STORAGE

WD60C40

Peripheral Cache

Manager Device

35

WESTERN DIGITAL

TABLE OF CONTENTS

Section	Title	Page
1.0	INTRODUCTION	35-1
	1.1 ARCHITECTURAL DESCRIPTION	35-1
	1.2 FEATURES	35-3
2.0	PIN DESCRIPTION	35-6
3.0	NON-CHANNEL REGISTERS	35-8
	3.1 OPTION REGISTER	35-8
	3.1.1 RRC3 thru RRC0 Refresh Rate Count Field	35-8
	3.1.2 SRAM Static RAM Mode (Bit 4)	35-8
	3.1.3 CAW1, CAW0 Column Address Width Field (Bits 6,5)	35-8
	3.1.4 MPAR Memory Parity Enable (Bit 7)	35-9
	3.2 OPTION REGISTER 2	35-10
	3.2.1 AINTE A Channel Interrupt Enable	35-10
	3.2.2 BINTE B Channel Interrupt Enable	35-10
	3.2.3 WAITE Wait Enable	35-10
	3.2.4 NOWAIT Non-Waitable Microprocessor Interface (Bit 5)	35-10
	3.3 MASTER STATUS REGISTER	35-11
	3.3.1 AINTR Channel A Interrupt Request Read Only Bit	35-11
	3.3.2 BINTR Channel B Interrupt Request Read Only Bit	35-11
	3.3.3 PPE Processor Parity Error	35-11
	3.3.4 BANR Buffer Access Not Ready Read Only Bit	35-11
	3.3.5 PRNR Power Reset Not Ready (Bit 4) Read Only Bit	35-11
	3.3.6 DNR Device Not Ready (Bit 7) Read Only Bit	35-11
	3.4 BUFFER DATA LATCH	35-12
	3.5 MICROPROCESSOR ADDRESS POINTER	35-12
	3.5.1 MP19 thru MP00 Microprocessor Address Pointers	35-12
	3.6 TEST ADDRESS & STATUS REGISTER	35-13
	3.7 RESET & TEST REGISTER	35-14
	3.7.1 MTPAF Microprocessor to Port A FIFO Test (Bit 0)	35-14
	3.7.2 MTPBF Microprocessor to Port B FIFO Test (Bit 1)	35-14
	3.7.3 CNTRT Counter Test Mode (Bit 2)	35-14
	3.7.4 TSMEM - Tri-State Memory Interface (Bit 3)	35-14
	3.7.5 ARST A Channel Reset (Bit 4)	35-14
	3.7.6 BRST B Channel Reset (Bit 5)	35-15
	3.7.7 SWRST Software Reset (Bit 7)	35-15
	3.8 BUFFER ACCESS REGISTER	35-15
	3.9 AUTOINCREMENT ACCESS REGISTER	35-16
4.0	DEVICE CHANNELS	35-16
	4.1 CHANNEL TIMING REGISTER	35-16
	4.1.1 SC0(1)A (SC0(1)B) Channel Strobe Control Field	35-17
	4.1.2 DLYA (DLYB) Delay Strobe Bit	35-17
	4.1.3 SDTCA (SDTCB) Strobe Deasserted Time Control Bit	35-18

TABLE OF CONTENTS, Continued

Section	Title	Page
4.0, Cont.	DEVICE CHANNELS	
	4.1.4 RQPLA (RQPLB) DMA Request Polarity Bit	35-18
	4.1.5 DKPLA (DKPLB) DMA Acknowledge Polarity Bit	35-18
	4.1.6 PPEA (PPEB) Port Parity Enable Bit	35-18
	4.1.7 LPBMA (LPBMB) Loop Back Mode Enable	35-18
	4.2 CHANNEL CONTROL REGISTER	35-19
	4.2.1 IBEA (IBEB) Interrupt on Busy Bit	35-19
	4.2.2 DIRA (DIRB) Channel Transfer Direction Bit	35-19
	4.2.3 PAUSA (PAUSB) Channel Pause Control Bit (Bit 3)	35-19
	4.2.4 EDACA (EDACB) EDAC Idle Enable (Bit 4)	35-20
	4.2.5 DISKA (DISKB) Disk Type Device (Bit 5)	35-20
	BRSTA (BRSTB) Burst Transfere Device (Bit 6)	35-20
	SLAVA (SLAVB) Slave Mode Interface (Bit 7)	35-20
	4.2.6 000 DMA Single Cycle Master	35-20
	4.2.7 002 DMA Single Cycle Disk	35-21
	4.2.8 010 DMA Burst Cycle Master	35-21
	4.2.9 011 Programmed I/O Mode	35-21
	4.2.10 100 Single Cycle Slave Mode	35-22
	4.2.11 Unused & Reserved Mode	35-22
	4.2.12 110 Burst Cycle Slave Mode	35-22
	4.2.13 111 Slave Burst Mode Disk	35-22
	4.3 CHANNEL STATUS REGISTER	35-23
	4.3.1 BSYA (BSYB) Channel Busy Status	35-23
	4.3.2 VBSYA (VBSYB) Channel Very Busy Status	35-23
	4.3.3 FMTA (FMTB) Channel FIFO Empty Status (Bit 2)	35-23
	4.3.4 PNRA (PNRB) Port Not Ready Status (Bit 3)	35-23
	4.3.5 RQSTA (RQSTB) Channel DRQ Pin Status (Bit 4)	35-23
	4.3.6 DKSTA (DKSTB) Channel DACK Pin Status (Bit 5)	35-23
	4.4 CHANNEL INTERRUPT STATUS REGISTER	35-24
	4.4.1 BSYIA (BSYIB) Interrupt From Busy	35-24
	4.4.2 VBIA (VBIB) Interrupt From Very Busy	35-24
	4.4.3 PERRA (PERRB) Channel Parity Error Bit	35-24
	4.4.4 LATEA (LATEB) Channel Data Late Error Bit (Bit 3)	35-24
	4.4.5 REJA (REJB) Channel Command Reject Error Bit (Bit 4)	35-24
	4.4.6 IOEA (IOEB) I/O Error Bit (Bit 5)	35-24
	4.4.7 IOPEA (IOPEB) Programmed I/O Parity Error (Bit 6)	35-25
	4.4.8 AERRA (AERRB) Any Error Bit (Bit 7)	35-25
	4.5 CHANNEL DATA LATCH	35-26
	4.6 CHANNEL BUFFER POINTER	35-27
	4.6.1 BP19A (B19B) thru BP00A (BP00B) Buffer Pointer Address Bits	35-27

TABLE OF CONTENTS, Continued

Section	Title	Page
4.0, Cont.	DEVICE CHANNELS	
	4.7 CHANNEL TRANSFER COUNTER	35-28
	4.7.1 TC15 thru TC00 Transfer Counter Bits	35-28
	4.8 CHANNEL START REGISTER	35-29
	4.9 CHANNEL STOP REGISTER	35-30
	4.10 CHANNEL POINTER CAPTURE REGISTER	35-30
	4.11 EDAC IDLE COUNTER	35-31
5.0	MICROPROCESSOR INTERFACE	35-31
6.0	I/O MEMORY MAP	35-32
7.0	INTERRUPTS	35-33
8.0	RESET SEQUENCES	35-33
9.0	READY SEQUENCES & LATENCIES	35-33
10.0	NON-CHANNEL REGISTERS MAP	35-35
11.0	CHANNEL REGISTERS MAP	35-36
12.0	ELECTRICAL CHARACTERISTICS	35-37
13.0	MISCELLANEOUS CHARACTERISTICS	35-38
14.0	TIMING DIAGRAMS	35-39
15.0	PACKAGE DIAGRAMS	35-58

LIST OF ILLUSTRATIONS

Figure	Title	Page
1	WD60C40 Block Diagram	35-4
2	WD60C40 in 80188 Board Environment	35-5
3	Pin Designation	35-7
4	Rise/Fall and Miscellaneous Timing	35-40
5	PIO Read/Write of External Devices	35-41
6	DMA Burst Mode Transfers	35-42
7	DMA Single Cycle Mode	35-43
8	Bus Master Disk Mode	35-44
9	Slave Burst Mode Transfers	35-45
10	Slave Single Cycle Transfer	35-46
11	Slave Burst Disk Mode (Read)	35-47
12	Slave Burst Disk Mode (Write)	34-48
13	EDAC Mode (DMA Single Cycle)	35-49
14	EDAC Mode (DMA Disk)	35-50
15	Miicroprocessor Bus Timing	35-51
16	RAS Only Refresh Timing	35-53
17	Page Mode Write Timing	35-54
18	Page Mode Read Timing	35-55
19	Static RAM Write Timing	35-56
20	Static RAM Read Timing	35-57
21	84 Lead PLCC	35-58
22	84 Lead PQFP	35-59

1.0 INTRODUCTION

1.1 ARCHITECTURAL DESCRIPTION

The WD60C40 peripheral cache manager (PCM) controls a large amount (up to 1 megabyte) of dynamic RAM, and the buffering required to allow the seemingly simultaneous access of this buffer by up to 3 requestors. The three requestors are assumed to be two block oriented devices, and a microprocessor.

The two devices interface to the dynamic RAM, are designed to support block oriented peripheral formatter chips, such as the ADS10C00 Hard Disk Formatter, or block oriented bus interface chips, such as the WD33C93A SCSI Bus Interface Chip. The implication of block oriented devices, is that the devices will not have the capability of randomly accessing data within the memory array. This restriction allows for the removal of the addressing pins that are normally associated with an interface. The 20 bit address counters that are required to select a byte within the array, are now incorporated into the WD60C40.

The buffer control logic interfaces with an external dynamic RAM array and arbitrates the access into this array from the three external sources. The internal arbitrator in the PCM also includes the refresh logic that is internal to the PCM into its arbitration. Each device port has built into it a byte-wide FIFO that will act as a buffer while the port is waiting for its time slot in the arbitration mechanism.

The dynamic RAM memories do not have the random access bandwidth to sustain multiple high speed devices. The dynamic RAM memories do incorporate a feature known as "page mode or static column" that increases their bandwidth to the point that they will sustain multiple high speed devices. The dynamic RAMs have a multiplexed address bus with the address into the array being divided into two parts, a row address and a column address.

The dynamic RAMs allow a feature that if the next access into the array is in the same row (same page) then only the column address must be up dated from the addressing logic. This scheme carries the penalty that a FIFO must be available to hold the data to/from the port to ensure that when a port gains access to the RAM, it can transfer data in a continuous block. The PCM incorporates 15 bytes of FIFO into each of the peripheral ports to support the page mode memory and CAS control static column architecture. This scheme has the advantage that the memory bandwidth is increased by a factor of about 75% over the random access bandwidth, allowing a large buffer memory to be mechanized by inexpensive dynamic RAMs, but having the performance characteristics of a static RAM buffer.

The external dynamic RAM buffer may be 8 bits or 9 bits wide. The depth of the array and the organization of the devices that make up the array are as follows:

SIZE	ORGANIZATION		
16 KB	2	216Kb x 4	(+ 64Kb x 1 parity)
64 KB	2	64Kb x 4	(+ 64Kb x 1 parity)
64 KB	8	64Kb x 1	(+ 64Kb x 1 parity)
256 KB	2	256Kb x 4	(+ 256Kb x 1 parity)
256 KB	8	256Kb x 1	(+ 256Kb x 1 parity)
1 MB	2	1Mb x 4	(+ 1Mb x 1 parity)
1 MB	8	1Mb x 1	(+ 1Mb x 1 parity)

In the following description, refer to Figure 1, the block diagram of the WD60C40, and to the example board environment in Figure 2. A typical situation to use as an example of the actions within the part is that of a disk controller. In this example, port 'B' of the device is connected to a bus interface controller (ex. WD33C93A), and port 'A' of the device is connected to a disk interface controller (ex. ADS10C00).

The example begins with the assumption that all devices are currently active, and the refresh time has arrived, and the microprocessor is requesting a data item. The PCM is currently servicing the FIFO that is attached to the device port 'A'. The microprocessor is currently at a wait bus cycle because of the RDY signal from the PCM being inactive. The PCM will move data to/from the memory and the port 'A' FIFO until the FIFO is full/empty, or the transfer counter of port 'A' exhausts. If during this sequence, the PCM detects that a dynamic RAM page boundary is being crossed, the PCM will update the row address, and continue the page mode transfer. While this action is occuring, any byte transfers to/from the other device port are stored in that device port's FIFO. When port 'A' has finished its data transfers, the PCM will first service the host data re-

quest, then the refresh request(s), both of which have priority over block transfers. The PCM will then initiate transfer between the port 'B' FIFO, and the dynamic RAMs.

The PCMs priority system when arbitrating is:

- microprocessor requests
- refresh requests
- the higher priority peripheral channel
- the lower priority peripheral channel

The control of the peripheral channel priority is under firmware control if the channels are peers in their capability of sustaining a pause in their transfers. If a channel is programmed as 'non-pausible', and the other channel is 'pausible' then this channel is given priority in the arbitration mechanism regardless of the other firmware priority controls. If the channels are peers, such as both being 'pausible' or both being 'non-pausible', then the priority of the channels is controlled by firmware selecting one of the channels as higher priority through option register 2. See also the descriptions of the AHI bit in option register 2, and PAUS bits in the channel control register.

Note that arbitration does not occur until the requestor currently being serviced has completed its entire burst. This means that page mode bursts between the channels and memory are not interrupted by requests from higher priority requestors. There are two exceptions to this general rule. The first exception is a non-pausible device that is selected by firmware as having higher arbiter priority. This channel is then allowed an "urgent request", when its FIFO is almost exhausted. This "urgent request" will send a false end of transfer signal through the other port and cause arbitration to occur, and this channel will then win arbitration if the single cycle devices are not requesting. This is done to minimize the occurence of over/under-run errors in high priority non-pausible devices. The second exception that causes a burst to be halted is for a refresh burst. The refresh request occurs at regular intervals (see option register section for specific details), but if it is not serviced because of a channel burst, then the request is queued into a counter. When 4 refresh cycles have been queued (about 60 µsec.), then the "urgent request" mechanism will force arbitration, and a burst of 4 refresh cycles will be performed. This is done to maintain memory data integrity.

To finish the architectural description of the device, the last items are the programmable features of the internal logic that allow the controller firmware to control and "tune" the actions of the PCM. The first item the PCM needs to be aware of is if the devices are capable of being throttled. Most devices that attach to peripherals have minimum data buffering, so the port logic must be programmed to look for overrun/underrun situations when the FIFO is full/empty. Some devices that attach to busses or other peripherals that have variable data rates, are capable of being 'stalled' when the FIFO is full/empty, and the port logic is capable of holding the port until the FIFO is ready.

The PCM also implements a "pipelined" pointer mechanism that allows the programmer to set the pointer (buffer address) of the current device transfer, and if required, set the pointer of the next transfer, while the current transfer is taking place. This address is held in the address pointer holding register to be automatically transferred to the address pointer at the end of the current transfer. If the next transfer's buffer is contiguous with the current transfer's buffer, then the address pointer "pipeline" need not be loaded, and when the next transfer begins the address pointer will continue from its current position. This address is therefore only necessary if the next transfer is at a memory address that is not contiguous with the end of the current transfer. The transfer counter is always loaded from the transfer counter holding register at the start of a new block transfer, but the transfer counter holding register need only be reloaded if the block size of a transfer needs to be changed. When the current transfer finishes, the firmware can be interrupted, while the hardware continues directly into the next transfer. With this mechanism, the firmware now has a block time to determine what is to be done when this transfer finishes, rather than doing this during the inter-block gap times. If pipelined (ie. continuous) transfers are required then microprocessor action is required whenever the pipeline becomes empty. If the next transfer is the same size as the current, and contiguous in address, then the firmware need only issue a new channel start command. Alternatively the firmware can load new values into the "pipeline" registers before issuing the start, and dynamically control the transfer size and buffer location. The firmware can poll the status or wait for an interrupt (if enabled) to indicate that the holding registers are available.

The device channels are mechanized with 8 or 9 data lines, and 5 control/handshake lines. Four control signals are programmable to allow interfaceing to multiple types of device controller chips. Features such as the master/slave relationships between the PCM and the external peripheral controller, the polarity and timing of the signals that do the handshake, input/output control direction, and the mode that the control lines emulate are all programmable.

1.2 FEATURES

- 5 Volt Only Operational Power.
- 1.25 micron CMOS for low power consumption.
- 84 pin JEDEC PLCC or 84 pin JEDEC PQFP packages.
- Supports dynamic memory configurations from 16 KB to 1 MB.
- Supports static memory configurations.
- Supports parity on the memory array.
- Supports parity on peripheral ports.
- Aggregate memory bandwidth of greater than 9.0 Mbytes/sec, using industry standard 100 ns. dynamic RAMs.
- Aggregate memory bandwidth of greater than 12 Mbytes/sec, using 80 ns. dynamic RAMs
- Supports 2 peripheral channels.
- High Speed Channels support peripheral data rates of:
  - 6.25 Mbytes/sec as bus master with 25 MHz input clock, the maximum data rate is input clock divided by 4.
  - 10.0 Mbytes/sec as bus slave, with 25 MHZ input clock, the maximum data rate is input clock divided by 2.5.
- Programmable configuration of device channel interface.
- Internal FIFOs on peripheral channels.
- Supports independant transfers on both peripheral ports and microprocessor simultaneously.
- Large memory allows peripheral controllers to be implemented that "decouple" the host and disk transfer rates, maintaining 1:1 disk interleave regardless of host transfer rate.
- Provides "minimum chip count" solution for mechanizing a peripheral cache controller.
- Supports high speed microprocessor bus cycles, such as Intel* 80186 or 80188 type processors running at 12.5 MHz with no wait states required when accessing the PCM task file registers.

