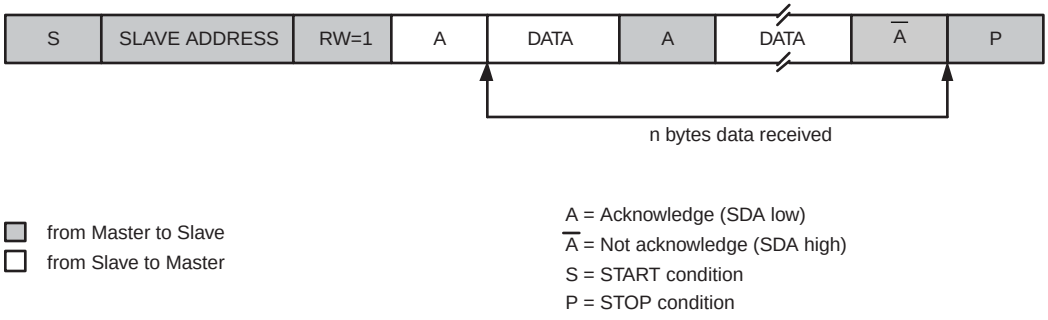


Fig 157. Format of Master Receiver mode

After a Repeated START condition, I^2C may switch to the master transmitter mode.

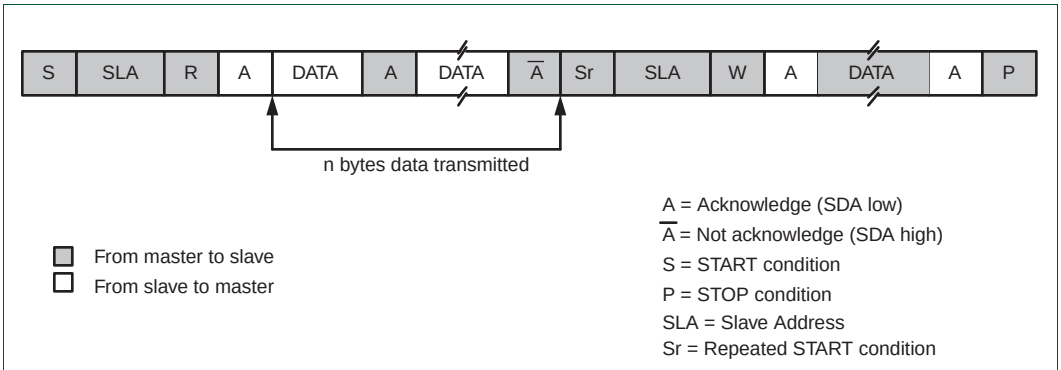


Fig 158. A Master Receiver switches to Master Transmitter after sending Repeated START

42.8.3 Slave Receiver mode

In the slave receiver mode, data bytes are received from a master transmitter. To initialize the slave receiver mode, write any of the Slave Address registers (ADR0-3) and write the I^2C Control Set register (CONSET) as shown in Table 966.

Table 966. CONSET used to configure Slave mode

Bit	7	6	5	4	3	2	1	0
Symbol	-	I2EN	STA	STO	SI	AA	-	-
Value	-	1	0	0	0	1	-	-

I2EN must be set to 1 to enable the I^2C function. AA bit must be set to 1 to acknowledge its own slave address or the General Call address. The STA, STO and SI bits are set to 0.

After ADR and CONSET are initialized, the I^2C interface waits until it is addressed by its own address or general address followed by the data direction bit. If the direction bit is 0 (W), it enters slave receiver mode. If the direction bit is 1 (R), it enters slave transmitter mode. After the address and direction bit have been received, the SI bit is set and a valid status code can be read from the Status register (STAT). Refer to Table 973 for the status codes and actions.

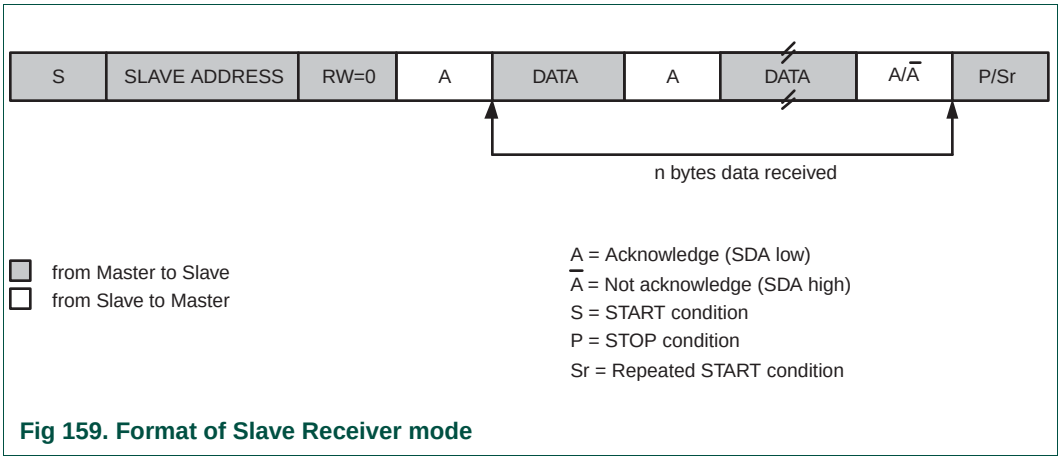


Fig 159. Format of Slave Receiver mode

42.8.4 Slave Transmitter mode

The first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will be 1, indicating a read operation. Serial data is transmitted via SDA while the serial clock is input through SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer. In a given application, I^2C may operate as a master and as a slave. In the slave mode, the I^2C hardware looks for its own slave address and the General Call address. If one of these addresses is detected, an interrupt is requested. When the microcontrollers wishes to become the bus master, the hardware waits until the bus is free before the master mode is entered so that a possible slave action is not interrupted. If bus arbitration is lost in the master mode, the I^2C interface switches to the slave mode immediately and can detect its own slave address in the same serial transfer.

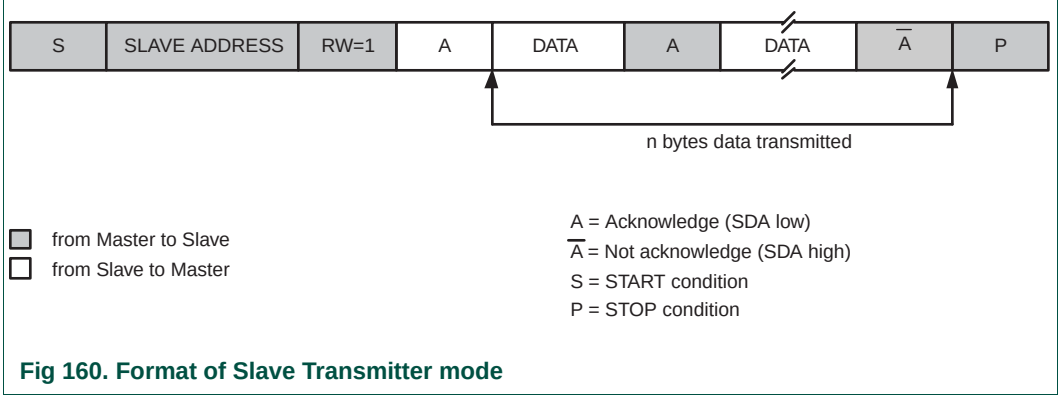


Fig 160. Format of Slave Transmitter mode

42.9 I²C implementation and operation

Figure 161 shows how the on-chip I²C-bus interface is implemented, and the following text describes the individual blocks.

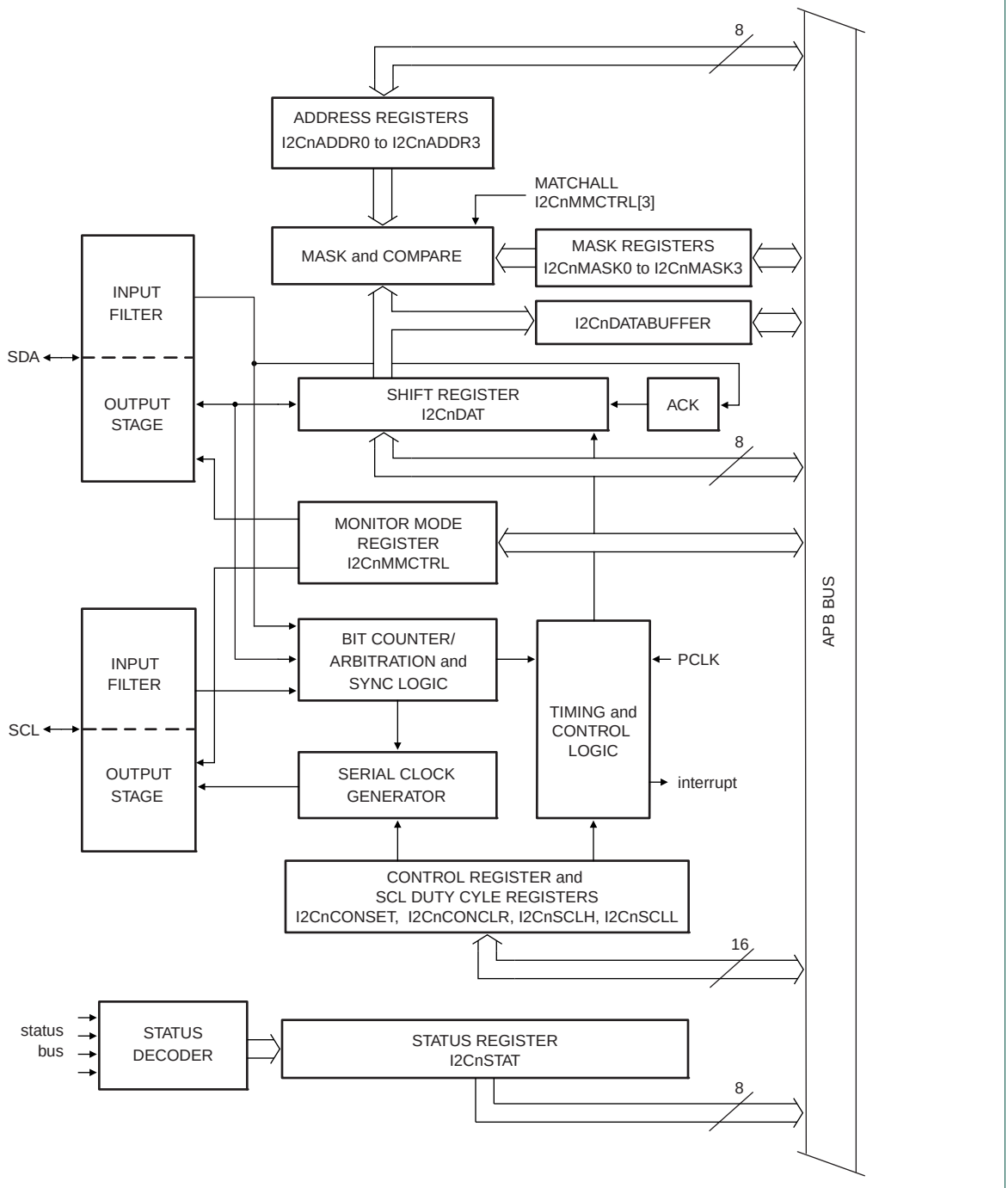


Fig 161. I²C serial interface block diagram

42.9.1 Input filters and output stages

Input signals are synchronized with the internal clock, and spikes shorter than three clocks are filtered out.

The output for I²C is a special pad designed to conform to the I²C specification.

42.9.2 Address Registers, ADR0 to ADR3

These registers may be loaded with the 7-bit slave address (7 most significant bits) to which the I^2C block will respond when programmed as a slave transmitter or receiver. The LSB (GC) is used to enable General Call address (0x00) recognition. When multiple slave addresses are enabled, the actual address received may be read from the DAT register at the state where the own slave address has been received.

42.9.3 Address mask registers, MASK0 to MASK3

The four mask registers each contain seven active bits (7:1). Any bit in these registers which is set to '1' will cause an automatic compare on the corresponding bit of the received address when it is compared to the ADRn register associated with that mask register. In other words, bits in an ADRn register which are masked are not taken into account in determining an address match.

When an address-match interrupt occurs, the processor will have to read the data register (I2DAT) to determine which received address actually caused the match.

42.9.4 Comparator

The comparator compares the received 7-bit slave address with its own slave address (7 most significant bits in ADR). It also compares the first received 8-bit byte with the General Call address (0x00). If an equality is found, the appropriate status bits are set and an interrupt is requested.

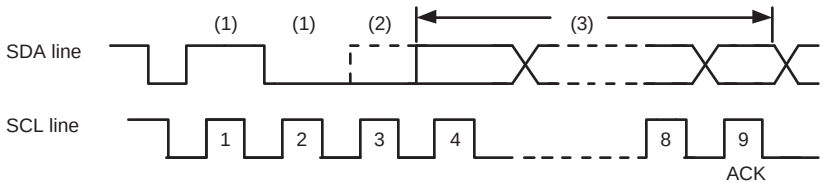
42.9.5 Shift register, DAT

This 8-bit register contains a byte of serial data to be transmitted or a byte which has just been received. Data in DAT is always shifted from right to left; the first bit to be transmitted is the MSB (bit 7) and, after a byte has been received, the first bit of received data is located at the MSB of DAT. While data is being shifted out, data on the bus is simultaneously being shifted in; DAT always contains the last byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in DAT.

42.9.6 Arbitration and synchronization logic

In the master transmitter mode, the arbitration logic checks that every transmitted logic 1 actually appears as a logic 1 on the I^2C-bus. If another device on the bus overrules a logic 1 and pulls the SDA line low, arbitration is lost, and the I^2C block immediately changes from master transmitter to slave receiver. The I^2C block will continue to output clock pulses (on SCL) until transmission of the current serial byte is complete.

Arbitration may also be lost in the master receiver mode. Loss of arbitration in this mode can only occur while the I^2C block is returning a "not acknowledge": (logic 1) to the bus. Arbitration is lost when another device on the bus pulls this signal low. Since this can occur only at the end of a serial byte, the I^2C block generates no further clock pulses. Figure 162 shows the arbitration procedure.



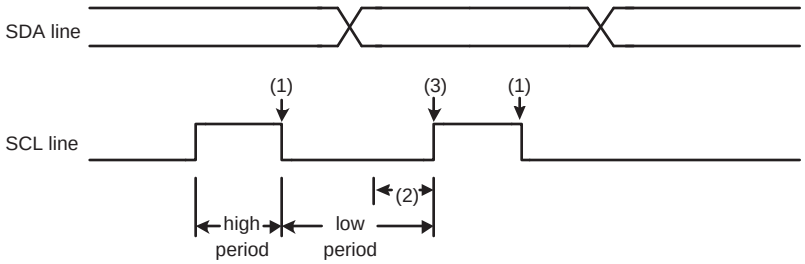
(1) Another device transmits serial data.

(2) Another device overrules a logic (dotted line) transmitted this I2C master by pulling the SDA line low. Arbitration is lost, and this I2C enters Slave Receiver mode.

(3) This I2C is in Slave Receiver mode but still generates clock pulses until the current byte has been transmitted. This I2C will not generate clock pulses for the next byte. Data on SDA originates from the new master once it has won arbitration.

Fig 162. Arbitration procedure

The synchronization logic will synchronize the serial clock generator with the clock pulses on the SCL line from another device. If two or more master devices generate clock pulses, the “mark” duration is determined by the device that generates the shortest “marks”, and the “space” duration is determined by the device that generates the longest “spaces”. Figure 163 shows the synchronization procedure.



(1) Another device pulls the SCL line low before this I2C has timed a complete high time. The other device effectively determines the (shorter) HIGH period.

(2) Another device continues to pull the SCL line low after this I2C has timed a complete low time and released SCL. The I2C clock generator is forced to wait until SCL goes HIGH. The other device effectively determines the (longer) LOW period.

(3) The SCL line is released , and the clock generator begins timing the HIGH time.

Fig 163. Serial clock synchronization

A slave may stretch the space duration to slow down the bus master. The space duration may also be stretched for handshaking purposes. This can be done after each bit or after a complete byte transfer. the I2C block will stretch the SCL space duration after a byte has been transmitted or received and the acknowledge bit has been transferred. The serial interrupt flag (SI) is set, and the stretching continues until the serial interrupt flag is cleared.

42.9.7 Serial clock generator

This programmable clock pulse generator provides the SCL clock pulses when the I2C block is in the master transmitter or master receiver mode. It is switched off when the I2C block is in slave mode. The I2C output clock frequency and duty cycle is programmable

via the I2C Clock Control Registers. See the description of the SCLL and SCLH registers for details. The output clock pulses have a duty cycle as programmed unless the bus is synchronizing with other SCL clock sources as described above.

42.9.8 Timing and control

The timing and control logic generates the timing and control signals for serial byte handling. This logic block provides the shift pulses for DAT, enables the comparator, generates and detects START and STOP conditions, receives and transmits acknowledge bits, controls the master and slave modes, contains interrupt request logic, and monitors the I2C-bus status.

42.9.9 Control register, CONSET and CONCLR

The I2C control register contains bits used to control the following I2C block functions: start and restart of a serial transfer, termination of a serial transfer, bit rate, address recognition, and acknowledgment.

The contents of the I2C control register may be read as CONSET. Writing to CONSET will set bits in the I2C control register that correspond to ones in the value written. Conversely, writing to CONCLR will clear bits in the I2C control register that correspond to ones in the value written.

42.9.10 Status decoder and status register

The status decoder takes all of the internal status bits and compresses them into a 5-bit code. This code is unique for each I2C-bus status. The 5-bit code may be used to generate vector addresses for fast processing of the various service routines. Each service routine processes a particular bus status. There are 26 possible bus states if all four modes of the I2C block are used. The 5-bit status code is latched into the five most significant bits of the status register when the serial interrupt flag is set (by hardware) and remains stable until the interrupt flag is cleared by software. The three least significant bits of the status register are always zero. If the status code is used as a vector to service routines, then the routines are displaced by eight address locations. Eight bytes of code is sufficient for most of the service routines (see the software example in this section).

42.10 Details of I2C operating modes

The four operating modes are:

- Master Transmitter
- Master Receiver
- Slave Receiver
- Slave Transmitter

Data transfers in each mode of operation are shown in Figure 164, Figure 165, Figure 166, Figure 167, and Figure 168. Table 967 lists abbreviations used in these figures when describing the I2C operating modes.

Table 967. Abbreviations used to describe an I²C operation

Abbreviation	Explanation
S	START Condition
SLA	7-bit slave address
R	Read bit (HIGH level at SDA)
W	Write bit (LOW level at SDA)
A	Acknowledge bit (LOW level at SDA)
$\overline{A}$	Not acknowledge bit (HIGH level at SDA)
Data	8-bit data byte
P	STOP condition

In Figure 164 to Figure 168, circles are used to indicate when the serial interrupt flag is set. The numbers in the circles show the status code held in the STAT register. At these points, a service routine must be executed to continue or complete the serial transfer. These service routines are not critical since the serial transfer is suspended until the serial interrupt flag is cleared by software.

When a serial interrupt routine is entered, the status code in STAT is used to branch to the appropriate service routine. For each status code, the required software action and details of the following serial transfer are given in tables from Table 969 to Table 975.

42.10.1 Master Transmitter mode

In the master transmitter mode, a number of data bytes are transmitted to a slave receiver (see Figure 164). Before the master transmitter mode can be entered, CON must be initialized as follows:

Table 968. CONSET used to initialize Master Transmitter mode

Bit	7	6	5	4	3	2	1	0
Symbol	-	I2EN	STA	STO	SI	AA	-	-
Value	-	1	0	0	0	x	-	-

The I²C rate must also be configured in the SCLL and SCLH registers. I2EN must be set to logic 1 to enable the I²C block. If the AA bit is reset, the I²C block will not acknowledge its own slave address or the General Call address in the event of another device becoming master of the bus. In other words, if AA is reset, the I²C interface cannot enter slave mode. STA, STO, and SI must be reset.

The master transmitter mode may now be entered by setting the STA bit. The I^2C logic will now test the I^2C-bus and generate a START condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI) is set, and the status code in the status register (STAT) will be 0x08. This status code is used by the interrupt service routine to enter the appropriate state service routine that loads DAT with the slave address and the data direction bit (SLA+W). The SI bit in CON must then be reset before the serial transfer can continue.

When the slave address and the direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in STAT are possible. There are 0x18, 0x20, or 0x38 for the master mode and also 0x68, 0x78, or 0xB0 if the slave mode was enabled (AA = logic 1). The appropriate action to be taken for each of these status codes is detailed in Table 969. After a Repeated START condition (state 0x10). The I^2C block may switch to the master receiver mode by loading DAT with SLA+R).

Table 969. Master Transmitter mode

Status Code (STAT)	Status of the I^2C-bus and hardware	Application software response					Next action taken by I^2C hardware
		To/From DAT	To CON				
			STA	STO	SI	AA	
0x08	A START condition has been transmitted.	Load SLA+W; clear STA	X	0	0	X	SLA+W will be transmitted; ACK bit will be received.
0x10	A Repeated START condition has been transmitted.	Load SLA+W or	X	0	0	X	As above.
		Load SLA+R; Clear STA	X	0	0	X	SLA+R will be transmitted; the I^2C block will be switched to MST/REC mode.
0x18	SLA+W has been transmitted; ACK has been received.	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received.
		No DAT action or	1	0	0	X	Repeated START will be transmitted.
		No DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
		No DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
0x20	SLA+W has been transmitted; NOT ACK has been received.	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received.
		No DAT action or	1	0	0	X	Repeated START will be transmitted.
		No DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
		No DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
0x28	Data byte in DAT has been transmitted; ACK has been received.	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received.
		No DAT action or	1	0	0	X	Repeated START will be transmitted.
		No DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
		No DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
0x30	Data byte in DAT has been transmitted; NOT ACK has been received.	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received.
		No DAT action or	1	0	0	X	Repeated START will be transmitted.
		No DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
		No DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
0x38	Arbitration lost in SLA+R/W or Data bytes.	No DAT action or	0	0	0	X	I^2C-bus will be released; not addressed slave will be entered.
		No DAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free.

