TEMIC

MATRA MHS

29C948

8 Channel HDLC/V.120 Protocol Controller

Introduction

The 29C948 is an 8 channel data link protocol controller circuit. It multiplexes/demultiplexes up to 8 full duplex data channels to support implementation of high speed data links based on either HDLC protocol or clear channel.

The device operates at layer 2 of OSI model as described by ISO standard. The 29C948 is connected to the serial PCM link on the line side and shared memory with a system host microprocessor on the system side.

The 29C948 processes transmit and receive data on a PCM communication medium in either E1 or T1 framing format. The device provides HDLC formatting/extracting functions for synchronous data and manages, for each active channel, access to the buffers into the shared memory. The 29C948 clear channel mode is programmable on any channel independently of all others.

The circuit is entirely compatible with ISDN specified by CCITT and it provides additional functions which support ISDN hyperchannel and user defined protocols. It ensures full compatibility with X.25, LAPB and LAPD protocols.

The 29C948 finds applications in several areas of telecommunication. This includes multimedia terminals and servers, network couplers, PABX signalling, multiplexers, etc.

Features

- Single chip CMOS monolithic device simplifies ISDN implementation
- Provides HDLC/V.120 or Clear Channel formatting for 8 full duplex 64 kbps channels
- Compatible with E1 and T1 PCM framing formats
- On chip receive and transmit context saving as well as buffer memory management function
- Provides 8 Full Duplex DMA channels (8 transmit, 8 receive)
- 8 bit data bus, 24 bit address bus, bus request/acknowledge handshake
- Master/Slave operation modes
- On chip oscillator, dynamic E1/T1 programmation
- 8 chained descriptors on chip for each active channel
- Independent and flexible transmit/receive PCM serial link
- On chip 16 bit CRC generation and checking using CCITT polynomial, automatic flag detection and transmission, zero bit insertion and deletion
- Programmable number of channels from 1 to 8 and loop back mode operations for test purposes
- Hyperchannel mode from 2 to 8 channels
- 33 MHz system clock
- Single 5V power supply, PLCC 68 package

Applications

- Multiplexers
- PABXs
- Computer Links
- Servers
- Routers
- Bridges
- Base stations

Application Diagram

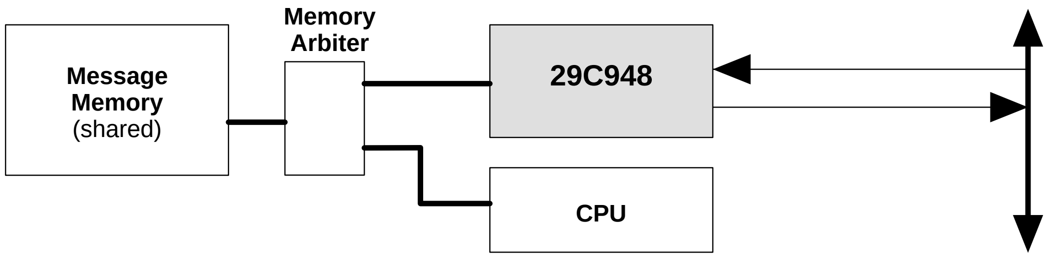


Figure 1. Functional Block Diagram

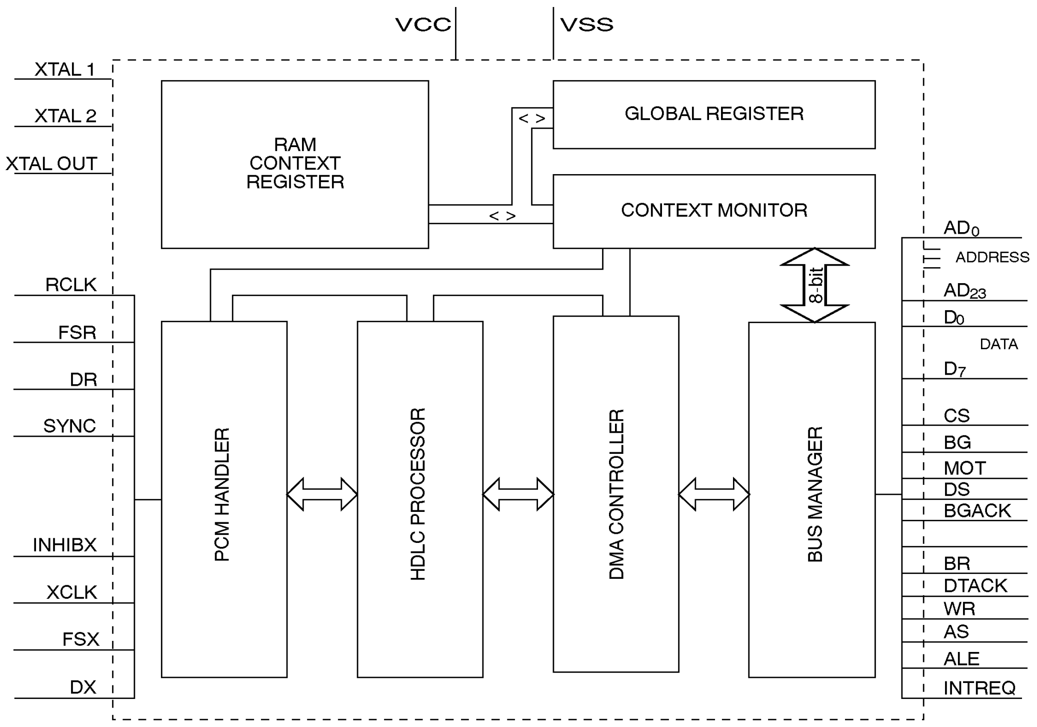
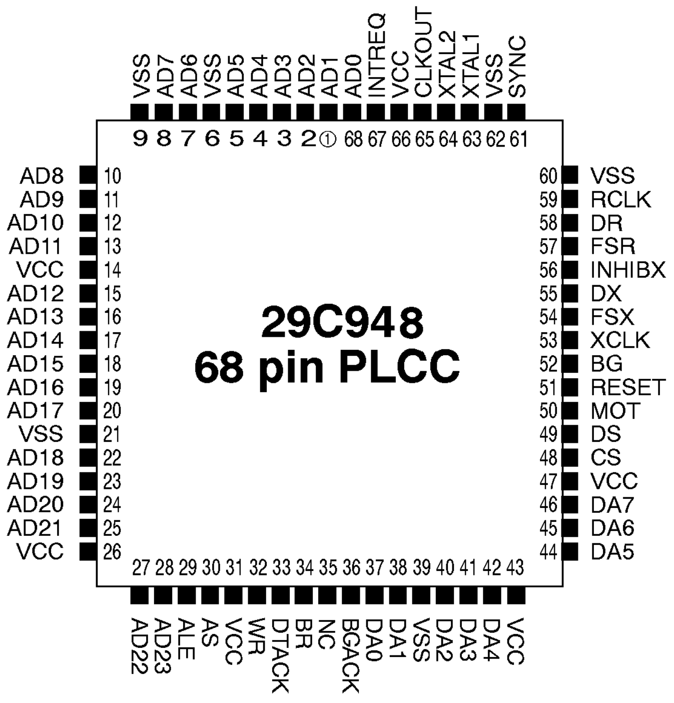


Figure 2. 68 Pin PLCC package pin out (top view).


29C948
68 pin PLCC
1 VSS, 2 AD0...
VSS 9, AD7 8, AD6 7, VSS 6, AD5 5, AD4 4, AD3 3, AD2 2, AD1 1, AD0 68, INTREQ 67, VCC 66, CLKOUT 65, XTAL2 64, XTAL1 63, VSS 62, SYNC 61
AD8 10, AD9 11, AD10 12, AD11 13, VCC 14, AD12 15, AD13 16, AD14 17, AD15 18, AD16 19, AD17 20, VSS 21, AD18 22, AD19 23, AD20 24, AD21 25, VCC 26
AD22 27, AD23 28, ALE 29, AS 30, VCC 31, WR 32, DTACK 33, BR 34, NC 35, BGACK 36, DA0 37, DA1 38, VSS 39, DA2 40, DA3 41, DA4 42, VCC 43
DA5 44, DA6 45, DA7 46, VCC 47, CS 48, DS 49, MOT 50, RESET 51, BG 52, XCLK 53, FSX 54, DX 55, INHIBX 56, FSR 57, DR 58, RCLK 59, VSS 60

1.0 Functional Description

The 29C948 multiplexes/demultiplexes up to 8 full duplex channels over the time slots of the PCM frames. The device can handle up to 8 separate channels or then hyperchannels. The hyperchannels are formed from two to eight time slots which can be dynamically concatenated to form an hyperchannel. The hyperchannel time slots can be adjacent or non adjacent in the PCM frame.

The internal functions of the 29C948 are partioned into 5 major operating blocks as shown in the functional block diagram.

The transmitted data is fetched time slot after time slot through DMA cycles in 8-bit parallel form from the buffers which are located in the external message memory. The data is processed by performing protocol data formatting and rate adaptation. After this operation the data is sent serially to the PCM link.

The received data from the PCM serial link is processed, channel after channel, by performing rate adaptation and data frame protocol deformatting. The receive data is transfered by DMA to the message memory in 8-bit parallel form into the channel allocated buffers.

For each channel is allocated eight chained buffers to transmit data and eight buffers to receive data. The buffers have a maximum capacity of 64 kbytes.

The 29C948 can address up to 128 buffers which are 64 kbytes each.

Two types of operating modes are available. Global and Channel Operating mode. These operating modes are specified in the control registers of the device and each channel is processed depending on the selected mode.

Global Operating mode

The Global Operating Mode applies to every channel and selects one of the two possible functionalities which are described in Table 1 :

Table 1 : Global Operation Modes.

T1/E1 mode (8-channels)	PCM simple or double clock
PCM slave or master mode	Loop back mode
PCM frame sync offset control	DX pin mode
Normal operating mode	Test mode

Channel Operating mode

The Channel Operating mode is programmable within the Global Mode on any channel independently of the others. Also, the receive mode is independent of the transmit mode and is programmed separately.

The programmable channel parameters are presented in the Table 2 :

Table 2 : Channel Operation Modes.

HDLC mode	Clear Channel mode
Time Slot valid	Time Slot unvalid
Flag sharing enable/disable	Reserved
Rate Adaptation/ Fill Mask	Reserved
Channel number	Reserved
Buffer descriptors DMA descriptors	Buffer size/word count Start address Processing status

The Global Operating mode parameters are permanently applied to the functional blocks as they do not change from one time slot to the other one.

The Channel Operating modes are switched on for every time slot in each channel.

This so called context switching is performed by the Context Monitor on each time slot. The context parameters are stored in the internal RAM. Processing of the time slot takes place over the time slot period. At the end of the time slot the context is updated and saved until its next occurance in the PCM frame. The Context Monitor also provides a path to the CPU to access the internal RAM and all registers through the Bus Manager.

The 29C948 operates at an internal frequency of 33 MHz driven either by an external clock or an internal crystal oscillator.

Pin Assignment

Pin number	Pin name	Pin number	Pin name	Pin number	Pin name	Pin number	Pin name
1	AD1	18	AD15	35	NC	52	BG
2	AD2	19	AD16	36	Vcc	53	XCLK
3	AD3	20	AD17	37	DA0	54	FSX
4	AD4	21	Vss	38	DA1	55	DX
5	AD5	22	AD18	39	Vss	56	INHIBX
6	AD6	23	AD19	40	DA2	57	FSR
7	AD7	24	AD20	41	DA3	58	DR
8	AD8	25	AD21	42	DA4	59	RCLK
9	Vss	26	Vcc	43	Vcc	60	Vss
10	AD8	27	AD22	44	DA5	61	SYNC
11	AD9	28	AD23	45	DA6	62	Vss
12	AD10	29	ALE	46	DA7	63	XTAL1
13	AD11	30	-AS	47	Vcc	64	XTAL2
14	Vcc	31	Vcc	48	-CS	65	CLKOUT
15	AD12	32	-WR	49	DS	66	Vcc
16	AD13	33	DTACK	50	MOT	67	INTREQ
17	AD14	34	-BR	51	RESET	68	AD0

Note : ’-’ indicates active low signal.
’NC’ indicates non connected pin.

Interface Signal Definitions

Bus Interface

Symbol	Type	Name and Description	Active INTEL	Mode MOTOROLA
AD0-AD23	I/O	**Address Bus :** bidirectional address bus between 29C948 and the shared memory or CPU.	NA	NA
DA0-DA7	I/O	**Data Bus :** Bidirectional data bus between 29C948 and the shared memory or CPU.	NA	NA
-CS	I	**Chip Select :** 29C948 chip select. Active low signal.	Low	Low
ALE	I/O	**Address Latch Enable :** indicates valid address on address bus when the signal goes high in Intel mode. Tied to ground in Motorola mode.	High	NA
-AS	I/O	**Address Strobe :** in Motorola mode address strobe. Read strobe in Intel mode	Low	Low
-WR	I/O	**Write Strobe :** performs either data write cycle into the shared memory or CPU write cycle into the 29C948 internal registers.	Low	Low
DTACK	I/O	**Read/Write Acknowledge :** when input, acknowledges DMA read or write cycles. When output, acknowledges CPU read or write cycles into the 29C948 internal RAM	High	Low
-BR	O	**Bus Request :** system bus request from 29C948. Open collector output.	Low	Low
BG	I	**Bus Grant :** 29C948 is granted system bus control	High	Low
BGACK	I/O	**Bus Grant Acknowledge Output :** Acknowledges that system bus is under 29C948 control.	NA	Low
DS	I	**Data Strobe :** indicates data reading from or data writing into the 29C948. Motorola mode, only.	NA	Low
INTREQ	O	**Interrupt Request :** interrupt request to CPU. Active high.	High	High
RESET	I	Reset : resets global mode registers. All registers and channels put in default values.	High	Low
PCM Handler				
XCLK	I	**Transmit Bit Clock :** square wave input from the master timing source for the transmit channel. 2.048 or 4.096 Mbps in E1 mode and 1.544 or 3.048 Mbps in T1 mode.	NA	NA
RCLK	I	**Receive Bit Clock :** square wave input from the master timing source for the receive channel. 2.048 or 4.096 Mbps in E1 mode and 1.544 or 3.048 Mbps in T1 mode.	NA	NA
FSX	I/O	**Transmit Frame Sync :** 8 kHz frame synchronisation pulse. Input in slave operating mode and output in master operating mode.	NA	NA
FSR	I/O	**Receive Frame Sync :** 8 kHz frame synchronisation pulse. Input in slave operating mode and output in master operating mode.	NA	NA
DX	O	**Transmit Data :** serial data output from the 29C948 to PCM bus.		
DR	I	**Receive Data :** serial data input from the PCM bus to 29C948.	NA	NA
INHIBX	O	**Transmit Time Slot Inhibit :** indicates that transmit of current time slot is inhibited. Active high.		
SYNC	I	**Sync :** input asserting that receive PCM alignement is correct.	High	High

Other

Symbol	Type	Name and Description
MOT	I	**Motorola/Intel :** selects system bus interface. High leval = Motorola mode
XTAL1	I	**Crystal oscillator or external chip clock input**
XTAL2	O	**Crystal oscillator output**
CLKOUT	O	**Chip clock buffered output**
Vcc	I	**Supply :** + 5V power supply
Vss	I	**Ground :** power supply ground

Note : ’-’ indicates active low signal.

Mapping of the internal RAM is shown in Table 3. Register definitions and functions are given starting on chapter 3.0.

Table 3 : Register Mapping into Internal RAM.

Modes		Addresses	RAM/Register Mapping
CHANNEL MODE	RECEIVE	0_H to $3F_H$	Receive context ; Time slot 0
		40_H to $7F_H$	Receive context ; Time slot 1
		.. ..	... ...
		.. ..	Receive context ; Time slot N–1
		.. ..	Receive context ; Time slot N
		.. ..	Receive context ; Time slot N+1
		.. ..	... ...
		$7A0_H$ to $7CF_H$	Receive context ; Time slot 30
		$7C0_H$ to $7FF_H$	Receive context ; Time slot 31
	TRANSMIT	800_H to $83F_H$	Transmit context ; Time slot 0
		840_H to $87F_H$	Transmit context ; Time slot 1
		.. ..	... ...
		.. ..	Transmit context ; Time slot N–1
		.. ..	Transmit context ; Time slot N
		.. ..	Transmit context ; Time slot N+1
		.. ..	... ...
		$F80_H$ to FBF_H	Transmit context ; Time slot 30
		FCO_H to FFF_H	Transmit context ; Time slot 31
GLOBAL MODE		1000_H to $100D_H$	Global mode parameters and interrupt request

Global operating mode parameters are permanently applied to the functional logic blocks as they do not change from one time slot to the other, except for a deliberate CPU intervention. On the contrary, the context parameters, pertaining to a specific time slot, are switched on every time slot. Context switching is performed by the CONTEXT MONITOR, as explained further on.

To summarize : On each successive time slot of the PCM frame and within the global operating mode, specific time slot context is retrieved from the internal RAM at the beginning of the time slot. Processing actions then take place over the time slot length, according parameters of the context, which, by the end of the slot, is updated and saved until its next occurence in the PCM frame, 125 µs later.

Let us remark that during a time slot (3.9 µs in CEPT), assuming it is valid and there are data being received and transmitted on this channel, several discrete actions are taking place ; ie. Context restoring- updating-saving, data fetching (transmit) and storing (receive), current CRC computation, serial data processing (transmit/receive), etc... To do so, the 29C948 operates at an internal frequency of up to 33 MHz (30 ns period) driven either by an external clock or an internal crystal oscillator as shown on AC Electrical Characteristics. Furthermore, the 29C948 uses throughout its synchronous design a pipeline architecture. Pipeline operations are illustrated in fig. 3.

The CONTEXT MONITOR also provides, through the BUS MANAGER, a path to the host CPU for accessing the internal RAM and thus, all registers.

Figure 3. Context Monitor pipe-line.