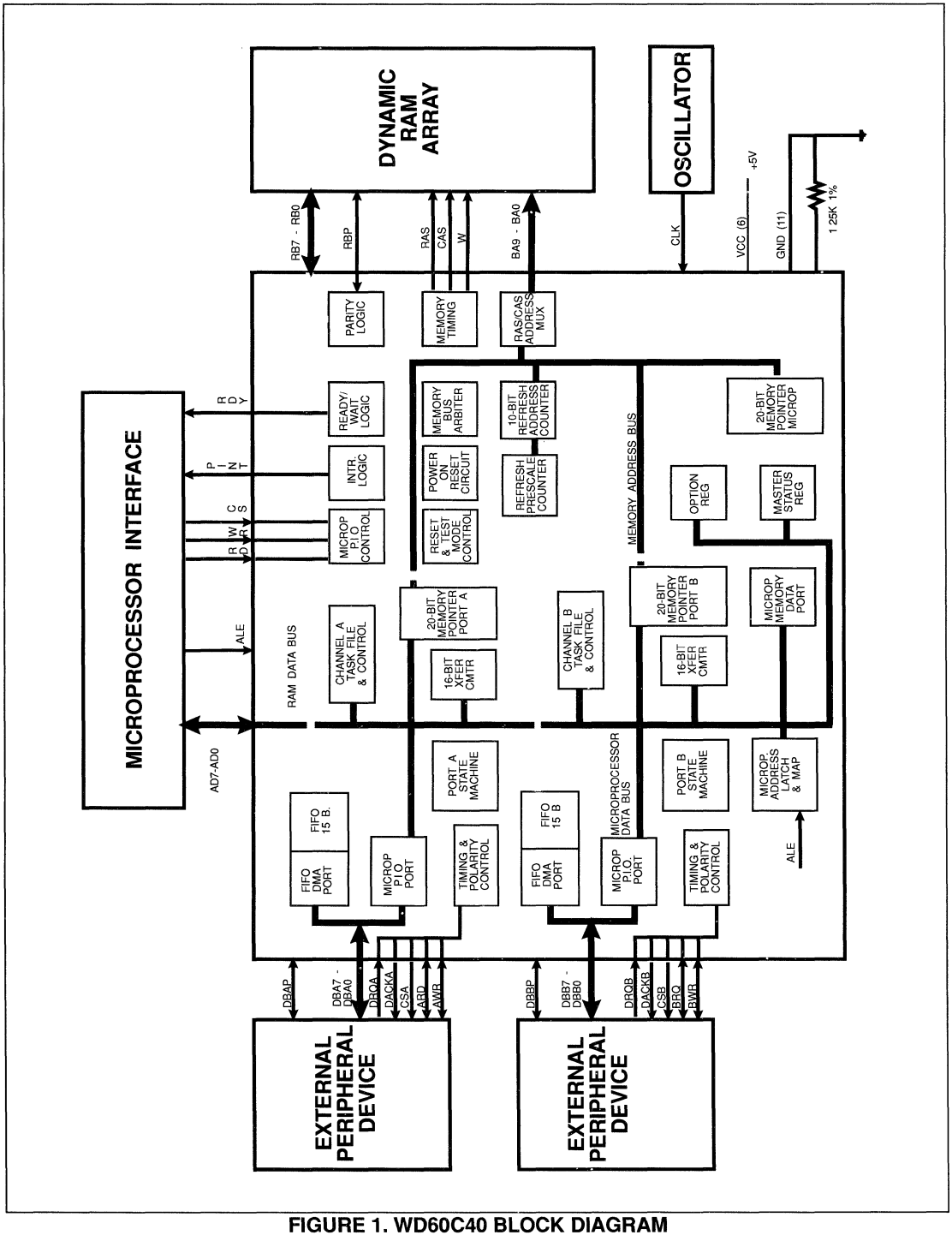


FIGURE 1. WD60C40 BLOCK DIAGRAM

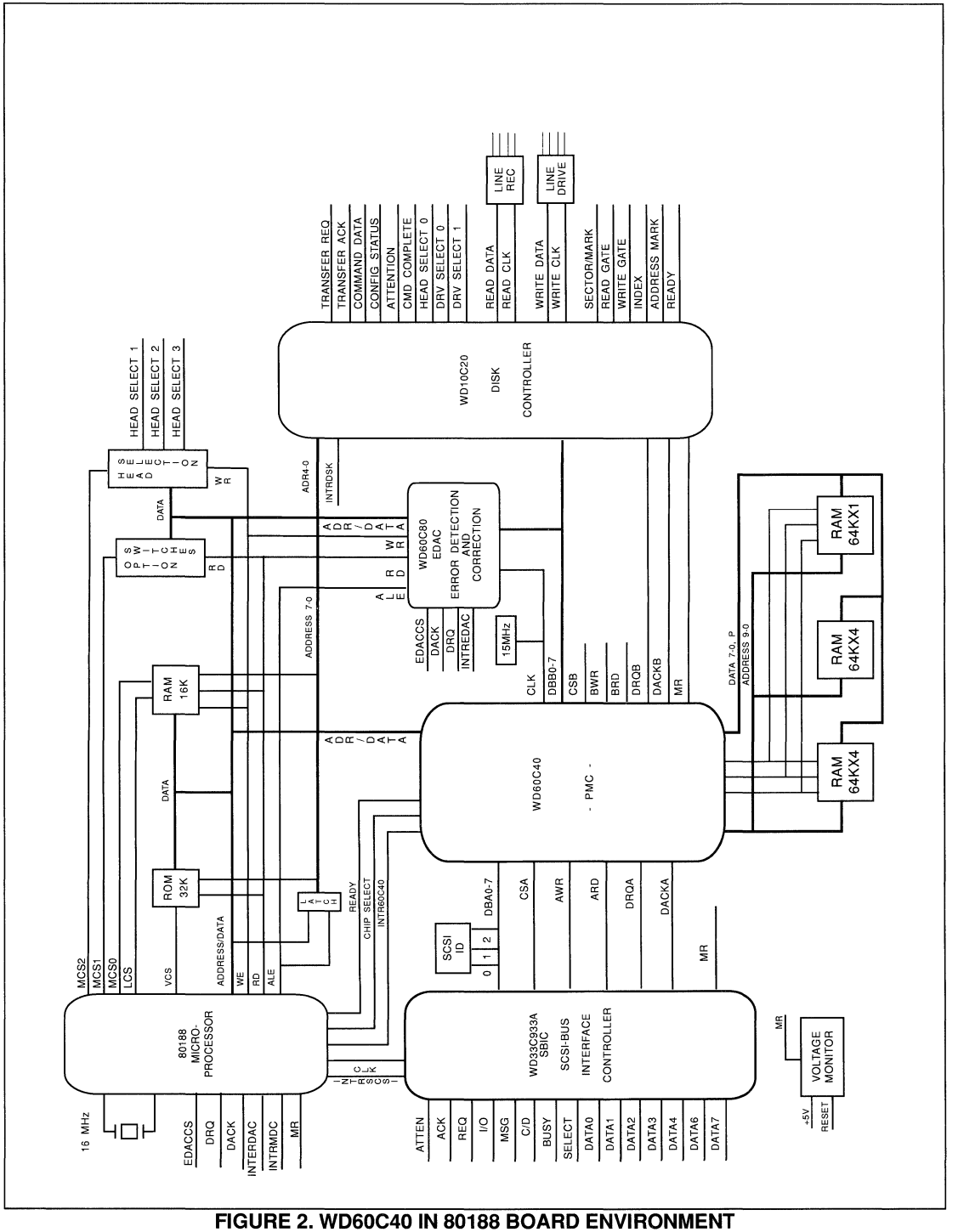


FIGURE 2. WD60C40 IN 80188 BOARD ENVIRONMENT

2.0 PIN DESCRIPTION

PIN NUMBER	MNEMONIC	SIGNAL NAME	I/O
1,22,30,43, 56,64	V_{CC}	+5 Volt power supply connection	I
2,11,21,23, 29,42,44,57, 63,65,75	GND	Ground power supply connection	I
3 thru 10	AD0 thru AD7	µP ADDRESS/DATA BUS	I/O
12 thru 19	DBB0 thru DBB7	DEVICE BUS B DATA	I/O
20	DBBP	DEVICE BUS B PARITY BIT	I/O
24	$\overline{CSB}$	CHANNEL SELECT B	O
25	DRQB	DEVICE PORT B CYCLE REQUEST	I/O[1]
26	DACKB	DEVICE PORT B CYCLE ACKNOWLEDGE	I/O[2]
27	$\overline{BRD}$	DEVICE PORT B READ STROBE	I/O
28	$\overline{BWR}$	DEVICE PORT B WRITE STROBE	I/O
31 thru 38	RB7 thru RB0	DYNAMIC RAM DATA BUS	I/O
39	RBP	DYNAMIC RAM PARITY BIT	I/O
40	W/$\overline{WE}$	DYNAMIC RAM WRITE/STATIC RAM $\overline{WE}$	O
41	$\overline{CAS}$/$\overline{OE}$	DYNAMIC RAM $\overline{CAS}$/STATUS RAM $\overline{OE}$	O
45	$\overline{RAS}$	RAM ROW ADDRESS STROBE	O
46 thru 55	BA0[3] thru BA9	RAM ADDRESS BUS[3]	O
58	$\overline{AWR}$	$\overline{DEVICE\ PORT\ A\ WRITE\ STROBE}$	I/O
59	$\overline{ARD}$	$\overline{DEVICE\ PORT\ A\ READ\ STROBE}$	I/O
60	DACKA	DEVICE PORT A CYCLE ACKNOWLEGE	I/O[2]
61	DRQA	$\overline{DEVICE\ PORT\ A\ CYCLE\ REQUEST}$	I/O[1]
62	$\overline{CSA}$	$\overline{CHANNEL\ SELECT\ A}$	O
66	DBAP	DEVICE BUS A PARITY BIT	I/O
67 thru 74	DBA7 thru DBA0	DEVICE BUS A DATA BUS	I/O
76	CLK	CLOCK	I
77	$\overline{RST}$	$\overline{RESET}$	I
78	$\overline{CS}$	$\overline{\mu P\ BUS\ CHIP\ SELECT}$	I
79	$\overline{WR}$	$\overline{\mu P\ BUS\ WRITE\ CYCLE\ STROBE}$	I
80	$\overline{RD}$	$\overline{\mu P\ BUS\ READ\ CYCLE\ STROBE}$	I
81	ALE	µP ADDRESS STROBE	I
82	$\overline{PINT}$	$\overline{\mu P\ INTERRUPT\ REQUEST}$	O[4]
83	RDY	µP WAIT CONTROL	O[4]
84	RBIAS[5]	RESISTOR BIAS[5]	I

NOTES:

1 I = WD60C40 bus master mode. O = WD60C40 bus slave mode

2 I = WD60C40 bus slave mode. O = WD60C40 bus master mode.

3 BA0 is the LSB.

4 Open drain output

5 Resistor bias for output drive circuitry 1.24KΩ ± 1% connected from this pin to ground.

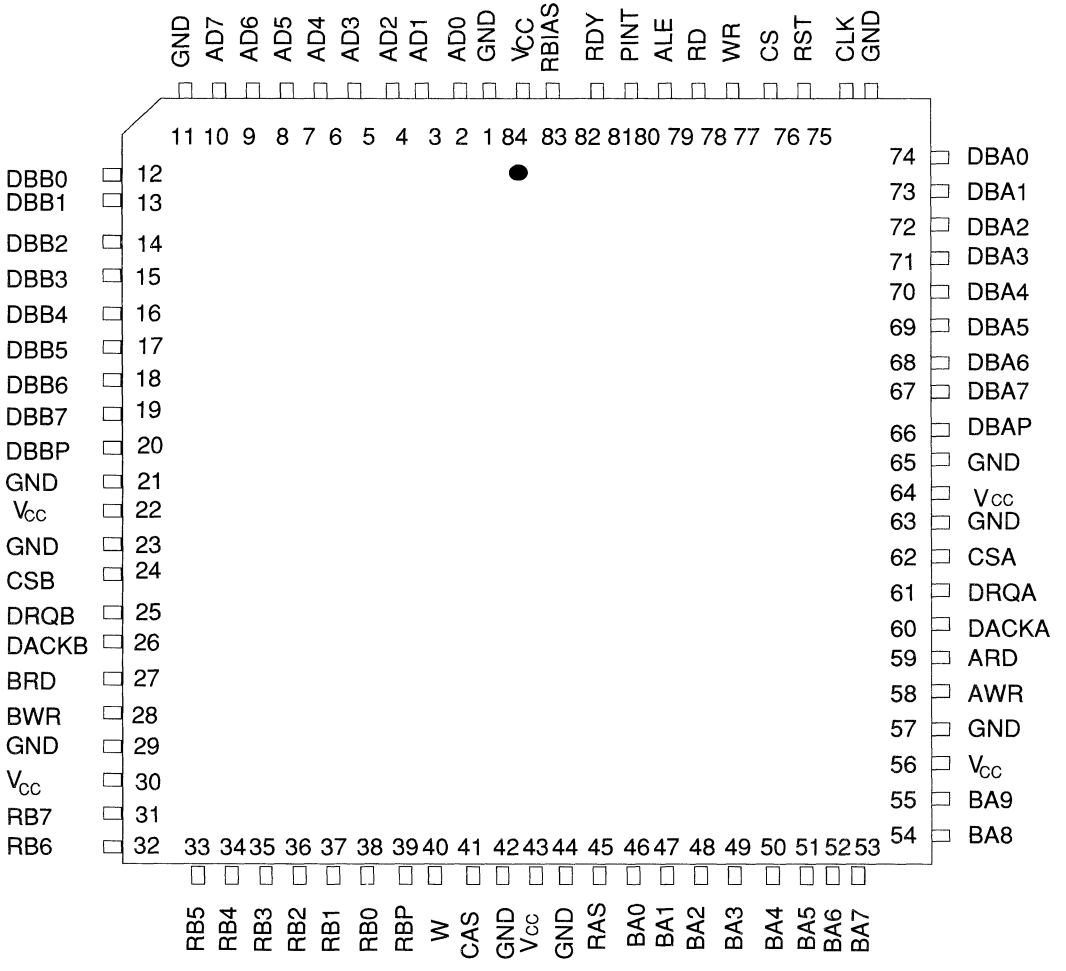


FIGURE 3. PIN DESIGNATION

35

3.0 NON-CHANNEL REGISTERS

The PCM was intended for use with the Intel 80186 processor and as such has been designed to interface easily with the 16 bit data bus of that processor. This is not to say that it cannot easily attach to other 8 or 16 bit microprocessors like Intel 8085, 8051, or 8096. The main requirement is that it be supplied with a multiplexed address/data bus with A7 to A0 multiplexed with D7 to D0 in that order.

As the PCM has only an eight bit data bus and it was decided not to handle the BHE signal, there are certain characteristics of the PCM that need explaining. First, all internal registers appear on even and the following odd byte address. This is because address signal A0 is not used inside of the PCM. The order of the registers was also arranged so that they appear in the order that they would normally be programmed. This allows the PCM to be attached to the 80186 and string move operations used to allow the fastest possible handling of loading the PCM.

Finally, the registers associated with the operation of the PCM itself do not decode the signal A7 so that they appear in both halves of the register map, facilitating an easier modulization of the driving firmware. Only the even addresses of the registers will be shown. Refer to page 10-35 for a summary of all the non-channel registers.

3.1 OPTION REGISTER

Address = 60H or E0H
Read/write register

7	6	5	4	3	2	1	0
MPAR	CAW1	CAW0	SRAM	RRC3	RRC2	RRC1	RRC0

The option register is a read/write only register to the firmware. The bits represent settings of options that must be selected to match the board design. There is the facility to read option switches and allow the firmware to assertain the hardware configuration in user 'adjustable' implementations. This register should be the first written after a reset sequence, and should not be written to thereafter unless another reset sequence has occurred.

3.1.1 RRC3 Thru RRC0 Refresh Rate Count Field

The refresh rate count field indicates to the PCM the prescale count to use for determing the refresh rate of the dynamic RAMs. The value of the refresh rate count field is effectively multiplied by 32, then subtracted from 512 to determine the number of clocks that will occur between refresh cycles. The value of this field is dependent upon the frequency of the PCMs clock. The following table shows the relationship of clock frequency, and refresh rate count field values, to produce 15.6 microsecond refresh cycles (for standard 128 cycle 2 ms. dynamic RAMs).

Field Value	Cycle Count	Freq. Range[1] Low	Freq. Range[1] High	Refresh Intrvl[2] Low	Refresh Intrvl[2] High
12	128	8.2	10.2	15.6	12.5
11	160	10.4	12.2	15.4	13.1
10	192	12.4	14.2	15.5	13.5
9	224	14.4	16.4	15.6	13.7
8	256	16.6	18.4	15.4	13.9
7	288	18.6	20.4	15.5	14.1
6	320	20.6	22.4	15.5	14.3
5	352	22.6	24.4	15.6	14.4

[1] Frequency ranges are in MHz.
[2] Refresh intervals are in µs.
NOTE:
The other field values are valid but are unlikely to be useful to the customer.

3.1.2 SRAM Static Ram Mode (bit 4)

The Static Ram Mode bit informs the PCM that the memory array is composed of static RAMs. When the bit is set, the PCM is in static RAM mode. This changes the signals that the PCM uses to access the memory array. Because of the multiplexed address bus of dynamic RAMs, it will be necessary for an external latch to be added to demultiplex the address lines. In addition, external address decoding will be necessary to expand the RAM to the full addressable capability. To assist in this, two of the RAM control signals change when the static RAM bit is set. First, the RAS signal is

used to control the external latch that is to demultiplex the address, when a valid row address is available on BA8 to BA0. RAS is intended to drive the clock input of a 74F373 type octal D-type latch, there is a direct analogy to the operation of dynamic RAMs at this point, with the exception that the row address latch is external to the memory devices. Secondly, static RAMs also have different read and write characteristics to dynamic RAMs. In dynamic mode, the W signal, signals the RAMs that a write cycle is about to occur, with the actual write action being generated by the falling edge of the CAS signal. This is known as early write mode. A read would occur if the CAS signal went active (low) with the W signal inactive (high). In static mode, the W and CAS signals change to support the separate WE and OE signals of static RAMs. In static mode, the CAS becomes the OE signal, still being active low but it will only occur on RAM read cycles, having the same shape and timing as for dynamic RAM cycles. The W signal becomes the WE signal, still being active low and only occuring in write cycles, but of a shape and timing similar to the CAS/OE signal. Finally, the arbitration logic no longer receives refresh requests as the static RAMs require no refresh, of course. At this stage it should be noted that the static RAM configuration is still liable to the row change overhead, when a row boundary is crossed during a burst of data cycles between the peripheral port FIFO and the memory.

The PCM is initialized to a special state at power up to ensure that there is no activity on the RAM control pins. This feature is used in the "power start" of the dynamic RAMs, which need to be left for a minimum time after power up. Please refer to the "Reset Sequences" section of this document for a description.

3.1.3 CAW1, CAW0 Column Address Width Field (bits 6,5)

The column address width field informs the PCM the type of RAM array that it is dealing with. The PCM uses the field to control the address multiplexer to the RAMs, and to determine the page mode boundaries of the RAMs. When a new page is detected, then the PCM executes a new RAS cycle, RAS goes inactive to precharge the RAS signal and then RAS goes active to latch the new row address and establish access in the new RAM page.

This field has no affect if the SRAM bit (bit 4) is set. When the SRAM mode is in effect, the PCM is forced to the ten bit column address mode.

00 = Ten bit column address (1meg DRAM). Bits 19 - 10 will be output on BA9 - BA0 during RAS time. Bits 9 - 0 will be output on BA9 - BA0 during CAS time.

01 = Nine bit column address (256K DRAM). Bits 18 - 9 will be output on BA9 - BA0 during RAS time. Bits 8 - 0 will be output on BA8 - BA0 during CAS time.

10 = Eight bit column address (64K DRAM). Bits 17 - 8 will be output on BA9 - BA0 during RAS time. Bits 7 - 0 will be output on BA7 - BA0 during CAS time.

11 = Six bit column address (16K DRAM). Bits 15 - 6 will be output on BA9 - BA0 during RAS time. Bits 5 - 0 will be output on BA5 - BA0 during CAS time.

3.1.4 MPAR = Memory Parity Enable (bit 7)

The parity enable bit informs the PCM that the memory array has a parity bit attached. When the bit is set, the PCM will generate odd parity (the sum of all one bits including the parity bit will be an odd number) on memory writes, and check parity on memory reads. If there is even parity on a memory read, the PCM will cause a parity error interrupt in the appropriate status register. When the bit is reset, the PCM will not generate parity on a write, or check parity on a read. The microprocessor firmware can use this bit to test the parity logic of the PCM, when there is parity memory. When the parity enable bit is reset, the RBP pin of the PCM will be at a logic one level during a write cycle.

3.2 OPTION REGISTER 2

Address = 62H or E2H
Read/write register
The option register 2 is initialized to zero by the PCM reset sequence

7	6	5	4	3	2	1	0
0	0	NOWAIT	WAITE	AHI	INTE	BINTE	AINTE

3.2.1 AINTE A Channel Interrupt Enable

The A channel interrupt enable bit allows interrupts from the "A" peripheral channel to exit the PCM by way of the PINT pin. This allows the programmer the capability to selectively disable interrupts from this channel while still allowing PCM interrupts from the other PCM resources. This bit only affects the transmission of the channels interrupt to the PINT pin, and does not affect the operation of any of the channels interrupt enables, or interrupt status flags.

3.2.2 BINTE B Channel Interrupt Enable

This bit performs the same function as AINTE, but for the "B" channel.

'A' CHANNEL	'B' CHANNEL	AHI	PRIORITY
Non-pausible	Pausible	X	A Channel
Pausible	Non-pausible	X	B Channel
Non-pausible	Non-pausible	1	A Channel
Non-pausible	Non-pausible	0	B Channel
Pausible	Pausible	1	A Channel
Pausible	Pausible	0	B Channel

3.2.3 WAITE Wait Enable

This bit is the master enable for any PCM wait condition to exit the PCM by the RDY pin. This allows the programmer the capability to selectively disable the RDY pin from the PCM. This bit only affects the transmission of the wait to the RDY pin, and does not affect the operation of any of the PCM wait sequences, or status.

3.2.4 NOWAT Non-waitable Microprocessor Interface (bit 5)

The "non-waitable" bit is used to distinguish to the PCM that the microprocessor or board design does not support the RDY function to extend bus cycles. This bit controls how the PCM will perform accesses to the data buffer and programmed I/O to the peripheral ports. This bit is reset by the PCM reset sequence.

When this bit is a zero, the PCM is to be used in a board with multiple devices sharing the microprocessor ready signal. The internal "ready" status of the PCM is gated with the CS signal, so only when the PCM is selected does it drive RDY. When the microprocessor accesses the data buffer or the peripherals for programmed I/O, then the PCM will assert RDY low to halt the bus cycle until it can perform the access. The PCM uses the leading edge of the data strobe to trigger the access, and if the firmware is writing data, then there is a specified time from the leading edge of the strobe that write data must become available within. When the access is complete, the PCM will deassert the RDY signal and allow the bus cycle to finish.

When this bit is a one, the PCM is used in a configuration with a microprocessor that does not support wait states, and the RDY signal is normally used as a status signal to a microprocessor's PIO input pin and polled by the firmware, or the internal status of the RDY pin is polled through the PCM task file register. In this configuration, the PCM will not gate the CS signal with the internal RDY status, so the RDY status will be available continuously. The PCM uses the trailing edge of the strobe to trigger the access, and as such there is no timing requirement on the write data from leading edge of strobe.