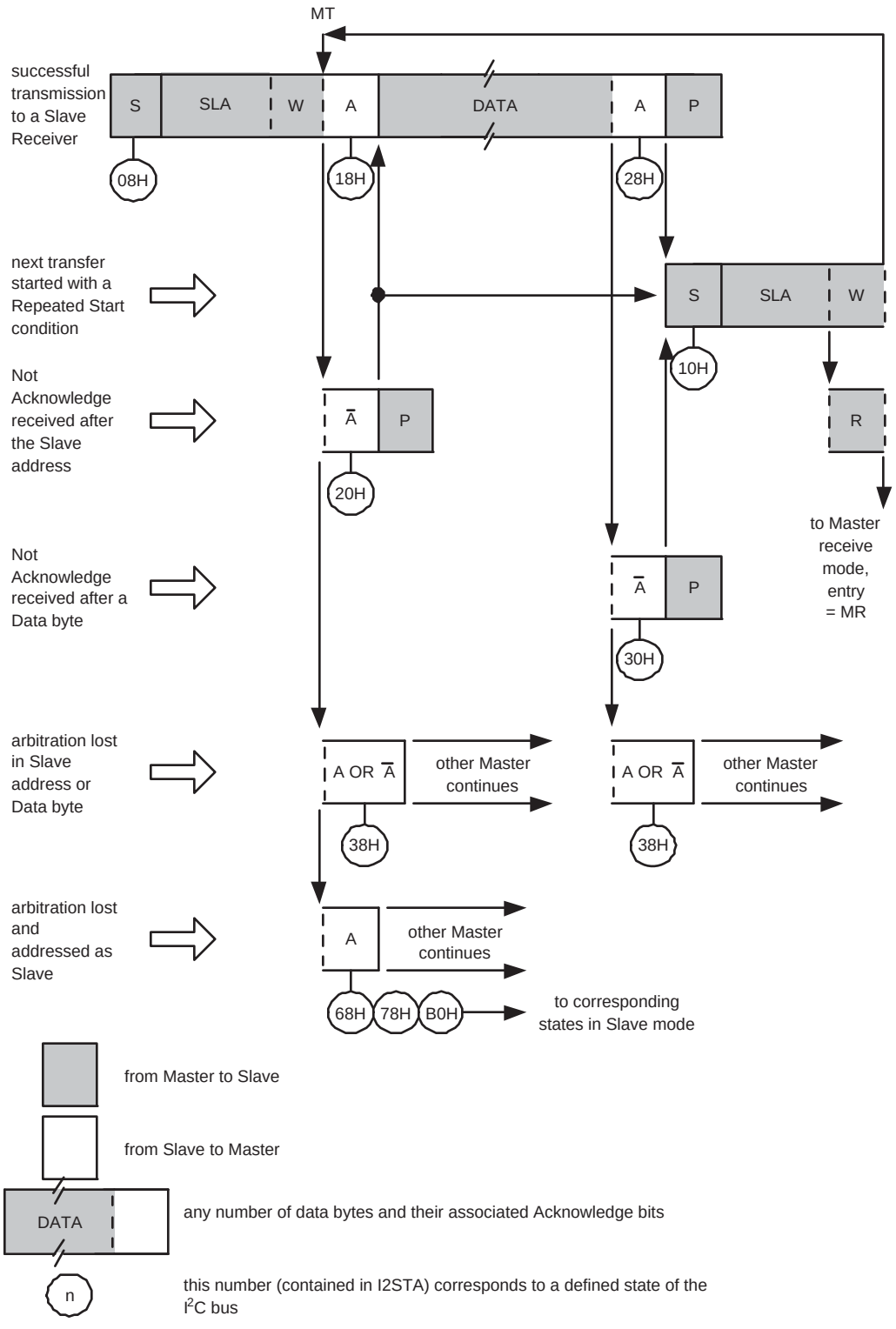


Fig 164. Format and states in the Master Transmitter mode

42.10.2 Master Receiver mode

In the master receiver mode, a number of data bytes are received from a slave transmitter (see Figure 165). The transfer is initialized as in the master transmitter mode. When the START condition has been transmitted, the interrupt service routine must load DAT with the 7-bit slave address and the data direction bit (SLA+R). The SI bit in CON must then be cleared before the serial transfer can continue.

When the slave address and the data direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in STAT are possible. These are 0x40, 0x48, or 0x38 for the master mode and also 0x68, 0x78, or 0xB0 if the slave mode was enabled (AA = 1). The appropriate action to be taken for each of these status codes is detailed in Table 970. After a Repeated START condition (state 0x10), the I^2C block may switch to the master transmitter mode by loading DAT with SLA+W.

Table 970. Master Receiver mode

Status Code (STAT)	Status of the I^2C-bus and hardware	Application software response					Next action taken by I^2C hardware
		To/From DAT	To CON				
			STA	STO	SI	AA	
0x08	A START condition has been transmitted.	Load SLA+R	X	0	0	X	SLA+R will be transmitted; ACK bit will be received.
0x10	A Repeated START condition has been transmitted.	Load SLA+R or	X	0	0	X	As above.
		Load SLA+W	X	0	0	X	SLA+W will be transmitted; the I^2C block will be switched to MST/TRX mode.
0x38	Arbitration lost in NOT ACK bit.	No DAT action or	0	0	0	X	I^2C-bus will be released; the I^2C block will enter slave mode.
		No DAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free.
0x40	SLA+R has been transmitted; ACK has been received.	No DAT action or	0	0	0	0	Data byte will be received; NOT ACK bit will be returned.
		No DAT action	0	0	0	1	Data byte will be received; ACK bit will be returned.
0x48	SLA+R has been transmitted; NOT ACK has been received.	No DAT action or	1	0	0	X	Repeated START condition will be transmitted.
		No DAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
		No DAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
0x50	Data byte has been received; ACK has been returned.	Read data byte or	0	0	0	0	Data byte will be received; NOT ACK bit will be returned.
		Read data byte	0	0	0	1	Data byte will be received; ACK bit will be returned.
0x58	Data byte has been received; NOT ACK has been returned.	Read data byte or	1	0	0	X	Repeated START condition will be transmitted.
		Read data byte or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
		Read data byte	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.

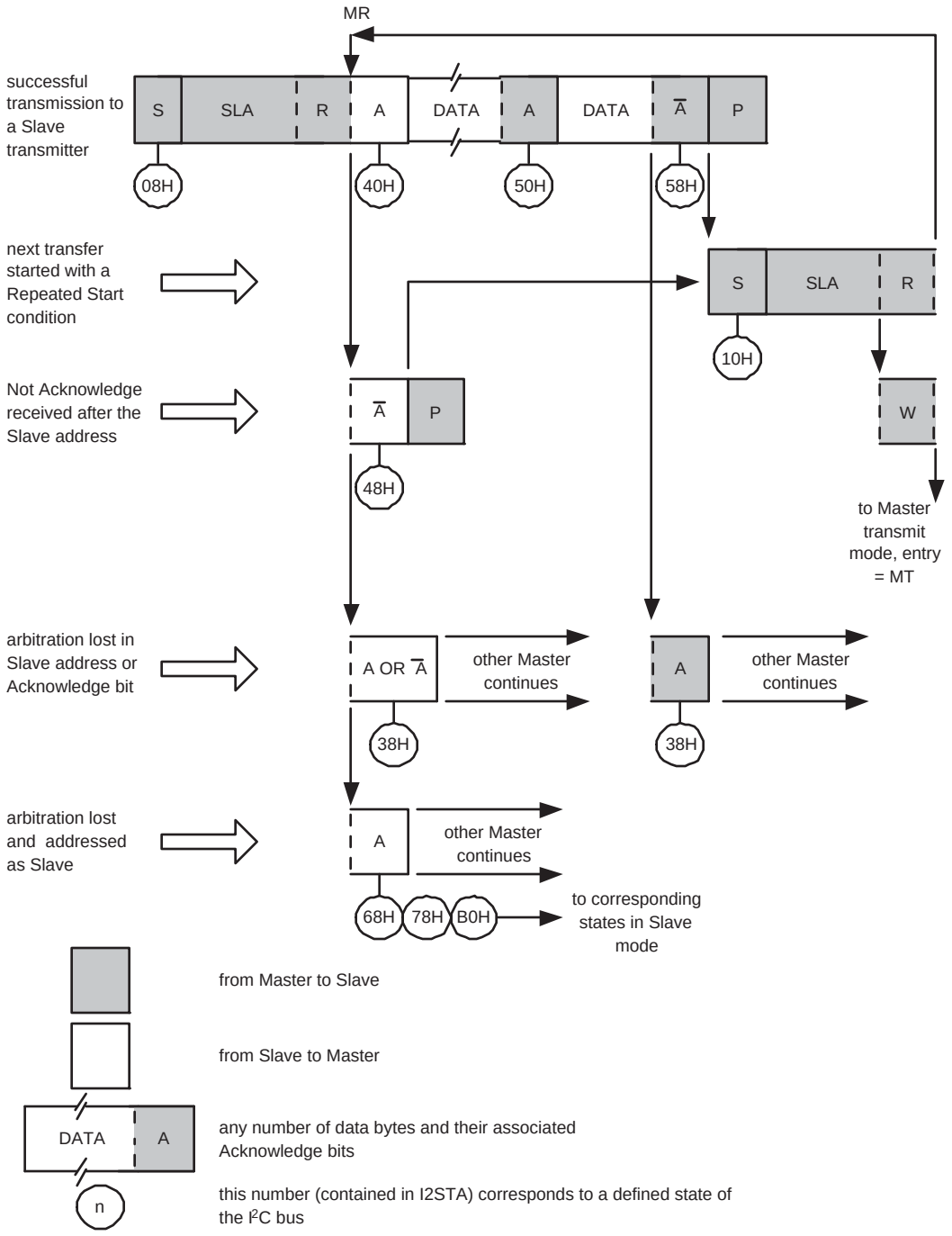


Fig 165. Format and states in the Master Receiver mode

42.10.3 Slave Receiver mode

In the slave receiver mode, a number of data bytes are received from a master transmitter (see Figure 166). To initiate the slave receiver mode, ADR and CON must be loaded as follows:

Table 971. ADR usage in Slave Receiver mode

Bit	7	6	5	4	3	2	1	0
Symbol	own slave 7-bit address							GC

The upper 7 bits are the address to which the I^2C block will respond when addressed by a master. If the LSB (GC) is set, the I^2C block will respond to the General Call address (0x00); otherwise it ignores the General Call address.

Table 972. CONSET used to initialize Slave Receiver mode

Bit	7	6	5	4	3	2	1	0
Symbol	-	I2EN	STA	STO	SI	AA	-	-
Value	-	1	0	0	0	1	-	-

The I^2C-bus rate settings do not affect the I^2C block in the slave mode. I2EN must be set to logic 1 to enable the I^2C block. The AA bit must be set to enable the I^2C block to acknowledge its own slave address or the General Call address. STA, STO, and SI must be reset.

When ADR and CON have been initialized, the I^2C block waits until it is addressed by its own slave address followed by the data direction bit which must be “0” (W) for the I^2C block to operate in the slave receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag (SI) is set and a valid status code can be read from STAT. This status code is used to vector to a state service routine. The appropriate action to be taken for each of these status codes is detailed in Table 973. The slave receiver mode may also be entered if arbitration is lost while the I^2C block is in the master mode (see status 0x68 and 0x78).

If the AA bit is reset during a transfer, the I^2C block will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, the I^2C block does not respond to its own slave address or a General Call address. However, the I^2C-bus is still monitored and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate the I^2C block from the I^2C-bus.

Table 973. Slave Receiver mode

Status Code (STAT)	Status of the I^2C-bus and hardware	Application software response					Next action taken by I^2C hardware
		To/From DAT	To CON				
			STA	STO	SI	AA	
0x60	Own SLA+W has been received; ACK has been returned.	No DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned.
		No DAT action	X	0	0	1	Data byte will be received and ACK will be returned.
0x68	Arbitration lost in SLA+R/W as master; Own SLA+W has been received, ACK returned.	No DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned.
		No DAT action	X	0	0	1	Data byte will be received and ACK will be returned.
0x70	General call address (0x00) has been received; ACK has been returned.	No DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned.
		No DAT action	X	0	0	1	Data byte will be received and ACK will be returned.
0x78	Arbitration lost in SLA+R/W as master; General call address has been received, ACK has been returned.	No DAT action or	X	0	0	0	Data byte will be received and NOT ACK will be returned.
		No DAT action	X	0	0	1	Data byte will be received and ACK will be returned.
0x80	Previously addressed with own SLV address; DATA has been received; ACK has been returned.	Read data byte or	X	0	0	0	Data byte will be received and NOT ACK will be returned.
		Read data byte	X	0	0	1	Data byte will be received and ACK will be returned.
0x88	Previously addressed with own SLA; DATA byte has been received; NOT ACK has been returned.	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address.
		Read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1.
		Read data byte or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		Read data byte	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1. A START condition will be transmitted when the bus becomes free.
0x90	Previously addressed with General Call; DATA byte has been received; ACK has been returned.	Read data byte or	X	0	0	0	Data byte will be received and NOT ACK will be returned.
		Read data byte	X	0	0	1	Data byte will be received and ACK will be returned.

Table 973. Slave Receiver mode *…continued*

Status Code (STAT)	Status of the I²C-bus and hardware	Application software response					Next action taken by I²C hardware
		To/From DAT	To CON				
			STA	STO	SI	AA	
0x98	Previously addressed with General Call; DATA byte has been received; NOT ACK has been returned.	Read data byte or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address.
		Read data byte or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1.
		Read data byte or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		Read data byte	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1. A START condition will be transmitted when the bus becomes free.
0xA0	A STOP condition or Repeated START condition has been received while still addressed as SLV/REC or SLV/TRX.	No STDAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address.
		No STDAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1.
		No STDAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		No STDAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1. A START condition will be transmitted when the bus becomes free.

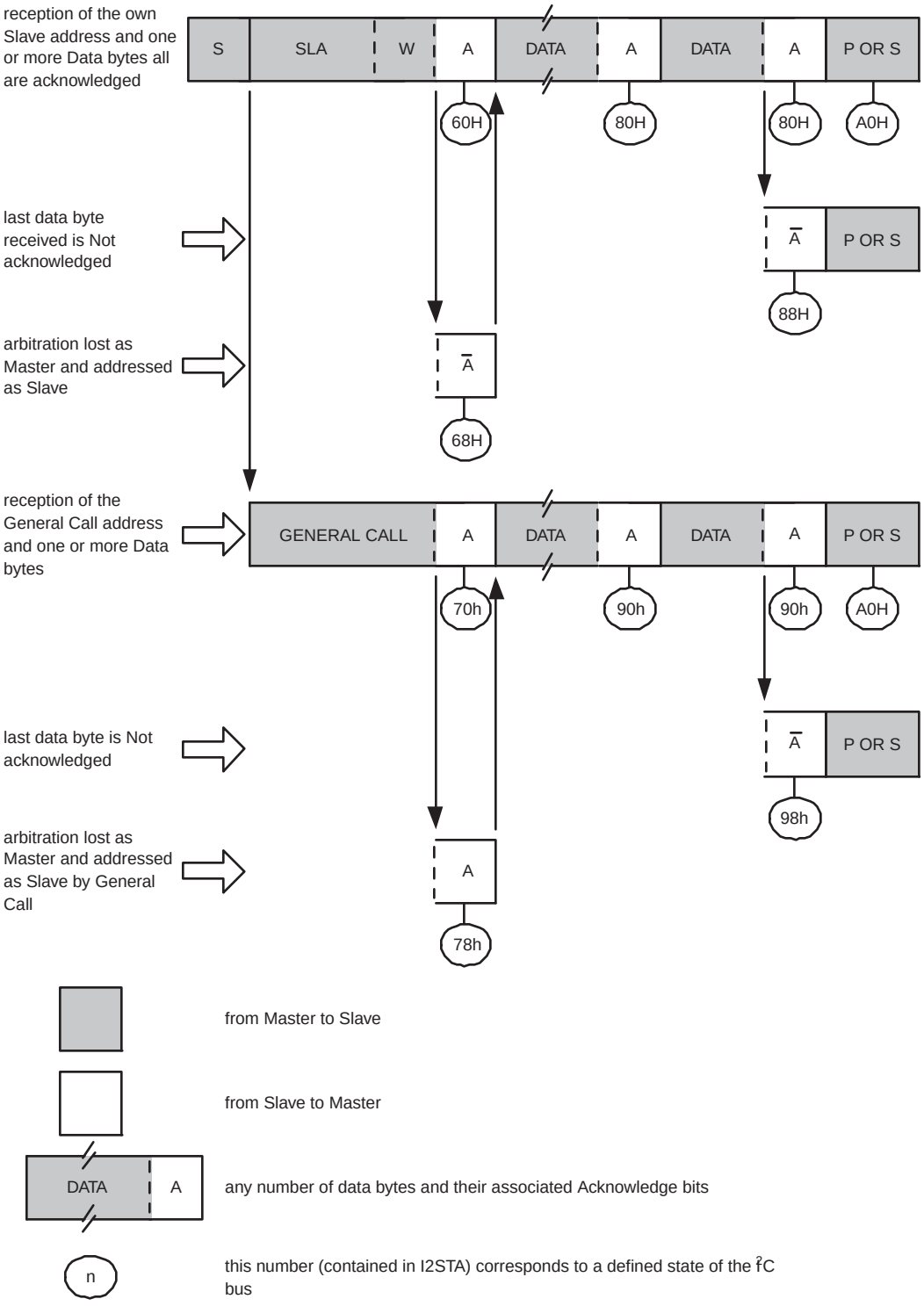


Fig 166. Format and states in the Slave Receiver mode

42.10.4 Slave Transmitter mode

In the slave transmitter mode, a number of data bytes are transmitted to a master receiver (see Figure 167). Data transfer is initialized as in the slave receiver mode. When ADR and CON have been initialized, the I^2C block waits until it is addressed by its own slave address followed by the data direction bit which must be “1” (R) for the I^2C block to operate in the slave transmitter mode. After its own slave address and the R bit have been received, the serial interrupt flag (SI) is set and a valid status code can be read from STAT. This status code is used to vector to a state service routine, and the appropriate action to be taken for each of these status codes is detailed in Table 974. The slave transmitter mode may also be entered if arbitration is lost while the I^2C block is in the master mode (see state 0xB0).

If the AA bit is reset during a transfer, the I^2C block will transmit the last byte of the transfer and enter state 0xC0 or 0xC8. The I^2C block is switched to the not addressed slave mode and will ignore the master receiver if it continues the transfer. Thus the master receiver receives all 1s as serial data. While AA is reset, the I^2C block does not respond to its own slave address or a General Call address. However, the I^2C-bus is still monitored, and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate the I^2C block from the I^2C-bus.

Table 974. Slave Transmitter mode

Status Code (STAT)	Status of the I²C-bus and hardware	Application software response					Next action taken by I²C hardware
		To/From DAT	To CON				
			STA	STO	SI	AA	
0xA8	Own SLA+R has been received; ACK has been returned.	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received.
		Load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received.
0xB0	Arbitration lost in SLA+R/W as master; Own SLA+R has been received, ACK has been returned.	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received.
		Load data byte	X	0	0	1	Data byte will be transmitted; ACK bit will be received.
0xB8	Data byte in DAT has been transmitted; ACK has been received.	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK bit will be received.
		Load data byte	X	0	0	1	Data byte will be transmitted; ACK bit will be received.
0xC0	Data byte in DAT has been transmitted; NOT ACK has been received.	No DAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address.
		No DAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1.
		No DAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		No DAT action	1	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1. A START condition will be transmitted when the bus becomes free.
0xC8	Last data byte in DAT has been transmitted (AA = 0); ACK has been received.	No DAT action or	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address.
		No DAT action or	0	0	0	1	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR[0] = logic 1.
		No DAT action or	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or General call address. A START condition will be transmitted when the bus becomes free.
		No DAT action	1	0	0	01	Switched to not addressed SLV mode; Own SLA will be recognized; General call address will be recognized if ADR.0 = logic 1. A START condition will be transmitted when the bus becomes free.

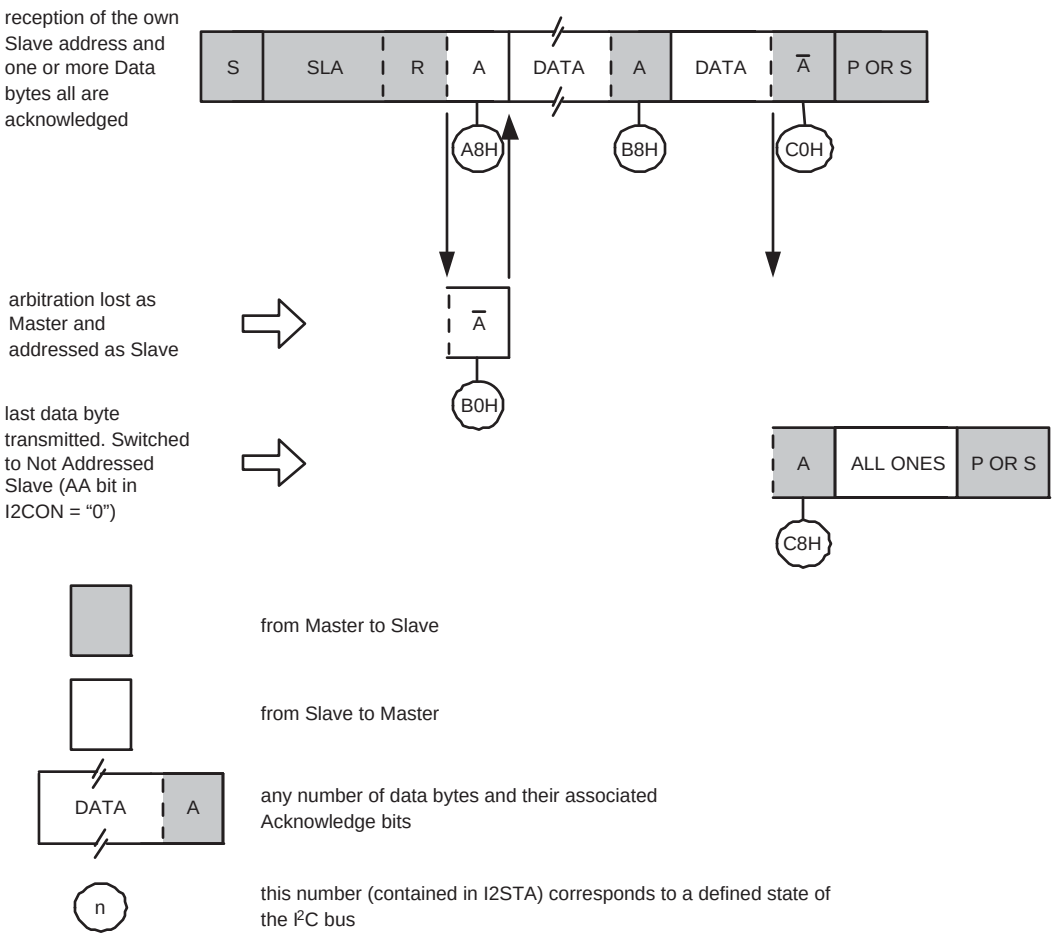


Fig 167. Format and states in the Slave Transmitter mode

42.10.5 Miscellaneous states

There are two STAT codes that do not correspond to a defined I^2C hardware state (see Table 975). These are discussed below.