PCM Frame →

Time slot N–1	Time slot N	Time slot N+1
Read context time slot N	Read context time slot N+1	Read context time slot N+2
Execute context time slot N–1	Execute context time slot N	Execute context time slot N+1
Save context time slot N–2	Save context time slot N–1	Save context time slot N

2.0 Functional Logic Blocks

Functional logic blocks performances are controlled and driven from the global and context registers, which mapping in the 29C948 internal RAM is shown in Table 3. A quicker understanding of the logic blocks operations will be helped by looking at details of the control register contents in the “REGISTER DEFINITION”, starting on chapter 3.0.

2.1. PCM Handler

The PCM HANDLER is mostly controlled by the global registers : ***MODE***, ***ROFFSET***, ***XOFFSET***, ***GLOBINH***, ***TEST*** and ***TESTEXT***. It transmits serial data on DX output and receives on DR input. It performs receive/transmit data synchronization using XCLK and RCLK clock inputs and generates an internal reference, bit 0/time slot 0, with an adjustable offset against the external PCM frame syncs, FSR/FSX. It also supplies the bit clock to the HDLC PROCESSOR logic block.

The whole 29C948 is programmable in either master or slave mode with respect to FSX and FSR frame sync (see ***MODE*** global register on chapter 3.2).

SLAVE mode : Frame sync, FSX/FSR, and bit clock, XCLK/RCLK, are input signals. Width of FSX/FSR pulses must be at least one XCLK/RCLK period.

MASTER mode : The 29C948 generates the frame sync FSX/FSR. They are derived from XCLK/RCLK bit clock inputs. FSX/FSR pulse widths are equal to one XCLK/RCLK period.

XCLK and RCLK bit clocks may be asynchronous, however the loop back mode is not then possible.

DX output driving mode is programmable as “tri-state”, or “always high” or “wired OR” (See DXC0, DXC1 in ***MODE*** chapter 3.2). DX is forced to “1” or high impedance when the time slot is not valid.

SYNC input is used in conjunction with a FRAMER device. SYNC asserts the status of the received frame, according the truth table below.

SYNC	”1”	”0”
Frame	synchronized	not synchronized

When SYNC goes low, incoming data is ignored on every time slot and a bit (Lsyn) is automatically set into ***RCTST*** status registers of all channels (Receive DMA descriptors chapter 3.11). Received data will again be only considered and only stored after SYNC recovery.

Whenever a framer device is not used, the SYNC input pin must be tied to Vcc.

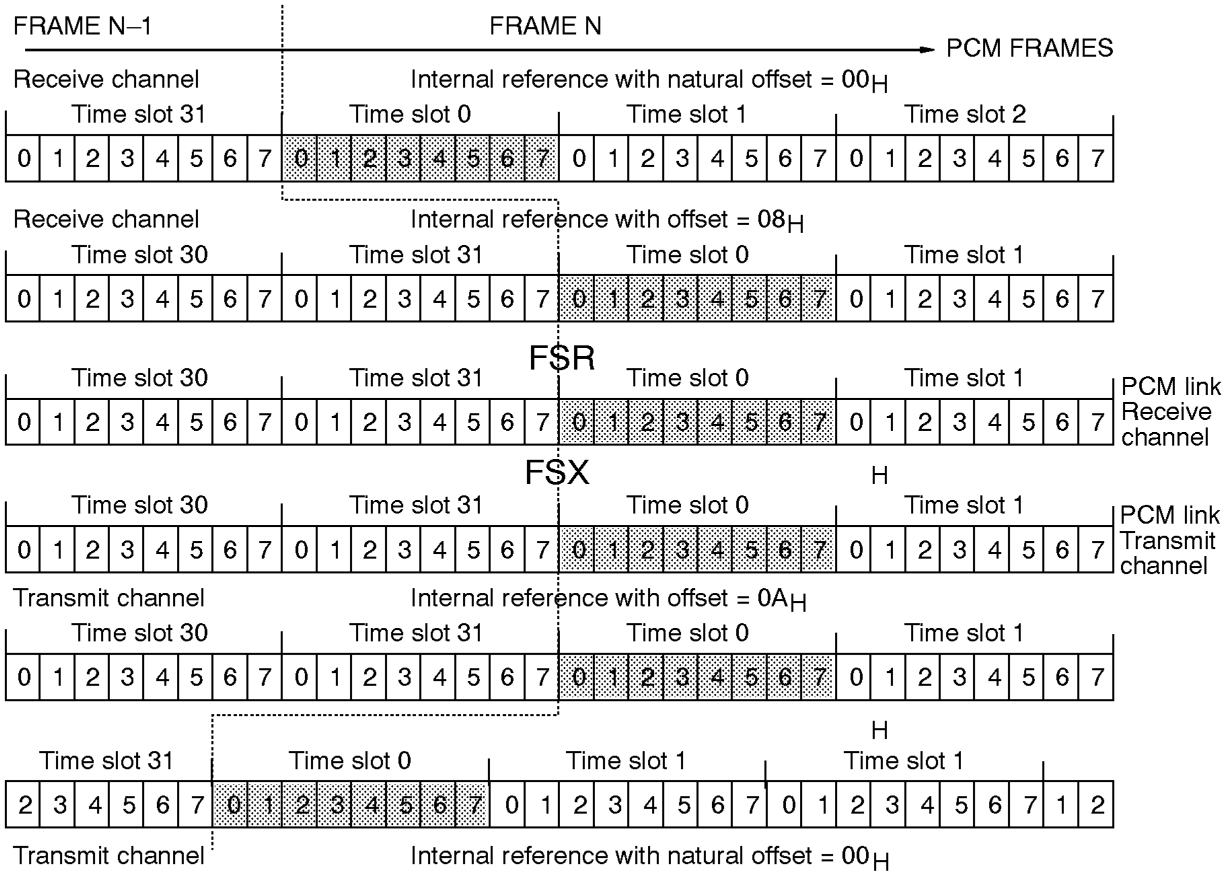
INHIBX asserts that the current time slot is inhibited. This output goes high, one bit clock period before the related time slot. INHIBX may be used with a framer device to force a preset idle code during the inhibited time slot. That feature is specially usefull with AMI coding in T1/DS1 application.

FRAME SYNC OFFSET. (***ROFFSET*** and ***XOFFSET*** registers, chapter 3.3)

Because of inherent design delays, the internal PCM frame reference-bit 0/time slot 0 – must be shifted so that, seen from the outside, it appears in sync with the FSX/FSR PCM link. Offsets are separately programmable on both transmit and receive side, either in CEPT or T1/DS1 mode.

Programming resolution is one bit clock period (1/2 bit clock resolution in double clock mode). Loop back mode is only possible when transmitter and receiver are synchronous and that transmit and receive channels are aligned. That is to say when both the external transmit and receive time slot references appear at the same time (same bit clock, same FRAME SYNC as shown fig 5). Fig. 4 shows the natural internal reference offsets against the external FSX/FSR ISDN line.

Figure 4. Internal References and Offsets.


FRAME N–1
FRAME N
PCM FRAMES
Receive channel
Internal reference with natural offset = 00H
Receive channel
Internal reference with offset = 08H
FSR
PCM link Receive channel
FSX
PCM link Transmit channel
Transmit channel
Internal reference with offset = 0AH
Transmit channel
Internal reference with natural offset = 00H
Time slot 31
Time slot 30
Time slot 0
Time slot 1
Time slot 2

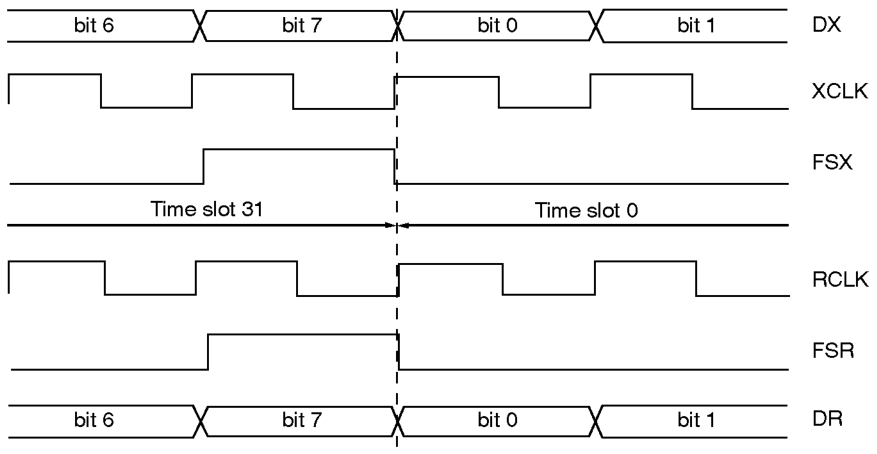
Natural offset values are 8 bit clock on the receive side and 10 bit clock on the transmit side. Thus, assuming that a given application brings in additional delay worth N bit clock, the minimum offset programming on the ***ROFFSET*** register becomes 08_H+N (*10_H+2N in double clock mode*) and will be OA_H+N (*14_H+2N in double clock mode*) on the ***XOFFSET*** register.

Delays on transmit and receive channels were designed different because of 29C948 synchronous design and its full duplex capability. On every time slot the transmit channel is processed first, two bit clock periods (1 µs in CEPT mode) before the receive channel. A side effect of this feature is to possibly allow, on a given time slot, a successfull receive DMA cycle completion, even if a time out has occured on the transmit DMA cycle.

As shown on the diagram figure 5, in slave mode, single clock, assuming that the transmit offset register is properly set, bit 0/time slot 0 of the transmit frame is output on first XCLK rising edge after FSX has been sampled to 1 by XCLK falling edge.

Similarly, if the receive offset register is correctly set, bit 0/time slot 0 of the received frame is sampled on the first RCLK falling edge after FSR was sampled to 1 by RCLK falling edge.

Figure 5. Offset Programming.



2.2. HDLC Processor

The HDLC PROCESSOR logic block is mostly driven and controlled by the context HDLC descriptor. It receives data bytes from the DMA CONTROLLER, processes and transfers them in series, LSB first, to the PCM HANDLER. Conversely, it receives data in series from the PCM HANDLER and, processes and sends them in byte form to the DMA CONTROLLER. The HDLC PROCESSOR also performs data rate adaptation with fill registers (transmit) or mask registers (receive).

The HDLC PROCESSOR is shared and successively used by all channels of the PCM frame. It is programmable in either HDLC or clear channel mode (see ***RMOD*** and ***XMOD*** registers, chapter 3.7).

HDLC Mode

In that mode basic HDLC frame formatting is performed, as shown in figure 7.

Figure 6. HDLC Framing Format.

$7E_H$	**$N \times 8$ bits**	**2×8 bits**	**$7E_H$**
Flag	Data	CRC	Flag

Frame check sequence (CRC) is computed from CCITT polynomial : $X^{16} + X^{12} + X^5 + 1$. Frame length is 64 kbytes maximum, plus flags and CRC. A data frame is always contained into one single buffer.

On any channel, flag sharing capability between successive transmitted data frames is programmed by setting up to “1” the **FLG**-bit into the ***XMOD*** register. In that mode the same flag may close a data frame and open the next frame.

Transmit :

Upon transmission the HDLC PROCESSOR performs protocol formatting and composes the HDLC frames with the data fetched in the buffers by the DMA CONTROLLER. It generates flags, abort and idle codes, inserts one “ZERO” after five consecutive “ONES” and computes the HDLC frame check sequence (CRC).

Flags are stuffed when there is no valid data to transmit while the time slot is valid.

Data frame closing occurs when either the related transmit data buffer is empty, (word count = 0), or after an unsuccessfull DMA attempt (time-out).

In the first case, a flag closes the frame and flags proceed to be sent while the time slot is valid with no data to transmit. In case of time out, the frame is closed with an **Abort** (**FF_H**) character. In both cases, a report is assembled and stored in the ***XCTST*** register of the related DMA descriptor.

Receive :

On the receive side, the HDLC PROCESSOR extracts serial data from the incoming stream on PCM HANDLER and performs protocol deformatting. It detects flags, abort and idle codes, suppresses inserted zeros, checks the HDLC frame check sequence (CRC). The resulting serial data is transferred to the DMA CONTROLLER to be stored in the buffers. All incoming bits following the detection of a flag ($7E_H$) are assembled in byte character.

Frame reception may be interrupted for various reasons outlined below. In all cases, the cause is reported, as explained hereafter, and stored into the related ***RCTST*** register of the DMA descriptor.

– Closing flag ($7E_H$) detection (**FTR** bit set to “1”),
– abort character (FF_H) detection (**ABRT** bit set to “1”)
– lost of Sync, (SYNC, pin# 61, = 0), (**LSYN** bit set to “1”)
– buffer full with yet a byte to store, (**UDF** bit set to “1”)
– DMA attempt is unsuccessfull, (**TOUT** bit set to “1”)
– attempt to use an unvalid buffer, (**DONE** bit set to “1”)

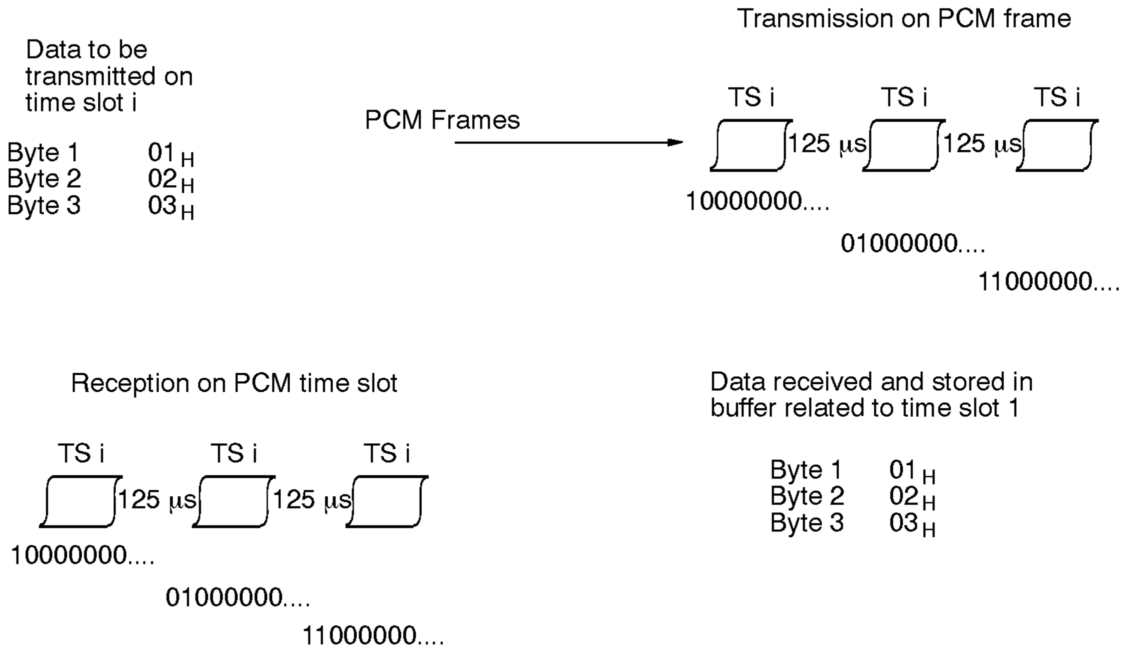
A minimum of 4 bytes (2 data bytes + CRC) is required between 2 flags.

CLEAR Channel Mode

Flag generation/detection, bit insertion and CRC computing are disabled in clear channel mode. Buffers are automatically chained, one after another. Frame length is not limited to one buffer as in HDLC mode.

”Ones” are transmitted when data are not available or upon a DMA time out. As in HDLC mode, a status is assembled in that later event (***RCTST*** and ***XCTST*** status registers, chapter 3.11).

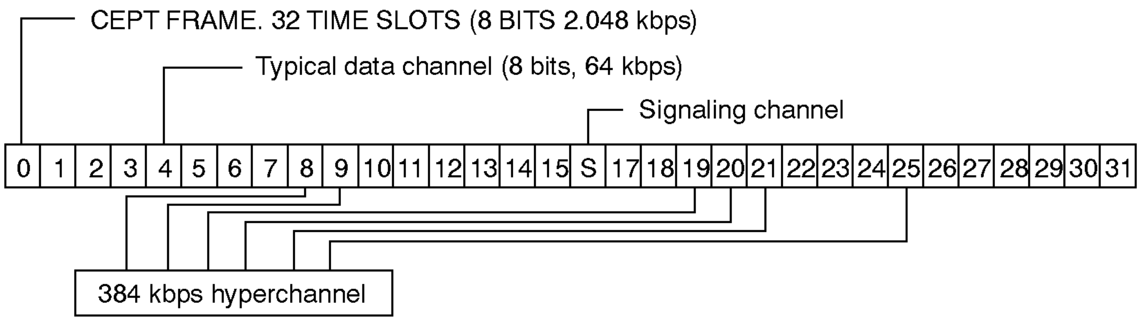
If the DMA CONTROLLER chains to a DMA descriptor that is not empty, then writing is not executed in the pointed buffer neither in the status register. Data characters alignment is kept on the PCM time slot. Following is an example of transmit/receive clear channel, 64 kbps, on time slot i.



Hyperchannel

In both, HDLC or Clear channel modes, data may be received or transmitted over an **hyperchannel** that is made of the concatenation of adjacent, or non-adjacent, time slots as shown on fig. 7, where the CEPT frame is represented.

Figure 7. 384 kbps hyperchannel.