3.3 MASTER STATUS REGISTER

Address = 64H or E4H
Read/write register

7	6	5	4	3	2	1	0
DNR	0	0	PRNR	BANR	PPE	BINTR	AINTR

3.3.1 AINTR Channel A Interrupt Request Read only bit

3.3.2 BINTR Channel B Interrupt Request Read only bit

Setting AINTR (BINTR) indicates that the channel detected a situation requiring microprocessor action. The normal use of the interrupt is that the channel has completed an operation. The interrupt bit does not indicate if the operation was completed successfully, or if it terminated because of an error. The channel interrupt bit for each channel resets when the microprocessor clears all the interrupting states in the particular channel status register. This bit represents the 'or' function of all the interrupt conditions in the channel.

3.3.3 PPE Processor Parity Error

The processor parity error bit is set when a buffer read operation from the microprocessor interface resulted in a parity error. The bit is reset by the writing a one to the PPE bit. Writing a zero to PPE does not affect this bit.

3.3.4 BANR Buffer Access Not Ready Read only bit

The PCM "buffer access not ready" status bit is used to signal the microprocessor that the PCM is currently performing an access of the buffer and the registers that are associated with the buffer are not available. When the access is complete, this bit will be reset. See also the "DNR" bit in the master status register.

3.3.5 PRNR Power Reset Not Ready (bit 4) Read only bit

The PCM "power reset not ready" status bit is used to signal the microprocessor that the PCM is currently performing a power on reset sequence, or a programmed reset sequence, and all registers in the PCM except this register are not available. When the reset sequence is complete, this bit will be reset. See also the "DNR" bit in the master status register.

3.3.6 DNR Device Not Ready (bit 7) Read only bit

- The PCM "device not ready" status bit is used to signal the microprocessor that the PCM is currently performing a task that does not allow access to some resource in the PCM. The bit is essentially a logical 'or' of four conditions in the device that require multiple clock times to resolve themselves. The conditions are:

- the buffer manager reset sequence
- the microprocessor access of the buffer memory
- the microprocessor access of either of the peripheral ports for programmed I/O.

When the power is applied to the PCM, or firmware issues a soft reset, the bit will set indicating that all internal registers except this register are inaccessable. This bit will then reset when the PCM has completed its reset sequence. The "PRNR" bit in the master status register will also reflect this condition.

When the microprocessor accesses either of the peripheral ports for programmed I/O, or the microprocessor accesses the data buffer, the "device not ready" will reflect the status of the RDY pin of the PCM. This feature allows the firmware to 'poll' the PCM during these accesses when the board design or the microprocessor does not support the RDY function in hardware. The three accesses that cause this bit go to true are all independent, and can be occuring simultaneously. Because of the 'or' mechanism the bit will be false only when all accesses have completed.

When the microprocessor interface does support the RDY function, then there is no need to check this bit after the PCM has completed its reset sequence. When this bit is set because of RDY, the microprocessor is still in a wait state while the specific access is performed, and when the microprocessor is released from the wait state, then this bit is known to be reset.

3.4 BUFFER DATA LATCH

Address = 68H or E8H
Read/write register

7	6	5	4	3	2	1	0
BDL7	BDL6	BDL5	BDL4	BDL3	BDL2	BDL1	BDL0

The buffer data latch register holds the data last transferred between the microprocessor and buffer. The register is in the task file address space, so access of the register does not cause RDY to go false (no wait states). Access of the buffer data latch not cause the triggering of a buffer access.

The register is used when the board or microprocessor does not support the RDY function in hardware. Refer to the buffer access register description for specific details of the interaction of this register and the Buffer Access Register. The register is also useful for diagnostic purposes to test the PCM internal data path between the microprocessor and the internal task file.

3.5 MICROPROCESSOR ADDRESS POINTER

Address = 6EH or EEH (MP19 - MP16)
6CH or ECH (MP15 - MP08)
6AH or EAH 9MP07 - MP00)
Read/write register

7	6	5	4	3	2	1	0
0	0	0	0	MP19	MP18	MP17	MP16
MP15	MP14	MP13	MPM12	MP11	MP10	MP09	MP08
MP07	MP06	MP05	MP04	MP03	MP02	MP01	MP00

3.5.1 MP19 thru MP00 Microprocessor Address Pointer

The Microprocessor Address Pointer is a 20 bit register/counter that supplies the address lines to the buffer memory when the microprocessor is requesting a buffer data access. These registers may be read or written when the "BANR" bit in the master status register is reset, to show or set the current RAM window. If the microprocessor accesses the buffer through the Autoincrement Access Register, then this pointer will increment after the access has been performed.

3.6 TEST ADDRESS AND STATUS REGISTER

Address = 78H or F8H
Read/write register

7	6	5	4	3	2	1	0
TAS7	TAS6	TAS5	TAS4	TAS3	TAS2	TAS1	TAS0

The Test Address and Status register is a window into the internal logic of the PCM to allow greater visibility of the internal functioning of the device, and therefor greater test comprehensiveness.

The mechanization is that there is an address register in the PCM that can be loaded when a microprocessor write to this register occurs. This allows the selection of several internal registers and counters, and state machine registers, and the subsequent display of their content by reading this location.

This location also interacts with the Reset and Test register to invoke the test functions. Specifically there is an interlock so that test mode is more difficult to invoke, and less prone to be enabled accidently. The sequence to enable test mode is to write 80H to this register when all the test bits in the Reset and Test register are zero. This must be followed with a write to the Reset and Test register with the selected test mode (the soft reset bit must be zero). Then write F0H to this register. Once this procedure has been executed, then write the test address to this location, and read this location to view the desired internal status. When test mode is invoked, the only way to restore normal operation is to issue a soft reset, or assert the $\overline{RST}$ pin.

3.7 RESET AND TEST REGISTER

Address = 7AH or FAH
Write only register

7	6	5	4	3	2	1	0
SWRST	0	BRST	ARST	TSMEM	CNTRT	MTPBF	MTPAF

The Reset and test register allows a software reset of the PCM device. It is important that, during the normal operation, the value written to this register is 80H. This register is also used for setting test modes and a value in this register other than 80H causes the PCM to behave unpredictably to the user. The actions that occur when the register is written with 80H are identical to those that occur during power up, and the reset sequences that are described in the reset sequences section. This register is register is initialized to 00H when either the reset input pin (RST) is asserted, or the software reset function is invoked.

3.7.1 MTPAF Microprocessor to Port A FIFO Test (bit 0)

The microprocessor to port A FIFO test is used to verify the integrity of the internal data paths from/to the port FIFO, and to allow testing of the FIFO data cells. When this bit is set, the memory controller sets the data path from the memory side of the port FIFO to the buffer access register in the microprocessor section. The port should be in loopback mode, and when the microprocessor writes or reads the buffer access register the data will be transferred from/to the port FIFO instead of the external memory. Specifically, if the port has the DIR bit set so that data is to go from memory to the peripheral, then the microprocessor would write to the buffer access register, and the data would be read back from the FIFO through the channel data latch in loop back mode. The reverse direction has the microprocessor writing to the channel data latch, and reading the resultant data in the buffer access register.

3.7.2 MTPBF Microprocessor to Port B FIFO Test (bit 1)

This is identical to bit 0 but uses the B channel.

3.7.3 CNTRT Counter Test Mode (bit 2)

When this bit is set, the 16 and 20 bit counters will be divided along 4 bit boundaries, and the carry input to these boundaries will be forced true. The example is of the Buffer Address Pointer, if it is initially set to 12345H, when an access through the autoincrement access register is made, the counter will contain as its next value 23456H. The counters affected by this bit are:

- port A address pointer
- port A transfer counter
- port A EDAC idle counter
- port B address pointer
- port B transfer counter
- port B EDAC idle counter
- buffer address pointer
- refresh address counter.

3.7.4 TSMEM Tri-state Memory Interface (bit 3)

When this bit is set the control signals to the memory interface are forced to a tri-state mode, to allow testing of the memory devices by an external test machine.

3.7.5 ARST A Channel Reset (bit 4)

When written with a one causes the 'A' Channel to receive a reset. This reset does not affect the memory controller, or the other peripheral channel. The reset is removed when a zero is written into the bit, or a PCM reset sequence is initiated. When a channel is receiving a reset it will have its PNRA bit in its Channel Status Register set.

3.7.6 BRST B Channel Reset (bit 5)

This is identical to bit 4 but resets the 'B' Channel.

3.7.7 SWRST Software Reset (bit 7)

When written with a one causes initiation of the PCM reset sequence. This bit is self resetting when the PCM completes the reset sequence (denoted by the PRNR bit in the master status register being cleared), and initiates the PCM reset sequence on the trailing edge of the write strobe of this register.

3.8 BUFFER ACCESS REGISTER

Address = 7CH or FCH
Read/write register

7	6	5	4	3	2	1	0
BAR7	BAR6	BAR5	BAR4	BAR3	BAR2	BAR1	BAR0

The Buffer Access Register is the window that the microprocessor uses to load/store data to/from the buffer memory. The microprocessor address pointer supplies the address to/from which the RAM transfer will be made.

When the microprocessor and board design support the RDY function in hardware, (the NOWAT bit is zero), then this register passes the data through to/from the microprocessor and buffer. The leading edge of the microprocessors read or write strobe causes a "not ready" signal to the microprocessor, holding this state until memory arbitration and the actual data transfer has been accomplished. When the transfer is complete by RDY going true, then the cycle will end and the task is complete. There is no need of the buffer data latch when using the PCM in this mode.

In the case where the RDY function is not supported in the hardware, (the NOWAT bit is one), then the firmware must perform a program sequence to complete a transfer. If the microprocessor is writing data to the buffer, then it will write to this register and data will be latched along with the fact it is a write. The trailing edge of the strobe will cause the PCM to request arbitration for the buffer. The firmware is free at this time to perform other tasks, or poll the "DNR" status bit or the "BANR" in the Master Status Register, or poll the PCM RDY pin. When the access is complete then "not ready" will go false, and the firmware is again allowed to access the buffer.

If the microprocessor desires to read the buffer, then it must first read this register and discard the data. This action is only used to initiate the arbiter request, and latch the fact it is a read access. The trailing edge of the strobe will cause the buffer access to initiate. When the "not ready" condition goes false, then the data that was fetched is available in this register, and also in the Buffer Data Latch. Note that reading the Buffer Data Latch does not cause a "not ready" condition, or perform another access of the buffer, where if data is read from this register then another access will be initiated.

3.9 AUTOINCREMENT ACCESS REGISTER

Address = 7E or FE
Read/write register

7	6	5	4	3	2	1	0
BAR7	BAR6	BAR5	BAR4	BAR3	BAR2	BAR1	BAR0

The autoincrement data register performs the identical function of the buffer access register with the added feature that at the end of the data transaction, the microprocessor address pointer increments to the next address.

If the firmware desires to read or write a small block of data in the buffer, then access through this register will simplify the programming sequence. When the PCM is in the "waitable" microprocessor mode, then firmware need only continuously read or write data through this register. The transfer of data to/from the buffer occurs while the microprocessor is in a wait state, and the incrementing of the Microprocessor Address Pointer occurs in the intervening time between consecutive microprocessor bus cycles. If the PCM is in the "non-waitable" microprocessor mode, then firmware must poll the "not ready" status between accesses of this register. In this mode when reading a string of bytes from the buffer, if the last byte is accessed from this register, then another buffer access will occur, and the address pointer will be incremented. If this is not desired, then the last byte read from the buffer should be read from the buffer data latch instead of from this register.

The autoincrement access register is intended for use when small blocks of data are to be moved to/from the buffer while both ports are transferring data. If a large amount of data is to be moved to/from the microprocessor it is suggested that one of the ports be used in loopback mode, to achieve the maximum data transfer rate.

4.0 DEVICE CHANNELS

There are two device channels that the PCM controls. The channel registers are independent and identical across the two channels. Within the PCM the channels appear as 64 byte memory spaces in the chips total address space of 256 bytes. The channels are referred to as A, and B, with channel A being the 64 bytes of address space from 00H to 3FH, channel B being the 64 bytes of address space from 80H to BFH. Because the devices attached to the channels get their address lines (if necessary) directly from the microprocessor bus the even only address characteristic of the internal registers does not apply.

The registers that control the channels, are in the 32 bytes of address space that immediately follow the respective channels address space. The registers for channel A are in the address space of 40H to 5FH. The registers for channel B are in the address space of C0H to DFH. As these registers are internal, the odd addresses in the channel's respective address blocks are just a ghost image of the registers at the even addresses.

When the controller firmware accesses a register that controls a channel, the microprocessor will not have to wait for access, because these registers are all within the PCM chip, and are always available for immediate access. When the controller firmware accesses a register in a device on a channel, the PCM must evaluate the state and mode of the channel to determine if access of the register is possible. If the PCM is the bus master, then it will arbitrate the access of the channel, and will insert wait states for the microprocessor. The number of wait states for this access is dependant upon both the strobe timing of the channel, and the arbitration time of the channel. If the PCM is in slave mode it will not attempt the access but cause an I/O error interrupt and status to occur.

The channels function is to allow access between a peripheral device and the data buffer. The characteristics of any transfer have both fixed and

variable components. The fixed components are associated with the peripheral device and the hardware that will perform the transfer. These components are set at the beginning of the total transfer, and do not vary for the duration of the transfer. The variable components of the transfer are associated with the data buffer itself, and the arrangement of data, and the firmware control of the data. The fixed components are not 'pipelined', and are set by firmware at the start of a (group of) transfer(s).

The channel 'pipeline' mechanism allows firmware to control the data buffer on the fly. In applications where the firmware is attempting to either control a ring buffer or a cache, the mechanism allows the firmware to 'look ahead' to its next operation.

The registers have been organized in the order in which they would normally be programmed. It is especially important to ensure that the timing register is set up before the control register.

4.1 CHANNEL TIMING REGISTER

The channel timing register is used to control hardware actions that occur during a transfer to the external device. The action the register performs is that of controlling the pulse timing on the channel when the PCM is the bus master. This allows peripheral devices of various timing characteristics to be interfaced to the PCM. The register also defines to the channel logic specific characteristics of the external peripheral device, such as its handshake signal polarities, and parity capability.

The firmware is only allowed to write to the channel timing register when the channel is idle. The channel state machine will set the I/O error bit in the channel interrupt status register if the channel timing register is written while the channel is busy.

All bits in the register are cleared by the reset sequence.

Channel A Address = 40H
Channel B Address = C0H
Read/write register - not pipelined

7	6	5	4	3	2	1	0
LPBM	PPE	DKPL	RQPL	SDTC	DLY	SC1	SC0

NOTE
There is an A or B appended to each bit name corresponding to the appropriate channel.

4.1.1 SC0(1)A (SC0(1)B) Channel Strobe Control Field

The channel strobe control field is a two bit field that is used to control the width of the read and write strobes to the peripheral devices when the channel is in modes where the PCM is the bus master (DMA), or when doing PIO operations. The width of the strobes is programmable by setting the field. When the PCM is doing DMA cycles, for values of 0,1,2,3 the strobes will be 2,4,6,8 clocks wide respectively. When the PCM is doing PIO cycles, for values of 0,1,2,3 the strobes will be 4,6,8,10 clocks wide respectively.

4.1.2 DLYA (DLYB) Delay Strobe Bit

Normally, the $\overline{CS}$ and DACK signals go active two PCM clock before the $\overline{RD}$ or $\overline{WR}$ signals. When this bit is set, this period is extended by 2 PCM clocks. The primary use of this feature is for external decoding logic if multiple peripheral devices exist on the channel.

4.1.3 SDTCA (SDTCB) Strobe Deasserted Time Control Bit

This bit is relevant when the PCM is the bus master and in burst data transfer mode, then this bit is used to control the time that the RD and WR signals are deasserted. When this bit is zero, the RD and WR strobes will be deasserted for two PCM clocks during a data burst. When this bit is one, the RD and WR signals will be deasserted for 4 PCM clocks during a data burst.

4.1.4 RQPLA (RQPLB) DMA Request Polarity Bit

This bit controls the polarity of the PCM port DRQ pin. When this bit is set the PCM will consider the DRQ pin as active high. This is true whether this pin is receiving the request when the PCM is bus master or transmitting the request when the PCM is bus slave.

4.1.5 DKPLA (DKPLB) DMA Acknowledge Polarity Bit

This bit controls the polarity of the PCM port DACK pin. When this bit is set, the PCM will consider the DACK pin as active high. This is true whether this pin is transmitting the acknowledge when the PCM is bus master or receiving the acknowledge when the PCM is bus slave. Note that both the above situations occur when EDAC redundancy byte counting is enabled.

4.1.6 PPEA (PPEB) Port Parity Enable Bit

The device port can be optionally set to check the parity on transfers to the PCM from the peripheral device. Odd parity is supported and is generated when this bit is set. When this bit is set, all reads (data going into PCM) of the device channel by the PCM (either as master or slave) will result in parity checking being done, and interrupts being generated if even parity is detected. It should be noted that **ALL** transfers, including programmed I/O, are checked if this bit is set. When the PPE bit is reset, the port parity bit will be at a logic one level when data is transferring from the PCM to the peripheral device.

4.1.7 LPBMA (LPBMB) Loop Back Mode Enable

The purpose of this bit is to enable the microprocessor to use the FIFO to speed block transfers between the microprocessor and the buffer RAM. When set, this bit paths the channel data latch to the FIFO. It inactivates the outputs of the channel (CS, DACK, DRQ, WR, RD, and data bus). The microprocessor will appear to the channel as an external device and the channel will appear as if in slave mode. More explicitly the loopback mode sets the channel to a state of a non-pausible slave mode. In this state the microprocessor can transfer data to/from the FIFO at its maximum speed with the channel monitoring the FIFO for overrun/underrun conditions. The channel mode bits in the Channel Control Register need not be changed, but the Direction, and Interrupt Enable bits need to be programmed as if programming for an external transfer. In all other channel registers programming is as if for a normal external transfer, with the normal pipeline mechanisms in operation. Note the Direction bit in the Channel Control Register still needs to reflect the direction of transfer to/from an external device. As such the direction bit is SET for a transfer from the buffer to microprocessor (similar to a peripheral write command), though the microprocessor issues READ cycles to access the FIFO data.

4.2 CHANNEL CONTROL REGISTER

Channel A Address = 42H
Channel B Address = C2H
Read/write register - not pipelined

7	6	5	4	3	2	1	0
SLAV	BRST	DISK	EDAC	PAUS	DIR	IVE	IBE

NOTE: There is an A or B appended to to each bit name corresponding to the appropriate channel.

Where the Timing Control Register sets the physical environment of the interface with the external device, the Channel Control Register sets the logical environment of the interface. The register contains two types of parameters about the transfer being programmed. The high order 5 bits are used to identify the logical protocol that the PCM is going to use to interface to the external device. The low 3 bits are used to set a firmware environment to control the manbagement of the transfer direction and interrupts desired.

The firmware is only allowed to write to the chanbnel control register when the channel is idle. The channel state machine will set the I/O error bit in the channel interrupt status register if the channel c ontrol register is written while the channel is busy.

Whenever the firmware writes to the channel control register, the PCM will take 10 PCM clocks to synchronize the action to the PCM clock, and advance the channels status. During this time the firmware can only access non-channel registers.

The register is initalized to A8H by the rest sequence.

4.2.1 IBEA (IBEB) Interrupt on Busy Bit

The busy interrupt bit indicates to the channel that the microprocessor desires an interrupt when the channel very busy status bit transitions from a one to a zero.

4.2.2 DIRA (DIRB) Channel Transfer Direction Bit

The channel transfer direction bit is used to inform the PCM the direction of the transfer of this channel. The PCM uses this information to determine the read/write control of the buffer. When the transfer direction bit is a zero, the device is transferring data to the buffer (device read mode, buffer write mode). When the transfer direction bit is a one, the device is transferring data from the buffer (device write mode, buffer read mode).

The transfer direction bit is used by the channel logic to position the transfer counter, and the input of the FIFO to the source of the data. When the direction bit is a zero, (data transfer from the peripheral to the buffer), the transfer counter is monitoring the device handshake signals, and the count reflects the number of bytes that remain to be transferred from the peripheral device, to the input of the FIFO. When the transfer count exhausts and the FIFO goes empty, the logic will signal the end of this transfer if EDAC mode is not in effect. If EDAC mode is in effect, the channel will monitor the DACK pin to count EDAC trans fers, and when the EDAC idle counter exhausts and the FIFO is empty, it will signal the end of this transfer.

When the direction bit is a one, (data transfer from the buffer to the peripheral), the transfer counter is monitoring the buffer signals, and the count reflects the number of bytes that remain to be transferred from the buffer to the input of the

FIFO. When the transfer count exhausts and the FIFO goes empty, the logic will signal the end of this transfer if EDAC mode is not in effect. If EDAC mode is in effect when the FIFO goes empty, the channel will monitor the DACK pin to count EDAC transfers, and when the EDAC idle counter exhausts it will signal the end of this transfer.