42.10.5.1 STAT = 0xF8

This status code indicates that no relevant information is available because the serial interrupt flag, SI, is not yet set. This occurs between other states and when the I^2C block is not involved in a serial transfer.

42.10.5.2 STAT = 0x00

This status code indicates that a bus error has occurred during an I^2C serial transfer. A bus error is caused when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte, or an acknowledge bit. A bus error may also be caused when external interference disturbs the internal I^2C block signals. When a bus error occurs, SI is set. To recover from a bus error, the STO flag must be set and SI must be cleared. This

causes the I²C block to enter the “not addressed” slave mode (a defined state) and to clear the STO flag (no other bits in CON are affected). The SDA and SCL lines are released (a STOP condition is not transmitted).

Table 975. Miscellaneous States

Status Code (STAT)	Status of the I²C-bus and hardware	Application software response					Next action taken by I²C hardware
		To/From DAT	To CON				
			STA	STO	SI	AA	
0xF8	No relevant state information available; SI = 0.	No DAT action	No CON action				Wait or proceed current transfer.
0x00	Bus error during MST or selected slave modes, due to an illegal START or STOP condition. State 0x00 can also occur when interference causes the I²C block to enter an undefined state.	No DAT action	0	1	0	X	Only the internal hardware is affected in the MST or addressed SLV modes. In all cases, the bus is released and the I²C block is switched to the not addressed SLV mode. STO is reset.

42.10.6 Some special cases

The I²C hardware has facilities to handle the following special cases that may occur during a serial transfer:

- Simultaneous Repeated START conditions from two masters
- Data transfer after loss of arbitration
- Forced access to the I²C-bus
- I²C-bus obstructed by a LOW level on SCL or SDA
- Bus error

42.10.6.1 Simultaneous Repeated START conditions from two masters

A Repeated START condition may be generated in the master transmitter or master receiver modes. A special case occurs if another master simultaneously generates a Repeated START condition (see Figure 168). Until this occurs, arbitration is not lost by either master since they were both transmitting the same data.

If the I²C hardware detects a Repeated START condition on the I²C-bus before generating a Repeated START condition itself, it will release the bus, and no interrupt request is generated. If another master frees the bus by generating a STOP condition, the I²C block will transmit a normal START condition (state 0x08), and a retry of the total serial data transfer can commence.

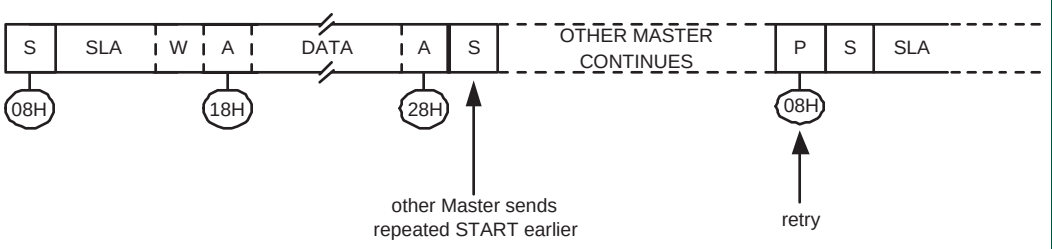


Fig 168. Simultaneous Repeated START conditions from two masters

42.10.6.2 Data transfer after loss of arbitration

Arbitration may be lost in the master transmitter and master receiver modes (see Figure 162). Loss of arbitration is indicated by the following states in STAT; 0x38, 0x68, 0x78, and 0xB0 (see Figure 164 and Figure 165).

If the STA flag in CON is set by the routines which service these states, then, if the bus is free again, a START condition (state 0x08) is transmitted without intervention by the CPU, and a retry of the total serial transfer can commence.

42.10.6.3 Forced access to the I²C-bus

In some applications, it may be possible for an uncontrolled source to cause a bus hang-up. In such situations, the problem may be caused by interference, temporary interruption of the bus or a temporary short-circuit between SDA and SCL.

If an uncontrolled source generates a superfluous START or masks a STOP condition, then the I²C-bus stays busy indefinitely. If the STA flag is set and bus access is not obtained within a reasonable amount of time, then a forced access to the I²C-bus is possible. This is achieved by setting the STO flag while the STA flag is still set. No STOP condition is transmitted. The I²C hardware behaves as if a STOP condition was received and is able to transmit a START condition. The STO flag is cleared by hardware (see Figure 169).

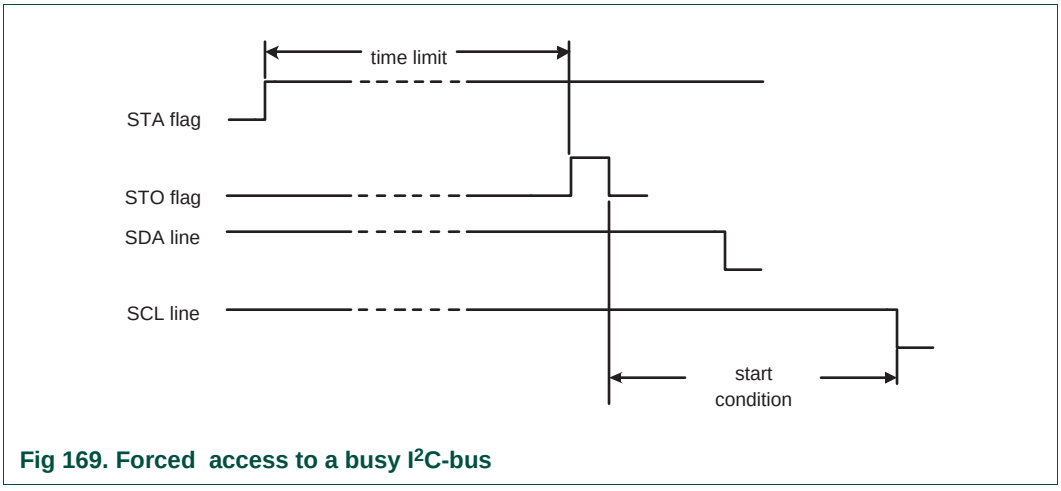
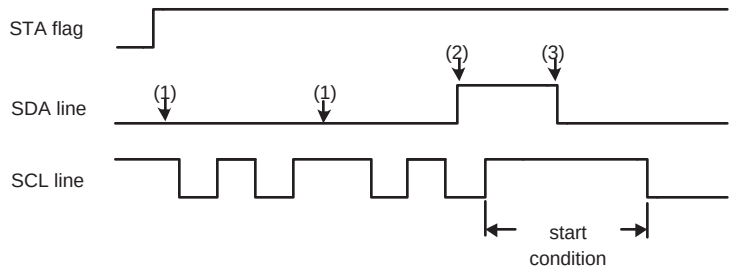


Fig 169. Forced access to a busy I²C-bus

42.10.6.4 I²C-bus obstructed by a LOW level on SCL or SDA

An I²C-bus hang-up can occur if either the SDA or SCL line is held LOW by any device on the bus. If the SCL line is obstructed (pulled LOW) by a device on the bus, no further serial transfer is possible, and the problem must be resolved by the device that is pulling the SCL bus line LOW.

Typically, the SDA line may be obstructed by another device on the bus that has become out of synchronization with the current bus master by either missing a clock, or by sensing a noise pulse as a clock. In this case, the problem can be solved by transmitting additional clock pulses on the SCL line (see Figure 170). The I²C interface does not include a dedicated time-out timer to detect an obstructed bus, but this can be implemented using another timer in the system. When detected, software can force clocks (up to 9 may be required) on SCL until SDA is released by the offending device. At that point, the slave may still be out of synchronization, so a START should be generated to insure that all I²C peripherals are synchronized.



(1) Unsuccessful attempt to send a START condition.

(2) SDA line is released.

(3) Successful attempt to send a START condition. State 08H is entered.

Fig 170. Recovering from a bus obstruction caused by a LOW level on SDA

42.10.6.5 Bus error

A bus error occurs when a START or STOP condition is detected at an illegal position in the format frame. Examples of illegal positions are during the serial transfer of an address byte, a data bit, or an acknowledge bit.

The I²C hardware only reacts to a bus error when it is involved in a serial transfer either as a master or an addressed slave. When a bus error is detected, the I²C block immediately switches to the not addressed slave mode, releases the SDA and SCL lines, sets the interrupt flag, and loads the status register with 0x00. This status code may be used to vector to a state service routine which either attempts the aborted serial transfer again or simply recovers from the error condition as shown in Table 975.

42.10.7 I²C state service routines

This section provides examples of operations that must be performed by various I²C state service routines. This includes:

- Initialization of the I²C block after a Reset.
- I²C Interrupt Service
- The 26 state service routines providing support for all four I²C operating modes.

42.10.8 Initialization

In the initialization example, the I^2C block is enabled for both master and slave modes. For each mode, a buffer is used for transmission and reception. The initialization routine performs the following functions:

- I2ADR is loaded with the part's own slave address and the General Call bit (GC)
- The I^2C interrupt enable and interrupt priority bits are set
- The slave mode is enabled by simultaneously setting the I2EN and AA bits in CON and the serial clock frequency (for master modes) is defined by is defined by loading the SCLH and SCLL registers. The master routines must be started in the main program.

The I^2C hardware now begins checking the I^2C-bus for its own slave address and General Call. If the General Call or the own slave address is detected, an interrupt is requested and STAT is loaded with the appropriate state information.

42.10.9 I^2C interrupt service

When the I^2C interrupt is entered, STAT contains a status code which identifies one of the 26 state services to be executed.

42.10.10 The state service routines

Each state routine is part of the I^2C interrupt routine and handles one of the 26 states.

42.10.11 Adapting state services to an application

The state service examples show the typical actions that must be performed in response to the 26 I^2C state codes. If one or more of the four I^2C operating modes are not used, the associated state services can be omitted, as long as care is taken that the those states can never occur.

In an application, it may be desirable to implement some kind of time-out during I^2C operations, in order to trap an inoperative bus or a lost service routine.

42.11 Software example

42.11.1 Initialization routine

Example to initialize I^2C Interface as a Slave and/or Master.

1. Load ADR with own Slave Address, enable General Call recognition if needed.
2. Enable I^2C interrupt.
3. Write 0x44 to CONSET to set the I2EN and AA bits, enabling Slave functions. For Master only functions, write 0x40 to CONSET.

42.11.2 Start Master Transmit function

Begin a Master Transmit operation by setting up the buffer, pointer, and data count, then initiating a START.

1. Initialize Master data counter.

2. Set up the Slave Address to which data will be transmitted, and add the Write bit.
3. Write 0x20 to CONSET to set the STA bit.
4. Set up data to be transmitted in Master Transmit buffer.
5. Initialize the Master data counter to match the length of the message being sent.
6. Exit

42.11.3 Start Master Receive function

Begin a Master Receive operation by setting up the buffer, pointer, and data count, then initiating a START.

1. Initialize Master data counter.
2. Set up the Slave Address to which data will be transmitted, and add the Read bit.
3. Write 0x20 to CONSET to set the STA bit.
4. Set up the Master Receive buffer.
5. Initialize the Master data counter to match the length of the message to be received.
6. Exit

42.11.4 I^2C interrupt routine

Determine the I^2C state and which state routine will be used to handle it.

1. Read the I^2C status from STA.
2. Use the status value to branch to one of 26 possible state routines.

42.11.5 Non mode specific states

42.11.5.1 State: 0x00

Bus Error. Enter not addressed Slave mode and release bus.

1. Write 0x14 to CONSET to set the STO and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.5.2 Master States

State 08 and State 10 are for both Master Transmit and Master Receive modes. The R/W bit decides whether the next state is within Master Transmit mode or Master Receive mode.

42.11.5.3 State: 0x08

A START condition has been transmitted. The Slave Address + R/W bit will be transmitted, an ACK bit will be received.

1. Write Slave Address with R/W bit to DAT.
2. Write 0x04 to CONSET to set the AA bit.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Set up Master Transmit mode data buffer.

5. Set up Master Receive mode data buffer.
6. Initialize Master data counter.
7. Exit

42.11.5.4 State: 0x10

A Repeated START condition has been transmitted. The Slave Address + R/W bit will be transmitted, an ACK bit will be received.

1. Write Slave Address with R/W bit to DAT.
2. Write 0x04 to CONSET to set the AA bit.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Set up Master Transmit mode data buffer.
5. Set up Master Receive mode data buffer.
6. Initialize Master data counter.
7. Exit

42.11.6 Master Transmitter states

42.11.6.1 State: 0x18

Previous state was State 8 or State 10, Slave Address + Write has been transmitted, ACK has been received. The first data byte will be transmitted, an ACK bit will be received.

1. Load DAT with first data byte from Master Transmit buffer.
2. Write 0x04 to CONSET to set the AA bit.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Increment Master Transmit buffer pointer.
5. Exit

42.11.6.2 State: 0x20

Slave Address + Write has been transmitted, NOT ACK has been received. A STOP condition will be transmitted.

1. Write 0x14 to CONSET to set the STO and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.6.3 State: 0x28

Data has been transmitted, ACK has been received. If the transmitted data was the last data byte then transmit a STOP condition, otherwise transmit the next data byte.

1. Decrement the Master data counter, skip to step 5 if not the last data byte.
2. Write 0x14 to CONSET to set the STO and AA bits.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Exit
5. Load DAT with next data byte from Master Transmit buffer.

6. Write 0x04 to CONSET to set the AA bit.
7. Write 0x08 to CONCLR to clear the SI flag.
8. Increment Master Transmit buffer pointer
9. Exit

42.11.6.4 State: 0x30

Data has been transmitted, NOT ACK received. A STOP condition will be transmitted.

1. Write 0x14 to CONSET to set the STO and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.6.5 State: 0x38

Arbitration has been lost during Slave Address + Write or data. The bus has been released and not addressed Slave mode is entered. A new START condition will be transmitted when the bus is free again.

1. Write 0x24 to CONSET to set the STA and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.7 Master Receive states

42.11.7.1 State: 0x40

Previous state was State 08 or State 10. Slave Address + Read has been transmitted, ACK has been received. Data will be received and ACK returned.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.7.2 State: 0x48

Slave Address + Read has been transmitted, NOT ACK has been received. A STOP condition will be transmitted.

1. Write 0x14 to CONSET to set the STO and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.7.3 State: 0x50

Data has been received, ACK has been returned. Data will be read from DAT. Additional data will be received. If this is the last data byte then NOT ACK will be returned, otherwise ACK will be returned.

1. Read data byte from DAT into Master Receive buffer.
2. Decrement the Master data counter, skip to step 5 if not the last data byte.
3. Write 0x0C to CONCLR to clear the SI flag and the AA bit.

4. Exit
5. Write 0x04 to CONSET to set the AA bit.
6. Write 0x08 to CONCLR to clear the SI flag.
7. Increment Master Receive buffer pointer
8. Exit

42.11.7.4 State: 0x58

Data has been received, NOT ACK has been returned. Data will be read from DAT. A STOP condition will be transmitted.

1. Read data byte from DAT into Master Receive buffer.
2. Write 0x14 to CONSET to set the STO and AA bits.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Exit

42.11.8 Slave Receiver states

42.11.8.1 State: 0x60

Own Slave Address + Write has been received, ACK has been returned. Data will be received and ACK returned.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Set up Slave Receive mode data buffer.
4. Initialize Slave data counter.
5. Exit

42.11.8.2 State: 0x68

Arbitration has been lost in Slave Address and R/W bit as bus Master. Own Slave Address + Write has been received, ACK has been returned. Data will be received and ACK will be returned. STA is set to restart Master mode after the bus is free again.

1. Write 0x24 to CONSET to set the STA and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Set up Slave Receive mode data buffer.
4. Initialize Slave data counter.
5. Exit.

42.11.8.3 State: 0x70

General call has been received, ACK has been returned. Data will be received and ACK returned.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Set up Slave Receive mode data buffer.

4. Initialize Slave data counter.
5. Exit

42.11.8.4 State: 0x78

Arbitration has been lost in Slave Address + R/W bit as bus Master. General call has been received and ACK has been returned. Data will be received and ACK returned. STA is set to restart Master mode after the bus is free again.

1. Write 0x24 to CONSET to set the STA and AA bits.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Set up Slave Receive mode data buffer.
4. Initialize Slave data counter.
5. Exit

42.11.8.5 State: 0x80

Previously addressed with own Slave Address. Data has been received and ACK has been returned. Additional data will be read.

1. Read data byte from DAT into the Slave Receive buffer.
2. Decrement the Slave data counter, skip to step 5 if not the last data byte.
3. Write 0x0C to CONCLR to clear the SI flag and the AA bit.
4. Exit.
5. Write 0x04 to CONSET to set the AA bit.
6. Write 0x08 to CONCLR to clear the SI flag.
7. Increment Slave Receive buffer pointer.
8. Exit

42.11.8.6 State: 0x88

Previously addressed with own Slave Address. Data has been received and NOT ACK has been returned. Received data will not be saved. Not addressed Slave mode is entered.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.8.7 State: 0x90

Previously addressed with General Call. Data has been received, ACK has been returned. Received data will be saved. Only the first data byte will be received with ACK. Additional data will be received with NOT ACK.

1. Read data byte from DAT into the Slave Receive buffer.
2. Write 0x0C to CONCLR to clear the SI flag and the AA bit.
3. Exit

42.11.8.8 State: 0x98

Previously addressed with General Call. Data has been received, NOT ACK has been returned. Received data will not be saved. Not addressed Slave mode is entered.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.8.9 State: 0xA0

A STOP condition or Repeated START has been received, while still addressed as a Slave. Data will not be saved. Not addressed Slave mode is entered.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

42.11.9 Slave Transmitter states

42.11.9.1 State: 0xA8

Own Slave Address + Read has been received, ACK has been returned. Data will be transmitted, ACK bit will be received.

1. Load DAT from Slave Transmit buffer with first data byte.
2. Write 0x04 to CONSET to set the AA bit.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Set up Slave Transmit mode data buffer.
5. Increment Slave Transmit buffer pointer.
6. Exit

42.11.9.2 State: 0xB0

Arbitration lost in Slave Address and R/W bit as bus Master. Own Slave Address + Read has been received, ACK has been returned. Data will be transmitted, ACK bit will be received. STA is set to restart Master mode after the bus is free again.

1. Load DAT from Slave Transmit buffer with first data byte.
2. Write 0x24 to CONSET to set the STA and AA bits.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Set up Slave Transmit mode data buffer.
5. Increment Slave Transmit buffer pointer.
6. Exit

42.11.9.3 State: 0xB8

Data has been transmitted, ACK has been received. Data will be transmitted, ACK bit will be received.

1. Load DAT from Slave Transmit buffer with data byte.

2. Write 0x04 to CONSET to set the AA bit.
3. Write 0x08 to CONCLR to clear the SI flag.
4. Increment Slave Transmit buffer pointer.
5. Exit

42.11.9.4 State: 0xC0

Data has been transmitted, NOT ACK has been received. Not addressed Slave mode is entered.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit.

42.11.9.5 State: 0xC8

The last data byte has been transmitted, ACK has been received. Not addressed Slave mode is entered.

1. Write 0x04 to CONSET to set the AA bit.
2. Write 0x08 to CONCLR to clear the SI flag.
3. Exit

43.1 How to read this chapter

ADC0 and ADC1 are available on all LPC43xx parts. The number of ADC channels is limited on small packages.

Table 976. ADC channels for different packages

Analog function	Pin	LBGA256	TFBGA180	TFBGA100	LQFP208	LQFP144	LQFP100
ADC0_0	P4_3	yes	yes	-	yes	yes	-
ADC0_1	P4_1	yes	yes	-	yes	yes	-
ADC0_2	PF_8	yes	-	-	-	-	-
ADC0_3	P7_5	yes	yes	-	yes	yes	-
ADC0_4	P7_4	yes	yes	-	yes	yes	-
ADC0_5	PF_10	yes	-	-	yes	-	yes
ADC0_6	PB_6	yes	yes	-	-	-	-
ADC1_0	PC_3	yes	-	-	yes	-	-
ADC1_1	PC_0	yes	yes	-	yes	-	-
ADC1_2	PF_9	yes	-	-	yes	-	-
ADC1_3	PF_6	yes	-	-	yes	-	-
ADC1_4	PF_5	yes	-	-	yes	-	-
ADC1_5	PF_11	yes	-	-	yes	-	yes
ADC1_6	P7_7	yes	yes	-	yes	yes	-
ADC1_7	PF_7	yes	-	-	yes	-	-
DAC	P4_4	yes	yes	-	yes	yes	-
ADC input channel 0. Shared between 10-bit ADC0/1 and DAC.	ADC0_0/ ADC1_0/DAC	yes	yes	yes	yes	yes	yes
ADC input channel 1. Shared between 10-bit ADC0/1.	ADC0_1/ ADC1_1	yes	yes	yes	yes	yes	yes
ADC input channel 2. Shared between 10-bit ADC0/1.	ADC0_2/ ADC1_2	yes	yes	yes	yes	yes	yes
ADC input channel 3. Shared between 10-bit ADC0/1.	ADC0_3/ ADC1_3	yes	yes	yes	yes	yes	yes
ADC input channel 4. Shared between 10-bit ADC0/1.	ADC0_4/ ADC1_4	yes	yes	-	yes	yes	-
ADC input channel 5. Shared between 10-bit ADC0/1.	ADC0_5/ ADC1_5	yes	yes	-	yes	yes	-
ADC input channel 6. Shared between 10-bit ADC0/1.	ADC0_6/ ADC1_6	yes	yes	-	yes	yes	-
ADC input channel 7. Shared between 10-bit ADC0/1.	ADC0_7/ ADC1_7	yes	yes	-	yes	yes	-

43.2 Basic configuration

The ADC0 and ADC1 are configured as follows:

- See Table 977 for clocking and power control.
- The ADC0 is reset by the ADC0_RST (reset # 40).
- The ADC1 is reset by the ADC1_RST (reset # 41).
- The ADC0 interrupt is connected to interrupt slot # 17 in the NVIC.
- The ADC1 interrupt is connected to interrupt slot # 21 in the NVIC.
- For connecting to the GPDMA, use the DMAMUX register (Table 41) in the CREG block and enable the GPDMA channel in the DMA Channel Configuration registers Section 19.6.20.
- External pins (ADCTRIG0/1)and the MOTOCON PWM MCOA2 output can be selected as conversion triggers for ADC0/1 (see Table 981).
- The ADC conversion triggers are connected through the GIMA (see Table 138) to the timers or SCT outputs.
- For the ADC0 and ADC1 inputs that are multiplexed with digital functions, the pins need to be configured using the ENAIO0/1 registers (see Section 15.4.6 and Section 15.4.8.
- The ADC1 channel 7 is connected to the bandgap reference (see Section 15.4.8.1).