CEPT FRAME. 32 TIME SLOTS (8 BITS 2.048 kbps)
Typical data channel (8 bits, 64 kbps)
Signaling channel
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 S 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
384 kbps hyperchannel

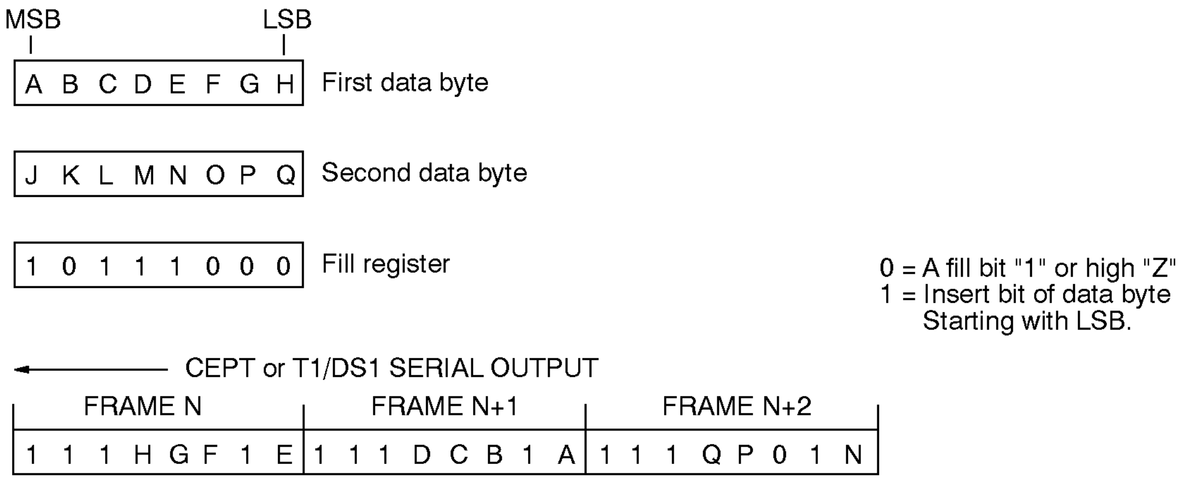
The example illustrates a 384 kbps hyperchannel made of six adjacent ***and*** non-adjacent time slots within the CEPT frame. This results in a PCM frame with twenty seven channels. Twenty six of them are single 64 kbps channels and the last is the 384 kbps channel. Up to 8 time slots can be concatenated to form a 512 kbps hyperchannel.

A time slot is recognized as part of an hyper channel when its "channel number" is different from its time slot number. The PCM counters (Transmit/receive) keep track of the time slot number while the channel number is user defined and stored into the related ***RMOD/XMOD*** register in the HDLC descriptor part of the time slot context.

Rate Adaptation (See **RSPEED** and **XSPEED** registers)

Receive and/or transmit data rate is adaptable on every channel. Basic 64 Kbps rate may be adapted to subrates that are 8 kbps multiples (8, 16, 24, ..). Fill/mask registers are applied to shift registers that data is passing through as illustrated in fig. 8a and 8b.

Figure 8a. Transmit Rate Adaptation : Single transmit channel 24 kbps.

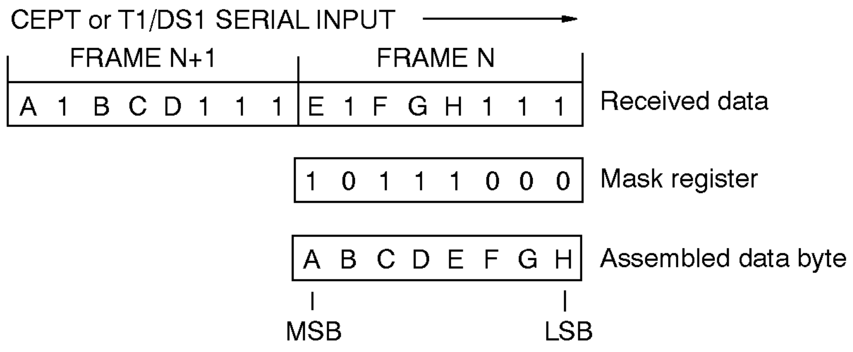


A bit set to "0" in the ***XSPEED*** register forces DX output to "1" or "High Z", according to DX driving mode. A "1" in the fill register let the data bit being output.

A bit set to "0" into the mask register, prevents the corresponding received data bit to be latched into the 29C948. Only when a full byte has been assembled, it will be transfered towards the buffer.

In high impedance driving mode, up to eight 29C948 DX outputs may be connected in parallel. Mutually exclusive fill registers could then be used to make subrate time division multiplexing over the 64 kbps channel.

Figure 8b. Receive Data : Single receive channel 24 kbps.



2.3. DMA Controller

The DMA CONTROLLER is mostly driven and controlled by the DMA descriptor, part of the time slot context control registers. It manages in and out data transfers between the HDLC PROCESSOR and buffers in the system memory. Data fetching and storage are achieved over the host CPU bus through a direct memory access cycle (DMA).

The DMA cycle takes place, when requested, inside the time slot period.

Bus contention is monitored by the 29C948 BUS MANAGER logic block which checks that the bus is available before allowing buffers access to the DMA CONTROLLER (see chapter 2.4 DMA CYCLE). DMA request is done each time a new data byte has to be, either transmitted or stored. Finally the DMA CONTROLLER checks that, before the end of the time slot, the DTACK signal (pin#33) is asserted, pointing out that the DMA cycle is complete. When it is not so, a time out is generated and the cycle aborted.

DMA cycle flow charts are shown in figure 9 and 10.

Each channel is allocated eight transmit buffers and eight receive ones. Channel Buffer addresses are maintained in the DMA descriptor that is part of the context (time slot context mapping in tables 8 and 9).

Initialized by the host CPU, the eight DMA descriptors contain :

– Word count/Data frame length.	16 bits
– Buffer address pointer	16 bits
– DMA status	8 bits
– Buffer page address.	8 bits

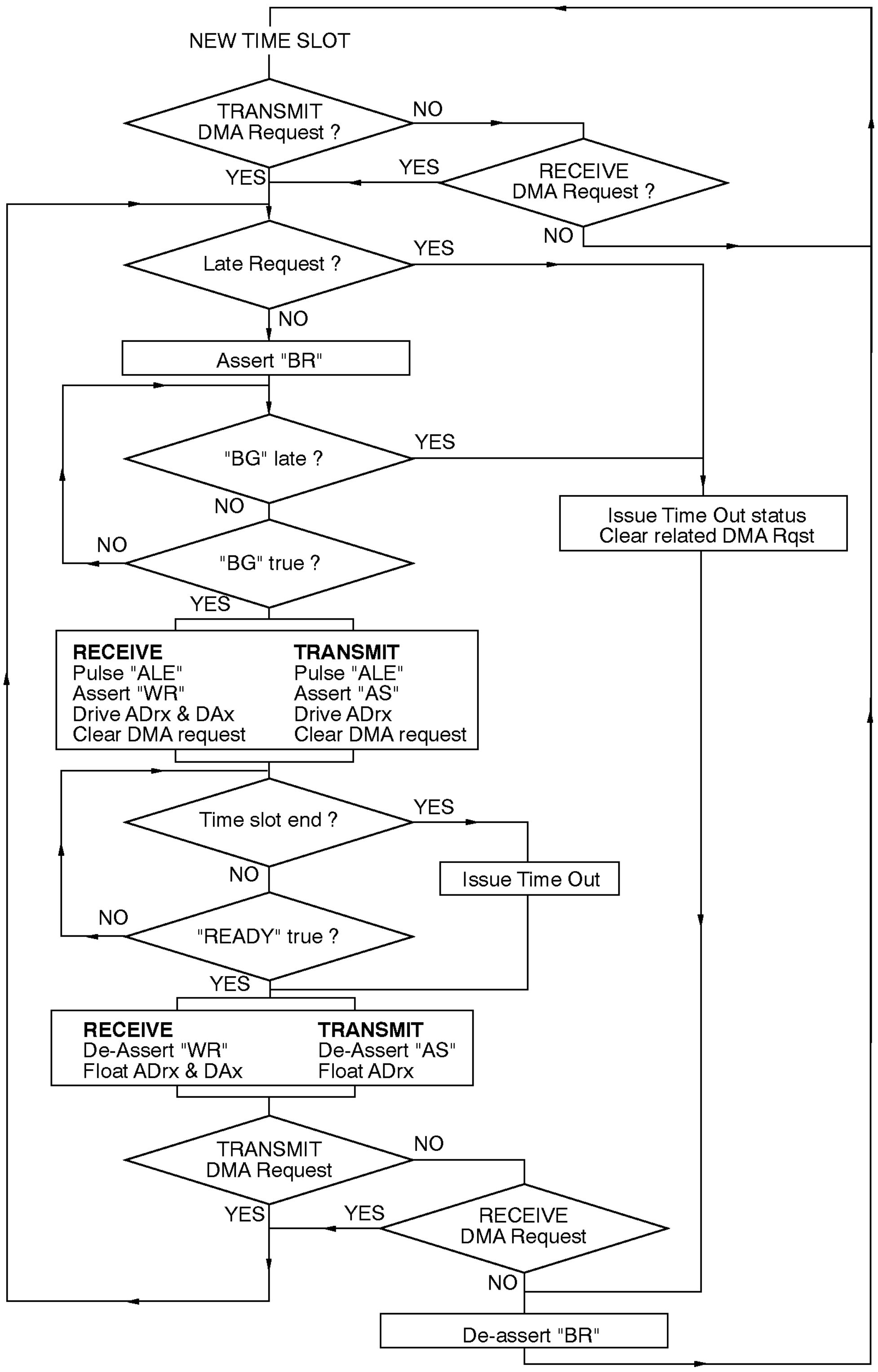
All the transmit/receive descriptors may be initialized in advance, so as to be ready when the transmit/receive activity is starting.

Transmit :

Each time a DMA transfer is successfully completed the word count is decremented and current address (pointer) incremented until that either the buffer is exhausted (word count=0) or a time out occurred. In any event, a status report is assembled then stored in ***XCTST*** register and the buffer index in ***XSTAT1*** register is incremented, modulo 8, to point to a new buffer. Each time there is a switch over in buffer index, an interrupt request is asserted through the set up to “1” of the ITX(x) bit in the ***XINTRQ*** global registers, with (x) being the channel number (see chapter 3.5).

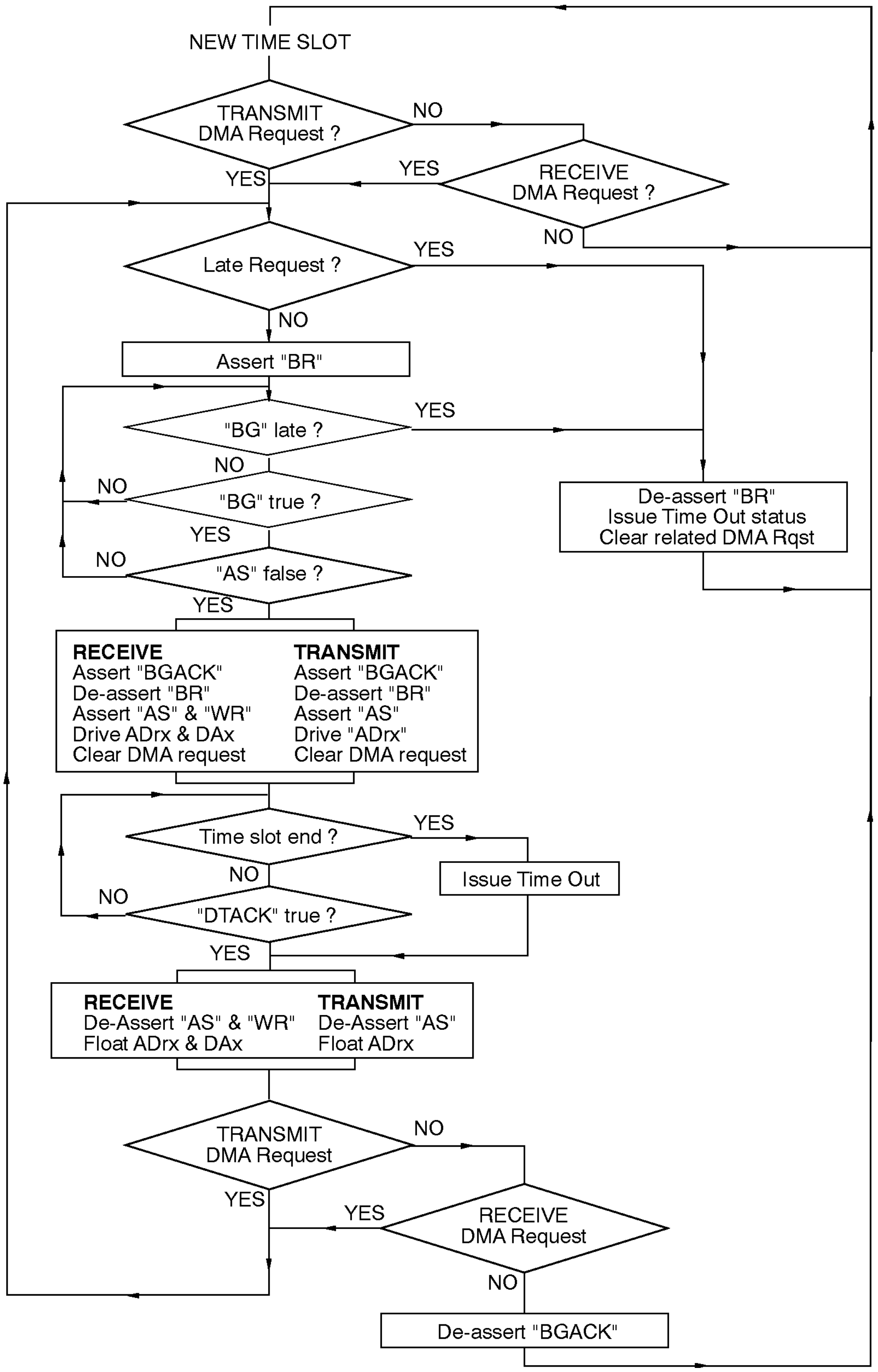
Upon interrupt request the CPU reads into the 29C948 ***INTRQ*** registers, memorizes all interrupt requests, resets the registers and re-initializes the buffers that triggered the interrupt.

Figure 9. DMA cycle with INTEL host CPU.



Note : Transmit and/or receive DMA requests rise, if they are present, when time slot begins.

Figure 10. DMA cycle with MOTOROLA host CPU.



Note : Transmit and/or receive DMA requests rise, if they are present, when time slot begins.

As soon as the system bus is granted, the transmit DMA cycle starts with AD (23:0) address output on system bus and AS/ALE/WR signals being asserted. The cycle ends with a DTACK (see DMA flow chart on figures 9 and 10).

Receive :

The whole process for DMA transfer of received data is very much like the transmit one, except that buffer status and buffer indexes are respectively reported in receive channel RCTST and RSTAT1 registers. In the same way, buffer switching is reported in ITR(x) thus triggering an interrupt request, which is CPU processed as explained above.

The receive DMA cycle starts with address, AD (23:0), and data, DA (7:0), output on system bus, and AS/ALE/WR signals being asserted. The cycle normally ends with DTACK (write acknowledge) asserted before the end of time slot.

In case of overflow (WC=0 and frame not finished) or in case of loss of sync (SYNC = 0) the current buffer is automatically closed and the 29C948 looks up for next opening flag.

If a time out occurs, the current buffer is closed, the buffer index incremented but the incoming data stream is ignored until detection of a new flag.

If the buffer index points to a DMA descriptor that has an exhausted buffer (**DONE** not reset), then writing into the buffer will not be allowed nor in the ***RCTST*** register.

If all DMA buffers are exhausted, current buffer incrementation is stopped until buffers are re-initialized by CPU. Then flags are forced on transmit side and the receiver ignores incoming data stream.

2.4. Bus Manager

The BUS MANAGER interfaces the 29C948 internal logic blocks with the system bus (see fig. 1). **MOT** input (pin#50) selects either INTEL or MOTOROLA interface types. The BUS MANAGER responds to commands originating from the DMA CONTROLLER (DMA request), or from the interrupt registers, or from the host CPU. The BUS MANAGER performs system bus take over for DMA cycles. Also, upon buffers switching, it generates interrupt requests that call for CPU actions. System bus take-over is accomplished after a **bus request/bus grant** hand-shake with the host CPU (BR pin#34/BG pin#52), authorizing the BUS MANAGER to drive the system bus. Timing diagrams are shown on chapter 6.1 to 6.9 for an INTEL CPU and chapter 7.1 to 7.9 for a MOTOROLA CPU.

The 29C948 data bus interface, D_0 through D_7, needs to be only 8-bit wide due to the low demand in data exchange. The interface address bus is 24-bit wide, AD_0 through AD_{23}, allowing 128 buffers of 64 kbits each. Only bit AD_0 through AD_{12} are used when the CPU accesses the 29C948 internal RAM/registers. The 29C948 automatically generates the acknowledgement, DTACK (pin#33), (READY in INTEL mode), when its RAM/registers are being accessed. It also waits for DTACK (READY in INTEL lode) acknowledgement to complete a DMA cycle.

CPU Access

The CPU accesses the global registers (Read/write) within three XTAL1 clock cycles maximum. Context registers access needs six XTAL1 periods maximum. Timing diagrams are shown on chapters 6.3 and 6.4 for INTEL CPU and chapters 7.3 and 7.4 for MOTOROLA.

The internal RAM looks like a dual port memory, which access is managed by the CONTEXT MONITOR. One port, 8-bit wide, is the path for the CPU through the BUS MANAGER, the other port, 16-bit wide, is the path to the functional logic blocks (see fig. 1 – 29C948 block diagram).

Because of delays to clock in and out of the RAM the context data (128 bytes to save/retrieve), compounded by the RAM own access time, the XTAL1 clock frequency cannot go lower than a limit fixed by the conditions listed below – worst case CEPT mode, shortest 3.9 µs time slot :

1. – $3.9 > (2/F) \times (44+N)$
2. – $3.9 > (1/F) \times 5N$
3. – $3.9 > ((2/F) \times N)*3$
4. – $3.9 > (1/FCPU) \times I \times N$

Where	N =	number of CPU access to the internal RAM,
	F =	XTAL1 frequency in MHz,
	I =	maximum number of CPU clock periods for one R/W access.

1. Is the condition for a full transmit/receive context exchange.
2. Reflects that the CPU read or write cycle into context registers, takes a minimum of five XTAL1 periods.
3. Is related to the design of the CONTEXT MONITOR that allows one CPU access every two RAM accesses.
4. Relates to the CPU own read/write limitations.

With the recommended XTAL1 clock frequency of F = 33 MHz, in CEPT mode, all conditions are true. Then 22 R/W accesses may be completed on the CPU port, thus giving it a bandwith of 5.64 Mbytes/s. That throughput allows the host to update DMA descriptors of a given channel while the DMA CONTROLLER is using part of the descriptor. The conditions also apply to T1/DS1 mode, except that the time slot lasts 5.2 μs instead of 3.9 μs.

