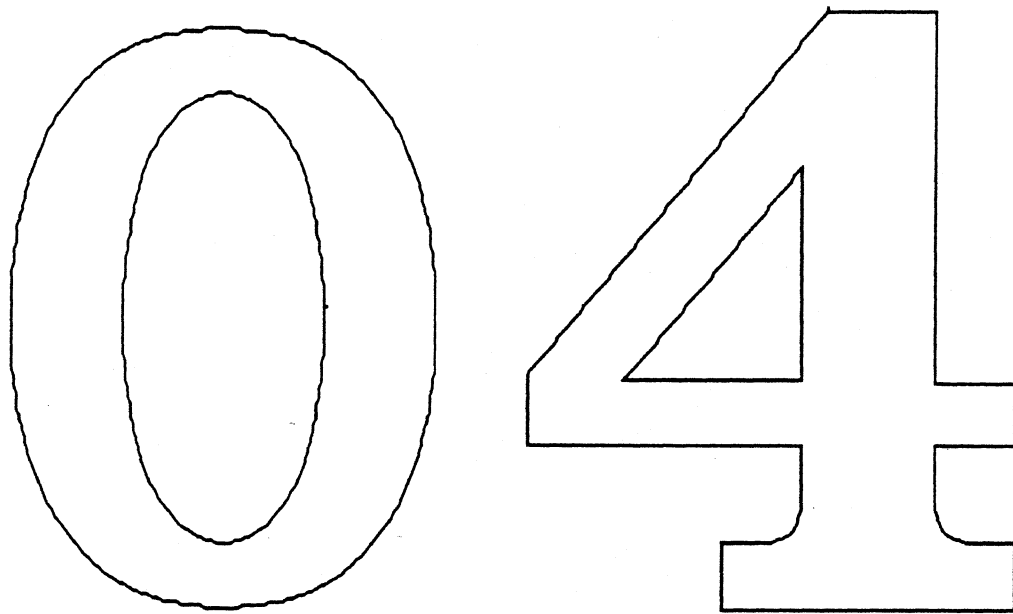




REV.	ZONE	ECO#	REVISION	APPD	DATE
A			PRODUCTION RELEASE		



**NOTE:**  
**MANUFACTURERS RECEIVING APPLE APPROVED VENDOR STATUS FOR THEIR PRODUCT UNDER THIS PART NUMBER PLEASE NOTE:** You must not change your part design, materials or manufacturing process from those used for the original samples submitted and approved by Apple without written approval of Apple. Proposed changes determined by Apple to be significant, will require the manufacturer to submit new samples and/or data for review and approval prior to product

343S0064-A

1/31

 <small>DIMENSIONS ARE IN MILLIMETERS.  DIMENSIONS IN BRACKETS [ ] ARE IN INCHES.</small> <b>TOLERANCES</b> X.X ± .3 [.01] X.XX ± .13 [.005] X.XXX ± .03 [.001] ANGLEs ± .1 or as noted DO NOT SCALE DRAWING	<b>METRIC</b>				 <b>Apple Computer, Inc.</b>	
	DRAFT	///	DESIGN CK	///	<small>NOTICE OF PROPRIETARY PROPERTY  THE INFORMATION CONTAINED HEREIN IS THE PROPRIETARY PROPERTY OF APPLE COMPUTER, INC. THE POSSESSOR AGREES TO THE FOLLOWING:  (i) TO MAINTAIN THIS DOCUMENT IN CONFIDENCE  (ii) NOT TO REPRODUCE OR COPY IT</small>	
	BT	///	MPG APPD	///		
	ENG APPD	///	DESIGNER	///	<b>TITLE</b> IC, CUSTOM SCSI DMA CONTROLLER, QFP-100	
	QA APPD	///	SCALE	///		
RELEASE	///			<b>DRAWING NUMBER</b>	<b>SHT</b>	
MATERIAL/FINISH NOTED AS APPLICABLE		SIZE <b>A</b>	<b>343S0064-A</b>		<b>1/31</b>	

**TABLE 1. NORMAL OPERATION MODE PIN PLACEMENT**

Note: Normal operation mode occurs when the TEST input (pin 65) is low.

1. VSS	26. D(17)	51. /SD(6)	76. VSS
2. MA(12)	27. D(18)	52. /SD(7)	77. SIZ0
3. MA(13)	28. D(19)	53. /SDBP	78. SIZ1
4. MA(14)	29. VSS	54. /SATN	79. /DSACK0
5. MA(15)	30. D(20)	55. /SBSY	80. /STERM
6. D(0)	31. D(21)	56. VSS	81. FC1
7. D(1)	32. D(22)	57. /SACK	82. R/W
8. D(2)	33. D(23)	58. /SRST	83. /DSACK1
9. D(3)	34. D(24)	59. /SMSG	84. /AS
10. VSS	35. D(25)	60. /SSEL	85. /ABEN
11. D(4)	36. D(26)	61. /SCD	86. /ALE
12. D(5)	37. D(27)	62. VSS	87. MA(0)
13. D(6)	38. VSS	63. /SREQ	88. MA(1)
14. D(7)	39. D(28)	64. /SIO	89. MA(2)
15. D(8)	40. D(29)	65. TEST	90. MA(3)
16. D(9)	41. D(30)	66. VDD	91. VDD
17. D(10)	42. D(31)	67. CPUCLK	92. VSS
18. D(11)	43. /SD(0)	68. /CS	93. MA(4)
19. VDD	44. /SD(1)	69. /INT	94. MA(5)
20. VSS	45. VSS	70. /RESET	95. MA(6)
21. D(12)	46. /SD(2)	71. /BERR	96. MA(7)
22. D(13)	47. /SD(3)	72. /BGACK	97. MA(8)
23. D(14)	48. /SD(4)	73. /BGOUT	98. MA(9)
24. D(15)	49. /SD(5)	74. /BGIN	99. MA(10)
25. D(16)	50. VSS	75. /BR	100. MA(11)

**TABLE 2. NORMAL OPERATION MODE PIN DESCRIPTION**

**INPUTS**

- /RESET** - Used to reset the SCSI DMA.
- /CS** - Chip Select. Used to select the SCSI DMA during PIO (Programmed I/O) transfers.
- CPUCLK** - Clock signal to the SCSI DMA.
- /BGIN** - The assertion of /BGIN signals that the bus is to be given to a requesting bus master when the current bus cycle is completed. The SCSI DMA uses this in combination with other signals to determine when it can take the bus.
- TEST** - Allows the 53C80 host adapter cell (a part of the logic internal to the SCSI DMA) to be tested directly. See Section 6.2.3 for further Test Mode information.
- /BERR** - Indicates an invalid or illegal bus operation is being attempted.
- /STERM** - Indicates the completion of a 32 bit synchronous data transfer.

**OUTPUTS**

- /BGOUT** - The outgoing daisy chained bus grant line. The SCSI DMA will normally pass the signal /BGIN to /BGOUT unless it intends to become the next bus master. It then breaks the daisy chain and negates /BGOUT.
- /DSACK1** - Indicates the completion of a 32 bit PIO data transfer.

- /ALE** - During DMA operations, the rising edge of this signal is used to latch the 16 high order DMA address bits. This signal is to supply the clock input to the necessary external address - latching registers. (2 registers, such as 74F574's, are required in a system to accommodate the multiplexed-address design of the SCSI DMA. See figure 5 for schematic.)
- /ABEN** - During DMA transfers, this signal is used to enable the high order address bits onto the address bus. This signal is to supply the output enable input to the necessary external address-latching registers.
- MA[15-9,3-0]** - With MA[8-4] forms the multiplexed address bus.
- /INT** - Interrupt signal from the SCSI DMA. (Open drain output. Timing specs derived from system with 3.3k pullup registers on open drain outputs.)
- /BR** - The SCSI DMA requests bus mastership using this line. (Open drain output.)
- FC1** - Function code bit 1. (Used during DMA cycles. This bit is driven low to indicate a non-CPU space operation.)

**BIDIRECTIONAL**

- D[31-0]** - Bidirectional 32 bit data bus used for both DMA and PIO transfers.
- MA[8-4]** - Bidirectional bits of the multiplexed address bus.  
PIO Mode: Input to indicate register to be accessed during PIO transfers.  
DMA Mode: Output to form multiplexed address bus with MA[15-9] and MA[3-0].
- /AS** - Indicates that valid address, size, and R/W information is on the bus.  
PIO Mode: Input driven by 68030.  
DMA Mode: Output.
- R/W** - Indicates a read (high) or write (low) direction.  
PIO Mode: Input driven by 68030.  
DMA Mode: Output.
- SIZ[1-0]** - Indicate the number of bytes to be transferred during a given bus cycle.  
PIO Mode: Input driven by 68030.  
DMA Mode: Output.
- /DSACK0** - Indicates termination of an asynchronous data transfer.  
PIO Mode: Output.  
DMA Mode: Input driven by asynchronous memory controller.  
(/STERM terminated memory cycles afford better performance.)
- /BGACK** - Indicates that an external device (not the CPU) has assumed bus mastership.
- /SDB[7-0]** - SCSI data bus.
- /SDBP** - Odd data parity for the SCSI data bus.
- /SATN** - Indicates an attention condition on the SCSI bus.
- /SBSY** - Indicates that the SCSI bus is being used.
- /SREQ** - Indicates a request for a SCSI REQ/ACK data transfer.
- /SACK** - Indicates an acknowledgement during a SCSI REQ/ACK data transfer.
- /SRST** - Triggers SCSI bus reset.
- /MSG** - Indicates a message condition on the SCSI bus.
- /SSEL** - Used by an initiator to select a target or by a target to reselect an initiator on the SCSI bus.
- /SCD** - Indicates that control or data is on the SCSI bus.
- /SIO** - Controls the direction of data movement on the SCSI bus. Also used to distinguish between selection and reselection phases.