Table 977. ADC0/1 clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
ADC0 clock	BASE_APB3_CLK	CLK_APB3_ADC0	up to 204 MHz	For register interface and ADC0 conversion rate.
ADC1 clock	BASE_APB3_CLK	CLK_APB3_ADC1	up to 204 MHz	For register interface and ADC1 conversion rate.

43.3 Features

- 10 bit successive approximation analog to digital converter.
- Input multiplexing among 8 pins.
- Power-down mode.
- Measurement range 0 to 3.3 V.
- 10-bit conversion time = 2.45 μs.
- Burst conversion mode for single or multiple inputs.
- Optional conversion on transition on input pin or Timer Match signal.
- Individual result registers for each A/D channel to reduce interrupt overhead.\
- Connected to bandgap reference (see Section 15.4.8.1).

43.4 General description

Basic clocking for the A/D converters is provided by the APB clocks (CLK_APB4_ADC0/1). A programmable divider is included in each converter to scale this clock to the 4.5 MHz (max) clock needed by the successive approximation process. A fully accurate conversion requires 11 of these clocks.

43.5 Pin description

Table 978 gives a brief summary of each of ADC related pins.

Table 978. ADC pin description

Pin function	Type	Description
ADC0_[7:0]/ ADC1_[7:0]	Input	Shared Analog Inputs. The A/D converter cell can measure the voltage on any of these input signals. The inputs are shared between ADC0 and ADC1 on analog-only pins **Remark:** The ADC0 pin is shared with the DAC0 pin.
ADC0_[6:0]	Input	Analog inputs.Inputs from multiplexed analog/digital pins to ADC0. The A/D converter cell can measure the voltage on any of these input signals. These pins are not shared with ADC1.
ADC1_[7:0]	Input	Analog inputs.Inputs from multiplexed analog/digital pins to ADC1. The A/D converter cell can measure the voltage on any of these input signals. These pins are not shared with ADC0.
ADCTRIG0	Input	Trigger inputs to the ADC0/1.
ADCTRIG1	Input	Trigger inputs to the ADC0/1.
VDDA	Power	Analog Power. Also voltage reference VREF for both ADCs.
VSSA	Ground	Analog ground.

43.6 Register description

The register addresses for the ADC0 are shown in Table 979

Table 979. Register overview: ADC0 (base address 0x400E 3000)

Name	Access	Address offset	Description	Reset value[1]	Reference
CR	R/W	0x000	A/D Control Register. The AD0CR register must be written to select the operating mode before A/D conversion can occur.	0x0000 0000	Table 981
GDR	R0	0x004	A/D Global Data Register. Contains the result of the most recent A/D conversion.	-	Table 982
-	-	0x008	Reserved.	-	
INTEN	R/W	0x00C	A/D Interrupt Enable Register. This register contains enable bits that allow the DONE flag of each A/D channel to be included or excluded from contributing to the generation of an A/D interrupt.	0x0000 0100	Table 983
DR0	RO	0x010	A/D Channel 0 Data Register. This register contains the result of the most recent conversion completed on channel 0	-	Table 984

Table 979. Register overview: ADC0 (base address 0x400E 3000)

Name	Access	Address offset	Description	Reset value[1]	Reference
DR1	RO	0x014	A/D Channel 1 Data Register. This register contains the result of the most recent conversion completed on channel 1.	-	Table 984
DR2	RO	0x018	A/D Channel 2 Data Register. This register contains the result of the most recent conversion completed on channel 2.	-	Table 984
DR3	RO	0x01C	A/D Channel 3 Data Register. This register contains the result of the most recent conversion completed on channel 3.	-	Table 984
DR4	RO	0x020	A/D Channel 4 Data Register. This register contains the result of the most recent conversion completed on channel 4.	-	Table 984
DR5	RO	0x024	A/D Channel 5 Data Register. This register contains the result of the most recent conversion completed on channel 5.	-	Table 984
DR6	RO	0x028	A/D Channel 6 Data Register. This register contains the result of the most recent conversion completed on channel 6.	-	Table 984
DR7	RO	0x02C	A/D Channel 7 Data Register. This register contains the result of the most recent conversion completed on channel 7.	-	Table 984
STAT	RO	0x030	A/D Status Register. This register contains DONE and OVERRUN flags for all of the A/D channels, as well as the A/D interrupt flag.	0	Table 985

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

Table 980. Register overview: ADC1 (base address 0x400E 4000)

Name	Access	Address offset	Description	Reset value[1]	Reference
CR	R/W	0x000	A/D Control Register. The AD1CR register must be written to select the operating mode before A/D conversion can occur.	0x0000 0000	Table 981
GDR	R0	0x004	A/D Global Data Register. Contains the result of the most recent A/D conversion.	-	Table 982
-	-	0x008	Reserved.	-	
INTEN	R/W	0x00C	A/D Interrupt Enable Register. This register contains enable bits that allow the DONE flag of each A/D channel to be included or excluded from contributing to the generation of an A/D interrupt.	0x0000 0100	Table 983
DR0	RO	0x010	A/D Channel 0 Data Register. This register contains the result of the most recent conversion completed on channel 0	-	Table 984
DR1	RO	0x014	A/D Channel 1 Data Register. This register contains the result of the most recent conversion completed on channel 1.	-	Table 984
DR2	RO	0x018	A/D Channel 2 Data Register. This register contains the result of the most recent conversion completed on channel 2.	-	Table 984
DR3	RO	0x01C	A/D Channel 3 Data Register. This register contains the result of the most recent conversion completed on channel 3.	-	Table 984
DR4	RO	0x020	A/D Channel 4 Data Register. This register contains the result of the most recent conversion completed on channel 4.	-	Table 984

Table 980. Register overview: ADC1 (base address 0x400E 4000)

Name	Access	Address offset	Description	Reset value[1]	Reference
DR5	RO	0x024	A/D Channel 5 Data Register. This register contains the result of the most recent conversion completed on channel 5.	-	Table 984
DR6	RO	0x028	A/D Channel 6 Data Register. This register contains the result of the most recent conversion completed on channel 6.	-	Table 984
DR7	RO	0x02C	A/D Channel 7 Data Register. This register contains the result of the most recent conversion completed on channel 7.	-	Table 984
STAT	RO	0x030	A/D Status Register. This register contains DONE and OVERRUN flags for all of the A/D channels, as well as the A/D interrupt flag.	0	Table 985

[1] Reset value reflects the data stored in used bits only. It does not include reserved bits content.

43.6.1 A/D Control register

The A/D Control Register provides bits to select A/D channels to be converted, A/D timing, A/D modes, and the A/D start trigger.

Table 981. A/D Control register (CR - address 0x400E 3000 (ADC0) and 0x400E 4000 (ADC1)) bit description

Bit	Symbol	Value	Description	Reset value
7:0	SEL		Selects which of the ADC[7:0] pins are to be sampled and converted. Bit 0 selects Pin ADC0, bit 1 selects pin AD1,..., and bit 7 selects pin ADC7. In software-controlled mode, only one of these bits should be 1. In hardware scan mode, any value containing 1 to 8 ones. All zeroes is equivalent to 0x01.	0
15:8	CLKDIV		The ADC clock is divided by the CLKDIV value plus one to produce the clock for the A/D converter, which should be less than or equal to 4.5 MHz. Typically, software should program the smallest value in this field that yields a clock of 4.5 MHz or slightly less, but in certain cases (such as a high-impedance analog source) a slower clock may be desirable.	0
16	BURST		Controls Burst mode	0
		0	Conversions are software controlled and require 11 clocks.	
		1	The AD converter does repeated conversions at the rate selected by the CLKS field, scanning (if necessary) through the pins selected by 1s in the SEL field. The first conversion after the start corresponds to the least-significant 1 in the SEL field, then higher numbered 1 bits (pins) if applicable. Repeated conversions can be terminated by clearing this bit, but the conversion that's in progress when this bit is cleared will be completed. **Important:** START bits must be 000 when BURST = 1 or conversions will not start.	

Table 981. A/D Control register (CR - address 0x400E 3000 (ADC0) and 0x400E 4000 (ADC1)) bit description

Bit	Symbol	Value	Description	Reset value
19:17	CLKS		This field selects the number of clocks used for each conversion in Burst mode and the number of bits of accuracy of the result in the LS bits of ADDR, between 11 clocks (10 bits) and 4 clocks (3 bits).	000
		0x0	11 clocks / 10 bits	
		0x1	10 clocks / 9 bits	
		0x2	9 clocks / 8 bits	
		0x3	8 clocks / 7 bits	
		0x4	7 clocks / 6 bits	
		0x5	6 clocks / 5 bits	
		0x6	5 clocks / 4 bits	
		0x7	4 clocks / 3 bits	
20	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
21	PDN		Power mode	0
		0	The A/D converter is in Power-down mode.	
		1	The A/D converter is operational.	
23:22	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
26:24	START		Controls the start of an A/D conversion when the BURST bit is 0.	0
		0x0	No start (this value should be used when clearing PDN to 0).	
		0x1	Start now.	
		0x2	Start conversion when the edge selected by bit 27 occurs on CTOUT_15 (combined timer output 15, ADC start0).	
		0x3	Start conversion when the edge selected by bit 27 occurs on CTOUT_8 (combined timer output 8, ADC start1).	
		0x4	Start conversion when the edge selected by bit 27 occurs on ADCTRIG0 input (ADC start3).	
		0x5	Start conversion when the edge selected by bit 27 occurs on ADCTRIG1 input (ADC start4).	
		0x6	Start conversion when the edge selected by bit 27 occurs on Motocon PWM output MCOA2 (ADC start5).	
		0x7	Reserved.	
27	EDGE		Controls rising or falling edge on the selected signal for the start of a conversion. This bit is significant only when the START field contains 0x2 -0x6).	0
		0	Rising edge.	
		1	Falling edge.	
31:28	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

43.6.2 A/D Global Data register

The A/D Global Data Register contains the result of the most recent A/D conversion. This includes the data, DONE, and Overrun flags, and the number of the A/D channel to which the data relates.

Table 982. A/D Global Data register (GDR - address 0x400E 3004 (ADC0) and 0x400E 4004 (ADC1)) bit description

Bit	Symbol	Description	Reset value
5:0	-	Reserved. These bits always read as zeroes.	0
15:6	V_VREF	When DONE is 1, this field contains a binary fraction representing the voltage on the ADCn pin selected by the SEL field, divided by the reference voltage on the VDDA pin. Zero in the field indicates that the voltage on the ADCn input pin was less than, equal to, or close to that on VSSA, while 0x3FF indicates that the voltage on ADCn input pin was close to, equal to, or greater than that on VDDA.	-
23:16	-	Reserved. These bits always read as zeroes.	0
26:24	CHN	These bits contain the channel from which the LS bits were converted.	-
29:27	-	Reserved. These bits always read as zeroes.	0
30	OVERRUN	This bit is 1 in burst mode if the results of one or more conversions was (were) lost and overwritten before the conversion that produced the result in the V_VREF bits.	0
31	DONE	This bit is set to 1 when an analog-to-digital conversion completes. It is cleared when this register is read and when the AD0/1CR register is written. If the AD0/1CR is written while a conversion is still in progress, this bit is set and a new conversion is started.	0

43.6.3 A/D Interrupt Enable register

This register allows control over which A/D channels generate an interrupt when a conversion is complete. For example, it may be desirable to use some A/D channels to monitor sensors by continuously performing conversions on them. The most recent results are read by the application program whenever they are needed. In this case, an interrupt is not desirable at the end of each conversion for some A/D channels.

Table 983. A/D Interrupt Enable register (INTEN - address 0x400E 300C (ADC0) and 0x400E 400C (ADC1)) bit description

Bit	Symbol	Description	Reset value
7:0	ADINTEN	These bits allow control over which A/D channels generate interrupts for conversion completion. When bit 0 is one, completion of a conversion on A/D channel 0 will generate an interrupt, when bit 1 is one, completion of a conversion on A/D channel 1 will generate an interrupt, etc.	0x00
8	ADGINTEN	When 1, enables the global DONE flag in ADDR to generate an interrupt. When 0, only the individual A/D channels enabled by ADINTEN 7:0 will generate interrupts.	1
31:9	-	Reserved. Always 0.	0

43.6.4 A/D Data Registers

The A/D Data Register hold the result when an A/D conversion is complete, and also include the flags that indicate when a conversion has been completed and when a conversion overrun has occurred.

Table 984. A/D Data registers (DR - addresses 0x400E 3010 (DR0) to 0x400E 302C (DR7) (ADC0); 0x400E 4010 (DR0) to 0x400E 402C (DR7) (ADC1)) bit description

Bit	Symbol	Description	Reset value
5:0	-	Reserved. Always 0.	0
15:6	V_VREF	When DONE is 1, this field contains a binary fraction representing the voltage on the ADCn input pin selected in Table 981, divided by the voltage on the VDDA pin. Zero in the field indicates that the voltage on the ADCn input pin was less than, equal to, or close to that on VDDA, while 0x3FF indicates that the voltage on ADCn input pin was close to, equal to, or greater than that on VDDA.	-
29:16	-	Reserved. Always 0.	0
30	OVERRUN	This bit is 1 in burst mode if the results of one or more conversions was (were) lost and overwritten before the conversion that produced the result in the V_VREF bits in this register.This bit is cleared by reading this register.	0
31	DONE	This bit is set to 1 when an A/D conversion completes. It is cleared when this register is read.	0

43.6.5 A/D Status register

The A/D Status register allows checking the status of all A/D channels simultaneously. The DONE and OVERRUN flags appearing in the AD0/1DRn register for each A/D channel n are mirrored in ADSTAT. The interrupt flag (the logical OR of all DONE flags) is also found in ADSTAT.

Table 985. A/D Status register (STAT - address 0x400E 3030 (ADC0) and 0x400E 4030 (ADC1)) bit description

Bit	Symbol	Description	Reset value
7:0	DONE	These bits mirror the DONE status flags that appear in the result register for each A/D channel.	0
15:8	OVERUN	These bits mirror the OVERRRUN status flags that appear in the result register for each A/D channel. Reading ADSTAT allows checking the status of all A/D channels simultaneously.	0
16	ADINT	This bit is the A/D interrupt flag. It is one when any of the individual A/D channel Done flags is asserted and enabled to contribute to the A/D interrupt via the ADINTEN register.	0
31:17	-	Reserved. Always 0.	0

43.7 Operation

43.7.1 Hardware-triggered conversion

If the BURST bit in the ADCR is 0 and the START field contains any value between 0x2 and 0x6, the A/D converter will start a conversion when a transition occurs on a selected pin or Timer signal. The choices include the two ADCTRIG external input pins, an output from the motocon PWM, and two combined timer outputs (see Section 43.6.1).

43.7.2 Interrupts

An interrupt is requested to the NVIC when the ADINT bit in the ADSTAT register is 1. The ADINT bit is one when any of the DONE bits of the A/D channels which are enabled for interrupts (via the ADINTEN register) are one. Software can use the Interrupt Enable bit in the NVIC that corresponds to the ADC to control whether this results in an interrupt. The result register of the A/D channel which is generating an interrupt must be read in order to clear the corresponding DONE flag.

43.7.3 DMA control

A DMA transfer request is generated from the ADC interrupt request line. To generate a DMA transfer the same conditions must be met as the conditions for generating an interrupt. A pending DMA request is cleared after the DMA has read from the requesting channel's A/D data register (DR[7:0]). Reading from the global data register (GDR) does not clear any pending DMA requests.

For DMA transfers, only burst requests are supported. The burst size can be set to one of the predefined burstsizes in the DMA channel control register (see Table 279). If the number of ADC channels is not equal to one of the predefined DMA-supported burst sizes (applicable DMA burst sizes are 1, 4, 8), set the burst size to one.

The DMA transfer size determines when a DMA interrupt is generated. The transfer size can be set to the number of ADC channels being converted (see Section 19.6.19). Non-contiguous channels can be transferred by the DMA using the scatter/gather linked lists (see Section 19.8.5).

44.1 How to read this chapter

The DAC is available on all LPC43xx parts.

44.2 Basic configuration

The DAC is configured as follows:

- The DMA_ENA bit in the DAC converter register (Table 989) must be enabled to obtain a valid DAC output.
- See Table 986 for clocking and power control.
- The DAC is reset by the DAC_RST (reset # 42).
- The DAC interrupt is connected to interrupt slot # 0 in the NVIC.
- For connecting to the GPDMA, use the DMAMUX register (Table 41) in the CREG block and enable the GPDMA channel in the DMA Channel Configuration registers Section 19.6.20.

Table 986. DAC clocking and power control

	Base clock	Branch clock	Operating frequency	Notes
Clock to the DAC register interface and rate clock for the DMA counter.	BASE_APB3_CLK	CLK_APB3_DAC	up to 204 MHz	-

44.3 Features

- 10-bit resolution
- Monotonic by design (resistor string architecture)
- Controllable conversion speed
- Can be optimized for speed and power
- Low power consumption
- Maximum update rate of 1 MHz
- DMA support

44.4 Pin description

Table 987 gives a brief summary of each of DAC related pins.

Table 987. DAC pin description

Pin	Type	Description
DAC	Output	Analog Output. After the selected settling time after the DACR is written with a new value, the voltage on this pin (with respect to V_{SSA}) is VALUE/1024 × VDDA. The DAC pin is shared with the channel 0 input pin of ADC0 and ADC1.
VDDA	Power	Analog power and voltage reference. This pin provides a voltage reference level VREF for the D/A converter.
VSSA	-	Ground.

44.5 Register description

Table 988. Register overview: DAC (base address 0x400E 1000)

Name	Access	Address offset	Description	Reset value	Reference
CR	R/W	0x000	DAC register. Holds the conversion data.	0	Table 989
CTRL	R/W	0x004	DAC control register.	0	Table 990
CNTVAL	R/W	0x008	DAC counter value register.	0	Table 991

44.5.1 D/A converter register

This read/write register includes the digital value to be converted to an analog output value and a bit that trades off performance vs. power.

Table 989: D/A Converter register (CR - address 0x400E 1000) bit description

Bit	Symbol	Value	Description	Reset value
5:0	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-
15:6	VALUE		After the selected settling time after this field is written with a new VALUE, the voltage on the DAC pin (with respect to V_{SSA}) is VALUE/1024 × VDDA.	0
16	BIAS		Settling time	0
		0	Shorter settling times and higher power consumption; allows for a maximum update rate of 1 MHz.	
		1	Longer settling times and lower power consumption; allows for a maximum update rate of 400 kHz.	
31:17	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

44.5.2 D/A Converter Control register

This read/write register enables the DMA operation and controls the DMA timer.

Table 990. D/A Control register (CTRL - address 0x400E 1004) bit description

Bit	Symbol	Value	Description	Reset value
0	INT_DMA_REQ		DMA request	0
		0	This bit is cleared on any write to the DACR register.	
		1	This bit is set by hardware when the timer times out.	
1	DBLBUF_ENA		DMA double-buffering	0
		0	DACR double-buffering is disabled.	
		1	When this bit and the CNT_ENA bit are both set, the double-buffering feature in the DACR register will be enabled. Writes to the DACR register are written to a pre-buffer and then transferred to the DACR on the next time-out of the counter.	
2	CNT_ENA		DMA time-out	0
		0	Time-out counter operation is disabled.	
		1	Time-out counter operation is enabled.	
3	DMA_ENA		DMA enable	0
		0	DMA access is disabled.	
		1	DMA Burst Request Input 15 is enabled for the DAC (see Table 260).	
31:4	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	-

44.5.3 D/A Converter Counter Value register

This read/write register contains the reload value for the Interrupt/DMA counter.

Table 991: D/A Converter counter value register (CNTVAL - address 0x400E 1008) bit description

Bit	Symbol	Description	Reset value
15:0	VALUE	16-bit reload value for the DAC interrupt/DMA timer.	0
31:16	-	Reserved.	-

44.6 Functional description

44.6.1 DMA counter

When the counter enable bit CNT_ENA in DACCTRL is set, a 16-bit counter will begin counting down, at the rate selected by CLK_APB3_DAC, from the value programmed into the DACCNTVAL register. The counter is decremented each time the counter reaches zero, the counter will be reloaded by the value of DACCNTVAL and the DMA request bit INT_DMA_REQ will be set in hardware.

Note that the contents of the DACCTRL and DACCNTVAL registers are read and write accessible, but the timer itself is not accessible for either read or write.

If the DMA_ENA bit is set in the DACCTRL register, the DAC DMA request will be routed to the GPDMA. When the DMA_ENA bit is cleared, the default state after a reset, DAC DMA requests are blocked.

44.6.2 Double buffering

Double-buffering is enabled only if both, the CNT_ENA and the DBLBUF_ENA bits are set in DACCTRL. In this case, any write to the DACR register will only load the pre-buffer, which shares its register address with the DACR register. The DACR itself will be loaded from the pre-buffer whenever the counter reaches zero and the DMA request is set. At the same time the counter is reloaded with the COUNTVAL register value.

Reading the DACR register will only return the contents of the DACR register itself, not the contents of the pre-buffer register.

If either the CNT_ENA or the DBLBUF_ENA bits are 0, any writes to the DACR address will go directly to the DACR register.