4.2.3 PAUSA (PAUSB) Channel Pause Control Bit (bit 3)

The bit is used by the memory arbiter. When this port is pausible (this bit set), and the other port is non-pausible (this bit reset), then the other port

will have higher priority in the memory arbiter. If this bit is set the same in both channel control registers, then priority is determined by the "AHI" bit in the option 2 register.

This bit enables the "urgent request" logic. Because it controls priority, if the highest priority port's FIFO is near to overrun/underrun condition, then an "urgent request" is sent to the arbiter to stop the other port's burst and force arbitration.

4.2.4 EDACA (EDACB) EDAC Idle Enable (bit 4)

The EDAC Idle Enable bit is used to allow the port interface to share control of the peripheral bus with the WD60C80 EDAC device. The EDAC (Error Detection And Correction) is used in disk applications to append redundancy information to data blocks to allow error correction. The bit when set will allow the PCM to strip this information and not transfer it to the buffer on reads, and to allow the EDAC to append the information to the device data on writes. The EDAC bit is only relevant when the channel is programmed to be a bus master. The Channel EDAC Idle Counter contains a count of the number of redundancy bytes to ignore between data transfers. When this bit is set the transfer counter must be programmed for a single block size. When the transfer count exhausts the channel will tri-state the DACK pin and begin to count DACK pulses until the EDAC idle counter exhausts. While the EDAC idle counter is enabled, no transfers are allowed between the port and the FIFO (the redundancy bytes transfer between peripheral and the EDAC device). When the EDAC idle count exhausts the channel will then continue normal pipeline operations if the firmware has programmed them.

4.2.5 DISKA (DISKB) Disk Type Device (bit 5)

BRSTA (BRSTB) Burst Transfer Device (bit 6)

SLAVA (SLAVB) Slave Mode Interface (bit 7)

These three bits are used collectively to define to the port logic the type of peripheral device it will interface with, and the protocol to use with the interface signals. The signals will be described as per their individual significance, and then their collective significance. The names are derived from relationships they assume when the bit is set.

The SLAVE bit is used to instruct the channel that the external device is the bus master, and as such will control the $\overline{WR}$ and $\overline{RD}$ interface signals. When the bit is zero, the PCM is the bus master and controls the $\overline{WR}$ and $\overline{RD}$ signals. The master/slave relationship controls whether the PCM has the capability of performing programmed I/O on the external bus. When the microprocessor attempts to do programmed I/O and the channel is in a master and non-disk mode, the channel will arbitrate the external bus and interleave the programmed I/O with the DMA transfers the external device is requesting. When the channel is in slave mode, programmed I/O is rejected. If the microprocessor attempts programmed I/O an I/O error interrupt and status are asserted.

The BURST bit defines to the channel that the multiple bytes may be transferred between the external device and the FIFO with a single iteration of the protocol pins (DRQ and DACK). When the bit is a zero then SINGLE cycle mode, where an iteration of the protocol pins result in a single byte being transferred.

The DISK bit defines to the channel that the external device is a disk formatter device, and causes the protocol pins to maintain a different protocol than the normal REQUEST and ACKNOWLEDGE. When the bit is a zero then the normal DRQ and DACK protocol is enforced. Programmed I/O is inhibited when in a disk mode.

In the following text the three bits are grouped with the ordering of SLAVA (SLAVB), BRSTA (BRSTB), and DISKA (DISKB).

4.2.6 000 DMA Single Cycle Master

In this mode, the DRQ pin is the data request and is an input, the DACK pin is the data acknowledge and an output. In this mode, the PCM is con sidered to be the bus master, and the peripheral chip is the bus slave. The peripheral device uses the DACK signal to qualify the data strobes from the PCM. The WR pin is a low true output pin from the PCM, and is active when the PCM is programmed with a channel transfer direction bit set, and the peripheral device has requested service. The WR signal is used to clock data from the PCM to the peripheral device. The RD pin is a low true output pin from the PCM, and is active when the PCM is programmed with a channel transfer direction bit cleared, and the peripheral device has requested service. The RD signal is used to

clock data from the peripheral device to the PCM. The PCM will also use the WR and RD signals when the microprocessor requests programmed input/output of the peripheral device. At this time, the PCM will output the chip select signal to the peripheral device. The timing of the WR and RD signals is programmed in the timing control register of the channel.

In the Single Cycle DMA mode, the PCM will cycle the DACK signal for each byte transferred. The peripheral device can use the DACK signal to deassert its DRQ. This mode is designed to interface with older DMA style peripherals that interfaced with the Intel 8237 DMA chip. This mode will only transfer a byte when the channel FIFO is capable, therefore it is implied that overrun/underrun detection is the responsibility of the peripheral device. The channel will continue to transfer data if a parity error is detected, with the channel counters being captured at the time the error is detected. This mode is compatible with the EDAC device, and supports interleave of programmed I/O through the port and the DMA transfers.

4.2.7 001 DMA Single Cycle Disk

This mode is specifically for the ADS10C00 Disk Formatter Chip. The interface is the same as the DMA Single Cycle Master mode, in that the peripheral device requests service with the DRQ signal, and the PCM which is the bus master acknowledges the request with the DACK signal. The timing of the DACK signal is modified for the ADS10C00 requirements, with the leading edge of the DACK being returned asynchronously in response to the DRQ, and the data and trailing edge occuring synchronous to the PCM clock. In the DMA single cycle disk mode the WR and RD signals are not used. Instead the direction of data transfer is known to both the external device, and the PCM device, so the direction strobes are not required. In this mode the DACK signal is all that is required, as it acts both as an acknowledgement of data transfer, but also as the timing strobe. In this mode the PCM will attempt to transfer a byte on demand from the external device. If the channel FIFO becomes empty or full when another byte transfer is to take place, the channel will latch the data late occurance, and capture the channel counters for the firmware. The transfer will continue until the current transfer count is exhausted. If a parity error is detected during the transfer, the parity error status is latched along with the channel counters, and the transfer continues until the transfer counter exhausts. Since the ADS10C00 device supplies a seperate port for programmed I/O, then programmed I/O through the PCM port is not allowed in this mode. This mode is compatible with the EDAC device.

4.2.8 010 DMA Burst Cycle Master

This mode, tuned for the WD33C93 device, is the same as the DMA Single Cycle Master with the exception that at the end of the data strobe (either WR or RD) the DRQ signal is sampled, and if it is still active and the channel FIFO is capable of transferring another byte, the channel logic will keep the DACK signal asserted, and proceed to transfer another byte of data. This burst sequence will continue until either the external device deasserts DRQ, or the channel FIFO becomes unable to transfer another byte, or the microprocessor has requested a programmed I/O cycle on the external bus. The channel will continue to transfer data if a parity error is detected. This mode allows the highest data rate of the PCM bus master modes, but also requires the bus slave to deassert the DRQ in the shortest amount of time. When the PCM is programmed for strobe timing of 2 PCM clocks (SC1A(B), SC0A(B) = 00), and strobe deassert time of 2 PCM clocks (SDTCA(B) = 0) then the PCM is capable of transferring data at 1/4 the PCM clock rate. This yields a transfer rate of 6.25 Mbytes per second with a 25 MHz clock to the PCM. **(DRQ deassertion time needs to be strictly followed to guarantee reliable operation.)** Since the DACK signal does not change for multiple byte bursts, this mode is incompatible with the EDAC device.

4.2.9 011 Programmed I/O Mode

This mode is used to allow programmed I/O to occur to slave devices. Since it is a master mode then programmed I/O is enabled, but since it does not specify one of the specified interface protocols of bus masters, then it will not perform any peripheral transfers, and the DACK output will remain inactive at all times. When programmed I/O is to occur in a slave such as the WD50C1X configuration, then the firmware will set this mode while programmed I/O is to occur, and set the slave mode again after the programmed I/O is complete. Note that for DMA Single Cycle Master mode, and DMA Burst Cycle Master mode, that programmed I/O requests are interleaved with the data transfers, and there is no need to set Programmed I/O mode to do programmed I/O.

4.2.10 100 Single Cycle Slave Mode

The Single Cycle Slave Mode is similar to the DMA Single Cycle Master Mode, with the sense of the master/slave relationship reversed. In this mode the PCM is assumed to be attached to a host system bus as a DMA slave device. In this mode the sense of the DRQ and the DACK pins are reversed, in that the PCM makes the DRQ pin an output, and requests data from the bus master, and the DACK pin is used by the bus master to signal the PCM that the data transaction is to take place. In this mode it is the responsibility of the bus master to drive the data strobe lines during a data transaction. This configuration is targeted for the XT* environment, and the interface is consistent with the protocol of the Intel 8237 DMA chip, when programmed as it is in that environment.

4.2.11 101 Unused and Reserved Mode

4.2.12 110 Burst Cycle Slave Mode

This mode is similar to the Single Cycle Slave Mode, with the exception that the PCM will not deassert the DRQ output signal unless the FIFO does not have the capability of accepting another byte of data.

4.2.13 111 Slave Burst Mode Disk

In this mode, the connotation of the DRQ pin and the DACK pin change. The PCM is considered to be the bus slave, and the peripheral device is considered to be the bus master. The DRQ pin in this mode becomes an input, and monitors the BDRQ of the peripheral device. The DACK pin in this mode becomes an output, and signals the peripheral device when the peripheral device can continue with its next burst of data. For this discussion, the DACK pin will be referred to as BRDY. The WR pin and the RD pin in this mode are driven by the external bus master, the disk formatter. The protocol of the DRQ and BRDY signals is meant to accomodate the WD50C1X series of disk formatter devices. If the disk formatter is reading the disk then it will transfer a sectors worth data to the FIFO and set DRQ active. It will then wait for the PCM to assert BRDY to indicate that it can proceed with the next sector transfer. If the PCM exhausts its transfer count, and the pipeline register is empty, the PCM will not assert BRDY until the channel pointers have been set for the next transfer. If the PCM has not exhausted its transfer count, or continues in a pipeline operation, then it will send BRDY to the disk formatter to let it proceed with its transfer. If the disk formatter is writing to the disk, it will assert DRQ at the start of the transfer, and wait for BRDY. The PCM will assert BRDY when the channel is started and data is available in the FIFO. At the end of the sector transfer the disk formatter will again signal with DRQ if it desires to continue. If the PCM has the proper conditions to allow the next transfer it will again send BRDY. When programmed I/O is to occur to the WD50C1X device, the firmware must first set the channels mode to master for programmed I/O, then perform the programmed I/O, then return to this mode before starting the channel. If an error occurs (either parity error or data late error) during the transfer of data, the PCM will latch the errorstatus, and the channel counters. When the disk formatter again sends BDRQ the PCM will not send the BRDY signal, effectively halting the disk formatter from proceeding. The user can handle the error condition in one of two ways. The user can issue a stop to the channel, and clear the error, then restart the channel which will then allow the disk formatter to proceed when it sees BRDY. The user can supply external hardware to allow the firmware to reset the disk formatter, as the formatter chip is known to be in a state where write gate is not being asserted.

4.3 CHANNEL STATUS REGISTER

Channel A Address = 44H
Channel B Address = C4H
Read only register

7	6	5	4	3	2	1	0
0	0	DKST	RQST	PNR	FMT	VBSY	BSY

NOTE:There is an A or B appended to each bit name corresponding to the appropriate channel.

4.3.1 BSYA (BSYB) Channel Busy Status

The channel is currently engaged in a transfer.

4.3.2 VBSYA (VBSYB) Channel Very Busy Status

Specifically, this status bit indicates that the pipeline registers cannot be written into. One reason these registers are not available is because the channel is currently engaged in a transfer, and the pipeline registers of the transfer counter and the buffer pointers are currently loaded with the values of the next transfer.. The two status bits inform the firmware the state of the channel. When the two bits are both false, then the channel is currently idle, and the firmware can load the buffer pointer and transfer counter, and initiate an operation on the device. When the busy status is set, but the very busy status is false, the channel is in the busy state. In this state the pipeline registers are available for the firmware to load the values of the next transfer. When the firmware loads the pipeline registers and issues the start channel pulse, the channel will show very busy status. This statis is the very busy state and indicates to the firmware that the channel is not available for any new pointers.

4.3.3 FMTA (FMTB) Channel FIFO Empty Status (Bit 2)

Indicates the current status of the channel FIFO.

4.3.4 PNRA (PNRB) Port Not Ready Status (Bit 3)

The PCM "port not ready" status bit is used to signal the microprocessor that the PCM is currently performing a programmed I/O access of the peripheral, and the registers that are associated with the channels programmed I/O are not available. When the access is complete, this bit will be reset. See also the "DNR" bit in the master status register. This bit is also set if the channel is currently receiving a reset.

4.3.5 RQSTA (RQSTB) Channel DRQ Pin Status (Bit 4)

In order to allow the firmware to poll the state of devices while the channel is in slave mode, the state of the channel's DRQ pin is available on this bit. The DRQ is an input which has programmable polarity. This bit DOES NOT show the actual level of the DRQ pin but rather the logical state; set if DRQ is active, and reset if DRQ is inactive.

4.3.6 DKSTA (DKSTB) Channel DACK Pin Status (bit 5)

Indicates the logical state of the channel DACK pin.

4.4 CHANNEL INTERRUPT STATUS REGISTER

Channel A Address = 46H
Channel B Address = C6H
Read and write/clear bits

7	6	5	4	3	2	1	0
AERR	IOPE	IOE	REJ	LATE	PERR	VBI	BSYI

4.4.1 BSYIA (BSYIB) Interrupt From Busy

This status bit will be set when the busy interrupt bit in the channels control register (bit 0) is set, and the channel busy status bit transi tions from a one to a zero (the channel state transitions from busy to idle). This bit is cleared when a one is written to it.

4.4.2 VBIA (VBIB) Interrupt From Very Busy

This status bit will be set when the very busy interrupt bit in the channels control register (bit 1) is set, and the channel very busy status bit transitions from a one to a zero (the channel state transitions from very busy to busy). This bit is cleared when a one is written to it.

4.4.3 PERRA (PERRB) Channel Parity Error Bit

The channel parity error bit indicates to the firmware that the channel detected a parity error during a read of data either from the dynamic RAMs, or from the peripheral device. When the channel detects a parity error, it will interrupt the microprocessor, set the channel parity error flag. The transfer will continue until the end of the current block, at which time the PCM will wait for the firmware to clear the error and restart transfers.

When the transfer direction is from the buffer to the peripheral then the error occurred in the memory, and the firmware can determine the location of the parity error by reading the current value of the channel pointer, which was captured at the time of the error. When the transfer direction is from the peripheral device to the buffer, then the error occurred on the channel external interface, and the transfer counter will determine the exact byte in error. This bit is cleared when a one is written to it.

4.4.4 LATEA (LATEB) Channel Data Late Error Bit (bit 3)

The channel data late error bit indicates to the firmware that the channel detected a data overrun/underrun error during a transfer of data to/from the channel FIFO. When the channel detects a data late error, it will interrupt the microprocessor, set the channel data late error status. The detection of the overrun/underrun condition is disabled by the channels pause bit being set. This bit is cleared when a one is written to it.

4.4.5 REJA (REJB) Channel Command Reject Error Bit (bit 4)

The channel command reject error bit indicates to the firmware that there was an attempt to start a transfer when the channel had very busy status bit set, or without clearing the channel after an error. When the channel detects this condition it will interrupt the microprocessor, and set the command reject status. The transfer will not be affected. When the current transfer finishes the channel will not perform the next transfer in the pipeline because it is assumed that the pipeline register was corrupted previous to the erroneous start. No transfer will be initiated by the erroneous start command. This bit is cleared when a one is written to it.

4.4.6 IOEA (IOEB) I/O Error Bit (bit 5)

The channel I/O error bit indicates to the firmware that there was an attempt to do a programmed I/O cycle to a channel that was in slave mode, or the channel control register was written while the channel was busy. The I/O error does not halt a transfer in progress, or corrupt an external transfer. The status indicates to the microprocessor that the previous programmed I/O transaction did not transfer valid data, or the load of the channel control register did not take place. This bit is cleared when a one is written to it.

4.4.7 IOPEA (IOPEB) Programmed I/O Parity Error (bit 6)

The channel programmed I/O parity error bit indicates to the firmware that a parity error occurred during a programmed I/O read of the peripheral device. The detection of a programmed I/O parity error does not affect a transfer in progress if the programmed I/O cycle was interleaved with device data transfers in either the DMA Single Cycle Master mode, or the DMA Burst Cycle Master mode. This bit is cleared when a one is written to it.

4.4.8 AERRA (AERRB) Any Error Bit (bit 7)

The Any Error status bit is a read only bit, and is the logical 'or' of the five error status bits. It is provided for convenience to the firmware to easily determine if the previous operation was successful.

sistor through an option jumper to either V_{CC} or GND will passively pull the data line to the desired signal level. The PCM will latch the state of the data bus lines at the end of its reset sequence, so there is a minimum of 5 µs. plus 16 PCM clocks for the resistor to pull the signal to an acceptable level. The value of the resistor is derived from the reset sequence time, and the capacitance of the data lines. The maximum resistor value should be used to reduce the DC and AC loading of the data lines. Since the channel data latch is used for other functions, it is a requirement of the firmware that if the reset status is to be saved, that the firmware must read the channel data latch and save it in its local memory before attempting either programmed I/O to/from the external device, or to put the channel in loopback mode.

4.5 CHANNEL DATA LATCH

Channel A Address = 48H
Channel B Address = C8H
Read / write register

7	6	5	4	3	2	1	0
CDL7	CDL6	CDL5	CDL4	CDL3	CDL2	CDL1	CDL0

NOTE: There is an A or B appended to each bit name corresponding to the appropriate channel.

The Channel Data Latch holds the data associated with 3 different functions performed on the channel. The access of the register does not cause a wait condition.

The first function is that of the "Reset Status Latch." When the PCM is reset either by power being applied, or by writing into the PCM reset register, the PCM samples the channel bus and saves the data in the channel data latch for the firmware. The intent is that the board designer will install passive resistor networks that allow option jumper status to be reflected on the channel bus at reset time. There is an assumption here that the external device will tri-state the channel data bus bits during the power reset condition. With the PCM and the external device both tri-state, a resistor through an option jumper to either V_{CC} or GND will passively pull the data line to the desired signal level. The PCM will latch the state of the data bus lines at the end of its reset sequence, so there is a minimum of 5 µs plus 16 PCM clocks for the resistor to pull the signal to an acceptable level. The value of the resistor is dereived from the reset sequence time, and the capacitance of the data lines. The maximum resistor value should be used to reduce the DC and AC loading of the data lines. Since the channel data latch is used for other functions, it is a requirement of the firmware that if the reset status is to be savbed, that the firmware must read the channel data latch and save it in its local memory before attempting either programmed I/O to/from the external device, or to put the channel in loopback mode.

The second function of the channel data latch is for the loopback mode of the channel. In the loopback mode the microprocessor will read data from this latch, and write data to the latch without wait states. The channel logic will transfer the data between the latch and the FIFO, and check for overrun conditions. See also the "LPBMA" ("LPBMB") bit in the Channel Timing Register.

The third function is that of the port programmed I/O data latch. When the "RDY" function is supported in hardware, then programmed I/O proceeds through the normal address space reserved for external peripheral I/O. The leading edge of the microprocessors read or write strobe causes a "not ready" signal to the microprocessor, holding this state until arbitration for the peripheral port and the actual data transfer has been accomplished. When the transfer is complete by RDY going true, then the cycle will end and the task is complete. In this case this register is not used, though it still performs its intended function of capturing the programmed I/O data to/from the peripheral.

In the case where the RDY function is not supported in the hardware, then the firmware must perform a program sequence to complete a transfer. If the microprocessor is writing data to the port, then it will write to the programmed I/O data space, and data will be latched along with the fact it is a write. The trailing edge of the strobe will cause the PCM to request arbitration for the channel. The firmware is free at this time to perform other tasks, or poll the "DNR" status bit or the "PNRA" ("PNRB") in the Channel Status Register, or poll the PCM RDY pin. When the access is complete then "not ready" will go false, and the firmware is again allowed to access the channel for programmed I/O.

If the microprocessor desires to read the channel, then it must first read from the programmed I/O data space, and discard the data. This action is only used to initiate the arbiter request, and latch the fact it is a read access. The trailing edge of the strobe will cause the buffer access to initiate. When the "not ready" condition goes false, then the data that was fetched is available in this

register, and also in the programmed I/O data space. If the data is then read from the programmed I/O data space, then another programmed I/O cycle will be initiated, while if the data is read from this register, another programmed I/O access will not take place.

