# DATA SHEET

# P90CL301BFH (C100)
## Low voltage 16-bit microcontroller

Preliminary specification
File under Integrated Circuits, IC17

1996 Dec 11

Philips
Semiconductors

PHILIPS

**CONTENTS**



## 1 FEATURES

- Fully 68000 software compatible
- Static design with 32-bit internal structure
- Power saving modes: Power-down, Standby and Idle mode
- External clock input: 27 MHz at 2.7 V
- Single supply voltage of 2.7 to 3.6 V; down to 1.8 V for RAM retention
- 68000 compatible bus interface
- Intel 8051 compatible bus interface
- 16 Mbytes program/data address range
- 8 programmable chip-selects
- Dynamic bus sizing, 16 or 8-bit memory bus port size
- 56 powerful instruction types:
  - 5 basic data types, and
  - 14 addressing modes
- 7 programmable interrupt inputs:
  - a Non-Maskable Interrupt input (NMIN)
  - 14 auto-vectored interrupts and 7 interrupt priority levels
- 24 port pins (multiplexed with other functions)
- 2 UART serial interfaces; an independent baud rate generator with two programmable outputs (UART0 and UART1)
- UART queue with maximum 256 bytes
- $I^2C$-bus serial interface 100 kbaud
- 2 timer arrays including:
  - two 16-bit reference counters and 8-bit programmable prescalers
  - six 16-bit match/capture registers with equality comparators
- Watchdog Timer with 21-bit resolution
- Two 8-bit Pulse Width Modulation (PWM) outputs with 8-bit prescaler
- Four 8-bit Analog-to-Digital Converter (ADC) inputs with Power-down mode
- 512 bytes RAM on-chip
- On-Circuit Emulation (ONCE) mode and internal Test-ROM (256 bytes) for on-board testing
- 80-pin LQFP package
- Temperature range −40 to +85 °C
- 0.5 micron CMOS low voltage technology.

## 2 DESCRIPTION

The P90CL301BFH is a highly integrated low-voltage 16/32-bit microcontroller especially suitable for digital mobile systems such as GSM, DCS1900, IS54/95 and other applications requiring low voltage, low power consumption and high computing power. It is fully software compatible with the 68000.

The P90CL301BFH optimizes system cost by providing both standard as well as advanced peripheral functions on-chip. The P90CL301BFH has a full static design and special Idle, Standby and Power-down modes which allow further reduction of the total system power consumption. An 80-pin LQFP package dramatically reduces system size requirements.

### 2.1 Compatibility between P90CL301AFH and P90CL301BFH

For functional compatibility between P90CL301AFH (SAC1 process) and P90CL301BFH (C100 process), the following points should be considered when using the P90CL301BFH:

- **Wake-up**; to wake-up the processor from Power-down mode via the activation of an external SPn pin, it is necessary to enable the interrupt mode first by setting the corresponding bit in the SPCON register.
- **SYSCON register**; for the P90CL301AFH bits 11 to 15 in the SYSCON register should not be set in order to keep additional functionality in the P90CL301BFH inactive.

## 3 ORDERING INFORMATION

| TYPE NUMBER | PACKAGE | | | TEMPERATURE RANGE (°C) |
|---|---|---|---|---|
| | NAME | DESCRIPTION | VERSION | |
| P90CL301BFH | LQFP80 | plastic low profile quad flat package; 80 leads; body 12 × 12 × 1.4 mm | SOT315-1 | −40 to +85 |

## 4 BLOCK DIAGRAM

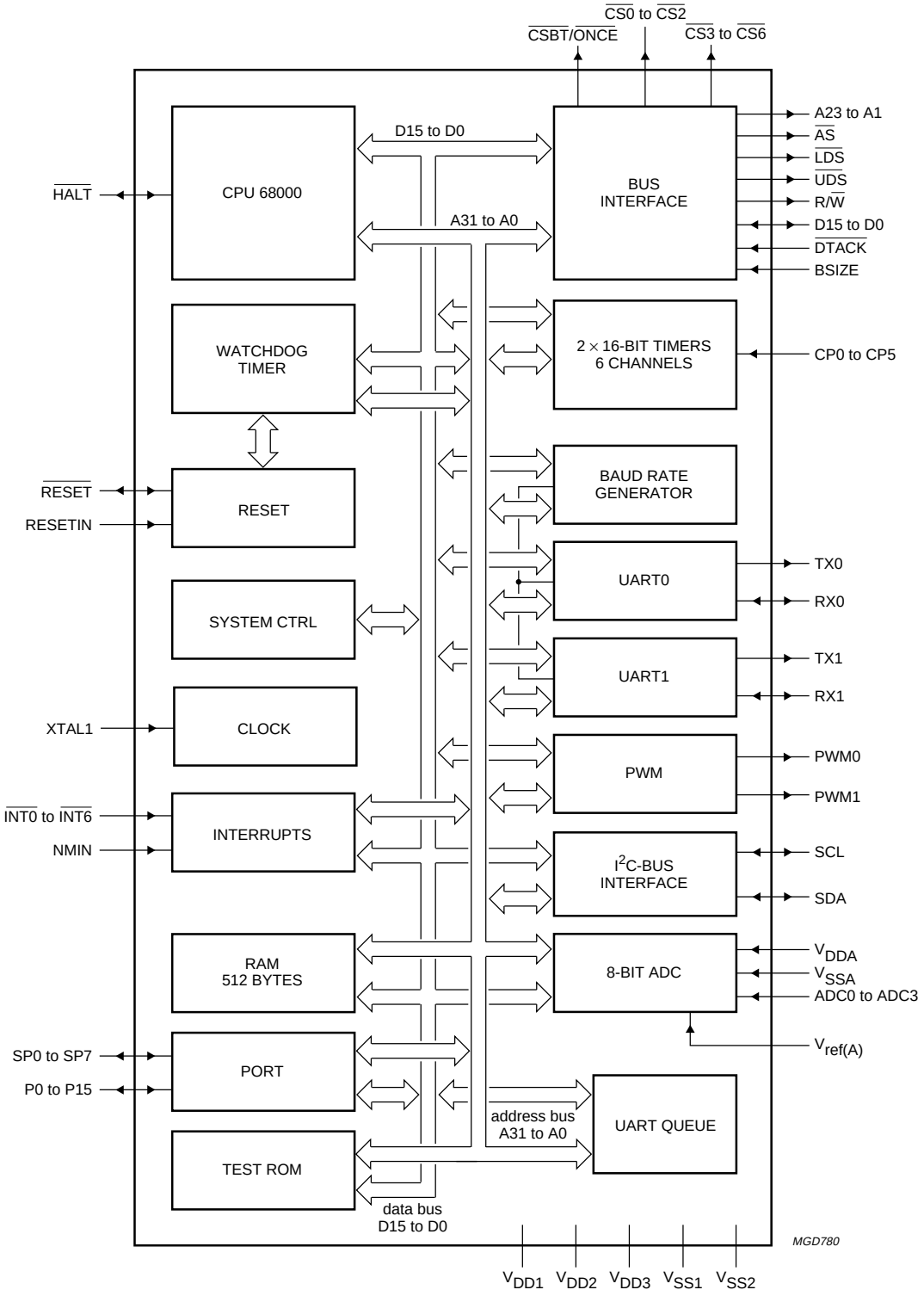



Fig.1 P90CL301BFH block diagram.

## 5 PINNING INFORMATION

### 5.1 Pinning

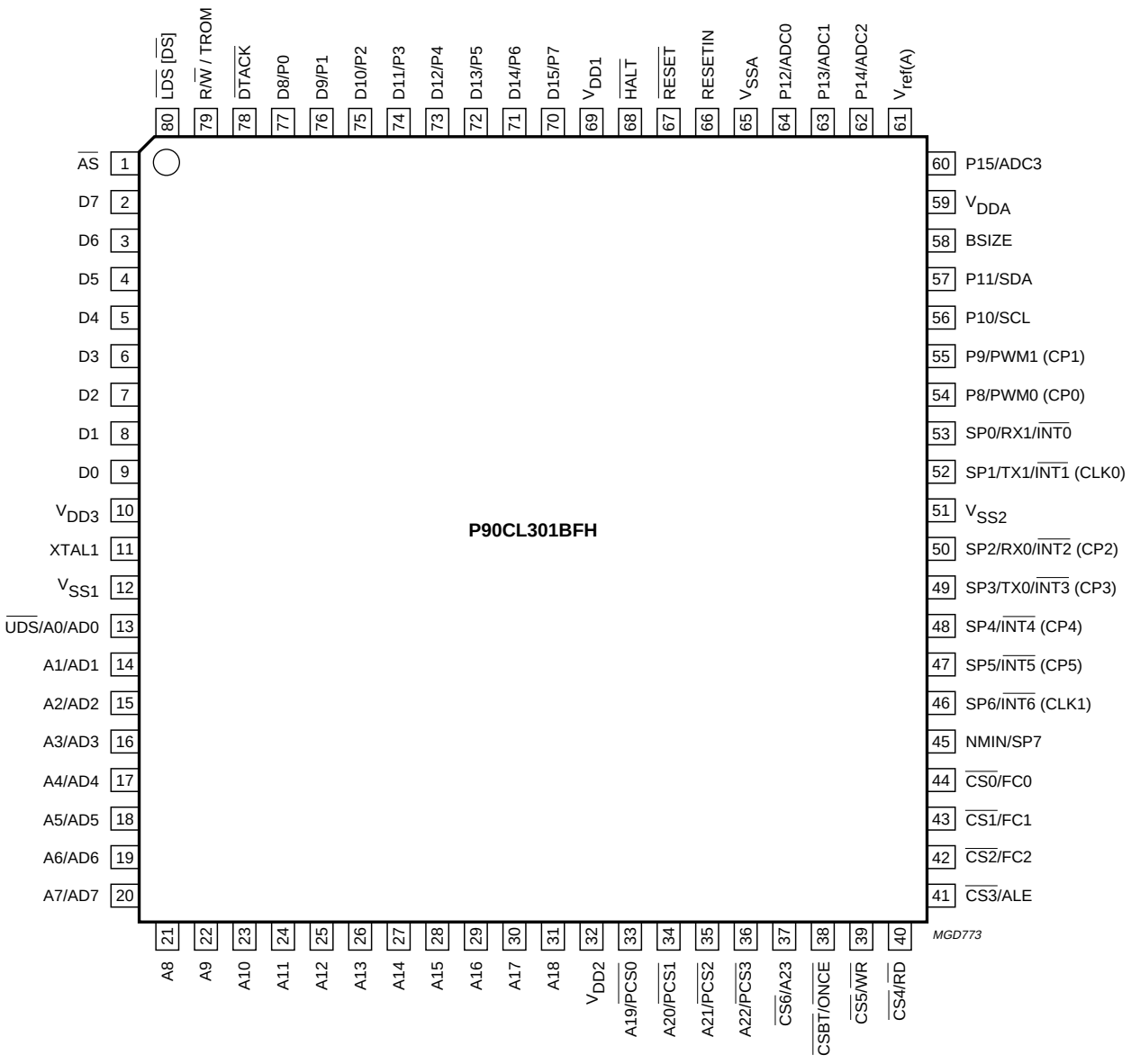



Fig.2 Pinning diagram of the P90CL301BFH (LQFP80).

### 5.2 Pin description

**Table 1** Pin description for the P90CL301BFH

| SYMBOL[1] | PIN | DESCRIPTION |
|---|---|---|
| $\overline{AS}$ | 1 | address strobe |
| D7 to D0 | 2 to 9 | lower 8-bits of data bus |
| $V_{DD3}$ | 10 | supply voltage; third pin |
| XTAL1 | 11 | external clock input |
| $V_{SS1}$ | 12 | ground; first pin |
| $\overline{UDS}$/A0/AD0 | 13 | upper data strobe or LSB of address bus or LSB of 8051 address/data |
| A1/AD1 to A7/AD7 | 14 to 20 | lower 7-bits of the 68000 address bus or lower 7-bits of the 8051 bus |
| A8 to A18 | 21 to 31 | upper 11-bits of the 68000 address bus |
| $V_{DD2}$ | 32 | supply voltage; second pin |
| A19/$\overline{PCS0}$ to A22/$\overline{PCS3}$ | 33 to 36 | upper 4-bits of the address bus or 8051 bus chip-select |
| $\overline{CS6}$/A23 | 37 | chip-select 6 or address bit 23 |
| $\overline{CSBT}$/$\overline{ONCE}$ | 38 | chip-select boot or ONCE mode forced input |
| $\overline{CS5}$/$\overline{WR}$ | 39 | chip-select 5 or 8051 bus write strobe |
| $\overline{CS4}$/$\overline{RD}$ | 40 | chip-select 4 or 8051 bus read strobe |
| $\overline{CS3}$/ALE | 41 | chip-select 3 or 8051 bus address latch |
| $\overline{CS2}$/FC2 to $\overline{CS0}$/FC0 | 42 to 44 | chip-select 2 to 0 or data bus function code 2 to 0 |
| NMIN/SP7 | 45 | Non-Maskable Interrupt or second port pin (bit 7) |
| SP6/$\overline{INT6}$ (CLK1) | 46 | second port pin (bit 6) external interrupt input 6 (external clock of timer 1) |
| SP5/$\overline{INT5}$ (CP5) | 47 | second port pin (bit 5) or external interrupt input 5 (Timer 1 capture input 5) |
| SP4/$\overline{INT4}$ (CP4) | 48 | second port pin (bit 4) or external interrupt input 4 (Timer 1 capture input 4) |
| SP3/TX0/$\overline{INT3}$ (CP3) | 49 | second port pin (bit 3) or Transmit data for UART0 or external interrupt input 3 (Timer 1 capture input 3) |
| SP2/RX0/$\overline{INT2}$ (CP2) | 50 | second port pin (bit 2) or Receive data for UART0 or external interrupt input 2 (Timer 0 capture input 2) |
| $V_{SS2}$ | 51 | ground; second pin |
| SP1/TX1/$\overline{INT1}$ (CLK0) | 52 | second port pin (bit 1) or transmit data for UART1 or external interrupt input 1 (external clock of Timer 0) |
| SP0/RX1/$\overline{INT0}$ | 53 | second port pin (bit 0) or receive data for UART1 or external interrupt input 0 |
| P8/PWM0 (CP0) | 54 | port pin (bit 8) or PWM0 output (Timer 0 capture input 0) |
| P9/PWM1 (CP1) | 55 | port pin (bit 9) or PWM1 output (Timer 0 capture input 1) |
| P10/SCL | 56 | port pin (bit 10) or $I^2C$-bus Serial Clock. |
| P11/SDA | 57 | port pin (bit 11) or $I^2C$-bus Serial Data. |
| BSIZE | 58 | data bus size; 8 or 16-bit wide |
| $V_{DDA}$ | 59 | ADC supply voltage |
| P15/ADC3 | 60 | port pin (bit 15) or ADC input 3 |
| $V_{ref(A)}$ | 61 | ADC reference voltage |
| P14/ADC2 to P12/ADC0 | 62 to 64 | port pin (bit 14 to bit 12) or ADC inputs 2 to 0 |
| $V_{SSA}$ | 65 | ADC ground |
| RESETIN | 66 | external Power-on-reset input |

| SYMBOL[1] | PIN | DESCRIPTION |
|---|---|---|
| $\overline{\text{RESET}}$ | 67 | reset (bidirectional) |
| $\overline{\text{HALT}}$ | 68 | halt (bidirectional) |
| $V_{DD1}$ | 69 | supply voltage; first pin |
| D15/P7 to D8/P0 | 70 to 77 | upper 8-bits of data bus or 8-bit Port 7 to Port 0; the selected function after reset is defined by pin BSIZE |
| $\overline{\text{DTACK}}$ | 78 | data transfer acknowledge |
| R/$\overline{\text{W}}$ / TROM | 79 | read/write bus control or Test-ROM forced input |
| $\overline{\text{LDS}}$ [$\overline{\text{DS}}$] | 80 | lower data strobe [word data strobe] |

**Note**

1. The following notation is used to describe the multiple pin definitions:
   a) Function1/Function2/Function3: multiplexed functions on the same pin. During and after reset the Function1 is selected.
   b) Function1 (Function2): function done in parallel.
   c) Function1 [Function2]: equivalent function.

## 6 SYSTEM CONTROL

### 6.1 Memory organization

The maximum external address space of the controller is 16 Mbytes. It can be partitioned into five address spaces. These address spaces are designated as either User or Supervisor space and as either Program or Data space or as interrupt acknowledge.

For slow memories the CPU can be programmed to insert a number of wait states. This is done via the eight Chip-select Control Registers CS0N to CS7N; further to be denoted as CSnN, where n = 0 to 7. The number of inserted wait states can vary from 0 to 6, or wait states are inserted until the $\overline{\text{DTACK}}$ is pulled LOW by the external address decoding circuitry. If $\overline{\text{DTACK}}$ is asserted continuously, the P90CL301BFH will run without wait states using bus cycles of three or four clock periods depending on the state of the FBC bit in the SYSCON register.

#### 6.1.1 MEMORY MAP

The memory address space is divided as shown in Table 2; short addressing space with A31 to A15 = 1.

**Table 2** Memory address space

| ADDRESS (HEX) | DESCRIPTION |
|---|---|
| 0000 0000 to 00FF FFFF | external 16 Mbytes memory |
| 0100 0000 to 8000 FFFF | not used |
| 8001 0000 to 8001 FFFF | off-chip 64 kbytes on 8051 bus |
| 8002 0000 to FFFF 7FFF | not used |
| FFFF 8000 to FFFF 8AFF | internal registers |
| FFFF 8B00 to FFFF 8FFF | not used |
| FFFF 9000 to FFFF 91FF | internal 512 bytes RAM |
| FFFF 9200 to FFFF BFFF | not used |
| FFFF C000 to FFFF C0FF | internal 256 bytes Test-ROM |
| FFFF C100 to FFFF FFFF | not used |

### 6.2 Programmable chip-select

In order to reduce the external components associated with memory interface, the P90CL301BFH provides 8 programmable chip-selects. A specific chip-select $\overline{\text{CSBT}}$ provides default reset values to support a bootstrap operation.

Each chip-select can be programmed with:

- A base address (A23 to A19)
- A memory bank width of 512 kbytes, 1, 2, 4 or 8 Mbytes memory size
- A number of wait states (0 to 6 states, or wait for $\overline{\text{DTACK}}$) to adapt the bus cycle to the memory cycle time.

Chip-selects can be synchronized with read, write, or both read and write, either Address strobe or Data strobe. They can also be programmed to address low byte, high byte or word.

Each chip-select is controlled by a control register CSnN (n = 0 to 7). The control registers are described in Table 3 to 7.

The RESET instruction does not affect the contents of the CSnN registers.

Register CS7N corresponds to register $\overline{\text{CSBT}}$ (address FFFF 8A0EH). After reset $\overline{\text{CSBT}}$ is programmed with a block size of 8 Mbytes with:

- A19 to A23 at logic 0
- M19 to M22 at logic 1
- 6 wait states
- read only mode.

The other chip-selects are held HIGH and will be activated after initialization of their control registers.

When programmed in reduced access mode (read only, write only, low byte, high byte), the wait states are generated internally and if there is any access-violation when the bit WD in the SYSCON register is set to a logic 1 (time-out), the processor will execute a bus error after the time-out delay.

### 6.2.1 Chip Select Control Registers (CS0N to CS7N)

**Table 3** Chip Select Control Registers CS0N to CS7N (address FFFF 8A00H to FFFF 8A0CH)

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M22 | M21 | M20 | M19 | RW1 | RW0 | MD1 | MD0 | A23 | A22 | A21 | A20 | A19 | WS2 | WS1 | WS0 |

**Table 4** Description of CS0N to CS7N bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 15 to 12 | M22 to M19 | Address mask for block size selection; see Table 5. |
| 11 to 10 | RW1 to RW0 | Read/Write bus control (R/$\overline{W}$); see Table 6. |
| 9 to 8 | MD1 to MD0 | MODE selection; see Table 7. |
| 7 to 3 | A23 to A19 | Decoded base address; this should be a multiple of the block size (other codes are reserved for test or reset state); after reset: A23 to A19 = 11111 except for $\overline{CSBT}$. |
| 2 to 0 | WS2 to WS0 | Wait states 0 to 6 (see Table 8); 7 wait states for $\overline{DTACK}$ to be pulled LOW by the external address decoding circuitry. The default value after reset is ‘110B’ for $\overline{CSBT}$ and ‘111B’ for the other chip-selects. |

**Table 5** Address mask for block size selection

| M22 | M21 | M20 | M19 | BLOCK SIZE |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 512 kbytes |
| 0 | 0 | 0 | 1 | 1 Mbyte |
| 0 | 0 | 1 | 1 | 2 Mbytes |
| 0 | 1 | 1 | 1 | 4 Mbytes |
| 1 | 1 | 1 | 1 | 8 Mbytes; default value after a CPU reset |

**Table 6** Read/Write bits (R/$\overline{W}$)

| RW1 | RW0 | FUNCTION |
|---|---|---|
| 0 | 0 | Read only with length of $\overline{AS}$ |
| 0 | 1 | Write only with length of $\overline{DS}$ |
| 1 | 0 | Write only with length of $\overline{AS}$ |
| 1 | 1 | Read/write with length of $\overline{AS}$; default value after a CPU reset |

**Table 7** Mode selection

| MD1 | MD0 | FUNCTION |
|---|---|---|
| 0 | 0 | Alternate function |
| 0 | 1 | Low byte access only |
| 1 | 0 | High byte access only |
| 1 | 1 | Word access; default value after a CPU reset |

**Table 8** Wait states selection

| WS2 | WS1 | WS0 | WAIT STATES |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 2 |
| 0 | 1 | 1 | 3 |
| 1 | 0 | 0 | 4 |
| 1 | 0 | 1 | 5 |
| 1 | 1 | 0 | 6[1] |

**Note**

1. The default value after a CPU reset.

**Table 9** Number of clock periods per bus cycle

Number of clock periods per bus cycle, dependent on the programmed length of FBC (Fast Bus Cycle bit in the SYSCON register) and $\overline{CSn}$ (chip-select).

| WAIT STATES | LENGTH OF $\overline{CSn}$ = LENGTH OF $\overline{AS}$ | | | LENGTH OF $\overline{CSn}$ = LENGTH OF $\overline{DS}$ | | | |
|---|---|---|---|---|---|---|---|
| | FBC = 1 | | FBC = 0 | FBC = 1 | | FBC = 0 | |
| | READ | WRITE | R/W | READ | WRITE | READ | WRITE |
| 0 | 3 | 4 | 4 | 3 | 4 | 4 | 4 |
| 1 | 4 | 4 | 4 | 4 | 5 | 4 | 5 |
| 2 | 5 | 5 | 5 | 5 | 6 | 5 | 6 |
| 3 | 6 | 6 | 6 | 6 | 7 | 6 | 7 |
| 4 | 7 | 7 | 7 | 7 | 8 | 7 | 8 |
| 5 | 8 | 8 | 8 | 8 | 9 | 8 | 9 |
| 6 | 9 | 9 | 9 | 9 | 10 | 9 | 10 |

## 6.3 Dynamic bus port sizing

The memory bus size can be selected to be 16 or 8-bit wide depending on the ports width of external memories and peripherals. It is possible via the register BSREG to define for each chip-select the bus width to 16-bit or 8-bit used for the transfer of data to or from external memory.

The 7-bit register BSREG defines the bus size associated with each chip-select function (except for $\overline{CSBT}$).

The bus size of the chip-select boot $\overline{CSBT}$ (CS7N) is hardware defined by the pin BSIZE.The state of the pin BSIZE is latched at the end of the reset sequence.

When an address generated by the CPU is identified by a chip-select block as belonging to it's address segment, the corresponding bit of the register BSREG is used to define the sequence of bus transfer in 16 or 8-bit mode. Several chip-selects with different bus sizes should not address the same memory segment. For each case the number of bus cycles necessary to transfer a byte, word or long word is a function of the bus size. For example, a word read on a 8-bit bus will take 2 bus cycles and the high byte is read first. The 8-bit port uses the pins D7 to D0.

See Table 11 and 12 and also Section 6.2 for more detailed information on the programmable chip-selects and the dynamic bus sizing.

### 6.3.1 BUS SIZE REGISTER (BSREG)

**Table 10** Bus Size Register (address FFFF A811H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | BS6 | BS5 | BS4 | BS3 | BS2 | BS1 | BS0 |

**Table 11** Description of BSREG bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | – | Reserved. |
| 6 to 0 | BS6 to BS0 | Bus size for the data transfer with respect to the corresponding chip-select ($\overline{CS6}$ to $\overline{CS0}$). If BSn = 0, then the bus size is in 16-bit mode; the default value after a CPU reset. If BSn = 1, then the bus size is in 8-bit mode. Where n = 0 to 6. |

**Table 12** Bus size depending on BSIZE, CSBTX and BSn (n = 0 to 6)

| PIN BSIZE | BIT CSBTX | BUS SIZE OF $\overline{CS0}$ TO $\overline{CS6}$[1] | | BUS SIZE OF CSBTX | | PORT PL AVAILABLE (P0 TO P7) |
|---|---|---|---|---|---|---|
| | | BSn = 0 | BSn = 1 | AT BOOT | AFTER BOOT | |
| 0 | 0 | 16 bit | 8 bit | 16 | 16 | no |
| 0 | 1 | 16 bit | 8 bit | 16 | 8 | yes |
| 1 | 0 | note 2 | 8 bit | 8 | 8 | yes |
| 1 | 1 | 16 bit | 8 bit | 8 | 16 | no |

**Notes**

1. Depending on bit BSn in register BSREG.
2. The default value after reset of bits BSn in register BSREG is logic 0 which corresponds to 16-bit mode for $\overline{CS0}$ to $\overline{CS6}$. In this case, it is recommended to set BSn to logic 1 in the boot routine. Afterwards if CSBTX is set to logic 1, BSn can be reset to logic 0 by software for further transfers in 16-bit mode.

### 6.4 System Control Register (SYSCON)

The P90CL301BFH uses a System Control Register (SYSCON) for adjusting system parameters.

**Table 13** System Control Register (address FFFF 8000H)

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7[1] | 6[1] | 5 | 4 | 3 | 2 | 1[2] | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WDSC | BPE | CSBTX | STBY | PCLK3 | PCLK2 | PDE | GF | PCLK1 | PCLK0 | IM | WD | FBC | PD | IDL | DOFF |

**Notes**

1. The default values after a CPU reset: PCLK1 = 1 and PCLK0 = 1; all other SYSCON bits are a logic 0.
2. All bits are reset by the RESET instruction, except the IDL bit which is only reset by a CPU reset.

**Table 14** Description of SYSCON bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 15 | WDSC | **Bus error Watchdog short cycle**. WDSC = 0 for normal mode; the bus error Watchdog counts 2048 periods before activating the bus error sequence. WDSC = 1 for Bus error Watchdog short cycle; the Watchdog counts 16 periods before activating the bus error sequence. |
| 14 | BPE | **Bus pull-up enable**. If BPE = 0, the Address and Data bus internal pull-ups are switched off. If BPE = 1, the Address and Data bus internal pull-ups are switched on. |
| 13 | CSBTX | **Invert bus size for chip select boot and mode of port P0 to P7**. CSBTX = 0 for normal mode; bus size is defined by the pin BSIZE. If CSBTX = 1, the chip select boot is defined by the inverted value of the pin BSIZE. The mode change should be executed from the internal RAM or from a memory activated by any other chip select than $\overline{\text{CSBT}}$. For further details see also Section 6.3. |
| 12 | STBY | **CPU Standby mode**. STBY = 0, for normal mode. STBY = 1, for Standby mode; only the CPU clock is switched off, the peripheral clocks are still running (see Fig.4). |
| 11, 7 and 6 | PCLK3, PCLK1 and PCLK0 | **Prescaler for primary peripheral clock (FCLK) and the UART clock in mode 0**. The CPU clock = CLK; FCLK = $^{1}/_{\text{divisor}} \times$ CLK. See Table 15 for the divisor values. |
| 10 | PCLK2 | **Prescaler for secondary peripheral clock FCLK2** (derived from the primary peripheral clock FCLK), used for the ADC; the maximum value of the FCLK2 clock is dependent on the supply voltage $V_{DD}$; see Section 19. If PCLK2 = 0, then FCLK is divided by 2; if PCLK2 = 1, then FCLK is divided by 4. |
| 9 | PDE | If PDE = 0, then bits A22 to A19 are in normal operation; If PDE =1, then bits A22 to A19 are used as 8051 peripheral chip-select $\overline{\text{PCS3}}$ to $\overline{\text{PCS0}}$. |
| 8 | GF | **General purpose flag bit**; reset to a logic 0 after CPU reset. |
| 5 | IM | For IM = 0, level 7 is loaded into the Status Register during interrupt processing to prevent the CPU from being interrupted by another interrupt source. For IM = 1, the current interrupt level is loaded into the Status Register allowing nested interrupts. |
| 4 | WD | For WD = 0, the time-out for bus error detection is switched off. If the time-out is not used, the Watchdog Timer can be used to stop a non-acknowledged bus transfer. For WD = 1, the time-out for bus error detection is activated. If no $\overline{\text{DTACK}}$ has been sent by the addressed device after 128 $\times$ 16 internal clock cycles the on-chip bus error signal is activated. |
| 3 | FBC | FBC = 0, normal bus cycle; FBC = 1, fast bus cycle. An external read bus cycle can take a minimum of 3 clock periods; the minimum write cycle is still 4 clock periods; in order to get this access time $\overline{\text{DTACK}}$ should be asserted on time. |
| 2 | PD | PD = 0, for normal mode; PD = 1, for Power-down mode (see Section 6.8). |
| 1 | IDL | IDL = 0, for normal mode; IDL = 1, for Idle mode (see Section 6.8). |
| 0 | DOFF | DOFF = 0, for normal mode. DOFF = 1, for delay counter off; if set at wake-up from Power-down the delay counter waiting period is skipped. |

**Table 15** Selection of prescaler divisor values

| PCLK3 | PCLK1 | PCLK0 | DIVISOR (D) | DIVISOR FOR UART IN MODE 0 |
|---|---|---|---|---|
| 0 | 0 | 0 | 2 | 6 |
| 0 | 0 | 1 | 3 | 6 |
| 0 | 1 | 0 | 4 | 6 |
| 0 | 1 | 1 | 5 (default value after a CPU reset) | 6 |
| 1 | 0 | 1 | 6 | 12 |
| 1 | 1 | 0 | 8 | 12 |
| 1 | 1 | 1 | 10 | 12 |

### 6.5 Reset operation

The reset circuitry of the P90CL301BFH is connected to the pins $\overline{\text{RESET}}$, $\overline{\text{HALT}}$, RESETIN and to the internal Watchdog Timer. A Schmitt trigger is used at the input pin for noise rejection. After Power-on a CPU reset is accomplished by holding the $\overline{\text{RESET}}$ pin and the $\overline{\text{HALT}}$ pin LOW for at least 50 oscillator clocks after the oscillator has stabilized.