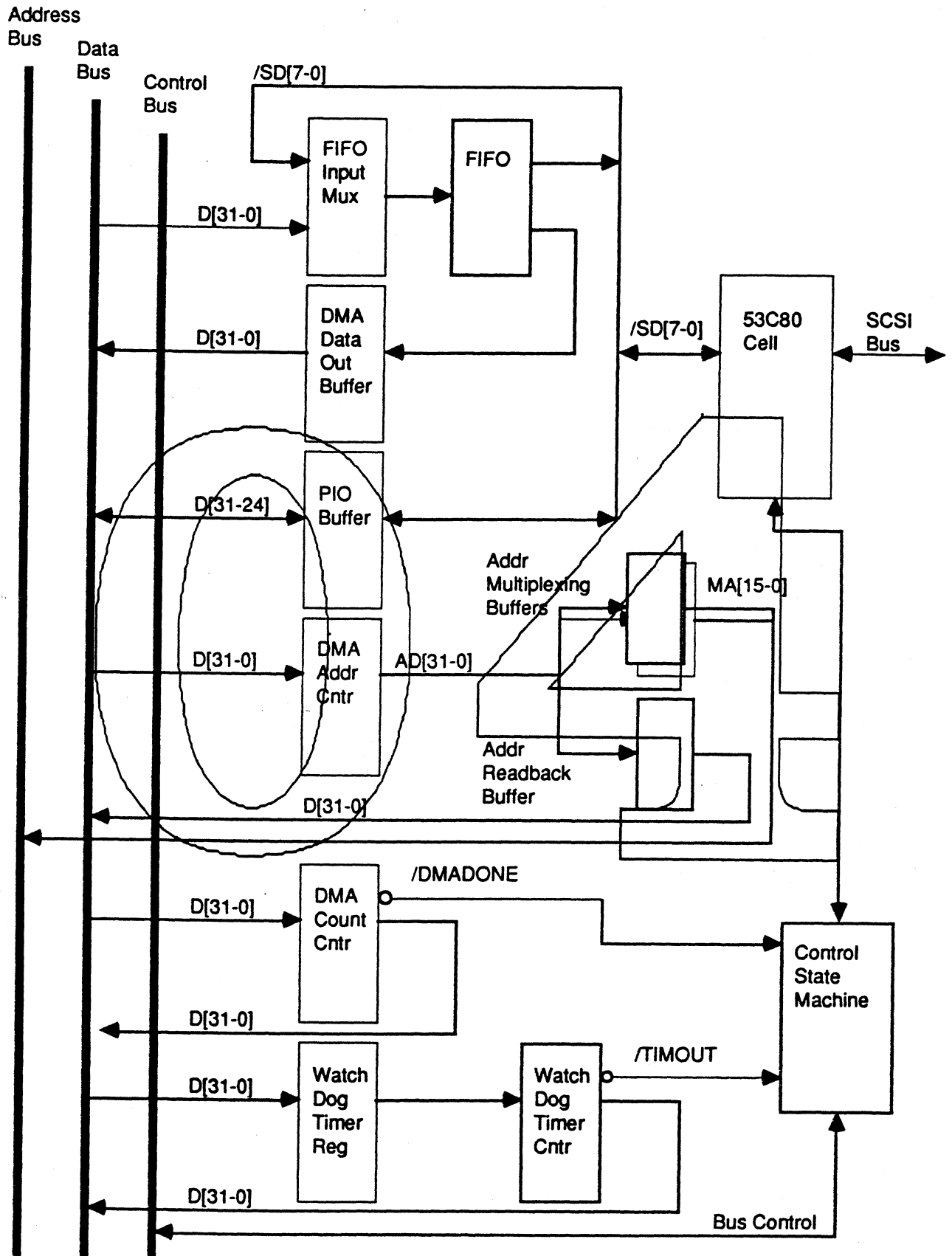


FIGURE 2. FLOW BLOCK DIAGRAM

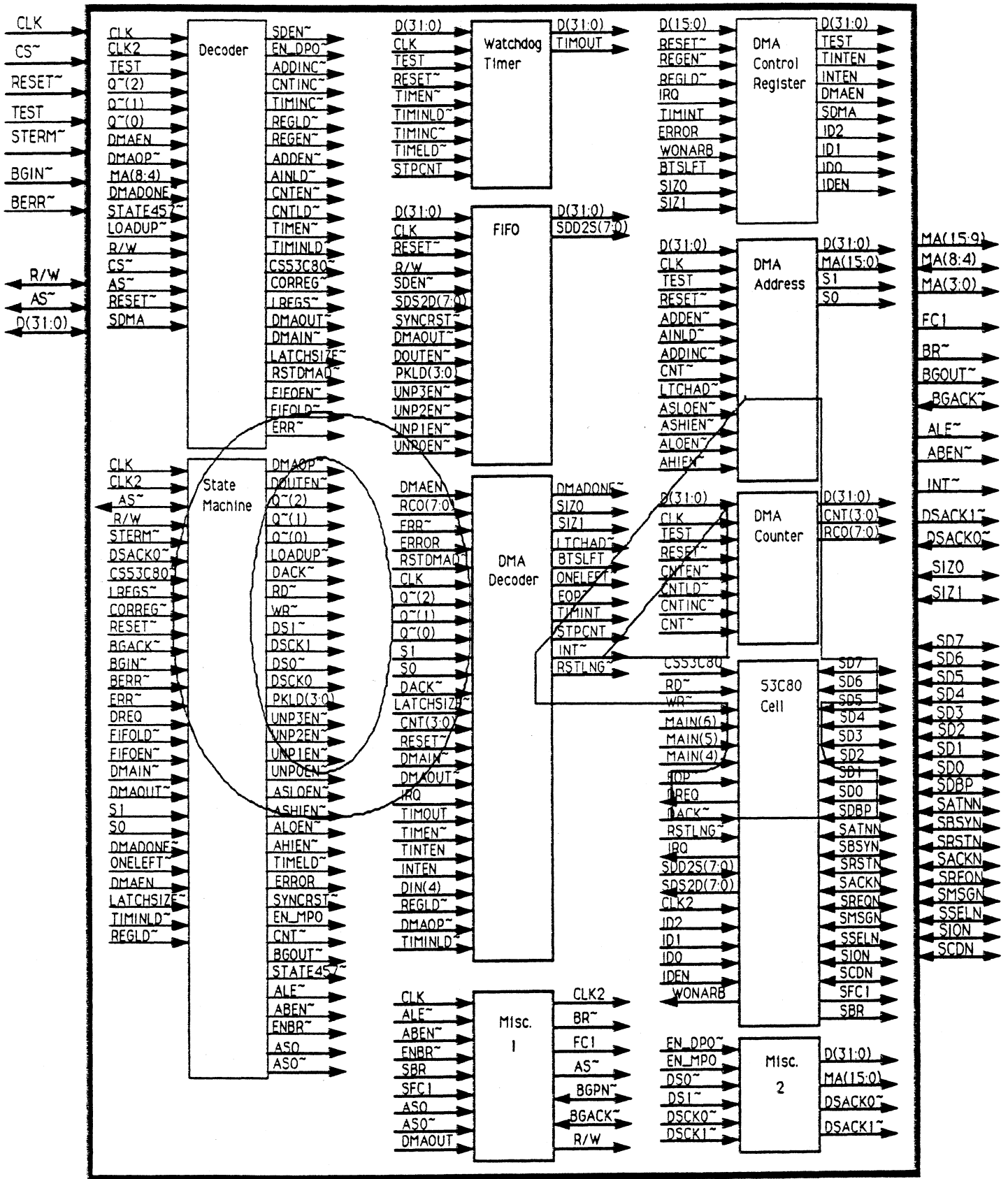


FIGURE 3. DETAILED BLOCK DIAGRAM

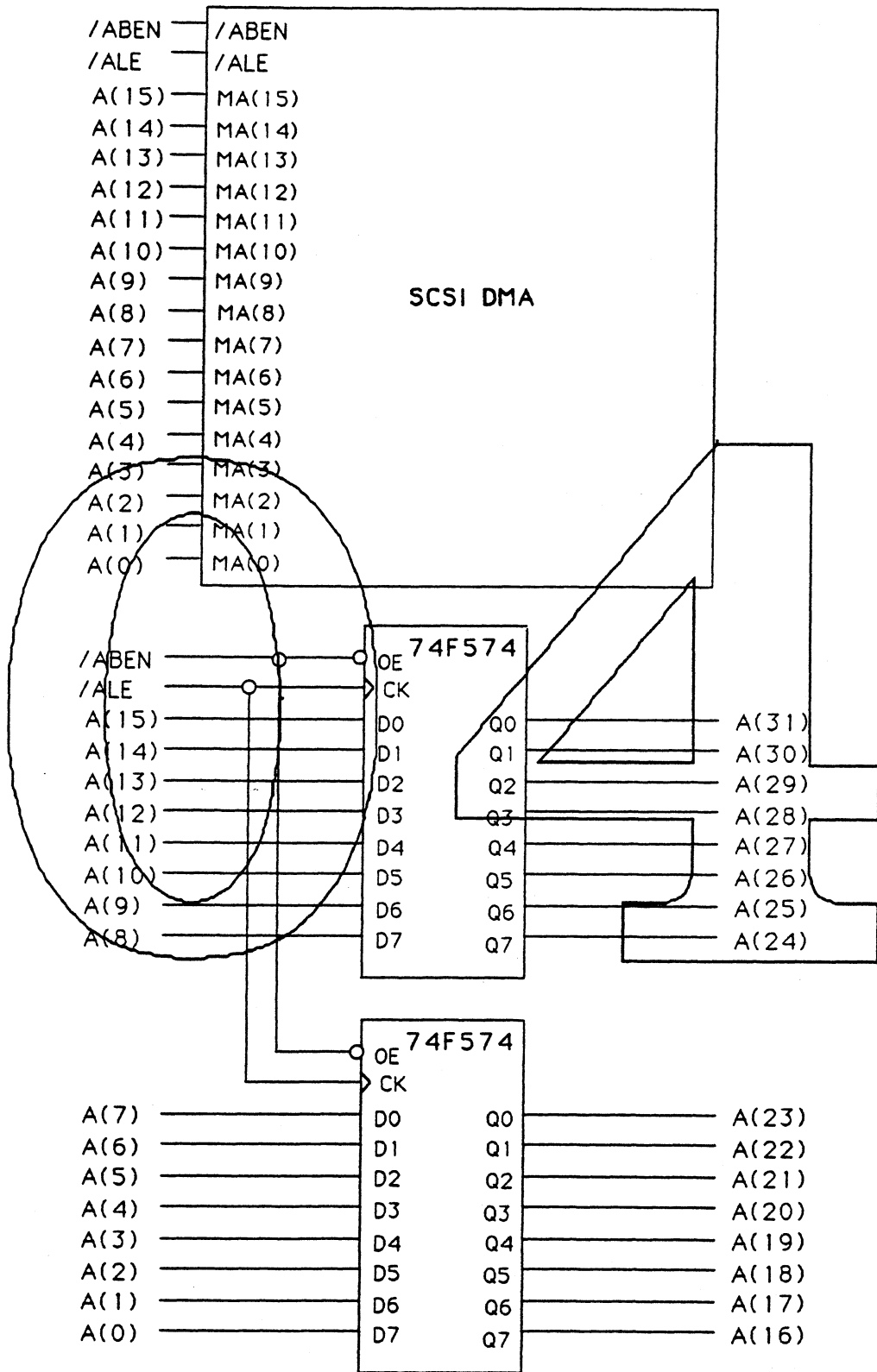


FIGURE 5. SCSI DMA WITH EXTERNAL ADDRESS REGISTERS SCHEMATIC

**TABLE 7. TIMING**  
VDD = 4.75V to 5.25V, Temperature = 0 - 70°C, CL = 100pF  
Refer to figures 6 thru 10

SIGNAL	TIMING DESCRIPTION	SYMBOL	MIN.	MAX.	UNITS	FIG.
--------	--------------------	--------	------	------	-------	------