4.6 CHANNEL BUFFER POINTER

Channel A Address = 4EH (BP19 - BP16)
Channel A Address = 4CH (BP15 - BP08)
Channel A Address = 4AH (BP07 - BP00)
Channel B Address = CEH (BP19 - BP16)
Channel B Address = CCH (BP15 - BP08)
Channel B Address = CAH (BP07 - BP00)
Pseudo-read/write registers

7	6	5	4	3	2	1	0
0	0	0	0	BP19	BP18	BP17	BP16
BP15	BP14	BP13	BP12	BP11	BP10	BP09	BP08
BP07	BP06	BP05	BP04	BP03	BP02	BP01	BP00

NOTE: There is an A or B appended to each bit name corresponding to the appropriate channel.

4.6.1 BP19A (BP19B) through BP00A (BP00B) Buffer Pointer Address Bits

The channel buffer pointer is composed of three registers that contain the 20 bit memory address of the data. The actual hardware that mechanizes the buffer pointer is a presettable 20 bit counter, which is the actual pointer, a 20 bit holding register connected to the preset inputs of the counter (the pipeline register), and a 20 bit latch connected to the outputs of the counter for reading the counter. The microprocessor can actually only load the pipeline register from its data bus. When the firmware writes to any of the bytes in the buffer address pointer, a flag in the PCM is set to remember that the pipeline register was written. The channel state machine will use this flag to indicate that the buffer address pointer is to be loaded with a new value. When the channel state machine enters the "busy" state it will test and clear the flag, and load the counter from the holding register if the flag was set. See also the description of channel "states" in the Channel Start Register.

If the channel enters the "busy" state and the flag is cleared, indicating that firmware did not load the pipeline register, then the counter continues from its current position. The firmware then can mechanize a ring buffer system with only the firmware overhead of writing the starting address at the initial transfer, and then only issuing start commands for succeeding transfers. The data will then appear as a single large contiguous block. If data is to be transferred from discontiguous areas of the data buffer as in a cache buffer system, then the firmware has the added overhead of writing the next address to the pipeline register for each transfer segment that is not contiguous.

If the microprocessor desires to read the counter while it is active, the firmware must first write to the capture pointer register, which will make a copy of the counter at the time, and the microprocessor can then read the 3 bytes of address without any sampling errors. When the channel detects an error during a transfer it will copy the counter to the capture latch register for reading. When the channel transitions to the idle state, and the previous operation had no error, the address pointer is automatically copied to the capture latch register for reading.

4.7 CHANNEL TRANSFER COUNTER

Channel A Address = 52H (TC15 - TC08)
Channel A Address = 50H (TC07 - TC00)
Channel B Address = D2H (TC15 - TC08)
Channel B Address = D0H (TC07 - TC00)
Pseudo-read/write registers

7	6	5	4	3	2	1	0
TC15	TC14	TC13	TC12	TC11	TC10	TC09	TC08
TC07	TC06	TC05	TC04	TC03	TC02	TC01	TC00

4.7.1 TC15 through TC00 Transfer Counter Bits

The channel transfer counter appears as a read/write register, but in fact it is a write only register, and a read only counter. When the firmware writes to the transfer counter it is writing to a pipeline register. When the transfer counter is read, it is reading a capture latch that is loaded from the actual counter. The reason for this mechanization is that the pipeline register needs to hold the byte count of the transfer, and to load this value into the counter each time the channel begins an operation. This value only need change when the firmware requires a different granularity of data. When the channel detects an error, both the address pointer and the transfer counter are captured. The data that the firmware requires is how far into the transfer the operation proceeded before the error was detected. When the firmware reads the actual counter, the value is the residual count of the number of bytes that are left in the transfer, and from this information the firmware can deduce the number of bytes that transferred sucessfully. The firmware can also sample the current transfer counter by writing to the Channel Pointer Capture Register.

The channel transfer counter is composed of two registers that contain the 16 bit value that is the number of data bytes of the current transfer. The actual hardware that mechanizes the transfer counter is a presettable 16 bit down counter, which is the actual counter, and a 16 bit holding register which is the pipeline. The counter is unidirectional going from number of bytes that is desired to be transferred, to zero. The microprocessor can actually only load the pipeline register from its data bus.

When the Channel Start Register is set, and the channel is currently idle, then the channel state machine will load the transfer counter from the pipeline register, and start the operation. At this time the pipeline register is now free, and the firmware can load the transfer count of the next transfer into the pipeline register.

It should be noted here that for most devices that are envisioned, the block size that will be entered into the transfer counter, will be the sector (frame, etc.) size of the attached block oriented device and from that time the firmware will probably not load the transfer counter pipeline register. Hence if the firmware starts the channel again, the transfer counter will continually be updated with the block size of the device, without the firmware repeatedly loading the value on each transfer. The transfer count can be multiples of the device block size, if the firmware desires to cluster blocks together and view them as larger data items. A special case exists when EDAC Idle Enable control bit is set in the Channel Control register, in that the transfer counter is used to count the number of bytes between redundancy fields, and therefore can not be multiples of the block size.

4.8 CHANNEL START REGISTER

Channel A Address = 54H
Channel B Address = D4H
Write onlyregister

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Whenever the firmware writes to the channel start register, the PCM will take 10 PCM clocks to synchronize the action to the PCM clock, and advance the channels status. During this time the firmware can only access non-channel registers.

Under normal conditions the PCM will have the channel enabled before the device is armed for a transfer. The exception to this is likely to be a programmable device that requires the PCM to be the bus slave (eg. WD501X type devices). In that case the mode will have to be changed to slave after the firmware has set up the device, as there is no programmed I/O facility in slave mode. It should be noted that once the operation of the peripheral device has been started, it may assume master status at any time, and as such, can fight the PCM if the firmware does not enter slave mode before this happens. Also, delay in starting the transfer by writing to this register, may result in over/underrun errors in the peripheral device. Writing into the channel start register will produce varied responses from the channel, depending upon the current mode and state of the channel.

When the channel is in the idle state (busy and very busy status are both cleared), and the firmware writes into the channel start register, this indicates to the channel that the firmware desires the channel to begin operations. The channel state machine will load the transfer counter from the transfer counter pipeline register, and if the channel buffer address pointer pipeline has been written into, it will load the buffer pointer from the pipeline register, the flag for the address pointer will be reset, and channel busy status will set. When the busy status is set the pipeline registers are now available for the firmware to write the parameters of the next transfer.

When the channel is in the busy state (busy status set, and very busy status cleared), and the firmware writes into the channel start register, this indicates to the channel that the firmware desires the channel to continue operations when the current operation finishes. The channel very busy status will set. When the very busy status is set the pipeline registers are not available for the firmware to write. The channel is now considered to be in the very busy state. The channel will clear the very busy status and exit the very busy state when the current transfer completes. When the transfer completes the channel state machine will advance to the busy state and perform the same sequences that are described going from idle to busy. The channel counters are now set with values for the new transfer and the pipeline registers are again available for the firmware to queue the parameters of another transfer and issue another start.

When the channel is in the very busy state (busy and very busy status are both set), and the firmware writes into the channel start register, this will cause the channel to set the command rejected status and terminate operations on the channel when the current transfer completes. The channel state machine assumes that the firmware has corrupted the values in the pipeline when it issued the last start command. When the current operation completes the channel state machine will stay in the command reject error state until a channel stop is issued, which will then bring the state machine back to idle.

Finally, any attempt to start a channel with an uncleared error will result in a command reject error.

4.9 CHANNEL STOP REGISTER

Channel A Address = 56H
Channel B Address = D6H
Writeonly register

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Whenever the firmware writes to the channel stop register, the PCM will take 10 PCM clocks to synchronize the action to the PCM clock, and advance the channels status. During this time the firmware can only access non-channel registers.

4.10 CHANNEL POINTER CAPTURE REGISTER

Channel A Address = 58H
Channel B Address = D8H
Write only register

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

Whenever the firmware writes to the channel pointer capture register, the PCM will take 10 PCM clocks to synchronize the action to the PCM clock, and advance the channels status. During this time the firmware can only access non-channel registers.

When channel pointer capture register is written into, the capture latches for the channel pointer and channel transfer counter will sample the current state of the counters. This allows the microprocessor to detect the current position of the pointer and counter while they are active.

4.11 EDAC IDLE COUNTER

Channel A Address = 5AH
Channel B Address = DAH
Pseudo-read/write register

7	6	5	4	3	2	1	0
EC07	EC06	EC05	EC04	EC03	EC02	EC01	EC00

In order for the PCM to function with an external Error Detection and Correction device (EDAC), it is necessary for the PCM to be able to ignore (or "idle") while data transfers intended for or generated by the EDAC device transpire. To this end, each channel has an 8 bit down counter called the EDAC idle counter. It is mechanized by an 8 bit holding register that is loaded by the microprocessor with the number of bytes that should be ignored at the end of each block transferred, and an 8 bit down counter, decremented on the trailing edge of DACK once the PCM has finished its transfer. This counter is not latched like the address pointers and transfer counters, so any attempt to read this counter while it is running may result in synchronization errors. This value need only be set, to suit the EDAC device being used, after a reset condition. Thereafter the EDAC mode can be enabled and disabled using the EDAC Idle Enable Bit in the Channel Control Register.

5.0 MICROPROCESSOR INTERFACE

The PCM interface to the controller microprocessor is designed to be compatible with microprocessors of the multiplexed address/data bus design. It is specifically targeted to interface to Intel 80186 microprocessor with no external logic, but with some external logic the PCM can interface to other microprocessors that have multiplexed buses, or to non-multiplexed bus microprocessors. Speed penalties will exist for these implementations.

The PCM appears to the microprocessor as a contiguous address space of 256 bytes. The 8 bit address that decodes the unique location within the PCM is supplied by 8 multiplexed address/data pins. The address bit 0 is however ignored as all internal registers are mechanised to appear on even byte boundaries to suit the 80186. The PCM latches the address during the ADDRESS LATCH ENABLE signal. The 8 bits are used as both address information at ALE time, and data information at all other times. The PCM latches the bus address on all microprocessor cycles, but only responds to those cycles that are accompanied by the chip select signal. This signal is derived by the controller designer by doing a decode of the address information above bit A07, or using the already decoded outputs of the 80186. The controller designer has the flexibility to map the PCM's address space into the microprocessors memory space, or the I/O space, or a combination of both.

The PCM's interface includes a general interrupt signal to inform the microprocessor of changes of states of status within the PCM, and two signals to control the direction and timing of microprocessor bus cycles that involve the PCM. There are also two signals that allow the PCM to synchronize resource sharing for the microprocessor. When the PCM detects a bus cycle within its address range (chip select asserted), it decodes the address that it has latched, and determines

the resource that is to be accessed. The PCM has control of 4 general resources (internal registers, buffer memory, channel A, and channel B), and maps these resources into the address space.

The microprocessor interface deviates at this point depending upon the microprocessor that is interfaced, and the microprocessor and board design to support the READY/WAIT signal. In the case where the microprocessor and board design support the RDY function, the PCM uses the microprocessor RDY line to hold the microprocessor in a wait condition if the resource is not ready for immediate access. The internal register set of the PCM does not insert any wait states. The access of the other resources may or may not insert wait states.

If the microprocessor is accessing a device on one of the peripheral device channels, the local bus arbitrator for that channel will deassert RDY to the microprocessor until the channel is free for the microprocessor access to proceed. When the channel has been acquired the PCM will assert CSA or CSB (peripheral chip select) to select the peripheral device. When the access of the peripheral device is finished, the PCM will assert RDY, which will allow the microprocessor to proceed. If the access of the peripheral channel results in an I/O error, there will be no wait states inserted. If the microprocessor is accessing the buffer RAM, the memory arbiter will deassert RDY to the microprocessor until the buffer access has been done.

In the case where the microprocessor or the board design does not support the RDY function, the previously described actions to initiate the PCM function are identical. The difference is when the PCM deasserts the RDY function. In this case the microprocessor will not go into a wait condition, and is free to continue executing code. Internally, the PCM latches the access initiated, and the direction of the access, and proceeds to complete the access without holding the microprocessor. Data latches in each port and the buffer area will hold the data for the microprocessor while the access is in the arbitration queue. The microprocessor now has the responsibility of testing status (either in a PCM internal register, or the RDY signal), and when the access is complete, and to fetch the accessed data if the access was a read.

6.0 I/O MEMORY MAP

The PCM has within it four resources that the controller's firmware can access. The first resource is the PCM's register file. This consists of various registers that the firmware can read and write to both control the operation of the PCM, and sense the status of operations. The registers that are associated with a particular channel, are in an address space that is adjacent to the channel, and registers that are not associated with a particular channel use a partial decode of addressing so they are made to appear in both channels address space. Since these registers are dedicated to the microprocessor, there is no arbitration required to access these registers, and bus cycles to these registers involve no wait states.

The second resource that the PCM controls is the memory array. The memory array is made to appear to the firmware as a single memory location. The access to the actual memory location must be arbitrated along with accesses from the two peripheral channels, so access to the buffer memory may encounter some significant delays when there is heavy buffer traffic. The addressing of the memory array can either be static by loading the microprocessor page register, or can be made to autoincrement when searching through the data buffer.

The other two resources that the PCM controls are the device buses. The 2 device buses are made to appear as 64 byte spaces. Channel A devices appear in the offset range of 00H to 3FH. Channel B devices appear in the offset range of 80H to BFH. The actual registers that can be accessed through the channel, and the locations that they are made to appear at are under control of the board designer.

FCH	DATA BUFFER
E0H	BUFFER MGR. REGISTERS
C0H	CHANNEL B REGISTERS
80H	CHANNEL B
7CH	DATA BUFFER
60H	BUFFER MGR. REGISTERS
40H	CHANNEL A REGISTERS
00H	CHANNEL A

7.0 INTERRUPTS

All interrupt conditions within the PCM are grouped into a single interrupt line going out of the WD60C40. Individual interrupt conditions can be enabled/disabled within the PCM, but there is no prioritization of interrupts within the device. Interrupt conditions are reset when the firmware writes a one to the bit to clear the particular interrupt con dition.

8.0 RESET SEQUENCES

The PCM chip will be initialized when the $\overline{RST}$ pin is asserted, or the firmware writes a one into the soft reset bit of the PCM reset register. When the PCM is reset, it aborts all transfers, sets channel control registers to 00 and tri-states the port's control lines ($\overline{RD}$, $\overline{WR}$, DRQ, DACK), disables refresh by resetting the RAM control logic to a special state, and clears all interrupts. The firmware is responsible for setting up refresh and initializing the RAM to suit the dynamic RAM requirements. Resetting the PCM terminates access of the dynamic RAM array, allowing the idle time requirement of dynamic RAMs at power on. When the PCM option register is written to, this signals the PCM that the idle time requirement of the dynamic RAM array has passed, and the PCM will begin refresh cycles if refresh is enabled, it is the requirement of the controller firmware to delay the required idle time interval from power on condition before writing into the PCM's option register, to delay the required time interval from enabling refresh until the dynamic RAMs are ready for operation, and to initialize the memory array to proper parity. The PCM is initialized to static RAM mode by master reset. When the PCM is reset, the memory signals will be deasserted, and the channel control signals will all be tri-state. The memory signals will return to normal operation when the firmware writes into the option register. The channel signals will return to normal operation when the firmware writes into the channel control register. It should be noted that programmed I/O to the peripheral port can not be performed until the channel control register is written as master mode after a reset.

9.0 READY SEQUENCES AND LATENCIES

The RDY (ready) signal will be asserted low when the PCM has an internal resource that is unavailable to the firmware. Then time that the signal will be asserted depends upon the mode that the microprocessor interface is in, and the resource that is being accessed. When the PCM is in the 'waitable' microprocessor mode, the RDY signal will assert low when the internal address decode with CS selects a resource. The resource will start its access 3 PCM clocks after the microprocessor's data strobe asserts. When the PCM is in the 'non-waitable' mode, the RDY signal will assert low 30 ns. after the microprocessor's data strobe ends. The resource will start its access after the 3 PCM clocks for synchronization. These times are in addition to the following discussion which describes the components that determine the time of the particular access.

When the firmware accesses the buffer, the time the access takes is dependant upon the microprocessor's placement in the memory arbitration scheme, and any latency associated with its request occuring while a port data burst is in progress. The microprocessor is the highest priority device in the memory arbitration scheme, since it is a single cycle device, and its request period can be controlled by the firmware. The latency associated with a port data burst is found from the equation:

TBRST = 2 (N) + 4 (P+1) expressed in PCM clocks

N = number of bytes in the ports burst

P = number of RAS cycles occuring during the burst, or # of page boundaries crossed during burst.

Example

If the port transfers 22 bytes during the burst, and crosses a page boundary, then the total burst will require 52 PCM clocks. For a PCM clock frequency of 16 Mhz. this will be a time of 3.25 microseconds.

The N factor in the above equation depends upon the number of bytes in the FIFO when the burst begins, and the memory transfer rate, and the peripheral transfer rate. The value of N should be

determined by simulation but an estimation can be made from the equation:

N = MTR (IFBC) / (MTR - PTR)

MTR = memory transfer rate, 1/2 PCM clock frequency

PTR = peripheral transfer rate, (in MHz)

IFBC = initial FIFO byte count at start of burst

NOTE:
IFBC is in the range of 0 to 15 for WD60C40

The memory transfer rate when the port is in a burst is 2 PCM clocks per byte. If the PCM is supplied with a 16 MHz clock, then the memory trans fer rate will be 8 MHz (8 Mbytes/second). The peripheral transfer rate is supplied in the same terms as the memory transfer rate. As an example the WD33C93A SCSI interface peripheral is capable of a transfer rate of 4 Mbytes/second, or 4 MHz. The FIFO size in the PCM is 15 bytes. If these values are applied to the equation, and use the value of 14 bytes for the initial FIFO condition, the result is 29 bytes for the expected burst. At the memory transfer rate of 8 Mbytes per second, this would yield a burst time of 3.625 microseconds.

It can be seen that as the peripheral transfer rate approaches the memory transfer rate, the denominator of the equation approaches zero, and the size of the burst increases rapidly. Therefore if the PCM is designed into a system with the peripheral transfer rate very close to the memory transfer rate, large microprocessor buffer access times can be expected. The size of burst is limited by the refresh mechanism of the PCM. The PCM will queue up to 4 refreshes, and then force an ar bitration, which the microprocessor will win because it is highest priority. In a system that has refreshes programmed at the normal rate of 15.6 microseconds, this effectively limits the maximum burst and the maximum latency to about 62 microseconds.

10.0 NON-CHANNEL REGISTERS MAP

7	6	5	4	3	2	1	0	REGISTER/ ADDRESS
MPAR	CAW1	CAW0	SRAM	RRC3	RRC2	RRC	RRC0	OPT 60H or E0H
0	0	NOWAIT	WAITE	AHI	INTE	BINTE	AINTE	OP2 62H or E2H
DNR	0	0	PRNR	BANR	PPE	BINTR	AINTR	MSR 64H or E4H
0	0	0	0	0	0	0	0	66H or E6H
BDL7	BDL6	BDL5	BDL4	BDL3	BDL2	BDL1	BDL0	BDL 68H or E8H
MP07	MP06	MP05	MP04	MP03	MP02	MP01	MP00	MAP 6AH or EAH
MP15	MP14	MP13	MP12	MP11	MP10	MP09	MP08	6CH or ECH
0	0	0	0	MP19	MP18	MP17	MP16	6EH or EEH
TAS7	TAS6	TAS5	TAS4	TAS3	TAS2	TAS1	TAS0	TAS 78H or F8H[1]
SWRST	0	BRST	ARST	TSMEM	CNTRT	MTPBF	MTPAF	RTR 7AH or FAH
BAR7	BAR6	BAR5	BAR4	BAR3	BAR2	BAR1	BAR0	BAR 7CH or FCH
BAR7	BAR6	BAR5	BAR4	BAR3	BAR2	BAR1	BAR0	AAR 7EH or FEH

NOTES
All "odd" addresses willl access the "even" address that is one lower.
[1]Contents of addresses 70H (or F0H) through 76H (or F6H) must be written as 00 and read as 00.

35

11.0 CHANNEL REGISTERS MAP

7	6	5	4	3	2	1	0	REGISTER/ ADDRESS
LPBM	PPE	DKPL	RQPL	SDTC	DLY	SC1	SC0	CTR 40H or C0H
SLAV	BRST	DISK	EDAC	PAUS	DIR	IVE	IBE	CCR 42H or C2H
0	0	DKST	RQST	PNR	FMT	VBSY	BSY	CSR 44H or C4H
AERR	IOPE	IOE	REJ	LATE	PERR	VB1	BSY1	ISR 46H or C6H
CDL7	CDL6	CDL5	CDL4	CDL3	CDL2	CDL1	CDL0	CDL 4BH or C8H
BP07	BP06	BP05	BP04	BP03	BP02	BP01	BP00	CBP 4AH or CAH
BP15	BP14	BP13	BP12	BP11	BP10	BP09	BP08	4CH or CCH
0	0	0	0	BP19	BP18	BP17	BP16	4EH or CEH
TC07 TC15	TC06 TC14	TC05 TC13	TC04 TC12	TC03 TC11	TC02 TC10	TC01 TC09	TC00 TC08	CTC 50H or D0H 52H or D2H
0	0	0	0	0	0	0	0	CST 54H or D4H
0	0	0	0	0	0	0	0	CSP 56H or D6H
0	0	0	0	0	0	0	0	CCP 58H or D8H
EC07	EC06	EC05	EC04	EC03	EC02	EC01	EC00	CEC 5AH or DAH
0	0	0	0	0	0	0	0	Reserved 5CH or DCH
0	0	0	0	0	0	0	0	Reserved 5EH or DEH

NOTE:
There is an A or B appended to each bit name corresponding to the appropriate channel.