DMA Cycle

Upon DMA request, the BUS MANAGER issues a bus request, BR, and waits for BG, the bus grant signal. Additionally, in MOTOROLA mode, a bus grant acknowledge (BGACK) is issued by the 29C948.

In MOTOROLA mode BR/BG handshake begins the DMA cycle, and the BGACK output is kept true until the end of the cycle. In INTEL mode BR is kept true until the end of the cycle.

Over the whole cycle, the 29C948 controls the system bus. It drives the 24 bit address, the $\overline{AS}$ (READ), $\overline{WR}$, ALE (in INTEL mode), strobes, and, for data write, the DATA lines. It checks for DTACK (READY).

$\overline{AS}$ (READ) $\overline{WR}$ and ALE (in INTEL mode) are de-asserted ($\overline{AS}$=$\overline{WR}$=1, ALE=0) at the beginning of the cycle, and de-asserted again by the end of the cycle.

Such feature simplifies the application design when selecting PULL-UP and PULL-DOWN resistors.

DMA timing considerations

At the beginning of a time slot, the BUS MANAGER scans for transmit and/or receive DMA request. If at least one is true, the DMA cycle starts. Therefore, on a given time slot, there may be :

– No DMA or,
– either one read or one write cycle (either transmit or receive), or
– two cycles, one read and one write.

The BUS MANAGER checks that DMA cycle completion has, or may occur. According to the conditions, it will take the following actions :

– De-assert BR, bus request, if BG, bus grant, is not yet true by the end of time slot,
– abort the read or write cycle, de-assert BR and release the system bus, if memory acknowledge is not true before time slot end,
– abort the DMA cycle, if BG is asserted close to the end of the time slot, which would make a read or write cycle impossible.

In all cases, a time out status is reported causing a data frame abort on transmit side or a frame error on the receive side. The longest period allowed to achieve a DMA cycle is DELMAX and is worth :

DELMAX=(Time slot period) – (0.5 bit-clock periods) – (4 XTAL1 periods)

If XTAL1 frequency is the recommended 33 MHz, then :

DELMAX = 3.5 μs in CEPT mode or 4.6 μs in T1/DS1 mode

Deadlock cannot occur because the circuit automatically recovers from unsuccessfull DMA attempt, such as trying to write or read into unexisting buffer addresses. When one read and one write cycle are requested on a given time slot, the BUS MANAGER does not de-assert BR after completion of the first cycle to save bus grant latency time.

2.5. Interrupt Controller

The interrupt controller generates an interrupt request each time a DMA buffer is closed (end of frame with normal or error status). The CPU identifies the interrupting channel by reading the eight 8-bit interrupt registers (see ***INTRQ*** registers on chapter 3.5). Purpose of the interrupt request is to re-initialize exhausted buffers by setting up the related DMA descriptors.

When a bit is set in one of the ***INTRQ*** register, the related ***RCTST*** and ***XCTST*** channel status registers shall be read. The ***INTRQ*** registers are read only type. Reading one register will automatically reset all 8 bits. However, no interrupt can be missed, because the setting of a bit arriving during a CPU read cycle is delayed and therefore, the register reset after read will not apply to this bit.

The INTERRUPT REQUEST output (pin#67) will be true if at least one bit of any one of the eight ***INTRQ*** is set. A flow chart of the interrupt routine is presented on figure 11.

3.0 Register Definitions

All the control registers data are stored into the 29C948 internal RAM, which mapping is shown on table 3. As previously described in FUNCTIONAL DESCRIPTION, starting on chapter 1.0, one can distinguished two kinds of control registers ; Global registers and context registers.

Organization and definition

3.1. Global Registers

Control of global operating mode requires fourteen 8-bit registers. Only six of them, on addresses 1000_H-1001_H-1002_H-1003_H-$100C_H$-$100D_H$, are eventually used to specify operating mode. The other eight are split into two stacks of four 8-bit registers, on addresses 1004_H-1005_H-1006_H-1007_H and 1008_H-1009_H-$100A_H$-$100B_H$, and are used to assert interrupt requests upon buffers switching. Register names and addresses, register status after reset and bit mnemonics are shown below, table 4.

Table 4 : Global Operating Mode Control Registers.

Register		MNEMONICS								Reset State
Name	Addr.	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	
MODE	1000_H	LB	DXC1	DXC0	DPR8	DPX8	DC	T1	MSTR	00_H
ROFFSET	1001_H	DPR7	DPR6	DPR5	DPR4	DPR3	DPR2	DPR1	DPR0	00_H
XOFFSET	1002_H	DPX7	DPX6	DPX5	DPX4	DPX3	DPX2	DPX1	DPX0	00_H
GLOBINH	1003_H	NA	NA	NA	NA	NA	NA	NA	GINH	00_H
XINTRQ	1004_H	ITX7	ITX6	ITX5	ITX4	ITX3	ITX2	ITX1	ITX0	00_H
	1005_H	ITX15	ITX14	ITX13	ITX12	ITX11	ITX10	ITX9	ITX8	00_H
	1006_H	ITX23	ITX22	ITX21	ITX20	ITX19	ITX18	ITX17	ITX16	00_H
	1007_H	ITX31	ITX30	ITX29	ITX28	ITX27	ITX26	ITX25	ITX24	00_H
RINTRQ	1008_H	ITR7	ITR6	ITR5	ITR4	ITR3	ITR2	ITR1	ITR0	00_H
	1009_H	ITR15	ITR14	ITR13	ITR12	ITR11	ITR10	ITR9	ITR8	00_H
	$100A_H$	ITR23	ITR22	ITR21	ITR20	ITR19	ITR18	ITR17	ITR16	00_H
	$100B_H$	ITR31	ITR30	ITR29	ITR28	ITR27	ITR26	ITR25	ITR24	00_H
TESTREG0	$100C_H$	TST7	TST6	TST5	TST4	TST3	TST2	TST1	TST0	FF_H
TESTREG1	$100D_H$	NA	NA	NA	NA	NA	NA	NA	TST8	01_H

Read/Write into Global Registers

CPU intervention, read or write, into the global registers needs three XTAL1 periods. To either read or write into the registers, the host CPU drives a valid address on the system bus (AD 12;0), with CS = 0 (Chip Select). Data bytes are transferred to and from the 29C948 under the host CPU control according the truth table of table 5. Both write or read cycles end with a 29C948 acknowledgement that asserts DTACK=0 in MOTOROLA mode or DTACK=1 in INTEL mode.

Timing diagrams of the read and write operations are shown on chapters 6.3/6.4 (INTEL) and 7.3/7.4 (MOTOROLA).

Table 5 : Write/Read operations on Global Control Registers.

Signal Active Levels		MOTOROLA Mode		INTEL Mode	
		Write	Read	Write	Read
MOTOROLA/INTEL MODE(1)	MOT	Vcc	Vcc	Gnd	Gnd
ADDRESS LATCH ENABLE STROBE(2)	ALE	Gnd	Gnd	1	1
CHIP SELECT.	CS	0	0	0	0
ADDRESS STROBE(3)	AS	0	0	1	0
DATA STROBE	DS	0	0	NA	NA
WRITE STROBE	WS	0	1	0	1

Notes : 1. The MOT pin (#50) must be definitely tied to VCC in MOTOROLA mode and to ground in INTEL mode.
2. The ALE pin (#29) should be tied to ground in MOTOROLA mode.
3. The $\overline{\text{DS}}$ pin should be tied to Vcc in INTEL mode.

Global Register Descriptions & Bit Definitions

3.2. MODE Register - Address 1000_H -

The register specifies the global operating mode of the 29C948 according bit control listed in the table 6.

Table 6 : Bit Functions in MODE Register.

Bit Mnemo	Bit level "1"	Bit level "0"
MSTR	Master mode	Slave mode
T1	T1/DS1 operating mode	CEPT operating mode
DC	Double clock	Simple clock
DPX8	Transmit offset extension bit	
DPR8	Receive offset extension bit	
DXC0 DXC1	DX output (pin # 55) driving mode programming	
	DXC1 / DXC0 / DX output: 0 0 High impedance; 0 1 Always driving; 1 0 Open collector; 1 1 Not used	
LB	Loop back, for test purpose	Normal operating mode

DXC1	DXC0	DX output
0	0	High impedance
0	1	Always driving
1	0	Open collector
1	1	Not used

When loop back mode is selected all transmit channels are looped back to their corresponding receive channels through the PCM HANDLER. In that operating mode the receive and transmit offset programming as well as byte alignment procedure must be as explained in the FRAME SYNC OFFSET chapter starting on chapter 2.1. Special attention should be devoted to,

1 - Set-up the DMA descriptors for both transmit and receive channels on every single or hyper channel.

2 - Program ***RSPEED/XSPEED*** registers of related channels with the same value.

3 - Program ***RMOD*** register, the channel number and **ITVR** bit, on every single and hyperchannel, ending with hyperchannel head as explained on chapter 3.7.

4 - Wait for all **ITVR** being set to “1” on every single and hyperchannel.

5 - Program ***XMOD*** register, the channel number and **ITVR** bit, on every single and hyperchannel, ending with hyperchannel head.

After completion of this procedure, the device is fully ready for transmit/receive loop.

3.3. OFFSET Registers – ROFFSET at 1001$_H$, XOFFSET at 1002$_H$ – TEST REGISTERS – TESTREGO at 100CH, TESTREG1 at 100DH –

OFFSET REGISTERS and TEST REGISTERS are working in conjunction. Binary count is used. The ***TESTREG*** specifies the number of time slot (one time slot=eight bit clock) to be used in the selected operating mode (24 T1/DS1, 32 CEPT). Up to 8 time slots can be used in T1/DS1 and CEPT modes.

Provided that the 9-bit word TST (8:0) contained in ***TESTREGO*** and ***TESTREG1*** registers is as shown in Table 7, then the PCM counters will count accordingly. Otherwise, the PCM counters will count by TST (8:0) + 1.

Table 7 : PCM Counters and Test Registers programming.

MODE	**Binary Value OF TST (8;0)**		**PCM Counters Decimal Count**	
	Single Clk	**Double Clk**	**Single Clk**	**Double Clk**
CEPT	011111111	111111111	256	512
T1/DS1	011000000	111000001	193	386

Master mode : The 29C948 generates the FSX/FSR frame sync at 8 kHz by using the XCLK and RCLK bit clocks.

The width of FSX/FSR sync pulses is one XCLK/RCLK period. Pulses are triggered when the PCM bit counters reach the value programmed in the offset registers (DPR8 and DPX8 bits are only used in double clock mode).

The 29C948 includes two PCM counters, one for transmitting and one for receiving. The counters are incremented by XCLK (transmit) and RCLK (receive) and thus keep track of time slots and PCM frame sequence. The DPR8/DPX8 extension bit allows to count up to 512, as necessary in double clock mode with a CEPT frame (for more information see technical note).

Slave mode : The FSR/FSX frame syncs are supplied to the 29C948. They are used to preset the PCM counters with the content of the offset registers. Counters presetting takes place *on falling edge of XCLK/RCLK with FSX/FSR high*. The frame sync pulse must be one bit clock period wide.

Double clock : This mode relates to GCI standard. For both XCLK and RCLK, two bit clock are used per single bit frame.

3.4. GLOBINH Register – Address 1003$_H$ –

Applying a RESET to the 29C948 (see table 1 RESET logic levels) results in a global inhibit of the device.

After reset the HDLC PROCESSOR is idle and DX output is in high impedance state. The global control registers status is shown on table 4 immediately after reset. However, reset applies only to the global registers and not to the context ones. Caution must be exercized for a proper initialisation of the whole device.

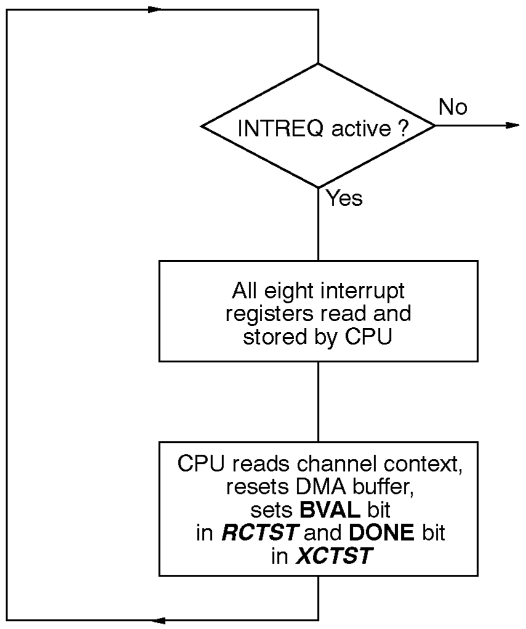
First, it is necessary to let GINH (bit 0 of GLOBINH register) be at low level for a least 2 full PCM frames with valid XCLK and RCLK. It is also mandatory to have the channels control registers set up by the host CPU before letting GINH be at logical "1". Not following this procedure would induce 29C948 erratic behaviour.

3.5. XINTRQ – Address 1004$_H$ thru 1007$_H$ – RINTRQ – Address 1008$_H$ thru 100B$_H$ –

These two sets of registers are used to signal that a data frame completion and therefore a buffer switching, has occurred on a receive or transmit channel. Upon data frame completion, a bit is automatically set to "1" in the related register, at the location that points to the channel number.

Any bit at "1" in the set of registers asserts the INTREQ output high (pin # 67), thus calling for the host CPU to read and memorize all eigth interrupt registers. The host will then read the status registers of the interrupting channel(s). The **XINTRQ/RINTRQ** registers are reset to zero when read by the cpu. Fig. 11 illustrates an interrupt routine.

Figure 11. Interrupt Routine.

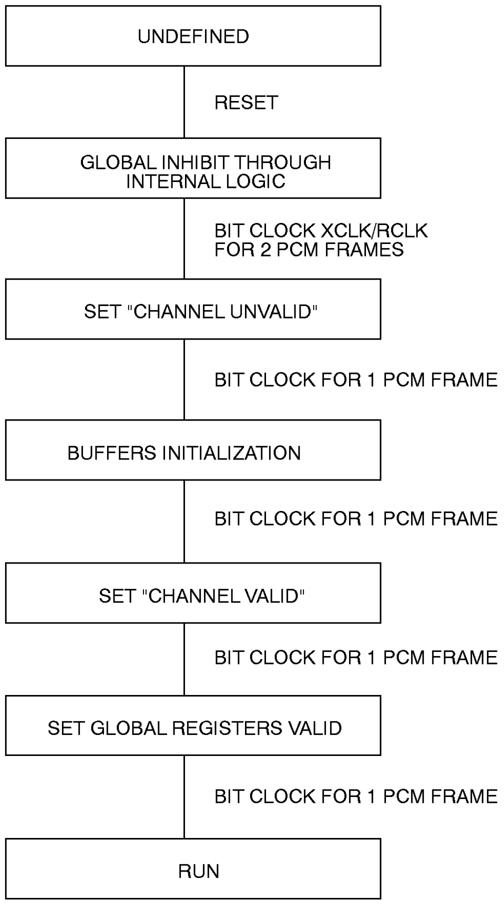

INTREQ active ?
No
Yes
All eight interrupt registers read and stored by CPU
CPU reads channel context, resets DMA buffer, sets **BVAL** bit in ***RCTST*** and **DONE** bit in ***XCTST***

Initialization

Upon power-up the 29C948 must be resetted. This is performed by asserting the RESET signal for, at least, two PCM frame periods with valid XCLK and RCLK bit clock. All global registers are set in reset values as indicated in table 4. The 29C948 is then in slave, single clock CEPT operating mode. The whole chip is inhibited, and frame offset is zero. However, all the context registers are still in undefined states and should be set up by the host CPU as shown in the following flow chart. Literate explanations of the procedure may be found on chapter 3.7 where an hyperchannel setting procedure is described.

A diagram of an initialization procedure example in Figure 12.

Figure 12. Exemple of initialization procedure.


UNDEFINED
RESET
GLOBAL INHIBIT THROUGH INTERNAL LOGIC
BIT CLOCK XCLK/RCLK FOR 2 PCM FRAMES
SET "CHANNEL UNVALID"
BIT CLOCK FOR 1 PCM FRAME
BUFFERS INITIALIZATION
BIT CLOCK FOR 1 PCM FRAME
SET "CHANNEL VALID"
BIT CLOCK FOR 1 PCM FRAME
SET GLOBAL REGISTERS VALID
BIT CLOCK FOR 1 PCM FRAME
RUN

3.6. Channel Context Register

Each PCM frame time slot has its own set of context registers. The registers are stacked into a double port RAM. An 8-bit wide port is used for CPU communication (Read/Write operations). The other port is 16-bit wide and is used in the successive operations of context restore-execute-save.

The 8-bit wide port to the outside world is sufficient because performances of the 29C948 is widely self-supported, CPU intervention is not frequent. The 16-bit port is the gateway to the 29C948 internal bus. It needs to be wider because there are 128 bytes of context data to switch time slot after time slot.

The 8 active receive contexts are stored on decimal addresses 0H through 7FFH, while the transmit contexts are stored from 800H through FFFH (see table 3, internal RAM mapping).

Context Registers Description & Bit Definition

Each time slot is associated with a set of receive and transmit context registers that define reciprocally the receive and transmit channel. Tables 8 and 9 thereafter, describe the details of receive and transmit context for time slot 0. These sets of registers are repeated for all 32 time slots as previously explained, that is to say a total of 64 single channels (32 receive + 32 transmit). But only 8 channels can be active simultaneously.

It should be noticed that the organization of the channel context is split in two parts. The first part, referred as the HDLC descriptor, is related to transmit/receive data processing for that time slot. It contains various operating parameters and data which purposes are defined further on.
The other part of the context data is called the DMA descriptor ; It contains word count, buffer address and DMA status. That second part of the channel context is replicated eight times, once per buffer, since for each channel is allocated eight chained buffers.

Table 8 : Time Slot 0, Receive Context Registers.