**GENERAL TIMING**

CPUCLK	Frequency			25	MHz	6-10
	Clock Period ( $\phi$ )		40		ns	6-10
	Clock high time		16		ns	6-10
	Clock low time		16		ns	6-10
/RESET	Assertion time		80		ns	
/RESET	Delay time from /RESET rising edge to full operation		8 $\phi$		ns	
TEST	Delay from TEST falling edge to 53C80 inputs and outputs valid on SCSI DMA pads			40	ns	
/AS	CPUCLK falling edge to /AS falling edge	t10	3	18	ns	6-9
/AS, R/W, MA[8/4]	Setup to /CS falling edge	t11	0		ns	6-9
/CS	Fall setup to CPUCLK rising edge to be captured during that cycle	t12	20		ns	6-9
/AS	CPUCLK falling edge to /AS rising edge	t13	0	18	ns	6-9
R/W, MA[8/4]	Hold from /AS rising edge	t14	2		ns	6-9
/CS	Hold from /AS rising edge	t15	0	15	ns	6-9

**PROGRAMMED I/O READ, WRITE FROM REGISTER TIMING**

/DSACK0, /DSACK1	CPUCLK rising edge to /DSACK falling edge	t18		30	ns	6,7
/DSACK0, /DSACK1	/AS rising edge to /DSACK rising edge	t19		20	ns	6,7
/AS	Hold time after /DSACK falling edge	t110	20	2 $\phi$	ns	6,7
D[31-0]	Setup to /DSACK falling edge	t16	25		ns	6
D[31-0]	Hold from /AS rising edge	t17	5		ns	6
D[31-0]	Setup to CPUCLK rising edge	t111	20		ns	7
D[31-0]	Hold from /AS rising edge	t112	0		ns	7

**PROGRAMMED I/O READ, WRITE WITH HARDWARE HANDSHAKE FROM/TO 53C80**

/DSACK0	CPUCLK rising edge to /DSACK0 falling edge	t18		30	ns	6,7
/DSACK0	/AS rising edge to /DSACK0 rising edge	t19		20	ns	8,9
/AS	Hold time after /DSACK0 falling edge	t110	20	2 $\phi$	ns	8,9
/SREQ	/SREQ falling edge to start of WO	t113	3 $\phi$	5 $\phi$	ns	8
D[31-24]	Setup to /DSACK0 falling edge	t114	10		ns	8
D[31-24]	Hold from /AS rising edge	t17	5		ns	8
/SREQ	/SREQ rising edge to start of WO	t115	3 $\phi$	5 $\phi$	ns	9
D[31-24]	Setup to CPUCLK rising edge	t116	50		ns	9
D[31-24]	Hold from /AS rising edge	t112	0		ns	9

**TABLE 7. TIMING (CONT.)**

VDD = 4.75V to 5.25V, Temperature = 0 - 70°C, CL = 100pF

Rpu = 3.3k - for open drain outputs /BR and /INT

Refer to figures 6 thru 10

SIGNAL	TIMING DESCRIPTION	SYM.	MIN.	MAX.	UNITS	FIG.
<b>DMA READ,WRITE TIMING</b>						
/BR, /BGOUT	CPUCLK rising edge to /BR falling edge and /BGOUT rising edge	t23		40	ns	10
/ALE	CPUCLK rising edge to /ALE falling edge	t20		45	ns	10
/BGIN	Falling edge to CPUCLK falling edge (to be seen that cycle)	t26	20		ns	10
/BGACK	CPUCLK rising edge to assertion of /BGACK	t28		65	ns	10
/BGACK, /AS /STERM, /DSACK0, /BERR	Signal valid to CPUCLK rising edge (to be seen that cycle)	t29		20	ns	10
MA[15-0]	CPUCLK rising edge to high order address valid	t17		55	ns	10
MA[15-0]	CPUCLK rising edge to high order address invalid and low order address valid	t18		50	ns	10
/ALE	CPUCLK rising edge to /ALE rising edge	t21		40	ns	10
/ABEN, R/W, SIZ0, SIZ1, FC1	CPUCLK rising edge to assertion of signal	t22		45	ns	10
/BGOUT	/BGIN assertion to /BGOUT assertion (/BR deasserted)	t25		40	ns	10
/BR	CPUCLK rising edge to /BR rising edge	t24		170	ns	10
/BGIN	CPUCLK falling edge to /BGIN rising edge	t27	20		ns	10
/BGIN	Rising edge to CPUCLK rising edge	t27a	0		ns	10
/AS	CPUCLK falling edge to /AS falling edge	t30		18	ns	10
/AS	CPUCLK falling edge to deassertion of signal	t31		18	ns	10
/ABEN, /BGACK, R/W, SIZ0, SIZ1, FC1, MA[15-0]	CPUCLK rising edge to deassertion of signals	t19		55	ns	10
D[31-0]	Valid from CPUCLK rising edge (write cycle)	t32	60		ns	10
D[31-0]	CPUCLK rising edge to data invalid (write cycle)	t33	15	75	ns	10
D[31-0]	Data valid setup time to CPUCLK falling edge (read cycle)	t34	2		ns	10
D[31-0]	CPUCLK rising edge to data invalid (read cycle)	t35	15		ns	10
/STERM	/STERM falling edge to CPUCLK rising edge	t36	2		ns	10
/STERM	CPUCLK rising edge to /STERM rising edge	t37	8		ns	10
/DSACK0	Falling edge to CPUCLK rising edge (to be seen that cycle)	t38	20		ns	10
/DSACK0	CPUCLK rising edge to /DSACK0 rising edge	t39	20		ns	10
/BERR	Falling edge to CPUCLK falling edge (to be seen in that cycle)	t40	20		ns	10
/BERR	CPUCLK falling edge to /BERR rising edge	t41	20		ns	10



Note 1: The data hold timing  $t_{35}$  (read cycle) is longer than the data hold timing of the 68030.

Note 2: Careful consideration of the daisy chained /BGIN to /BGOUT signal is required when using this part in a system with more than two bus masters (other than the 68030). The delay from the breaking of the daisy chain (the time until /BGOUT) to the set up time of /BGACK ( $t_{29}$ ) must be less than the round trip delay of the daisy chain above the SCSI DMA.

**TABLE 8. OUTPUT RISE AND FALL TIME**  
(VDD = 4.75V to 5.25V, Temperature = 0 - 70°C)

OUTPUT	TIMING DESCRIPTION	LOAD CAP	MAX	UNIT
/AS /DSACK0, /DSACK1	Rise and Fall Times	100 pF	5	ns
ALL OTHER OUTPUTS	Rise and Fall Times	100 pF	17	ns