12.0 ELECTRICAL CHARACTERISTICS

PARAMETER	SYMBOL	MIN	MAX	UNIT
Power Supply	V_{CC}	4.5	5.5	V
Input Low Voltage	V_{IL}		0.8	V
Input High Voltage	V_{IH}	2.0		V
Ouptut Low Voltage I_{OL} = 2.0 mA	V_{OL}		0.4	V
Output High Voltage, IOH = 400 µA	V_{OH}	2.8		V
Leakage Current Low	I_{LL}		10	µA
Leakage Current High	I_{LH}		-10	µA
Supply Current	I_{CC}		100	mA
Power Dissipation	PD		500	mW

The output loading depends on the pin function and are as follows:

IOL	IOH	SIGNAL NAME
6 mA	-2.5mA	AD7 to AD0
2 mA	-1.0 mA	CSA, CSB DRQA, DRQB DACKA, DACKB ARD, BRD AWR, BWR DBA7 to DBA0, DBAP DBB7 to DBB0, DBBP RB7 to RB0, RBP BA9 to BA0 RAS, CAS, $\overline{W}$
6 mA	O.D.	RDY PINT

35

13.0 MISCELLANEOUS CHARACTERISTICS

Operating Temperature	0^{o} to 70^{o} C
Absolute Maximum Ratings	All voltages referenced to V_{SS}
V_{CC}	7.0 Volts
Voltage to any pin	-0.3 to V_{CC} + 0.3 Volts
Storage Temperature	-40^{o} to $+125^{o}$ C

NOTE:
Maximum limits indicate the point where permanent device damage occurs. Continuous operation at these limits is not intended, and should be limited to those conditions specified in Electrical Characteristics sections.

14.0 TIMING SPECIFICATION

In these timing diagrams the following assumptions have been made:

(1) As port B is identical to port A, only port A timings need be shown.

(2) The DRQA signal has been programmed as active HIGH.

(3) The DACKA signal has been programmed as active LOW.

(4) The timing specifications assume the following loading on each pin.

MAX LOADING	PIN NAMES
120 PF MAX	BA9 to BA0 RB7 to RB0, $\overline{RBP}$ RAS, CAS, W
100 PF MAX.	AD7 to AD0
50 PF MAX.	CSA, CSB DACKA, DACKB ARD, BRD AWR, BWR DBA7 to DBA0, DBAP DBB7 to DBB0, DBBP RDY PINT

14.0 TIMING DIAGRAMS

The timing diagrams use the following nomenclature:

REF

The Reference number on the timing waveform diagram.

NAME

Abbreviated Symbol by which the timing is referred.

DESCRIPTION

Description of the timing referred.

REFERENCE

The reference edge to which the timing is specified.

LE = Leading edge

TE = Trailing edge

RE = Rising edge

FE = Falling edge

IC = Initial Condition

R/S

Classification of requirement/specification.

R = Requirement of the external circuit.

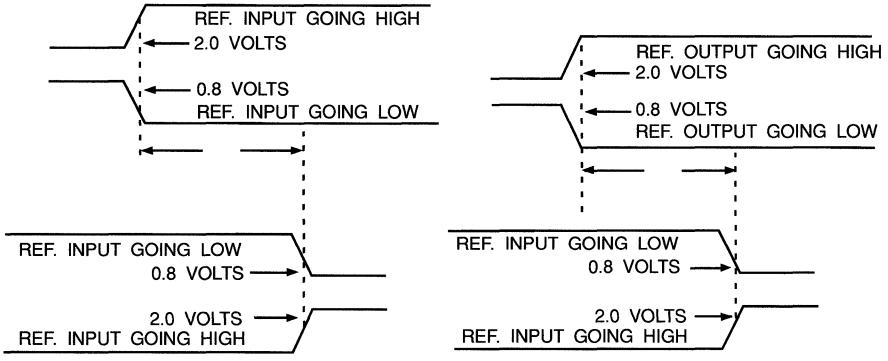
S = WD60C40 output timing specification.

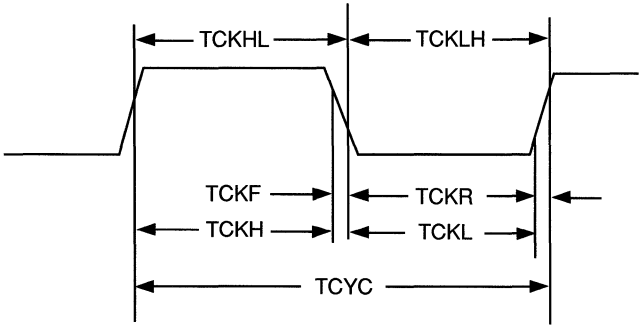
TIMING

Value, unit, and characteristic of timing.

AC TIMINGS

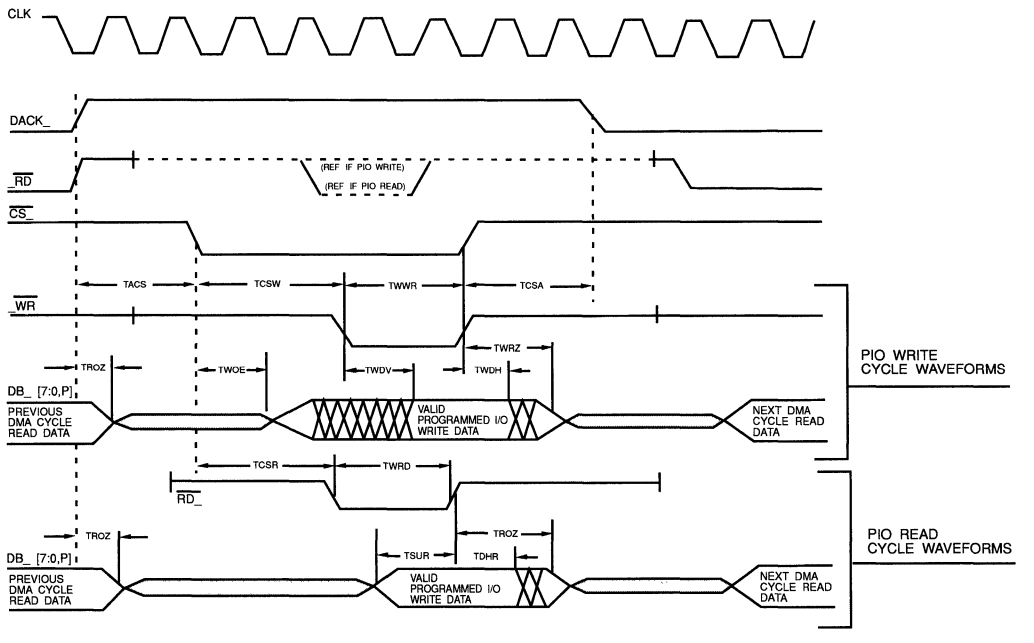
** RAM PORT VIL = 0.8 VOLTS, VIH = 2.4 VOLTS
VOL = 0.8 VOLTS, VOH = 2.4 VOLTS





NAME	REF	R/S	MINIMUM	MAXIMUM
TCYC = Input Clock Period		R	40 NS	
TCKL = Input Clock Low		R	17 NS	
TCKH = Input Clock High		R	17 NS	
TCKR = Input Clock Rise		R		5 NS
TCKF = Input Clock Fall		R		5 NS
TCKHL = Input Clock High to Low		R	20 NS	
TCKLH = Input Clock Low to High		R	20 NS	

FIGURE 4. RISE/FALL AND MISC. TIMING

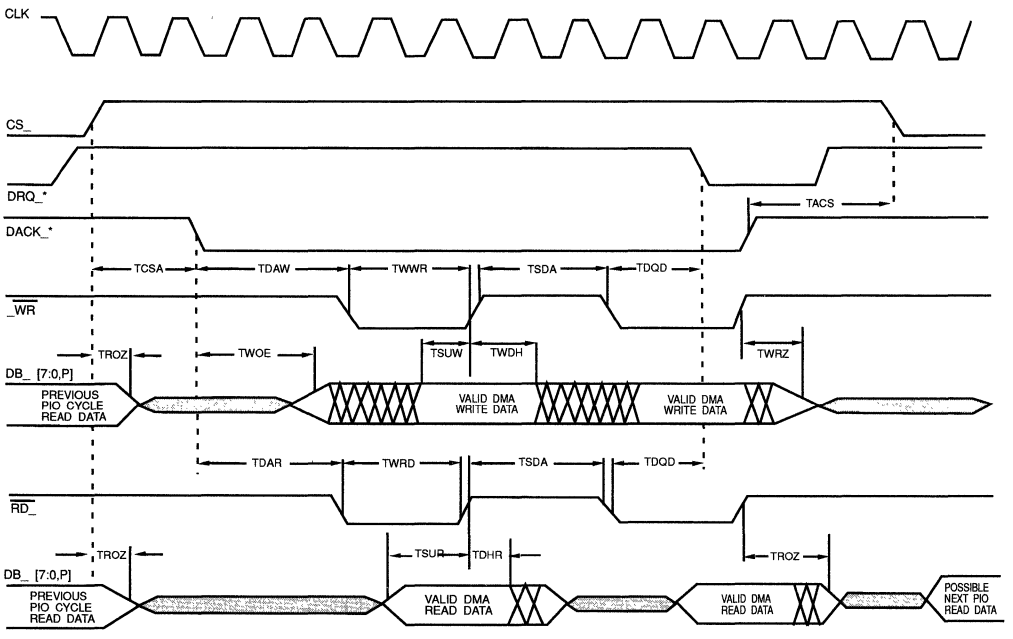


NAME	REF	R/S	MINIMUM	MAXIMUM
TACS = Time of DACK to Chip Select	TE	S	2 TCYC - 15 NS **	
TCSA = Time of Chip Select to DACK	TE	S	2 TCYC - 15 NS **	
TCSW = Chip Select to Write Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWWR = Width of Write Strobe (See SC in Section 4.1)		S	4 TCYC - 15 NS	10 TCYC + 15 NS
TWOE = Time to Output Enable in Write	LE	S	0 NS	
TWDV = Time to Data Valid in Write	LE	S		40 NS
TWDH = Time of Data Hold in Write	TE	S	TCKH - 5 NS	
TWRZ = Time of Write Strobe to High Z	TE	S	TCKH - 5 NS	TCKH + 20 NS
TCSR = Chip Select to Read Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWRD = Width of Read Strobe (See SC in Section 4.1)		S	4 TCYC - 15 NS	10 TCYC + 15 NS
TSUR = Data Setup Time of Read Data	TE	R	30 NS	
TDHR = Data Hold Time of Read Data	TE	R	0 NS	
TROZ = Time of Read Strobe to High Z	TE	R		2 TCYC **

** If SDTC in Section 4.1 is set, then value is 4 TCYC

FIGURE 5. PIO READ/WRITE OF EXTERNAL DEVICES

35



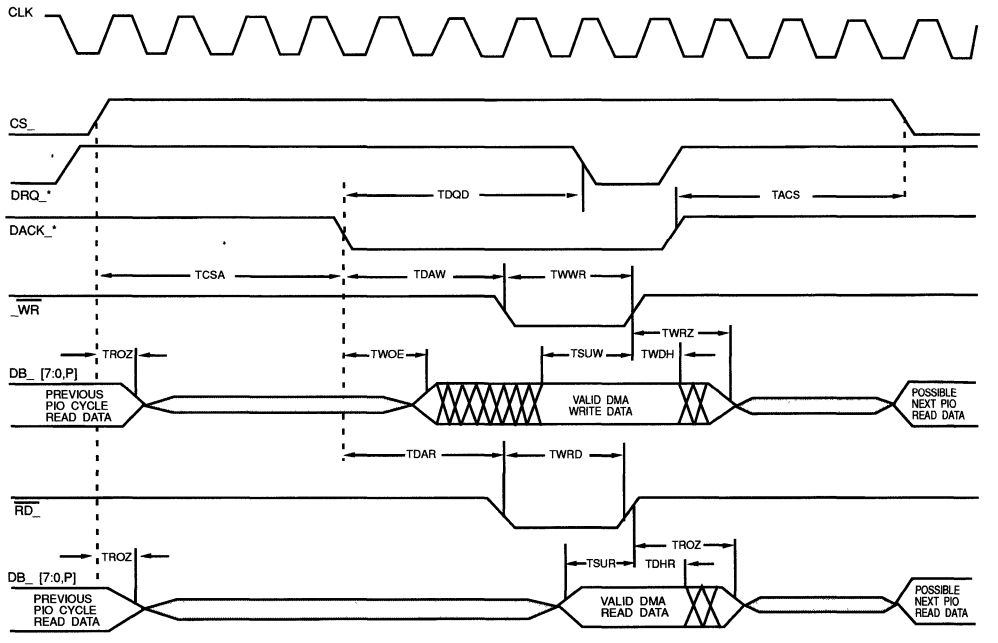
NAME	REF	R/S	MINIMUM	MAXIMUM
TACS = Time of DACK to Chip Select	TE	S	2 TCYC - 15 NS **	
TCSA = Time of Chip Select to DACK	TE	S	2 TCYC - 15 NS **	
TDAW = Time DACK to Write Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWWR = Width of Write Strobe (See SC in Section 4.1)		S	2 TCYC - 15 NS	8 TCYC + 15 NS
TWOE = Time to Output Enable in Write	LE	S	0 NS	
TSUW = Data Setup Time to Write	TE	S	2 TCYC - 40 NS	8 TCYC + 15 NS
TWDH = Time of Data Hold in Write	TE	S	TCKH - 5 NS	
TWRZ = Time of Write Strobe to High Z	TE	S	TCKH - 5 NS	TCKH + 20 NS
TDAR = Time DACK to Read Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWRD = Width of Read Strobe (See SC in Section 4.1)		S	2 TCYC - 15 NS	8 TCYC + 15 NS
TSUR = Data Setup Time of Read Data	TE	R	30 NS	
TDHR = Data Hold Time of Read Data	TE	R	0 NS	
TROZ = Time of Read Strobe to High Z	TE	R		2 TCYC **
TDQD = Time to DRQ Deasserted (to stop Burst)	LE	R		2 TCYC - 20 NS ***
TSDA = Time of Strobe Deactivated (See SDTC in Section 4.1)		S	2 TCYC - 15 NS	4 TCYC + 15 NS

* DRQ Polarity shown high true; DACK polarity shown low true.

** If SDTC in Section 4.1 is set, then value is 4 TCYC.

*** Based on SC and SDTC set to zero. Add 2 TCYC for each programmed state.

FIGURE 6. DMA BURST MODE TRANSFERS



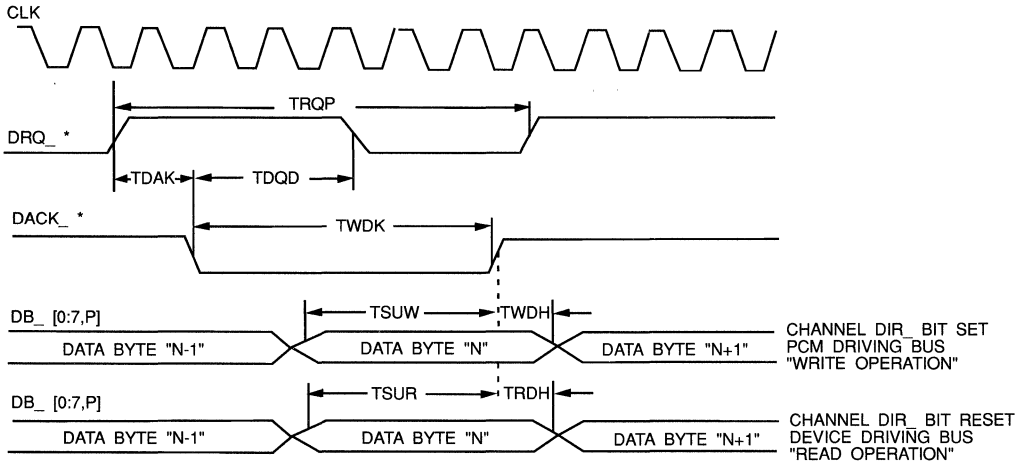
NAME	REF	R/S	MINIMUM	MAXIMUM
TACS = Time of DACK to Chip Select	TE	S	2 TCYC - 15 NS **	
TCSA = Time of Chip Select to DACK	TE	S	2 TCYC - 15 NS **	
TDAW = Time DACK to Write Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWWR = Width of Write Strobe (See SC in Section 4.1)		S	2 TCYC - 15 NS	8 TCYC + 15 NS
TWOE = Time to Output Enable in Write	LE	S	0 NS	
TSUW = Data Setup Time to Write	TE	S	2 TCYC - 40 NS	8 TCYC + 15 NS
TWDH = Time of Data Hold in Write	TE	S	TCKH - 5 NS	
TWRZ = Time of Write Strobe to High Z	TE	S	TCKH - 5 NS	TCKH + 20 NS
TDAR = Time DACK to Read Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWRD = Width of Read Strobe (See SC in Section 4.1)		S	2 TCYC - 15 NS	8 TCYC + 15 NS
TSUR = Data Setup Time of Read Data	TE	R	30 NS	
TDHR = Data Hold Time of Read Data	TE	R	0 NS	
TROZ = Time of Read Strobe to High Z	TE	R		2 TCYC **
TDQD = Time to DRQ Deasserted (to stop Burst)	LE	R		2 TCYC - 20 NS ***
TSDA = Time of Strobe Deactivated (See SDTC in Section 4.1)		S	2 TCYC - 15 NS	4 TCYC + 15 NS

* DRQ Polarity shown high true; DACK polarity shown low true.