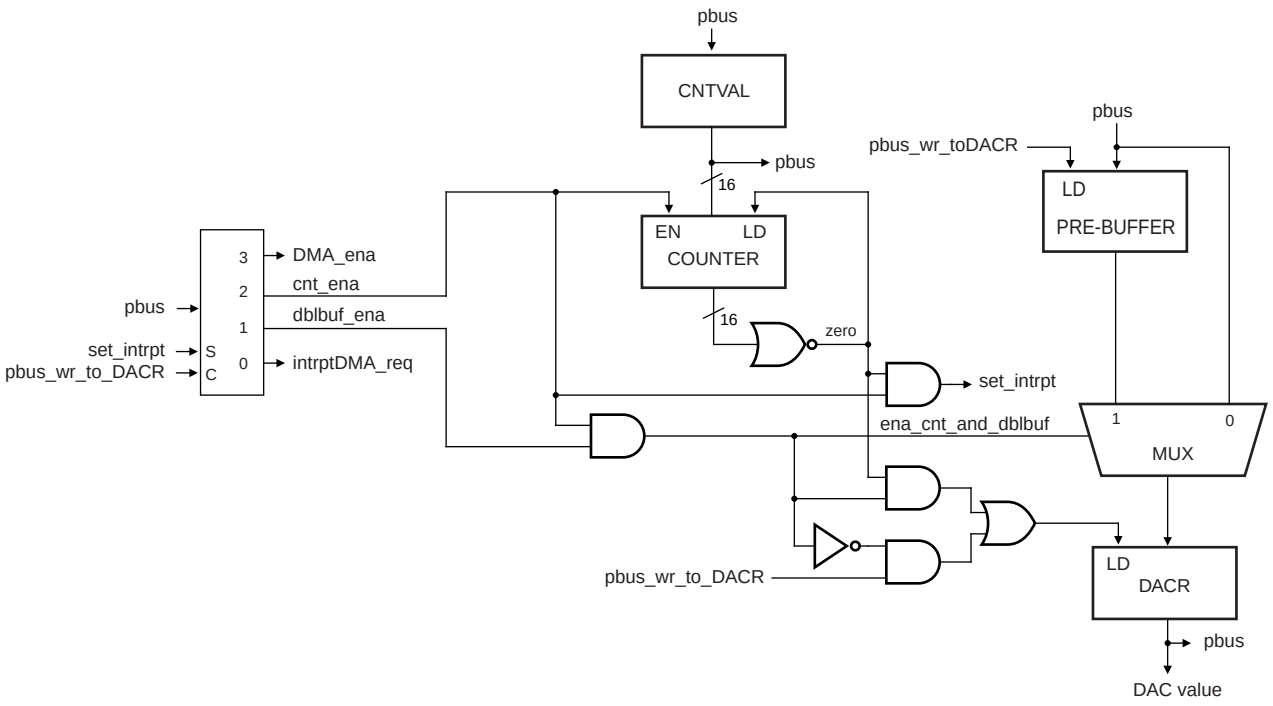


Fig 171. DAC control with DMA interrupt and timer

45.1 How to read this chapter

The flash programming API is available for parts with on-chip flash. A reduced set of In-System-Programming (ISP) commands is supported for flashless parts (see Table 995). See Chapter 5 for details on the boot process for flashless parts.

45.2 Introduction

The boot loader controls initial operation after reset and also provides the tools for programming the flash memory. This could be initial programming of a blank device, erasure and re-programming of a previously programmed device, or programming of the flash memory by the application program in a running system.

45.3 Features

- In-System Programming: In-System programming (ISP) is programming or reprogramming the on-chip flash memory, using the boot loader software and UART0 serial port. This can be done when the part resides in the end-user board.
- For parts without on-chip flash, ISP allows to load data to on-chip SRAM and to execute code from on-chip SRAM.
- In Application Programming: In-Application (IAP) programming is performing erase and write operation on the on-chip flash memory, as directed by the end-user application code.
- Flash signature generation: built-in hardware can generate a signature for a range of flash addresses or for the entire flash memory.

45.4 Description

The flash boot loader code is executed every time the part is powered on or reset. The loader can execute the ISP command handler or the user application code. A LOW level after reset at pin P2_7 is considered an external hardware request to start the ISP command handler. Assuming that power supply pins are on their nominal levels when the rising edge on $\overline{\text{RESET}}$ pin is generated, it may take up to 3 ms before P2_7 is sampled and the decision on whether to continue with user code or ISP handler is made. If P2_7 is sampled low and the watchdog overflow flag is set, the external hardware request to start the ISP command handler is ignored. If there is no request for the ISP command handler execution (P2_7 is sampled HIGH after reset), a search is made for a valid user program. If a valid user program is found then the execution control is transferred to it. If a valid user program is not found, the auto-baud routine is invoked.

Pin P2_7 is used as a hardware request signal for ISP and therefore requires special attention. Since P2_7 is in high impedance mode after reset, it is important that the user provides external hardware (a pull-up resistor or other device) to put the pin in a defined state. Otherwise unintended entry into ISP mode may occur.

When ISP mode is entered after a power on reset, the IRC and PLL1 are used to generate the CCLK of 96 MHz. The UART0 pins are set to P2_0 and P2_1.

After determining the host's baud rate, the test string "Synchronized" is sent to a host. After a successful handshake, ISP enters the command interpret mode.

A hardware flash signature generation capability is built into the flash memory. This feature can be used to create a signature that can then be used to verify flash contents. Details of flash signature generation are shown in Section 45.11.

45.4.1 Memory map after any reset

When a user program begins execution after reset, the interrupt vectors are set to point to the beginning of flash memory (see Figure 3).

45.4.1.1 Criterion for Valid User Code

The reserved Cortex-M4 exception vector location 7 (offset 0x 001C in the vector table) should contain the 2's complement of the check-sum of table entries 0 through 6. This causes the checksum of the first 8 table entries to be 0. The boot loader code checksums the first 8 locations in sector 0 of the flash. If the result is 0, then execution control is transferred to the user code.

If the signature is not valid, the auto-baud routine synchronizes with the host via serial port 0. The host should send a "?" (0x3F) as a synchronization character and wait for a response. The host side serial port settings should be 8 data bits, 1 stop bit and no parity. The auto-baud routine measures the bit time of the received synchronization character in terms of its own frequency and programs the baud rate generator of the serial port. It also sends an ASCII string ("Synchronized<CR><LF>") to the host. In response to this the host should send the same string ("Synchronized<CR><LF>"). The auto-baud routine looks at the received characters to verify synchronization. If synchronization is verified then "OK<CR><LF>" string is sent to the host. The host should respond by sending the crystal frequency (in kHz) at which the part is running. For example, if the part is running at 10 MHz, the response from the host should be "10000<CR><LF>". "OK<CR><LF>" string is sent to the host after receiving the crystal frequency. If synchronization is not verified then the auto-baud routine waits again for a synchronization character. For auto-baud to work correctly in case of user invoked ISP, the CCLK frequency should be greater than or equal to 10 MHz.

Once the crystal frequency is received the part is initialized and the ISP command handler is invoked. For safety reasons an "Unlock" command is required before executing the commands resulting in flash erase/write operations and the "Go" command. The rest of the commands can be executed without the unlock command. The Unlock command is required to be executed once per ISP session. The Unlock command is explained in Section 45.8 "ISP commands" on page 1088.

45.4.2 Communication protocol

All ISP commands should be sent as single ASCII strings. Strings should be terminated with Carriage Return (CR) and/or Line Feed (LF) control characters. Extra <CR> and <LF> characters are ignored. All ISP responses are sent as <CR><LF> terminated ASCII strings. Data is sent and received in UU-encoded format.

45.4.2.1 ISP command format

"Command Parameter_0 Parameter_1 … Parameter_n<CR><LF>" "Data" (Data only for Write commands).

45.4.2.2 ISP response format

"Return_Code<CR><LF>Response_0<CR><LF>Response_1<CR><LF> … Response_n<CR><LF>" "Data" (Data only for Read commands).

45.4.2.3 ISP data format

The data stream is in UU-encoded format. The UU-encode algorithm converts 3 bytes of binary data in to 4 bytes of printable ASCII character set. It is more efficient than Hex format which converts 1 byte of binary data in to 2 bytes of ASCII hex. The sender should send the check-sum after transmitting 20 UU-encoded lines. The length of any UU-encoded line should not exceed 61 characters (bytes) i.e. it can hold 45 data bytes. The receiver should compare it with the check-sum of the received bytes. If the check-sum matches then the receiver should respond with "OK<CR><LF>" to continue further transmission. If the check-sum does not match the receiver should respond with "RESEND<CR><LF>". In response the sender should retransmit the bytes.

45.4.2.4 ISP flow control

A software XON/XOFF flow control scheme is used to prevent data loss due to buffer overrun. When the data arrives rapidly, the ASCII control character DC3 (0x13) is sent to stop the flow of data. Data flow is resumed by sending the ASCII control character DC1 (0x11). The host should also support the same flow control scheme.

45.4.2.5 ISP command abort

Commands can be aborted by sending the ASCII control character "ESC" (0x1B). This feature is not documented as a command under "ISP Commands" section. Once the escape code is received the ISP command handler waits for a new command.

45.4.2.6 Interrupts during IAP

The on-chip flash memory is not accessible during erase/write operations. When the user application code starts executing the interrupt vectors from the user flash area are active. The user should either disable interrupts, or ensure that user interrupt vectors are active in RAM and that the interrupt handlers reside in RAM, before making a flash erase/write IAP call. The IAP code does not use or disable interrupts.

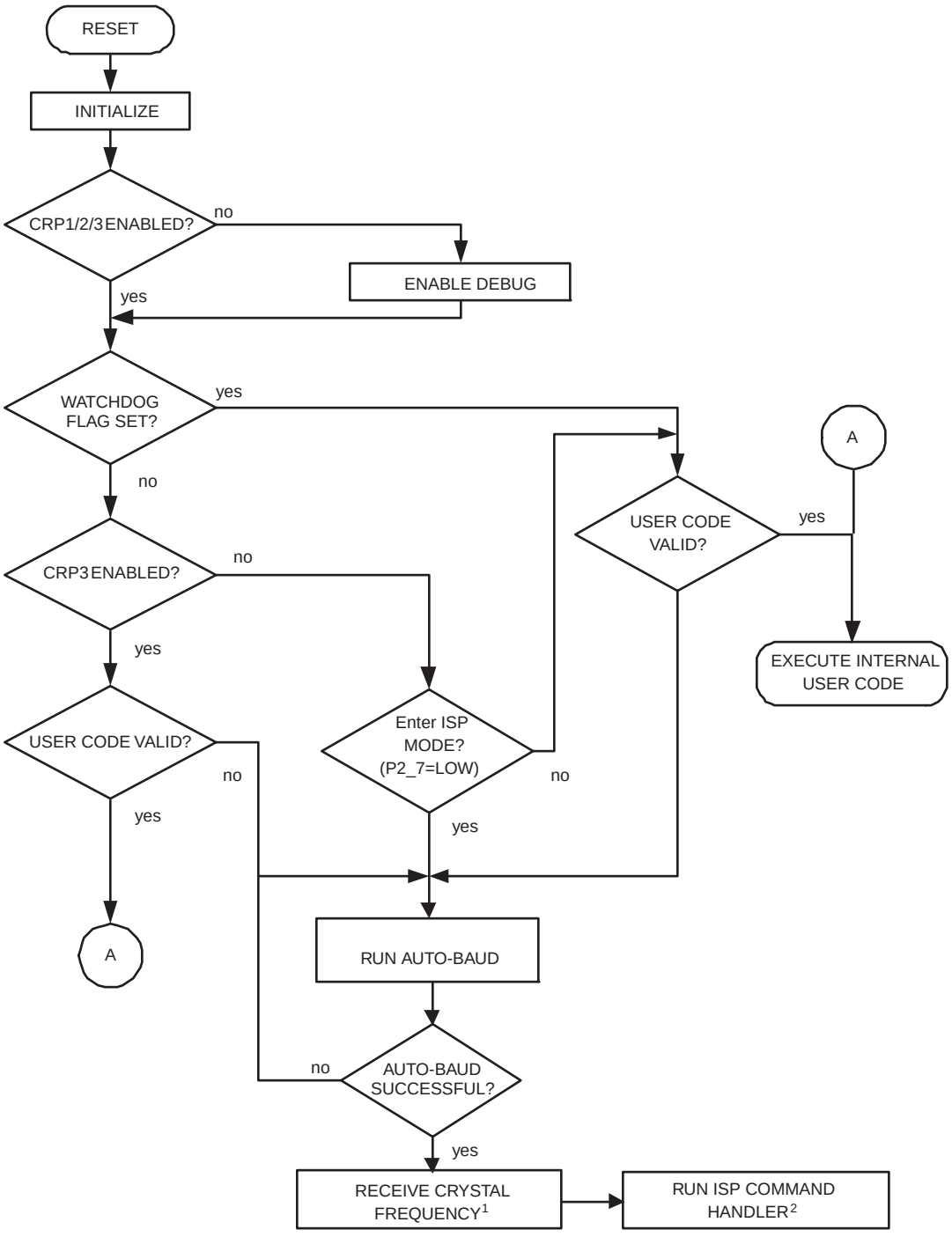
45.4.2.7 RAM used by ISP command handler

ISP commands use on-chip RAM from 0x1000 0118 to 0x1000 01FF. The user could use this area, but the contents may be lost upon reset. Flash programming commands use the top 32 bytes of on-chip RAM. The stack is located at RAM top - 32. The maximum stack usage is 256 bytes and grows downwards.

45.4.2.8 RAM used by IAP command handler

Flash programming commands use the top 32 bytes of on-chip RAM. The maximum stack usage in the user allocated stack space is 128 bytes and grows downwards.

45.5 Boot process flowchart for LPC43xx parts with flash



(1) For details on handling the crystal frequency, see Section 45.9.9 “Re-invoke ISP” on page 1101

(2) For details on available ISP commands based on the CRP settings see Section 45.7 “Code Read Protection (CRP)”

Fig 172. Boot process flowchart for LPC43xx parts with flash

45.6 Sector numbers

Some IAP and ISP commands operate on "sectors" and specify sector numbers. The following table indicate the correspondence between sector numbers and memory addresses for LPC43xx device. IAP and ISP routines are located in the Boot ROM.

Table 992. Flash configuration

Flash bank	Sector number	Sector size [kB]	Start address	End address	LPC43x2	LPC43x3	LPC43x5	LPC43x7
A	0	8	0x1A00 0000	0x1A00 1FFF	x	x	x	x
A	1	8	0x1A00 2000	0x1A00 3FFF	x	x	x	x
A	2	8	0x1A00 4000	0x1A00 5FFF	x	x	x	x
A	3	8	0x1A00 6000	0x1A00 7FFF	x	x	x	x
A	4	8	0x1A00 8000	0x1A00 9FFF	x	x	x	x
A	5	8	0x1A00 A000	0x1A00 BFFF	x	x	x	x
A	6	8	0x1A00 C000	0x1A00 DFFF	x	x	x	x
A	7	8	0x1A00 E000	0x1A00 FFFF	x	x	x	x
A	8	64	0x1A01 0000	0x1A01 FFFF	x	x	x	x
A	9	64	0x1A02 0000	0x1A02 FFFF	x	x	x	x
A	10	64	0x1A03 0000	0x1A03 FFFF	x	x	x	x
A	11	64	0x1A04 0000	0x1A04 FFFF	x		x	x
A	12	64	0x1A05 0000	0x1A05 FFFF	x		x	x
A	13	64	0x1A06 0000	0x1A06 FFFF	x			x
A	14	64	0x1A07 0000	0x1A07 FFFF	x			x
B	0	8	0x1B00 0000	0x1B00 1FFF		x	x	x
B	1	8	0x1B00 2000	0x1B00 3FFF		x	x	x
B	2	8	0x1B00 4000	0x1B00 5FFF		x	x	x
B	3	8	0x1B00 6000	0x1B00 7FFF		x	x	x
B	4	8	0x1B00 8000	0x1B00 9FFF		x	x	x
B	5	8	0x1B00 A000	0x1B00 BFFF		x	x	x
B	6	8	0x1B00 C000	0x1B00 DFFF		x	x	x
B	7	8	0x1B00 E000	0x1B00 FFFF		x	x	x
B	8	64	0x1B01 0000	0x1B01 FFFF		x	x	x
B	9	64	0x1B02 0000	0x1B02 FFFF		x	x	x
B	10	64	0x1B03 0000	0x1B03 FFFF		x	x	x
B	11	64	0x1B04 0000	0x1B04 FFFF			x	x
B	12	64	0x1B05 0000	0x1B05 FFFF			x	x
B	13	64	0x1B06 0000	0x1B06 FFFF				x
B	14	64	0x1B07 0000	0x1B07 FFFF				x

45.7 Code Read Protection (CRP)

Code Read Protection is a mechanism that allows user to enable different levels of security in the system so that access to the on-chip flash and use of the ISP can be restricted. When needed, CRP is invoked by programming a specific pattern in flash location at 0x000002FC. IAP commands are not affected by the code read protection.

Important: Any CRP change becomes effective only after the device has gone through a power cycle.

Table 993. Code Read Protection options

Name	Pattern programmed in 0x000002FC	Description
CRP1	0x12345678	Access to chip via the JTAG pins is disabled. This mode allows partial flash update using the following ISP commands and restrictions: • Because the ISP code uses SRAM, The Write to RAM command can not access SRAM below 0x1000 0200, see Section 45.4.2.7. • Read Memory command: disabled. • Copy RAM to Flash command: cannot write to Sector 0. • Go command: disabled. • Erase sector(s) command: can erase any individual sector except sector 0 only, or can erase all sectors at once. • Compare command: disabled This mode is useful when CRP is required and flash field updates are needed but all sectors can not be erased. The compare command is disabled, so in the case of partial flash updates the secondary loader should implement a checksum mechanism to verify the integrity of the flash.
CRP2	0x87654321	This is similar to CRP1 with the following additions: • Write to RAM command: disabled. • Copy RAM to Flash: disabled. • Erase command: only allows erase of all sectors.
CRP3	0x43218765	This is similar to CRP2, but ISP entry by pulling P2_7 LOW is disabled if a valid user code is present in flash sector 0. This mode effectively disables ISP override using the P2_7 pin. It is up to the user's application to provide for flash updates by using IAP calls or by invoking ISP with UART0. **Caution: If CRP3 is selected, no future factory testing can be performed on the device.**

Table 994. Code Read Protection hardware/software interaction

CRP option	User Code Valid	P2_7 pin at reset	JTAG enabled	LPC43xx enters ISP mode	partial flash update in ISP mode
None	No	X	Yes	Yes	Yes
	Yes	High	Yes	No	NA
	Yes	Low	Yes	Yes	Yes
CRP1	No	x	No	Yes	Yes
	Yes	High	No	No	NA
	Yes	Low	No	Yes	Yes
CRP2	No	x	No	Yes	No
	Yes	High	No	No	NA
	Yes	Low	No	Yes	No
CRP3	No	x	No	Yes	No
	Yes	x	No	No	NA

If any CRP mode is enabled and access to the chip is allowed via the ISP, an unsupported or restricted ISP command will be terminated with return code CODE_READ_PROTECTION_ENABLED.

45.8 ISP commands

The following commands are accepted by the ISP command handler. Detailed status codes are supported for each command. The command handler sends the return code INVALID_COMMAND when an undefined command is received. Commands and return codes are in ASCII format.

CMD_SUCCESS is sent by ISP command handler only when received ISP command has been completely executed and the new ISP command can be given by the host. Exceptions from this rule are "Set Baud Rate", "Write to RAM", "Read Memory", and "Go" commands.

Table 995. ISP command summary

ISP Command	Usage	Flashless parts	Parts with flash	Described in
Unlock	U <Unlock Code>	yes	yes	Table 996
Set Baud Rate	B <Baud Rate> <stop bit>	yes	yes	Table 997
Echo	A <setting>	yes	yes	Table 999
Write to RAM	W <start address> <number of bytes>	yes	yes	Table 1000
Read Memory	R <address> <number of bytes>	yes	yes	Table 1001
Prepare sector(s) for write operation	P <start sector number> <end sector number>	no	yes	Table 1002
Copy RAM to Flash	C <flash address> <RAM address> <number of bytes>	no	yes	Table 1003
Go	G <address> <Mode>	yes	yes	Table 1004
Erase sector(s)	E <start sector number> <end sector number>	no	yes	Table 1005
Blank check sector(s)	I <start sector number> <end sector number>	no	yes	Table 1006
Read Part ID	J	yes	yes	Table 1007
Read Boot Code version	K	yes	yes	Table 1009
Read serial number	N	yes	yes	Table 1010
Compare	M <address1> <address2> <number of bytes>	no	yes	Table 1011

45.8.1 Unlock <Unlock code>

Table 996. ISP Unlock command

Command	U
Input	Unlock code: 23130_{10}
Return Code	CMD_SUCCESS | INVALID_CODE | PARAM_ERROR
Description	This command is used to unlock Flash Write, Erase, and Go commands.
Example	"U 23130<CR><LF>" unlocks the Flash Write/Erase & Go commands.

45.8.2 Set Baud Rate <Baud Rate> <stop bit>

Table 997. ISP Set Baud Rate command

Command	**B**
Input	Baud Rate: 9600 | 19200 | 38400 | 57600 | 115200 Stop bit: 1 | 2
Return Code	CMD_SUCCESS | INVALID_BAUD_RATE | INVALID_STOP_BIT | PARAM_ERROR
Description	This command is used to change the baud rate. The new baud rate is effective after the command handler sends the CMD_SUCCESS return code.
Example	"B 57600 1<CR><LF>" sets the serial port to baud rate 57600 bps and 1 stop bit.

Table 998. Correlation between possible ISP baudrates and CCLK frequency (in MHz)

ISP Baudrate .vs. CCLK Frequency	**9600**	**19200**	**38400**	**57600**	**115200**
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>
<tbd>	<tbd>	<tbd>	<tbd>	<tbd>	<tbd>

[1] ISP entry after reset uses the on-chip IRC and PLL1 to run the device at CCLK = 96 MHz.

45.8.3 Echo <setting>

Table 999. ISP Echo command

Command	**A**
Input	Setting: ON = 1 | OFF = 0
Return Code	CMD_SUCCESS | PARAM_ERROR
Description	The default setting for echo command is ON. When ON the ISP command handler sends the received serial data back to the host.
Example	"A 0<CR><LF>" turns echo off.