For further information on the clock generation, see Section 6.6. The CPU responds by reading the reset vectors; the long word at address 000000H is loaded into the Supervisor stack and the long word data at address 000004H is loaded into the program counter PC. The interrupt level is set to 7 in the Status Register and execution starts at the PC location. By pulling the $\overline{\text{RESET}}$ pin LOW and keeping $\overline{\text{HALT}}$ HIGH, only the peripherals are reset.

When $V_{DD}$ is turned on and its rise time does not exceed 10 ms, an automatic reset can be performed by connecting the RESETIN pin to $V_{DD}$ via an external capacitor. The external capacitor is charged via an internal pull-down resistor.

The $\overline{\text{RESET}}$ pin can also be pulled LOW internally by a pull-down transistor activated by an overflow of the Watchdog Timer. When the CPU executes a RESET instruction, the $\overline{\text{RESET}}$ pin is pulled LOW. When the CPU is internally halted (at double bus fault), the $\overline{\text{HALT}}$ pin is pulled LOW and only a CPU reset can restart the processor.

The internal signal RESET_AS (Reset Asynchronous) resets the core and all registers.

When an internal Watchdog Timer overflow occurs, an internal CPU reset is generated which resets all registers except the SYSCON, PCON, PRL and PRH registers and pulls the $\overline{\text{RESET}}$ pin LOW during 12 clock cycles.

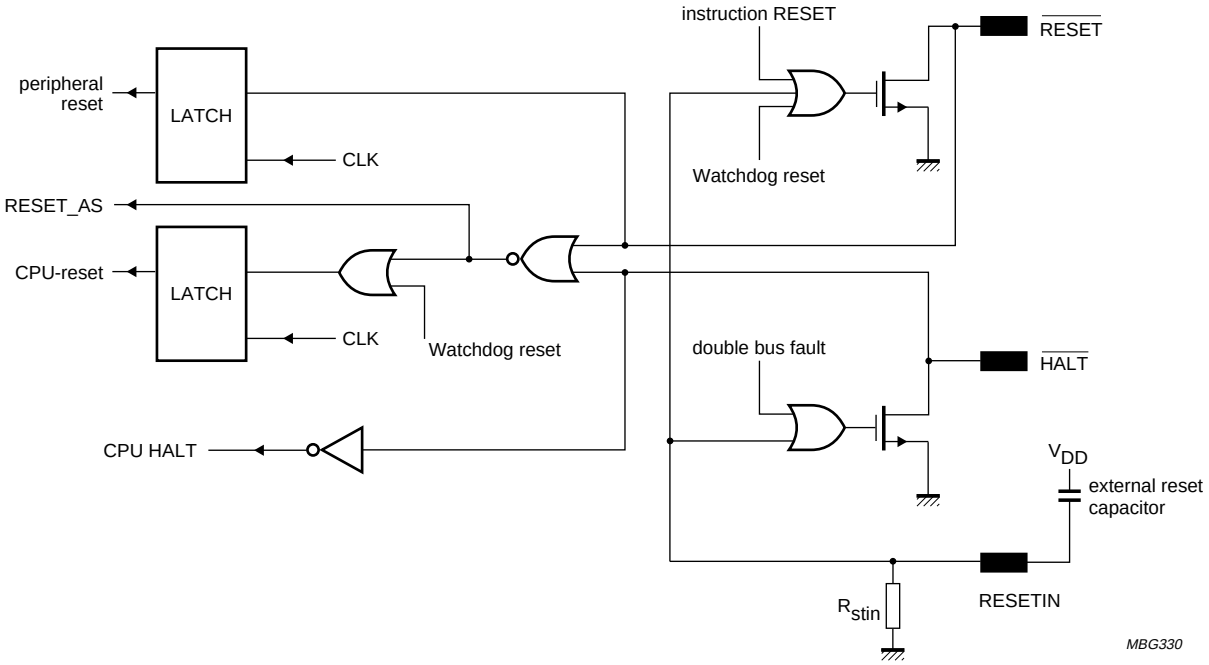



Fig.3 Reset circuitry.

### 6.6 Clock generation

An external clock can be used with the P90CL301BFH. The duty cycle of the external clock should be 50/50 ±5% over the full temperature and voltage range.

For peripherals like Watchdog Timer, I²C-bus, PWM, Timer and baud rate generator, a programmable prescaler generates a peripheral clock FCLK.

The prescaler is controlled by the System Control Register (SYSCON). The internal clock is divided by a factor 2, 3, 4, 5, 6, 8 or 10 (function of bits PCLK0, PCLK1 and PCLK3; see Table 15).

For the ADC a secondary peripheral clock FCLK2 is derived from the peripheral clock by dividing it either by 4 or 2 (function of the bit PCLK2; see Table 14).

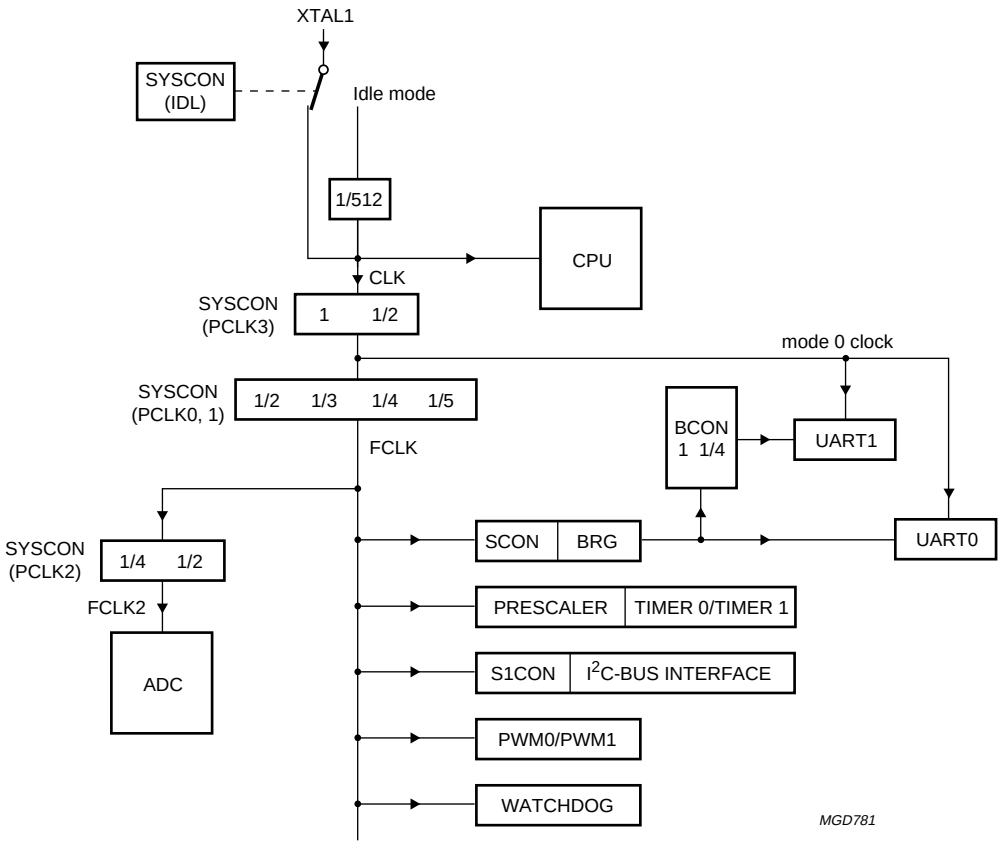



Fig.4 P90CL301BFH internal clock generation.

## 6.7 Interrupt controller

An interrupt controller handles all internal and external interrupts. It delivers the interrupt with the highest priority level to the CPU. The following interrupt requests are generated by the on-chip peripherals:

- $I^2C$-bus
- UARTs: received data / transmitted data
- Timers: two flags for the timers T0 and T1
- ADC: analog-to-digital conversion completed.

The external interrupt requests are generated with the pins NMIN and the seven external interrupts $\overline{INT0}$ to $\overline{INT6}$.

### 6.7.1 Interrupt arbitration

The interrupt priority levels are programmable with a value between 0 and 7. Level 7 has the highest priority, level 0 disables the corresponding interrupt source. In case of interrupt requests of equal priority level at the same time a hardware priority mechanism gives priority order as shown in Table 16.

The execution of interrupt routines can be interrupted by another interrupt request of a higher priority level. In 68070 mode (SYSCON bit IM = 1) when an interrupt is serviced by the CPU, the corresponding level is loaded into the Status Register. This prevents the current interrupt from getting interrupted by any other interrupt request on the same or a lower priority level. If IM is reset, priority level 7 will always be loaded into the Status Register and so the current interrupt cannot be interrupted by an interrupt request of a level less than 7.

Each on-chip peripheral unit including the eight interrupt lines generate only auto-vectored interrupts. No acknowledge is necessary. For external interrupts the vectors 25 to 31 are used, for on-chip peripheral circuits a second table of 7 vectors are used (57 to 63); see Section 7.3.2.

**Table 16** Priority order

| SIGNAL | PRIORITY ORDER |
|---|---|
| NMIN | highest |
| $\overline{INT6}$ | |
| $\overline{INT5}$ | |
| $\overline{INT4}$ | |
| $\overline{INT3}$ | |
| $\overline{INT2}$ | |
| $\overline{INT1}$ | |
| $\overline{INT0}$ | |
| $I^2C$-bus | |
| ADC | |
| UART1 receiver | |
| UART1 transmitter | |
| UART0 receiver | |
| UART0 transmitter | |
| Timer 1 | |
| Timer 0 | lowest |

### 6.7.2 External latched interrupts

NMIN and $\overline{INT0}$ to $\overline{INT6}$ are 8 external interrupt inputs. These pins are connected to the interrupt function only when the corresponding bit in the SPCON control register is set (see Section 8.2; Table 29). Seven interrupt inputs $\overline{INT0}$ to $\overline{INT6}$ are edge sensitive on HIGH-to-LOW transition and their priority levels are programmable. The interrupt NMIN is non-maskable (except if it is programmed as a port) and is also edge sensitive on HIGH-to-LOW transition. The priority level of NMIN is fixed to 7.

The external interrupts are controlled by the registers LIR0 to LIR3; see Tables 17 and 18.

#### *6.7.2.1 Latched Interrupt Registers (LIR0 to LIR3)*

**Table 17** Latched Interrupt Registers

| ADDRESS | REGISTER | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FFF 8101H | LIR0 | PIR1 | IPL1.2 | IPL1.1 | IPL1.0 | PIR0 | IPL0.2 | IPL0.1 | IPL0.0 |
| FFF 8103H | LIR1 | PIR3 | IPL3.2 | IPL3.1 | IPL3.0 | PIR2 | IPL2.2 | IPL2.1 | IPL2.0 |
| FFF 8105H | LIR2 | PIR5 | IPL5.2 | IPL5.1 | IPL5.0 | PIR4 | IPL4.2 | IPL4.1 | IPL4.0 |
| FFF 8107H | LIR3 | PIR7 | 1 | 1 | 1 | PIR6 | IPL6.2 | IPL6.1 | IPL6.0 |

**Table 18** Description of LIR0 to LIR3 bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 and 3 | PIRn | **Pending interrupt request**. n = 0 to 7; $\overline{\text{INT7}}$ corresponds to the interrupt NMIN; PIRn = 1, pending interrupt request for pin $\overline{\text{INTn}}$. PIRn = 0 (default value after a CPU reset), no pending interrupt. When a valid interrupt request has been detected this bit is set. It is automatically reset by the interrupt acknowledge cycle from the CPU. It can be reset by software by writing a logic 0, however writing a logic 1 has no effect on the flag. To reset only one flag, a logic 0 should be written to the bit address and a logic 1 to the other interrupt requests. The use of BCLR instruction should be avoided (PIR7 is cleared when the pin NMIN is set HIGH) |
| 6 to 4 | IPLm.2 to IPLm.0 | Interrupt priority level of pins $\overline{\text{INT0}}$ to $\overline{\text{INT6}}$ (fixed to '111B' for NMIN in LIR3); m = 0 to 6. |
| 2 to 0 | | |

*6.7.2.2 Pending Interrupt Flag Register (PIFR)*

An additional register PIFR contains copies of the PIR flags. The PIF flags are set at the same time as the PIR flags when an interrupt is activated, but these flags are not reset automatically during the interrupt acknowledge cycle. They can only be cleared by software and keep a trace of the interrupt event. The detection of an external interrupt is indicated by the corresponding PIF-bit being set to a logic 1.

**Table 19** Pending Interrupt Flag Register (address FFFF 810F)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PIF7 | PIF6 | PIF5 | PIF4 | PIF3 | PIF2 | PIF1 | PIF0 |

### 6.7.3 NOTE ON SIMULTANEOUS INTERRUPTS

If an internal interrupt is immediately followed by an external interrupt (i.e. both interrupts occurring within 12 clock cycles) and both these interrupts have the same interrupt level, then the CPU might hang up during the acknowledge cycle of the internal interrupt.

In the interrupt controller a flag WIN is set for each interrupt as soon as the interrupt is activated and will be reset when an interrupt of higher priority occurs or during the acknowledge cycle. The WIN flag is used to determine which PIR flag should be reset.

A conflict occurs if within the interval starting at the CPU sampling of the first internal interrupt and ending at the acknowledge cycle, a second external interrupt resets the WIN flag of the first interrupt (external interrupts have higher priority than internal).

When the CPU acknowledges the first internal interrupt the auto-vector acknowledge signal cannot be asserted as its WIN flag was reset, and the CPU hangs up.

This situation can be solved by using the bus time-out counter controlled by the System Control Register (SYSCON) with the bits WD and WDSC set. In the case of hang-up an internal bus error condition will be asserted after 16 clocks and the CPU will execute the exception SPURIOUS INTERRUPT at vector 60H. In the exception service routine the interrupt flags PIR should be polled to detect which interrupts caused the conflict, the corresponding PIR flags should be cleared by software and a call to the interrupt routines executed.

## 6.8 Power reduction modes

The P90CL301BFH supports three power reduction modes. A Power-down mode where the clock is frozen, a Standby mode where only the CPU is stopped, and an Idle mode where the external clock is divided by 512 (see Fig.4).

### 6.8.1 POWER-DOWN MODE

The Power-down operation freezes the oscillator. It can only be activated by setting the PD bit in the SYSCON register and thereafter execute the STOP instruction.

The instruction flow to enter the Power-down mode is:

```
BSET #PD, SYSCON
STOP #$2700.
```

In this state all the register contents are preserved. The CPU remains in this state until an internal reset occurs or a LOW level is present on any of the external interrupt pins $\overline{INT0}$ to $\overline{INT6}$ or NMIN. If the wake-up is done via an external interrupt, the processor will first execute an external interrupt of level 7. If the IPL level in the LIR register is set to 7, a second interrupt of level 7 will be executed. It is preferable to set the IPL to 0.

In Power-down mode $V_{DD}$ may be reduced to minimize power consumption. However, the supply voltage must not be reduced until Power-down mode is active, and must be restored before a external reset or an interrupt is activated.

In case of an external reset, the pin should be held active until the external oscillator has restarted and stabilized.

In case of an external interrupt wake-up, any $\overline{INTn}$ or NMIN pin should go LOW and the corresponding bit ESn (n = 0 to 7) in register SPCON should be set. If the DOFF bit in the SYSCON is not set, an internal delay counter ensures that the internal clock is not active before 1536 clock cycles. After that time the oscillator is stable and normal exception processing can be executed. The PD bit is cleared automatically during the wake-up.

In order to have a fast start-up the DOFF bit should be set, switching off the delay counter and enabling the immediate clocking and restart of the controller.

For minimum power consumption during Power-down mode, the address and data pins should be pulled HIGH externally or bit BPE in register SYSCON should be set (i.e. internal pull-ups enabled).

### 6.8.2 STANDBY MODE

When the STBY bit in the SYSCON register is set, the CPU clock is stopped and the status of the processor is frozen, however, the clocks of all other on-chip peripherals are still running at the nominal frequency; these peripherals are:

- Timers
- External and internal interrupts
- UARTs and baud rate generator
- $I^2C$-bus interface
- Watchdog Timer
- PWMs
- ADC.

The CPU exits this mode when an internal or external interrupt is activated, and proceeds with the normal program execution.

For minimum power consumption internal pull-ups on address and data buses can be switched on by setting the control bit BPE in the SYSCON register. The pull-ups should be switched off in normal mode if not needed.

### 6.8.3 IDLE MODE

In the Idle mode the crystal or external clock is divided by a factor 512. The current is reduced drastically but the controller continues to operate. This mode is entered by setting the bit IDL in the SYSCON register. The next instruction will be executed at a slower speed. To return to normal mode the IDL bit should be reset.

It should be noted that all peripheral functions are also slowed down, and some cannot be used normally, for example UART, $I^2C$-bus, ADC and PWM.
The Power-down mode can also be entered from the Idle mode. After a wake-up the controller restarts in Idle mode.

## 7 CPU FUNCTIONAL DESCRIPTION

### 7.1 General

The CPU of the P90CL301BFH is software compatible with the Motorola MC68000, hence programs written for the MC68000 will run on the P90CL301BFH without modifications. However, for certain applications the following differences between processors should be noted:

- Differences exist in the address/bus error exception processing since the P90CL301BFH can provide full error recovery.
- The timing is different for the P90CL301BFH due to a new internal architecture and technology.
  The instruction execution timing is different for the same reasons.

### 7.2 Programming model and data organization

The programming model is identical to that of the MC68000 (see Fig.5), with seventeen 32-bit registers, a 32-bit Program Counter and a 16-bit Status Register. The eight data registers (D0 to D7) are used for byte, word and long-word operations. The Address Registers (A0 to A6) and the System Stack Pointer A7 can be used as software stack pointers and base address registers. In addition, these registers can be used for word and long-word address operations. All seventeen registers can be used as index registers.

The P90CL301BFH supports 8, 16 and 32-bit integers as well as BCD data and 32-bit addresses. Each data type is arranged in the memory as shown in Fig.6.

**Table 20** Format of the Status Register and description of the bits; r = reserved

| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T | – | S | – | – | 12 | 11 | 10 | – | – | – | X | N | Z | V | C |
| Trace mode | r | Supervisor | r | | Interrupt mask | | | r | | | Extend | Negative | Zero | Overflow | Carry |

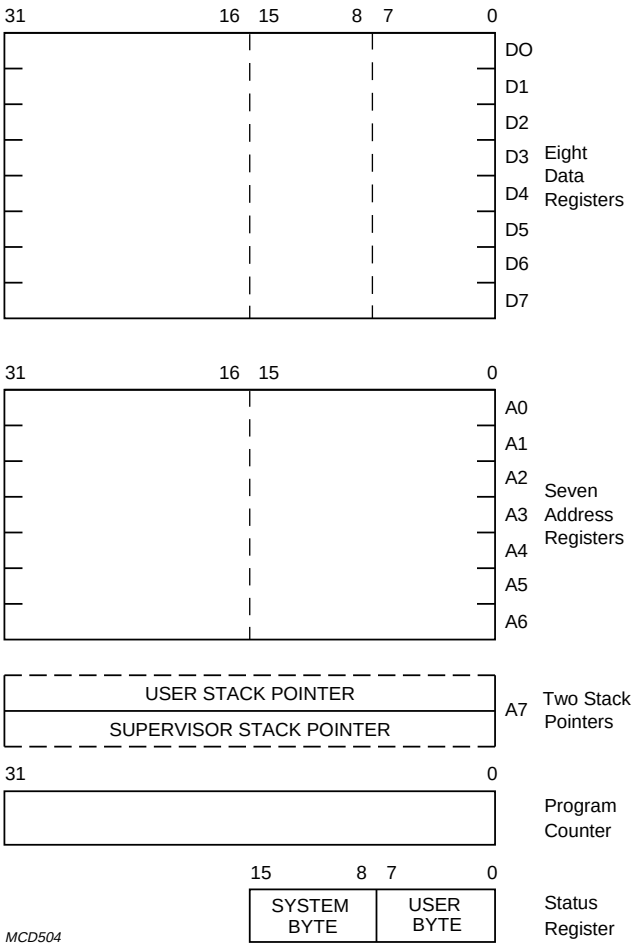
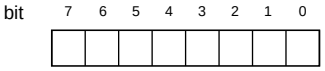
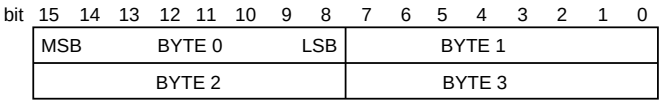


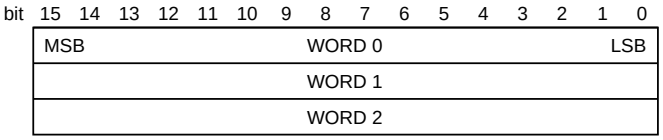

Fig.5 Programming model.



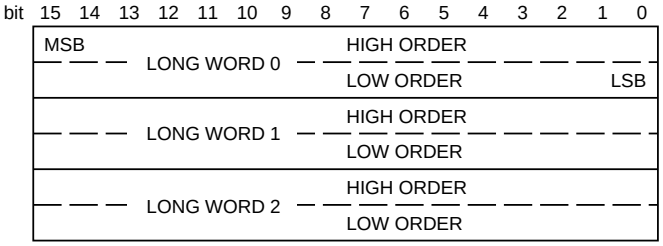

(a) Bit data (1 Byte = 8 bits).



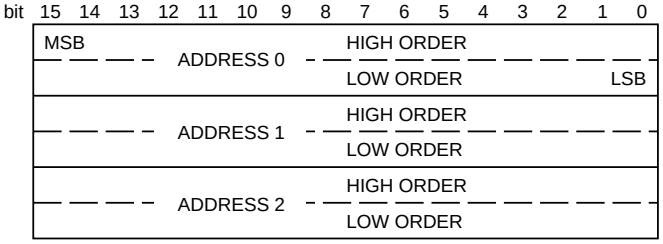

(b) Integer data (1 Byte = 8 bits).



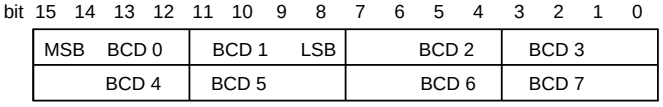

(c) Word data (16 bits).




(d) Long-word data (32 bits).




(e) Addresses (1 address =32 bits).




(f) BCD data (2 BCD digits = 1 Byte). *MCD505*

Fig.6 Memory data organization.

## 7.3 Processing states and exception processing

The P90CL301BFH operates with a maximum internal clock frequency of 27 MHz down to static operation. Each clock cycle is divided into 2 states. A non-access machine cycle has 3 clock cycles or 6 states (S0 to S5). A minimum bus cycle normally consists of 3 clock cycles (6 states). When $\overline{\text{DTACK}}$ is not asserted, indicating that data transfer has not yet been terminated, wait states (WS) are inserted in multiples of 2.

The CPU is always in one of the four processing states:

- Normal
- Exception
- Halt
- Stopped.

The Normal processing state is associated with instruction execution; the memory references fetch instructions or load/save results. A special case of the Normal state is the Stopped state which is entered by the processor when a STOP instruction is executed. In this state the CPU does not make any further memory references.

The Exception state is associated with interrupts, trap instruction, tracing and other exceptional conditions. The exception may be generated internally by an instruction or by any unusual condition arising during the execution of an instruction. Externally, exception processing can be forced by an interrupt or by reset.

The halted processing state is an indication of a catastrophic hardware failure. For example, if during exception processing of a bus error another bus error occurs, the CPU assumes that the system is unusable and halts. Only an external reset can restart a halted processor. Note that a CPU in the stopped state is not in the halted state or vice versa.

The Supervisor can work in the User or Supervisor state determined by the state of bit S in the Status Register. Accesses to the on-chip peripherals are achieved in the Supervisor state.

All exception processing is performed in the Supervisor state once the current contents of the Status Register has been saved. Then the exception vector number is determined and copies of the Status Register, the program counter and the format/vector number are saved on the Supervisor stack using the Supervisor Stack Pointer (SSP). Finally the contents of the exception vector location is fetched and loaded into the Program Counter (PC).

### 7.3.1 Reference classification

When the processor makes a reference, it classifies the kind of reference being made, using the encoding of the three function code internal lines. This allows external translation of addresses, control of access, and differentiation of special processor states, such as interrupt acknowledge. Table 21 shows the classification of references.

**Table 21** Reference classification

| FUNCTION CODE | | | REFERENCE CLASS |
|---|---|---|---|
| FC2 | FC1 | FC0 | |
| 0 | 0 | 0 | unassigned |
| 0 | 0 | 1 | User Data |
| 0 | 1 | 0 | User Program |
| 0 | 1 | 1 | unassigned |
| 1 | 0 | 0 | unassigned |
| 1 | 0 | 1 | Supervisor Data |
| 1 | 1 | 0 | Supervisor Program |
| 1 | 1 | 1 | interrupt acknowledge |

### 7.3.2 Exception vectors

Exception vectors are memory locations from where the CPU fetches the address of a routine that will handle that exception. All exception vectors are 2 words long, except for the reset vector which consists of 4 words, containing the PC and the SSP. All exception vectors are in the Supervisor Data space.

A vector number is an 8-bit number which, multiplied by 4, gives the address of an exception vector. Vector numbers are generated internally. The memory map for the exception vectors is shown in the Table 22.

**Table 22** Exception vector assignment

| VECTOR NO. | DECIMAL | HEX | ASSIGNMENT |
|---|---|---|---|
| 0 | 0 | 000 | reset: initial SSP |
| – | 4 | 004 | reset: initial PC |
| 2 | 8 | 008 | bus error |
| 3 | 12 | 00C | address error |
| 4 | 16 | 010 | illegal instruction |
| 5 | 20 | 014 | zero divide |
| 6 | 24 | 018 | CHK instruction |
| 7 | 28 | 01C | TRAPV instruction |
| 8 | 32 | 020 | privilege violation |
| 9 | 36 | 024 | trace |
| 10 | 40 | 028 | line 1010 emulator |
| 11 | 44 | 02C | line 1111 emulator |
| 12[1] | 48 | 030 | unassigned, reserved |
| 13[1] | 52 | 034 | unassigned, reserved |
| 14 | 56 | 038 | format error |
| 15 | 60 | 03C | uninitialized interrupt vector |
| 16 to 23[1] | 64 to 95 | 040 to 05C | unassigned, reserved |
| 24 | 96 | 060 | spurious interrupt |
| 25 | 100 | 064 | level 1 external interrupt auto-vector |
| 26 | 104 | 068 | level 2 external interrupt auto-vector |
| 27 | 108 | 06C | level 3 external interrupt auto-vector |
| 28 | 112 | 070 | level 4 external interrupt auto-vector |
| 29 | 116 | 074 | level 5 external interrupt auto-vector |
| 30 | 120 | 078 | level 6 external interrupt auto-vector |
| 31 | 124 | 07C | level 7 external interrupt auto-vector |
| 32 to 47 | 128 to 191 | 080 to 0BF | TRAP instruction vectors |
| 48 to 56[1] | 192 to 227 | 0C0 to 0E3 | reserved |
| 57 | 228 | 0E4 | level 1 on-chip interrupt auto-vector |
| 58 | 232 | 0E8 | level 2 on-chip interrupt auto-vector |
| 59 | 236 | 0EC | level 3 on-chip interrupt auto-vector |
| 60 | 240 | 0F0 | level 4 on-chip interrupt auto-vector |
| 61 | 244 | 0F4 | level 5 on-chip interrupt auto-vector |
| 62 | 248 | 0F8 | level 6 on-chip interrupt auto-vector |
| 63 | 252 | 0FC | level 7 on-chip interrupt auto-vector |
| 64 to 255 | 256 to 1023 | 100 to 3FF | reserved |

**Note**

1. Vectors 12, 13, 16 to 23 and 48 to 56 are reserved for future enhancements.

#### 7.3.3 INSTRUCTION TRAPS

Traps are exceptions caused by instructions arising from CPU recognition of abnormal conditions during instruction execution or from instructions whose normal behaviour is to cause traps.

Some instructions are used specifically to generate traps. The TRAP instruction always forces an exception and is useful for implementing system calls for User Programs. The TRAPV and CHK instructions force an exception if the User Program detects a run-time error, possibly an arithmetic overflow or a subscript out of bounds.
The signed divide (DIVS) and unsigned divide (DIVU) instructions will force an exception if a divide-by-zero operation is attempted.

#### 7.3.4 ILLEGAL AND UNIMPLEMENTED INSTRUCTIONS

Illegal instruction is the term used to refer to any word that is not the first word of a legal instruction. During execution, if such an instruction is fetched an illegal exception occurs.

Words with bits 15 to 12 equal to ‘1010’ or ‘1111’ are defined as unimplemented instructions and separate exception vectors are allocated to these patterns for efficient emulation. This facility means the operating system can detect program errors, or can emulate unimplemented instructions in software.

#### 7.3.5 PRIVILEGE VIOLATIONS

To provide system security, various instructions are privileged and any attempt to execute one of the privileged instruction while the CPU is in the User state provokes an exception. The privileged instructions are:

- STOP
- RESET
- RTE
- MOVE to SR
- AND (word) immediate to SR
- EOR (word) immediate to SR
- OR (word) immediate to SR
- MOVE to USP.

### 7.4 Tracing

The CPU includes a facility to trace instructions one by one to assist in program development. In the trace state, after each instruction is executed, an exception is forced so that the debugging program can monitor execution of the program under test.

The trace facility uses the T-bit in the Supervisor part of the Status Register. If the T-bit is cleared, tracing is disabled and instructions are executed normally. If the T-bit is set at the beginning of the execution of an instruction, a trace exception will be generated once the instruction has been executed. If the instruction is not executed, either because of an interrupt, or because the instruction is illegal or privileged, the trace exception does also not occur if the instruction is aborted by a reset, bus error, or address error exception. If the instruction is executed, and an interrupt is pending, the trace exception is processed before the interrupt. If the execution of an instruction forces an exception, the forced exception is processed before the trace exception.

As an extreme illustration of the above rules, consider the arrival of an interrupt during the execution of a TRAP instruction, while tracing is enabled. First the trap exception is processed, followed by the trace exception, and finally the interrupt handling routine.

### 7.5 Stack format

The stack format for exception processing is similar to the MC68010 although the instruction stored is not the same, due to the different architecture. To handle this format the P90CL301BFH differs from the MC68000 in that:

- The stack format is changed.
- The minimum number of words put into or restored from stack is 4 (MC68010 compatible, not 3 as with the MC68000).
- The RTE instruction decides (with the aid of the 4 format bits) whether or not more information has to be restored as follows:
  - The P90CL301BFH long format is used for bus errors and address error exceptions.
  - All other exceptions use the short format.
- If another format code, other than those listed above, is detected during the restored action, a FORMAT ERROR occurs.