	MNEMONICS								Context Register	
	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	Name	address
HDLC DESCRIPTOR	Channel Number (bit “log4” thru “log0”)					N.U[1]	XPRT	ITV	RMOD	0000_H
	Rate adaptor – Fill Mask. (bit “RSP_7” thru “RSP_0”)								RSPEED	0001_H
	Current Buffer number			FTR	DMAD	Data Frame Status			RSTAT1	0002_H
	ITac	N.U[1]	N.U(1)	State Mac		Bit count			RSTAT2	0003_H
	Shift Register 3								Shift 3	0004_H
	Shift Register 2								Shift 2	0005_H
	Shift Register 1								Shift 1	0006_H
	DATA								FIFO	0007_H
	Frame check sequence – low byte –								CRCL	0008_H
	Frame check sequence – high byte –								CRCH	0009_H
DMA Descriptor										
DMA DESCRIPTOR	Buffer 0. – Word count – low byte –								WCL_0	$000A_H$
	Buffer 0. – Word count – high byte –								WCH_0	$000B_H$
	Buffer 0. – Current Address pointer – low byte –								$ACTL_0$	$000C_H$
	Buffer 0. Current Address pointer – high byte –								$ACTH_0$	$000D_H$
	Lsyn	Ftr	Tout	Udf	Nba	Abrt	CRCe	Done	RCTST	$000E_H$
	Buffer 0.			Page Address					$SEGMT_0$	$000F_H$
	Buffer 1. Word count ....., ... Address pointer ....., Buffer 1. ..... Page, Status									0010_H thru 0015_H
	Buffer 2. Word count ....., ... Address pointer ....., Buffer 2. ..... Page, Status									0016_H thru $001B_H$
	Buffer 3. Word count ....., ... Address pointer ....., Buffer 3. ..... Page									$001C_H$ thru 0021_H
	Buffer 4. Word count ....., ... Address pointer ....., Buffer 4. ..... Page, Status									0022_H thru 0027_H
	Buffer 5. Word count ....., ... Address pointer ....., Buffer 5. ..... Page, Status									0028_H thru $002D_H$
	Buffer 6. Word count ....., ... Address pointer ....., Buffer 6. ..... Page, Status									$002E_H$ thru 0033_H
	Buffer 7. Word count ....., ... Address pointer ....., Buffer 7. ..... Page, Status									0034_H thru 0039_H

Note : 1. N.U = Not Used.

Table 9 : Time Slot 0, Receive Context Registers.

	MNEMONICS								Context Register	
	bit 7	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**	**Name**	**address**
HDLC DESCRIPTOR	Channel Number (bit “log4” thru “log0”)					FLG	XPRT	ITVX	XMOD	0800_H
	Rate adaptor – Fill Mask. (bit “XSP7” thru “RSP$_0$”)								XSPEED	0801_H
	Current Buffer number			FTR	DMAD	Data Frame Status			XSTAT1	0802_H
	ITac	N.U(1)	Transmit State Mac			Bit count			XSTAT2	0803_H
	Shift Register								Shift 1	0804_H
	N.U(1)	FIFO CONTROL			TAG 2		TAG 1		TAG	0805_H
	FIFO 1								FIFO1	0806_H
	FIFO 2								FIFO2	0807_H
	Frame check sequence – low byte –								CRCL	0808_H
	Frame check sequence – high byte –								CRCH	0809_H
Transmit DMA Descriptor										
DMA DESCRIPTOR	Buffer 0. – Word count – low byte –								WCL$_0$	$080A_H$
	Buffer 0. – Word count – high byte –								WCH$_0$	$080B_H$
	Buffer 0. – Current Address pointer – low byte –								ACTL$_0$	$080C_H$
	Buffer 0. Current Address pointer – high byte –								ACTH$_0$	$080D_H$
	N.U(1)	N.U(1)	N.U(1)	N.U(1)	N.U(1)	TOUT	EOF	BVAL	XCTST	$080E_H$
	Buffer 0.			Page Address					SEGMT$_0$	$080F_H$
	Buffer 1. Word count ....., ... Address pointer ....., Buffer 1. ..... Page, Status									0810_H thru 0815_H
	Buffer 2. Word count ....., ... Address pointer ....., Buffer 2. ..... Page, Status									0816_H thru $081B_H$
	Buffer 3. Word count ....., ... Address pointer ....., Buffer 3. ..... Page, Status									$081C_H$ thru 0821_H
	Buffer 4. Word count ....., ... Address pointer ....., Buffer 4. ..... Page, Status									0822_H thru 0827_H
	Buffer 5. Word count ....., ... Address pointer ....., Buffer 5. ..... Page, Status									0828_H thru $082D_H$
	Buffer 6. Word count ....., ... Address pointer ....., Buffer 6. ..... Page, Status									$082E_H$ thru 0833_H
	Buffer 7. Word count ....., ... Address pointer ....., Buffer 7. ..... Page, Status									0834_H thru 0839_H

Note : 1. N.U = Not Used.

DLC Descriptor

Only the R*MOD/XMOD* and ***RSPEED/XSPEED*** registers are used in programming control of the HDLC PROCESSOR logic block. All other HDLC registers are automatically maintained by the internal logic of the 29C948. Writing into them would induce erratic behaviour of the device.

3.7. RMOD REGISTERS : Add : 000_H + $N^{(1)} \times (40_H)$. XMOD REGISTERS : Add :800_H + $N^{(1)} \times (40_H)$.

The registers control the HDLC PROCESSOR logic according the truth tables below.

Receive Channel

Table 10 : XMOD Register Truth Table.

Bit MNEMO	Bit level	
	”1”	**”0”**
ITVR	Channel Enable	Channel disable
XPRT	Clear Channel mode	HDLC format
log0 thru log4	Binary Channel Number	

Transmit Channel

Table 11 : RMOD Register Truth Table.

Bit MNEMO	Bit level	
	”1”	**”0”**
ITVX	Channel Enable	Channel disable
XPRT	Clear Channel mode	HDLC format
FLG	Flag sharing mode	Flag not shared
log0 thru log4	Binary Channel Number	

Hyperchannel

The $\log_0$ through $\log_4$ bits specify the channel number.

– When this binary channel number is equal to the current time slot number maintained in the PCM counter, then the current time slot is a single channel or the head of an hyperchannel.

– When the channel number ($\log_0$ through $\log_4$ bits) is not equal to the current time slot number, then that time slot is part of an hyper channel and is concatenated with the time slot defined by $\log_0$ through $\log_4$.

Notes : 1. N equal time slot number,

Particular attention should be paid to the fact that all time slots pertaining to an hyperchannel must have identical ***RMOD*** and ***RSPEED*** register contents on the receive channel as well as identical ***XMOD*** and ***XSPEED*** contents on the transmit side.

The DMA descriptor of an hyperchannel head is shared by all time slots concatenated to the channel head.

Procedure to set-up a receive hyperchannel is :

1. Reset to 0 the ITVR bit of all time slots to be concatenated in the hyperchannel.
2. Set-up the DMA descriptor of hyperchannel head.
3. Successively, **ending with the hyperchannel head,** write identical content in ***RMOD*** and ***RSPEED*** registers of the related time slots, setting ITVR = 1.

The same procedure applies to set a transmit hyperchannel, except that the involved registers are ***XMOD*** and ***XSPEED***.

Conversely, the procedure to de-activate a receive hyperchannel is :

1. Reset to 0 the ITVR bit of hyperchannel head.
2. Wait for ITVR being copied into ITac of RSTAT register of the hyperchannel head.
3. Update channels configuration.

Again, identical procedure applies for a transmit hyperchannel, except that it is the transmit context registers that are involved.

3.8. RSPEED REGISTERS.Add : $001_H + N^{(1)} \times (40_H)$. XSPEED REGISTERS.Add : $801_H + N^{(1)} \times (40_H)$.

Those are the fill/mask registers used for rate adaptation as explained in fig 8A and 8B.

Other HDLC Descriptor registers

All other HDLC registers may be read, but should not be written into, which would cause unpredictable operations. Of particular importance is the **ITac** bit (bit 7) of, respectively, the ***RSTA2*** and ***XSTAT2*** registers. Each time a time slot context is restored, the time slot enable bit (ITVR/ITVX) is copied into ITac. Precisely, ITVR is copied into ITac of ***RSTAT2*** and ITVX is copied into ITac of ***XSTAT2***, with exception for hyperchannel. ITac may be regarded as the acknowledge bit of the time slot enable. A safe procedure to change a channel operating mode is shown in table 12.

The following registers are described for information only and users should not temper with them.

Table 12 : Operating Mode Modification Procedure.

	Transmit	Receive
1	Reset ITVX to 0	Reset ITVR to 0
2	Wait for ITac = 0	Wait for ITac = 0
3	Modify as required XMOD, XSPEED, DMA descriptors,	Modify as required RMOD, XSPEED, DMA descriptors
4	Set ITVX to 1.	Set ITVX to 1.

Notes : 1. N equal time slot number, max. 8 active timeslots.

3.9. RSTAT1 REGISTERS.Add : $002_H + N^{(1)} \times (40_H)$. XSTAT1 REGISTERS.Add : $802_H + N^{(1)} \times (40_H)$.

Table 13 : RSTAT1/XSTAT1 bit definitions

Bit	MNEMONIC	Receive Functions		Transmit Functions	
		Frame Status + Data	Result	Frame Status	Result
0 1 2	Data Frame status	101 (5_H) 0 110 (6_H) 0 110 (6_H) 1	Delete Zero Flag Abort	101 (5_H)	Insert one ”ZERO” if currently transmitting data or CRC.
3	DMAD	1 = DMA request upon next occurence of logical channel.		1 = DMA request upon next occurence of logical channel	
4	FTR	1 = End of frame on next occurence of logical channel.		1 = End of frame on next occurence of logical channel.	
5 6 7	INDEX	Binary number of buffer currently being used.		Binary number of buffer currently being used.	

The **INDEX** bits are automatically reset to zero when a channel of hyperchannel is disabled. When the channel will be again enabled, the index will point to the first descriptor.

3.10. RSTAT2 REGISTERS.Add : $003_H + N^{(1)} \times (40_H)$. XSTAT2 REGISTERS.Add : $803_H + N^{(1)} \times (40_H)$.

Table 14 : RSTAT2/XSTAT2 Bit Definition

Bit	MNEMONIC	Receive Functions		Bit	MNEMONIC	Transmit Functions	
0 1 2	BitCount	Binary count of received bits on current byte 0 < N < 7		0 1 2	BitCount	Binary count of transmitted bits on current byte 0 < N < 7	
	Stat Mac	Internal logic status			Stat Mac	Internal logic status	
		4 . 3	Status			5.4.3.	Status
3 4		0 . 0 0 . 1 1 . 0 1 . 1	Idle Flag received Data ; DMA not author. Data ; DMA authorized	3 4		0.0.0 0.1.0 0.1.1 1.0.0	Opening flag Data being transmitted Data being transmitted CRC being transmitted
5		Not used		5		1.0.1 1.1.0 0.0.1	CRC being transmitted Closing flag Abort
6		Not used		6		Not used	
7	ITac	Copy/acknowledge of ITVR (Bit 0 of *RMOD* registers)		7	ITac	Copy/acknowledge of ITVR (Bit 0 of *XMOD* registers)	

Notes : 1. N equal time slot number. Max 8 active timeslots.

SHIFT3 REGISTERS. Add : $004_H + N^{(1)} \times (40_H)$.

Receive channel. ***SHIFT3*** contains a received byte that will be later on transferred to the DMA CONTROLLER to be eventually stored into a buffer.

SHIFT REGISTERS. Add : $804_H + N^{(1)} \times (40_H)$.

Transmit channel. Contains a data byte that is being serialized for transmission.

SHIFT2 REGISTERS. Add : $005_H + N^{(1)} \times (40_H)$.

Receive channel. Contains a data byte that will be later moved to ***SHIT3*** and thus eventually stored into a buffer.

TAG REGISTERS. Add : $805_H + N^{(1)} \times (40_H)$.

Transmit channel. ***TAG*** contains three sets of control bits. The first two sets, bit 0 through 3, are referred as tag1 and tag2. They are used as a qualifier for data transfer from the DMA CONTROLLER to HDLC PROCESSOR as shown in table 15. **TAG1** qualifies the byte in ***FIFO1***, and **TAG2** qualifies the byte in ***FIFO2***.

The third set of control bits extends from bit 4 to bit 6. It is referred as FIFO control and performs a continuous data supply into the transmit pipeline.

SHIFT1 REGISTERS. Add : $006_H + N^{(1)} \times (40_H)$.

Receive channel. Contains a data byte that will be later moved to ***SHIT2*** and thus stored eventually into a buffer.

Table 15 : Tag definitions.

TAG	Definitions
00	Unvalid data byte resulting from time out.
01	Valid data
10	Last byte of data frame

FIFO1 REGISTERS. Add : $806_H + N^{(1)} \times (40_H)$.

Transmit channel. One of the slots of the transmit pipeline.

FIFO REGISTERS. Add : $007_H + N^{(1)} \times (40_H)$.

Receive channel. One of the slots of receive pipeline.

FIFO2 REGISTERS. Add : $807_H + N^{(1)} \times (40_H)$.

Transmit channel. One of the slots of the transmit pipeline.

RCRCL REGISTERS. Add : $008_H + N^{(1)} \times (40_H)$.
RCRCH REGISTERS. Add : $009_H + N^{(1)} \times (40_H)$.

Receive channel. Those two registers contain the current CRC computation related to the data frame being received. ***RCRCL*** is CRC low byte, and ***RCRCH*** the high one. They are compared to the CRC that is received at frame end, just before the closing flag.

XCRCL REGISTERS. Add : $808_H + N^{(1)} \times (40_H)$.
XCRCH REGISTERS. Add : $809_H + N^{(1)} \times (40_H)$.

Transmit channel. Those two registers contains the current CRC computation in relation with the data frame being transmitted. ***XCRCL*** is CRC low byte, and ***XCRCH*** the high one. They are transmitted on the ISDN line, MSB first, after the last data byte, just before the closing flag.

3.11. DMA Descriptors

Both transmit and receive DMA descriptors have the same organization, except for the ***RSTAT*** and ***XSTAT*** status registers. Each channel has eigth descriptors as shown on tables 8 and 9, context mapping.

RWCL REGISTERS. Add :
$00A_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.
RWCH REGISTERS. Add :
$00B_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Receive channel. Contain the low and high bytes of the 16-bit word count. This define the buffer capacity. The word count is decremented by one each time a data byte is transferred into the buffer. When word count reaches zero, the buffer index in ***RSTAT1*** is incremented by one.

XWCL REGISTERS. Add :
$80A_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.
XWCH REGISTERS. Add :
$80B_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Transmit channel. Same as ***RWCL*** and ***RWCH***, except that word count is decremented after a data byte was fetched into the buffer.

Notes : 1. N equal time slot number. Max. 8 active timeslots.
2. I equal buffer index number.

RACTL REGISTERS. Add :
$00C_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.
RACTH REGISTERS. Add :
$00D_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Receive channel. Contain the first 16 bits of the 24-bit buffer address. Upon initialization the host CPU sets up the starting address. Then, each time a byte is transferred into the buffer, the address is incremented by one.

XACTL REGISTERS. Add :
$80C_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.
XACTH REGISTERS. Add :
$80D_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Transmit channel. Same as ***RACTL*** and ***RACTH***, except that address is incremented after a data fetching.

RSEGMT REGISTERS. Add :
$00F_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Receive channel. Contains the highest 8 bits of the 24 bit buffer address. This is a fixed number, set by the host CPU.

XSEGMT REGISTERS. Add :
$80F_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Transmit channel. Same as ***RSEGMT.***

RCTST REGISTERS. Add :
$00E_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Receive channel. DMA descriptor status register.

This is the status register of the receive DMA descriptor. **DONE** is a semaphore bit that is cleared to zero by users to point out that word count (***RWCL/RWCH***), and address (***RACTL/RACTH/ RSEGMT***) are valid. Then incoming data stream may be stored into that buffer.

When the buffer index points to a DMA descriptor where **DONE** is already set to “1”, the buffer is not written nor is the ***RCTST*** register. This state is refered as an overload state. However, since the previous buffer closing has triggered an interrupt request, users is warned that the buffers must be re-initialized. To recover from the overload state, users may :

1. Read the **INDEX** in ***RSTAT1***, and re-initialize that descriptor where the DMA CONTROLLER is waiting, or
2.
   - Clear **ITVR** into ***RMOD*** to inhibit receiving data
   - Wait for **ITac** acknowledge (read ***RSTAT2***)
   - Re-initialise the entire set of descriptors
   - Set **ITVR** back to “1”

Notes : 1. N equal time slot number. Max 8 active timeslots.
2. I equal buffer index number.

In procedure 2, the buffer index is reset to zero.

To update a descriptor, it is recommended to successively set ***RWCL/RWCH*** (word count), ***RACTL/ RACTH/RSEGMT*** (address) and to end up with ***RCTST*** programming.

Table 16 : Receive Status, DMA Descriptor.

Bit #	Bit MNEMO	BIT LEVEL "1"	BIT LEVEL "0"
0	DONE	Buffer unvalid	Buffer valid
1	CRCE	CRC error/end of frame	CRC valid/end of frame
2	ABRT	Abort recognized	Valid frame
3	NBA	Bit alignement error	Bit alignment valid
4	UDF	WC = ∅ and closing flag not detected	Frame correct
5	TOUT	Time out	
6	FTR	End of frame	
7	LSYN	Lost of sync	

XCTST REGISTERS. Add : $80E_H + N^{(1)} \times (40_H + I^{(2)} \times 6_H)$.

Transmit channel. DMA descriptor status register.

This is the status register of the transmit DMA descriptor. **BVAL** is a handshake bit that is preset to one by users to point out that word count (***XWCL/XWCH***), and address (***XACTL/XACTH/ XSEGMT***) are valid. When word count reaches zero, the DMA CONTROLLER clears **BVAL** and sets **EOF** to "1". If a time out occures in a DMA cycle, the DMA CONTROLLER also clears **BVAL** and sets **EOF** and **TOUT** to "1". In that case an ABORT code is transmitted when operating in HDLC mode.

If **BVAL** is at level zero, the DMA CONTROLLER waits until it is reset to "1", buffer index is not incremented, flags are transmitted in HDLC mode and "ONES" in clear channel mode.

It is recommanded to update a DMA descriptor by successively programming word count, address and then ***XCTST*** register. It is mandatory to reset this later register to 01H to clear any previous **TOUT** and **EOF**.

Table 17 : Transmit Status, DMA Descriptor.