04

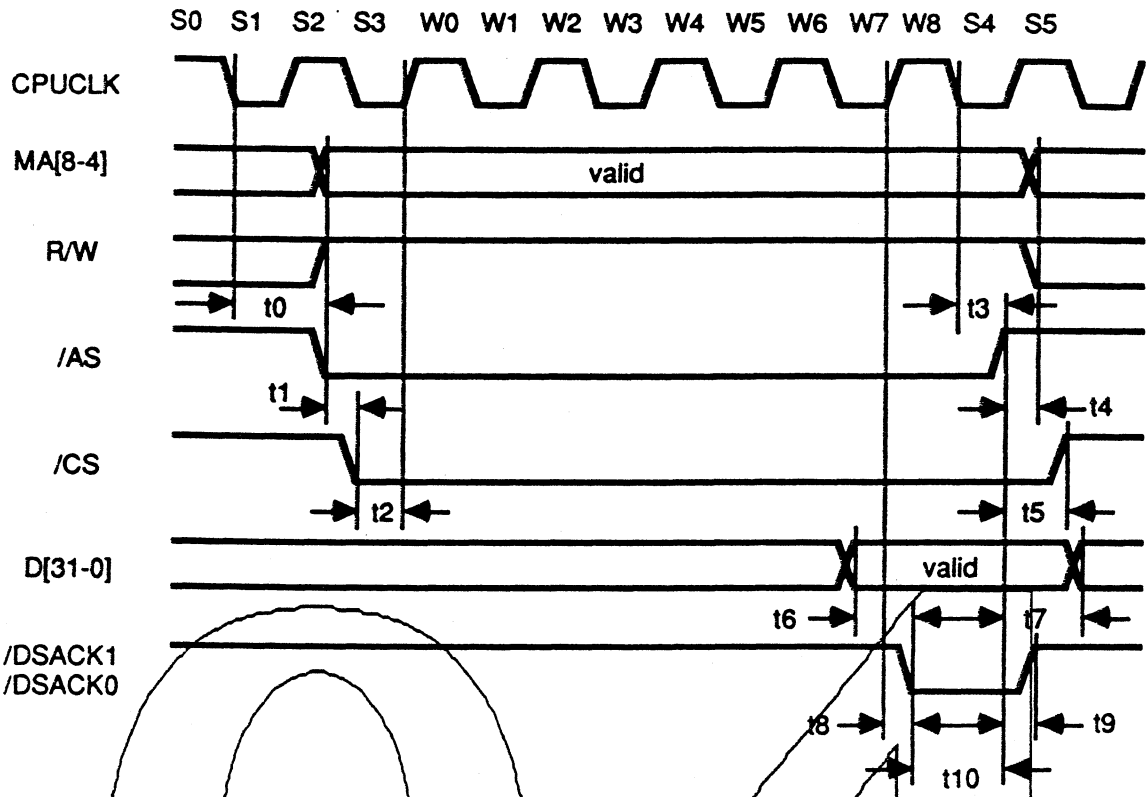


FIGURE 6. PROGRAMMED I/O READ FROM REGISTERS TIMING

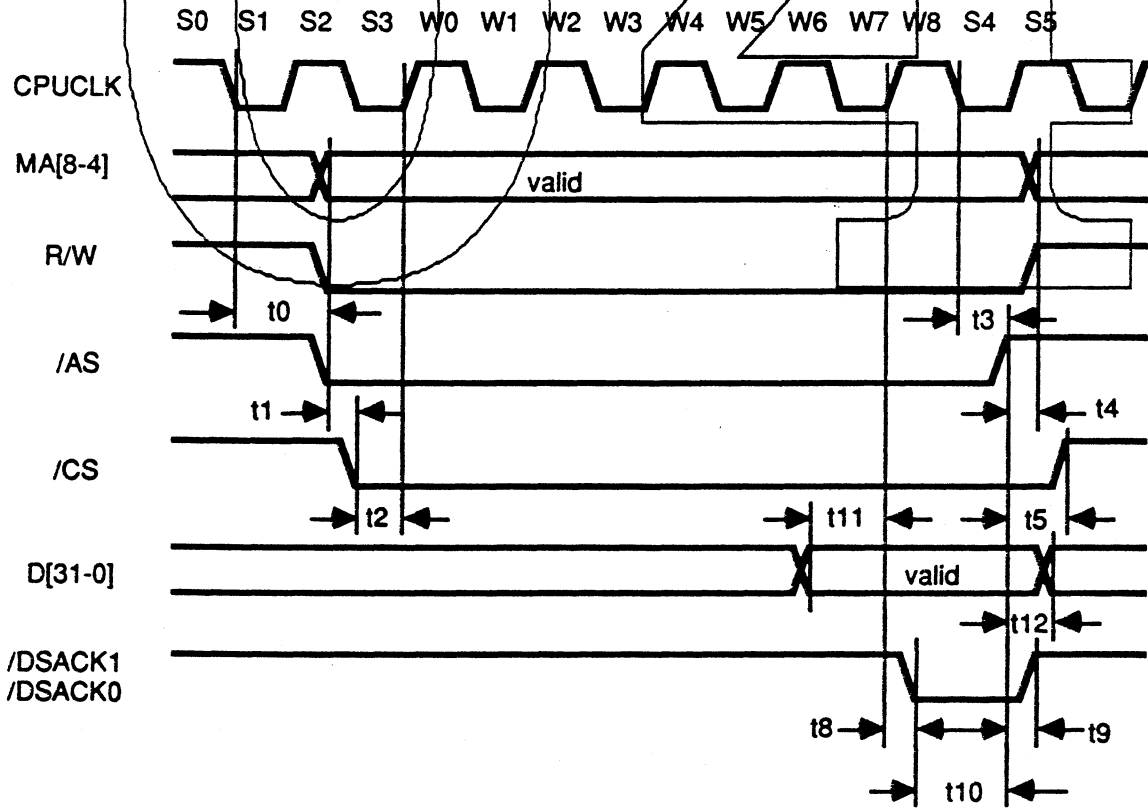


FIGURE 7. PROGRAMMED I/O WRITE TO REGISTERS TIMING

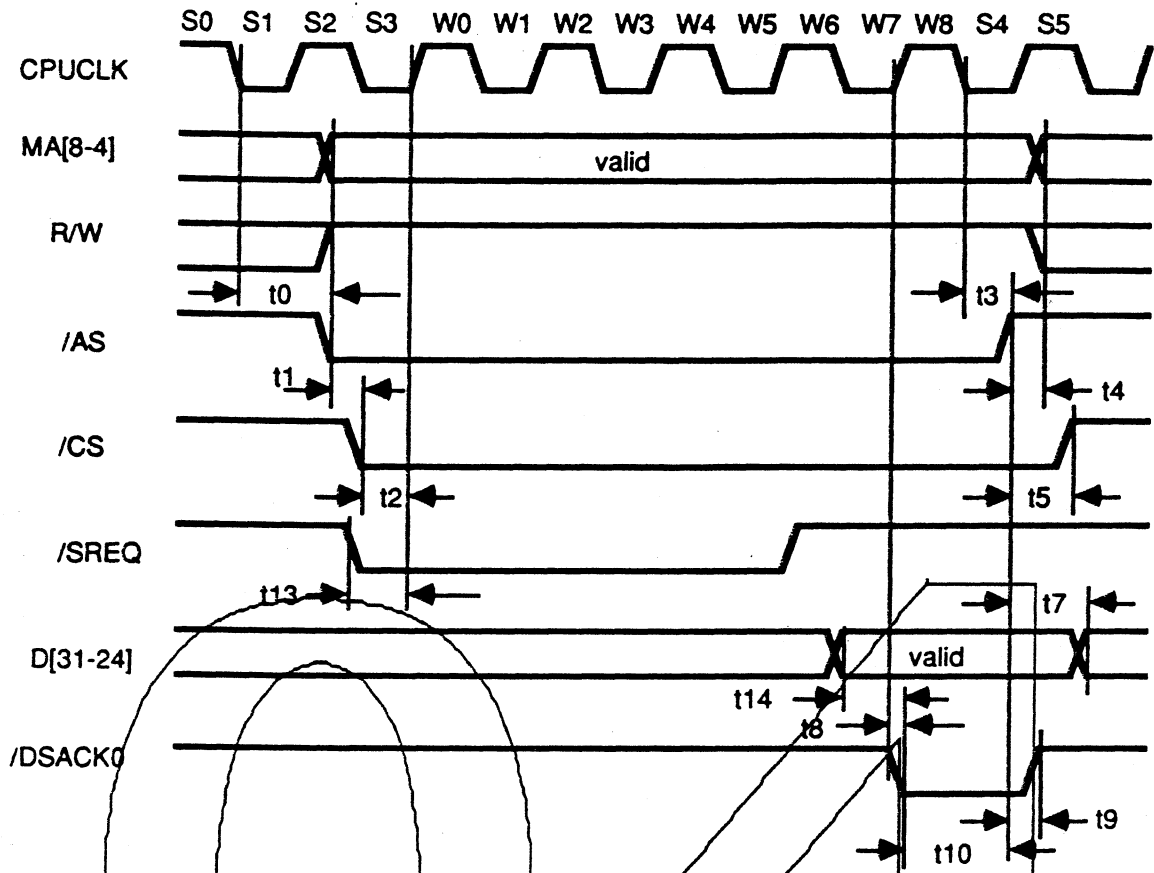


FIGURE 8. PROGRAMMED I/O READ WITH HARDWARE HANDSHAKE FROM 53C80 (READ REGISTER \$060)

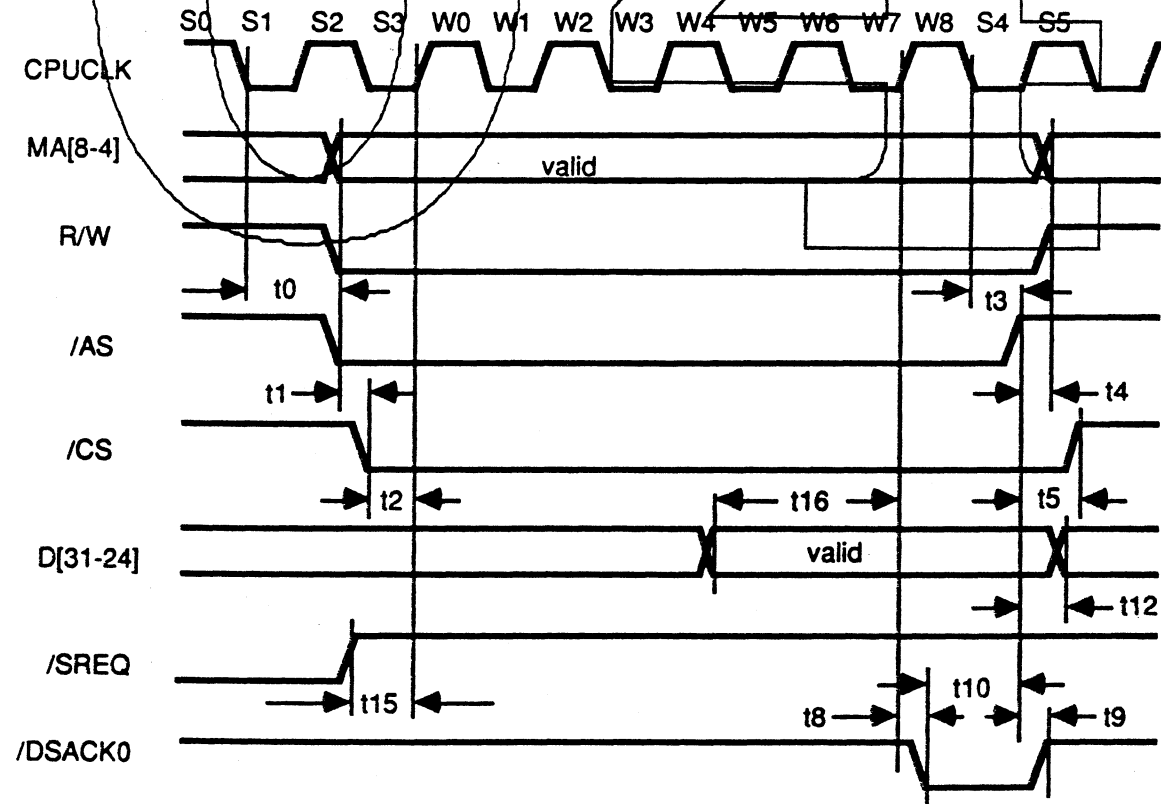


FIGURE 9. PROGRAMMED I/O WRITE WITH HARDWARE HANDSHAKE TO 53C80 (WRITE REGISTER \$000)