If the user wants to finish the instruction in which the bus or address error occurred, the P90CL301BFH format must be used on RTE. If no changes to the stack are required during exception processing, the stack format is transparent to the user.

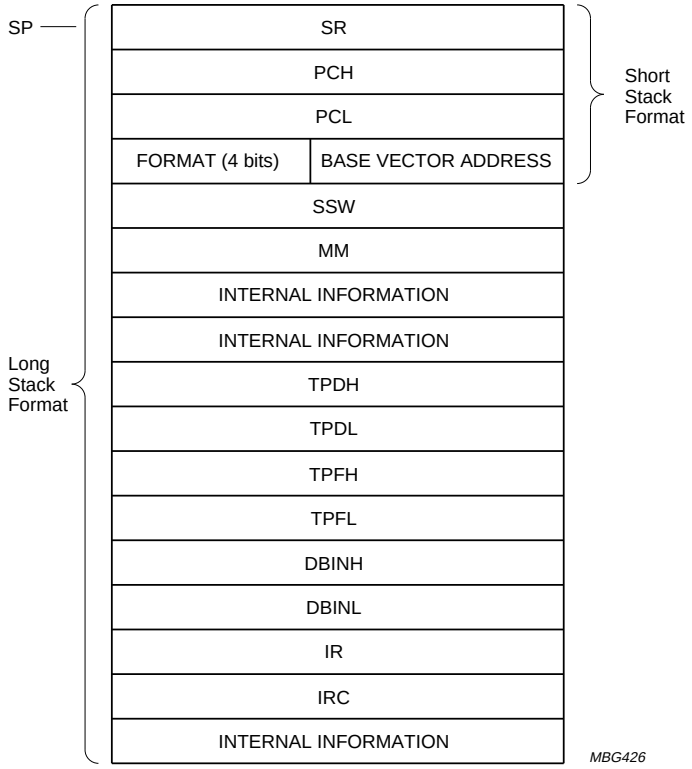



Fig.7 Stack format; see Table 23.

**Table 23** Description of the stack format

| SYMBOL | DESCRIPTION |
|---|---|
| SR | Status Register. |
| PCH/PCL | Program Counter High/Low Word. |
| FORMAT | Indicating either a short stack (only the first four words), or the long for bus and address error exceptions. |
| BASE VECTOR ADDRESS | The base vector address of the exception in the vector table; e.g. 8 for a bus error and 12 for an address error. |
| SSW | Special Status Word. |
| MM | Current Move Multiple Mask. |
| TPDH/TPDL | In the event of faulty write cycle, the data can be found here. |
| TPFH/TPFL | The address used during the faulty bus cycle. |
| DBINH/DBINL | Data that has been read prior to the faulty bus cycle can in some cases be found here. |
| IR | Holds the present instruction executed. |
| IRC | Holds either the present instruction executed or the prefetched instruction. |

### 7.6 CPU interrupt processing

The general interrupt handling mechanism is described in Section 6.7. An interrupt controller handles all interrupts, resolves the priority problem and passes the highest level interrupt to the CPU.

The CPU interrupt handling follows the same basic rules as in the MC68000. However, some remarks must be made:

- Interrupts with a priority level equal to or lower than the current priority level will not be accepted.
- During the acknowledge cycle of an interrupt, the IPL bits of the Status Register are set to the priority of the acknowledged interrupt or to 7. An exception occurs when bit IM = 0 (SYSCON bit 5). In this case level 7 is loaded into the Status Register (see Section 6.4; Table 14).

If the priority level of the pending interrupt is greater than the current processor priority then:

- The exception processing sequence is started
- A copy of the Status Register is saved
- The privilege level is set to Supervisor state
- Tracing is suppressed
- The priority level of the processor is set to that of the interrupt being acknowledged or to 7 depending on the IM flag in the System Control Register.

The processor then gets the vector number from the interrupting device, classifies it as an interrupt acknowledge and displays the interrupt level number being acknowledged on the internal address bus.

As all P90CL301BFH interrupts are auto-vectored, the processor internally generates a vector number corresponding to the interrupt level number.

The processor starts normal exception processing by saving the format word, program counter and Status Register on the Supervisor stack. The value of the vector in the format word is an internally generated vector number multiplied by 4 (format is all zeros). The program counter value is the address of the instruction that would have been executed if the interrupt had not been present. Then the interrupt vector contents are fetched and loaded into the program counter. The interrupt handling routine starts with normal instruction execution.

### 7.7 Bus arbitration

If the $\overline{\text{HALT}}$ pin is held LOW with $\overline{\text{RESET}}$ HIGH the CPU will stop after completion of the current bus cycle. As long as $\overline{\text{HALT}}$ is LOW, all control signals are inactive and all 3-state lines are placed in the high-impedance state. If the $\overline{\text{HALT}}$ pin is held LOW during the transfer of a word in 8-bit mode, the CPU will continue the transfer of the two bytes before it halts.

## 8 PORTS

For general purpose input/output operations the following ports can be used:

- 16-bit bidirectional port lines P15 to P0 composed of two 8-bit ports PL (P7 to P0) and PH (P15 to P8)
- 8-bit port lines SP7 to SP0.

All port pins are multiplexed with other functions, but each one can be individually switched to the port function by setting the corresponding bit in the Port P Control Register (PCON) for ‘port Pn’ and Port SP Control Register (SPCON) for ‘port SPn’.

The port P7 to P0 is multiplexed with the data bus D15 to D8 and is selected by the pin BSIZE.

Each port pin consists of a latch, an output driver with pull-ups and an input buffer.

To use the port as input the port latch should be written with a logic 1. This means only a weak pull-up is on and can be overwritten by an external source logic 0.

When outputting a logic 1, a strong pull-up is turned on only for 1 clock period, and then only the weak pull-up maintains the HIGH level. In read mode, two different internal addresses correspond to the port latch or the port pin.The port values are read via register PPL and PPH.

After reset all ports are initialized as input, and the pins are connected to the port latch with exception for the pin NMIN/SP7 which is connected to the interrupt block.

### 8.1 Port P Control Register (PCON)

The port Pn is controlled via the Port P Control Register (PCON). The register PCON is only reset by an external reset, and not by the RESET instruction. The port latches are accessed through the registers PRL and PRH.

**Table 24** Port P Control Register (address FFFF 8503H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| E15 | E14 | E13 | E12 | E11 | E10 | E9 | E8 |

**Table 25** Description of PCON bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 0 | E15 to E8 | If En = 0, then ‘port Pn’ is enabled; if En = 1, then the alternate function is enabled; n = 8 to 15. The default value after reset is logic 0. |

#### 8.1.1 PORT P LATCHES

**Table 26** Port P Latch least significant byte (PRL; address FFFF 8505H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |

**Table 27** Port Latches High most significant byte (PRH; address FFFF 8509H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |

### 8.2 Port SP Control Register (SPCON)

The special ports SPn (SP0 to SP7) consist of 8 I/O lines and are controlled via the two registers SPCON and SPR. The registers SPCON and SPR are reset by a peripheral reset. The port latch is accessed through the register SPR.

#### 8.2.1 PORT SP CONTROL REGISTER (SPCON)

**Table 28** Port SP Control Register (address FFFF 8109H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ES7 | ES6 | ES5 | ES4 | ES3 | ES2 | ES1 | ES0 |

**Table 29** Description of SPCON bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 0 | ES7 to ES0 | If ESn = 0, then 'port SPn' is enabled; if ESn = 1, then the alternate function is enabled; n = 0 to 7. The default value after reset is logic 0, except for ES7 which is set at reset. |

#### 8.2.2 PORT SP LATCH (SPR)

**Table 30** Port SP latch (FFFF 810BH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SP7 | SP6 | SP5 | SP4 | SP3 | SP2 | SP1 | SP0 |

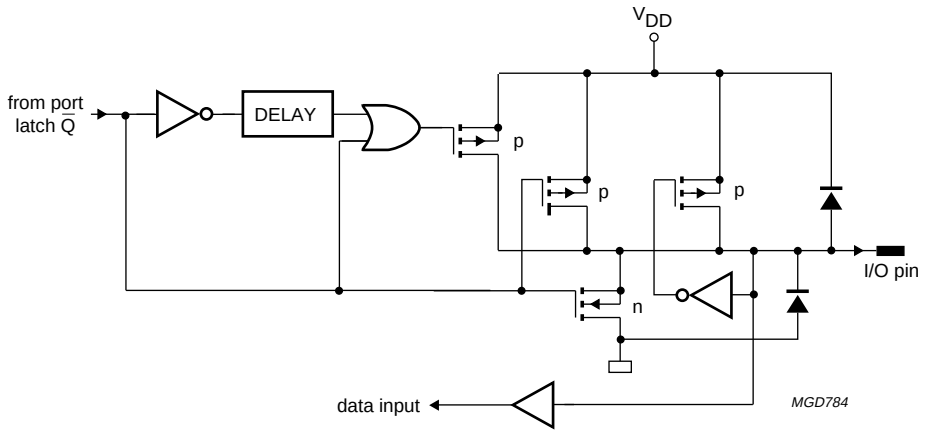
#### 8.2.3 ALTERNATIVE FUNCTIONS FOR PORTS P AND SP

**Table 31** Alternative functions for P0 to P15 and SP0 to SP7 pins

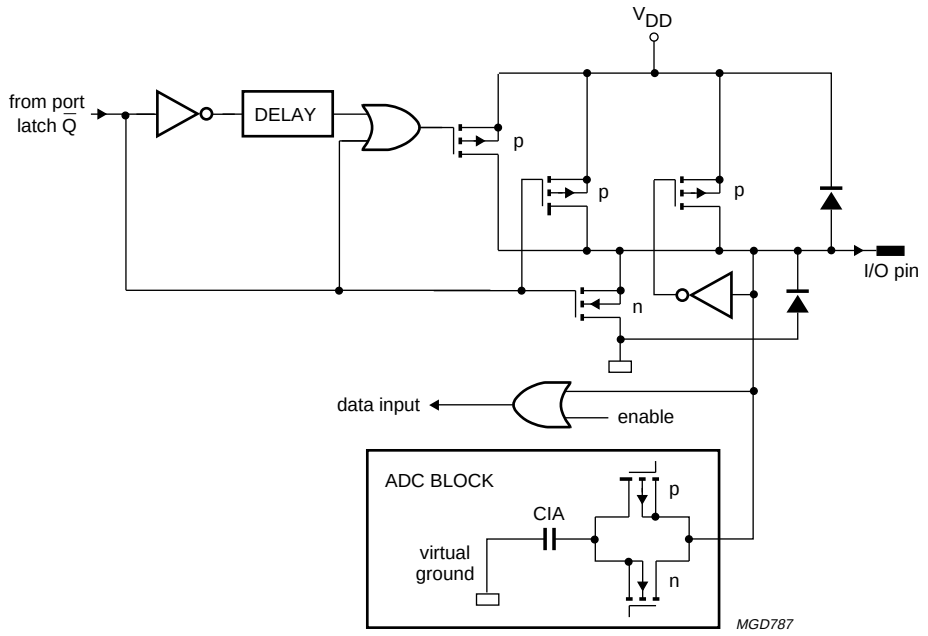
Functions within brackets are parallel functions.

| PORT PIN | ALTERNATE FUNCTION |
|---|---|
| P0 | D8 |
| P1 | D9 |
| P2 | D10 |
| P3 | D11 |
| P4 | D12 |
| P5 | D13 |
| P6 | D14 |
| P7 | D15 |
| P8 | PWM0 (CP0) |
| P9 | PWM1 (CP1) |
| P10 | SCL |
| P11 | SDA |

| PORT PIN | ALTERNATE FUNCTION |
|---|---|
| P12 | ADC0 |
| P13 | ADC1 |
| P14 | ADC2 |
| P15 | ADC3 |
| SP0 | RX1/$\overline{\text{INT0}}$ |
| SP1 | TX1/$\overline{\text{INT1}}$ (CLK0) |
| SP2 | RX0/$\overline{\text{INT2}}$ (CP2) |
| SP3 | TX0/$\overline{\text{INT3}}$ (CP3) |
| SP4 | $\overline{\text{INT4}}$ (CP4) |
| SP5 | $\overline{\text{INT5}}$ (CP5) |
| SP6 | $\overline{\text{INT6}}$ (CLK1) |
| SP7 | NMIN |

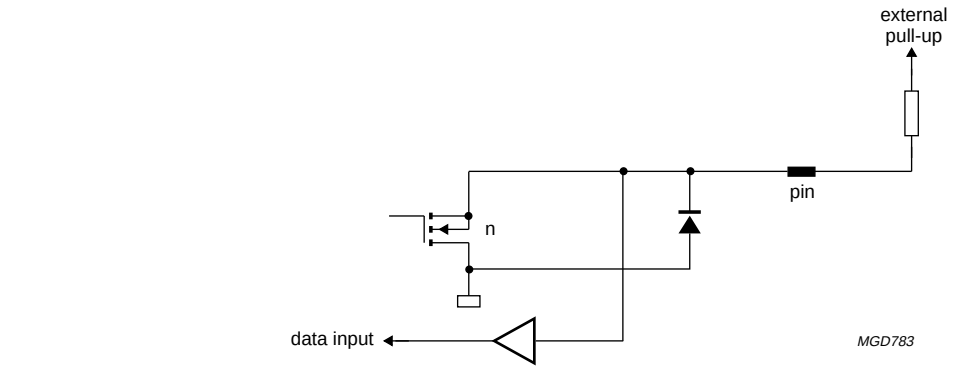



a. WP2 + WP4 port.

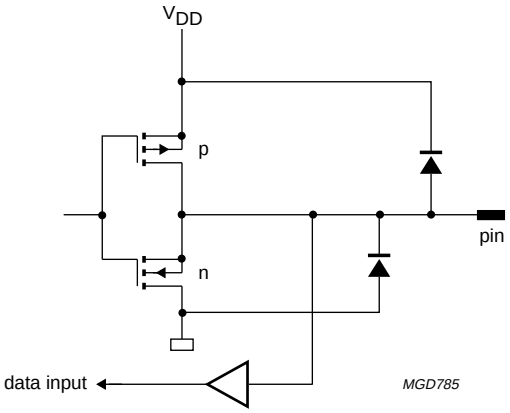



b. AN + WP2 (P15 to P12) port.

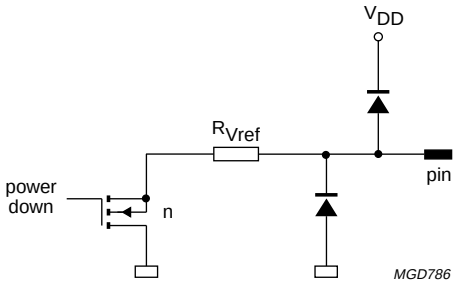
Fig.8 Port schematics (continued in Fig.9).




a. Open-drain port.




b. 3-state port.




c. AREF input.

Fig.9 Port schematics (continued from Fig.8).

## 9 8051 PERIPHERAL BUS

The P90CL301BFH can also directly access the peripheral circuits which are compatible with the 8048/8051 bus.

When the CPU accesses locations located in the 64 kbytes peripheral space, an Address/Data multiplexed access is generated using the AD0 to AD7 lines, the non-multiplexed A8 to A15 lines and the 8051 control bus (ALE, $\overline{RD}$, $\overline{WR}$). In order to use these three signals the alternate mode of the $\overline{CS5}$ to $\overline{CS3}$ should be set. A 8051 bus access is performed by addressing a byte in the 8001 0000H to 8001 FFFFH range.

To reduce the number of interface circuits, the address lines A22 to A19 can be used as peripheral chip-select outputs $\overline{PCS0}$ to $\overline{PCS3}$. This is done by setting the PDE bit (SYSCON) to a logic 1;

- $\overline{PCS0}$ selects memory range 0 kbytes to 16 kbytes
- $\overline{PCS1}$ selects memory range 16 kbytes to 32 kbytes
- $\overline{PCS2}$ selects memory range 32 kbytes to 48 kbytes
- $\overline{PCS3}$ selects memory range 48 kbytes to 64 kbytes.

The timing of the peripheral bus is fixed and compatible with the 8051 peripheral circuits.

## 10 ON-CHIP PERIPHERAL FUNCTIONS

The P90CL301BFH integrates a number of peripheral functions connected to the internal bus:

- Timers (T0 and T1)
- Watchdog
- 2 UART interfaces with one UART queue controller using the internal RAM as data buffers.
- $I^2C$-bus interface
- PWM (Pulse Width Modulation)
- ADC (Analog-to-Digital Converter).

These functions are accessible as memory locations on a byte or word basis. The access is auto-acknowledged by on-chip logic. The on-chip peripheral functions can generate auto-vectored interrupts to the CPU using the second vector table (vectors 57 to 63).

### 10.1 Peripheral interrupt control

The timers T0 and T1, $I^2C$-bus, UART and ADC use a common set of Peripheral Interrupt Control Registers (PICRn; n = 0 to 3). These registers are accessible from the CPU and contain the Interrupt Priority Level flags IPL2 to IPL0 as well as the Pending Interrupt flags PIR.

PIR is set when a valid interrupt request has been detected. It is automatically reset by the interrupt acknowledge cycle from the CPU. The PIR flag can be reset by software.

The Interrupt Priority Level code ‘111B’ represents the interrupt with the highest priority. The code ‘000B’ inhibits the interrupt.

#### 10.1.1 Timer Interrupt Register (PICR0)

On timer overflow or on channel capture/match the pending interrupt request flag PIRTn is set. If the interrupt priority level is different from zero, the timer activates an interrupt to the CPU.

**Table 32** Timer Interrupt Register (address FFFF 8701H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PIRT1 | IPLT1.2 | IPLT1.1 | IPLT1.0 | PIRT0 | IPLT0.2 | IPLT0.1 | IPLT0.0 |

**Table 33** Description of PICR0 bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | PIRT1 | pending interrupt for timer T1 |
| 6 to 4 | IPLT1.2 to IPLT1.0 | interrupt priority level for timer T1 |
| 3 | PIRT0 | pending interrupt for timer T0 |
| 2 to 0 | IPLT0.2 to IPLT0.0 | interrupt priority level for timer T0 |

### 10.1.2 UART INTERRUPT REGISTERS

Each UART can generate two interrupts in transmission and reception via the two registers PICR1 and PICR2.

**Table 34** UART Interrupt Registers PICR1 (address FFFF 8703H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PIRR0 | IPLR0.2 | IPLR0.1 | IPLR0.0 | PIRT0 | IPLT0.2 | IPLT0.1 | IPLT0.0 |

**Table 35** Description of PICR1 bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | PIRR0 | pending interrupt for UART0 in reception |
| 6 to 4 | IPLR0.2 to IPLR0.0 | interrupt priority level for UART0 in reception |
| 3 | PIRT0 | pending interrupt for UART0 in transmission |
| 2 to 0 | IPLT0.2 to IPLT0.0 | interrupt priority level for UART0 in transmission |

**Table 36** UART Interrupt Registers PICR2 (address FFFF 8705H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PIRR1 | IPLR1.2 | IPLR1.2 | IPLR1.2 | PIRT1 | IPLT1.2 | IPLT1.1 | IPLT1.0 |

**Table 37** Description of PICR2 bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | PIRR1 | pending interrupt for UART1 in reception |
| 6 to 4 | IPLR1.2 to IPLR1.0 | interrupt priority level for UART1 in reception |
| 3 | PIRT1 | pending interrupt for UART1 in transmission |
| 2 to 0 | IPLT1.2 to IPLT1.0 | interrupt priority level for UART1 in transmission |

### 10.1.3 $I^2C$-BUS AND ADC INTERRUPT REGISTER (PICR3)

The $I^2C$-bus and the ADC respectively, can generate one interrupt.

**Table 38** $I^2C$-bus and ADC Interrupt Register (address FFFF 8707H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PIRI | IPLI2 | IPLI1 | IPLI0 | PIRA | IPLA2 | IPLA1 | IPLA0 |

**Table 39** Description of PICR3 bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | PIRI | pending interrupt for $I^2C$-bus |
| 6 to 4 | IPLI2 to IPLI0 | interrupt priority level for $I^2C$-bus |
| 3 | PIRA | pending interrupt for ADC |
| 2 to 0 | IPLA2 to IPLA0 | interrupt priority level for ADC |

## 11 TIMERS

### 11.1 Timer array

Two identical 16-bit timer blocks are provided:

- Timer 0 (T0)
- Timer 1 (T1).

Each timer block consists of:

- A timebase
- Three capture/compare channels
- A Control Register
- A Status Register.

### 11.2 Timebase

The timebase contains an 8-bit prescaler with a write only reload register, and a 16-bit counter register. This counter register can only be read by software. The prescaler is clocked either by the peripheral clock FCLK or by an external clock enabled by the flag C/TN in the timer control register TnCR (T0CT for timer T0 and T1CR for timer T1). On prescaler overflow the prescaler reload value is loaded into the prescaler, which starts incrementing.

The 16-bit counter register is incremented at each prescaler overflow. When the counter reaches FFFFH, the status flag TOV is set and on the next clock the counter reload value is loaded into the counter. By resetting the control bit RUN in the timer control register the timebase is stopped, and by setting this bit, the prescaler and counter are reloaded and incremented on the next external or internal clock.

### 11.3 Channel function

Each channel consists of a register and an equality comparator. For each of the three channels two modes can be selected:

- **Compare mode:** sets the status flag CFn in TnSR when there is a match between the counter register and the channel register value.
- **Capture mode:** stores the counter register value into the channel register and sets the status flag CFn when a transition occurs at the corresponding input pin CPn.

In both modes, each channel can generate a global interrupt request if the corresponding enable bit in the Control Register TnCR is set.

### 11.4 Pin parallel functions for the timer

In order to use the multiplexed pins for the timer, the other functions using these pins as output pins should be forced HIGH via a weak pull-up, enabling an external source to drive them LOW.

**Table 40** Parallel functions

| PIN | SETTING | PARALLEL FUNCTION |
|---|---|---|
| SP1/TX1/$\overline{\text{INT1}}$ | if SPCON.1 = 0, SPR.1 = 1; else UART1 should not be used | CLK0 |
| SP2/RX0/$\overline{\text{INT2}}$ | if SPCON.2 = 0, SPR.2 = 1; else UART0 should not be used | CP2 |
| SP3/TX0/$\overline{\text{INT3}}$ | if SPCON.3 = 0, SPR.3 = 1; else UART0 should not be used | CP3 |
| SP4/$\overline{\text{INT4}}$ | if SPCON.4 = 0, SPR.4 = 1 | CP4 |
| SP5/$\overline{\text{INT5}}$ | if SPCON.5 = 0, SPR.5 = 1 | CP5 |
| SP6/$\overline{\text{INT6}}$ | if SPCON.6 = 0, SPR.6 = 1 | CLK1 |
| P8/PWM0 | if PCON.0 = 0, PWM0 should output a logic 1 (write 00H to register PWM0) | CP0 |
| P9/PWM1 | if PCON.1 = 0, PWM1 should output a logic 1 (write 00H to register PWM1) | CP1 |

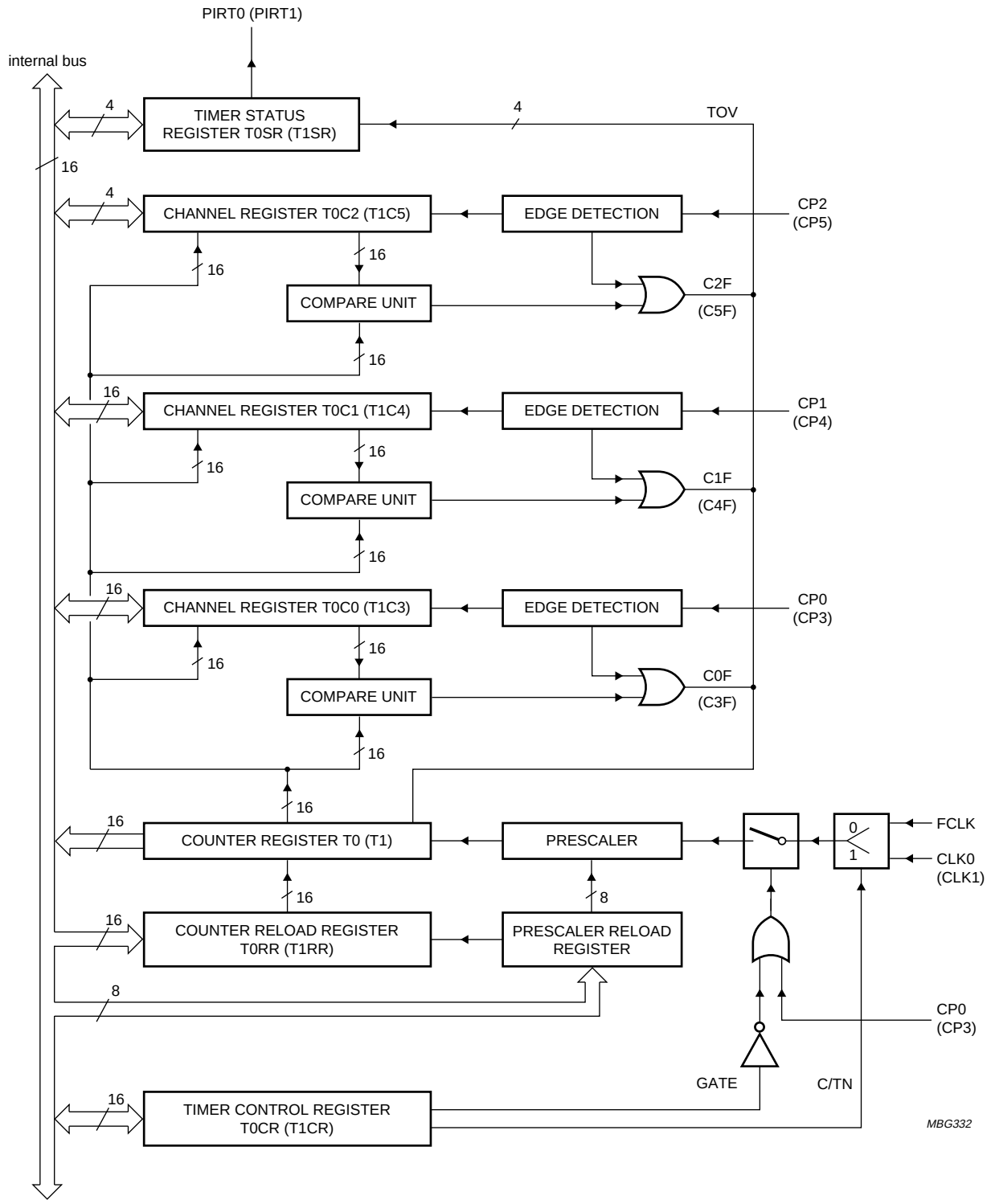



Fig.10 Timer block diagram T0 (identical with timer block T1, corresponding names indicated within brackets).

### 11.5 Timer Control Registers

The Timer 0 (T0) is controlled via Timer 0 Control Registers (T0CRH and T0CRL), and Timer 1 (T1) via Timer 1 Control Registers (T1CRH and T1CRL); see Fig.10 and Tables 41 to 44. The default value after a CPU reset for all bits of T0CRH; T1CRH; T0CRL and T1CRL is a logic 0.

**Table 41** Timer Control Registers T0CRH and T1CRH

| ADDRESS | REGISTER | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| FFFF 8300H | T0CRH | ECM2 | C2M2 | C2M1 | C2M0 | ECM1 | C1M2 | C1M1 | C1M0 |
| FFFF 8310H | T1CRH | | | | | | | | |

**Table 42** Timer Control Registers T0CRL and T1CRL

| ADDRESS | REGISTER | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FFFF 8301H | T0CRL | ECM0 | C0M2 | C0M1 | C0M0 | ETOV | GATE | C/TN | RUN |
| FFFF 8311H | T1CRL | | | | | | | | |

**Table 43** Description of T0CRH; T1CRH; T0CRL and T1CRL bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 15, 11 and 7 | ECM2 to ECM0 | **Channel n interrupt enable** (n = 0 to 2);<br>ECMn = 0, the channel n interrupt is disabled;<br>ECMn = 1, the channel n interrupt is enabled. |
| 14 to 12 | C2M2 to C2M0 | **Channel mode**; see Table 44. |
| 10 to 8 | C1M2 to C1M0 | |
| 6 to 4 | C0M2 to C0M0 | |
| 3 | ETOV | **Timer overflow interrupt enable**;<br>ETOV = 0, the timer overflow interrupt is disabled;<br>ETOV = 1, the timer overflow interrupt is enabled. |
| 2 | GATE | **Gated external clock**;<br>GATE = 0, disable gate function;<br>GATE = 1, the prescaler increments only if the CP0 pin is HIGH for each rising edge transition of CLK0 if C/TN = 1 or with FCLK if C/TN = 0. |
| 1 | C/TN | **Counter/timer mode;**<br>C/TN = 0, timer mode; the prescaler is incremented on the rising edge of the peripheral clock (FCLK);<br>C/TN = 1, counter mode; the prescaler increments on the rising edge of CLK0 for Timer 0 (CLK1 for Timer 1). |
| 0 | RUN | **Timer run enable**;<br>RUN = 0, timer prescaler stopped and registers value held;<br>RUN = 1, when set the prescaler and counter are loaded and the prescaler is then incremented. |

**Table 44** Description of channel mode; n = 0 to 5; X = don't care

| CnM2 | CnM1 | CnM0 | DESCRIPTION |
|---|---|---|---|
| 0 | 0 | 0 | channel n inhibited |
| 0 | 0 | 1 | channel n capture on LOW-to-HIGH transition of pin CPn |
| 0 | 1 | 0 | channel n capture on HIGH-to-LOW transition of pin CPn |
| 0 | 1 | 1 | channel n capture on any transitions of pin CPn |
| 1 | X | X | channel compare mode |

### 11.6 Timer Status Registers

Four events can occur: a timer overflow or three channel matches/captures. These event flags are stored in the 4-bit Timer 0 Status Register (T0SR for T0) and Timer 1 Status Register (T1SR for T1). They can be cleared by software but cannot be set. By writing a logic 1 the flags stay unchanged. In order to clear a particular flag one has to write a logic 0 to the corresponding position and logic 1s to the others. One should avoid to use the instruction BCLR, which can reset accidentally several flags.