45.8.4 Write to RAM <start address> <number of bytes>

The host should send the data only after receiving the CMD_SUCCESS return code. The host should send the check-sum after transmitting 20 UU-encoded lines. The checksum is generated by adding raw data (before UU-encoding) bytes and is reset after transmitting 20 UU-encoded lines. The length of any UU-encoded line should not exceed 61 characters (bytes) i.e. it can hold 45 data bytes. When the data fits in less than 20 UU-encoded lines then the check-sum should be of the actual number of bytes sent.

The ISP command handler compares it with the check-sum of the received bytes. If the check-sum matches, the ISP command handler responds with "OK<CR><LF>" to continue further transmission. If the check-sum does not match, the ISP command handler responds with "RESEND<CR><LF>". In response the host should retransmit the bytes.

Table 1000.ISP Write to RAM command

Command	W
Input	**Start Address:** RAM address where data bytes are to be written. This address should be a word boundary. **Number of Bytes:** Number of bytes to be written. Count should be a multiple of 4
Return Code	CMD_SUCCESS | ADDR_ERROR (Address not on word boundary) | ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not multiple of 4) | PARAM_ERROR | CODE_READ_PROTECTION_ENABLED
Description	This command is used to download data to RAM. Data should be in UU-encoded format. This command is blocked when code read protection levels CRP2 or CRP3 are enabled.
Example	"W 268435968 4<CR><LF>" writes 4 bytes of data to address 0x1000 0200.

45.8.5 Read Memory <address> <no. of bytes>

The data stream is followed by the command success return code. The check-sum is sent after transmitting 20 UU-encoded lines. The checksum is generated by adding raw data (before UU-encoding) bytes and is reset after transmitting 20 UU-encoded lines. The length of any UU-encoded line should not exceed 61 characters (bytes) i.e. it can hold 45 data bytes. When the data fits in less than 20 UU-encoded lines then the check-sum is of actual number of bytes sent. The host should compare it with the checksum of the received bytes. If the check-sum matches then the host should respond with "OK<CR><LF>" to continue further transmission. If the check-sum does not match then the host should respond with "RESEND<CR><LF>". In response the ISP command handler sends the data again.

Table 1001.ISP Read Memory command

Command	R
Input	**Start Address:** Address from where data bytes are to be read. This address should be a word boundary. **Number of Bytes:** Number of bytes to be read. Count should be a multiple of 4.
Return Code	CMD_SUCCESS followed by <actual data (UU-encoded)> | ADDR_ERROR (Address not on word boundary) | ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not a multiple of 4) | PARAM_ERROR | CODE_READ_PROTECTION_ENABLED
Description	This command is used to read data from RAM or flash memory. This command is blocked when any level of code read protection is enabled.
Example	"R 268435968 4<CR><LF>" reads 4 bytes of data from address 0x1000 0200.

45.8.6 Prepare sector(s) for write operation <start sector number> <end sector number>

This command makes flash write/erase operation a two step process.

Table 1002.ISP Prepare sector(s) for write operation command

Command	P
Input	**Start Sector Number** **End Sector Number:** Should be greater than or equal to start sector number.
Return Code	CMD_SUCCESS | BUSY | INVALID_SECTOR | PARAM_ERROR
Description	This command must be executed before executing "Copy RAM to Flash" or "Erase Sector(s)" command. Successful execution of the "Copy RAM to Flash" or "Erase Sector(s)" command causes relevant sectors to be protected again. To prepare a single sector use the same "Start" and "End" sector numbers.
Example	"P 0 0<CR><LF>" prepares the flash sector 0.

45.8.7 Copy RAM to Flash <flash address> <RAM address> <no of bytes>

Table 1003.ISP Copy command

Command	C
Input	**Flash Address(DST):** Destination flash address where data bytes are to be written. The destination address should be a 256 byte boundary. **RAM Address(SRC):** Source RAM address from where data bytes are to be read. **Number of Bytes:** Number of bytes to be written. Should be 256 | 512 | 1024 | 4096.
Return Code	CMD_SUCCESS | SRC_ADDR_ERROR (Address not on word boundary) | DST_ADDR_ERROR (Address not on correct boundary) | SRC_ADDR_NOT_MAPPED | DST_ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not 256 | 512 | 1024 | 4096) | SECTOR_NOT_PREPARED_FOR WRITE_OPERATION | BUSY | CMD_LOCKED | PARAM_ERROR | CODE_READ_PROTECTION_ENABLED
Description	This command is used to program the flash memory. The "Prepare Sector(s) for Write Operation" command should precede this command. The affected sectors are automatically protected again once the copy command is successfully executed. This command is blocked when code read protection levels CRP2 or CRP3 are enabled. When code read protection level CRP1 is enabled, individual sectors other than sector 0 can be written.
Example	"C 0 268468224 512<CR><LF>" copies 512 bytes from the RAM address 0x1000 8000 to the flash address 0.

45.8.8 Go <address> <mode>

Table 1004.ISP Go command

Command	G
Input	**Address:** Flash or RAM address from which the code execution is to be started. This address should be on a word boundary. **Mode (retained for backward compatibility):** T (Execute program in Thumb Mode) | A (not allowed).
Return Code	CMD_SUCCESS | ADDR_ERROR | ADDR_NOT_MAPPED | CMD_LOCKED | PARAM_ERROR | CODE_READ_PROTECTION_ENABLED
Description	This command is used to execute a program residing in RAM or flash memory. It may not be possible to return to the ISP command handler once this command is successfully executed. This command is blocked when any level of code read protection is enabled.
Example	"G 0 T<CR><LF>" branches to address 0x0000 0000.

When the GO command is used, execution begins at the specified address (assuming it is an executable address) with the device left as it was configured for the ISP code. This means that some things are different than they would be for entering user code directly following a chip reset. Most importantly, the main PLL will be running and connected, configured to generate a CPU clock with a frequency of approximately 14.7456 MHz.

45.8.9 Erase sector(s) <start sector number> <end sector number>

Table 1005.ISP Erase sector command

Command	E
Input	**Start Sector Number** **End Sector Number:** Should be greater than or equal to start sector number.
Return Code	CMD_SUCCESS | BUSY | INVALID_SECTOR | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | CMD_LOCKED | PARAM_ERROR | CODE_READ_PROTECTION_ENABLED
Description	This command is used to erase one or more sector(s) of on-chip flash memory. This command is blocked when code read protection level CRP3 is enabled. When code read protection level CRP1 is enabled, individual sectors other than sector 0 can be erased. All sectors can be erased at once in CRP1 and CRP2.
Example	"E 2 3<CR><LF>" erases the flash sectors 2 and 3.

45.8.10 Blank check sector(s) <sector number> <end sector number>

Table 1006.ISP Blank check sector command

Command	I
Input	**Start Sector Number:** **End Sector Number:** Should be greater than or equal to start sector number.
Return Code	CMD_SUCCESS | SECTOR_NOT_BLANK (followed by <Offset of the first non blank word location> <Contents of non blank word location>) | INVALID_SECTOR | PARAM_ERROR |
Description	This command is used to blank check one or more sectors of on-chip flash memory.
Example	"I 2 3<CR><LF>" blank checks the flash sectors 2 and 3.

45.8.11 Read Part Identification number

Table 1007.ISP Read Part Identification command

Command	J
Input	None.
Return Code	CMD_SUCCESS followed by part identification number in ASCII (see Table 1008 "LPC43xx part identification numbers").
Description	This command is used to read the part identification number. The part identification number maps to a feature subset within a device family. This number will not normally change as a result of technical revisions.

Table 1008.LPC43xx part identification numbers

Device	ASCII/dec coding	Hex coding
<tbd>	<tbd>	<tbd>

45.8.12 Read Boot Code version number

Table 1009.ISP Read Boot Code version number command

Command	K
Input	None
Return Code	CMD_SUCCESS followed by 2 bytes of boot code version number in ASCII format. It is to be interpreted as <byte1(Major)>.<byte0(Minor)>.
Description	This command is used to read the boot code version number.

45.8.13 Read device serial number

Table 1010.ISP Read device serial number command

Command	N
Input	None.
Return Code	CMD_SUCCESS followed by the device serial number in 4 decimal ASCII groups, each representing a 32-bit value.
Description	This command is used to read the device serial number. The serial number may be used to uniquely identify a single unit among all LPC43xx devices.

45.8.14 Compare <address1> <address2> <no of bytes>

Table 1011. ISP Compare command

Command	M
Input	**Address1 (DST):** Starting flash or RAM address of data bytes to be compared. This address should be a word boundary. **Address2 (SRC):** Starting flash or RAM address of data bytes to be compared. This address should be a word boundary. **Number of Bytes:** Number of bytes to be compared; should be a multiple of 4.
Return Code	CMD_SUCCESS | (Source and destination data are equal) COMPARE_ERROR | (Followed by the offset of first mismatch) COUNT_ERROR (Byte count is not a multiple of 4) | ADDR_ERROR | ADDR_NOT_MAPPED | PARAM_ERROR |
Description	This command is used to compare the memory contents at two locations. This command is blocked when any level of code read protection is enabled.
Example	"M 8192 268435968 4<CR><LF>" compares 4 bytes from the RAM address 0x1000 0200 to the 4 bytes from the flash address 0x2000.

45.8.15 ISP Return Codes

Table 1012. ISP Return Codes Summary

Return Code	Mnemonic	Description
0	CMD_SUCCESS	Command is executed successfully. Sent by ISP handler only when command given by the host has been completely and successfully executed.
1	INVALID_COMMAND	Invalid command.
2	SRC_ADDR_ERROR	Source address is not on word boundary.
3	DST_ADDR_ERROR	Destination address is not on a correct boundary.
4	SRC_ADDR_NOT_MAPPED	Source address is not mapped in the memory map. Count value is taken into consideration where applicable.
5	DST_ADDR_NOT_MAPPED	Destination address is not mapped in the memory map. Count value is taken into consideration where applicable.
6	COUNT_ERROR	Byte count is not multiple of 4 or is not a permitted value.
7	INVALID_SECTOR	Sector number is invalid or end sector number is greater than start sector number.
8	SECTOR_NOT_BLANK	Sector is not blank.
9	SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION	Command to prepare sector for write operation was not executed.
10	COMPARE_ERROR	Source and destination data not equal.
11	BUSY	Flash programming hardware interface is busy.
12	PARAM_ERROR	Insufficient number of parameters or invalid parameter.

Table 1012.ISP Return Codes Summary

Return Code	Mnemonic	Description
13	ADDR_ERROR	Address is not on word boundary.
14	ADDR_NOT_MAPPED	Address is not mapped in the memory map. Count value is taken in to consideration where applicable.
15	CMD_LOCKED	Command is locked.
16	INVALID_CODE	Unlock code is invalid.
17	INVALID_BAUD_RATE	Invalid baud rate setting.
18	INVALID_STOP_BIT	Invalid stop bit setting.
19	CODE_READ_PROTECTION_ ENABLED	Code read protection enabled.

45.9 IAP commands

For in application programming the IAP routine should be called with a word pointer in register r0 pointing to memory (RAM) containing command code and parameters. Result of the IAP command is returned in the result table pointed to by register r1. The user can reuse the command table for result by passing the same pointer in registers r0 and r1. The parameter table should be big enough to hold all the results in case if number of results are more than number of parameters. Parameter passing is illustrated in the Figure 173. The number of parameters and results vary according to the IAP command. The maximum number of parameters is 5, passed to the "Copy RAM to Flash" command. The maximum number of results is 4, returned by the "Read device serial number" command. The command handler sends the status code INVALID_COMMAND when an undefined command is received. The IAP routine resides at location 0x1FFF 1FF0.

The IAP function could be called in the following way using C.

Define the IAP location entry point. Bit 0 of the IAP location is set since the Cortex-M4 uses only Thumb mode.

```
#define IAP_LOCATION 0x1FFF1FF1
```

Define data structure or pointers to pass IAP command table and result table to the IAP function:

```
unsigned long command[5];
unsigned long result[5];
```

or

```
unsigned long * command;
unsigned long * result;
command=(unsigned long *) 0x...
result= (unsigned long *) 0x...
```

Define pointer to function type, which takes two parameters and returns void. Note the IAP returns the result with the base address of the table residing in R1.

```
typedef void (*IAP)(unsigned int [],unsigned int[]);
IAP iap_entry;
```

Setting function pointer:

```
iap_entry=(IAP) IAP_LOCATION;
```

Whenever you wish to call IAP you could use the following statement.

```
iap_entry (command, result);
```

The IAP call could be simplified further by using the symbol definition file feature supported by ARM Linker in ADS (ARM Developer Suite). You could also call the IAP routine using assembly code.

As per the ARM specification (The ARM Thumb Procedure Call Standard SWS ESPC 0002 A-05) up to 4 parameters can be passed in the r0, r1, r2 and r3 registers respectively. Additional parameters are passed on the stack. Up to 4 parameters can be

returned in the r0, r1, r2 and r3 registers respectively. Additional parameters are returned indirectly via memory. Some of the IAP calls require more than 4 parameters. If the ARM suggested scheme is used for the parameter passing/returning then it might create problems due to difference in the C compiler implementation from different vendors. The suggested parameter passing scheme reduces such risk.

The flash memory is not accessible during a write or erase operation. IAP commands, which results in a flash write/erase operation, use 32 bytes of space in the top portion of the on-chip RAM for execution. The user program should not be use this space if IAP flash programming is permitted in the application.

Table 1013.IAP Command Summary

IAP Command	Command Code	Described in
Prepare sector(s) for write operation	50_{10}	Table 1014
Copy RAM to Flash	51_{10}	Table 1015
Erase sector(s)	52_{10}	Table 1016
Blank check sector(s)	53_{10}	Table 1017
Read part ID	54_{10}	Table 1018
Read Boot Code version	55_{10}	Table 1019
Read device serial number	58_{10}	Table 1020
Compare	56_{10}	Table 1021
Reinvoke ISP	57_{10}	Table 1022

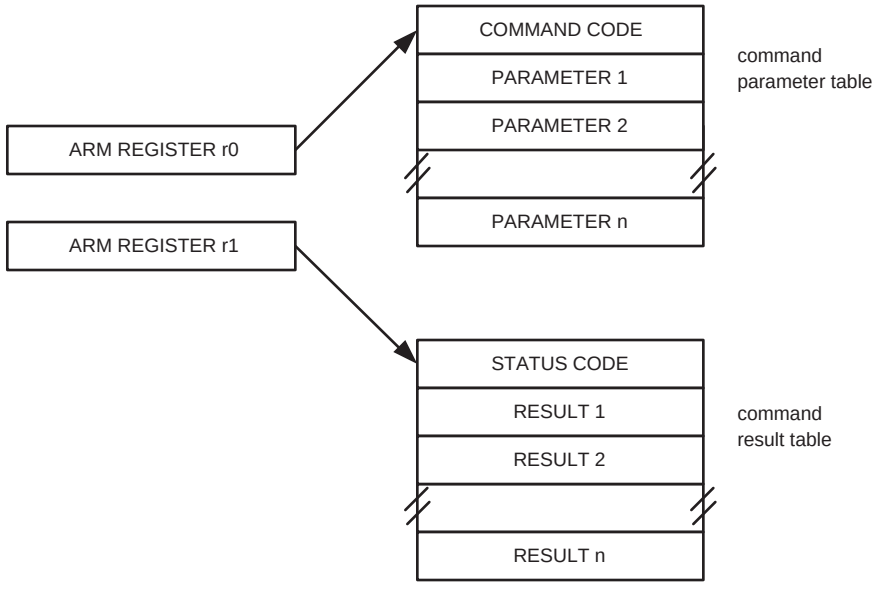


Fig 173. IAP parameter passing

45.9.1 Prepare sector(s) for write operation

This command makes flash write/erase operation a two step process.

Table 1014.IAP Prepare sector(s) for write operation command

Command	Prepare sector(s) for write operation
Input	**Command code: 50 (decimal)** **Param0:** Start Sector Number **Param1:** End Sector Number (should be greater than or equal to start sector number).
Return Code	CMD_SUCCESS | BUSY | INVALID_SECTOR
Result	None
Description	This command must be executed before executing "Copy RAM to Flash" or "Erase Sector(s)" command. Successful execution of the "Copy RAM to Flash" or "Erase Sector(s)" command causes relevant sectors to be protected again. To prepare a single sector use the same "Start" and "End" sector numbers.

45.9.2 Copy RAM to Flash

Table 1015.IAP Copy RAM to Flash command

Command	Copy RAM to Flash
Input	**Command code: 51 (decimal)** **Param0(DST):** Destination flash address where data bytes are to be written. This address should be a 256 byte boundary. **Param1(SRC):** Source RAM address from which data bytes are to be read. This address should be a word boundary. **Param2:** Number of bytes to be written. Should be 256 | 512 | 1024 | 4096. **Param3:** CPU Clock Frequency (CCLK) in kHz.
Return Code	CMD_SUCCESS | SRC_ADDR_ERROR (Address not a word boundary) | DST_ADDR_ERROR (Address not on correct boundary) | SRC_ADDR_NOT_MAPPED | DST_ADDR_NOT_MAPPED | COUNT_ERROR (Byte count is not 256 | 512 | 1024 | 4096) | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | BUSY |
Result	None
Description	This command is used to program the flash memory. The affected sectors should be prepared first by calling "Prepare Sector for Write Operation" command. The affected sectors are automatically protected again once the copy command is successfully executed.

45.9.3 Erase Sector(s)

Table 1016.IAP Erase Sector(s) command

Command	Erase Sector(s)
Input	**Command code: 52 (decimal)** **Param0:** Start Sector Number **Param1:** End Sector Number (should be greater than or equal to start sector number). **Param2:** CPU Clock Frequency (CCLK) in kHz.
Return Code	CMD_SUCCESS | BUSY | SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION | INVALID_SECTOR
Result	None
Description	This command is used to erase a sector or multiple sectors of on-chip flash memory. To erase a single sector use the same "Start" and "End" sector numbers.

45.9.4 Blank check sector(s)

Table 1017.IAP Blank check sector(s) command

Command	Blank check sector(s)
Input	**Command code: 53 (decimal)** **Param0:** Start Sector Number **Param1:** End Sector Number (should be greater than or equal to start sector number).
Return Code	CMD_SUCCESS | BUSY | SECTOR_NOT_BLANK | INVALID_SECTOR
Result	**Result0:** Offset of the first non blank word location if the Status Code is SECTOR_NOT_BLANK. **Result1:** Contents of non blank word location.
Description	This command is used to blank check a sector or multiple sectors of on-chip flash memory. To blank check a single sector use the same "Start" and "End" sector numbers.

45.9.5 Read part identification number

Table 1018.IAP Read part identification number command

Command	Read part identification number
Input	**Command code: 54 (decimal)** **Parameters:** None
Return Code	CMD_SUCCESS |
Result	**Result0:** Part Identification Number.
Description	This command is used to read the part identification number. The value returned is the hexadecimal version of the part ID. See Table 1008 “LPC43xx part identification numbers”.

45.9.6 Read Boot Code version number

Table 1019.IAP Read Boot Code version number command

Command	Read boot code version number
Input	**Command code: 55 (decimal)** **Parameters:** None
Return Code	CMD_SUCCESS |
Result	**Result0:** 2 bytes of boot code version number in ASCII format. It is to be interpreted as <byte1(Major)>.<byte0(Minor)>
Description	This command is used to read the boot code version number.

45.9.7 Read device serial number

Table 1020.IAP Read device serial number command

Command	Read device serial number
Input	**Command code: 58 (decimal)** **Parameters:** None
Return Code	CMD_SUCCESS |
Result	**Result0:** First 32-bit word of Device Identification Number (at the lowest address) **Result1:** Second 32-bit word of Device Identification Number **Result2:** Third 32-bit word of Device Identification Number **Result3:** Fourth 32-bit word of Device Identification Number
Description	This command is used to read the device identification number. The serial number may be used to uniquely identify a single unit among all LPC43xx devices.

45.9.8 Compare <address1> <address2> <no of bytes>

Table 1021.IAP Compare command

Command	Compare
Input	**Command code: 56 (decimal)** **Param0(DST):** Starting flash or RAM address of data bytes to be compared. This address should be a word boundary. **Param1(SRC):** Starting flash or RAM address of data bytes to be compared. This address should be a word boundary. **Param2:** Number of bytes to be compared; should be a multiple of 4.
Return Code	CMD_SUCCESS | COMPARE_ERROR | COUNT_ERROR (Byte count is not a multiple of 4) | ADDR_ERROR | ADDR_NOT_MAPPED
Result	**Result0:** Offset of the first mismatch if the Status Code is COMPARE_ERROR.
Description	This command is used to compare the memory contents at two locations. **The result may not be correct when the source or destination includes any of the first 64 bytes starting from address zero. The first 64 bytes can be re-mapped to RAM.**

45.9.9 Re-invoke ISP

Table 1022.Re-invoke ISP

Command	Compare
Input	**Command code: 57 (decimal)**
Return Code	None
Result	**None.**
Description	This command is used to invoke the boot loader in ISP mode. It maps boot vectors, sets the clock to 96 MHz, configures UART0 pins U0_RX and U0_TX, resets TIMER1 and resets the U0FDR (see Table 817). This command may be used when a valid user program is present in the internal flash memory and the P2_7 pin is not accessible to force the ISP mode. The command does not disable the PLL1 hence it is possible to invoke the boot loader when the part is running off the PLL1. In this case, the ISP utility must pass the PLL1 output frequency after the auto baud handshake. Another option is to disable the PLL1 and select the IRC as the clock source before making this IAP call. In this case, the frequency sent by ISP is ignored and IRC and PLL1 are used to generate a 14.748 MHz clock.

45.9.10 IAP Status Codes

Table 1023.IAP Status Codes Summary

Status Code	Mnemonic	Description
0	CMD_SUCCESS	Command is executed successfully.
1	INVALID_COMMAND	Invalid command.
2	SRC_ADDR_ERROR	Source address is not on a word boundary.
3	DST_ADDR_ERROR	Destination address is not on a correct boundary.
4	SRC_ADDR_NOT_MAPPED	Source address is not mapped in the memory map. Count value is taken in to consideration where applicable.
5	DST_ADDR_NOT_MAPPED	Destination address is not mapped in the memory map. Count value is taken in to consideration where applicable.
6	COUNT_ERROR	Byte count is not multiple of 4 or is not a permitted value.
7	INVALID_SECTOR	Sector number is invalid.
8	SECTOR_NOT_BLANK	Sector is not blank.
9	SECTOR_NOT_PREPARED_FOR_WRITE_OPERATION	Command to prepare sector for write operation was not executed.
10	COMPARE_ERROR	Source and destination data is not same.
11	BUSY	Flash programming hardware interface is busy.