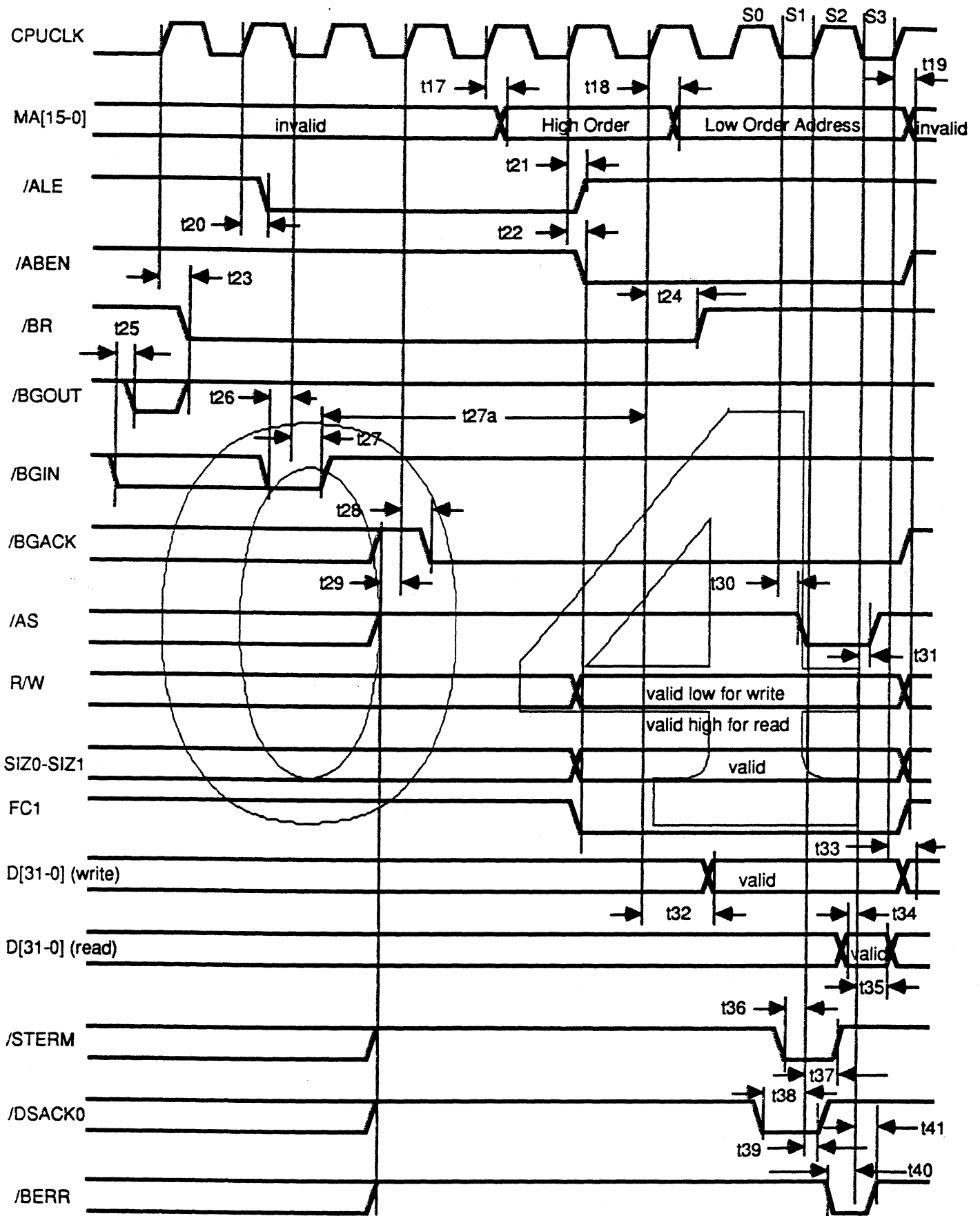


FIGURE 10. DMA READ, WRITE TIMING

343S0064-A

**6.0 SUPPLEMENTAL INFORMATION:** The following is not to be used for the acceptance nor rejection of the part herein.

The SCSI DMA chip is intended to provide a high speed interface from the 68030 bus to the ANSI Small Computer Systems Interface (SCSI) bus. It provides all the features of, and supports the software written for the 53C80 chip and in addition handles data transfers via DMA to the 68030 host processor. Only data transfers are handled by the DMA channel; other SCSI bus protocol is handled in software running on the host processor. During Programmed I/O (PIO) mode, control and command bytes are passed to the SCSI DMA from the host at one byte per transfer. Using DMA mode, data transfers are normally 4 bytes (32 bits) per transfer. Two eight bit registers (74F574 for example) are required external to the SCSI DMA to allow for multiplexing the address bus and thus reducing the pin count of the SCSI DMA. See Figure 5 for schematic.

Some features of the SCSI DMA include:

- \* 53C80 host adapter cell with enhancements
- \* DMA bypass mode (to remain compatible with current software written for the 53C80)
- \* Hardware handshake mode (bytes are transferred under software control and no polling for available bytes is necessary)
- \* 32 bit DMA data transfers
- \* 4 gigabyte direct addressing range (32-bit DMA address register)
- \* Block transfers of 4 gigabytes supported (32 bit DMA byte counter)
- \* Supports misaligned (non-modulo 4) data buffer addresses into "normal (not Nubus)" memory
- \* Supports non-modulo 4 DMA byte counts
- \* Programmable watchdog timer
- \* 3 MB/sec asynchronous transfer rate
- \* Automatic SCSI bus arbitration ("old" arbitration scheme supported also)

**6.1 THEORY OF OPERATION**

**6.1.1 BLOCK DIAGRAMS:**

**6.1.1.1 Flow Block Diagram:** Refer to Figure 2 (Flow Block Diagram) for the following explanations.

**FIFO Input Mux/Logic:** This multiplexer provides byte routing into the FIFO. The data going into the FIFO either comes from the data bus or from the 53C80.

**DMA Data Out Logic:** These buffers provide byte routing from the FIFO onto the processor data bus.

**FIFO Logic:** The FIFO is used to assemble or disassemble 4 byte longwords from, or into bytes.

**PIO Buffer:** This buffer allows control or status information to be transferred directly to the 53C80 cell.

**53C80 Cell:** This is an enhancement of the familiar 53C80 circuit. It will allow transfers of up to 3 megabyte/second and will also provide automatic SCSI bus arbitration.

**DMA Address Counter:** This counter contains the DMA address.

**Address Multiplexing Buffers:** These buffers allow the multiplexing of the 32 bit address into two 16 bit halves.

**Address Readback Buffer:** This buffer allows the DMA address to be read by the CPU.

**DMA Count Counter:** This counter contains the number of bytes left to transfer in the current DMA operation.

**Watchdog Timer Register:** This register contains the initial value of the time delay of the watchdog timer.

**Watchdog Timer Counter:** This counter contains the current value of the time delay of the watchdog timer.

**Control State Machine:** This circuitry controls all operations of the SCSI DMA chip.

**6.1.1.2 Detailed Block Diagram:** Refer to Figure 3 (Detailed Block Diagram) for the following explanations. The Detailed Block Diagram notes the main logic blocks and their interconnections.

**Decoder:** Based on state bits Q~(2:0) and other inputs, this block decodes information to supply control signals to other blocks.

**State Machine:** Design's single state machine that supplies state bits Q~(2:0), control signals, and chip outputs.

**Watchdog Timer:** Performs Watchdog Timer logic. Contains Watchdog Timer Register (Register \$140).

**FIFO:** Performs FIFO logic. Contains FIFO Register (Register \$180).

**DMA Decoder:** Based on state bits Q~(2:0) and other inputs, this block decodes information to supply DMA control signals to other blocks.

**DMA Control Register:** Performs logic to update DMA Control Register (Register \$080), which is also contained in this block.

**DMA Address:** Performs logic to control DMA address functions. Contains DMA Address Register (Register \$100).

**DMA Counter:** Performs logic to control DMA byte counter functions. Contains DMA Count Register (Register \$0C0).

**53C80 Cell:** Enhanced 53C80 cell included in SCSI DMA. Performs all SCSI transactions. Contains Registers \$070 to \$000.

**Misc. 1:** Miscellaneous logic external to major blocks listed above.

**Misc. 2:** Miscellaneous logic external to major blocks listed above.

## 6.2 MODES OF OPERATION

The SCSI DMA may be operated in four modes—Slave (PIO) Mode, Master (DMA) Mode, Test Mode, and Reset Mode.

**6.2.1 Slave (PIO) Mode Operation:** In Slave Mode, the SCSI DMA operates much like a 53C80 with five extra registers. The 68030 is the bus master and accesses the SCSI DMA's registers by writing to or reading from the system-defined address space of the SCSI DMA, which, through separate system logic, will assert the SCSI DMA's /CS input. The SCSI DMA looks like an asynchronous (/DSACK-driving) memory controller to the 68030 during Slave Mode. Slave Mode is utilized to perform PIO (Programmed I/O) register set-up and data transfers.

**6.2.2 Master (DMA) Mode Operation:** When the SCSI DMA operates in Master (DMA) Mode, it becomes the bus master and operates much like a less-powerful 68030. To perform DMA transfers, the SCSI DMA assumes bus mastership through the regular 68030 bus arbitration procedures and then carries out the data transfers between main memory and the SCSI Bus without 68030 assistance, thereby freeing the 68030 to perform other tasks.

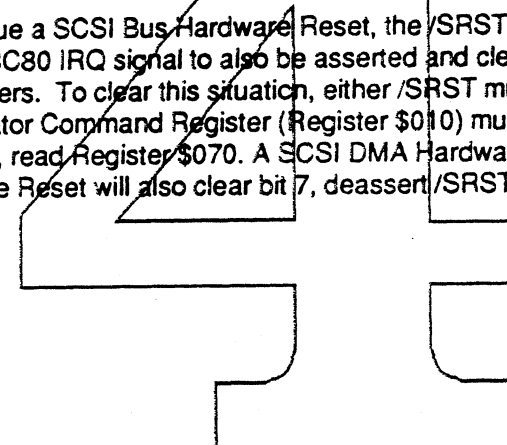
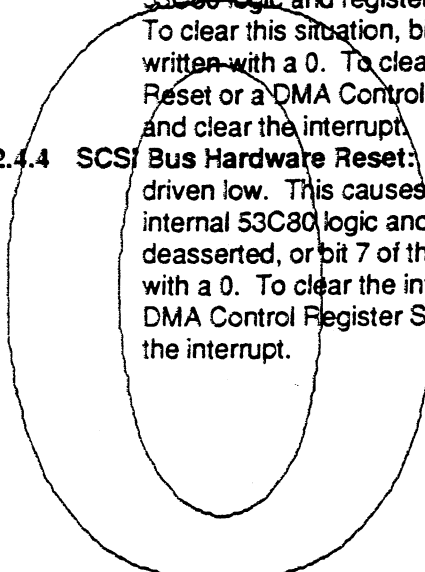
**6.2.3 Test Mode:** The SCSI DMA supports three test modes: 53C80 Isolation Test, Tri-State on Register 1 bit 6 Assertion Test, and FIFO Loopback/Internal Counters Test.