#### 11.6.1 Timer 0 Status Register (T0SR)

**Table 45** Timer 0 Status Register (address FFFF 830DH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | C2F | C1F | C0F | TOV |

**Table 46** Description of T0SR bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 4 | – | Reserved. |
| 3 to 1 | C2F to C0F | Channel n event flag (n = 2 to 0); CnF = 0, no event (default value after a CPU reset). CnF = 1, capture mode: a capture occurred. |
| 0 | TOV | Timer Overflow Flag; TOV = 0, no overflow (default value after a CPU reset). TOV = 1, timer overflow occurred. |

#### 11.6.2 Timer 1 Status Register (T1SR)

**Table 47** Timer 1 Status Register (address FFFF 831DH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | C5F | C4F | C3F | TOV |

**Table 48** Description of T1SR bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 4 | – | Reserved. |
| 3 to 1 | C5F to C3F | Channel n event flag (n = 5 to 3); CnF = 0, no event (default value after a CPU reset). CnF = 1, capture mode: a capture occurred. |
| 0 | TOV | Timer Overflow Flag; TOV = 0, no overflow (default value after a CPU reset). TOV = 1, timer overflow occurred. |

### 11.7 Watchdog Timer

The P90CL301BFH contains a Watchdog Timer consisting of a 13-bit prescaler and an 8-bit timer WDTIM.
The prescaler is incremented by the peripheral clock.
The 8-bit timer is incremented every 8192 cycles of the peripheral clock FCLK.

If the FCLK frequency is 2 MHz, the Watchdog Timer can operate in the range of 4.1 ms up to 1 s. The Watchdog Timer is disabled after reset. It can be enabled by writing any value to the WDCON register. The only way to disable a running Watchdog Timer is to reset the P90CL301BFH.

When a timer overflow occurs the microcontroller will be reset (except registers SYSCON, PCON, PRL and PRH which will not be reset). To prevent an overflow of the Watchdog Timer, the User Program must reload the Watchdog register within a period shorter than the programmed timer interval.

This timer interval is determined by the 8-bit timer value written to the register WDTIM.

For FCLK in MHz, the Watchdog period is:

$$(256 - WDTIM) \times \frac{8192}{FCLK}\ \mu s$$

The Watchdog Timer is controlled by the register WDCON. A value of A5H in WDCON clears both the prescaler and timer WDTIM. After reset, WDCON contains A5H.

Every value other than A5H in WDCON enables the Watchdog Timer. Since the bit 0 of the WDCON input is tied to a logic 0 by hardware during write operations on WDCON, the reset value A5H can not be programmed again and can only be restored by a reset.

Timer WDTIM can be written only if WDCON has previously been loaded with 5AH, otherwise WDTIM and the prescaler are not affected. A successful write operation to WDTIM also clears the prescaler and clears WDCON.

Only the values A5H or 5AH are stored, all other values are stored with a dummy value 00H.

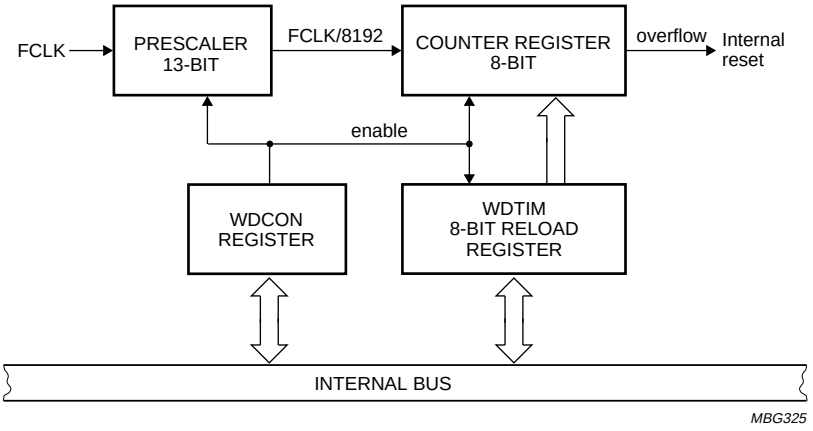



Fig.11 Watchdog Timer block diagram.

## 12 SERIAL INTERFACES

### 12.1 UART interface

The UART can operate in 4 modes. The baud rate for receive and transmit can be generated internally or by the baud rate generator. The UART is full duplex, meaning it can receive and transmit simultaneously. The receive and transmit registers are both accessed as a unique register SBUF. Writing to SBUF loads the transmit register, and reading from SBUF accesses a physically separate receive register.

#### 12.1.1 UART OPERATING MODES

The serial port can operate in one of the four modes:

Mode 0 Serial data enters and exits through RXD. TXD pin delivers the synchronous shift clock. 8 bits are transmitted/received (LSB first). When the bit PCLK3 in the SYSCON register is reset, the baud rate is equal to $1/6 \times$ CLK. When the bit PCLK3 in register SYSCON is set, the baud rate is equal to $1/12 \times$ CLK. The UART baud rate should not exceeds 4.5 Mbaud.

Mode 1 10 bits are transmitted (through TXD) or received (through RXD): a start bit at logic 0, 8 data bits (LSB first) and a stop bit at logic 1. On receive the stop bit goes into RB8 in the register SCON. The baud rate is given by the baud rate generator output BGCLK0 for the UART0 and BGCLK1 for the UART1.

Mode 2 11 bits are transmitted (through TXD) or received (through RXD): a start bit at logic 0, 8 data bits (LSB first) a programmable 9$^{th}$ data bit, and a stop bit at logic 1. On transmit the 9$^{th}$ bit is taken from the bit TB8 from the SCON register. On receive the 9$^{th}$ bit goes into RB8 of SCON, while the stop bit is ignored. The baud rate is equal to $1/6 \times$ CLK. The UART clock should not exceed 4.5 Mbaud.

Mode 3 Same as mode 2 except for the baud rate, which is given by the baud rate generator output BGCLK0 for the UART0 and BGCLK1 for the UART1.

In all four modes, transmission is initiated by any instruction loading SBUF. In Mode 0, reception is initiated by the condition RI = 0 and REN = 1. In the remaining modes reception is initiated by the incoming start bit if REN = 1.

### 12.1.2 UART CONTROL REGISTERS SCON0 AND SCON1

The registers SCON0 and SCON1 control UART0 and UART1 modes respectively, and contain the interrupt flags.

**Table 49** UART Control Registers SCON0 and SCON1

| ADDRESS | REGISTER | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FFFF 8603H | SCON0 | SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |
| FFFF 8607H | SCON1 | | | | | | | | |

**Table 50** Description of register SCON0 and SCON1 bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 6 | SM0 to SM1 | Mode bits; see Table 51. |
| 5 | SM2 | **Multiprocessor**; enable the multiprocessor communication feature in Modes 2 and 3. If SM2 is set the RI will not be activated if the received 9$^{th}$ data bit RB8 = 0. In Mode 1, if SM2 is set the RI will not be activated if a valid stop bit is not received. In Mode 0, SM2 should be a logic 0. |
| 4 | REN | **Receive enable**; enables serial reception; set and cleared by software. |
| 3 | TB8 | **Transmit extra bit**; 9$^{th}$ data bit that will be transmitted in Modes 2 and 3; set and cleared by software. |
| 2 | RB8 | **Receive extra bit**; in Modes 2 and 3, RB8 is the 9$^{th}$ bit received. In Mode 1, if SM2 = 0, RB8 is the stop bit which is received. |
| 1 | TI | **Transmit interrupt**; it is set by hardware at the end of the 8$^{th}$ bit time in Mode 0, or halfway through the stop bit in the other modes (except: see bit SM2). TI must be cleared by software (cannot be set by software). By writing a logic 1 the flags stay unchanged. In order to clear a particular flag one has to write a logic 0 to the corresponding position and a logic 1 to the others. One should avoid to use the instruction BCLR, which can reset accidentally several flags. |
| 0 | RI | **Receive interrupt**; set by hardware at the end of the 8$^{th}$ bit time in Mode 0, or halfway through the stop bit in the other modes (except: see SM2). RI must be cleared by software (cannot be set by software). By writing a logic 1 the flags stay unchanged. In order to clear a particular flag one has to write a logic 0 to the corresponding position and a logic 1 to the others. One should avoid to use the instruction BCLR, which can reset accidentally several flags. |

**Table 51** Mode defined by bits SM0 and SM1

| SM0 | SM1 | MODE | DESCRIPTION |
|---|---|---|---|
| 0 | 0 | 0 | shift register; $\frac{1}{6} \times$ CLK |
| 0 | 1 | 1 | 8-bit UART; BGCLK0 and BGCLK1 |
| 1 | 0 | 2 | 9-bit UART; $\frac{1}{16} \times$ CLK |
| 1 | 1 | 3 | 9-bit UART; BGCLK0 and BGCLK1 |

### 12.2 Baud rate generator

A dedicated baud rate generator is directly connected to the UART0. For the UART1 this clock can be divided by 1 or 4 as a function of the bit BDIV in the BCON control register.

The baud rate generator consists of a 16-bit timer, two 8-bit registers BREGL (least significant byte) and BREGH (most significant byte) to store the 16-bit reload value, and a control register BCON.

When an overflow occurs the timer is reloaded with the contents of the registers BREGH, BREGL.

The timer is clocked by the peripheral clock. The baud rates for UART0 and UART1 in Mode 1 and 3 are determined by the timer overflow rate as follows (FCLK is in Hz):

$$BGCLK0 = \frac{FCLK}{(16 x (65536 - BREG))}$$

$$BGCLK1 = \frac{FCLK}{\left(16 \times (65536 - BREG) x 4^{BDIV}\right)}$$

#### 12.2.1 UART Baud Rate Control Register (BCON)

The default value after a CPU reset for all bits of BCON is a logic 0.

**Table 52** UART Baud Rate Control Register (address FFFF 860FH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | BST | BDIV |

**Table 53** Description of BCON bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 2 | – | Reserved. |
| 1 | BST | BST = 0, stop timer; BST = 1, start timer increment after loading of timer register with the reload register value. |
| 0 | BDIV | BDIV = 0, UART1 baud rate not divided; BDIV = 1, UART1 baud rate divided by 4. |

### 12.3 UART queue

The UART queue performs the sending and receiving of a frame of bytes of variable length through the UART without the support of the CPU. Only the UART0 has a frame buffer located at the lower 256 bytes section of the internal RAM. A controller ensures the sequencing of the transfers between the RAM and the UART and generates interrupts to the CPU. This UART queue can be used for transmission and reception simultaneously or for only one of the two modes.

The RAM can be accessed by the CPU any time.
The queue controller accesses the RAM either in read mode for the transmission or in write mode for the reception. When the queue controller accesses the RAM, the CPU waits for the end of the access cycle (maximum 4 CLK clocks). The RAM space can be partitioned in one or several buffers for transmission or reception or for normal data storage. The maximum size of a buffer is limited to 256 bytes. In addition to these buffers the queue consists of a set of control and data registers:

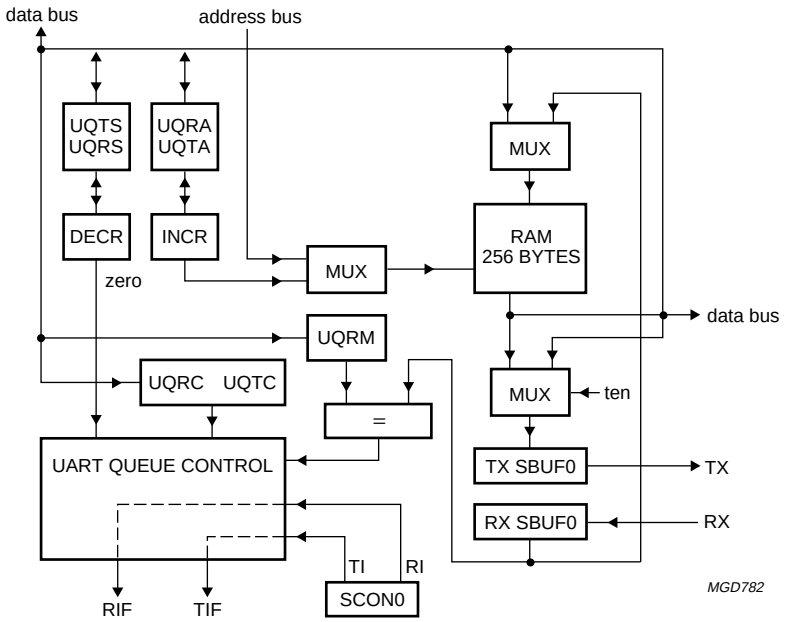



Fig.12 UART queue block diagram.

**Table 54** Function of UART queue registers

| NAME | FUNCTION | DESCRIPTION | SIZE |
|---|---|---|---|
| UQRC(1) | Reception Control Register | Reception control and status flags. | byte |
| UQTC(1) | Transmission Control Register and Interrupt Flags. | Transmission control and status flags and interrupt flags. | byte |
| UQTA(2) | Transmit Buffer Address Register | Start address of transmission buffer from 00H to FFH, corresponds to CPU address from FFFF 9000H to FFFF 90FFH. | byte |
| UQTS (2) | Transmit Buffer Size Register | Size of the transmission buffer. Limited to 256 bytes. | byte |
| UQRA(3) | Reception Buffer Address Register | Start address of reception buffer from 00H to FFH, corresponds to CPU address from FFFF 9000H to FFFF 90FFH. | byte |
| UQRS(3) | Reception Buffer Size Register | Size of the reception buffer. Limited to 256 bytes. | byte |
| UQRM | Reception Match Register | The received characters are compared with the value contained in this register and an interrupt is generated when they are equal. | byte |

**Notes**

1. UQRC and UQTC can be accessed together as a word or as two bytes.
2. For each byte transmitted the UQTA is incremented, the UQTS is decremented.
3. For each byte received the UQRA is incremented, the UQRS is decremented.The CPU can read this register on the fly, but in this case the accuracy is not guaranteed so it is recommended to halt the queue and read the values.

### 12.3.1 Reception Control Register (UQRC)

In order to keep the bit unchanged when writing to the control register, it is recommended to write a logic 1 when it can only be reset, and to write a logic 0 when it can only be set. After peripheral reset all bits are set to a logic 0.

**Table 55** Reception Control Register (address FFFF 8B00H)

| ACTION OF | BIT 7 | BIT 6 | BIT 5 | BIT 4 | BIT 3 | BIT 2 | BIT 1 | BIT 0 |
|---|---|---|---|---|---|---|---|---|
| | REN | RME | RIE | ROE | ROF | RAR | RHLT | RSTF |
| CPU[1] | S/R | S/R | S/R | S/R | R | S/R | S/R | S |
| QUEUE[2] | – | – | – | – | S | – | – | R |

**Notes**

1. CPU. R: the CPU can reset this bit. S: the CPU can set this bit.
2. QUEUE. R: the queue controller can reset this bit. S: the queue controller can set this bit.

**Table 56** Description of UQRC bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | REN | **Receive queue enable**. This bit enables the queue controller. It connects the reception data buffer SBUF0 to the queue controller. It should be set before activating the RSTF bit. When it is reset SBUF0 can be accessed directly by the CPU. REN = 0 means receive queue disable. Received byte can be read directly from SBUF0. REN = 1 means receive queue enable: The transfers from the SBUF0 to the RAM can be activated by setting the bit RSTF. |
| 6 | RME | **Reception match enable**. If it is set each received byte is compared with the content of the UART Queue Receive Match register (UQRM) and if their value match the receive interrupt flag RIF is set. RME = 0 means match function disabled. RME = 1 means match function enabled. |
| 5 | RIE | **Reception interrupt enable**. When this bit is set, each time a byte is received the receive interrupt flag RIF is set. If it is not set, an interrupt is only generated at the end of the frame. RIE = 0 means no interrupt after the reception of each byte, only at the end of the frame. RIE = 1 means interrupt after the reception of each byte. |
| 4 | ROE | **Reception overflow enable**. When this bit is set, the RSTF bit is not reset when the reception buffer size reached 0, setting the RIF flag, so the reception of further bytes is allowed. The bit ROF is not set because RSTF stays set. This bit can be set in conjunction of RAR to implement a circular buffer. ROE = 0 means no overflow enable. ROE = 1 means overflow enable. |
| 3 | ROF | **Reception overflow flag**. This flag is set by the queue controller, when a character is received with the RSTF flag reset and REN set. This event can occur after the end of reception of a frame, and if the CPU had no time to unload the RAM and set RSTF. ROF = 0 means no overflow detection. ROF = 1 means overflow. |

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 2 | RAR | **Reception address reset**. If this flag is set, when the buffer size has been decremented to zero, the reception address is reset. This way a circular reception buffer can be located at address 0. RAR = 0 means no reset of reception address. RAR = 1 means reset of reception address. |
| 1 | RHLT | **Reception halt**. This bit is set by the CPU to interrupt the reception of the frame. The byte currently received by the UART will be stored in the buffer, but the next bytes will be lost until the CPU reset the bit RHLT. In order to stop all activity in the UART it is preferable to reset the bit REN reception enable of the register SCON0. RHLT = 0 means reception not halted. RHLT = 1 means reception halted. |
| 0 | RSTF | **Reception start flag**. This bit is set by the CPU to enable the reception of a frame through the UART and it is reset automatically by the queue controller at the end of reception. When RHLT is set this bit stays set. When REN is reset, this bit is reset. RSTF = 0 means reception not started or ended. RSTF = 1 means reception started and in progress. |

### 12.3.2 Transmission Control Register and Interrupt Flags (UQTC)

**Table 57** Transmission Control Register and Interrupt Flags (address FFFF 8B01H)

| ACTION OF | BIT 7 | BIT 6 | BIT 5 | BIT 4 | BIT 3 | BIT 2 | BIT 1 | BIT 0 |
|---|---|---|---|---|---|---|---|---|
| | TIF | RIF | reserved | TIWF | TEN | TIE | THLT | TSTF |
| CPU[1] | R | R | – | – | S/R | S/R | S/R | S |
| QUEUE[2] | S | S | – | S/R | – | – | – | R |

**Notes**

1. CPU. R: the CPU can reset this bit. S: the CPU can set this bit.
2. QUEUE. R: the queue controller can reset this bit. S: the queue controller can set this bit.

**Table 58** Description of UQTC bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 | TIF | **Transmission interrupt flag**. This flag is set either at the end of the transmission buffer or at the transmission of each byte if TIE is set. The TIF flag should be reset by the CPU in the exception routine in order to detect further interrupts as they are edge detected for LOW-to-HIGH transitions. |
| 6 | RIF | **Reception interrupt flag**. This flag is set either at the end of the reception buffer or during a character match if RME is set or at the reception of each byte if RIE is set. The RIF flag should be reset by the CPU in the exception routine in order to detect further interrupts as they are edge detected for LOW-to-HIGH transitions. |
| 5 | – | Reserved. |
| 4 | TIWF | **Transmission interrupt waiting**. TIWF = 0[1] means queue controller is not waiting for UART transmit interrupt.TIWF = 1 means queue controller is waiting for UART transmit interrupt. |
| 3 | TEN | **Transmission queue enable**. TEN = 0[1] means transmission queue disable. Transmitted byte can be written directly into SBUF0. TEN = 1 means transmission queue enable; the transfers from the RAM to SBUF0 can be activated by setting the bit TSTF. |

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 2 | TIE | **Transmission interrupt enable**. If it is set, each time a byte is transmitted the transmit interrupt flag TIF is set. If it is not set, an interrupt is only generated at the end of the frame. TIE = 0(1) means no interrupt after the reception of each byte. TIE = 1 means interrupt after the reception of each byte. |
| 1 | THLT | **Halt transmission**. This bit is set by the CPU to interrupt the transmission of the frame. The byte currently loaded in the UART will be transmitted entirely, but the next byte will wait until the CPU reset the bit HLTT. THLT = 0(1) means transmission not halted. THLT = 1 means transmission halted. |
| 0 | TSTF | **Start transmission**. This bit is set by the CPU to start the transmission of a frame through the UART and it is reset automatically by the queue controller at the end of transmission. TSTF = 0(1) means transmission not started or ended. TSTF = 1 means transmission started and in progress. |

**Note**

1. State after peripheral reset.

### 12.3.3 UART QUEUE REGISTERS

**Table 59** UART Queue Registers

| REGISTER | ADDRESS | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| UQTA | FFFF 8B03H | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| UQTS | FFFF 8B05H | S7 | S6 | S5 | S4 | S3 | S2 | S1 | S0 |
| UQRA | FFFF 8B07H | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 |
| UQRS | FFFF 8B09H | S7 | S6 | S5 | S4 | S3 | S2 | S1 | S0 |
| UQRM | FFFF 8B0BH | M7 | M6 | M5 | M4 | M3 | M2 | M1 | M0 |

### 12.3.4 UART QUEUE OPERATION: TRANSMISSION

The UART queue transmit operation is as follows:

1. The UART control register is initialized for a certain transmission mode (0, 1, 2 and 3) and the baud rate generator loaded for a defined baud rate.
2. The CPU loads the data to be transmitted (for example 80 characters) at successive addresses of the internal RAM starting at a certain base address (for example FFFF 9010H). Then it writes the buffer start address and the buffer size in the pointer registers, and initializes the control register.
3. The queue controller reads the byte at the address pointed by the address register and writes it to the transmit data buffer of the UART and the buffer size register is decremented, the address register is incremented pointing to the next byte in the buffer. The transmission starts. The controller waits for the end of transmission, then compares the buffer size value to zero, if they are not equal the same operation is repeated automatically.
4. If the buffer size is zero the transmit interrupt flag TIF is set issuing an interrupt to the CPU.The interrupt routine should reset TIF and can reload the buffer with other values.
5. Before checking the buffer size value, the halt bit THLT is tested and if it is set the controller enters a transmission wait state.

**Table 60** Transmission routine

```
move.b   #$50,UQTS    ;buffer size
move.b   #$10,UQTA    ;buffer start address
bset     TEN,UQTC;    ;Enable transmission queue
bset     STF,UQTC;    ;Start transmission.
```

### 12.3.5 UART QUEUE OPERATION: RECEPTION

The UART queue reception operation is as follows:

The UART control register is initialized for a certain reception mode (Mode 0, 1, 2 and 3) and the baud rate generator loaded for a defined baud rate.

The CPU writes the buffer start address and the buffer size in the data registers, and the control register. Several modes can be used:

#### *12.3.5.1 Mode 0: Normal reception buffer.*

We want to receive 80 characters, store then in a buffer starting at the address FFFF 9020H and generate an interrupt. The CPU is able to down-load the 80 characters, before the reception of any further character.

After reception of the first character the queue controller reads the data reception register SBUF0 and transfers it's contents into the buffer at the address of the UQRA register, at the same time the buffer size register UQRS is decremented, the address register UQRA is incremented to point to the next byte. If the buffer size is not equal to zero the same operation is repeated automatically for the next byte to be transmitted.

If the buffer size is zero the receive interrupt flag RIF is set issuing an interrupt to the CPU. The interrupt routine should reset RIF and can read the content of the buffer and re-initialize the control registers.

**Table 61** Reception routine

```
move.b   #$50, UQRS    ;set buffer size
move.    #$20, UQRA    ;set buffer start address
bset     REN, UQRC     ;Enable queue controller
bset     RSTF, UQRC    ;Start reception.
```

**Table 62** Interrupt routine

```
         move.b   #$BE,UQTC          ;reset RIF bit
         move.b   #$28,d0            ;buffer size in words
         move.l   #$FFFF9020,a0      ;buffer start address
L1       move.l   #$00008000,a1      ;external memory
                                     start address
         dbne     d0,L1              ;loop
```

*12.3.5.2 Mode 1: Special termination character match.*

Suppose that we want to generate an interrupt after the reception of a Carriage Return character, we load in the reception match register the value 0DH, to guarantee that the buffer does not overflow if the buffer size is limited to 80 characters. The buffer is located in RAM at the address FFFF 9050H.

The same operations as described before are performed but in addition each received characters compared with the character Carriage Return and if they match the receive interrupt flag RIF is set, RSTF is reset and the reception queue is stopped.

**Table 63** Mode 1 routine

| | | |
|---|---|---|
| move.b | #$50,UQRS | ; buffer size |
| move.b | #$50,UQRA | ; buffer start address |
| move.b | #$0d,UQRM | ; set match character |
| bset | REN,UQRC | ; enable queue |
| bset | RME,UQRC | ; reception match enable |
| bset | RSTF,UQRC | ; start reception (note 1) |

**Note**

1. All these control bits can be set at the same time.

*12.3.5.3 Mode 2: Linear buffer with continuous reception.*

If we want to continue to receive characters in the buffer after the end of the buffer and the setting of RIF:

In this case RSTF is not reset at the end of the buffer, but the CPU will receive an interrupt (RIF = 1) when the size register UQRS equals zero.

**Table 64** Mode 2 routine

| | | |
|---|---|---|
| move.b | #$50,UQRS | ; buffer size |
| move.b | #$50,UQRA | ; buffer start address |
| bset | REN,UQRC | ; enable queue |
| bset | ROE,UQRC | ; reception overflow enable |
| bset | RSTF,UQRC | ; start reception (note 1) |

**Note**

1. All these control bits can be set at the same time.

*12.3.5.4 Mode 3: Circular buffer with interrupt.*

If we want to implement a circular buffer which generates an interrupt each time the size register is equal to 0, the UQRA address register is reset and points to the beginning of the RAM.

**Table 65** Mode 3 routine

| | | |
|---|---|---|
| move.b | #$50,UQRS | ; buffer size |
| move.b | #$00,UQRA | ; buffer start address |
| bset | REN,UQRC | ; enable queue |
| bset | RAR,UQRC | ; reception reset address |
| bset | RSTF,UQRC | ; start reception (note 1) |

**Note**

1. All these control bits can be set at the same time.

### 12.3.6 UART QUEUE OPERATION: RECEPTION HALT

Before to check the buffer size value, the halt bit HLTR0 is tested and if it is set the controller enters a reception wait state.

### 12.3.7 UART QUEUE OPERATION: EMULATION

When the pin $\overline{\text{PHALT}}$ (on the emulation package) is asserted LOW, the queue is halted the same way as when THLT and RHLT are set. The queue operation is continued when the pin $\overline{\text{PHALT}}$ is released HIGH.

### 12.4 I²C-bus interface

The serial port supports the twin line I²C-bus. The I²C-bus consists of a data line SDA and a clock line SCL. These lines also function as I/O port lines P11 and P10 respectively (always open drain). The system is unique because data transport, clock generation, address recognition and bus control arbitration are all controlled by hardware. The I²C-bus serial I/O has complete autonomy in byte handling and operates in four modes:

- Master transmitter mode
- Master receiver mode
- Slave transmitter mode
- Slave receiver mode.

These functions are controlled by the SCON register. SSTA is the Status Register whose contents may be used as a vector to various service routines. SDAT is the data shift register and SADR the slave address register. Slave address recognition is performed by hardware.

For more details on the I²C-bus functions, see user manual *"The I²C-bus and how to use it (including specifications)"*; order number 9398 393 40011.

### 12.5 Serial Control Register (SCON)

**Table 66** Serial Control Register (address FFFF 8207H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CR2 | ENS | STA | STO | SI | AA | CR1 | CR0 |

**Table 67** Serial Control Register SCON bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7, 1 and 0 | CR2 to CR0 | These three bits determine the serial clock frequency when SIO is in a master mode function of the peripheral clock FCLK (see Tables 68 and 69). |
| 6 | ENS | **Enable serial I/O**. If ENS = 0, the serial interface I/O is disabled and reset; if ENS = 1, the serial interface is enabled. |
| 5 | STA | **Start flag**. When this bit is set in slave mode, the hardware checks the I²C-bus and generates a START condition if the bus is free or after the bus becomes free. If the device operates in master mode it will generate a repeated START condition. |
| 4 | STO | **Stop flag**. If this bit is set in the master mode a STOP condition is generated. A STOP condition detected on the I²C-bus clears this bit. The STOP bit may also be set in slave mode in order to recover from an error condition. In this case no STOP condition is generated to the I²C-bus, but the hardware releases the SDA and SCL lines and switches to the not selected slave receiver mode. The STOP flag is cleared by the hardware. |
| 3 | SI | **Serial Interrupt flag**. This flag is set, and an interrupt is generated, after any of the following events occur:<br>• A START condition is generated in master mode.<br>• The own slave address has been received during AA = 1.<br>• The general call address has been received while bit SADR.0 = 1 and AA = 1.<br>• A data byte has been received or transmitted in master mode.<br>• A data byte has been received or transmitted as selected slave.<br>• A STOP or START condition is received as selected slave receiver or transmitter.<br>While the SI flag is set, SCL remains LOW and the serial transfer is suspended. SI must be reset by software. |

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 2 | AA | **Assert Acknowledge** When this bit is set, an acknowledge is returned after any one of the following conditions:<br>• Slave address is received.<br>• The general call address is received (bit SADR.0 = 1).<br>• A data byte is received, while the device is programmed to be a master receiver.<br>• A data byte is received, while the device is a selected slave receiver.<br>When bit AA is reset, no acknowledgement is returned. Consequently, no interrupt is requested when the own slave address or general call address is received. |

**Table 68** CLK/SCL divide factor

Values greater than 100 kbits are outside the specified frequency range.

| CR2 | CR1 | CR0 | CLK/SCL DIVIDE FACTOR | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | D = 2(1) | D = 3 | D = 4 | D = 5 | D=6 | D=8 | D=10 |
| 0 | 0 | 0 | 128 | 192 | 256 | 320 | 384 | 512 | 640 |
| 0 | 0 | 1 | 112 | 168 | 224 | 280 | 336 | 448 | 560 |
| 0 | 1 | 0 | 96 | 144 | 192 | 240 | 288 | 384 | 480 |
| 0 | 1 | 1 | 80 | 120 | 160 | 200 | 240 | 320 | 400 |
| 1 | 0 | 0 | 480 | 720 | 960 | 1200 | 1440 | 1920 | 2400 |
| 1 | 0 | 1 | 60 | 90 | 120 | 150 | 180 | 240 | 300 |
| 1 | 1 | 0 | 30 | 45 | 60 | 75 | 90 | 120 | 150 |

**Table 69** $I^2C$-bus serial clock rates

Values greater than 100 kbits are outside the specified frequency range.

| CR2 | CR1 | CR0 | BIT FREQUENCY (kHz) AT CLK = 26 MHz | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | D = 2(1) | D = 3 | D = 4 | D = 5 | D = 6 | D = 8 | D = 10 |
| 0 | 0 | 0 | – | – | 101 | 81 | 68 | 51 | 41 |
| 0 | 0 | 1 | – | – | – | 93 | 77 | 58 | 46 |
| 0 | 1 | 0 | – | – | – | – | 90 | 68 | 54 |
| 0 | 1 | 1 | – | – | – | – | – | 81 | 65 |
| 1 | 0 | 0 | 54 | 36 | 27 | 22 | 18 | 13 | 10 |
| 1 | 0 | 1 | – | – | – | – | – | – | 87 |
| 1 | 1 | 0 | – | – | – | – | – | – | – |

**Note to Tables 68 and 69**

1. D = divisor = $^{CLK}/_{FCLK}$; see Table 15.

### 12.5.1 I²C-BUS STATUS REGISTER (SSTA)

SSTA is an 8-bit read only Special Function Register. The contents of SSTA may be used as a vector to a service routine. This optimizes response time of the software and consequently that of the I²C-bus. Tables 73 to 77 show the list of the status codes defined by the contents of register SSTA.