45.10 JTAG flash programming interface

Debug tools can write parts of the flash image to the RAM and then execute the IAP call "Copy RAM to Flash" repeatedly with proper offset.

45.11 Flash signature generation

The flash module contains a built-in signature generator. This generator can produce a 128-bit signature from a range of flash memory. A typical usage is to verify the flashed contents against a calculated signature (e.g. during programming).

The address range for generating a signature must be aligned on flash-word boundaries, i.e. 128-bit boundaries. Once started, signature generation completes independently. While signature generation is in progress, the flash memory cannot be accessed for other purposes, and an attempted read will cause a wait state to be asserted until signature generation is complete. Code outside of the flash (e.g. internal RAM) can be executed during signature generation. This can include interrupt services, if the interrupt vector table is re-mapped to memory other than the flash memory. The code that initiates signature generation should also be placed outside of the flash memory.

45.11.1 Register description for signature generation

Table 1024.Register overview: FMC (base address 0x4008 4000)

Name	Description	Access	Reset Value	Address	Reference
FMSSTART	Signature start address register	R/W	0	0x4008 4020	Table 1025
FMSSTOP	Signature stop-address register	R/W	0	0x4008 4024	Table 1026
FMSW0	128-bit signature Word 0	R	-	0x4008 402C	Table 1027
FMSW1	128-bit signature Word 1	R	-	0x4008 4030	Table 1028
FMSW2	128-bit signature Word 2	R	-	0x4008 4034	Table 1029
FMSW3	128-bit signature Word 3	R	-	0x4008 4038	Table 1030
FMSTAT	Signature generation status register	R	0	0x4008 4FE0	Section 45.11.1.3
FMSTATCLR	Signature generation status clear register	W	-	0x4008 4FE8	Section 45.11.1.4

45.11.1.1 Signature generation address and control registers

These registers control automatic signature generation. A signature can be generated for any part of the flash memory contents. The address range to be used for generation is defined by writing the start address to the signature start address register (FMSSTART) and the stop address to the signature stop address register (FMSSTOP. The start and stop addresses must be aligned to 128-bit boundaries and can be derived by dividing the byte address by 16.

Signature generation is started by setting the SIG_START bit in the FMSSTOP register. Setting the SIG_START bit is typically combined with the signature stop address in a single write.

Table 1025 and Table 1026 show the bit assignments in the FMSSTART and FMSSTOP registers respectively.

Table 1025.Flash Module Signature Start register (FMSSTART - 0x4008 4020) bit description

Bit	Symbol	Description	Reset Value
31:17	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
16:0	START	Signature generation start address (corresponds to AHB byte address bits[20:4]).	0

Table 1026.Flash Module Signature Stop register (FMSSTOP - 0x4008 4024) bit description

Bit	Symbol	Value	Description	Reset Value
31:18	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
17	SIG_START		Start control bit for signature generation.	0
		0	Signature generation is stopped	
		1	Initiate signature generation	
16:0	STOP		BIST stop address divided by 16 (corresponds to AHB byte address [20:4]).	0

45.11.1.2 Signature generation result registers

The signature generation result registers return the flash signature produced by the embedded signature generator. The 128-bit signature is reflected by the four registers FMSW0, FMSW1, FMSW2 and FMSW3.

The generated flash signature can be used to verify the flash memory contents. The generated signature can be compared with an expected signature and thus makes saves time and code space. The method for generating the signature is described in Section 45.11.2.

Table 1030 show bit assignment of the FMSW0 and FMSW1, FMSW2, FMSW3 registers respectively.

Table 1027.FMSW0 register bit description (FMSW0, address: 0x4008 402C)

Bit	Symbol	Description	Reset Value
31:0	SW0[31:0]	Word 0 of 128-bit signature (bits 31 to 0).	-

Table 1028.FMSW1 register bit description (FMSW1, address: 0x4008 4030)

Bit	Symbol	Description	Reset Value
31:0	SW1[63:32]	Word 1 of 128-bit signature (bits 63 to 32).	-

Table 1029.FMSW2 register bit description (FMSW2, address: 0x4008 4034)

Bit	Symbol	Description	Reset Value
31:0	SW2[95:64]	Word 2 of 128-bit signature (bits 95 to 64).	-

Table 1030.FMSW3 register bit description (FMSW3, address: 0x4008 4038)

Bit	Symbol	Description	Reset Value
31:0	SW3[127:96]	Word 3 of 128-bit signature (bits 127 to 96).	-

45.11.1.3 Flash Module Status register (FMSTAT - 0x0x4008 4FE0)

The read-only FMSTAT register provides a means of determining when signature generation has completed. Completion of signature generation can be checked by polling the SIG_DONE bit in FMSTAT. SIG_DONE should be cleared via the FMSTATCLR register before starting a signature generation operation, otherwise the status might indicate completion of a previous operation.

Table 1031.Flash module Status register (FMSTAT - 0x4008 4FE0) bit description

Bit	Symbol	Description	Reset Value
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
2	SIG_DONE	When 1, a previously started signature generation has completed. See FMSTATCLR register description for clearing this flag.	0
1:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

45.11.1.4 Flash Module Status Clear register (FMSTATCLR - 0x0x4008 4FE8)

The FMSTATCLR register is used to clear the signature generation completion flag.

Table 1032.Flash Module Status Clear register (FMSTATCLR - 0x0x4008 4FE8) bit description

Bit	Symbol	Description	Reset Value
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA
2	SIG_DONE_CLR	Writing a 1 to this bits clears the signature generation completion flag (SIG_DONE) in the FMSTAT register.	0
1:0	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

45.11.2 Algorithm and procedure for signature generation

Signature generation

A signature can be generated for any part of the flash contents. The address range to be used for signature generation is defined by writing the start address to the FMSSTART register, and the stop address to the FMSSTOP register.

The signature generation is started by writing a '1' to FMSSTOP.MISR_START. Starting the signature generation is typically combined with defining the stop address, which is done in another field FMSSTOP.FMSSTOP of the same register.

The time that the signature generation takes is proportional to the address range for which the signature is generated. Reading of the flash memory for signature generation uses a self-timed read mechanism and does not depend on any configurable timing settings for the flash. A safe estimation for the duration of the signature generation is:

Duration = int((60 / tcy) + 3) x (FMSSTOP - FMSSTART + 1)

When signature generation is triggered via software, the duration is in AHB clock cycles, and tcy is the time in ns for one AHB clock. The SIG_DONE bit in FMSTAT can be polled by software to determine when signature generation is complete.

If signature generation is triggered via JTAG, the duration is in JTAG tck cycles, and tcy is the time in ns for one JTAG clock. Polling the SIG_DONE bit in FMSTAT is not possible in this case.

After signature generation, a 128-bit signature can be read from the FMSW0 to FMSW3 registers. The 128-bit signature reflects the corrected data read from the flash. The 128-bit signature reflects flash parity bits and check bit values.

Content verification

The signature as it is read from the FMSW0 to FMSW3 registers must be equal to the reference signature. The algorithms to derive the reference signature is given in Figure 174.

```
sign = 0
FOR address = FMSTART.FMSTART TO FMSTOP.FMSTOP
{
        FOR i = 0 TO 126
                nextSign[i] = f_Q[address][i] XOR sign[i+1]
                nextSign[127] = f_Q[address][127] XOR sign[0] XOR sign[2] XOR
                        sign[27] XOR sign[29]
        sign = nextSign
}
signature128 = sign
```

Fig 174. Algorithm for generating a 128 bit signature

46.1 How to read this chapter

The debug functions are identical on all LPC43xx parts. The parallel trace port is not available on the 100-pin packages.

46.2 Basic configuration

- Select between Debug modes:
  - When the DBGEN pin is HIGH, the ARM debug mode is entered.
  - When the DBGEN pin is LOW, JTAG BSDL is entered.
- The core frequency must be lower than 80 MHz to use the SWO function.
- The ETM trace capture speed is limited by the trace pin properties. The maximum speed is 80 MHz when the EHS bit is set in the pin configuration register for the multiplexed trace pins (Table 123).
- For JTAG programming the pin RTC_ALARM must be LOW.
- Enable ETB SRAM in the CREG block (see Section 9.4.5).

46.3 Features

- Supports both standard JTAG and ARM Serial Wire Debug modes.
- Direct debug access to all memories, registers, and peripherals.
- No target resources are required for the debugging session.
- Trace port provides CPU instruction trace capability. Output can be via a 4-bit trace data port, or Serial Wire Viewer.
- Eight Breakpoints. Six instruction breakpoints that can also be used to remap instruction addresses for code patches. Two data comparators that can be used to remap addresses for patches to literal values.
- Four data Watchpoints that can also be used as trace triggers.
- Instrumentation Trace Macrocell allows additional software controlled trace.

46.4 Description

Debug and trace functions are integrated into the ARM Cortex-M4. Serial wire debug and trace functions are supported in addition to a standard JTAG debug and parallel trace functions. The ARM Cortex-M4 is configured to support up to eight breakpoints and four watchpoints.

Debugging with the LPC43xx defaults to JTAG. Once in the JTAG debug mode, the debug tool can switch to Serial Wire Debug mode.

Trace can be performed using either a 4-bit parallel interface or the Serial Wire Output.

46.5 Pin Description

Table 1033 to Table 1035 indicate the various pin functions related to debug and trace. Some of these functions share pins with other functions which therefore may not be used at the same time. Use of the JTAG port excludes use of Serial Wire Debug and Serial Wire Output. Use of the parallel trace requires five pins that may be part of the user application, limiting debug possibilities for those features. Trace using the Serial Wire Output does not have this limitation but the bandwidth is limited.

Table 1033.JTAG pin description

Pin Name	Type	Description
TCK	Input	**JTAG Test Clock.** This pin is the clock for debug logic when in the JTAG debug mode.
TMS	Input	**JTAG Test Mode Select.** The TMS pin selects the next state in the TAP state machine.
TDI	Input	**JTAG Test Data In.** This is the serial data input for the shift register.
TDO	Output	**JTAG Test Data Output.** This is the serial data output from the shift register. Data is shifted out of the device on the negative edge of the TCK signal.
$\overline{\text{TRST}}$	Input	**JTAG Test Reset.** The $\overline{\text{TRST}}$ pin can be used to reset the test logic within the debug logic.

Table 1034.Serial Wire Debug pin description

Pin Name	Type	Description
SWDCLK	Input	**Serial Wire Clock.** This pin is the clock for debug logic when in the Serial Wire Debug mode.
SWDIO	Input / Output	**Serial wire debug data input/output.** The SWDIO pin is used by an external debug tool to communicate with and control the Cortex-M4 CPU.
SWO	Output	**Serial Wire Output.** The SWO pin optionally provides data from the ITM and/or the ETM for an external debug tool to evaluate. **Remark:** The core frequency must be lower than 80 MHz to use the SWO.

Table 1035.Parallel Trace pin description

Pin Name	Type	Description
TRACECLK	Input	**Trace Clock.** This pin provides the sample clock for trace data on the TRACEDATA pins when tracing is enabled by an external debug tool.
TRACEDATA[3:0]	Output	**Trace Data bits 3 to 0.** These pins provide ETM trace data when tracing is enabled by an external debug tool. The debug tool can then interpret the compressed information and make it available to the user.

46.6 Debug connections

The LPC43xx supplies dedicated pins for JTAG and Serial Wire Debug (SWD). When a debug session is started, the part will be in JTAG debug mode. Once in debug mode, the debugger can switch the device to SWD mode.

Connections from a target board to the debugger can vary. Selecting a debug connector to add to a new board design depends on the debug tools that will be used.

46.6.1 ARM Standard JTAG connector (20-pin)

Figure 175 shows a standard JTAG connector. The ARM Standard JTAG Connector provides support for Serial Wire and JTAG interface modes in a 20-pin (0.1") connector. It can be used to access all SWD, SWV, and JTAG signals.

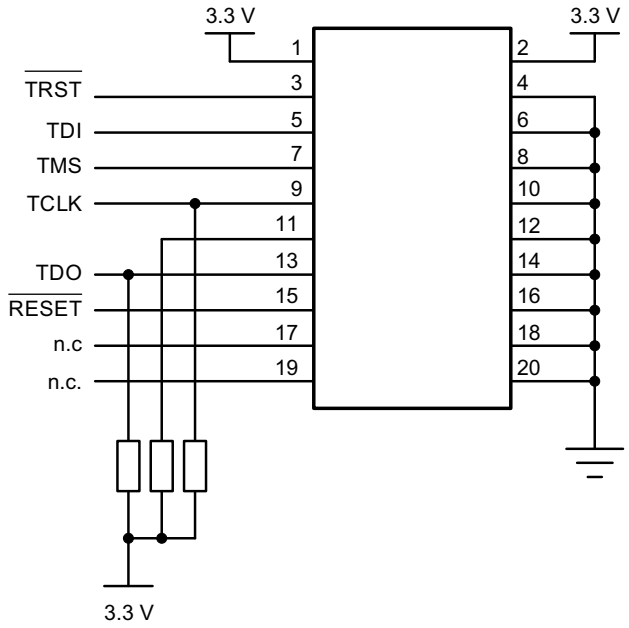


Fig 175. ARM Standard JTAG Connector

46.6.2 Cortex debug connector (10-pin)

Figure 176 shows the Cortex debug connector (10-pin). This connector provides support for Serial Wire and JTAG in a very small connector. The 10-pin Cortex debug connector can access to all SWD, SWV, and JTAG signals.

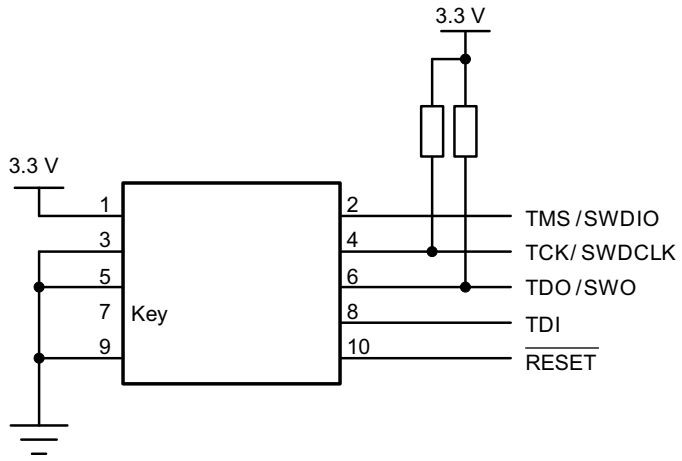


Fig 176. Cortex Debug Connector

46.6.3 Cortex Debug + ETM connector (20-pin)

If the debug trace feature will be used, there is also a debug-with-trace connector specification as shown in Figure 177. This small 20-pin (0.05") connector provides access to SWD, SWV, JTAG, and ETM (4-bit) signals.

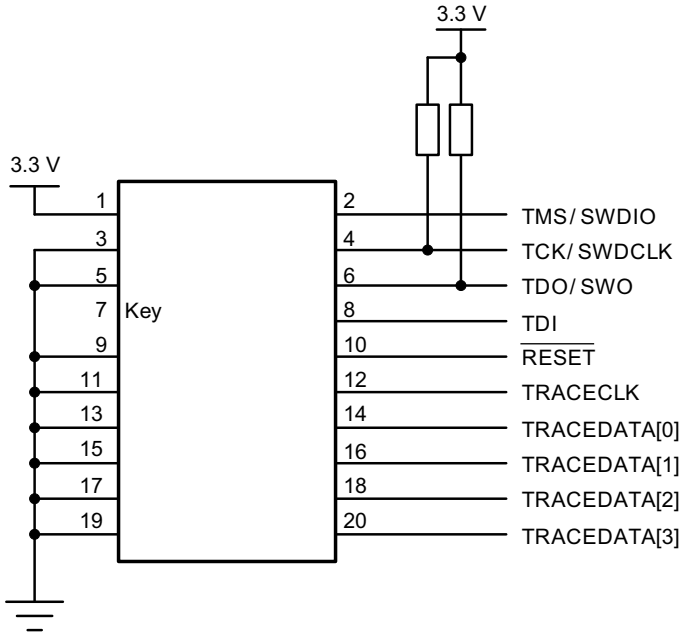


Fig 177. Cortex Debug + ETM Connector

46.7 Debug Notes

Important: The user should be aware of certain limitations during debugging. The most important is that, due to limitations of the Cortex-M4 integration, the LPC43xx cannot wake up in the usual manner from Deep Sleep and Power-down modes. It is recommended not to use these modes during debug.

Another issue is that debug mode changes the way in which reduced power modes are handled by the Cortex-M4 CPU. This causes power modes at the device level to be different from normal modes operation. These differences mean that power measurements should not be made while debugging, the results will be higher than during normal operation in an application.

During a debugging session, the System Tick Timer and the Repetitive Interrupt Timers are automatically stopped whenever the CPU is stopped. Other peripherals are not affected. If the Repetitive Interrupt Timer is configured such that its clock rate is lower than the CPU clock rate, the RIT may not increment predictably during some debug operations, such as single stepping.

Debugging is disabled if code read protection is enabled.

46.8 Debug memory re-mapping

Following chip reset, a portion of the Boot ROM is mapped to address 0 so that it will be automatically executed. The Boot ROM switches the map to point to 0x1000 0000 or 0x1C00 0000 (when booting from EMC) or 0x8000 0000 (when booting from SPIFI). Code execution can start from address 0x0000 0000 using the M4 memory mapping register (Table 39).

The register mapping is normally not transparent to the user. However, when a debugger halts CPU execution immediately following reset, the Boot ROM is still mapped to address 0 and can cause confusion. Ideally, the debugger should correct the mapping automatically in this case, so that the user does not need to deal with it.

46.9 JTAG TAP Identification

The JTAG TAP controller contains device ID that can be used by debugging software to identify the general type of device.

Table 1036.JTAG TAP identification

Mode	ID code
JTAG mode	0x4BA0 0477
SWD mode	0x2BA0 1477
Cortex-M0 co-processor	0x0BA0 1477

47.1 How to read this chapter

Details of the ARM Cortex-M0 and ARM Cortex-M4 specification can be found in the following documents:

- *Cortex-M4 Devices Generic User Guide*
- *Cortex-M0 Devices Generic User Guide*

47.2 Cortex-M4 instruction set summary

Table 1037 shows the Cortex-M4 instruction set with the following abbreviations to calculate the number of cycles:

- P = The number of cycles required for a pipeline refill. This ranges from 1 to 3 depending on the alignment and width of the target instruction, and whether the processor manages to speculate the address early.
- B = The number of cycles required to perform the barrier operation. For DSB and DMB, the minimum number of cycles is zero. For ISB, the minimum number of cycles is equivalent to the number required for a pipeline refill.
- N = The number of registers in the register list to be loaded or stored, including PC or LR.
- W = The number of cycles spent waiting for an appropriate event.