**6.2.3.1 53C80 Isolation Test:** To put the SCSI DMA into 53C80 Isolation Test mode, the TEST input must be high. In this mode, the pinout of the chip is redefined to bring internal 53C80 signals to the pads to allow 53C80 isolation tests to be performed. (See Table 3 for 53C80 Isolation Test Pinout.) This test mode is designed for use by the chip manufacturer.

**6.2.3.2 Tri-State on Register 1 bit 6 Assertion Test:** Tri-State on Register 1 bit 6 Assertion Test occurs by writing bit 6 of Register \$010 with a 1. This causes all output lines on the SCSI DMA to be held in a high impedance state until bit 6 of Register \$010 is written with a 0.

**6.2.3.3 FIFO Loopback/Internal Counters Test:** The SCSI DMA is put into FIFO Loopback/Internal Counters Test mode by writing bit 5 of Register \$080 (DMA Control Register) with a 1. This mode allows each bit of Register \$180 (FIFO Register) to be written and read. In normal operation, one can only read the contents of the FIFO Register. Also in this mode, the DMA Count, DMA Address, and Watchdog counters are connected such that each nibble of each counter will count independently and will be incremented or decremented (depending on the counter—the DMA Count and Watchdog counters decrement while the DMA Address counter increments) by a read of their respective initial-value-holding registers (Registers \$0C0, \$100, or \$140).

- 6.2.4 Reset Mode:** The SCSI DMA may be reset in four ways: SCSI DMA Hardware Reset, DMA Control Register Software Reset, Initiator Command Register Software Reset, and SCSI Bus Hardware Reset.
- 6.2.4.1 SCSI DMA Hardware Reset:** To reset the entire SCSI DMA chip (including the 53C80 cell) the /RESET signal must be low for at least 80 ns. Logic is included to delay the reset input to the 53C80 cell to assure that it will meet its 100 ns minimum reset-assertion-time requirement. This reset mode resets all parts of the SCSI DMA's DMA logic and registers and issues a hardware reset to the 53C80 cell.
- 6.2.4.2 DMA Control Register Software Reset:** By writing bit 4 of the DMA Control Register (Register \$080) with a 1, the DMA Control Register Software Reset is employed. This reset mode essentially performs a 53C80 hardware reset, in that it causes the RESET input to the 53C80 cell to be held active long enough to meet its 100 ns minimum reset-assertion-time requirement. It is not necessary to clear bit 4 of the DMA Control Register after performing a DMA Control Register Software Reset; the 53C80 cell will only be hardware reset once per assertion of bit 4 of the DMA Control Register.
- 6.2.4.3 Initiator Command Register Software Reset:** An Initiator Command Register Software Reset is executed by writing bit 7 of the Initiator Command Register (Register \$010) with a 1. This asserts the /SRST line and the 53C80 IRQ signal, but clears all other internal 53C80 logic and registers; this essentially issues a SCSI Bus Reset through software. To clear this situation, bit 7 of the Initiator Command Register (Register \$010) must be written with a 0. To clear the interrupt, read Register \$070. A SCSI DMA Hardware Reset or a DMA Control Register Software Reset will also clear bit 7, deassert /SRST, and clear the interrupt.
- 6.2.4.4 SCSI Bus Hardware Reset:** To issue a SCSI Bus Hardware Reset, the /SRST line must be driven low. This causes the 53C80 IRQ signal to also be asserted and clears all other internal 53C80 logic and registers. To clear this situation, either /SRST must be deasserted, or bit 7 of the Initiator Command Register (Register \$010) must be written with a 0. To clear the interrupt, read Register \$070. A SCSI DMA Hardware Reset or a DMA Control Register Software Reset will also clear bit 7, deassert /SRST, and clear the interrupt.



### 6.3 SCSI HOST ADAPTER

An enhanced 53C80 is the SCSI host adapter. The enhancements included in this cell over the regular 53C80 are autoarbitration capability and improved asynchronous data transfer. This cell handles all SCSI bus protocol and data transfers. It is accessed directly by the host for all command and control (PIO) transfers to or from the SCSI bus. The SCSI DMA has the DMA capability to offload the host from the tedious job of transferring data byte by byte under direct processor control, although it supports this means of transfer. Refer to the 53C80 manual for details on the operation of the host adapter cell.

### 6.4 BYTE ROUTING AND FIFO LOGIC

The purpose of the byte router logic is to interface between the 32 bit CPU data bus and the 8 bit SD (SCSI Data) bus. The SD bus is an internal 8 bit data bus connected to the 53C80 cell. The byte router logic provides support for misaligned DMA data transfers, programmed I/O (PIO) transfers directly to the 53C80, and a one long word FIFO for DMA data transfers.

### 6.5 DMA CHANNEL

**6.5.1 Signal Timing Scenario:** The SCSI DMA can do all data transfers via direct memory access (DMA) to the processor (host) bus. After all control information is written to load up the DMA address counter and the DMA byte count and set up the 53C80 to do data transfers, the DMA operation will begin by writing to one of the DMA start registers. (See section 6.7.2.3 for details). The 53C80 requests each DMA transfer (when the FIFO is full) via the internal signal DREQ, and that request is passed to the DMA control logic. The SCSI DMA then asserts /BR (Bus Request) in order to gain bus mastership of the 68030 bus. If the processor intends to release the bus after the current bus cycle has completed, it asserts the signal /BG (bus grant) and this signal is passed to all bus masters via a daisy chain. The /BG signal will enter the SCSI DMA via the pin /BGIN. The control logic recognizes /BGIN, and after the current bus cycle has completed, the SCSI DMA asserts /BGACK to hold the bus. As soon as it recognizes that the bus is obtained, the SCSI DMA gates the high order DMA address A[31-16] on the lines MA[15-0] and asserts /ALE. The required external register (74F574) will latch this address. The SCSI DMA then asserts /ABEN and gates the low order DMA address on the lines MA[15-0]. The signal /ABEN is used by the external register to enable the high order address onto the address bus A[31-16]. The SCSI DMA will then drive R/W, FC1, SIZ0, SIZ1, and /AS. If a read (DMA read from memory, write to SCSI) operation is requested, the SCSI DMA will latch the data from the processor data bus. If a write (DMA write to memory, read from SCSI) operation is requested, the SCSI DMA will gate the data onto the processor data bus. It is intended that all DMA transfers with memory be synchronous on the 68030 bus (/STERM supplied by memory controller). However, the signals /DSACK0 and /BERR will also terminate a DMA data transfer.

**6.5.2 Watchdog Timer:** The watchdog timer can be used to monitor the DMA activity. If no DMA activity happens before it counts down, it will interrupt the CPU (if watchdog timer interrupts are enabled - DMA Control Register bit 2 asserted) and terminate the DMA operation. Any DMA activity causes the timer to be reloaded with the value stored in the Watchdog Timer Register (Register \$140).

**6.5.3 Non-Modulo 4 Transfers:** The method by which the SCSI DMA handles non-modulo 4 addresses and non-modulo 4 byte counts is by adjusting for non-modulo 4 addresses on the first 68030 bus transfer and adjusting for non-modulo 4 byte counts on the last 68030 bus transfer. The address is detected to be non-modulo 4 if either or both of the least significant two address bits are non-zero. The value of these address bits determines which of the byte lanes are to be filled. The byte count is decremented by one and the address is incremented by one each time a byte is transferred to or from the 53C80. When the most significant byte lane has been filled, a 68030 DMA transfer is initiated and the size bits will be determined by the number of bytes in the FIFO. The DMA address which will be gated onto the address bus will be the current address minus the number of bytes in the FIFO. Consider the following examples:



**Example 1: WRITE**

As shown, the byte count and DMA address are changed after each byte is transferred from the 53C80 to the FIFO. The byte count is decremented by one and the address is incremented by one. The bytes are placed in the correct byte lane as they are transferred from the 53C80. The two least significant address bits of the current address determine the byte lane. As each byte is placed in its byte lane, the size counter is incremented by one. This counter contains the number of bytes in the FIFO and the number of bytes to transfer when a DMA transfer is initiated.

Byte Count	DMA Addr	Byte Lane	Size1	Size0	Comments
1001	0101	1- D(23:16)	0	1	MSB's go in LS addresses
1000	0110	2- D(15:8)	1	0	
0111	0111	3- D(7:0)	1	1	-> DMA transfer 3 bytes to addr 0101

0110	1000	0- D(31:24)	0	1	
0101	1001	1- D(23:16)	1	0	
0100	1010	2- D(8:15)	1	1	
0011	1011	3- D(7:0)	0	0	-> DMA transfer 4 bytes to addr 1000

0010	1100	0- D(31:24)	0	1	
0001	1101	1- D(23:16)	1	0	-> DMA transfer 2 bytes to addr 1100

0000 -> Done

Count = 0 -> DMA Complete

**Example 2**

Byte Count	DMA Addr	Byte Lane	Size1	Size0	Comments
0100	0101	1- D(23:16)	0	1	
0011	0110	2- D(15:8)	1	0	
0010	0111	3- D(7:0)	1	1	-> DMA transfer 3 bytes to addr 0101

001 1000 0- D(31:24) 0 1 -> DMA transfer 1 byte to addr 1000

000 -> Done

Count = 0 -> DMA Complete

**Example 3**

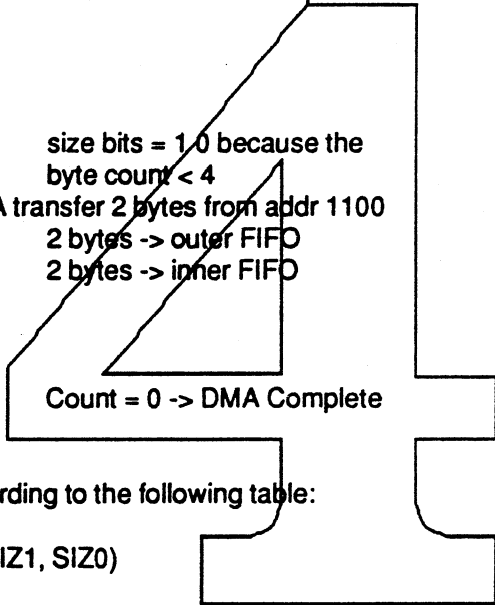
Byte Count	DMA Addr	Byte Lane	Size1	Size0	Comments
0010	0101	1- D(23:16)	0	1	
0001	0110	2- D(15:8)	1	0	-> DMA transfer 2 bytes to addr 0101