**Table 70** I²C-bus Status Register (address FFFF 8205H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SC4 | SC3 | SC2 | SC1 | SC0 | – | – | – |

**Table 71** Description of SSTA bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 3 | SC4 to SC0 | The bits SC4 to SC0 hold a status code. |
| 2 to 0 | – | Reserved; held LOW. |

**Table 72** Used abbreviations in the mode descriptions; see Tables 73 to 77

| SYMBOL | DESCRIPTION |
|---|---|
| SLA | 7-bit slave address |
| R | read bit |
| W | write bit |
| ACK | acknowledgement (acknowledge bit = 0) |
| ACKNOT | not acknowledge (acknowledge bit = 1) |
| DATA | 8-bit (byte) to or from the I²C-bus |
| MST | master |
| SLV | slave |
| TRX | transmitter |
| REC | receiver |

**Table 73** Master transmitter (MST/TRX) mode

| SSTA VALUE | DESCRIPTION |
|---|---|
| 08H | A START condition has been transmitted |
| 10H | A repeated START condition has been transmitted |
| 18H | SLA and W have been transmitted, ACK has been received |
| 20H | SLA and W have been transmitted, ACKNOT received |
| 28H | DATA of S1DAT has been transmitted, ACK received |
| 30H | DATA of S1DAT has been transmitted, ACKNOT received |
| 38H | Arbitration lost in SLA, R/$\overline{W}$ or DATA |

**Table 74** Master receiver (MST/REC) mode

| SSTA VALUE | DESCRIPTION |
|---|---|
| 38H | Arbitration lost while returning ACKNOT |
| 40H | SLA and R have been transmitted, ACK received |
| 48H | SLA and R have been transmitted, ACKNOT received |
| 50H | DATA has been received, ACK returned |
| 58H | DATA has been received, ACKNOT returned |

**Table 75** Slave transmitter (SLV/TRX) mode

| S1STA VALUE | DESCRIPTION |
|---|---|
| A8H | Own SLA and R received, ACK returned |
| B0H | Arbitration lost in SLA, R/$\overline{W}$ as MST. Own SLA and R received, ACK returned |
| B8H | DATA byte has been transmitted, ACK received |
| C0H | DATA byte has been transmitted, ACK received |
| C8H | Last DATA byte has been transmitted, ACKNOT received |

**Table 76** Slave receiver (SLV/REC) mode

| SSTA VALUE | DESCRIPTION |
|---|---|
| 60H | Own SLA and W have been received, ACK returned |
| 68H | Arbitration lost in SLA, R/$\overline{W}$ as MST. Own SLA and W have been received, ACK returned |
| 70H | General call has been received, ACK returned |
| 78H | Arbitration lost in SLA, R/$\overline{W}$ as MST. General call received, ACK returned |
| 80H | Previously addressed with own SLA. DATA byte received, ACK returned |
| 88H | Previously addressed with own SLA. DATA byte received, ACKNOT returned |
| 90H | Previously addressed with general call. DATA byte received, ACK has been returned |
| 98H | Previously addressed with general call. DATA byte received, ACKNOT has been returned |
| A0H | A STOP condition or repeated START condition received while still addressed as SLV/REC or SLV/TRX |

**Table 77** Miscellaneous

| S1STA VALUE | DESCRIPTION |
|---|---|
| 00H | Bus error during MST mode or selected SLV mode, due to an erroneous START or STOP condition |

### 12.5.2 I²C-BUS DATA SHIFT REGISTER (SDAT)

**Table 78** I²C-bus Data Shift Register (address FFFF 8201H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DATA.7 | DATA.6 | DATA.5 | DATA.4 | DATA.3 | DATA.2 | DATA.1 | DATA.0 |

**Table 79** Description of SDAT bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 0 | DATA.7 to DATA.0 | The serial data to be transmitted or data that has just been received. Bit 7 is transmitted or received first; i.e. data is shifted from right to left. |

### 12.5.3 I²C-BUS ADDRESS REGISTER (SADR)

This 8-bit register may be loaded with the 7-bit address to which the controller will respond when programmed as a slave receiver/transmitter.

**Table 80** I²C-bus Address Register (address FFFF 8203H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SADR.7 | SADR.6 | SADR.5 | SADR.4 | SADR.3 | SADR.2 | SADR.1 | SADR.0 |

**Table 81** Description of SADR bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 1 | SADR.7 to SADR.1 | Slave address. |
| 0 | SADR.0 | SADR.0 = GC, is used to determine whether the general CALL address is recognized. If GC = 0, general CALL address is not recognized (default value after a CPU reset). If GC = 1, general CALL address is recognized. |

## 13 PULSE WIDTH MODULATION (PWM) OUTPUTS

Two Pulse Width Modulation outputs are provided on the P90CL301. These channels output pulses of programmable length and interval. The repetition frequency is defined by an 8-bit prescaler PWMP, which generates the clock for the counter. The 8-bit counter counts modulo 255 (from 0 to 254 inclusive).

The prescaler and counter are used for the two channel outputs. The value of the 8-bit counter is compared to the content of the registers PWM0 (resp. PWM1) for the channel output PWM0 (resp. PWM1). Provided the content of this register is greater than the counter value, the output of PWM0 (resp. PWM1) is set LOW. If the content of this register is equal to, or less than the counter value, the output will stay high. The pulse width ratio is therefore defined by the content of the register PWM0 (respectively PWM1).

The pulse width ratio is in the range of 0 to $^{255}/_{255}$ and may be programmed in increments of $^{1}/_{255}$.

The repetition frequency:

$$f_{PWM} = \frac{FCLK}{(1 + PWMP) \times 255} Hz\,;\ \text{for FCLK in Hz.}$$

When using a peripheral clock of 6 MHz for example, the above formula gives a repetition frequency range of 23 kHz to 91 Hz.

By loading the PWM0 (resp. PWM1) with either 00H or FFH, the PWM0 output can be retained at a constant HIGH or LOW level respectively. When loading FFH to the PWM0 (respectively PWM1) register, the 8-bit counter will never actually reach this value.

### 13.1 Prescaler PWM Register (PWMP)

**Table 82** Prescaler PWM Register (address FFFF 8801H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PWMP.7 | PWMP.6 | PWMP.5 | PWMP.4 | PWMP.3 | PWMP.2 | PWMP.1 | PWMP.0 |

**Table 83** Description of PWMP bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 0 | PWMP.7 to PWMP.0 | Prescaler division factor = (PWMP + 1). |

### 13.2 PWM Data Registers (PWM0 and PWM1)

**Table 84** PWM Data Registers PWM0 and PWM1

| ADDRESS | REGISTER | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| FFFF 8803H | PWM0 | PWM0.7 | PWM0.6 | PWM0.5 | PWM0.4 | PWM0.3 | PWM0.2 | PWM0.1 | PWM0.0 |
| FFFF 8805H | PWM1 | PWM1.7 | PWM1.6 | PWM1.5 | PWM1.4 | PWM1.3 | PWM1.2 | PWM1.1 | PWM1.0 |

**Table 85** Description of PWM0 and PWM1 bits; n = 0 to 1

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7 to 0 | PWMn.7 to PWMn.0 | Pulse width ratio. LOW/HIGH ratio of PWMn signals $= \frac{(PWMn)}{255 - (PWMn)}$ |

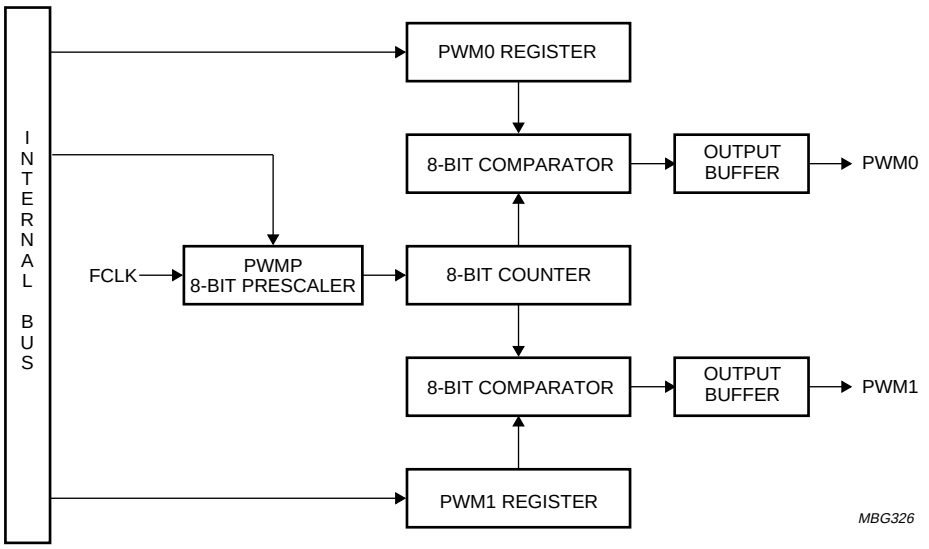



Fig.13 PWM block diagram.

## 14 ANALOG-TO-DIGITAL CONVERTER (ADC)

The analog input circuitry consist of a 4 input analog multiplexer and an analog-to-digital converter (ADC) with 8-bit resolution. The analog reference voltage $V_{ref(A)}$ and the analog supplies $V_{DDA}$, $V_{SSA}$ are connected via separate input pins.

The conversion time takes 24 periods of the secondary peripheral clock FCLK2 (see Section 6.6). The maximum value of the FCLK2 clock is dependant on the supply voltage (see Section 20).

As the ADC is based on a successive approximation algorithm using a resistor scale connected to $V_{ref(A)}$ and $V_{SSA}$, a continuous current flows in this resistor.

By resetting the EADC bit in the control register ADCON or by entering Power-down it is possible to switch off this current to reduce the static power consumption.

The ADC is controlled using the ADCON control register. Input channels are selected by the analog multiplexer function of register bits ADCON.0 and ADCON.1.
The completion of the 8-bit ADC conversion is flagged by ADCI in the ADCON register and the result is stored in the register ADCDAT (address FFFF 8809H). The result of a completed conversion remains unaffected provided ADCI is HIGH. While ADCS or ADCI are HIGH, a new ADC start will be blocked and consequently lost. An ADC conversion already in progress is aborted when Power-down mode is entered.

### 14.1 ADC Control Register (ADCON)

**Table 86** ADC Control Register (address FFFF 8807H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | EADC | – | ADCI | ADCS | – | A1 | A0 |

**Table 87** Description of ADCON bits

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7, 5 and 2 | – | Reserved; set to LOW. |
| 6 | EADC | **ADC enable**. If EADC = 1, then ADC is enabled. If EADC = 0, then ADC is disabled; the resistor reference is switched off to save power even while the CPU is operating. |
| 4 | ADCI | **ADC interrupt flag**. This flag is set when an ADC conversion result is ready to be read. An interrupt is invoked if the level IPLA is different from ‘0’. The flag must be cleared by software (it cannot be set by software). The ADCI bit must be cleared before a new conversion is started. |
| 3 | ADCS | **ADC start and status**. Setting this bit starts a conversion. The logic ensures that this signal is HIGH while the conversion is in progress. On completion, ADCS is reset at the same time the interrupt flag ADCI is set. ADCS cannot be reset by software. |
| 1, 0 | A1, A0 | **Analog input select**. This binary coded address selects one of the four analog inputs ADC0 to ADC3. It can only be changed when ADCI and ADCS are both LOW. A1 is the MSB; e.g. ‘11’ selects analog input channel ADC3. |

**Table 88** Operation of ADCI and ADCS

| ADCI | ADCS | OPERATION |
|---|---|---|
| 0 | 0 | ADC not busy, a conversion can be started. |
| 0 | 1 | ADC busy, start of a new conversion is blocked. |
| 1 | 0 | Conversion completed, start of a new conversion is blocked. |
| 1 | 1 | Intermediate status for a maximum of one machine cycle before conversion is completed (ADCI = 1, ADCS = 0). |

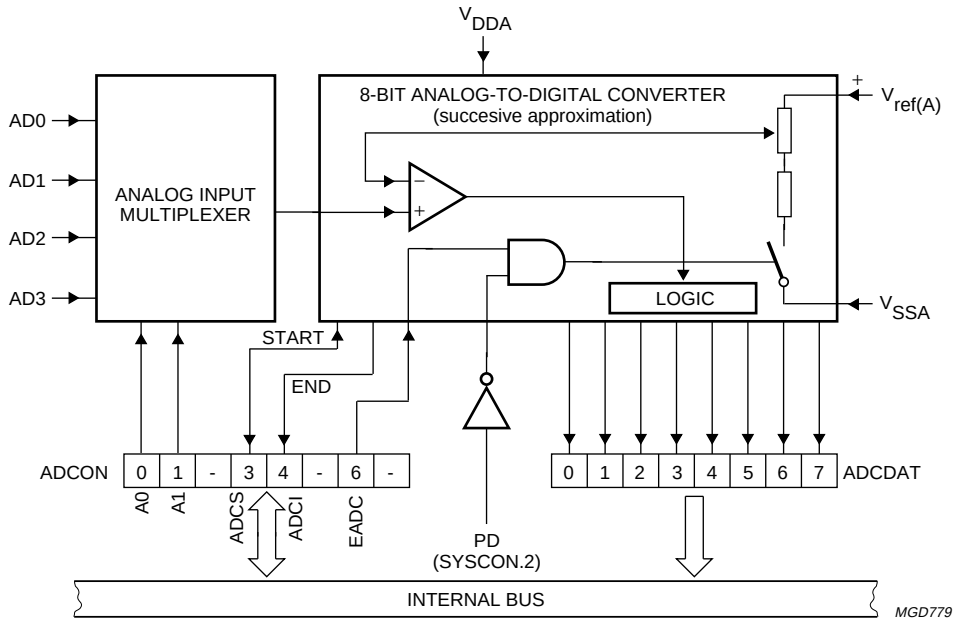



Fig.14 Functional diagram of the ADC.

## 15 ON-BOARD TEST CONCEPT

To improve the on-board debugging two functions are implemented, the ON-Circuit Emulation (ONCE) mode and the on-chip Test-ROM.

### 15.1 ONCE mode

The ON-Circuit Emulation (ONCE) mode eases the testing of an application without having to remove the controller from the board. The ONCE mode is entered by pulling $\overline{\text{CSBT}}$ LOW during reset. In this mode the address bus, data bus and bus control signals are in 3-state mode, all other output or bidirectional pins are weakly pulled HIGH. In this mode an emulator probe can be hooked-up to the circuit. Normal operation is restored with a normal reset.

### 15.2 Test-ROM

A second on-board debugging function is introduced for the situation where no extra connector can be placed on the PCB. It consists of an internal Test-ROM of 256 bytes which is used as boot ROM after a special test mode is activated during reset. The CPU will execute the code placed in the Test-ROM and initialize the UART0 and its baud rate generator and wait for commands to be sent to UART0.

The internal access time is in this case 3 cycles long. It can only be accessed in supervisor mode.

The purpose of the Test-ROM is to offer the user a simple software interface to load programs for testing its own application and to transmit back the test result.

The program can be loaded from the host into either the on-chip RAM or the external memory. The Test-ROM mode is entered by pulling LOW the R/$\overline{\text{W}}$ / TROM pin during reset.

Just after the RESET initialization, the user should send a character of 9 bits (one stop bit plus eight data bits) with all bits being zero, on the RX0 line.

Using the timer, the character length is captured and then the baud rate is automatically calculated and the baud rate generator is initialized. The UART0 is then initialized in Mode 3 with SM2 multiprocessor bit set, REN and TB8 bit set (SCON = F8H). The hardware is now ready to handle the protocol using the following 4 commands (Code 00 to 11).

**Table 89** Command format

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Code | | NB byte – 1 | | | | | |

**Table 90** Command description

| BIT | SYMBOL | DESCRIPTION |
|---|---|---|
| 7, 6 | Code | Pointer commands; see Table 91. |
| 5 to 0 | NB byte – 1 | Indicates the length of the transfer; e.g. (NB byte – 1) = 0 means a 1 byte transfer, (NB byte – 1) = 63 means a 64 byte transfer. |

**Table 91** Pointer commands

| CODE | | DESCRIPTION |
|---|---|---|
| BIT 7 | BIT 6 | |
| 0 | 0 | The pointer (A0 register) is initialized with a value depending of the number of transferred bytes. The most significant byte should be transferred first. Protocol:<br>To start a data transfer, the pointer should be initialized first. It is incremented by one at each byte transfer between the memory and the host. The following registers are reserved for the protocol and should not be used by the user: D0, D1, D2, D3, A0, A1 and A2. |
| 0 | 1 | Read command. Read 1 to 64 bytes (load to the host). The pointer is incremented at each transfer. |
| 1 | 0 | Write command. Write 1 to 64 bytes (load from the host). The pointer is incremented at each transfer. |
| 1 | 1 | Jump command. If the NB field is 0 then a jump to the pointer address (A0) is done to start code execution. If the NB field ≠ 0, the complete protocol initialization is restarted (same effect as reset and R/$\overline{W}$ / TROM = 0). |

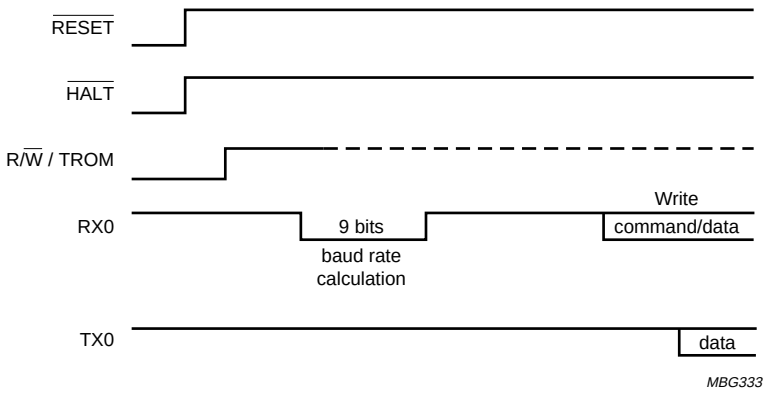



Fig.15 Test-ROM: Timing data transfer.

## 16 ON-CHIP RAM

The P90CL301BFH contains a 512 bytes RAM which can be used to store program code or data. As this memory does not need wait states, it can speed up some time consuming tasks like stack operation, table references, or small program loops, compared with slow external memory or when using the 8-bit data bus. For a read or write access, 3 CPU clocks are used. The memory content is kept even when the supply voltage is lowered down to 1.8 V after entering Power-down mode. The base address is FFFF 9000H. It can be accessed in long word, word or bytes.

## 17 REGISTER MAPPING

The internal register map of the P90CL301BFH is summarized in Table 92. Note that the internal registers can be accessed:

- only in Supervisor mode for version P90CL301BFH-3/4
- both in Supervisor and User mode for version P90CL301BFH-5.

**Table 92** Register map

| ADDRESS (HEX) | SYMBOL | WIDTH(1) | STATE AFTER RESET (HEX)(2) | REGISTER | ACCESS(3) |
|---|---|---|---|---|---|
| **System register** | | | | | |
| FFFF 8000 | SYSCON | W | 00C0 | System Control Register | R/W |
| **Interrupt registers** | | | | | |
| FFFF 8101 | LIR0 | B | 00 | Latched Interrupt 0/1 Register | R/W |
| FFFF 8103 | LIR1 | B | 00 | Latched Interrupt 2/3 Register | R/W |
| FFFF 8105 | LIR2 | B | 00 | Latched Interrupt 4/5 Register | R/W |
| FFFF 8107 | LIR3 | B | 00 | Latched Interrupt 6/7 Register | R/W |
| FFFF 810F | PIFR | B | 00 | Pending Interrupt Flag Register | R/C |
| **$I^2C$-bus registers** | | | | | |
| FFFF 8201 | SDAT | B | 00 | $I^2C$-bus Data Register | R/W |
| FFFF 8203 | SADR | B | 00 | $I^2C$-bus Address Register | R/W |
| FFFF 8205 | SSTA | B | F8 | $I^2C$-bus Status Register | R |
| FFFF 8207 | SCON | B | 00 | $I^2C$-bus Control Register | R/W |
| **Timers registers** | | | | | |
| FFFF 8300 | T0CRH | B/W | 0000 | Timer 0 Control Register (High byte) | R/W |
| FFFF 8301 | T0CRL | B | 00 | Timer 0 Control Register (Low byte) | R/W |
| FFFF 8302 | T0RR | W | 0000 | Timer 0 Reload Register | W |
| FFFF 8304 | T0 | W | 0000 | Timer 0 Register | R |
| FFFF 8306 | T0C0 | W | XXXX | Timer 0 Channel 0 Register | R/W |
| FFFF 8308 | T0C1 | W | XXXX | Timer 0 Channel 1 Register | R/W |
| FFFF 830A | T0C2 | W | XXXX | Timer 0 Channel 2 Register | R/W |
| FFFF 830D | T0SR | B | X0 | Timer 0 Status Register | R/C |
| FFFF 830F | T0PR | B | 00 | Timer 0 Prescaler Reload Register | W |
| FFFF 8310 | T1CRH | B/W | 0000 | Timer 1 Control Register (High byte) | R/W |
| FFFF 8311 | T1CRL | B | 00 | Timer 1 Control Register (Low byte) | R/W |
| FFFF 8312 | T1RR | W | 0000 | Timer 1 Reload Register | W |
| FFFF 8314 | T1 | W | 0000 | Timer 1 Register | R |
| FFFF 8316 | T1C0 | W | XXXX | Timer 1 Channel 0 Register | R/W |
| FFFF 8318 | T1C1 | W | XXXX | Timer 1 Channel 1 Register | R/W |
| FFFF 831A | T1C2 | W | XXXX | Timer 1 Channel 2 Register | R/W |
| FFFF 831D | T1SR | B | X0 | Timer 1 Status Register | R/C |

| ADDRESS (HEX) | SYMBOL | WIDTH[1] | STATE AFTER RESET (HEX)[2] | REGISTER | ACCESS[3] |
|---|---|---|---|---|---|
| FFFF 831F | T1PR | B | 00 | Timer 1 Prescaler Reload Register | W |
| FFFF 8401 | WDTIM | B | 00 | Watchdog Timer Register | R/W |
| FFFF 8403 | WDCON | B | A5 | Watchdog Control Register (only A5H or 5AH) | S |
| **Port registers** | | | | | |
| FFFF 8503 | PCON | B | 00 | Port Control Register | R/W |
| FFFF 8505 | PRL | B | FF | P Port Latch (least significant byte) | R/W |
| FFFF 8507 | PPL | B | FF | P Port Pin (least significant byte) | R |
| FFFF 8509 | PRH | B | FF | P Port Latch (most significant byte) | R/W |
| FFFF 850B | PPH | B | FF | P Port Pin (most significant byte) | R |
| FFFF 8109 | SPCON | B | 80 | SP Port Control Register | R/W |
| FFFF 810B | SPR | B | FF | SP Port Latch | R/W |
| FFFF 810D | SPP | B | FF | SP Port Pin | R |
| **UART registers** | | | | | |
| FFFF 8601 | SBUF0 | B | XX | UART0 Transmit/Receive Register | R/W |
| FFFF 8603 | SCON0 | B | 00 | UART0 Control Register | R/W |
| FFFF 8605 | SBUF1 | B | XX | UART1 Transmit/Receive Register | R/W |
| FFFF 8607 | SCON1 | B | 00 | UART1 Control Register | R/W |
| **Baud rate generator registers** | | | | | |
| FFFF 860B | BREGL | B | 00 | UART Baud Rate Register (least significant byte) | R/W |
| FFFF 860D | BREGH | B | 00 | UART Baud Rate Register (most significant byte) | R/W |
| FFFF 860F | BCON | B | 00 | UART Baud Rate Control Register | R/W |
| **Peripheral interrupt registers** | | | | | |
| FFFF 8701 | PICR0 | B | 00 | Timer Interrupt Register | R/W |
| FFFF 8703 | PICR1 | B | 00 | UART0 Interrupt Register | R/W |
| FFFF 8705 | PICR2 | B | 00 | UART1 Interrupt Register | R/W |
| FFFF 8707 | PICR3 | B | 00 | $I^2C$ and ADC Interrupt Register | R/W |
| **Pulse Width Modulation registers** | | | | | |
| FFFF 8801 | PWMP | B | 00 | PWM Prescaler Register | W |
| FFFF 8803 | PWM0 | B | 00 | PWM0 Data Register | R/W |
| FFFF 8805 | PWM1 | B | 00 | PWM1 Data Register | R/W |
| **ADC registers** | | | | | |
| FFFF 8807 | ADCON | B | 00 | ADC Control Register | R/W |
| FFFF 8809 | ADCDAT | B | FF | ADC Data Register | R |
| **Chip-select registers** | | | | | |
| FFFF 8A00 | CS0N | W | FFFF | Chip-select 0 Control Register | R/W |

| ADDRESS (HEX) | SYMBOL | WIDTH[1] | STATE AFTER RESET (HEX)[2] | REGISTER | ACCESS[3] |
|---|---|---|---|---|---|
| FFFF 8A02 | CS1N | W | FFFF | Chip-select 1 Control Register | R/W |
| FFFF 8A04 | CS2N | W | FFFF | Chip-select 2 Control Register | R/W |
| FFFF 8A06 | CS3N | W | FFFF | Chip-select 3 Control Register | R/W |
| FFFF 8A08 | CS4N | W | FFFF | Chip-select 4 Control Register | R/W |
| FFFF 8A0A | CS5N | W | FFFF | Chip-select 5 Control Register | R/W |
| FFFF 8A0C | CS6N | W | FFFF | Chip-select 6 Control Register | R/W |
| FFFF 8A0E | $\overline{\text{CSBT}}$ | W | F306 | Chip-select Boot Control Register | R/W |
| FFFF 8A11 | BSREG | B | 00 | Bus Size Register | R/W |
| **UART queue registers** | | | | | |
| FFFF 8B00 | UQRC | B | 00 | UART Queue Receive Control Register | R/W |
| FFFF 8B01 | UQTC | B | 00 | UART Queue Transmit Control Register | R/W |
| FFFF 8B03 | UQTA | B | 00 | UART Queue Transmit Address Register | R/W |
| FFFF 8B05 | UQTS | B | 00 | UART Queue Transmit Status Register | R/C |
| FFFF 8B07 | UQRA | B | 00 | UART Queue Receive Address Register | R/W |
| FFFF 8B09 | UQRS | B | 00 | UART Queue Receive Status Register | R/C |
| FFFF 8B0B | UQRM | B | 00 | UART Queue Receive Match Register | R/W |

**Notes**

1. Width when specified is in byte (B) or word (W).
2. X = don’t care.
3. Access when specified is in read (R) write (W) or clear (C) only. The Watchdog Control Register is special (S).

## 18 LIMITING VALUES

In accordance with the Absolute Maximum Rating System (IEC 134).

| SYMBOL | PARAMETER | MIN. | MAX. | UNIT |
|---|---|---|---|---|
| $V_{DD}$ | supply voltage | –0.5 | +3.6 | V |
| $V_I$ | input voltage on any pin with respect to ground ($V_{SS}$) | –0.5 | $V_{DD}$ + 0.5 | V |
| $I_I$, $I_O$ | DC current into any input or output | – | 5 | mA |
| $P_{tot}$ | total power dissipation | – | 300 | mW |
| $T_{stg}$ | storage temperature range | –65 | +150 | °C |
| $T_{amb}$ | operating ambient temperature range | –40 | +85 | °C |
| $T_j$ | operating junction temperature range | – | +125 | °C |

## 19 DC CHARACTERISTICS

$V_{DD}$ = 2.7 to 3.6 V; $V_{SS}$ = 0 V; $T_{amb}$ = –40 to +85 °C; all voltages with respect to $V_{SS}$ unless otherwise specified.

| SYMBOL | PARAMETER | CONDITIONS | MIN. | TYP. | MAX. | UNIT |
|---|---|---|---|---|---|---|
| **Supply** | | | | | | |
| $V_{DD}$ | supply voltage | | 2.7 | – | 3.6 | V |
| $I_{DD}$ | supply current operating; note 1 | $V_{DD}$ = 3 V; CLK = 13.8 MHz | – | 16 | 22 | mA |
| | | $V_{DD}$ = 3 V; CLK = 27 MHz | – | 32 | 40 | mA |
| $I_{DD(ID)}$ | supply current Idle mode; note 2a | $V_{DD}$ = 3 V; CLK = 13.8 MHz | – | 400 | 500 | µA |
| | | $V_{DD}$ = 3 V; CLK = 27 MHz | – | 800 | 1000 | µA |
| $I_{DD(STB)}$ | supply current Standby mode; note 2b | $V_{DD}$ = 3 V; CLK = 13.8 MHz | – | 9 | 15 | mA |
| | | $V_{DD}$ = 3 V; CLK = 27 MHz | – | 18 | 25 | mA |
| $I_{DD(PD)}$ | supply current Power-down mode; note 3 | $V_{DD}$ = 3 V | – | 2 | 40 | µA |
| **Inputs** | | | | | | |
| $V_{IL}$ | LOW level input voltage | | $V_{SS}$ | – | $0.3V_{DD}$ | V |
| $V_{IL}$ | LOW level input voltage; D15 to D8, XTAL1, $\overline{HALT}$, $\overline{RESET}$, RESETIN | | $V_{SS}$ | – | $0.1V_{DD}$ | V |
| $V_{IH}$ | HIGH level input voltage | | $0.7V_{DD}$ | – | $V_{DD}$ | V |
| $I_{IL}$ | LOW level input current | $V_{DD}$ = 3 V; $V_{IN}$ = 0.4 V | – | 13 | 50 | µA |
| $I_{TL}$ | input current HIGH-to-LOW transition | $V_{DD}$ = 3 V; $V_{IN}$ = $0.5V_{DD}$ | – | 140 | 500 | µA |
| $I_{TSI}$ | 3-state input current | | – | 1 | 10 | µA |
| **Outputs** | | | | | | |
| $I_{OH4}$ | HIGH level output current; TS4 and OD4; note 4 | $V_{DD}$ = 3 V; $V_{OH}$ = $V_{DD}$ – 0.4 V | 4 | 13 | – | mA |
| $I_{OH2}$ | HIGH level output current; WP2; note 4 | $V_{DD}$ = 3 V; $V_{OH}$ = $V_{DD}$ – 0.4 V | 2 | 7 | – | mA |
| $I_{OL8}$ | LOW level output current; OD8 and S8; note 4 | $V_{DD}$ = 3 V; $V_{OL}$ = 0.4 V | 8 | 24 | – | mA |
| $I_{OL4}$ | LOW level output current; TS4 and OD4; note 4 | $V_{DD}$ = 3 V; $V_{OL}$ = 0.4 V | 4 | 15 | – | mA |
| $I_{OL2}$ | LOW level output current; WP2; note 4 | $V_{DD}$ = 3 V; $V_{OL}$ = 0.4 V | 2 | 8 | – | mA |
| $C_{IN}$ | input capacitance; note 5 | | – | – | 10 | pF |

| SYMBOL | PARAMETER | CONDITIONS | MIN. | TYP. | MAX. | UNIT |
|---|---|---|---|---|---|---|
| $R_{UP}$ | pull-up resistor UP; note 6a | | 16 | 26 | 60 | kΩ |
| $R_{UP2}$ | pull-up resistor UP2; note 6b | | 8 | 15 | 30 | kΩ |
| $R_{UP3}$ | pull-up resistor UP3; note 6c | | 70 | 100 | 500 | kΩ |
| $R_{STIN}$ | RESETIN resistor | | 15 | 31 | 120 | kΩ |

**Notes**

1. The operating supply current through $V_{DD1}$, $V_{DD2}$ and $V_{DD3}$ is measured with all output pins disconnected; RESETIN = $\overline{RESET}$ = $\overline{HALT}$ = 0; A23 to A0 = $V_{DD}$; D15 to D0 = $V_{DD}$.
2. Idle and Standby current:
   a) The Idle supply current through $V_{DD1}$, $V_{DD2}$ and $V_{DD3}$ is measured with all port pins disconnected; A23 to A0 = $V_{DD}$; D15 to D0 = $V_{DD}$; the circuit is executing NOP instructions from an external memory.
   b) The Standby current through VDD1, VDD2 and VDD3 is measured with all port pins disconnected; A23 to A0 = VDD; D15 to D0 = VDD;
3. The Power-down current through $V_{DD1}$, $V_{DD2}$ and $V_{DD3}$ is measured with all output pins disconnected; XTAL1 = $\overline{RESET}$ = HALTN = $V_{DD}$; A23 to A0 = $V_{DD}$; D15 to D0 = $V_{DD}$; RESETIN = $V_{SS}$.
4. See Table 95 for the different types.
5. Not tested in production.
6. Pull-ups:
   a) These pull-ups are only present on the emulation pins $\overline{PHALT}$ and NMINE.
   b) These active pull-ups are active on all WP2 WP4 port pins for output voltages greater than Vdd/2. They are only active during the reset sequence on the pins $\overline{CS0}$, $\overline{CS1}$, R/$\overline{W}$, $\overline{CSBT}$ and $\overline{FETCH}$ for test purpose.
   c) These active pull-ups are only active on D15 to D0 and A23 to A0 pins when BPE is set in the SYSCON register.