** If SDTC in Section 4.1 is set, then value is 4 TCYC.

*** Based on SC and SDTC set to zero. Add 2 TCYC for each programmed state.

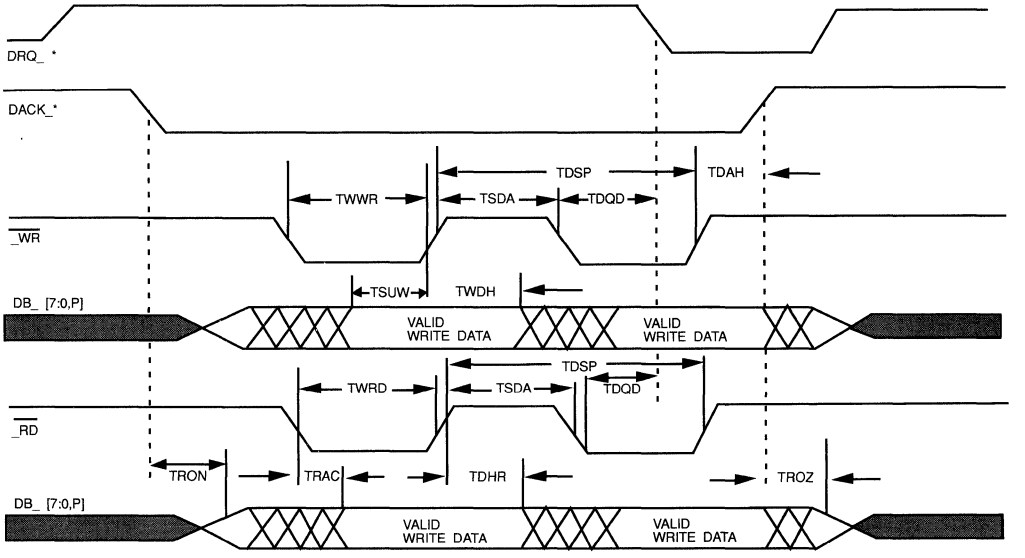
FIGURE 7. DMA SINGLE CYCLE MODE



NAME	REF	R/S	MINIMUM	MAXIMUM
TDAK = Time DRQ to Acknowledge	LE	S		35 NS
TDQD = Time to DRQ Deasserted	LE	R		3 TCYC
TRQP = Time of Request Period		R	6.5 TCYC	
TWDK = Width of DACK Signal		S	3.5 TCYC	5.5 TCYC + 15 NS
TSUW = Data Setup Time of Write Data	TE	S	2 TCYC - 40 NS	
TWDH = Data Hold Time of Write Data	TE	S	TCKH - 5 NS	
TSUR = Data Setup Time of Read Data	TE	R	30 NS	
TDHR = Data Hold Time of Read Data	TE	R	0 NS	

* DRQ polarity shown high true; DACK polarity shown low true
** If SDTC in Section 4.1 is set, then value is 4 TCYC
*** Based on SC and SDTC set to zero. Add 2 TCYC for each programmed state

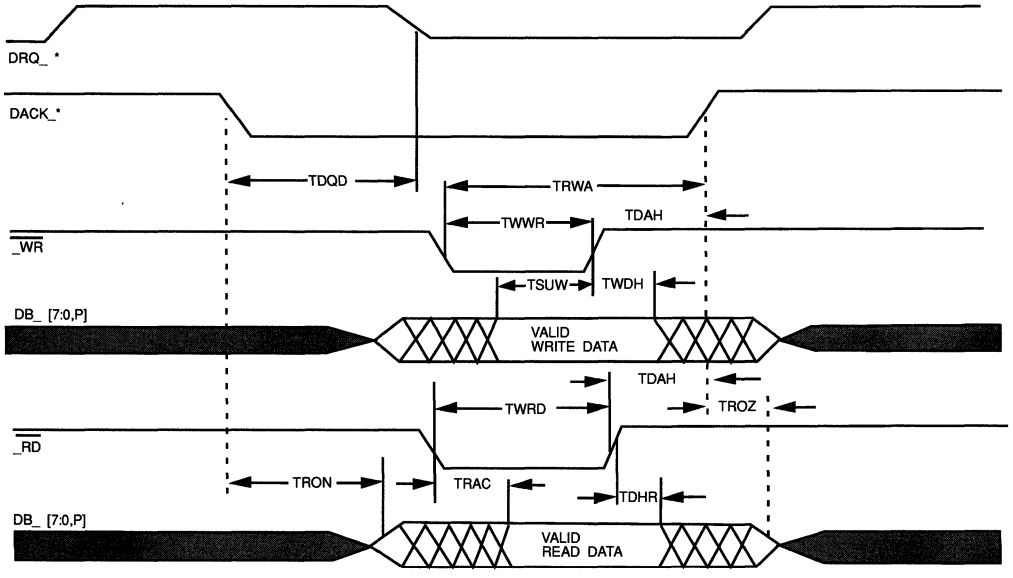
FIGURE 8. BUS MASTER DISK MODE



NAME	REF	R/S	MINIMUM	MAXIMUM
TWWR - Width of Write Strobe		R	50 NS	
TSDA = Time of Strobe Deactivated		R	50 NS	
TDQD = Time to DRQ Deasserted	LE	S		35 NS
TDSP = Time of Data Strobe Period		R	2.5 TCYC	
TDAH = Time of DACK Hold	TE	R	0 NS	
TSUW = Data Setup Time to Write	TE	R	20 NS	
TWDH = Time of Data Hold in Write	TE	R	10 NS	
TRON = Time Read Output Enable from DACK (If DIR bit is set)	LE	S	0 NS	50 NS
TWRD = Width of Read Strobe		R	50 NS	
TRAC = Data Access Time	LE	S		45 NS
TDHR = Data Hold Time of Read Data		S	10 NS	45 NS
TROZ = Time of Read Strobe to High Z		S		50 NS

* DRQ polarity shown high true; DACK polarity shown low true

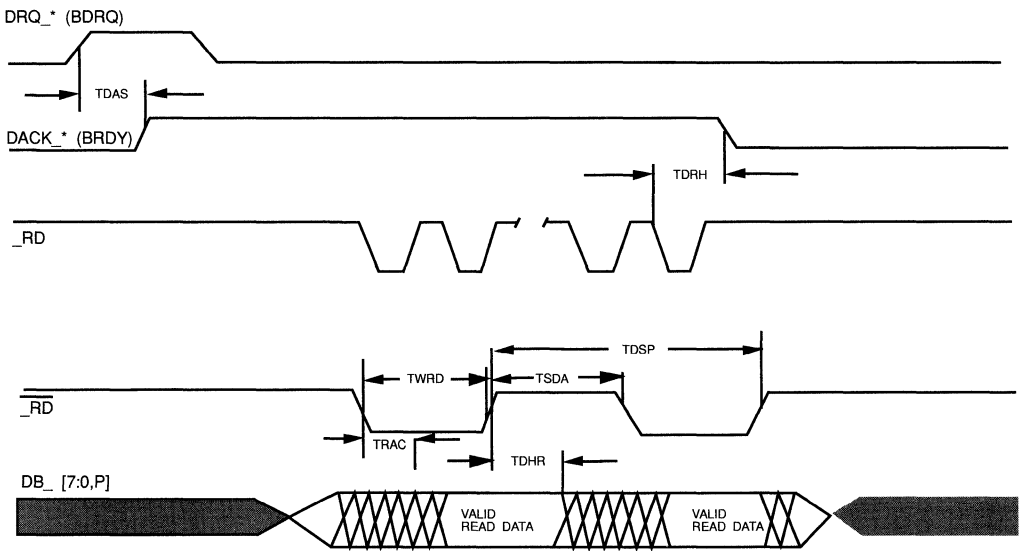
FIGURE 9. SLAVE BURST MODE TRANSFERS



NAME	REF	R/S	MINIMUM	MAXIMUM
TWWR - Width of Write Strobe		R	50 NS	
TDQD = Time to DRQ Deasserted	LE	S	1.5 TCYC	3.5 TCYC + 15 NS
TDAH = Time of DACK Hold	TE	R	0 NS	
TSUW = Data Setup Time to Write	TE	R	20 NS	
TWDH = Time of Data Hold in Write	TE	R	10 NS	
TRON = Time Read Output Enable from DACK (If DIR bit is set)	LE	S	0 NS	50 NS
TWRD = Width of Read Strobe		R	50 NS	
TRAC = Data Access Time	LE	S		45 NS
TDHR = Data Hold Time of Read Data	TE	S	10 NS	45 NS
TROZ = Time of Read Strobe to High Z	TE	S		50 NS
TRWA = Time of R/W to DACK Inactive	LE	S	2 TCYC + 15 NS	

* DRQ polarity shown high true; DACK polarity shown low true

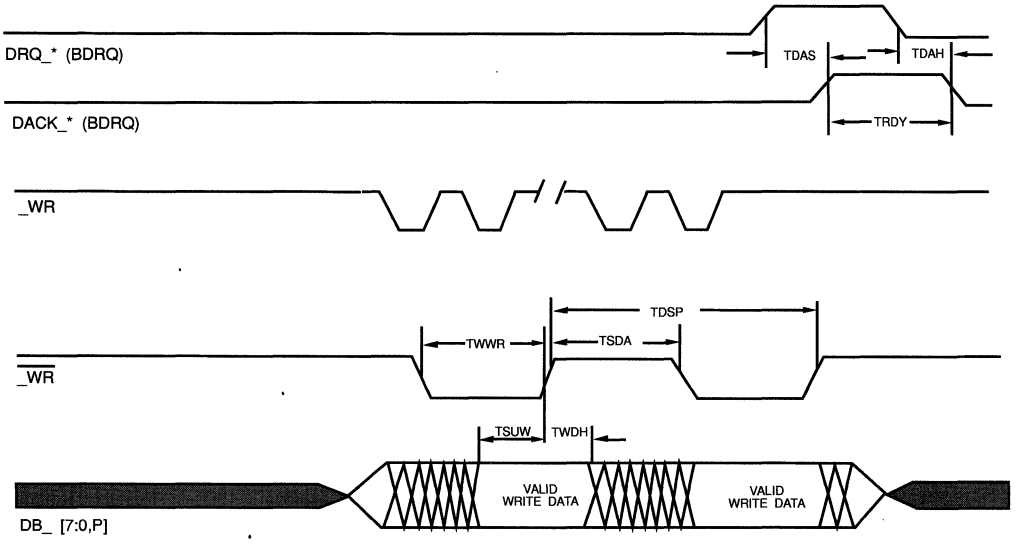
FIGURE 10. SLAVE SINGLE CYCLE TRANSFER



NAME	REF	R/S	MINIMUM	MAXIMUM
TDAS = Time of DACK Setup	LE	S	3.5 TCYC	5.5 TCYC + 15 NS
TSDA = Time of Strobe Deactivated		R	50 NS	
TDSP = Time of Data Strobe Period		R	2.5 TCYC	
TDAH = Time of DACK Hold	LE	S		6 TCYC
TRAC = Data Access Time	LE	S		45 NS
TDHR = Data Hold Time of Read Data	LE	S	10 NS	
TWRD = Width of Read STrobe		R	50 NS	

* DRQ polarity shown high true; DACK polarity shown low true

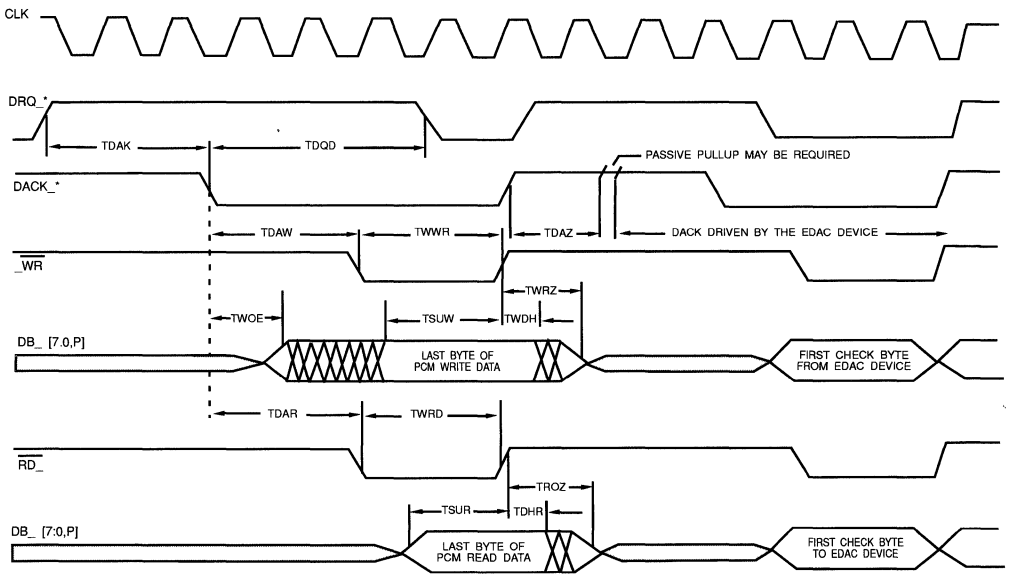
FIGURE 11. SLAVE BURST DISK MODE (READ)



NAME	REF	R/S	MINIMUM	MAXIMUM
TWWR = Width of Write Strobe		R	50 NS	
TSDA = Time of Strobe Deactivated		R	50 NS	
TDSP = Time of Data Strobe Period		R	2.5 TCYC	
TDAH = Time of DACK Hold	TE	R	2 TCYC	
TSUW = Data Setup Time to Write	TE	R	20 NS	
TWDH = Time of Data Hold in Write	TE	R	10 NS	
TRDY = Width of BRDY (DACK)		R	10 TCYC	
TDAS = Time of DACK Setup		S		6 TCYC

* DRQ polarity shown high true; DACK polarity shown low true

FIGURE 12. SLAVE BURST DISK MODE (WRITE)



NAME	REF	R/S	MINIMUM	MAXIMUM
TACS = Time of DACK to Chip Select	TE	S	2 TCYC - 15 NS **	
TCSA = Time of Chip Select to DACK	TE	S	2 TCYC - 15 NS **	
TDAW = Time DACK to Write Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWWR = Width of Write Strobe (See SC in Section 4.1)		S	2 TCYC - 15 NS	8 TCYC + 15 NS
TWOE = Time to Output Enable in Write	LE	S	0 NS	
TSUW = Data Setup Time to Write	TE	S	2 TCYC - 40 NS	8 TCYC + 15 NS
TWDH = Time of Data Hold in Write	TE	S	TCKH - 5 NS	
TWRZ = Time of Write Strobe to High Z	TE	S	TCKH - 5 NS	6.5 TCYC
TDAR = Time DACK to Read Strobe (See DLY in Section 4.1)	LE	S	2 TCYC - 15 NS	4 TCYC + 15 NS
TWRD = Width of Read Strobe (See SC in Section 4.1)		S	2 TCYC - 15 NS	8 TCYC + 15 NS
TSUR = Data Setup Time of Read Data	TE	R	30 NS	
TDHR = Data Hold Time of Read Data	TE	R	0 NS	
TROZ = Time of Read Strobe to High Z	TE	R		2 TCYC **
TDQD = Time to DRQ Deasserted (to stop Burst)	LE	R		2 TCYC - 20 NS ***
TSDA = Time of Strobe Deactivated (See SDTC in Section 4.1)		S	2 TCYC - 15 NS	4 TCYC + 15 NS
TDAZ = Time to Dack High Z	TE	S		6 TCYC

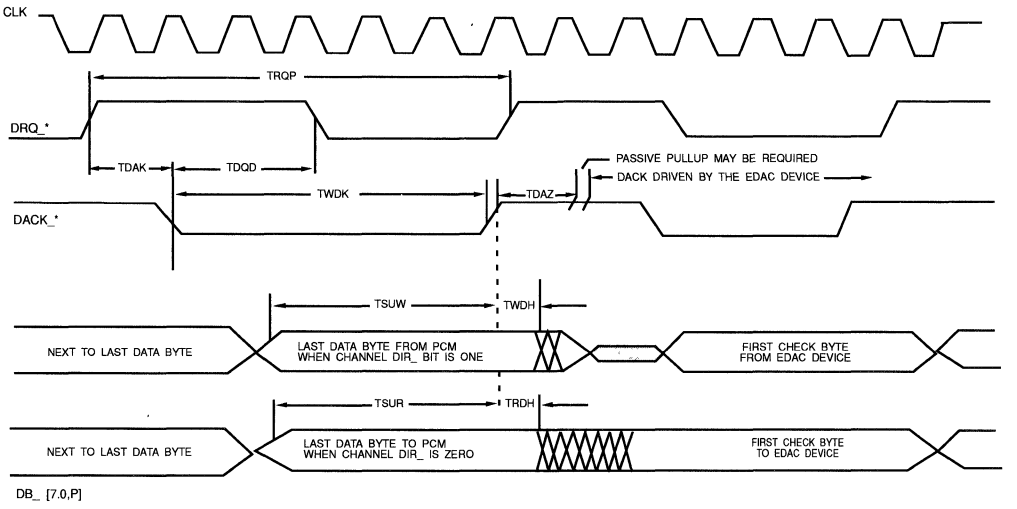
* DRQ Polarity shown high true; DACK polarity shown low true.

** If SDTC in Section 4.1 is set, then value is 4 TCYC.

*** Based on SC and SDTC set to zero. Add 2 TCYC for each programmed state.

FIGURE 13. EDAC MODE (DMA SINGLE CYCLE)

35



NAME	REF	R/S	MINIMUM	MAXIMUM
TDAK = Time DRQ to Acknowledge	LE	S		35 NS
TDQD = Time to DRQ Deasserted	LE	R		3 TCYC
TRQP = Time of Request Period		R	6.5 TCYC	
TWDK = Width of DACK Signal		S	3.5 TCYC	5.5 TCYC + 15 NS
TSUW = Data Setup Time of Write Data	TE	S	2 TCYC - 40 NS	
TWDH = Data Hold Time of Write Data	TE	S	TCKH - 5 NS	
TSUR = Data Setup Time of Read Data	TE	R	30 NS	
TDHR = Data Hold Time of Read Data	TE	R	0 NS	
TDAZ = Time to DACK High Z	TE	S		6 TCYC

* DRQ polarity shown high true; DACK polarity shown low true
** If SDTC in Section 4.1 is set, then value is 4 TCYC
*** Based on SC and SDTC set to zero. Add 2 TCYC for each programmed state

FIGURE 14. EDAC MODE (DMA DISK)

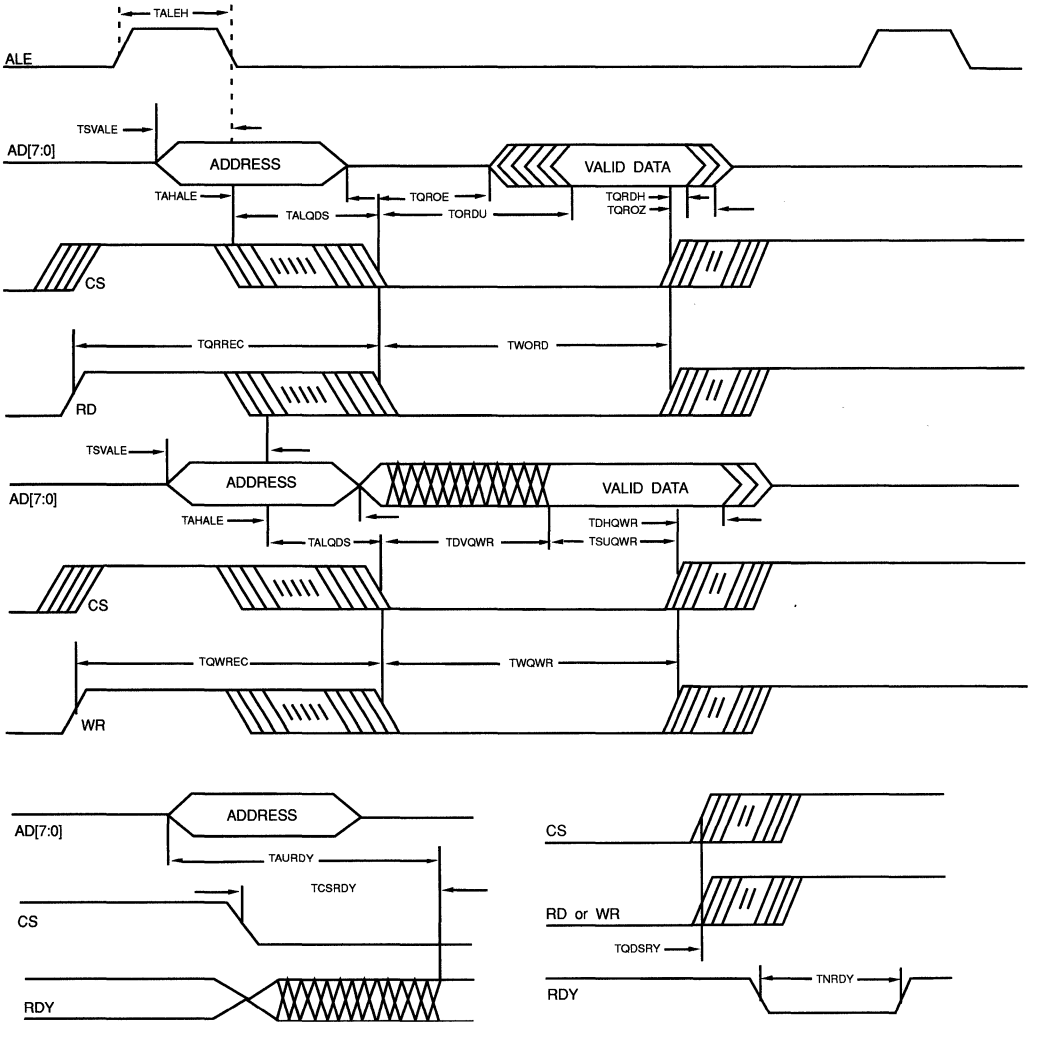


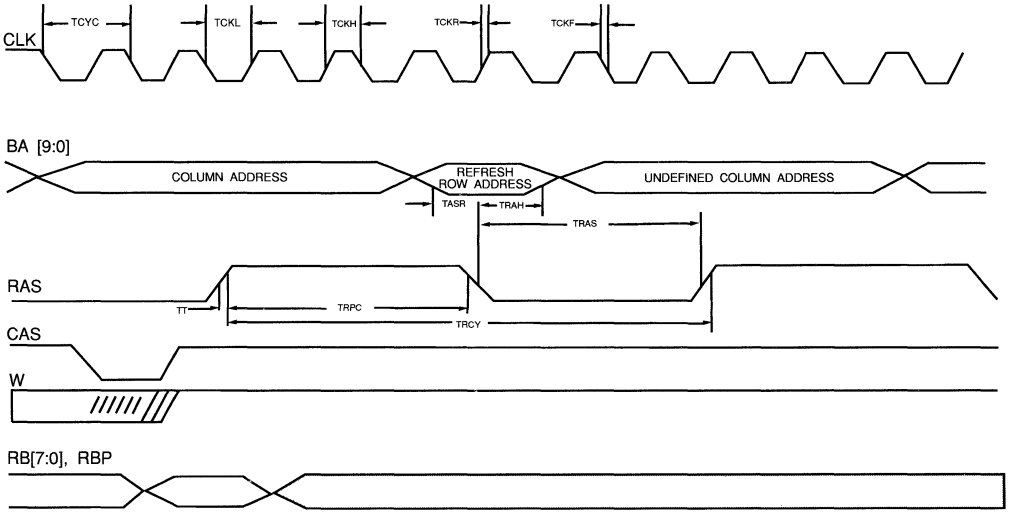
FIGURE 15. MICROPROCESSOR BUS TIMING

35

NAME	REF.	R/S	TIMING
TALEH = ALE High Pulse Width		R	35 NS MIN.
TSVALE = Address Setup Before T.E. ALE	TE	R	15 NS MIN.
TAHALE = Address Hold after T.E. ALE	TE	R	15 NS MIN.
TALQDS = ALE T.E. to Qualified Data Strobe	TE	R	30 NS MIN.
TWORD = Pulse Width Qualified Read Strobe		R	110 NS MIN.
TQROE = Qualified Read Strobe to Output Enabled	LE	S	40 NS MAX.
TQRDV = Qualified Read Strobe to Data Valid	LE	S	95 NS MAX.
TQRDH = Data Hold from Qualified Read Strobe	TE	S	0 NS MIN.
TQROZ = T.E. Qualified Read Strobe to High Impedance	TE	S	60 NS MAX.
TQRREC = Recovery Time after Qualified Read Strobe	TE	R	120 NS MIN.
TWQWR = Pulse Width Qualified Write Strobe		R	110 NS MIN.
TDVQWR = Data Valid from L.E. Qualified Write	*	R	3 CLK MAX.
TSUQWR = Data Setup to T.E. Qualified Write	TE	R	30 NS MIN.
TDHQWR = Data Hold from T.E. Qualified Write	TE	R	10 NS MIN.
TQWREC = Recovery Time after Qualified Write Strobe	TE	R	120 NS MIN.
Waitable Microprocessor Interface			
TAURDY = Address Valid to Ready Valid	LE	S	50 NS MAX.
TCSRDY = Chip Select to Readh Valid	LE	S	35 NS MAX.
Non-Waitable Microprocessor Interface			
TQDSRY = T.E. Qualified Data Strobe to Not Ready	TE	S	60 NS MAX.
Either Microprocessor Interface			
TNRDY = Time Not Ready	**	S	6 CLK MIN.