000 -> Done

Count = 0 -> DMA Complete

Example 4: READ

Byte Count	DMA Addr	Byte Lane	Size1	Size0	Comments
			1	1	size bits are determined by the starting address
					-> DMA transfer 3 bytes from addr 0101 3 bytes -> outer FIFO 3 bytes -> inner FIFO DMA addr determines byte
1001	0101	1- D(23:16)	0	0	
	lane				
1000	0110	2- D(15:8)	0	0	
0111	0111	3- D(7:0)	0	0	
			0	0	size bits = 0 0 if byte count > 4
					-> DMA transfer 4 bytes from addr 1000 4 bytes -> outer FIFO 4 bytes -> inner FIFO
0110	1000	0- D(31:24)	0	0	
0101	1001	1- D(23:16)	0	0	
0100	1010	2- D(15:8)	0	0	
0011	1011	3- D(7:0)	1	0	
			1	0	size bits = 1 0 because the byte count < 4
					-> DMA transfer 2 bytes from addr 1100 2 bytes -> outer FIFO 2 bytes -> inner FIFO
0010	1100	0- D(31:24)	0	1	
0001	1101	1- D(23:16)	0	0	
0000					-> Done



The initial size bits are determined by the starting address according to the following table:

Addr (A[1:0])	Size (SIZ1, SIZ0)
01	11
10	10
11	01
00	00

Size Output Encodings

Siz1	Siz0	Size
0	1	Byte
1	0	Word
1	1	3 Byte
0	0	Long Word

## 6.6 MEMORY MAP AND PROGRAMMERS MODEL

Refer to Figure 11 for register memory map assignments. The memory map presupposes that the signal /CS is asserted.

### 6.6.1 SCSI HOST ADAPTER REGISTERS (Registers \$000 - \$070)

The byte-wide 53C80 registers are mapped to address bits MA[8-4] = \$00 to \$07.

Refer to the 53C80 design manual for a detailed description of each bit's function.

### 6.6.2 FIFO REGISTER (Register \$180)

This 32 bit register contains the FIFO contents. In normal operation it may only be read. In FIFO Loopback/Internal Counters Test Mode, it may be both read and written.

### 6.6.3 WATCHDOG TIMER REGISTER (Register \$140)

This register is written with the watchdog timer count. The watchdog counter is loaded with this count and begins to count down immediately at a rate of 1/2 the frequency of the CPUCLK input. During each DMA transfer the counter is reloaded with the count. If no DMA operation occurs, no reload will take place and the watchdog counter will count down to zero. After it has counted to zero it will not continue to count. If the watchdog counter counts down to zero, it will set the WD Timer Interrupt Pending bit (bit 7 of the DMA Control Register) and generate an interrupt if WD Timer Interrupts are Enabled bit (bit 2 of the DMA Control Register) is set. Watchdog timer interrupts may be cleared by reading or writing the watchdog counter register (register \$140).

### 6.6.4 DMA ADDRESS REGISTER (Register \$100)

This 32 bit register contains the current DMA address. It may be read or written.

### 6.6.5 DMA BYTE COUNT REGISTER (Register \$0C0)

This 32 bit register contains the current DMA byte count. It may be read or written.

### 6.6.6 DMA CONTROL REGISTER (Register \$080)

This 32 bit register contains the Control and configuration information for the SCSI DMA. The 32 bit DMA Control Register bit definitions are as follows:

BIT	NAME	READ/WRITE	DESCRIPTION
0	DMA Enabled	R/W	low-DMA operations disabled, high-DMA enabled
1	SCSI Interrupts Enabled	R/W	low-interrupts from DMA operations or from the SCSI bus disabled, high-interrupts enabled
2	WD Timer Interrupts Enabled	R/W	low-watchdog timer interrupts disabled, high-interrupts enabled
3	Hardware Handshake Mode	R/W	high-set hardware handshake mode, low-disable hardware handshake mode
4	53C80 Reset/Bytes Left in FIFO	R/W	write only: high-reset the 53C80 (internal cell) (does not tri-state the SCSI DMA)
5	FIFO Loopback/Internal Counters Test Mode	R/W	read only: high-indicates that bytes are left in the FIFO high-enable FIFO Loopback/Internal Counters Test mode, low-disable FIFO Loopback/Internal Counters Test mode.
6	SCSI Interrupt Pending	R only	high-an interrupt from the SCSI bus is pending or the DMA operation has terminated successfully
7	WD Timer Interrupt Pending	R only	high-an interrupt from the watchdog timer is pending
8	DMA halted by bus error	R only	high-the requested DMA operation has been halted due to a bus error (clear by writing DMACT)
9-11	ID0-2	R/W	These bits are used as our SCSI Address during the arbitration phase.
12	ARBEN	R/W	Enable autoarbitration. (See section 6.7.1 for further information on autoarbitration)
13	WONARB	R only	Indicates that arbitration has been won by the SCSI DMA.
14-31	(reserved for future expansion)		

## Programmer's Model

Each register of the SCSI DMA controller is shown below with its address (A[8-0]) and size (number of bits).