Bit #	Bit MNEMO	Bit Level "1"	Bit Level "0"
0	BVAL	Buffer valid	Buffer unvalid
1	EOF	End of frame	
2	TOUT	Time out	
3	N.U	Not used	
4	N.U	Not used	
5	N.U	Not used	
6	N.U	Not used	
7	N.U	Not used	

Notes : 1. N equal time slot number, Max 8 active timeslots.
2. I equal buffer index number.

4.0 Electrical Characteristics

Absolute Maximum Ratings

VCC to GND	–0.5V to +7V
Input/Output voltage	–0.3V to VCC + 0.3V
Storage temperature	–65 to 150°C

Operating Conditions

Voltage range (VCC)	4.5 to 5.5V
Temperature range	0 to 70°C

DC Electrical Characteristics

Parameter	Conditions	Min	Max	Unit
Low level input voltage VIL	I= 5uA (except XTAL1)		0.8	Volt
High level input voltage VIH	I= 5uA (except XTAL1)	2.2		Volt
Low level input voltage VIL	I= 5uA (for XTAL1)		1.5	Volt
High level input voltage VIH	I= 5uA (for XTAL1)	3.5		Volt
Input leakage current			10	uA
3 STATE output leakage current			10	uA
Low level output voltage VOL	I= – 6.4mA for AD and DA buses, AS, ALE, WR, DTACK, BR, BGACK, INTREQ		0.4	Volt
	I= – 12.8mA for FSX, FSR DX, INHIBX, CLKOUT			
High level output voltage VOH	I= 16mA for AD and DA buses, AS, ALE, WR, DTACK, BGACK, INTREQ	2.4		Volt
	I=3.2mA for FSX, FSR, DX, INHIBX, CLKOUT			
Standby current	VCC= 5V		200uA	
Operating current	VCC= 5V, 33MHZ clock with internal OSC 150pF load on all output except CLKOUT 50pF		100mA	

BR output is an open collector.

DX output may be configured as 3-state, open collector always driving.

5.0 AC Electrical Characteristics

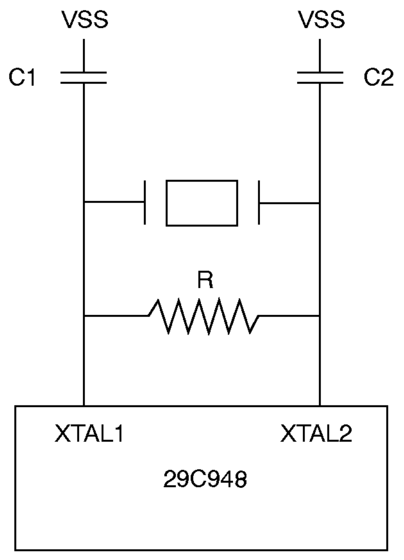
PCM Interface

VCC = 5 V ± 10 %, Temperature = 0 to 70°C
150 pF Load on all Outputs except CLKOUT (50 pF)

NAME	DESCRIPTION	MIN ns	MAX ns	NOTE
tdf	Delay XCLK to FSX, RCLK to FSR		25	
tdx	Delay XCLK to DX		25	
tsf	Setup FSX to XCLK, FSR to RCLK	6		
thf	Hold FSX to XCLK, FSR to RCLK	4		
tdzi	Delay XCLK to INHIBX		25	
tzdx	Delay XCLK to DX driving		25	note1
tdxz	Delay XCLK to DX float		25	note1
tsr	Set-up DR to RCLK	25		
thr	Hold DR to RCLK	25		

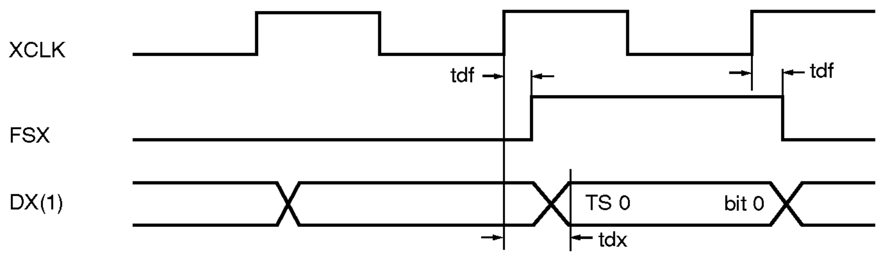
Note : 1. Applicable when DX output is programmed in either tri-state or open collector mode.

Internal Oscillator Schematic



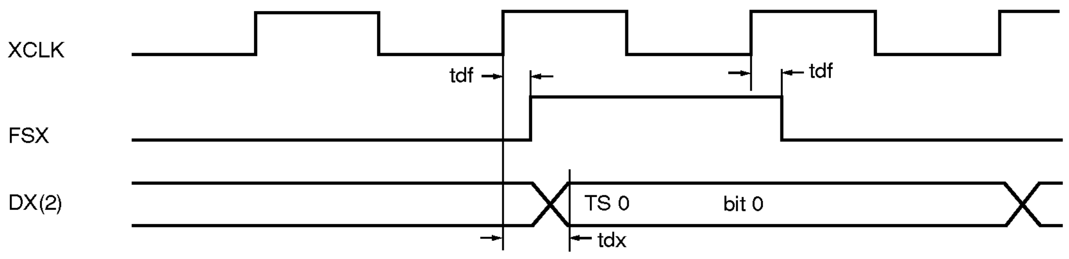
Typical value : R = 4.7 Mohm, C1 = 22 pF, C2 = 33 pF
When using an external clock XTAL is the clock input.

5.1. Transmit Simple Clock Master Mode

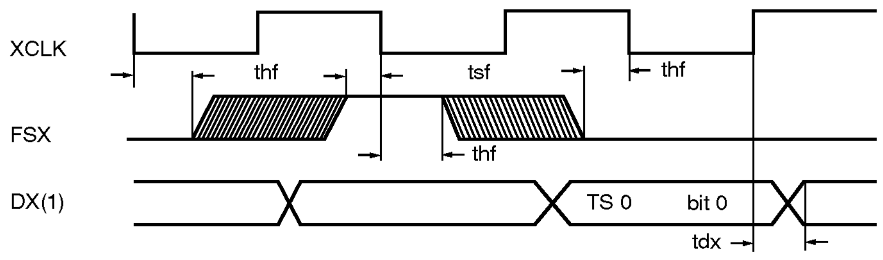


(1) assuming TRANSMIT offset is = OAH
(2) assuming TRANSMIT offset is = 14H

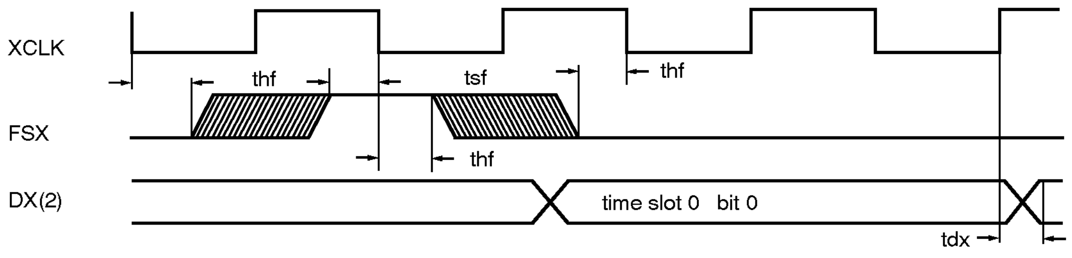
5.2. Transmit Double Clock Master Mode



5.3. Transmit Simple Clock Slave Mode

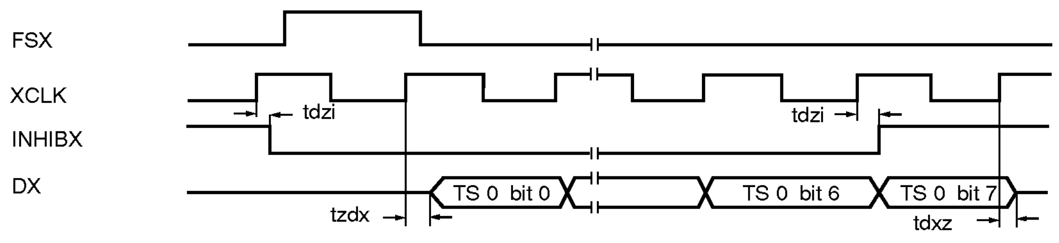


5.4. Transmit Double Clock Slave Mode



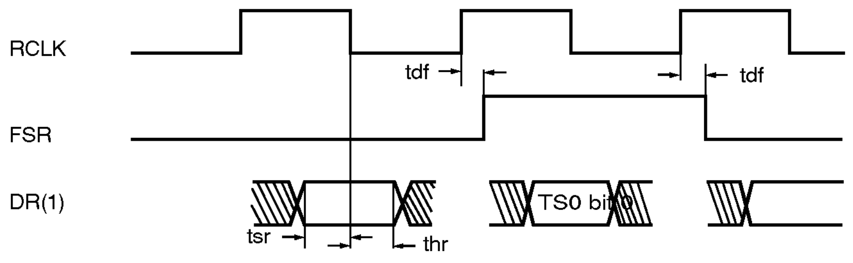
(1) assuming TRANSMIT offset is = OAH
(2) assuming TRANSMIT offset is = 14H

5.5. Transmit INHIBX Timing (In Master Simple Clock CEPT Mode)

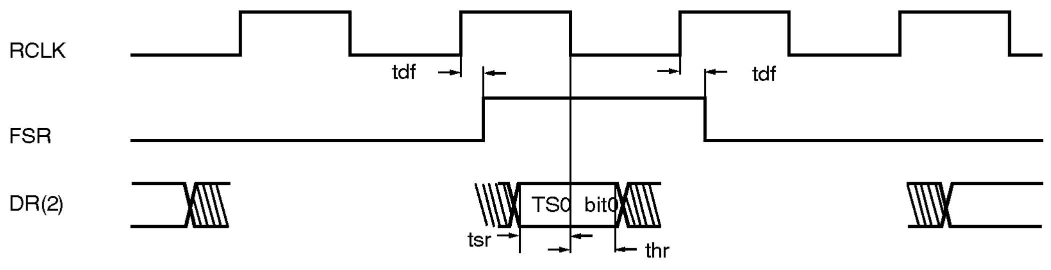


CONDITIONS
– TIME SLOT 0 VALID WITH FULL B CHANNEL (x s p = F F H)
– TIME SLOT 1 AND TIME SLOT 31 NON VALID (I T V X = 0)
– TRANSMIT : Offset = OAH
– DX OUTPUT = TRISTATE

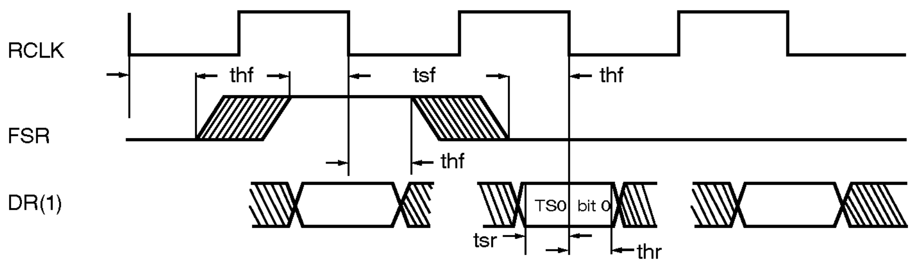
5.6. Receive Simple Clock Master Mode



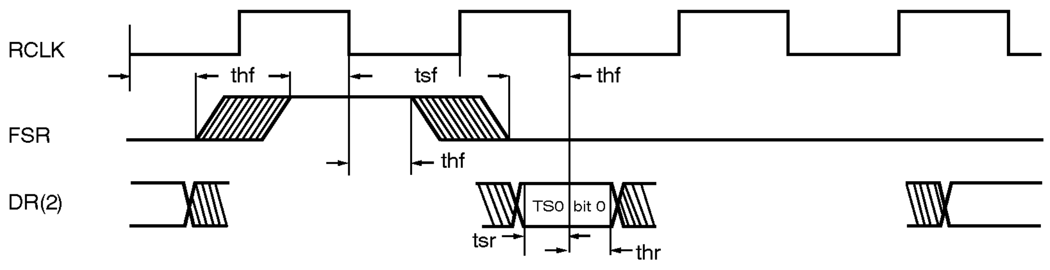
5.7. Receive Double Clock Master Mode



5.8. Receive Simple Clock Master Mode



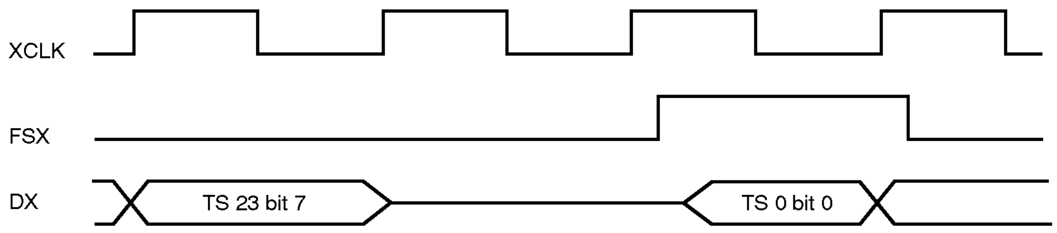
5.9. Receive Double Clock Slave Mode



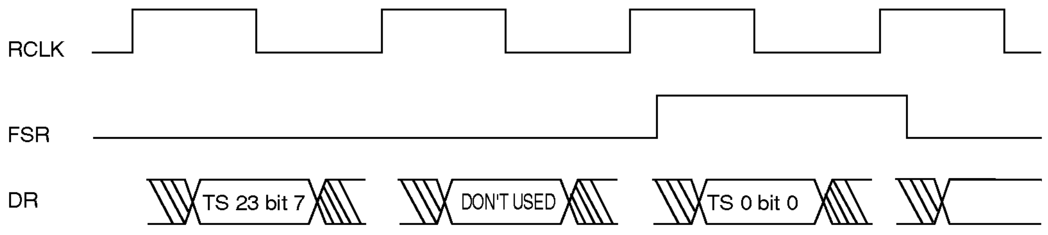
(1) assuming receive offset = 08H
(2) assuming receive offset = 10H

Detail TI/DSI Mode

5.10. Transmit Simple Clock Master Mode



5.11. Receive Simple Clock Master Mode



CONDITIONS
– DX OUTPUT = TRISTATE
– Xoffset = 0AH
– Roffset = 08H

6.0 System Bus Interface

Intel CPU Driving System Bus

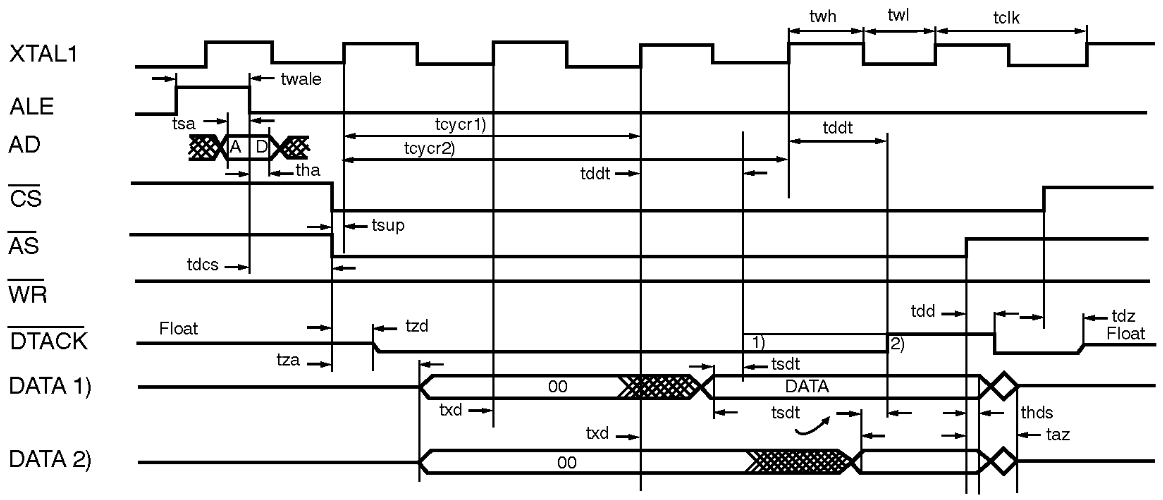
VCC = 5 V ± 10 %, Temperature = 0 to 70°C
150 pF Load on all Outputs except CLKOUT (50 pF)

Name	Description	Min ns	Typ ns	Max ns	Note
t_{clkmin}	XTAL clock period	30			
t_{wh}	XTAL high pulse width	12			
t_{wl}	XTAL low pulse width	12			
t_{wale}	ALE high pulse width	15	5		
t_{sa}	setup address to ALE	2			
t_{ha}	hold address to ALE	8			
t_{dcs}	delay ALE to AS/WR/CS	0			
t_{sup}	setup AS/WR/CS to XTAL1	0			note 4
t_{zd}	delay CS to DTACK driving			12	
t_{dd}	delay AS/WR to DTACK			10	
t_{ddt}	delay XTAL edge to DTACK			34	
t_{ddtr}	delay XTAL edge to DTACK			24	note 3
t_{dz}	delay CS to DTACK floatting			20	
t_{sdata}	setup DATA to DTACK	45			note 2
t_{hdata}	hold DATA to DTACK	– 15			
t_{sdatar}	setup DATA to DTACK	29			note 3
t_{hdatar}	hold DATA to DTACK	– 8			note 3
t_{cycw}	internal write cycle time	2/3 tclk	2/3 tclk	2/3 tclk	note 2
t_{cycr}	internal read cycle time	2/3 tclk	2/3 tclk	2/3 tclk	note 2
t_{za}	delay AS to DATA driving			14	
t_{az}	delay AS to DATA floatting			22	
t_{xd}	delay XTAL1 to data valid			62	
t_{sdt}	setup DATA to DTACK				tclk+tddt–txd
t_{sdtr}	setup DATA to DTACK	6			note 3
t_{hds}	hold DATA to AS	6			

Notes :
1. AS, WR, CS may be asynchronous with respect to XTAL1, however “tsup” setup time to the following XTAL1 rising edge must be res to insure that this rising edge will trig the internal R/W cycle overwize, one additionnal tclk will be added to thz R/W cycltime ti.
2. As the internal RAM uses 2 XTAL periods for internal R/W cycle, 2 or 3 XTAL periods are necessary from AS/WR resynchronized falling edge to internal cycle completion (DTACK rising).
3. Register access only.
   Remark : It is possible to maintain ALE = 1 (no address latch) but in that case, address should be valid during the entire I/O cycle. Setup address to CS falling = tsa + tdcs, hold time address to CS rising = 0 ns.
4. Care should be taken on ALE generation.
5. For CPU access without ALE signal (ALE tied to Vcc), the address must be valid during the entire cycle.