Table 1037.Cortex-M4 instruction set summary

Operation	Description	Assembler	Cycles
Move	Register	MOV Rd, <op2>	1
	16-bit immediate	MOVW Rd, #<imm>	1
	Immediate into top	MOVT Rd, #<imm>	1
	To PC	MOV PC, Rm	1 + P
Add	Add	ADD Rd, Rn, <op2>	1
	Add to PC	ADD PC, PC, Rm	1 + P
	Add with carry	ADC Rd, Rn, <op2>	1
	Form address	ADR Rd, <label>	1
Subtract	Subtract	SUB Rd, Rn, <op2>	1
	Subtract with borrow	SBC Rd, Rn, <op2>	1
	Reverse	RSB Rd, Rn, <op2>	1

Table 1037.Cortex-M4 instruction set summary

Operation	Description	Assembler	Cycles
Multiply	Multiply	MUL Rd, Rn, Rm	1
	Multiply accumulate	MLA Rd, Rn, Rm	1
	Multiply subtract	MLS Rd, Rn, Rm	1
	Long signed	SMULL RdLo, RdHi, Rn, Rm	1
	Long unsigned	UMULL RdLo, RdHi, Rn, Rm	1
	Long signed accumulate	SMLAL RdLo, RdHi, Rn, Rm	1
	Long unsigned accumulate	UMLAL RdLo, RdHi, Rn, Rm	1
Divide	Signed	SDIV Rd, Rn, Rm	2 to 12[1]
	Unsigned	UDIV Rd, Rn, Rm	2 to 12[1]
Saturate	Signed	SSAT Rd, #<imm>, <op2>	1
	Unsigned	USAT Rd, #<imm>, <op2>	1
Compare	Compare	CMP Rn, <op2>	1
Negative	CMN Rn, <op2>	1	
Logical	AND	AND Rd, Rn, <op2>	1
	Exclusive OR	EOR Rd, Rn, <op2>	1
	OR	ORR Rd, Rn, <op2>	1
	OR NOT	ORN Rd, Rn, <op2>	1
	Bit clear	BIC Rd, Rn, <op2>	1
	Move NOT	MVN Rd, <op2>	1
	AND test	TST Rn, <op2>	1
	Exclusive OR test	TEQ Rn, <op1>	
Shift	Logical shift left	LSL Rd, Rn, #<imm>	1
	Logical shift left	LSL Rd, Rn, Rs	1
	Logical shift right	LSR Rd, Rn, #<imm>	1
	Logical shift right	LSR Rd, Rn, Rs	1
	Arithmetic shift right	ASR Rd, Rn, #<imm>	1
	Arithmetic shift right	ASR Rd, Rn, Rs	1
Rotate	Rotate right	ROR Rd, Rn, #<imm>	1
	Rotate right	ROR Rd, Rn, Rs	1
	With extension	RRX Rd, Rn	1
Count	Leading zeroes	CLZ Rd, Rn	1

Table 1037.Cortex-M4 instruction set summary

Operation	Description	Assembler	Cycles
Load	Word	LDR Rd, [Rn, <op2>]	2[2]
	To PC	LDR PC, [Rn, <op2>]	2[2] + P
	Halfword	LDRH Rd, [Rn, <op2>]	2[2]
	Byte	LDRB Rd, [Rn, <op2>]	2[2]
	Signed halfword	LDRSH Rd, [Rn, <op2>]	2[2]
	Signed byte	LDRSB Rd, [Rn, <op2>]	2[2]
	User word	LDRT Rd, [Rn, #<imm>]	2[2]
	User halfword	LDRHT Rd, [Rn, #<imm>]	2[2]
	User byte	LDRBT Rd, [Rn, #<imm>]	2[2]
	User signed halfword	LDRSHT Rd, [Rn, #<imm>]	2[2]
	User signed byte	LDRSBT Rd, [Rn, #<imm>]	2[2]
	PC relative	LDR Rd,[PC, #<imm>]	2[2]
	Doubleword	LDRD Rd, Rd, [Rn, #<imm>]	1 + N
	Multiple	LDM Rn, {<reglist>}	1 + N
	Multiple including PC	LDM Rn, {<reglist>, PC}	1 + N + P
Store	Word	STR Rd, [Rn, <op2>]	2[2]
	Halfword	STRH Rd, [Rn, <op2>]	2[2]
	Byte	STRB Rd, [Rn, <op2>]	2[2]
	Signed halfword	STRSH Rd, [Rn, <op2>]	2[2]
	Signed byte	STRSB Rd, [Rn, <op2>]	2[2]
	User word	STRT Rd, [Rn, #<imm>]	2[2]
	User halfword	STRHT Rd, [Rn, #<imm>]	2[2]
	User byte	STRBT Rd, [Rn, #<imm>]	2[2]
	User signed halfword	STRSHT Rd, [Rn, #<imm>]	2[2]
	User signed byte	STRSBT Rd, [Rn, #<imm>]	2[2]
	Doubleword	STRD Rd, Rd, [Rn, #<imm>]	1 + N
	Multiple	STM Rn, {<reglist>}	1 + N
Push	Push	PUSH {<reglist>}	1 + N
	Push with link register	PUSH {<reglist>, LR}	1 + N
Pop	Pop	POP {<reglist>}	1 + N
	Pop and return	POP {<reglist>, PC}	1 + N + P
Semaphore	Load exclusive	LDREX Rd, [Rn, #<imm>]	2
	Load exclusive half	LDREXH Rd, [Rn]	2
	Load exclusive byte	LDREXB Rd, [Rn]	2
	Store exclusive	STREX Rd, Rt, [Rn, #<imm>]	2
	Store exclusive half	STREXH Rd, Rt, [Rn]	2
	Store exclusive byte	STREXB Rd, Rt, [Rn]	2
	Clear exclusive monitor	CLREX	1

Table 1037.Cortex-M4 instruction set summary

Operation	Description	Assembler	Cycles
Branch	Conditional	B<cc> <label>	1 or 1 + P[3]
	Unconditional	B <label>	1 + P
	With link	BL <label>	1 + P
	With exchange	BX Rm	1 + P
	With link and exchange	BLX Rm	1 + P
	Branch if zero	CBZ Rn, <label>	1 or 1 + P[3]
	Branch if non-zero	CBNZ Rn, <label>	1 or 1 + P[3]
	Byte table branch	TBB [Rn, Rm]	2 + P
	Halfword table branch	TBH [Rn, Rm, LSL#1]	2 + P
	State change	Supervisor call	SVC #<imm>
	If-then-else	IT... <cond>	1[4]
	Disable interrupts	CPSID <flags>	1 or 2
	Enable interrupts	CPSIE <flags>	1 or 2
	Read special register	MRS Rd, <specreg>	1 or 2
	Write special register	MSR <specreg>, Rn	1 or 2
	Breakpoint	BKPT #<imm>	-
Extend	Signed halfword to word	SXTH Rd, <op2>	1
	Signed byte to word	SXTB Rd, <op2>	1
	Unsigned halfword	UXTH Rd, <op2>	1
	Unsigned byte	UXTB Rd, <op2>	1
Bit field	Extract unsigned	UBFX Rd, Rn, #<imm>, #<imm>	1
	Extract signed	SBFX Rd, Rn, #<imm>, #<imm>	1
	Clear	BFC Rd, Rn, #<imm>, #<imm>	1
	Insert	BFI Rd, Rn, #<imm>, #<imm>	1
Reverse	Bytes in word	REV Rd, Rm	1
	Bytes in both halfwords	REV16 Rd, Rm	1
	Signed bottom halfword	REVSH Rd, Rm	1
	Bits in word	RBIT Rd, Rm	1
Hint	Send event	SEV	1
	Wait for event	WFE	1 + W
	Wait for interrupt	WFI	1 + W
	No operation	NOP	1
Barriers	Instruction synchronization	ISB	1 + B
	Data memory	DMB	1 + B
	Data synchronization	DSB <flags>	1 + B

[1] Division operations use early termination to minimize the number of cycles required based on the number of leading ones and zeroes in the input operands.

[2] Neighboring load and store single instructions can pipeline their address and data phases. This enables these instructions to complete in a single execution cycle.

[3] Conditional branch completes in a single cycle if the branch is not taken.

[4] An IT instruction can be folded onto a preceding 16-bit Thumb instruction, enabling execution in zero cycles.

Table 1038.Cortex-M4 DSP instruction set summary

Operation	Description	Assembler	Cycles
Multiply	32-bit multiply with 32-most-significant-bit accumulate	SMMLA	1
	32-bit multiply with 32-most-significant-bit subtract	SMMLS	1
	32-bit multiply returning 32-most-significant-bits	SMMUL	1
	32-bit multiply with rounded 32-most-significant-bit accumulate	SMMLAR	1
	32-bit multiply with rounded 32-most-significant-bit subtract	SMMLSR	1
	32-bit multiply returning rounded 32-most-significant-bits	SMMULR	1
Signed Multiply	Q setting 16-bit signed multiply with 32-bit accumulate, bottom by bottom	SMLABB	1
	Q setting 16-bit signed multiply with 32-bit accumulate, bottom by top	SMLABT	1
	16-bit signed multiply with 64-bit accumulate, bottom by bottom	SMLALBB	1
	16-bit signed multiply with 64-bit accumulate, bottom by top	SMLALBT	1
	Dual 16-bit signed multiply with single 64-bit accumulator	SMLALD{X}	1
	16-bit signed multiply with 64-bit accumulate, top by bottom	SMLALTB	1
	16-bit signed multiply with 64-bit accumulate, top by top	SMLALTT	1
	16-bit signed multiply yielding 32-bit result, bottom by bottom	SMULBB	1
	16-bit signed multiply yielding 32-bit result, bottom by top	SMULBT	1
	16-bit signed multiply yielding 32-bit result, top by bottom	SMULTB	1
	16-bit signed multiply yielding 32-bit result, top by bottom	SMULTT	1
	16-bit by 32-bit signed multiply returning 32-most-significant-bits, bottom	SMULWB	1
	16-bit by 32-bit signed multiply returning 32-most-significant-bits, top	SMULWT	1
	Dual 16-bit signed multiply returning difference	SMUSD{X}	1
	Q setting 16-bit signed multiply with 32-bit accumulate, top by bottom	SMLATB	1
	Q setting 16-bit signed multiply with 32-bit accumulate, top by top	SMLATT	1
	Q setting dual 16-bit signed multiply with single 32-bit accumulator	SMLAD{X}	1
	Q setting 16-bit by 32-bit signed multiply with 32-bit accumulate, bottom	SMLAWB	1
	Q setting 16-bit by 32-bit signed multiply with 32-bit accumulate, top	SMLAWT	1
	Q setting dual 16-bit signed multiply subtract with 32-bit accumulate	SMLSD{X}	1
	Q setting dual 16-bit signed multiply subtract with 64-bit accumulate	SMLSLD{X}	1
	Q setting sum of dual 16-bit signed multiply	SMUAD{X}	1
Unsigned Multiply	32-bit unsigned multiply with double 32-bit accumulation yielding 64-bit result	UMAAL	1
Saturate	Q setting dual 16-bit saturate	SSAT16	1
	Q setting dual 16-bit unsigned saturate	USAT16	1

Table 1038.Cortex-M4 DSP instruction set summary

Operation	Description	Assembler	Cycles
Packing and Unpacking	Pack half word top with shifted bottom	PKHTB	1
	Pack half word bottom with shifted top	PKHBT	1
	Extract 8-bits and sign extend to 32-bits	SXTB	1
	Dual extract 8-bits and sign extend each to 16-bits	SXTB16	1
	Extract 16-bits and sign extend to 32-bits	SXTH	1
	Extract 8-bits and zero-extend to 32-bits	UXTB	1
	Dual extract 8-bits and zero-extend to 16-bits	UXTB16	1
	Extract 16-bits and zero-extend to 32-bits	UXTH	1
	Extract 8-bit to 32-bit unsigned addition	UXTAB	1
	Dual extracted 8-bit to 16-bit unsigned addition	UXTAB16	1
	Extracted 16-bit to 32-bit unsigned addition	UXTAH	1
	Extracted 8-bit to 32-bit signed addition	SXTAB	1
	Dual extracted 8-bit to 16-bit signed addition	SXTAB16	1
	Extracted 16-bit to 32-bit signed addition	SXTAH	1
Miscellaneous Data Processing	Select bytes based on GE bits	SEL	1
	Unsigned sum of quad 8-bit unsigned absolute difference	USAD8	1
	Unsigned sum of quad 8-bit unsigned absolute difference with 32-bit accumulate	USADA8	1
Addition	Dual 16-bit unsigned saturating addition	UQADD16	1
	Quad 8-bit unsigned saturating addition	UQADD8	1
	Q setting saturating add	QADD	1
	Q setting dual 16-bit saturating add	QADD16	1
	Q setting quad 8-bit saturating add	QADD8	1
	Q setting saturating double and add	QDADD	1
	GE setting quad 8-bit signed addition	SADD8	1
	GE setting dual 16-bit signed addition	SADD16	1
	Dual 16-bit signed addition with halved results	SHADD16	1
	Quad 8-bit signed addition with halved results	SHADD8	1
	GE setting dual 16-bit unsigned addition	UADD16	1
	GE setting quad 8-bit unsigned addition	UADD8	1
	Dual 16-bit unsigned addition with halved results	UHADD16	1
	Quad 8-bit unsigned addition with halved results	UHADD8	1

Table 1038.Cortex-M4 DSP instruction set summary

Operation	Description	Assembler	Cycles
Subtraction	Q setting saturating double and subtract	QDSUB	1
	Dual 16-bit unsigned saturating subtraction	UQSUB16	1
	Quad 8-bit unsigned saturating subtraction	UQSUB8	1
	Q setting saturating subtract	QSUB	1
	Q setting dual 16-bit saturating subtract	QSUB16	1
	Q setting quad 8-bit saturating subtract	QSUB8	1
	Dual 16-bit signed subtraction with halved results	SHSUB16	1
	Quad 8-bit signed subtraction with halved results	SHSUB8	1
	GE setting dual 16-bit signed subtraction	SSUB16	1
	GE setting quad 8-bit signed subtraction	SSUB8	1
	Dual 16-bit unsigned subtraction with halved results	UHSUB16	1
	Quad 8-bit unsigned subtraction with halved results	UHSUB8	1
	GE setting dual 16-bit unsigned subtract	USUB16	1
	GE setting quad 8-bit unsigned subtract	USUB8	1
Parallel Addition and Subtraction	Dual 16-bit unsigned saturating addition and subtraction with exchange	UQASX	1
	Dual 16-bit unsigned saturating subtraction and addition with exchange	UQSAX	1
	GE setting dual 16-bit addition and subtraction with exchange	SASX	1
	Q setting dual 16-bit add and subtract with exchange	QASX	1
	Q setting dual 16-bit subtract and add with exchange	QSAX	1
	Dual 16-bit signed addition and subtraction with halved results	SHASX	1
	Dual 16-bit signed subtraction and addition with halved results	SHSAX	1
	GE setting dual 16-bit signed subtraction and addition with exchange	SSAX	1
	GE setting dual 16-bit unsigned addition and subtraction with exchange	UASX	1
	Dual 16-bit unsigned addition and subtraction with halved results and exchange	UHASX	1
	Dual 16-bit unsigned subtraction and addition with halved results and exchange	UHSAX	1
	GE setting dual 16-bit unsigned subtract and add with exchange	USAX	1

47.3 Cortex-M0 instruction set summary

Table 1039.Cortex M0- instruction set summary

Operation	Description	Assembler	Cycles
Move	8-bit immediate	MOVS Rd, #<imm>	1
	Lo to Lo	MOVS Rd, Rm	1
	Any to Any	MOV Rd, Rm	1
	Any to PC	MOV PC, Rm	3
Add	3-bit immediate	ADDS Rd, Rn, #<imm>	1
	All registers Lo	ADDS Rd, Rn, Rm	1
	Any to Any	ADD Rd, Rd, Rm	1
	Any to PC	ADD PC, PC, Rm	3
	8-bit immediate	ADDS Rd, Rd, #<imm>	1
	With carry	ADCS Rd, Rd, Rm	1
	Immediate to SP	ADD SP, SP, #<imm>	1
	From address from SP	ADD Rd, SP, #<imm>	1
	From address from PC	ADR Rd, <label>	1
Subtract	Lo and Lo	SUBS Rd, Rn, Rm	1
	3-bit immediate	SUBS Rd, Rn, #<imm>	1
	8-bit immediate	SUBS Rd, Rd, #<imm>	1
	With carry	SBCS Rd, Rd, Rm	1
	Immediate from SP	SUB SP, SP, #<imm>	1
	Negate	RSBS Rd, Rn, #0	1
Multiply	Multiply	MULS Rd, Rm, Rd	32
Compare	Compare	CMP Rn, Rm	1
	Negative	CMN Rn, Rm	1
	Immediate	CMP Rn, #<imm>	1
Logical	AND	ANDS Rd, Rd, Rm	1
	Exclusive OR	EORS Rd, Rd, Rm	1
	OR	ORRS Rd, Rd, Rm	1
	Bit clear	BICS Rd, Rd, Rm	1
	Move NOT	MVNS Rd, Rm	1
	AND test	TST Rn, Rm	1
Shift	Logical shift left by immediate	LSLS Rd, Rm, #<shift>	1
	Logical shift left by register	LSLS Rd, Rd, Rs	1
	Logical shift right by immediate	LSRS Rd, Rm, #<shift>	1
	Logical shift right by register	LSRS Rd, Rd, Rs	1
	Arithmetic shift right	ASRS Rd, Rm, #<shift>	1
	Arithmetic shift right by register	ASRS Rd, Rd, Rs	1
Rotate	Rotate right by register	RORS Rd, Rd, Rs	1

Table 1039.Cortex M0- instruction set summary

Operation	Description	Assembler	Cycles
Load	Word, immediate offset	LDR Rd, [Rn, #<imm>]	2
	Halfword, immediate offset	LDRH Rd, [Rn, #<imm>]	2
	Byte, immediate offset	LDRB Rd, [Rn, #<imm>]	2
	Word, register offset	LDR Rd, [Rn, Rm]	2
	Halfword, register offset	LDRH Rd, [Rn, Rm]	2
	Signed halfword, register offset	LDRSH Rd, [Rn, Rm]	2
	Byte, register offset	LDRB Rd, [Rn, Rm]	2
	Signed byte, register offset	LDRSB Rd, [Rn, Rm]	2
	PC-relative	LDR Rd, <label>	2
	SP-relative	LDR Rd, [SP, #<imm>]	2
	Multiple, excluding base	LDM Rn!, {<loreglist>}	1 + N[1]
	Multiple, including base	LDM Rn, {<loreglist>}	1 + N[1]
Store	Word, immediate offset	STR Rd, [Rn, #<imm>]	2
	Halfword, immediate offset	STRH Rd, [Rn, #<imm>]	2
	Byte, immediate offset	STRB Rd, [Rn, #<imm>]	2
	Word, register offset	STR Rd, [Rn, Rm]	2
	Halfword, register offset	STRH Rd, [Rn, Rm]	2
	Byte, register offset	STRB Rd, [Rn, Rm]	2
	SP-relative	STR Rd, [SP, #<imm>]	2
	Multiple	STM Rn!, {<loreglist>}	1 + N[1]
Push	Push	PUSH {<loreglist>}	1 + N[1]
	Push with link register	PUSH {<loreglist>, LR}	1 + N[1]
Pop	Pop	POP {<loreglist>}	1 + N[1]
	Pop and return	POP {<loreglist>, PC}	4 + N[2]
Branch	Conditional	B<cc> <label>	1 or 3[3]
	Unconditional	B <label>	3
	With link	BL <label>	4
	With exchange	BX Rm	3
	With link and exchange	BLX Rm	3
Extend	Signed halfword to word	SXTH Rd, Rm	1
	Signed byte to word	SXTB Rd, Rm	1
	Unsigned halfword	UXTH Rd, Rm	1
	Unsigned byte	UXTB Rd, Rm	1
Reverse	Bytes in word	REV Rd, Rm	1
	Bytes in both halfwords	REV16 Rd, Rm	1
	Signed bottom half word	REVSH Rd, Rm	1
State change	Supervisor Call	SVC <imm>	-[4]
	Disable interrupts	CPSID i	1
	Enable interrupts	CPSIE i	1
	Read special register	MRS Rd, <specreg>	4
	Write special register	MSR <specreg>, Rn	4

Table 1039.Cortex M0- instruction set summary

Operation	Description	Assembler	Cycles
Hint	Send event	SEV	1
	Wait for event	WFE	2[5]
	Wait for interrupt	WFI	2[5]
	Yield	YIELD[6]	1
	No operation	NOP	1
Barriers	Instruction synchronization	ISB	4
	Data memory	DMB	4
	Data synchronization	DSB	4

[1] N is the number of elements.

[2] N is the number of elements in the stack-pop list including PC and assumes load or store does not generate a HardFault exception.

[3] 3 if taken, 1 if not taken.

[4] Cycle count depends on core and debug configuration.

[5] Excludes time spend waiting for an interrupt or event.

[6] Executes as NOP.

48.1 Abbreviations

Table 1040.Abbreviations

Acronym	Description
ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
AHB	Advanced High-performance Bus
APB	Advanced Peripheral Bus
API	Application Programming Interface
BOD	BrownOut Detection
CAN	Controller Area Network
CMAC	Cipher-based Message Authentication Code
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DAC	Digital-to-Analog Converter
DC-DC	Direct Current-to-Direct Current
DMA	Direct Memory Access
GPIO	General Purpose Input/Output
IRC	Internal RC
IrDA	Infrared Data Association
JTAG	Joint Test Action Group
LCD	Liquid Crystal Display
LSB	Least Significant Bit
MAC	Media Access Control
MCU	MicroController Unit
MIIM	Media Independent Interface Management
n.c.	not connected
OHCI	Open Host Controller Interface
OTG	On-The-Go
PHY	Physical Layer
PLL	Phase-Locked Loop
PMC	Power Mode Control
PWM	Pulse Width Modulator
RIT	Repetitive Interrupt Timer
RMII	Reduced Media Independent Interface
SDRAM	Synchronous Dynamic Random Access Memory
SIMD	Single Instruction Multiple Data
SPI	Serial Peripheral Interface
SSI	Serial Synchronous Interface
SSP	Synchronous Serial Port
TCP/IP	Transmission Control Protocol/Internet Protocol

Table 1040. Abbreviations *…continued*

Acronym	Description
TTL	Transistor-Transistor Logic
UART	Universal Asynchronous Receiver/Transmitter
ULPI	UTMI+ Low Pin Interface
USART	Universal Synchronous Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
UTMI	USB2.0 Transceiver Macrocell Interface

48.2 Legal information

48.2.1 User manual status

Document status[1]	Product status[2]	Definition
Objective user manual	Development	This document contains data from the objective specification for product development.
Preliminary user manual	Qualification	This document contains data from the preliminary specification.
Product user manual	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.

[2] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL http://www.nxp.com.

48.2.2 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

48.2.3 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

48.2.4 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

I²C-bus — logo is a trademark of NXP B.V.

48.3 Tables

48.4 Figures

48.5 Contents

Chapter 8: LPC43xx Event router

Chapter 9: LPC43xx Configuration Registers (CREG)

Chapter 10: LPC43xx Power Management Controller (PMC)

Chapter 11: LPC43xx Clock Generation Unit (CGU)

Chapter 12: LPC43xx Clock Control Unit (CCU)

Chapter 13: LPC43xx Reset Generation Unit (RGU)

Chapter 14: LPC43xx Pin configuration

Chapter 15: LPC43xx System Control Unit (SCU)/ IO configuration

Chapter 16: LPC43xx Global Input Multiplexer Array (GIMA)

Chapter 17: LPC43xx GPIO

Chapter 18: LPC43xx Serial GPIO (SGPIO)

Chapter 19: LPC43xx General Purpose DMA (GPDMA) controller

Chapter 20: LPC43xx SD/MMC interface

Chapter 21: LPC43xx External Memory Controller (EMC)

Chapter 22: LPC43xx SPI Flash Interface (SPIFI)

Chapter 23: LPC43xx USB0 Host/Device/OTG controller

Chapter 24: LPC43xx USB1 Host/Device controller

Chapter 25: LPC43xx USB API

Chapter 26: LPC43xx Ethernet

Chapter 27: LPC43xx LCD

Chapter 28: LPC43xx State Configurable Timer (SCT)

Chapter 29: LPC43xx Timer0/1/2/3

Chapter 30: LPC43xx Motor Control PWM (MOTOCONPWM)

Chapter 31: LPC43xx Quadrature Encoder Interface (QEI)

Chapter 32: LPC43xx Repetitive Interrupt Timer (RIT)

Chapter 33: LPC43xx Alarm timer

Chapter 34: LPC43xx Windowed Watchdog timer (WWDT)

Chapter 35: LPC43xx Real-Time Clock (RTC)

Chapter 36: LPC43xx USART0_2_3

Chapter 37: LPC43xx UART1

Chapter 38: LPC43xx SSP0/1

Chapter 39: LPC43xx SPI

Chapter 40: LPC43xx I2S interface

Chapter 41: LPC43xx C_CAN

Chapter 42: LPC43xx I2C-bus interface

Chapter 43: LPC43xx 10-bit ADC0/1

Chapter 44: LPC43xx DAC

Chapter 45: LPC43xx flash programming/ISP and IAP

Chapter 46: LPC43xx JTAG, Serial Wire Debug (SWD), and trace functions

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 12 December 2011
Document identifier: UM10503