* Required in Waitable Mode Only
** See Section 5.0 for maximum

FIGURE 15. MICROPROCESSOR BUS TIMING, Continued



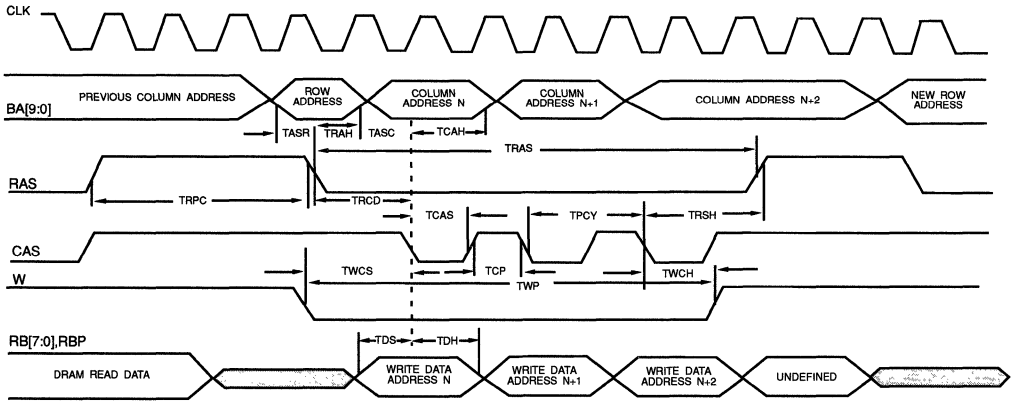
NAME	REF	R/S	MINIMUM	MAXIMUM
TCYC = Input CLock Period		R	40 NS	
TCKL = Input Clock Low		R	17 NS	
TCKH = Input Clock High		R	17 NS	
TCKR = Input Clock Rise		R		5 NS
TCKF = Input Clock Fall		R		5 NS
TRAS = Time of RAS Low		S	3.5 TCYC - 15 NS	
TRPC = Time of RAS Precharge (RAS High)		S	2.5 TCYC - 5 NS	TRFC
TASR = Address Setup to L.E. RAS	FE	S	TCYC - 20 NS	TCYC
TRAH = Address Hold from L.E. RAS	FE	S	TCYC - 15 NS	
TRCY = RAS Cycle Time		S	6 TCYC	TRFC **
TRFC = Refresh Cycle Time (Programmable)		S	32 TCYC	512 TCYC ***
TT = Transition Time All Outputs		S	3 NS	20 NS

** This specification assumes no other memory requests other than Refresh.

*** This is the maximum programmable value. The programmed value is based on DRAM specification.

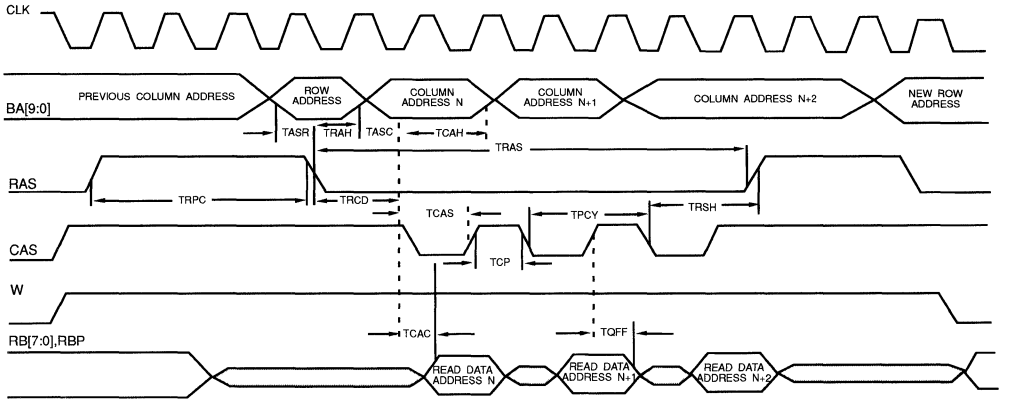
FIGURE 16. RAS ONLY REFRESH TIMING

35



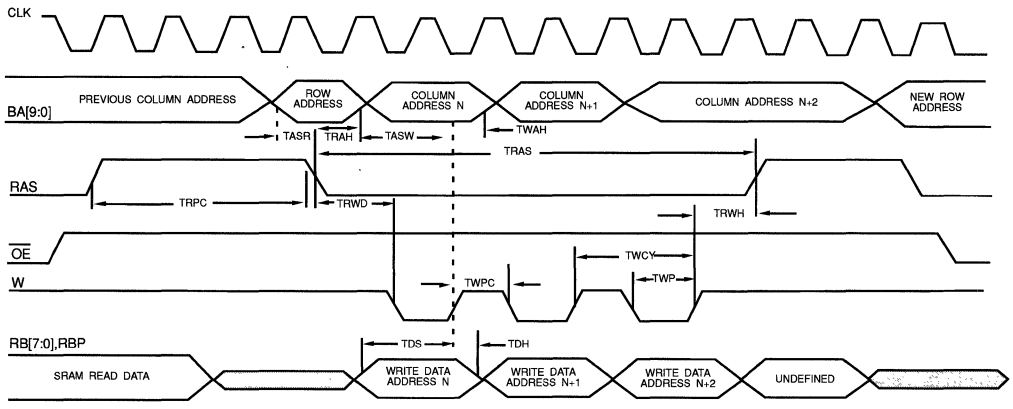
NAME	REF	R/S	MINIMUM	MAXIMUM
TRAS = Time of RAS Asserted		S	3.5 TCYC - 15 NS	
TRPC = Time of RAS Precharge (RAS High)		S	2.5 TCYC - 5 NS	TRFC
TASR = Address Setup to L.E. RAS	FE	S	TCYC - 20 NS	TCYC
TRAH = Address Hold from L.E. RAS	FE	S	TCYC - 15 NS	
TRCY = RAS Cycle Time		S	6 TCYC	
TT = Transition Time All Outputs		S	3 NS	20 NS
TRCD = L.E. RAS to L.E. CAS Delay	LE	S	2 TCYC - 25 NS	
TCAS = Time of CAS Asserted		S	TCYC	
TCP = Time of CAS Precharge (CAS High)		S	TCYC - 20 NS	
TWP = Time of Write Asserted		S	2 TCYC	
TASC = Address Setup to L.E. CAS	LE	S	TCKH - 15 NS	
TCAH = Address Hold from L.E. CAS	LE	S	1.5 TCYC - 15 NS	
TDS = Data Setup to L.E. CAS	LE	S	TCKH - 15 NS	
TDH = Data Hold from L.E. CAS	LE	S	1.5 TCYC - 15 NS	
TRSH = RAS Hold from L.E. CAS	LE	S	TCYC	
TWCS = Write Setup to L.E. CAS	LE	S	TCYC	
TWCH = Write Hold from L.E. CAS	LE	S	TCYC	

FIGURE 17. PAGE MODE WRITE TIMING



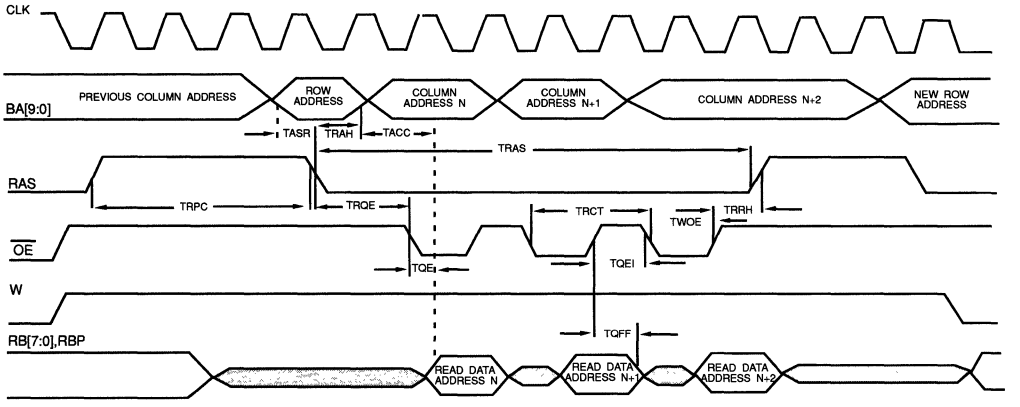
NAME	REF	R/S	MINIMUM	MAXIMUM
TRAS = Time of RAS Asserted		S	3.5 TCYC - 15 NS	
TRPC = Time of RAS Precharge (RAS High)		S	2.5 TCYC - 5 NS	TRFC
TASR = Address Setup to L.E. RAS	FE	S	TCYC - 20 NS	TCYC
TRAH = Address Hold from L.E. RAS	FE	S	TCYC - 15 NS	
TRCY = RAS Cycle Time		S	6 TCYC	
TT = Transition Time All Outputs		S	3 NS	20 NS
TRCD = L.E. RAS to L.E. CAS Delay	LE	S	1.5 TCYC - 5 NS	
TCAS = Time of CAS Asserted		S	TCYC	
TCP = Time of CAS Precharge (CAS High)		S	TCYC - 20 NS	
TASC = Address Setup to L.E. CAS	LE	S	TCKH - 15 NS	
TCAH = Address Hold from L.E. CAS	LE	S	1.5 TCYC - 15 NS	
TRSH = RAS Hold from L.E. CAS	LE	S	TCYC	
TCAC = CAS Access Time	LE	R		TCYC
TOFF = Data High-Z Delay	TE	R	0 NS	30 NS

FIGURE 18. PAGE MODE READ TIMING



NAME	REF	R/S	MINIMUM	MAXIMUM
TRAS = Time of RAS Asserted		S	3.5 TCYC - 15 NS	
TRPC = Time of RAS Precharge (RAS High)		S	2.5 TCYC - 5 NS	TRFC
TASR = Address Setup to L.E. RAS	FE	S	TCYC - 20 NS	TCYC
TRAH = Address Hold from L.E. RAS	FE	S	TCYC - 15 NS	
TRCY = RAS Cycle Time		S	6 TCYC	
TT = Transition Time All Outputs		S	3 NS	20 NS
TRWD = L.E. RAS to L.E. Write Delay	LE	S	2 TCYC - 25 NS	
TWCY = Write Cycle		S	2 TCYC	
TWPC = Time of Write Precharge (Write High)		S	TCYC - 20 NS	
TWP = Time of Write Asserted		S	TCYC	
TASW = Address Setup to T.E. Write	TE	S	1.5 TCYC - 15 NS	
TWAH = Address Hold from T.E. Write	TE	S	5 NS	
TDS = Data Setup to T.E. Write	TE	S	1.5 TCYC - 15 NS	
TDH = Data Hold from T.E. Write	TE	S	5 NS	
TRWH = RAS Hold Time from T.E. Write	TE	S	0 NS	

FIGURE 19. STATIC RAM WRITE TIMING



NAME	REF	R/S	MINIMUM	MAXIMUM
TRAS = Time of RAS Asserted		S	3.5 TCYC - 15 NS	
TRPC = Time of RAS Precharge (RAS High)		S	2.5 TCYC - 5 NS	TRFC
TASR = Address Setup to L.E. RAS	FE	S	TCYC - 20 NS	TCYC
TRAH = Address Hold from L.E. RAS	FE	S	TCYC - 15 NS	
TRCY = RAS Cycle Time		S	6 TCYC - 15 NS	
TT = Transition Time All Outputs		S	3 NS	20 NS
TROE	LE	S	2 TCYC - 25 NS	
TWOE		S	TCYC	
TOEI		S	TCYC - 20 NS	
TACC		R	TCYC	1.5 TCYC - 15 NS
TRRH	TE	S	0 NS	
TOE	LE	R		TCYC
TOFF	TE	R	0 NS	30 NS
TRCT		S	2 TCYC	

FIGURE 20. STATIC RAM READ TIMING

15.0 PACKAGE DIAGRAMS

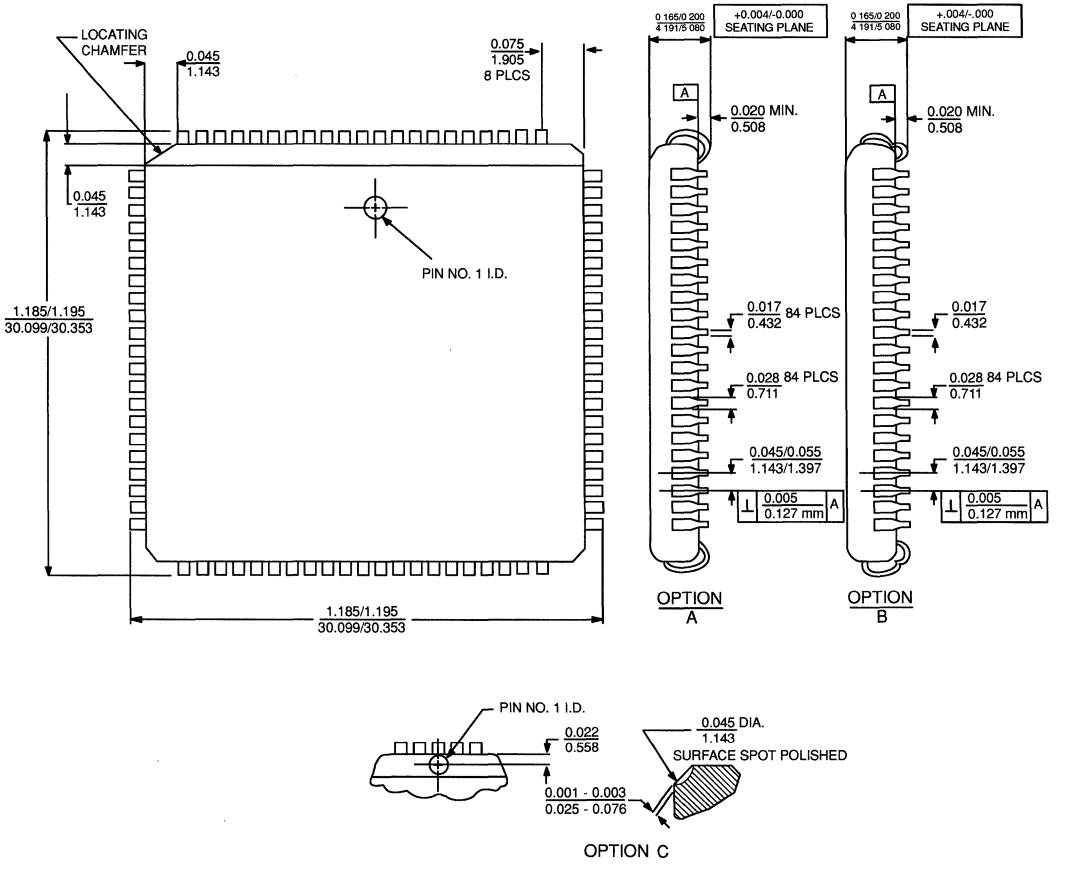


FIGURE 21. 84 LEAD PLCC

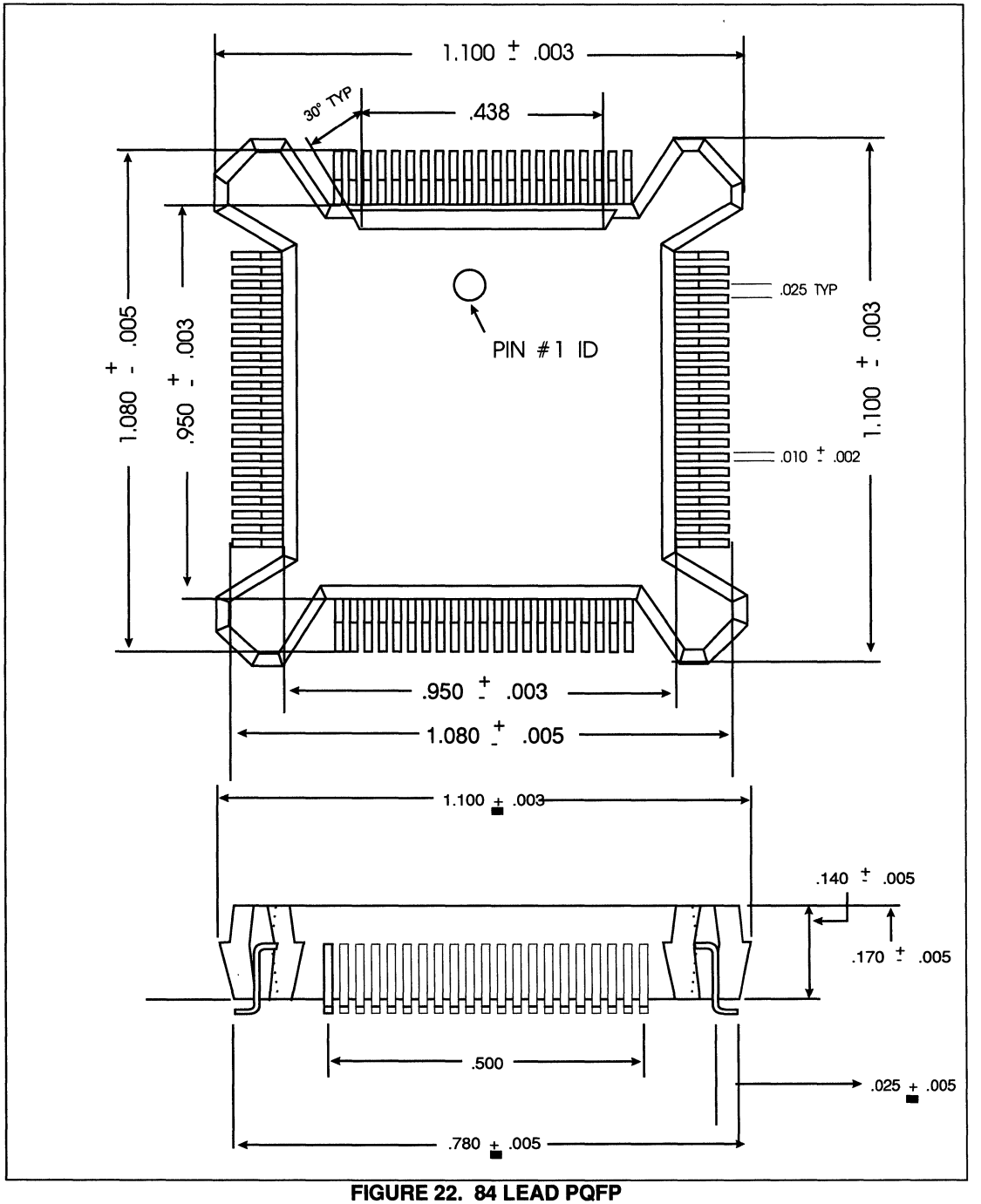


FIGURE 22. 84 LEAD PQFP

35