6.1. INTEL Mode

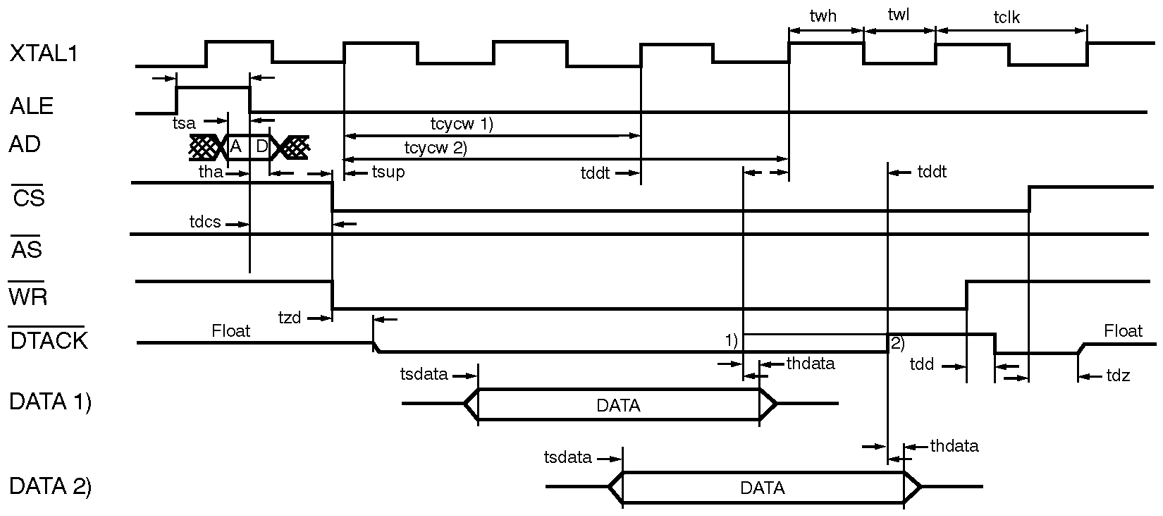
Memory Read



1) Shortest cycle
2) Longest cycle

6.2. INTEL Mode

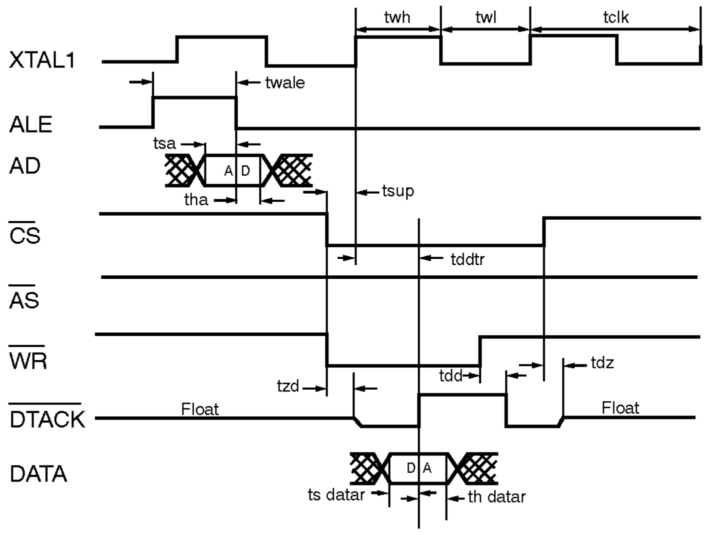
Memory Write



1) Shortest cycle
2) Longest cycle

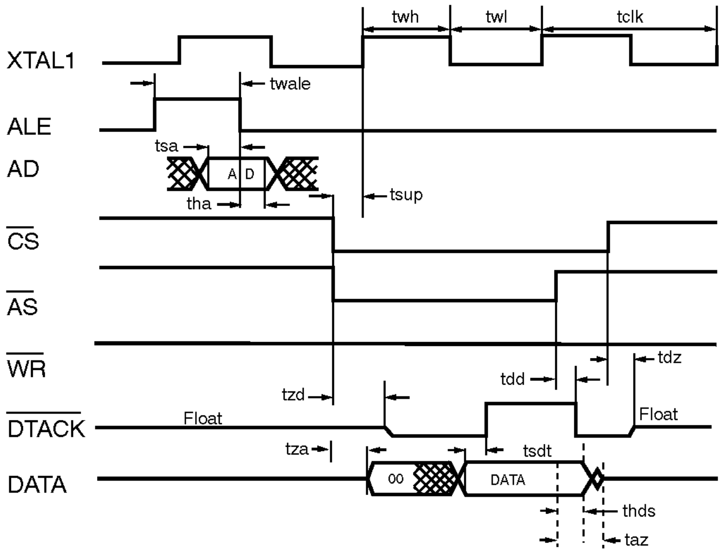
6.3. INTEL Mode

Register Write



6.4. INTEL Mode

Register Read



6.5. INTEL CPU DMA Cycle

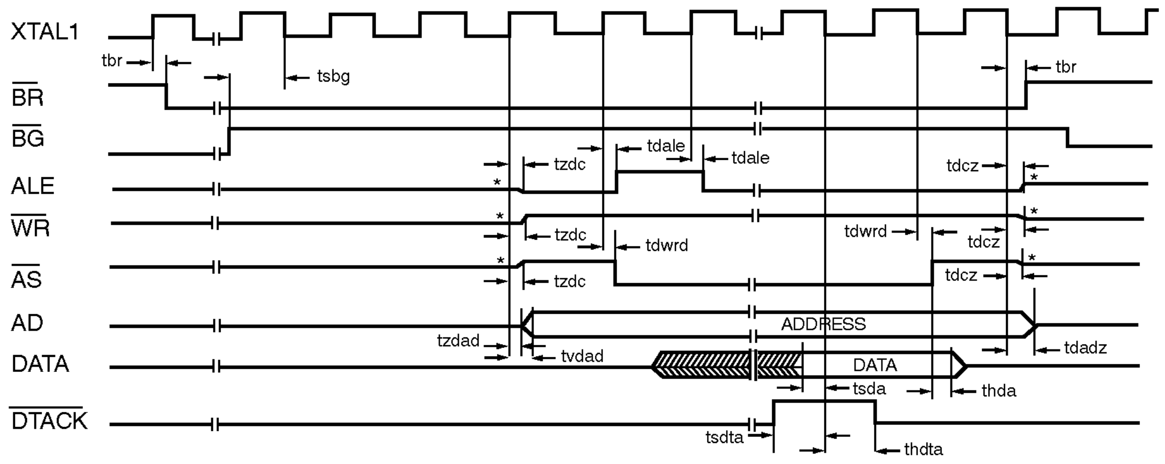
VCC = 5 V ± 10 %, Temperature = 0 to 70°C,
150 pF Load on all Outputs except CLKOUT (50 pF)

Name	Description	Min ns	Typ ns	Max ns	Note
t_{br}	Delay XTAL1 to BR			30	
t_{sbg}	Set-up BG to XTAL1	0			
t_{zdc}	Delay XTAL1 to Bus control driving (AS, WR, ALE)			27	
t_{zdad}	Delay XTAL1 to Add bus driving			38	
t_{vdad}	Delay XTAL1 to Add bus valid			39	
t_{zdda}	Delay XTAL1 to Data bus driv.			47	
t_{vdda}	Delay XTAL1 to Data bus valid			48	
t_{dale}	Delay XTAL1 to ALE			27	
t_{dwrd}	Delay XTAL1 to WR/AS			29	
t_{dcz}	Delay XTAL1 to bus control float (AS, WR, ALE)			45	
t_{dadz}	Delay XTAL1 to Add. bus float			52	
t_{ddaz}	Delay XTAL1 to Data bus float			45	
t_{sdta}	Set-up DTACK to XTAL1	0			
t_{hdta}	Hold DTACK to XTAL1	17			
t_{sda}	Set-up Data bus to XTAL1	2			
t_{hda}	Hold Data bus to AS	– 20			note 1

Notes : 1. thda shall be referenced to the XTAL1 falling edge that sample DT with the same figure.
2 DMA cycle Read then Write or Write then Read can be executed under the same BR/BG bus mastership cycle.

6.6. INTEL Mode

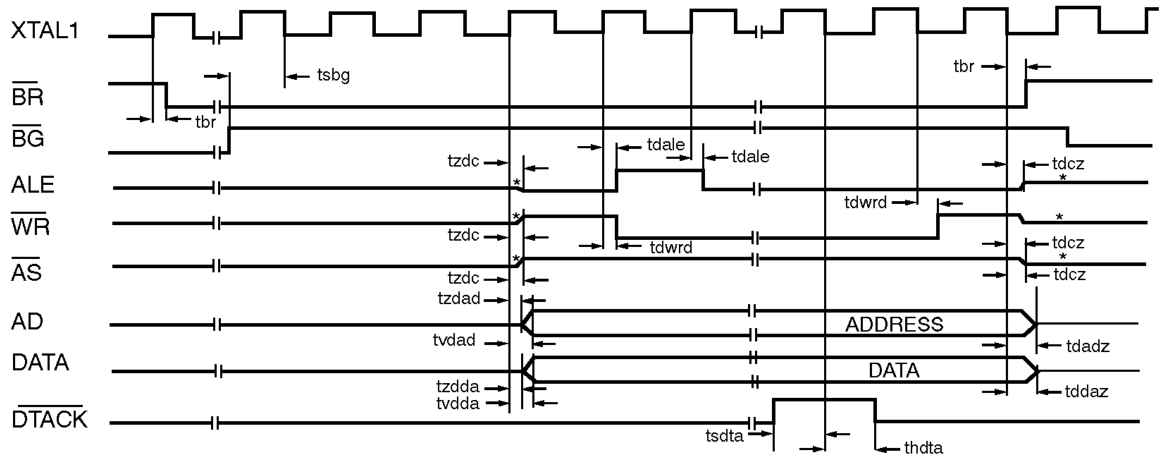
DMA Transmit



* $\overline{WR}$, $\overline{AS}$ need external pull-up. ALE need external pull-down.

6.7. INTEL Mode

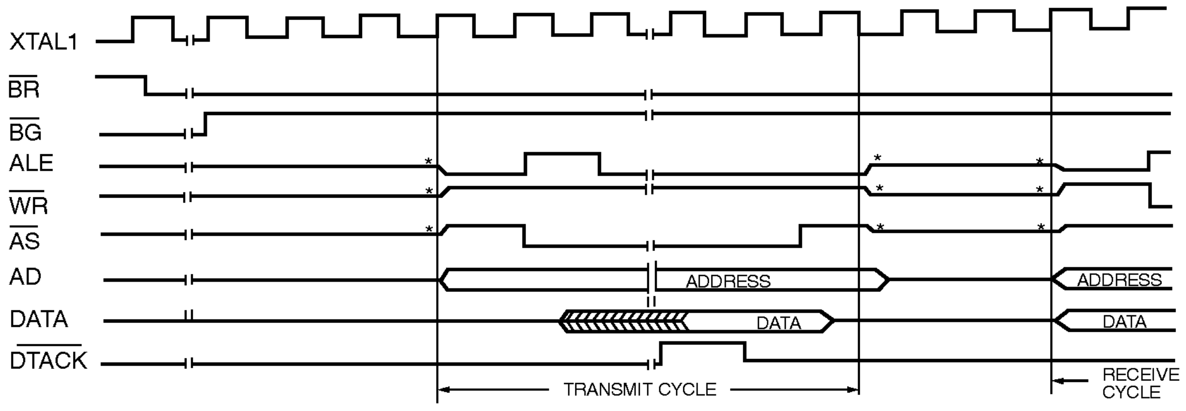
DMA Receive



* $\overline{WR}$, $\overline{AS}$ need external pull-up. ALE need external pull-down.

6.8. INTEL Mode DMA XMIT

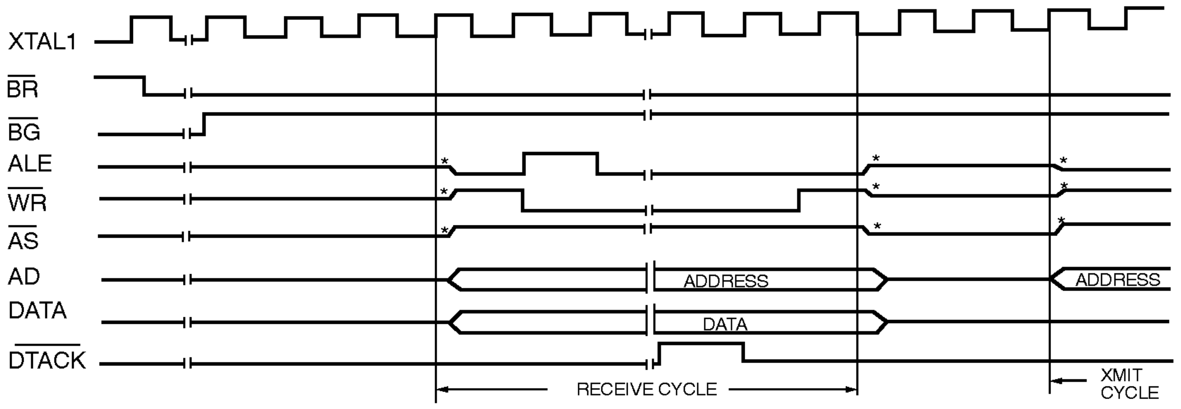
Then DMA receive under the same Bus Request



* $\overline{\text{WR}}$, $\overline{\text{AS}}$ need external pull-up. ALE need external pull-down.

6.9. INTEL Mode DMA Receive

Then DMA XMIT under the same Bus Request



* $\overline{\text{WR}}$, $\overline{\text{AS}}$ need external pull-up. ALE need external pull-down.

7.0 MOTOROLA CPU Driving System Bus

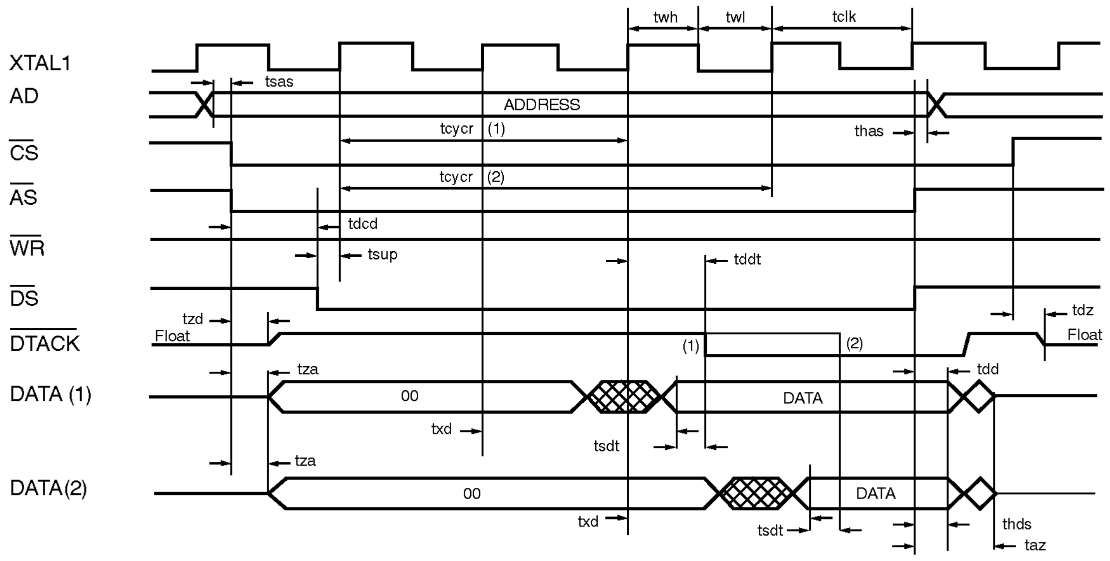
VCC = 5 V ± 10 %, Temperature = 0 to 70°C,
150 pF Load on all Outputs except CLKOUT (50 pF)

Name	Description	Min ns	Typ ns	Max ns	Note
t_{clk}	XTAL clock period	30			
t_{wh}	XTAL high pulse width	12			
t_{wl}	XTAL low pulse width	12			
t_{sas}	setup address to AS	0			
t_{has}	hold address to AS	10			
t_{dwcs}	setup WR to AS/CS	5			
t_{dcd}	setup AS/CS to DS	0			
t_{sup}	setup DS to XTAL1	0			note 1
t_{zd}	delay CS to DTACK driving			12	
t_{dd}	delay DS to DTACK			10	
t_{ddt}	delay XTAL edge to DTACK			34	
t_{ddtr}	delay XTAL edge to DTACK			26	note 3
t_{dz}	delay CS to DTACK floatting			24	
t_{sdata}	setup DATA to DTACK	45			note 2
t_{hdata}	hold DATA to DTACK	– 15			
t_{sdatar}	setup DATA to DTACK	29			note 3
t_{hdatar}	hold DATA to DTACK	– 8			note 3
t_{cycw}	internal write cycle time	2/3 tclk	2/3 tclk	2/3 tclk	note 2
t_{cycr}	internal read cycle time	2/3 tclk	2/3 tclk	2/3 tclk	note 2
t_{za}	delay AS to DATA driving			14	
t_{az}	delay AS to DATA floatting			17	
t_{xd}	delay XTAL1 to data valid			70	
t_{sdt}	setup DATA to DTACK				tclk+tddt–txd
t_{sdtr}	setup DATA to DTACK	6			note 3
t_{hds}	hold DATA to DS	6			

Notes :
1. AS, WR, CS may be asynchronous with respect to XTAL1, however "tsup" setup time to the following XTAL1 rising edge must be res to insure that this rising edge will trig the internal R/W cycle overwize, one additionnal tclk will be added to thz R/W cycltime ti.
2. As the internal RAM uses 2 XTAL periods for internal R/W cycle, 2 or 3 XTAL periods are necessary from AS/WR resynchronized falling edge to internal cycle completion (DTACK rising).
3. Register access only.