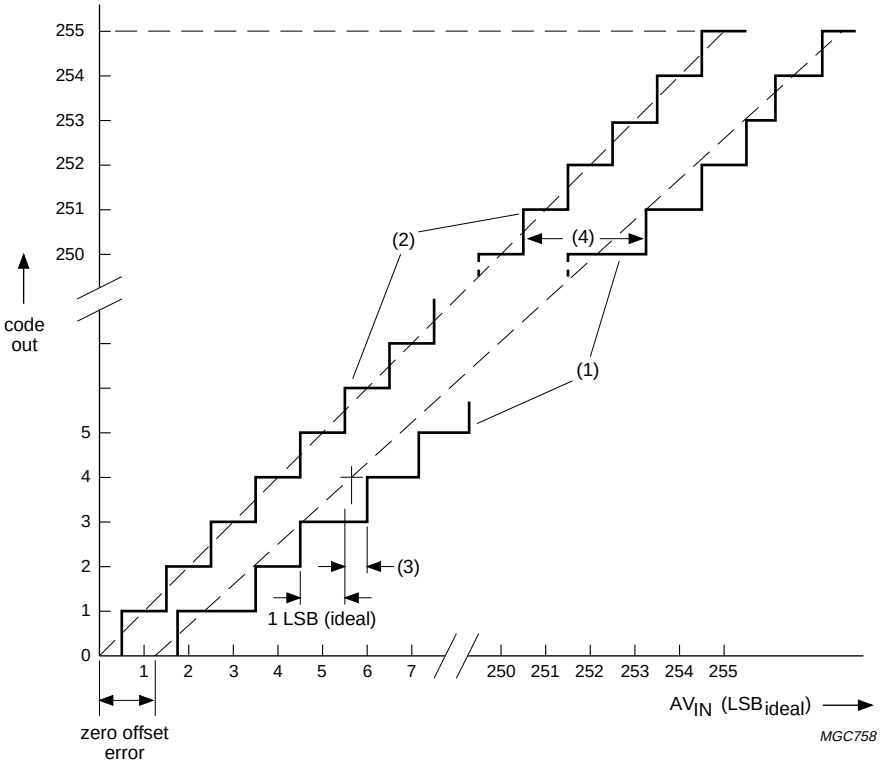
## 20 ADC CHARACTERISTICS

$V_{DD}$ = 2.7 to 3.6 V; $V_{ref(A)} = V_{DDA} = V_{DD}$; $V_{SSA} = V_{SS}$; $V_{SS}$ = 0 V; FCLK2 = 250 kHz to 2 MHz; $T_{amb}$ = –40 to +85 °C; for ADC test conditions see note 1; all voltages with respect to $V_{SS}$ unless otherwise specified.

| SYMBOL | PARAMETER | CONDITIONS | MIN. | TYP. | MAX. | UNIT |
|---|---|---|---|---|---|---|
| $V_{DDA}$ | analog supply voltage | | $V_{DD}$ – 0.2 | – | $V_{DD}$ + 0.2 | V |
| $V_{ref(A)}$ | analog reference voltage | | $V_{DD}$ – 0.2 | – | $V_{DD}$ + 0.2 | V |
| $V_{SSA}$ | analog ground | | $V_{SS}$ – 0.2 | – | $V_{SS}$ + 0.2 | V |
| $V_{IN(A)}$ | analog input voltage | | 0 | – | $V_{ref(A)}$ | V |
| $I_{DDA}$ | supply current operating | $V_{DDA}$ = 3.0 V | – | 150 | 250 | µA |
| $I_{DD(PD)(A)}$ | analog supply current Power-down mode | $V_{DDA}$ = 3.0 V | – | 0.1 | 5 | µA |
| $R_{Vref}$ | resistor between $V_{ref(A)}$ and $V_{SSA}$ | note 2 | 20 | 34 | 150 | kΩ |
| $C_{IA}$ | analog input capacitance | note 3 | – | – | 12 | pF |
| $I_{IA}$ | input leakage current | $V_{DDA}$ = 3.0 V | – | – | 1 | µA |
| FCLK2 | ADC clock frequency; | $V_{DDA}$ = 2.7 V; note 4 | 0.25 | – | 2 | MHz |
| $t_{ADS}$ | sampling time | | – | $6 \times t_{FCLK2}$ | – | µs |
| $t_{ADC}$ | total conversion time | | – | $24 \times t_{FCLK2}$ | – | µs |
| $A_e$ | absolute voltage error | note 1 and 5 | – | – | 1 | LSB |
| $OS_e$ | offset error | note 1 and 6 | – | – | 1 | LSB |
| $IL_e$ | integral non-linearity | note 1 and 7 | – | – | 1 | LSB |
| $DL_e$ | differential non-linearity | note 1 and 8 | – | – | 1 | LSB |
| $M_{ctc}$ | channel-to-channel matching | note 3 and 9 | – | – | 1 | LSB |

**Notes**

1. ADC test conditions: $V_{DD}$ = 2.7 V, $V_{ref(A)}$ = 2.7 V, CLK = 20 MHz, FCLK2 = 2 MHz.
2. This resistor is switched off during Power-down mode and when the ADC is switched off (EADC = 0).
3. Parameter not measured in production, only verified on sampling basis.
4. See Fig.17 for specific FCLK2 range as function of $V_{DD}$.
5. Absolute voltage error: the maximum difference between actual and ideal code transitions. Absolute voltage error accounts for all deviations of an actual converter from an ideal converter.
6. Offset error: the difference between the actual and ideal input voltage corresponding to the first actual code transition.
7. Integral non-linearity: the maximum deviation between the edges of the steps of the transfer curve and the edges of the steps of the ideal curve. The ideal step curve follows the line of least squares.
8. Differential non-linearity: the maximum deviation of the actual code width from the average code width.
9. Channel-to-channel matching: The difference between corresponding code transitions of actual characteristics taken from different channels under the same temperature, voltage and frequency conditions.




(1) Example of an actual transfer curve.
(2) The ideal transfer curve.
(3) Differential non-linearity.
(4) Absolute voltage error.

$$1\ \text{LSB} = \frac{V_{ref(A)} - V_{SSA}}{256}$$

Fig.16 ADC conversion characteristics.

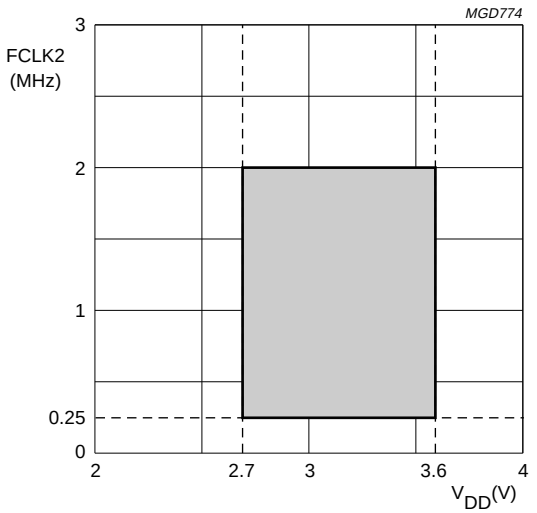



Fig.17 ADC clock (FCLK2) frequency range as a function of $V_{DD}$.

## 21 AC CHARACTERISTICS

$V_{SS}$ = 0 V; $T_{amb}$ = –40 to +85 °C; $t_{CLK}$ = CPU clock cycle time; no fast bus cycle (FBC = 0); no wait status; all voltages with respect to $V_{SS}$ unless otherwise specified.

| SYMBOL | PARAMETER | MIN. | TYP. | MAX. | UNIT |
|---|---|---|---|---|---|
| $t_{AVSL}$ | address valid to $\overline{AS}$ LOW | $0.5t_{CLK} - 10$ | $0.5t_{CLK} + 2$ | – | ns |
| $t_{SL}$ | $\overline{AS}$/$\overline{DS}$ LOW level | $2.5t_{CLK} - 10$ | $2.5t_{CLK} + 2$ | – | ns |
| $t_{SHAZ}$ | $\overline{AS}$ HIGH to address invalid | $0.5t_{CLK} - 10$ | $0.5t_{CLK}$ | – | ns |
| $t_{ASCSL}$ | $\overline{AS}$/$\overline{DS}$ to $\overline{CS}$ LOW | –5 | 1 | 5 | ns |
| $t_{ASCSH}$ | $\overline{AS}$/$\overline{DS}$ to $\overline{CS}$ HIGH | –5 | 1 | 5 | ns |
| $t_{SLSH}$ | $\overline{AS}$ LOW to $\overline{DS}$ LOW (write) | $t_{CLK} - 15$ | $t_{CLK}$ | $t_{CLK} + 15$ | ns |
| $t_{DSL}$ | $\overline{DS}$ LOW level (write) | $1.5t_{CLK} - 10$ | $1.5t_{CLK} + 2$ | – | ns |
| $t_{AVRL}$ | address valid to R/$\overline{W}$ LOW (write) | $t_{CLK}$ - 5 | $t_{CLK}$ | – | ns |
| $t_{CLSL}$ | R/$\overline{W}$ LOW to $\overline{DS}$ LOW (write) | $t_{CLK} - 10$ | $t_{CLK} - 2$ | – | ns |
| $t_{DOSL}$ | DATA-OUT valid to $\overline{DS}$ LOW (write) | $0.5t_{CLK} - 10$ | $0.5t_{CLK} - 1$ | – | ns |
| $t_{SHDO}$ | $\overline{AS}$ HIGH to DATA-OUT invalid | $0.5t_{CLK} - 10$ | $0.5t_{CLK} - 3$ | – | ns |
| $t_{HRPW}$ | $\overline{HALT}$/$\overline{RESET}$ pulse width | $24t_{CLK}$ | – | – | ns |
| $t_{ASLDTA}$ | $\overline{AS}$ LOW to $\overline{DTACK}$ LOW | – | $1.5t_{CLK} - 28$ | $1.5t_{clk} - 10$ | ns |
| $t_{ASHDTA}$ | $\overline{AS}$ HIGH to $\overline{DTACK}$ HIGH | – | $2.5t_{CLK} - 25$ | $2.5t_{clk}$ | ns |
| $t_{DCLDI}$ | $\overline{DTACK}$ LOW to DATA-IN (set-up time) | – | $t_{CLK}$ | $t_{CLK} + 10$ | ns |
| $t_{DATSETUP}$ | $\overline{AS}$ LOW to DATA-IN (set-up time) | - | $2.5t_{CLK} - 25$ | $2.5t_{CLK} - 20$ | ns |
| $t_{SHDI}$ | $\overline{AS}$ HIGH to DATA invalid (hold time) | 0 | 0 | – | ns |
| $t_{SHRH}$ | $\overline{AS}$ HIGH to R/$\overline{W}$ HIGH (write) | $0.5t_{CLK} - 5$ | $0.5t_{CLK} - 2$ | – | ns |
| $t_{SHAH}$ | $\overline{AS}$ HIGH to A0 HIGH | $t_{CLK} - 10$ | $t_{CLK} + 3$ | $t_{CLK} + 10$ | ns |
| $t_{SHAWH}$ | $\overline{AS}$ HIGH to A0 (first byte of word cycle in 8-bit mode) | $0.5t_{CLK} - 10$ | – | $0.5t_{CLK} + 10$ | ns |
| $t_{SHW}$ | LDS HIGH level before write | $2.5t_{CLK} - 5$ | $2.5t_{CLK} - 2$ | – | ns |

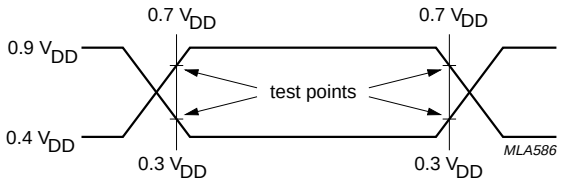



Fig.18 AC testing input waveform.

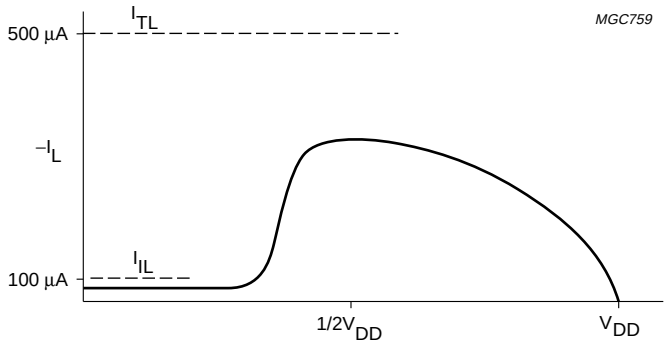



Fig.19 Input current.

## 22 8051 BUS TIMING

$V_{DD}$ = 2.7 V to 3.6 V; $V_{SS}$ = 0 V; $T_{amb}$ = −40 to +85 °C; $t_{CLK}$ = CPU clock cycle time; all voltages with respect to $V_{SS}$ unless otherwise specified. These AC parameters are not tested in production.

| SYMBOL | PARAMETER | MIN. | MAX. | UNIT |
|---|---|---|---|---|
| $t_{RR}$ | read pulse duration | $4.5t_{CLK} - 10$ | $4.5t_{CLK} + 10$ | ns |
| $t_{WW}$ | write pulse duration | $4.5t_{CLK} - 10$ | $4.5t_{CLK} + 10$ | ns |
| $t_{AL}$ | address set-up time | $1.5t_{CLK} - 20$ | – | ns |
| $t_{LA}$ | address hold time | $t_{CLK} - 5$ | – | ns |
| $t_{RD}$ | RD to valid data input | – | $3.5t_{CLK} - 15$ | ns |
| $t_{DFR}$ | data float after read | – | $2t_{CLK} - 10$ | ns |
| $t_{LD}$ | ALE to valid data input | – | $6t_{CLK} - 20$ | ns |
| $t_{LW}$ | ALE to $\overline{RD}$ $\overline{WR}$ | $3t_{CLK} - 20$ | $3t_{CLK} + 20$ | ns |
| $t_{DW}$ | data set-up time before $\overline{WR}$ | $6.5t_{CLK} - 20$ | – | ns |
| $t_{WD}$ | data hold time after $\overline{WR}$ | $0.5t_{CLK} - 10$ | – | ns |
| $t_{WHLH}$ | $\overline{RD}$ $\overline{WR}$ HIGH to ALE HIGH | $t_{CLK} - 10$ | $t_{CLK} + 10$ | ns |

## 23 TIMING DIAGRAMS

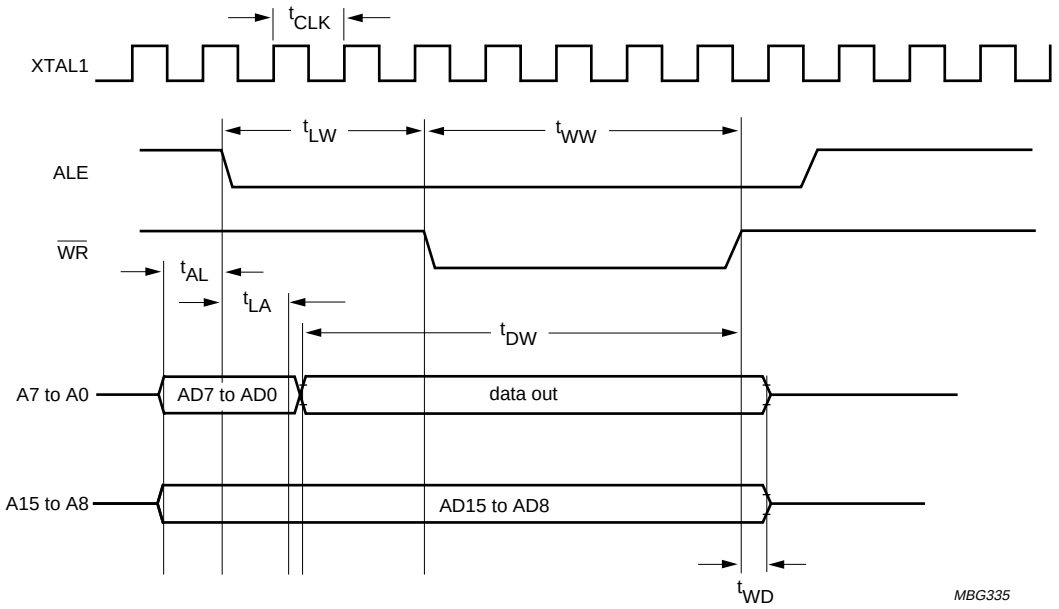



Fig.20 Write to 8051-compatible peripheral circuits.

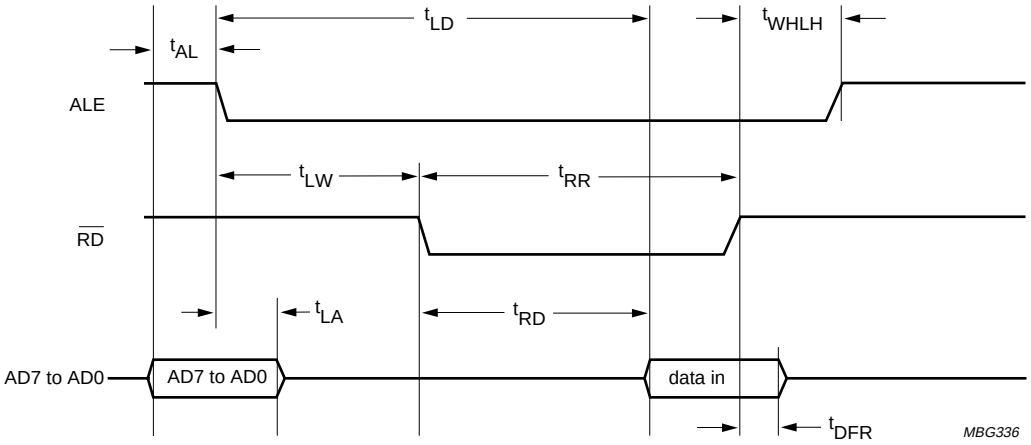



Fig.21 Read from 8051-compatible peripheral circuits.

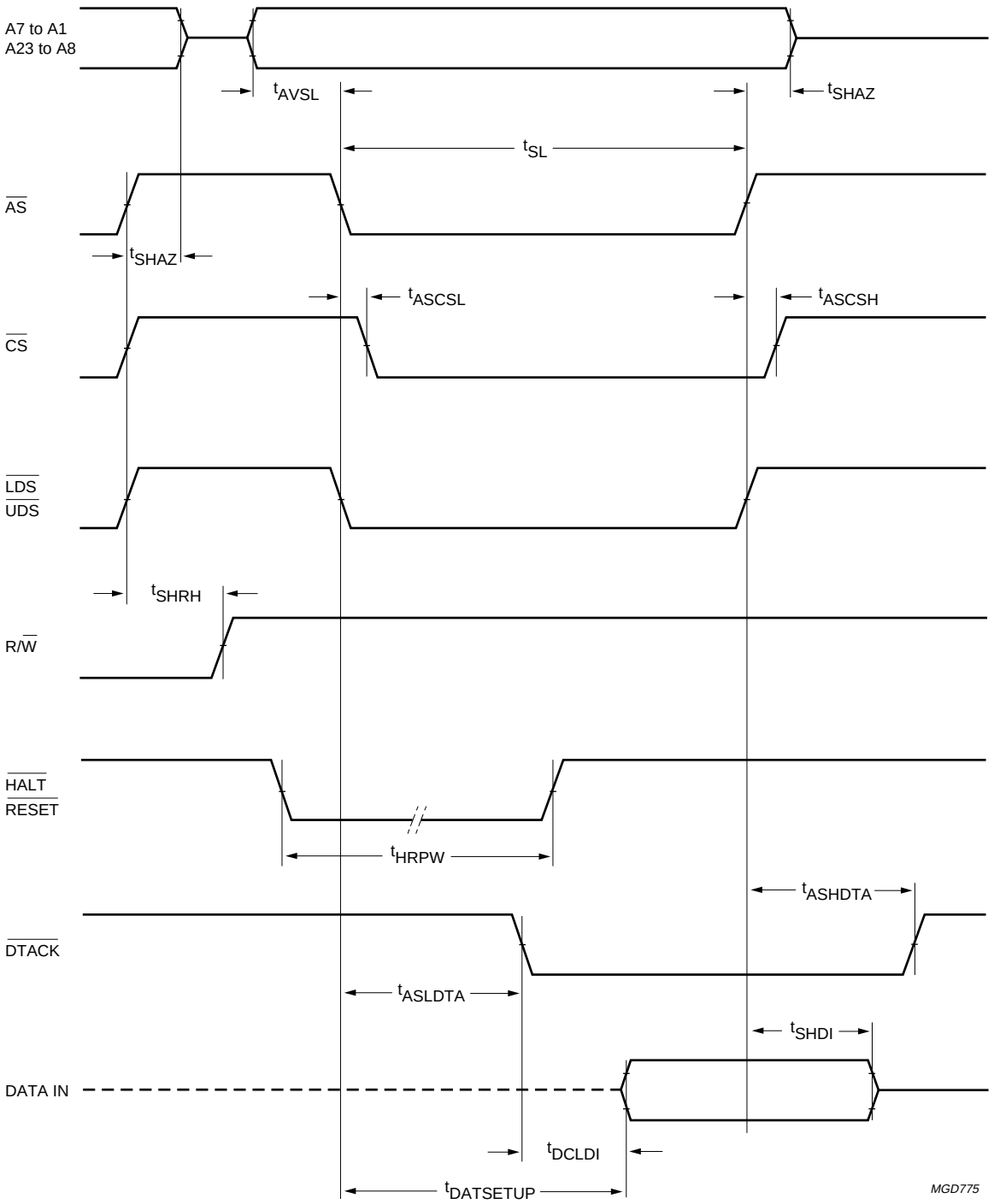



Fig.22 Read cycle timing 16-bit mode.

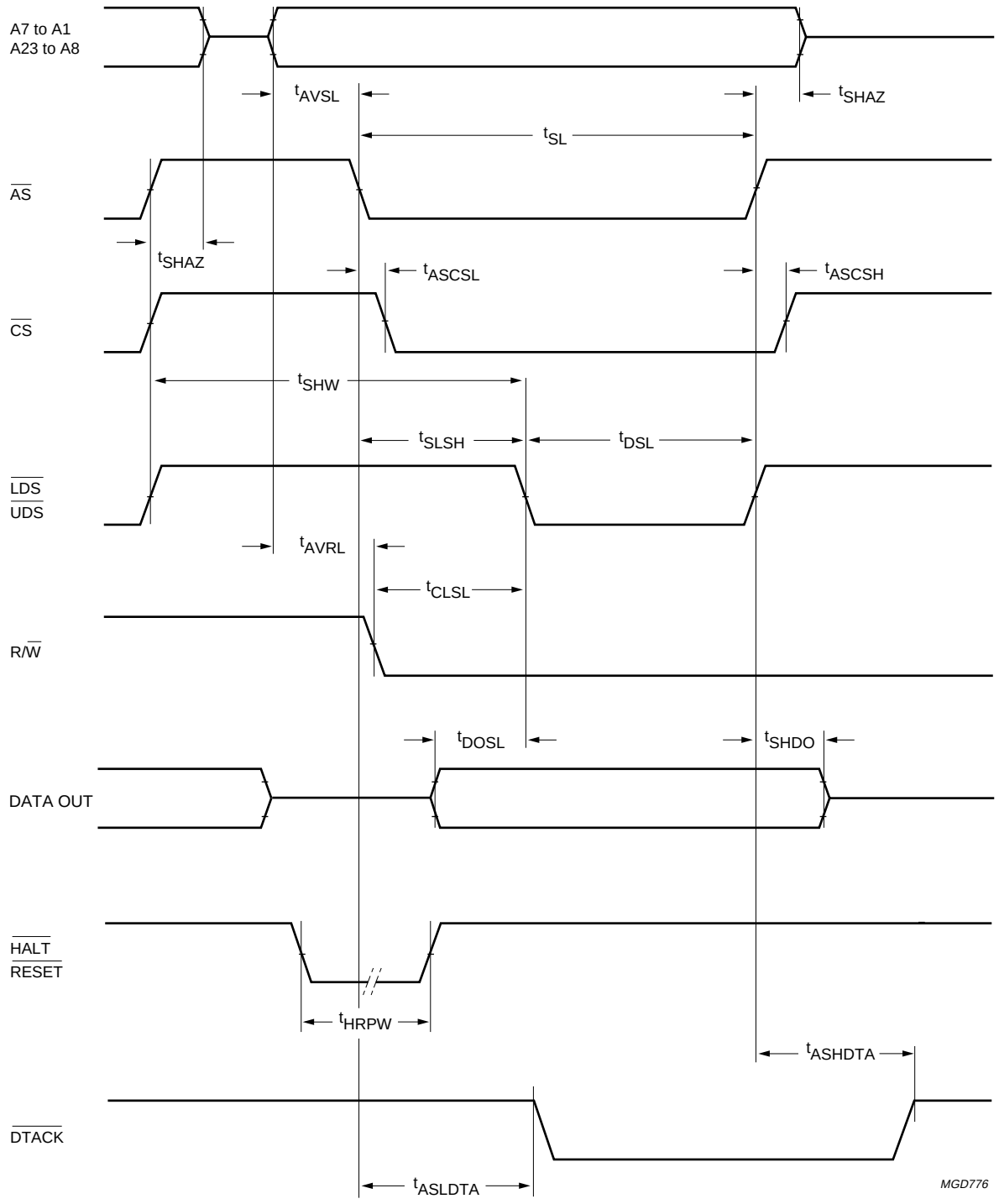
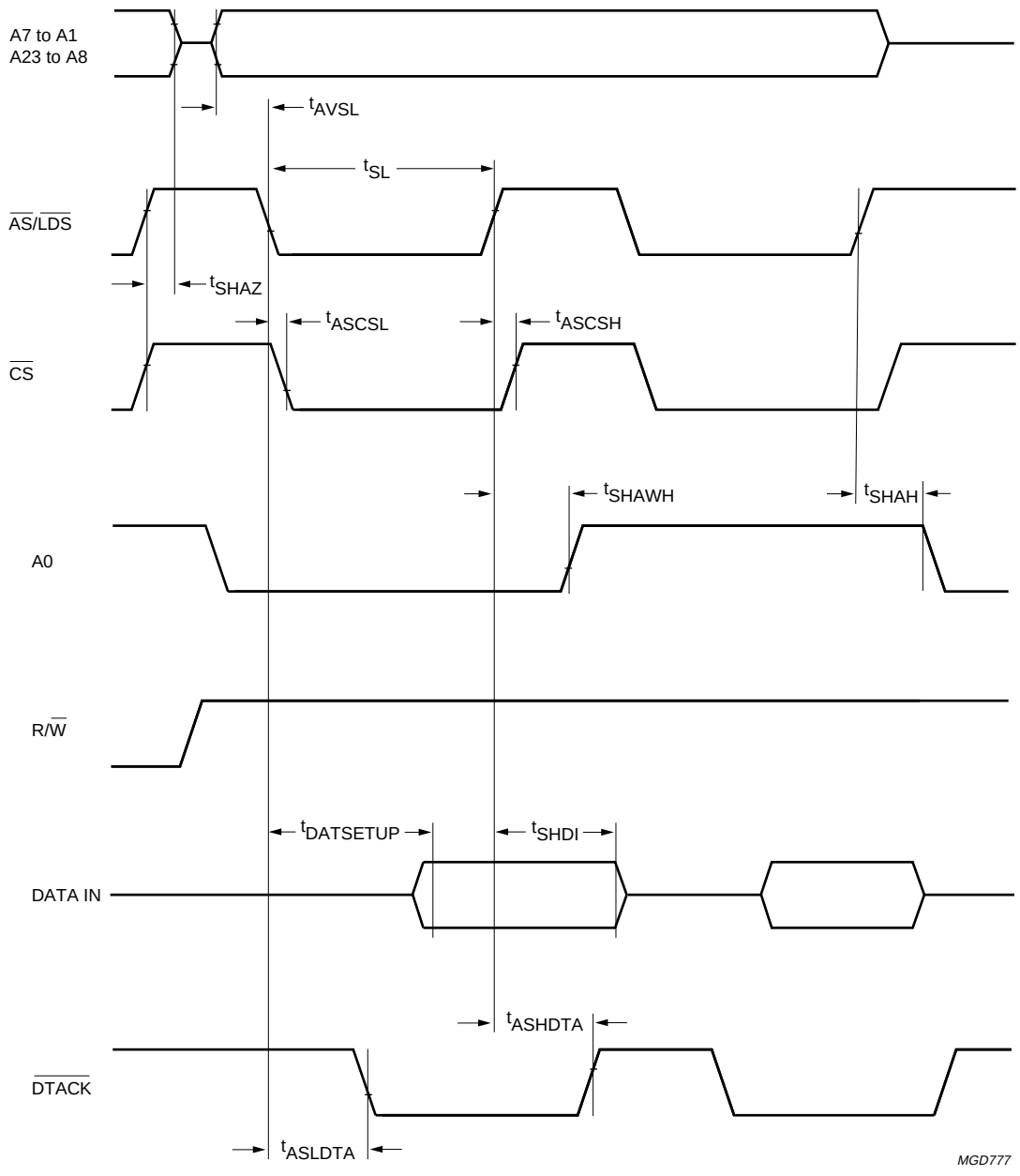



Fig.23 Write cycle timing 16-bit mode.




(no fast bus cycle, FBC = 0)

Fig.24 Read cycle timing 8-bit mode.

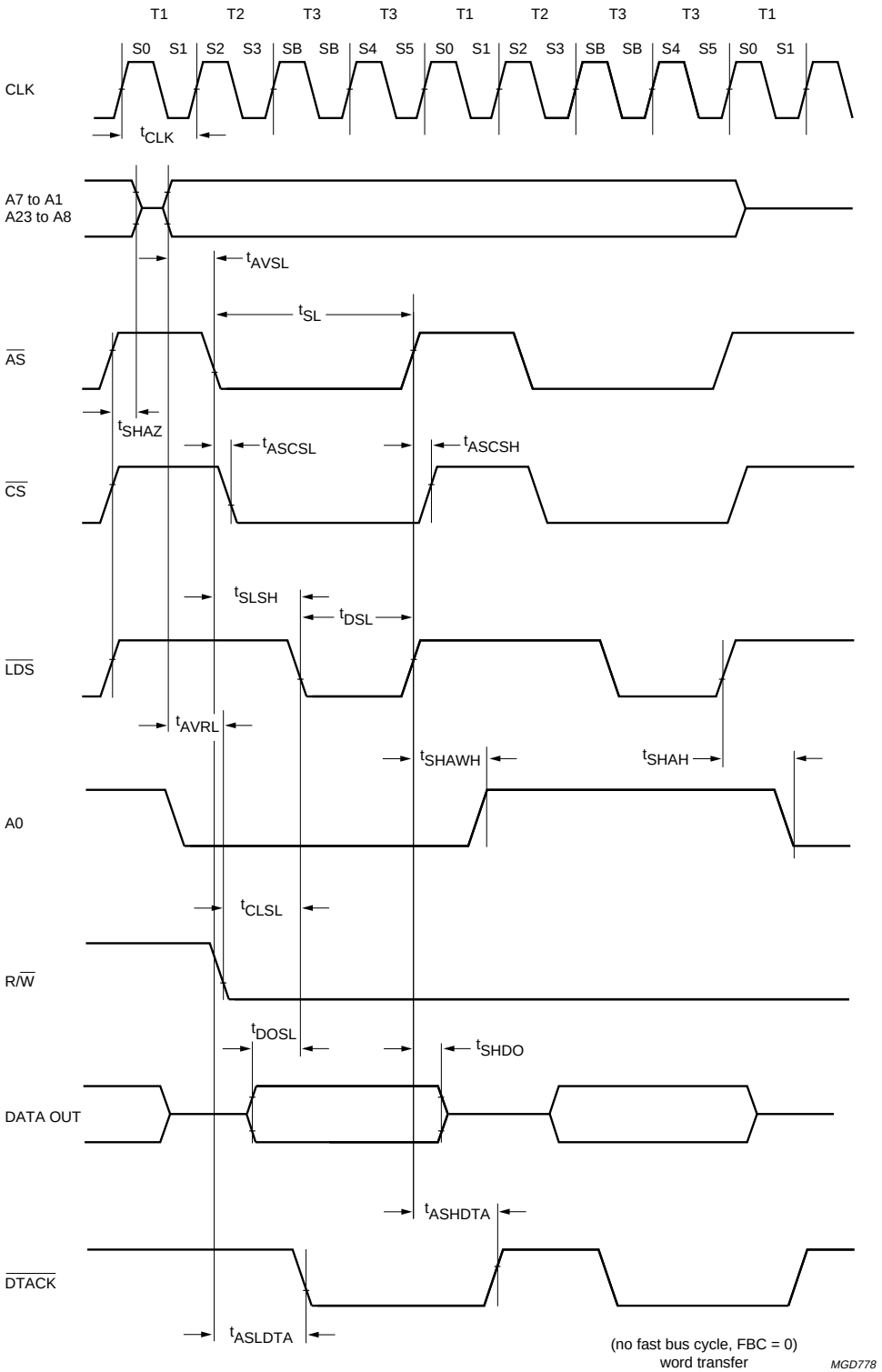



Fig.25 Write cycle timing 8-bit mode clock timing.

## 24 CLOCK TIMING

**Table 93** P90CL301BFH clock timing

$V_{DD}$ = 2.7 V.

| SYMBOL | PARAMETER | MIN. | MAX. | UNIT |
|---|---|---|---|---|
| $f_{XTAL1}$ | input frequency | 0 | 27 | MHz |
| $t_{CLK}$ | cycle time | 37 | – | ns |
| $t_{CL}$ | pulse width LOW | 13 | – | ns |
| $t_{CH}$ | pulse width HIGH | 13 | – | ns |
| $t_{CR}$ | rise time | – | 5 | ns |
| $t_{CF}$ | fall time | – | 5 | ns |
| $\frac{t_{CH}}{t_{CLK}}$ | duty cycle | 45 | 55 | % |

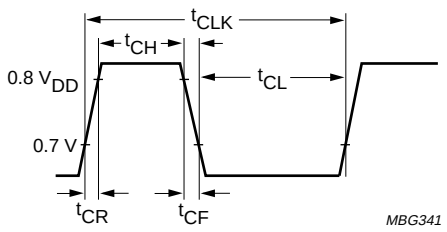



Fig.26 P90CL301BFH clock timing.

## 25 PIN STATES IN VARIOUS MODES

Table 94 describes the function, I/O, type and state in various modes - $\overline{\text{RESET}}$, Power-down, $\overline{\text{HALT}}$, $\overline{\text{ONCE}}$ and BPE (Bus Pull-up Enable) - of the pins.

**Table 94** Pin states in various modes

| PIN | FUNCTION | I/O(1) | TYPE(2) | STATE(3) | | | | BPE ON |
|---|---|---|---|---|---|---|---|---|
| | | | | $\overline{\text{RESET}}$ | PD | $\overline{\text{HALT}}$ | $\overline{\text{ONCE}}$ | |
| A22 to A19 | address bus | O | TSW4 | Z | Z | Z | Z | W |
| $\overline{\text{PCS0}}$ to $\overline{\text{PCS3}}$ | 8051 chip-select | O | TS4 | – | H | Z | – | W |
| A18 to A1 | address bus | O | TSW4 | Z | Z | Z | Z | W |
| AD7 to AD1 | 8051 data bus | I/O | TSW4 | – | Z | Z | – | W |
| D7 to D0 | lower 8-bits of data bus | I/O | TSW4 | Z | Z | Z | Z | W |
| D15 to D8 | upper 8-bits of data bus | I/O | TSW4 | Z | W | Z | Z | W |
| PL7 to PL0 | port PL | I/O | WP4 | – | S | W | W | W |
| $\overline{\text{AS}}$ | address strobe | O | TS4 | H | H | Z | Z | – |
| $\overline{\text{LDS}}$ | low data strobe | O | TS4 | H | H | Z | Z | – |
| $\overline{\text{UDS}}$ | upper data strobe | O | TS4 | H | H | Z | Z | W |
| A0 | address 0 | O | TSW4 | H | H | Z | Z | W |
| AD0 | 8051 address/data 0 | I/O | TSW4 | – | – | Z | Z | W |
| R/$\overline{\text{W}}$ | read write strobe | O | TS4 | Z | H | Z | Z | – |
| TROM | Test-ROM mode | I | UP2 | – | – | – | – | – |
| $\overline{\text{DTACK}}$ | data transfer acknowledgement | I | N | – | – | – | – | – |
| $\overline{\text{RESET}}$ | CPU peripheral reset | I | N | – | – | – | – | – |
| | peripheral reset output | OD | OD8 | L | Z | Z | Z | – |
| RESETIN | external power-on-reset | I | RS | – | – | – | – | – |
| $\overline{\text{HALT}}$ | reset input; HALT input | I | N | – | – | – | – | – |
| | peripheral reset; fault output | OD | OD8 | L | Z | Z | Z | – |
| BSIZE | data bus size | I | N | – | – | – | – | – |
| NMIACK | emulation NMIN acknowledgement | OD | OD8 | Z | Z | Z | Z | – |
| SP0 | second port pin 0 | I/O | WP2 | W | S | W | W | – |
| RX1 | UART1 receive | I/O | WP2 | – | S | W | W | – |
| $\overline{\text{INT0}}$ | interrupt input 0 | I | N | – | – | – | – | – |
| SP1 | second port pin 1 | I/O | WP2 | W | S | W | W | – |
| TX1 | UART1 transmit | O | WP2 | – | S | W | W | – |
| $\overline{\text{INT1}}$ | interrupt input 1 | I | N | – | – | – | – | – |
| CLK0 | external clock Timer 0 | I | N | – | – | – | – | – |
| SP2 | second port pin 2 | I/O | WP2 | W | S | W | W | – |
| RX0 | UART0 receive | I/O | N | – | – | – | – | – |
| $\overline{\text{INT2}}$ | interrupt input 2 | I | N | – | – | – | – | – |
| CP2 | timer capture 2 | I | N | – | – | – | – | – |
| SP3 | second port pin 3 | I/O | WP2 | W | S | W | W | – |
| TX0 | UART0 transmit | I/O | WP2 | – | S | W | W | – |

| PIN | FUNCTION | I/O[1] | TYPE[2] | STATE[3] | | | | BPE ON |
|---|---|---|---|---|---|---|---|---|
| | | | | $\overline{\text{RESET}}$ | PD | $\overline{\text{HALT}}$ | $\overline{\text{ONCE}}$ | |
| $\overline{\text{INT3}}$ | interrupt input 3 | I | N | – | – | – | – | – |
| CP3 | timer capture 3 | I | N | – | – | – | – | – |
| SP4 | second port pin 4 | I/O | WP2 | W | S | W | W | – |
| $\overline{\text{INT4}}$ | interrupt input 4 | I | N | – | – | – | – | – |
| CP4 | timer capture 4 | I | N | – | – | – | – | – |
| SP5 | second port pin 5 | I/O | WP2 | W | S | W | W | – |
| $\overline{\text{INT5}}$ | interrupt input 5 | I | N | – | – | – | – | – |
| CP5 | timer capture 5 | I | N | – | – | – | – | – |
| SP6 | second port pin 6 | I/O | WP2 | W | S | W | W | – |
| $\overline{\text{INT6}}$ | interrupt input 6 | I | N | – | – | – | – | – |
| CLK1 | external clock Timer 1 | I | N | – | – | – | – | – |
| NMIN | non-maskable interrupt | I | N | – | – | – | – | – |
| SP7 | second port pin 7 | I/O | WP2 | W | S | W | W | – |
| P8 | port PH pin 8 | I/O | WP2 | W | S | W | W | – |
| PWM0 | PWM output 0 | O | WP2 | – | H | W | W | – |
| CP0 | timer capture 0 | I | N | – | – | – | – | – |
| P9 | port PH pin 9 | I/O | WP2 | W | S | W | W | – |
| PWM1 | PWM output 1 | O | WP2 | – | H | W | W | – |
| CP1 | timer capture 1 | I | N | – | – | – | – | – |
| XTAL1 | external crystal input | I | XI | – | – | – | – | – |
| $\overline{\text{CS1}}$ to $\overline{\text{CS0}}$ | chip-select 1 to 0 | O | TS4 | W | H | Z | Z | – |
| FC1 to FC0 | function code | O | TS4 | – | S | Z | Z | – |
| TSM1 to TSM0 | test mode inputs multiplexed with CS1N/0N for test purpose only. | I | UP2 | – | – | – | – | – |
| $\overline{\text{CS2}}$ | chip-select 2 | O | TS4 | H | H | Z | Z | – |
| $\overline{\text{CS3}}$ | chip-select 3 | O | TS4 | H | H | Z | Z | – |
| ALE | 8051 address strobe | O | TS4 | – | H | Z | Z | – |
| $\overline{\text{CS4}}$ | chip-select 4 | O | TS4 | H | H | Z | Z | – |
| $\overline{\text{RD}}$ | 8051 read strobe | O | TS4 | – | H | Z | Z | – |
| $\overline{\text{CS5}}$ | chip-select 5 | O | TS4 | H | H | Z | Z | – |
| $\overline{\text{WR}}$ | 8051 write strobe | O | TS4 | – | H | Z | Z | – |
| P10 | port PH pin 10 | I/O | OD8 | Z | Z | Z | Z | – |
| SCL | $I^2C$-bus clock | OD | OD8 | – | Z | Z | Z | – |
| P11 | port PH pin 11 | I/O | OD8 | Z | Z | Z | Z | – |
| SDA | $I^2C$-bus data | OD | OD8 | – | Z | Z | Z | – |
| $\overline{\text{CS6}}$ | chip-select 6 | O | TS4 | – | H | Z | Z | – |
| A23 | address pin 23 | O | TS4 | H | S | Z | Z | – |
| $\overline{\text{CSBT}}$ | chip-select boot | O | TS4 | W | H | Z | Z | – |
| $\overline{\text{ONCE}}$ | ONCE mode | I | UP2 | – | – | – | – | – |
| P15 to P12 | port PH pins 15 to 12 | I/O | WP2 | W | W | Z | Z | – |

| PIN | FUNCTION | I/O[1] | TYPE[2] | STATE[3] | | | | BPE ON |
|---|---|---|---|---|---|---|---|---|
| | | | | $\overline{\text{RESET}}$ | PD | $\overline{\text{HALT}}$ | $\overline{\text{ONCE}}$ | |
| ADC3 to ADC0 | analog inputs 3 to 0 | I | AN | – | – | – | – | – |
| $V_{ref(A)}$ | ADC reference voltage | I | AREF | Z | Z | R | R | – |
| $\overline{\text{FETCH}}$[4] | fetch output | O | TS4 | W | Z | Z | Z | – |
| $\overline{\text{EMUL}}$[4] | emulation mode | I | UP2 | – | – | – | – | – |
| NMINE[4] | emulation NMIN | I | UP | – | – | – | – | – |
| CLKOUT[4] | emulation clock output | O | S4 | S | S | S | S | – |
| $\overline{\text{PHALT}}$[4] | emulation HALT | I | UP | – | – | – | – | – |

**Notes to the pin states in various modes**

1. I = input; O = output; I/O = bidirectional.
2. See Table 95 for pin type description.
3. State of the pin in different modes $\overline{\text{RESET}}$, PD (Power-down), $\overline{\text{HALT}}$, $\overline{\text{ONCE}}$ and BPE (Bus Pull-up Enable).
   a) – = not available.
   b) Z = 3-state.
   c) W = weak pull-up.
   d) S = state logic 0 or logic 1.
   e) R = resistive
   f) H = HIGH state.
   g) L = LOW state.
4. Emulation version only.

**Table 95** Pin type description

| PIN TYPE | DESCRIPTION | MAXIMUM LOAD (pF) |
|---|---|---|
| TS4 | 3-state output, normal input | 100 |
| TSW4 | 3-state output, normal input with internal pull-up | 100 |
| WP2 | weak pull-up output, normal input | 80 |
| WP4 | weak pull-up output, normal input | 80 |
| N | normal input | – |
| UP | input with internal pull-up | – |
| UP2 | input with internal pull-up | – |
| OD8 | open drain | 400 |
| AN | analog input | – |
| S4 | strong output | 100 |
| RS | Schmitt trigger input | – |
| AREF | analog reference input | – |

## 26 INSTRUCTION SET AND ADDRESSING MODES

The P90CL301BFH is completely code compatible with the 68000, which means that programs developed for the 68000 will run on the P90CL301BFH. This applies to both the source and object codes. The instruction set was designed to minimize the number of mnemonics that the programmer has to remember. Following tables give an overview of the instruction set and the different addressing modes.

**Table 96** Instruction set; for Condition codes see notes 1 to 7

| MNEMONIC | DESCRIPTION | OPERATION | CONDITION CODES | | | | |
|---|---|---|---|---|---|---|---|
| | | | X | N | Z | V | C |
| ABCD | Add Decimal with Extend | $(Destination)_{10} + (Source)_{10} + X \rightarrow Destination$ | * | U | * | U | * |
| ADD | Add Binary | (Destination) + (Source) → Destination | * | * | * | * | * |
| ADDA | Add Address | (Destination) + (Source) → Destination | – | – | – | – | – |
| ADDI | Add Immediate | (Destination) + Immediate Data → Destination | * | * | * | * | * |
| ADDQ | Add Quick | (Destination) + Immediate Data → Destination | * | * | * | * | * |
| ADDX | Add Extended | (Destination) + (Source) + X → Destination | * | * | * | * | * |
| AND | AND Logical | (Destination) ∧ (Source) → Destination | – | * | * | 0 | 0 |
| ANDI | AND Immediate | (Destination) ∧ Immediate Data → Destination | – | * | * | 0 | 0 |
| ASL, ASR | Arithmetic Shift | (Destination) Shifted by <count > → Destination | * | * | * | * | * |
| $B_{CC}$ | Branch Conditionally | If CC then PC + d → PC | – | – | – | – | – |
| BCHG | Test a Bit and Change | ~(< bit number >) of Destination → Z<br>~(< bit number >) of Destination → < bit number > of Destination | – | – | * | – | – |
| BCLR | Test a Bit and Clear | ~(< bit number >) of Destination → Z | – | – | * | – | – |
| BRA | Branch Always | PC + d → PC | – | – | – | – | – |
| BSET | Test a Bit and Set | ~(< bit number >) of Destination → Z<br>1 → < bit number > of Destination | – | – | * | – | – |
| BSR | Branch to Subroutine | PC → SP @ –; PC + d → PC | – | – | – | – | – |
| BTST | Test a Bit | ~(< bit number >) of Destination → Z | – | – | * | – | – |
| CHK | Check Register against Bounds | If Dn < 0 or Dn > (< source >) then TRAP | – | * | U | U | U |
| CLR | Clear an Operand | 0 → Destination | – | 0 | 1 | 0 | 0 |
| CMP | Compare | (Destination) – (Source) | – | * | * | * | * |
| CMPA | Compare Address | (Destination) – (Source) | – | * | * | * | * |
| CMPI | Compare Immediate | (Destination) – Immediate Data | – | * | * | * | * |
| CMPM | Compare Memory | (Destination) – (Source) | – | * | * | * | * |
| $DB_{cc}$ | Test Condition, Decrement & Branch | If (not CC) then Dn – 1 → Dn;<br>if Dn ≠ –1 then PC + d → PC | – | – | – | – | – |
| DIVS | Signed Divide | (Destination) / (Source) → Destination | – | * | * | * | 0 |
| DIVU | Unsigned Divide | (Destination) / (Source) → Destination | – | * | * | * | 0 |
| EOR | Exclusive OR Logical | (Destination) ⊕ (Source) → Destination | – | * | * | 0 | 0 |
| EORI | Exclusive OR Immediate | (Destination) ⊕ Immediate Data → Destination | – | * | * | 0 | 0 |
| EXG | Exchange Register | Rx ↔ Ry | – | – | – | – | – |

| MNEMONIC | DESCRIPTION | OPERATION | CONDITION CODES | | | | |
|---|---|---|---|---|---|---|---|
| | | | X | N | Z | V | C |
| EXT | Sign Extend | (Destination) Sign – extended → Destination | – | * | * | 0 | 0 |
| JMP | Jump | Destination → PC | – | – | – | – | – |
| JSR | Jump to Subroutine | PC → SP @ –; Destination → PC | – | – | – | – | – |
| LEA | Load Effective Address | Destination → An | – | – | – | – | – |
| LINK | Link and Allocate | An → SP @ –; SP → An; SP + d → SP | – | – | – | – | – |
| LSL, LSR | Logical Shift | (Destination) Shifted by < count > → Destination | * | * | * | 0 | * |
| MOVE | Move Data from Source to Destination | (Source) → Destination | – | * | * | 0 | 0 |
| MOVE to CCR | Move to Condition Code | (Source) → CCR | * | * | * | * | * |
| MOVE to SR | Move to the Status Register | (Source) → SR | * | * | * | * | * |
| MOVE from SR | Move from the Status Register | SR → Destination | – | – | – | – | – |
| MOVE USP | Move User Stack Pointer | USP → An; An → USP | – | – | – | – | – |
| MOVEA | Move Address | (Source) → Destination | – | – | – | – | – |
| MOVEM | Move Multiple Registers | Registers → Destination; (Source) → Registers | – | – | – | – | – |
| MOVEP | Move Peripheral Data | (Source) → Destination | – | – | – | – | – |
| MOVEQ | Move Quick | Immediate Data → Destination | – | * | * | 0 | 0 |
| MULS | Signed Multiply | (Destination) * (Source) → Destination | – | * | * | * | 0 |
| MULU | Unsigned Multiply | (Destination) * (Source) → Destination | – | * | * | * | 0 |
| NBCD | Negate Decimal with Extend | $0 – (Destination)_{10} – X \rightarrow Destination$ | * | U | * | U | * |
| NEG | Negate | 0 – (Destination) → Destination | * | * | * | * | * |
| NEGX | Negate with Extend | 0 – (Destination) – X → Destination | * | * | * | * | * |
| NOP | No Operation | – | – | – | – | – | – |
| NOT | Logical Complement | ~(Destination) → Destination | – | * | * | 0 | 0 |
| OR | Inclusive OR Logical | (Destination) ∨ (Source) → Destination | – | * | * | 0 | 0 |
| ORI | Inclusive OR Immediate | (Destination) ∨ Immediate Data → Destination | – | * | * | 0 | 0 |
| PEA | Push Effective Address | Destination → SP @ – | – | – | – | – | – |
| RESET | Reset External Devices | – | – | – | – | – | – |
| ROL, ROR | Rotate (Without Extend) | (Destination) Rotated by < count > → Destination | – | * | * | 0 | * |
| ROXL, ROXR | Rotate with Extend | (Destination) Rotated by < count > → Destination | * | * | * | 0 | * |
| RTE | Return from Exception | SP @ + → SR; SP @ + → PC | * | * | * | * | * |
| RTR | Return and Restore Condition Codes | SP @ + → CC; SP @ + → PC | * | * | * | * | * |
| RTS | Return from Subroutine | SP @ + → PC | – | – | – | – | – |
| SBCD | Subtract Decimal with Extend | $(Destination)_{10} – (Source)_{10} – X \rightarrow Destination$ | * | U | * | U | * |
| $S_{CC}$ | Set According to Condition | if CC then 1 → Destination; else 0 → Destination | – | – | – | – | – |

| MNEMONIC | DESCRIPTION | OPERATION | CONDITION CODES | | | | |
|---|---|---|---|---|---|---|---|
| | | | X | N | Z | V | C |
| STOP | Load Status Register and Stop | Immediate Data → SR; STOP | * | * | * | * | * |
| SUB | Subtract Binary | (Destination) – (Source) → Destination | * | * | * | * | * |
| SUBA | Subtract Address | (Destination) – (Source) → Destination | – | – | – | – | – |
| SUBI | Subtract Immediate | (Destination) – Immediate Data → Destination | * | * | * | * | * |
| SUBQ | Subtract Quick | (Destination) – Immediate Data → Destination | * | * | * | * | * |
| SUBX | Subtract with Extend | (Destination) – (Source) – X → Destination | * | * | * | * | * |
| SWAP | Swap Register Halves | Register [ 31:16 ] ↔ Register [ 15:0 ] | – | * | * | 0 | 0 |
| TAS | Test and Set an Operand | (Destination) Tested → CC; 1 → [ 7 ] of Destination | – | * | * | 0 | 0 |
| TRAP | Trap | PC → SSP @ –; SR → SSP @ –; (Vector) → PC | – | – | – | – | – |
| TRAPV | Trap on Overflow | If V then TRAP | – | – | – | – | – |
| TST | Test and Operand | (Destination) Tested → CC | – | * | * | 0 | 0 |
| UNLK | Unlink | An → SP; SP @ + → An | – | – | – | – | – |

**Notes**

1. [ ] = bit number.
2. * = affected.
3. – = unaffected.
4. 0 = cleared.
5. 1 = set.
6. U = defined.
7. @ = location addressed by.

### 26.1 Addressing modes

**Table 97** Data addressing modes; see notes 1 to 14

| MODE | GENERATION |
|---|---|
| **Register Direct Addressing** | |
| Data Register Direct | EA = Dn |
| Address Register Direct | EA = An |
| **Absolute Data Addressing** | |
| Absolute Short | EA = (Next Words) |
| Absolute Long | EA = (Next Two Words) |
| **Program Counter Relative Addressing** | |
| Relative with Offset | EA = (PC) + $d_{16}$ |
| Relative with Index and Offset | EA = (PC) + (Xn) + $d_8$ |
| **Register Indirect Addressing** | |
| Register Indirect | EA = (An) |
| Postincrement Register Indirect | EA = (An), An ← An + N |
| Predecrement Register Indirect | An ← An – N, EA = (An) |
| Register Indirect with Offset | EA = (An) + $d_{16}$ |
| Indexed Register Indirect with Offset | EA = (An) + (Xn) + $d_8$ |
| **Immediate Data Addressing** | |
| Immediate | DATA = Next Word(s) |
| Quick Immediate | Inherent Data |
| **Implied Addressing** | |
| Implied Register | EA = SR, USP, SSP, PC, SP |

**Notes**

1. EA = Effective Address.
2. An = Address Register.
3. Dn = Data Register.
4. Xn = Address or Data Register used as Index Register.
5. N = 1 for bytes; 2 for words; 4 for long words.
6. ← = Replaces.
7. SR = Status Register.
8. PC = Program Counter.
9. () = Contents of.
10. $d_8$ = 8-bit offset (displacement).
11. $d_{16}$ = 16-bit offset (displacement).
12. SP = Stack Pointer.
13. SSP = System Stack Pointer.
14. USP = User Stack Pointer.

## 27 INSTRUCTION TIMING

In the Tables 98 to 110 the number of bus read and write cycles are shown in parentheses as (R/W). The timing is given for operation in 16-bit mode. For operation in 8-bit mode the numbers shown in parentheses should be multiplied by a factor 2.