7.1. MOTOROLA Mode

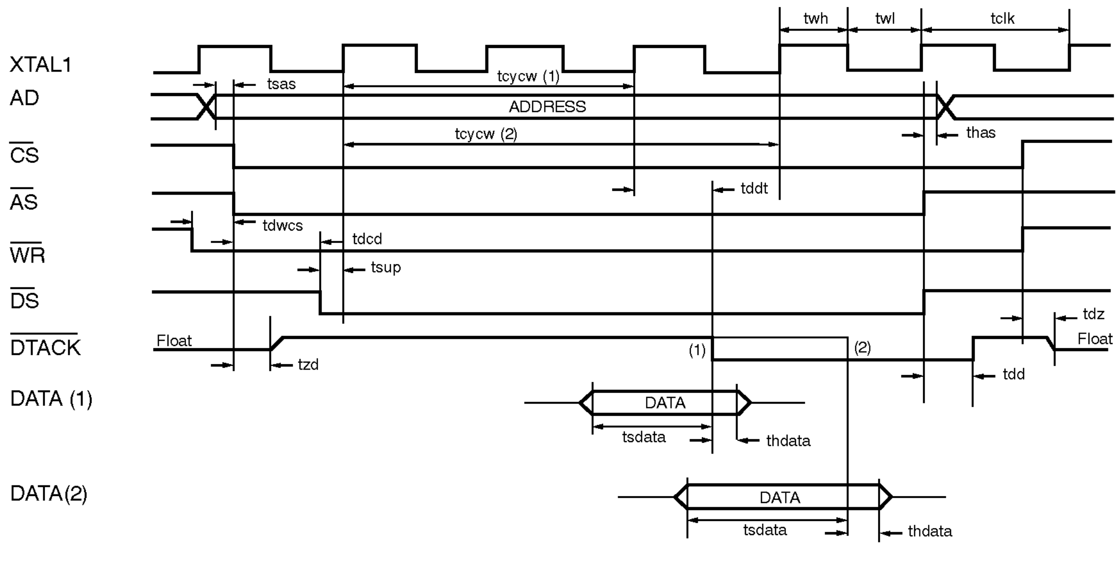
Memory Read



1) Shortest cycle
2) Longest cycle

7.2. MOTOROLA Mode

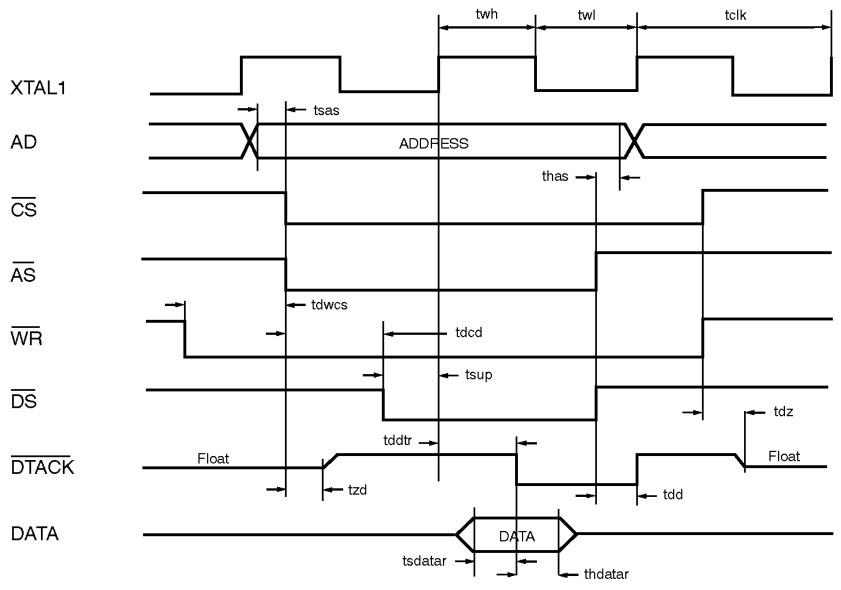
Memory Write



1) Shortest cycle
2) Longest cycle

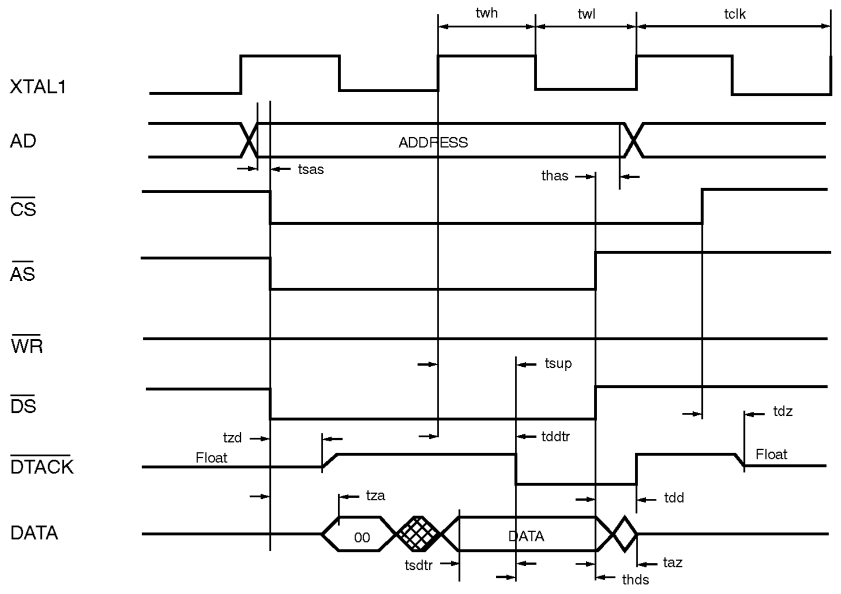
7.3. MOTOROLA Mode

Register Write



7.4. MOTOROLA Mode

Register Read



7.5. MOTOROLA CPU – DMA Cycle.

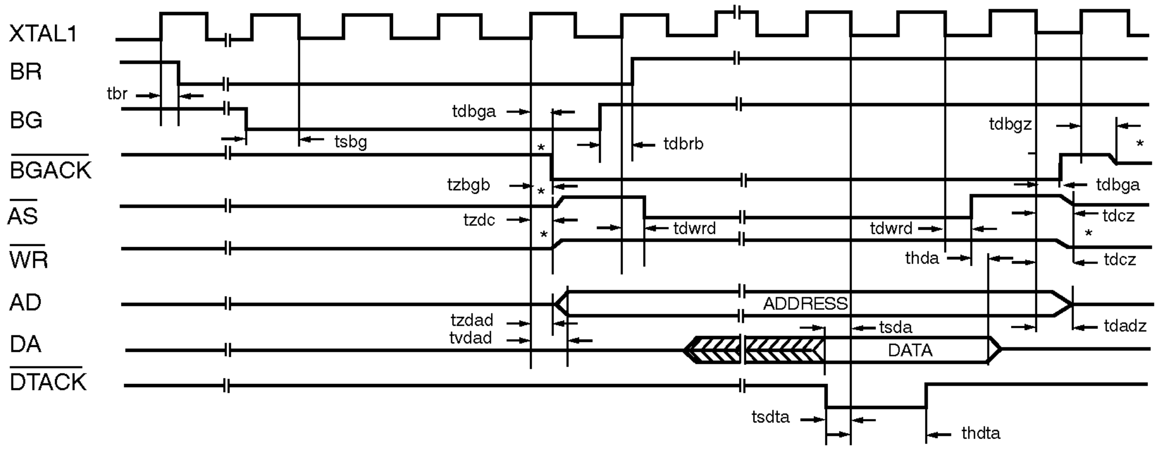
VCC = 5 V ± 10 %, Temperature = 0 to 70°C,
150 pF Load on all Outputs except CLKOUT (50 pF)

Name	Description	Min ns	Typ ns	Max ns	Note
t_{br}	Delay XTAL1 to BR			30	
t_{sbg}	Set-up BG to XTAL1	0			
t_{sasb}	Set-up AS to XTAL1	9			
t_{dbrb}	Delay BG to BR			27	
t_{dbga}	Delay XTAL1 to BGACK			30	
t_{zdc}	Delay XTAL1 to Bus control driving (AS, WR, ALE)			27	
t_{zbgb}	Delay XTAL1 to BGACK driving			28	
t_{zdad}	Delay XTAL1 to Add bus driving			38	
t_{vdad}	Delay XTAL1 to Add bus valid			39	
t_{zdda}	Delay XTAL1 to Data bus driv.			47	
t_{vdda}	Delay XTAL1 to Data bus valid			48	
t_{dwrd}	Delay XTAL1 to AS/WR			26	
t_{dcz}	Delay XTAL1 to bus control float (AS, WR)			45	
t_{dadz}	Delay XTAL1 to Add. bus float			52	
t_{ddaz}	Delay XTAL1 to Data bus float			45	
t_{sdta}	Set-up DTACK to XTAL1	0			
t_{hdta}	Hold DTACK to XTAL1	17			
t_{sda}	Set-up Data bus to XTAL1	2			
t_{hda}	Hold Data bus to AS	– 20			note 1

Notes : 1. thda shall be referenced to the XTAL1 falling edge that sample DT with the same figure.
Remarks : 2 DMA cycle Read then Write then Read can be executed under the same BR/BG bus mastership cycle.

MOTOROLA Mode

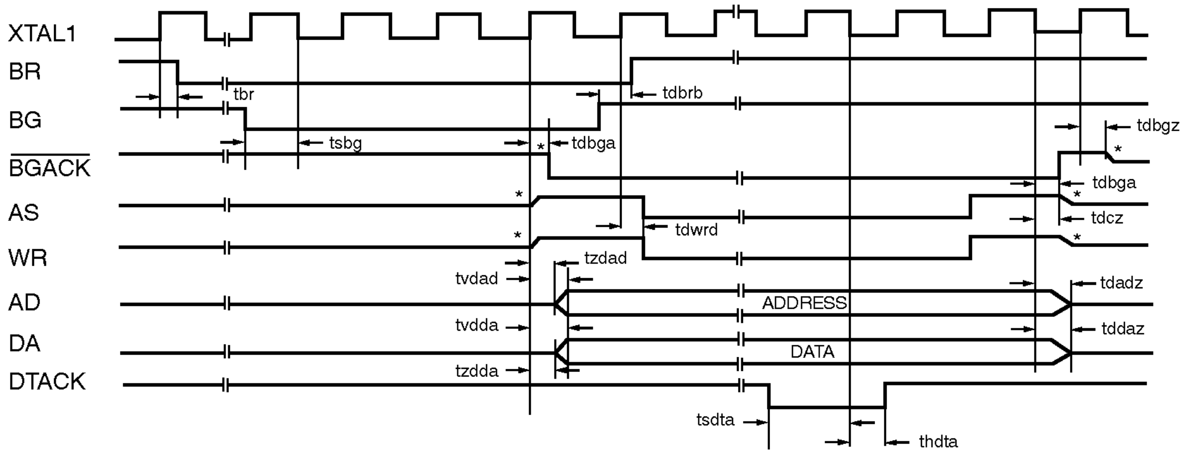
7.6. DMA Transmit



* $\overline{\text{BGACK}}$, $\overline{\text{AS}}$, $\overline{\text{WR}}$ need external pull-up.

MOTOROLA Mode

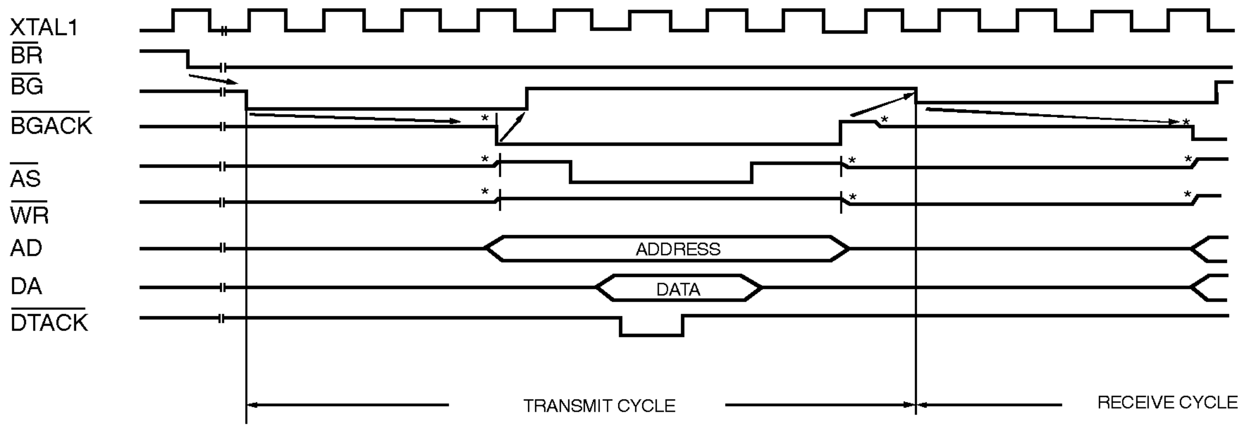
7.7. DMA Receiver



* $\overline{\text{BGACK}}$, $\overline{\text{AS}}$, $\overline{\text{WR}}$ need external pull-up.

MOTOROLA Mode DMA XMIT

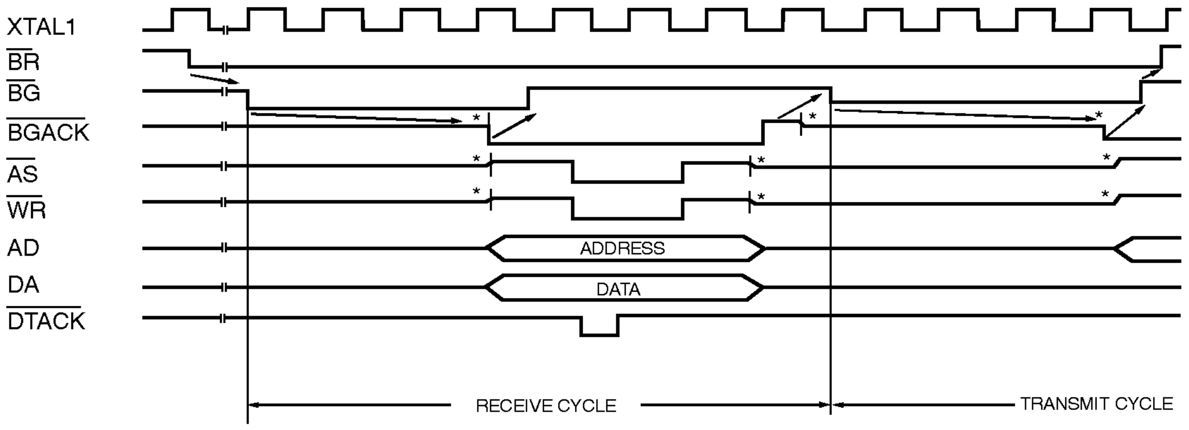
7.8. Then DMA Receive under the same Bus Request



* $\overline{\text{BGACK}}$, $\overline{\text{AS}}$, $\overline{\text{WR}}$ need external pull-up.

MOTOROLA Mode DMA Receive

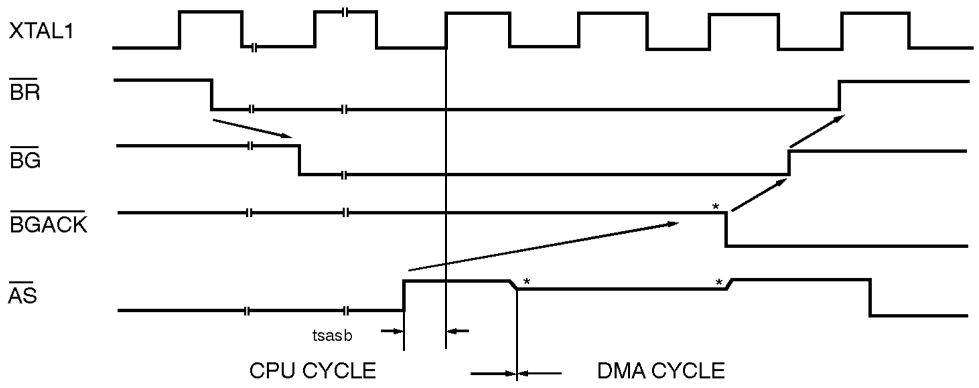
7.9. Then DMA Receive under the same Bus Request



* $\overline{\text{BGACK}}$, $\overline{\text{AS}}$, $\overline{\text{WR}}$ need external pull-up.

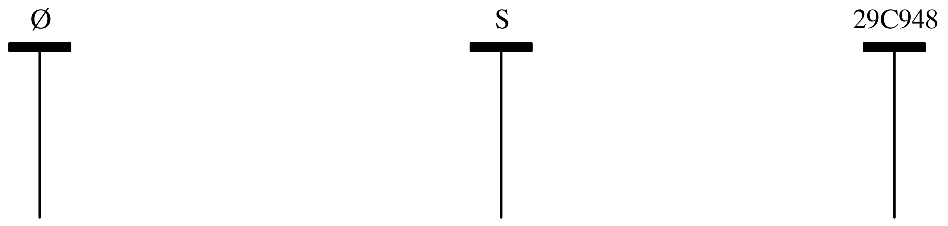
MOTOROLA Mode DMA Cycle

7.10. Starting after CPU Cycle



* $\overline{\text{BGACK}}$, $\overline{\text{AS}}$, $\overline{\text{WR}}$ need external pull-up.

Ordering Information



TEMPERATURE RANGE
Ø COMMERCIAL 0 TO 70 ˚C
I INDUSTRIAL −40 TO 85 ˚C

PACKAGE : S = PLCC

Part number
29C948

The information contained herein is subject to change without notice. No responsibility is assumed by MATRA MHS SA for using this publication and/or circuits described herein : nor for any possible infringements of patents or other rights of third parties which may result from its use.