**Table 98** Effective address calculation times

| INSTRUCTION | ADDRESSING MODE | BYTE; WORD | LONG |
|---|---|---|---|
| Rn | Data or Address Register Direct | 0 (0/0) | 0 (0/0) |
| (An) | Address Register Indirect | 4 (1/0) | 8 (2/0) |
| (An)+ | Address Register Indirect postincrement | 4 (1/0) | 8 (2/0) |
| –(An) | Address Register Indirect predecrement | 7 (1/0) | 11 (2/0) |
| d(An) | Address Register Indirect Displacement | 11 (2/0) | 12 (3/0) |
| d(An, Xi) | Address Register Indirect with Index | 14 (2/0) | 8 (3/0) |
| xxx.S | Absolute Short | 8 (2/0) | 12 (3/0) |
| xxx.L | Absolute Long | 12 (3/0) | 16 (4/0) |
| d(PC) | Program Counter with Displacement | 11 (2/0) | 15 (3/0) |
| d(PC, Xi) | Program Counter with Index | 14 (2/0) | 16 (4/0) |
| #xxx | Immediate | 4 (1/0) | 8 (2/0) |

**Table 99** MOVE Byte and MOVE Word Instruction clock periods

| INSTR. | Rn | (An) | (An)+ | –(An) | d(An) | d(An, Xi) | xxx.S | xxx.L |
|---|---|---|---|---|---|---|---|---|
| Rn | 7 (1/0) | 11 (1/1) | 11 (1/1) | 14 (1/1) | 18 (1/1) | 21 (1/1) | 15 (1/1) | 19 (1/1) |
| (An) | 11 (2/0) | 15 (2/1) | 15 (2/1) | 18 (2/1) | 22 (2/1) | 25 (2/1) | 19 (2/1) | 23 (2/1) |
| (An)+ | 11 (2/0) | 15 (2/1) | 15 (2/1) | 18 (2/1) | 22 (2/1) | 25 (2/1) | 19 (2/1) | 23 (2/1) |
| –(An) | 14 (2/0) | 18 (2/1) | 18 (2/1) | 22 (2/1) | 25 (2/1) | 28 (2/1) | 22 (2/1) | 26 (2/1) |
| d(An) | 18 (3/0) | 22 (3/1) | 22 (3/1) | 25 (2/1) | 29 (2/1) | 32 (2/1) | 26 (2/1) | 30 (2/1) |
| d(An, Xi) | 21 (3/0) | 25 (3/1) | 25 (3/1) | 28 (3/1) | 32 (3/1) | 35 (3/1) | 29 (3/1) | 33 (3/1) |
| xxx.S | 15 (3/0) | 19 (3/1) | 19 (3/1) | 22 (3/1) | 26 (3/1) | 29 (3/1) | 23 (3/1) | 27 (3/1) |
| xxx.L | 19 (4/0) | 23 (4/1) | 23 (4/1) | 26 (4/1) | 30 (4/1) | 33 (4/1) | 27 (4/1) | 31 (4/1) |
| d(PC) | 18 (3/0) | 22 (3/1) | 22 (3/1) | 25 (3/1) | 29 (3/1) | 32 (3/1) | 26 (3/1) | 30 (3/1) |
| d(PC, Xi) | 21 (3/0) | 25 (3/1) | 25 (3/1) | 28 (3/1) | 32 (3/1) | 35 (3/1) | 29 (3/1) | 33 (3/1) |
| #xxx | 11 (3/0) | 15 (2/1) | 15 (2/1) | 18 (2/1) | 22 (2/1) | 25 (2/1) | 19 (2/1) | 23 (2/1) |

**Table 100** MOVE long instruction clock periods

| INSTR. | Rn | (An) | (An)+ | –(An) | d(An) | d(An, Xi) | xxx.S | xxx.L |
|---|---|---|---|---|---|---|---|---|
| Rn | 7 (1/0) | 15 (1/2) | 15 (1/2) | 18 (1/2) | 22 (2/2) | 25 (2/2) | 19 (2/2) | 23 (3/2) |
| (An) | 15 (2/0) | 23 (2/2) | 23 (2/2) | 26 (2/2) | 30 (4/2) | 33 (4/2) | 27 (4/2) | 31 (5/2) |
| (An)+ | 15 (3/0) | 23 (3/2) | 23 (3/2) | 26 (3/2) | 30 (4/2) | 33 (4/2) | 27 (4/2) | 31 (5/2) |
| –(An) | 18 (3/0) | 26 (3/2) | 26 (3/2) | 29 (3/2) | 33 (4/2) | 36 (4/2) | 30 (4/2) | 34 (5/2) |
| d(An) | 22 (4/0) | 30 (4/2) | 30 (4/2) | 33 (4/2) | 37 (5/2) | 40 (5/2) | 34 (5/2) | 38 (6/2) |
| d(An, Xi) | 25 (4/0) | 33 (4/2) | 33 (4/2) | 36 (4/2) | 40 (5/2) | 43 (5/2) | 37 (5/2) | 41 (6/2) |
| xxx.S | 19 (4/0) | 27 (4/2) | 27 (4/2) | 30 (4/2) | 34 (5/2) | 37 (5/2) | 31 (5/2) | 35 (6/2) |
| xxx.L | 23 (5/0) | 31 (5/2) | 31 (5/2) | 34 (5/2) | 38 (6/2) | 41 (6/2) | 35 (6/2) | 39 (7/2) |
| d(PC) | 22 (4/0) | 30 (4/2) | 30 (4/2) | 33 (4/2) | 37 (5/2) | 40 (5/2) | 34 (5/2) | 38 (6/2)) |
| d(PC, Xi) | 25 (4/0) | 33 (4/2) | 33 (4/2) | 36 (4/2) | 40 (5/2) | 43 (5/2) | 37 (5/2) | 41 (6/2) |
| #xxx | 15 (3/0) | 23 (3/2) | 23 (3/2) | 26 (3/2) | 30 (4/2) | 33 (4/2) | 27 (4/2) | 31 (5/2) |

**Table 101** Standard Instruction clock periods

| INSTRUCTION | SIZE | op<ea>, An | op<ea>, Dn | op<ea>, M |
|---|---|---|---|---|
| ADD | Byte, Word | 7(1) (1/0) | 7(1) (1/0) | 11(1) (1/1) |
| | Long | 7(1) (1/0) | 7(1) (1/0) | 15(1) (1/2) |
| AND | Byte, Word | – | 7(1) (1/0) | 11(1) (1/1) |
| | Long | – | 7(1) (1/0) | 15(1) (1/2) |
| CMP | Byte, Word | 7(1) (1/0) | 7(1) (1/0) | – |
| | Long | 7(1) (1/0) | 7(1) (1/0) | – |
| DIVS | – | – | 169(1)(2) (1/0) | – |
| DIVU | – | – | 130(1)(3) (1/0) | – |
| EOR | Byte, Word | – | 7(1) (1/0) | 11(1) (1/1) |
| | Long | – | 7(1) (1/0) | 15(1) (1/2) |
| MULS | – | – | 76(1)(3) (1/0) | – |
| MULU | – | – | 76(1)(3) (1/0) | – |
| OR | Byte, Word | – | 7(1) (1/0) | 11(1) (1/1) |
| | Long | – | 7(1) (1/0) | 15(1) (1/2) |
| SUB | Byte, Word | 7(1) (1/0) | 7(1) (1/0) | 11(1) (1/1) |
| | Long | 7(1) (1/0) | 7(1) (1/0) | 15(1) (1/2) |

**Notes**

1. Add effective address calculation time.
2. Indicates maximum value.
3. The duration of the instruction is constant.

**Table 102** Immediate instruction clock periods

| INSTRUCTION | SIZE | op<#>, Dn | op<#>, An | op<#>, M |
|---|---|---|---|---|
| ADDI | Byte, Word | 14 (2/0) | – | 18[1] (2/1) |
| | Long | 18 (3/0) | – | 26[1] (3/2) |
| ADDQ | Byte, Word | 7[1] (1/0) | 7[1] (1/0) | 11[1] (1/1) |
| | Long | 7[1] (1/0) | 7[1] (1/0) | 15[1] (1/2) |
| ANDI | Byte, Word | 14 (2/0) | – | 18[1] (2/1) |
| | Long | 18 (3/0) | – | 26[1] (3/2) |
| CMPI | Byte, Word | 14 (2/0) | – | 14 (2/0) |
| | Long | 18 (3/0) | – | 18 (3/0) |
| EORI | Byte, Word | 14 (2/0) | – | 18[1] (2/1) |
| | Long | – | – | 26[1] (3/2) |
| MOVEQ | Long | 7 (1/0) | – | – |
| ORI | Byte, Word | 14 (2/0) | – | 18[1] (2/1) |
| | Long | 18 (3/0) | – | 26[1] (3/2) |
| SUBI | Byte, Word | 14 (2/0) | – | 18[1] (2/1) |
| | Long | 18 (3/0) | – | 26[1] (3/2) |
| SUBQ | Byte, Word | 7[1] (1/0) | 7 (1/0) | 11[1] (1/1) |
| | Long | 7[1] (1/0) | 7 (1/0) | 15[1] (1/2) |

**Note**

1. Add effective address calculation time.

**Table 103** Shift/rotate instruction clock periods

| INSTRUCTION | SIZE | REGISTER | MEMORY |
|---|---|---|---|
| ASR, ASL | Byte | 13 + 3n (1/0) | 14 (1/1)[1] |
| | Word | 13 + 3n (1/0) | – |
| LSR, LSL | Byte, Word | 13 + 3n (1/0) | 14 (1/1)[1] |
| | Long | 13 + 3n (1/0) | – |
| ROR, ROL | Byte, Word | 13 + 3n (1/0) | 14 (1/1)[1] |
| | Long | 13 + 3n (1/0) | – |
| ROXR, ROXL | Byte, Word | 13 + 3n (1/0) | 14 (1/1)[1] |
| | Long | 13 + 3n (1/0) | – |

**Note**

1. Add effective address calculation time.

**Table 104** Single operand instruction clock periods

| INSTRUCTION | SIZE | REGISTER | MEMORY |
|---|---|---|---|
| CLR | Byte, Word | 7 (1/0) | 11 (1/1)[1][2] |
| | Long | 7 (1/0) | 15 (1/2)[1][3] |
| NBCD | Byte, Word | 10 (1/0) | 14 (1/1)[1] |
| NEG | Byte, Word | 7 (1/0) | 11 (1/1)[1] |
| | Long | 7 (1/0) | 15 (1/2)[1] |
| NEGX | Byte, Word | 7 (1/0) | 11 (1/1)[1] |
| | Long | 7 (1/0) | 15 (1/2)[1] |
| NOT | Byte, Word | 7 (1/0) | 11 (1/1)[1] |
| | Long | 7 (1/0) | 15 (1/2)[1] |
| Scc | Byte, Word | 13 (1/0) | 17 (1/1)[1] |
| | Long | 13 (1/0) | 14 (1/1)[1] |
| TAS | Byte | 10 (1/0) | 15 (2/1)[1][2] |
| TST | Byte, Word | 7 (1/0) | 7 (1/0)[1] |
| | Long | 7 (1/0) | 7 (1/0)[1] |

**Notes**

1. Add effective address calculation time.
2. Subtract one read cycle (–4(1/0)) from effective address calculation.
3. Subtract two read cycles (–8(2/0)) from effective address calculation.

**Table 105** Bit manipulation instruction clock periods

| INSTRUCTION | SIZE | DYNAMIC | | STATIC | |
|---|---|---|---|---|---|
| | | REGISTER | MEMORY | REGISTER | MEMORY |
| BCHG | Byte | – | 14 (1/1)[1] | – | 21 (2/1)[1] |
| | Long | 10 (1/0) | – | 17 (2/0) | – |
| BCLR | Byte | – | 14 (1/1)[1] | – | 21 (2/1)[1] |
| | Long | 10 (1/0) | – | 17 (2/0) | – |
| BSET | Byte | – | 14 (1/1)[1] | – | 21 (2/1)[1] |
| | Long | 10 (1/0) | – | 17 (2/0) | – |
| BTST | Byte | – | 7 (1/0)[1] | – | 14 (2/0)[1] |
| | Long | 7 (1/0) | – | 14 (2/0) | – |

**Note**

1. Add effective address calculation time.

**Table 106** Conditional instruction clock periods

| INSTRUCTION | DISPLAY | TRAP OR BRANCH | |
|---|---|---|---|
| | | TAKEN | NOT TAKEN |
| Bcc | Byte | 13 (1/0) | 13 (1/0) |
| | Word | 14 (2/0) | 14 (2/0) |
| BRA | Byte | 13 (1/0) | – |
| | Word | 14 (2/0) | – |
| BSR | Byte | 21 (1/2) | – |
| | Word | 22 (2/2) | – |
| DBcc | cc True | – | 14 (2/0) |
| | cc False | 17 (2/0) | 17 (3/2) |
| CHK | – | 70 (3/4)(1) | 19 (1/0)(1) |
| TRAPV | – | 55 (3/4) | 10 (1/0) |

**Note**

1. Add effective address calculation time.

**Table 107** JMP, JSR, LEA, PEA, MOVEM instruction clock periods
n = number of registers to move.

| INSTRUCTION | SIZE | (An) | (An)+ | –(An) | d(An) | d(An, Xi) | xxx.S | xxx.L | d(PC) | d(PC, Xi) |
|---|---|---|---|---|---|---|---|---|---|---|
| JMP | – | 7 (1/0) | – | – | 14 (2/0) | 17 (2/0) | 14 (2/0) | 18 (3/0) | 14 (2/0) | 17 (2/0) |
| JSR | – | 18 (1/2) | – | – | 25 (2/2) | 28 (2/2) | 25 (2/2) | 28 (2/2) | 25 (2/2) | 28 (2/2) |
| LEA | – | 7 (1/0) | – | – | 14 (2/0) | 17 (2/0) | 14 (2/0) | 18 (3/0) | 14 (2/0) | 17 (2/0) |
| PEA | – | 18 (1/2) | – | – | 25 (2/2) | 28 (2/2) | 25 (2/2) | 28 (2/2) | 25 (2/2) | 28 (2/2) |
| MOVEM M → R | Word | 26+7n (2+n/0) | 26+7n (2+n/0) | – | 30+7n (3+n/0) | 33+7n (3+n/0) | 30+7n (3+n/0) | 34+7n (4+n/0) | 30+7n (3+n/0) | 33+7n (3+n/0) |
| | Long | 26+11n (2+2n/0) | 26+11n (2+2n/0) | – | 30+11n (3+2n/0) | 33+11n (3+2n/0) | 30+11n (3+2n/0) | 34+11n (4+2n/0) | 30+11n (3+2n/0) | 33+11n (3+2n/0) |
| MOVEM R → M | Word | 23+7n (2/n) | – | 23+7n (2/n) | 27+7n (3/n) | 30+7n (3/n) | 27+7n (3/n) | 31+7n (4/n) | – | – |
| | Long | 23+11n (2/2n) | – | 23+11n (2/2n) | 27+11n (3/2n) | 30+11n (3/2n) | 27+11n (3/2n) | 31+11n (4/2n) | – | – |

**Table 108** Multi-precision Instruction Clock Periods

| INSTRUCTION | SIZE | op Dn, An | op M, M |
|---|---|---|---|
| ADDX | Byte, Word | 7 (1/0) | 28 (3/1) |
| | Long | 7 (1/0) | 40 (5/2) |
| CMPM | Byte, Word | – | 18 (3/0) |
| | Long | – | 26 (5/0) |
| SUBX | Byte, Word | 7 (1/0) | 28 (3/1) |
| | Long | 7 (1/0) | 40 (5/2) |
| ABCD | Byte | 10 (1/0) | 31 (3/1) |
| SBCD | Byte | 10 (1/0) | 31 (3/1) |

**Table 109** Miscellaneous Clock Periods

| INSTRUCTION | SIZE | REGISTER | MEMORY | REGISTER TO MEMORY | MEMORY TO REGISTER |
|---|---|---|---|---|---|
| ANDI to CCR | – | 14 (2/0) | – | – | – |
| ANDI to SR | – | 14 (2/0) | – | – | – |
| EORI to CCR | – | 14 (2/0) | – | – | – |
| EORI to SR | – | 14 (2/0) | – | – | – |
| EXG | – | 13 (2/0) | – | – | – |
| EXT | Word | 7 (1/0) | – | – | – |
| | Long | 7 (1/0) | – | – | – |
| LINK | – | 25 (2/2) | – | – | – |
| MOVE from SR | – | 7 (1/0) | 11 (1/1)[1] | – | – |
| MOVE to CCR | – | 10 (1/0) | 10 (1/0)[1] | – | – |
| MOVE to SR | – | 10 (1/0) | 10 (1/0)[1] | – | – |
| MOVE from USP | – | 7 (1/0) | – | – | – |
| MOVE to USP | – | 7 (1/0) | – | – | – |
| MOVEP | Word | – | – | 25 (2/2) | 22 (4/0) |
| | Long | – | – | 39 (2/4) | 36 (6/0) |
| NOP | – | 7 (1/0) | – | – | – |
| ORI to CCR | – | 14 (2/0) | – | – | – |
| ORI to SR | – | 14 (2/0) | – | – | – |
| RESET | – | 154 (1/0) | – | – | – |
| RTE short format | – | – | – | – | – |
| RTE long format | | | | | – |
| no rerun | – | 140 (18/0) | – | – | – |
| with rerun | – | 146 (18/0) | – | – | – |
| return of TAS | – | 151 (19/0) | – | – | – |
| RTR | – | 22 (4/0) | – | – | – |
| RTS | – | 15 (3/0) | – | – | – |
| STOP | – | 17 (2/0) | – | – | – |
| SWAP | – | 7 (1/0) | – | – | – |
| UNLK | – | 15 (3/0) | – | – | – |

**Note**

1. Add effective address calculation time.

**Table 110** Exception processing clock periods

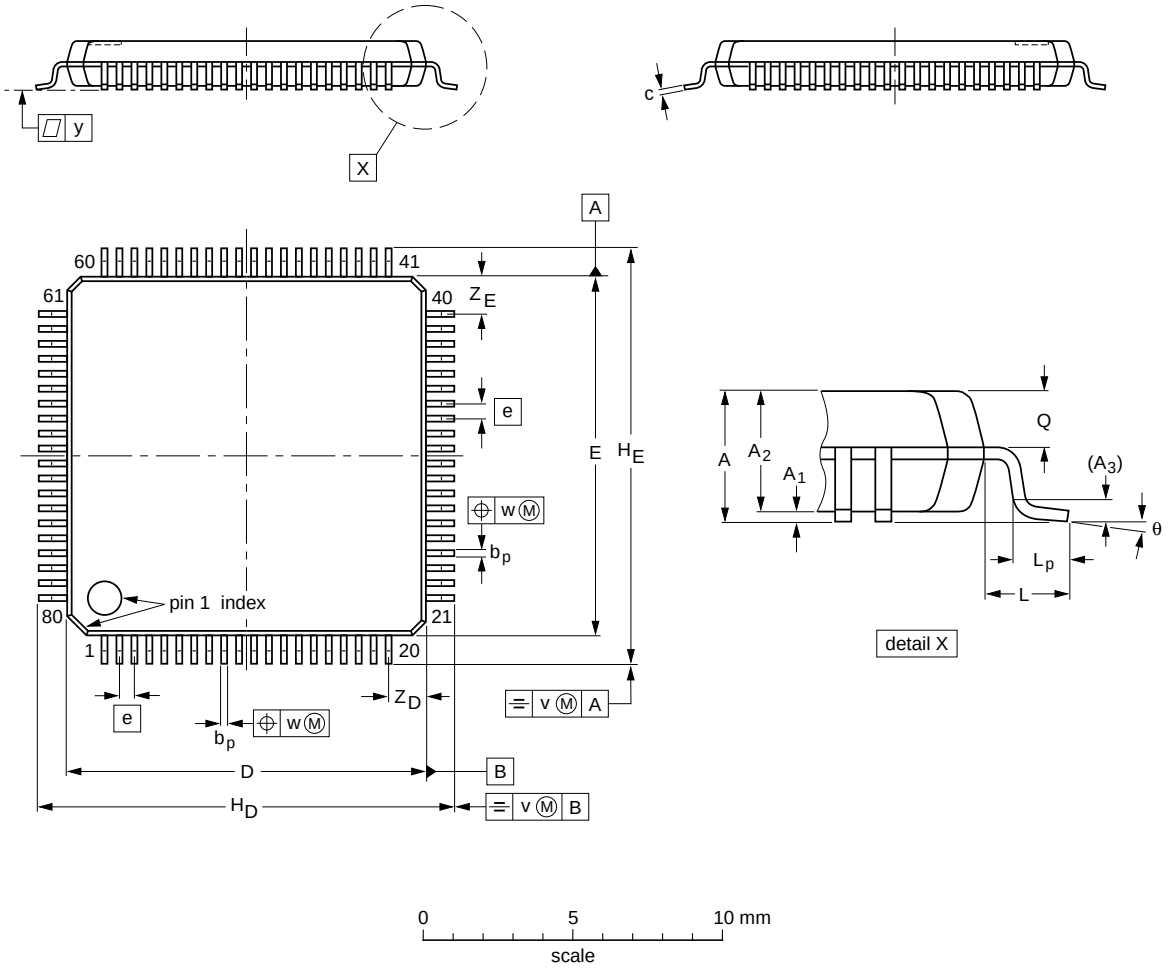
| EXCEPTION | NUMBER OF CLOCK PERIODS |
|---|---|
| Address error | 158 (3/17) |
| Bus error | 158 (3/17) |
| Interrupt | 65 (4/4)[1] |
| Illegal instruction | 55 (3/4) |
| Privilege instruction | 55 (3/4) |
| Trace | 55 (3/4) |
| Trap | 52 (3/4) |
| Divide by zero | 64 (3/4)[2] |
| RESET[3] | 43 (4/0) |

**Notes**

1. The interrupt acknowledge bus cycle is assumed to take four external clock periods.
2. Add effective address calculation time.
3. Indicates the maximum time from when $\overline{\text{RESET}}$ and $\overline{\text{HALT}}$ are first sampled as negated to first instruction fetch.

## 28 PACKAGE OUTLINE

**LQFP80: plastic low profile quad flat package; 80 leads; body 12 x 12 x 1.4 mm** **SOT315-1**




**DIMENSIONS (mm are the original dimensions)**

| UNIT | A max. | $A_1$ | $A_2$ | $A_3$ | $b_p$ | c | $D^{(1)}$ | $E^{(1)}$ | e | $H_D$ | $H_E$ | L | $L_p$ | Q | v | w | y | $Z_D^{(1)}$ | $Z_E^{(1)}$ | θ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| mm | 1.6 | 0.16<br>0.04 | 1.5<br>1.3 | 0.25 | 0.25<br>0.13 | 0.18<br>0.12 | 12.1<br>11.9 | 12.1<br>11.9 | 0.5 | 14.15<br>13.85 | 14.15<br>13.85 | 1.0 | 0.7<br>0.3 | 0.70<br>0.58 | 0.2 | 0.15 | 0.1 | 1.45<br>1.05 | 1.45<br>1.05 | $4^o$<br>$0^o$ |

**Note**

1. Plastic or metal protrusions of 0.25 mm maximum per side are not included.

| OUTLINE VERSION | REFERENCES | | | | EUROPEAN PROJECTION | ISSUE DATE |
|---|---|---|---|---|---|---|
| | IEC | JEDEC | EIAJ | | | |
| SOT315-1 | | | | | | ~~92-03-24~~<br>95-12-19 |

## 29 SOLDERING

### 29.1 Introduction

There is no soldering method that is ideal for all IC packages. Wave soldering is often preferred when through-hole and surface mounted components are mixed on one printed-circuit board. However, wave soldering is not always suitable for surface mounted ICs, or for printed-circuits with high population densities. In these situations reflow soldering is often used.

This text gives a very brief insight to a complex technology. A more in-depth account of soldering ICs can be found in our *"IC Package Databook"* (order code 9398 652 90011).

### 29.2 Reflow soldering

Reflow soldering techniques are suitable for all LQFP packages.

Reflow soldering requires solder paste (a suspension of fine solder particles, flux and binding agent) to be applied to the printed-circuit board by screen printing, stencilling or pressure-syringe dispensing before package placement.

Several techniques exist for reflowing; for example, thermal conduction by heated belt. Dwell times vary between 50 and 300 seconds depending on heating method. Typical reflow temperatures range from 215 to 250 °C.

Preheating is necessary to dry the paste and evaporate the binding agent. Preheating duration: 45 minutes at 45 °C.

### 29.3 Wave soldering

Wave soldering is **not** recommended for LQFP packages. This is because of the likelihood of solder bridging due to closely-spaced leads and the possibility of incomplete solder penetration in multi-lead devices.

**If wave soldering cannot be avoided, the following conditions must be observed:**

- **A double-wave (a turbulent wave with high upward pressure followed by a smooth laminar wave) soldering technique should be used.**
- **The footprint must be at an angle of 45° to the board direction and must incorporate solder thieves downstream and at the side corners.**

**Even with these conditions, do not consider wave soldering LQFP packages LQFP48 (SOT313-2), LQFP64 (SOT314-2) or LQFP80 (SOT315-1).**

During placement and before soldering, the package must be fixed with a droplet of adhesive. The adhesive can be applied by screen printing, pin transfer or syringe dispensing. The package can be soldered after the adhesive is cured.

Maximum permissible solder temperature is 260 °C, and maximum duration of package immersion in solder is 10 seconds, if cooled to less than 150 °C within 6 seconds. Typical dwell time is 4 seconds at 250 °C.

A mildly-activated flux will eliminate the need for removal of corrosive residues in most applications.

### 29.4 Repairing soldered joints

Fix the component by first soldering two diagonally-opposite end leads. Use only a low voltage soldering iron (less than 24 V) applied to the flat part of the lead. Contact time must be limited to 10 seconds at up to 300 °C. When using a dedicated tool, all other leads can be soldered in one operation within 2 to 5 seconds between 270 and 320 °C.

## 30 DEFINITIONS

| **Data sheet status** | |
|---|---|
| Objective specification | This data sheet contains target or goal specifications for product development. |
| Preliminary specification | This data sheet contains preliminary data; supplementary data may be published later. |
| Product specification | This data sheet contains final product specifications. |
| **Limiting values** | |
| Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability. | |
| **Application information** | |
| Where application information is given, it is advisory and does not form part of the specification. | |

## 31 LIFE SUPPORT APPLICATIONS

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips for any damages resulting from such improper use or sale.

## 32 PURCHASE OF PHILIPS I²C COMPONENTS

Purchase of Philips I²C components conveys a license under the Philips' I²C patent to use the components in the I²C system provided the system conforms to the I²C specification defined by Philips. This specification can be ordered using the code 9398 393 40011.

**NOTES**

**NOTES**

# *Philips Semiconductors – a worldwide company*


**Argentina:** see South America

**Australia:** 34 Waterloo Road, NORTH RYDE, NSW 2113,
Tel. +61 2 9805 4455, Fax. +61 2 9805 4466

**Austria:** Computerstr. 6, A-1101 WIEN, P.O. Box 213,
Tel. +43 1 60 101, Fax. +43 1 60 101 1210

**Belarus:** Hotel Minsk Business Center, Bld. 3, r. 1211, Volodarski Str. 6,
220050 MINSK, Tel. +375 172 200 733, Fax. +375 172 200 773

**Belgium:** see The Netherlands

**Brazil:** see South America

**Bulgaria:** Philips Bulgaria Ltd., Energoproject, 15th floor,
51 James Bourchier Blvd., 1407 SOFIA,
Tel. +359 2 689 211, Fax. +359 2 689 102

**Canada:** PHILIPS SEMICONDUCTORS/COMPONENTS,
Tel. +1 800 234 7381

**China/Hong Kong:** 501 Hong Kong Industrial Technology Centre,
72 Tat Chee Avenue, Kowloon Tong, HONG KONG,
Tel. +852 2319 7888, Fax. +852 2319 7700

**Colombia:** see South America

**Czech Republic:** see Austria

**Denmark:** Prags Boulevard 80, PB 1919, DK-2300 COPENHAGEN S,
Tel. +45 32 88 2636, Fax. +45 31 57 1949

**Finland:** Sinikalliontie 3, FIN-02630 ESPOO,
Tel. +358 9 615800, Fax. +358 9 61580/xxx

**France:** 4 Rue du Port-aux-Vins, BP317, 92156 SURESNES Cedex,
Tel. +33 1 40 99 6161, Fax. +33 1 40 99 6427

**Germany:** Hammerbrookstraße 69, D-20097 HAMBURG,
Tel. +49 40 23 53 60, Fax. +49 40 23 536 300

**Greece:** No. 15, 25th March Street, GR 17778 TAVROS/ATHENS,
Tel. +30 1 4894 339/239, Fax. +30 1 4814 240

**Hungary:** see Austria

**India:** Philips INDIA Ltd, Shivsagar Estate, A Block, Dr. Annie Besant Rd.
Worli, MUMBAI 400 018, Tel. +91 22 4938 541, Fax. +91 22 4938 722

**Indonesia:** see Singapore

**Ireland:** Newstead, Clonskeagh, DUBLIN 14,
Tel. +353 1 7640 000, Fax. +353 1 7640 200

**Israel:** RAPAC Electronics, 7 Kehilat Saloniki St, TEL AVIV 61180,
Tel. +972 3 645 0444, Fax. +972 3 649 1007

**Italy:** PHILIPS SEMICONDUCTORS, Piazza IV Novembre 3,
20124 MILANO, Tel. +39 2 6752 2531, Fax. +39 2 6752 2557

**Japan:** Philips Bldg 13-37, Kohnan 2-chome, Minato-ku, TOKYO 108,
Tel. +81 3 3740 5130, Fax. +81 3 3740 5077

**Korea:** Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL,
Tel. +82 2 709 1412, Fax. +82 2 709 1415

**Malaysia:** No. 76 Jalan Universiti, 46200 PETALING JAYA, SELANGOR,
Tel. +60 3 750 5214, Fax. +60 3 757 4880

**Mexico:** 5900 Gateway East, Suite 200, EL PASO, TEXAS 79905,
Tel. +9-5 800 234 7381

**Middle East:** see Italy

**Netherlands:** Postbus 90050, 5600 PB EINDHOVEN, Bldg. VB,
Tel. +31 40 27 82785, Fax. +31 40 27 88399

**New Zealand:** 2 Wagener Place, C.P.O. Box 1041, AUCKLAND,
Tel. +64 9 849 4160, Fax. +64 9 849 7811

**Norway:** Box 1, Manglerud 0612, OSLO,
Tel. +47 22 74 8000, Fax. +47 22 74 8341

**Philippines:** Philips Semiconductors Philippines Inc.,
106 Valero St. Salcedo Village, P.O. Box 2108 MCC, MAKATI,
Metro MANILA, Tel. +63 2 816 6380, Fax. +63 2 817 3474

**Poland:** Ul. Lukiska 10, PL 04-123 WARSZAWA,
Tel. +48 22 612 2831, Fax. +48 22 612 2327

**Portugal:** see Spain

**Romania:** see Italy

**Russia:** Philips Russia, Ul. Usatcheva 35A, 119048 MOSCOW,
Tel. +7 095 247 9145, Fax. +7 095 247 9144

**Singapore:** Lorong 1, Toa Payoh, SINGAPORE 1231,
Tel. +65 350 2538, Fax. +65 251 6500

**Slovakia:** see Austria

**Slovenia:** see Italy

**South Africa:** S.A. PHILIPS Pty Ltd., 195-215 Main Road Martindale,
2092 JOHANNESBURG, P.O. Box 7430 Johannesburg 2000,
Tel. +27 11 470 5911, Fax. +27 11 470 5494

**South America:** Rua do Rocio 220, 5th floor, Suite 51,
04552-903 São Paulo, SÃO PAULO - SP, Brazil,
Tel. +55 11 821 2333, Fax. +55 11 829 1849

**Spain:** Balmes 22, 08007 BARCELONA,
Tel. +34 3 301 6312, Fax. +34 3 301 4107

**Sweden:** Kottbygatan 7, Akalla, S-16485 STOCKHOLM,
Tel. +46 8 632 2000, Fax. +46 8 632 2745

**Switzerland:** Allmendstrasse 140, CH-8027 ZÜRICH,
Tel. +41 1 488 2686, Fax. +41 1 481 7730

**Taiwan:** PHILIPS TAIWAN Ltd., 23-30F, 66,
Chung Hsiao West Road, Sec. 1, P.O. Box 22978,
TAIPEI 100, Tel. +886 2 382 4443, Fax. +886 2 382 4444

**Thailand:** PHILIPS ELECTRONICS (THAILAND) Ltd.,
209/2 Sanpavuth-Bangna Road Prakanong, BANGKOK 10260,
Tel. +66 2 745 4090, Fax. +66 2 398 0793

**Turkey:** Talatpasa Cad. No. 5, 80640 GÜLTEPE/ISTANBUL,
Tel. +90 212 279 2770, Fax. +90 212 282 6707

**Ukraine**: PHILIPS UKRAINE, 4 Patrice Lumumba str., Building B, Floor 7,
252042 KIEV, Tel. +380 44 264 2776, Fax. +380 44 268 0461

**United Kingdom:** Philips Semiconductors Ltd., 276 Bath Road, Hayes,
MIDDLESEX UB3 5BX, Tel. +44 181 730 5000, Fax. +44 181 754 8421

**United States:** 811 East Arques Avenue, SUNNYVALE, CA 94088-3409,
Tel. +1 800 234 7381

**Uruguay:** see South America

**Vietnam:** see Singapore

**Yugoslavia:** PHILIPS, Trg N. Pasica 5/v, 11000 BEOGRAD,
Tel. +381 11 625 344, Fax.+381 11 635 777

**For all other countries apply to:** Philips Semiconductors, Marketing & Sales Communications,
Building BE-p, P.O. Box 218, 5600 MD EINDHOVEN, The Netherlands, Fax. +31 40 27 24825

**Internet:** http://www.semiconductors.philips.com





Printed in The Netherlands 647021/50/01/pp92 Date of release: 1996 Dec 11 Document order number: 9397 750 01261


*Let's make things better.*

**Philips Semiconductors**

PHILIPS