### DMA REGISTERS:

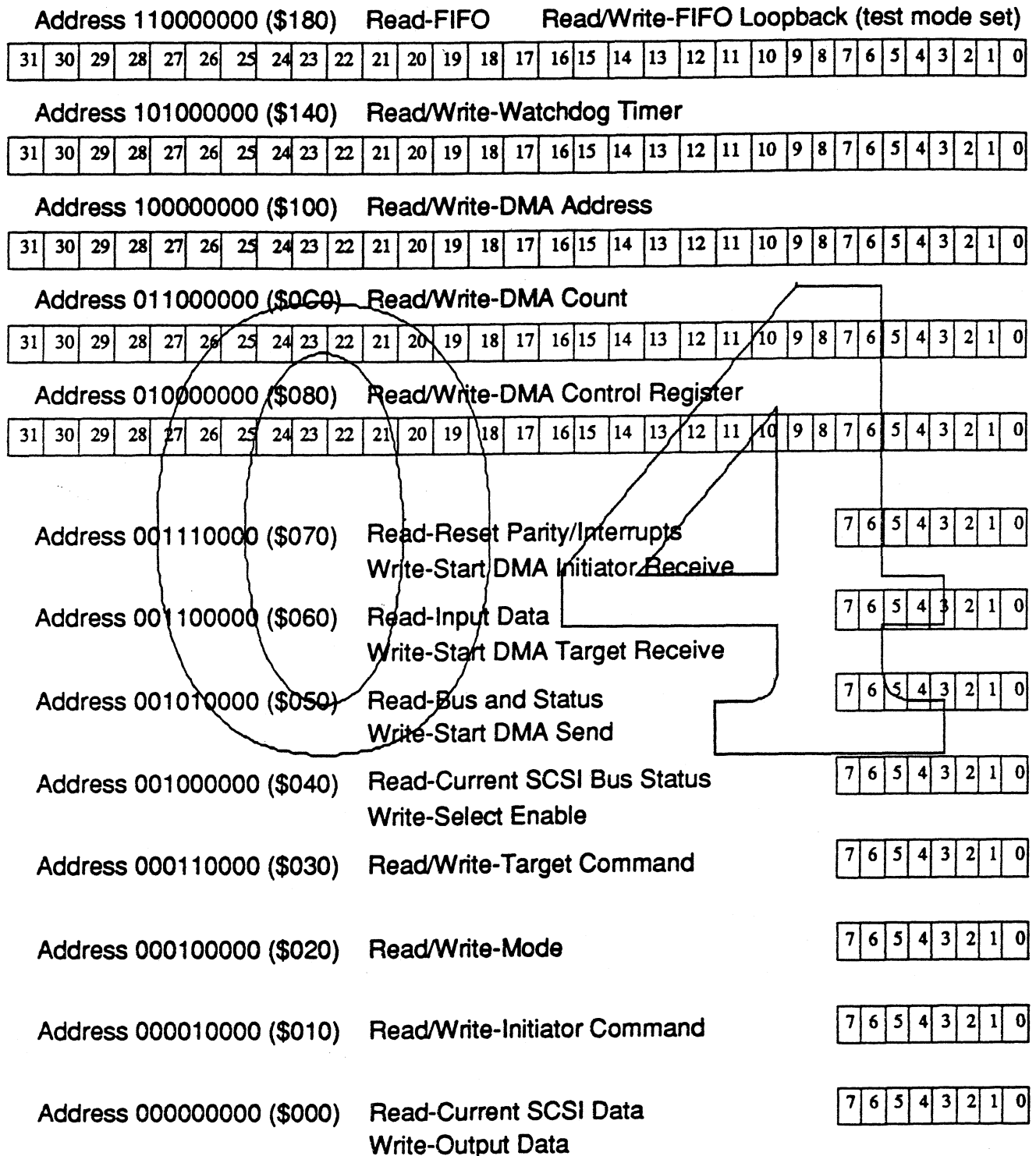


FIGURE 11. MEMORY MAP

## 6.7 Autoarbitration and Data Transfer with the SCSI DMA (Software Enhancements Guide)

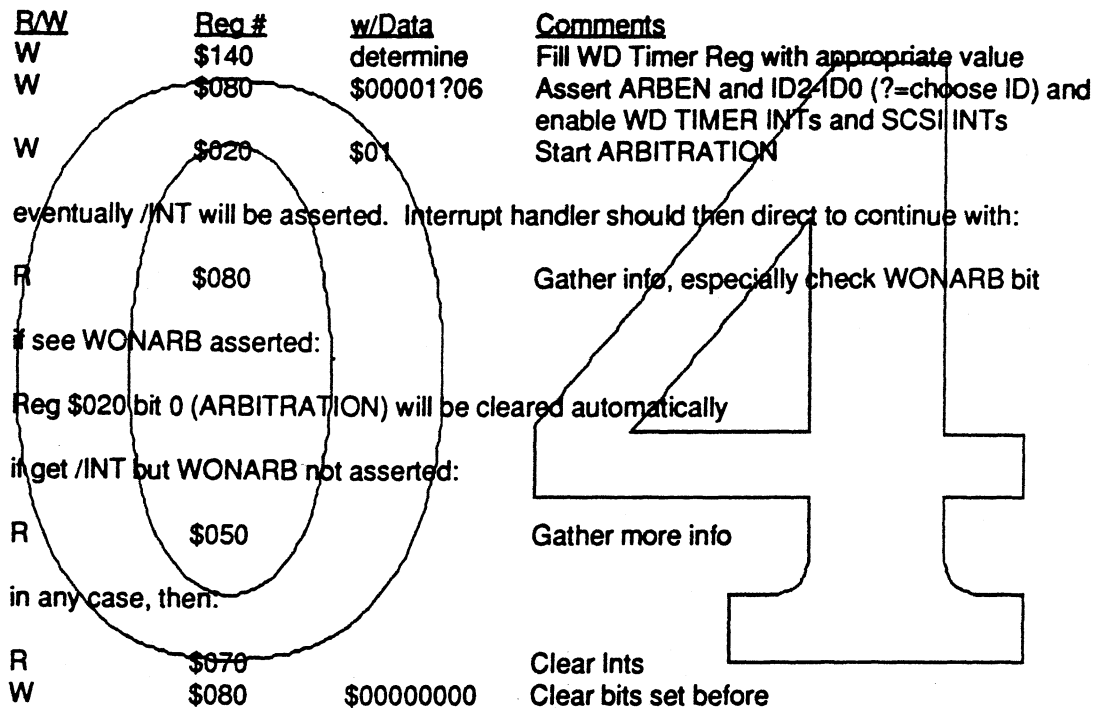
The two main advantages that the SCSI DMA offers over a single 53C80 chip are:

- (1) Autoarbitration and
- (2) DMA Data Transfer

The "old" methods of arbitration and data transfer (explained in the 53C80 Design Manual) are also supported in the SCSI DMA. Noted below is a guide for the software enhancements needed to take advantage of the SCSI DMA's capabilities for autoarbitration and DMA data transfer. Also discussed are the "old" methods of data transfer: Polled Mode and Hardware Handshake Mode.

Note: This is only a software guide and may not include all necessary code.

**6.7.1 Autoarbitration:** The enhanced 53C80 cell included in the SCSI DMA has logic to perform automatic arbitration (autoarbitration). To utilize this feature to perform the Arbitration Phase, the following is suggested:



**6.7.2 Data Transfer:** The SCSI DMA may be configured to perform SCSI data transfers in three ways: Polled Mode, Hardware Handshake (a.k.a. pseudo DMA and blind) Mode, and DMA Mode. Note that these modes of operation refer only to the way data bytes are transferred to or from the SCSI Bus. All control information is transferred to or from the SCSI Bus via the 8 byte-wide 53C80 registers no matter which mode of operation is chosen.

**6.7.2.1 Polled Mode:** The Polled Mode uses software polling to determine when a byte is available to be read from or written to the SCSI DMA. Refer to the 53C80 Design Manual for further information on Polled Mode data transfer. No special register setup is needed to perform Polled Mode transfer.

**6.7.2.2 Hardware Handshake Mode:** The Hardware Handshake Mode does not poll to see when a byte is available; rather it transfers bytes "blindly". In this mode, when a MOVE command is executed by the 68030, the SCSI DMA will respond by asserting /DSACK0 only after the byte transfer has been executed on the SCSI Bus. If the transfer does not occur for some reason, the 68030 bus cycle will be terminated by a bus error generated by a system bus cycle timer (external to the SCSI DMA). To employ Hardware Handshake transfers, bit 3 of the DMA Control Register (Register \$080) must be written with a 1.

Suggested code progression for Hardware Handshake (HWHS) is as follows:

For HWHS Writes (data from main memory to SCSI Bus):

R/W	Req #	w/Data	Comments
W	\$080	\$00000008	HWHS En
W	\$020	\$02	DMA Mode
W	\$010	<del>\$09</del> \$01	Assert SCSI Data Bus, <del>Assert Busy</del>
W	\$030	\$00	Make sure IO not asserted-for phase match
W	\$050	\$XX	Start DMA Send

W \$080 00  
020 00

Repeat until all data is transferred

W	\$000	data	Data from processor bus; will get to SCSI
---	-------	------	---

For HWHS Reads (data from SCSI Bus to main memory):

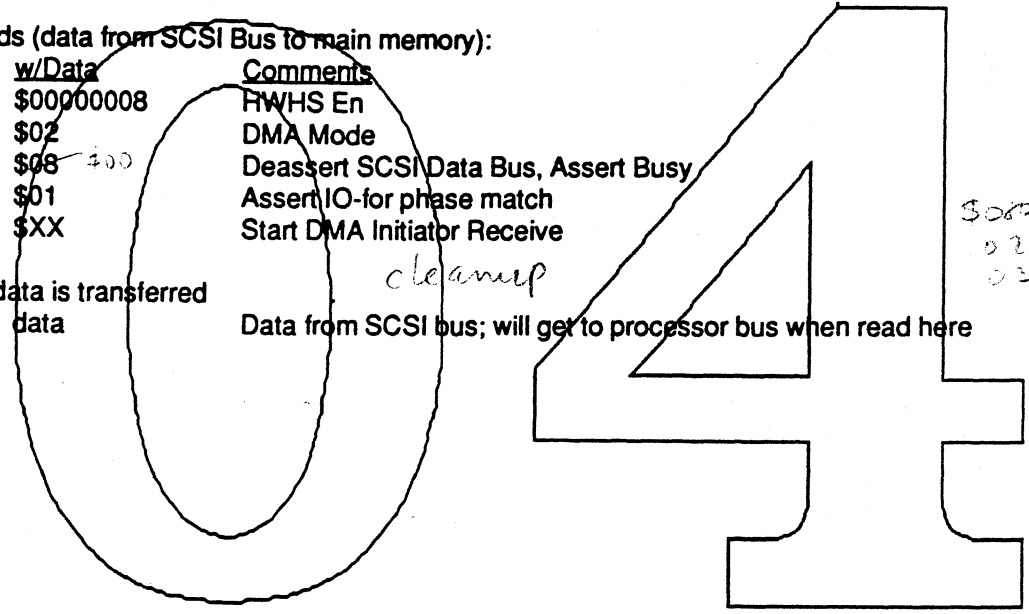
R/W	Req #	w/Data	Comments
W	\$080	\$00000008	HWHS En
W	\$020	\$02	DMA Mode
W	\$010	<del>\$08</del> \$00	Deassert SCSI Data Bus, Assert Busy
W	\$030	\$01	Assert IO-for phase match
W	\$070	\$XX	Start DMA Initiator Receive

as on  
next page

Repeat until all data is transferred

R	\$060	data	Data from SCSI bus; will get to processor bus when read here
---	-------	------	--

\$080 20-0  
020 00  
030 00



6.7.2.3 DMA Mode: The DMA Mode transfers bytes to and from main memory independent of the 68030.

To put the SCSI DMA into DMA Mode, the following must be done:

*writes*

**For DMA Reads (data from main memory to SCSI Bus):**

R/W	Reg #	w/Data	Comments
W	\$140	determine	Fill WD Timer Reg with appropriate value
W	\$080	\$00000007	WD Timer Ints En, SCSI Ints En, DMA En
W	\$100	starting addr	Set DMA Addr Reg with starting address
W	\$0C0	byte count	Set DMA Count Reg with # of bytes to transfer
W	\$020	\$0A	En EOP Int, DMA Mode
W	\$010	\$09 <i>3)</i>	Assert SCSI Data Bus, <del>Assert Busy</del>
W	\$030	\$00	Make sure IO not asserted-for phase match
W	\$050	\$XX	Start DMA Send

after have received interrupt from SCSI DMA:

R	\$080		Check bits 8-6 to determine reason for int
R	\$050		Check reason for SCSI interrupt
R	\$070		Clear 53C80 interrupt
W	\$020	\$00	Clear bits set before
W	\$080	\$00000000	Clear bits set before
W	\$010	\$00	Clear bits set before

*CLEANUP*

*reads*

**For DMA Writes (data from SCSI Bus to main memory):**

R/W	Reg #	w/Data	Comments
W	\$140	determine	Fill WD Timer Reg with appropriate value
W	\$080	\$00000007	WD Timer Ints En, SCSI Ints En, DMA En
W	\$100	starting addr	Set DMA Addr Reg with starting address
W	\$0C0	byte count	Set DMA Count Reg with # of bytes to transfer
W	\$020	\$0A	En EOP Int, DMA Mode
W	\$010	\$08 <i>400</i>	Deassert SCSI Data Bus, <del>Assert Busy</del>
W	\$030	\$01	Assert IO-for phase match
W	\$070	\$XX	Start DMA Initiator Receive

after have received interrupt from SCSI DMA:

R	\$080		Check bits 8-6 to determine reason for int
R	\$050		Check reason for SCSI interrupt
R	\$070		Clear 53C80 interrupt
W	\$020	\$00	Clear bits set before
W	\$080	\$00000000	Clear bits set before
W	\$010	\$00	Clear bits set before
W	\$030	\$00	Clear bits set before

- 6.8 SCSI DMA Interrupts:** Interrupts from the SCSI DMA appear at the /INT signal. Interrupts may be generated from the 53C80 (SCSI Interrupts) or from the DMA logic (WD Timer Interrupts).
- 6.8.1 SCSI interrupts:** The following interrupts originate from the 53C80 SCSI host adapter. Noted below are the situations that will cause an interrupt request to be generated, how the situation occurs, and the steps required to enable that request to appear at the /INT output. To enable all of the following SCSI Interrupts to appear at the /INT output, one must assert Register \$080 bit 1 (SCSI Interrupts Enabled). This allows the internal signal IRQ (interrupt request from the 53C80 cell) to get to the /INT output. A read of Register \$080 bit 6 (SCSI Interrupt Pending) will indicate whether the IRQ has reached the DMA logic side to be propagated to the /INT signal. To clear IRQ (and therefore /INT), read Register \$070 or assert the SCSI DMA's /RESET signal.
- 6.8.1.1 Assertion of /SRST signal:**  
 Assert the /SRST signal or assert Register \$010 bit 7 (Assert /SRST). This will assert /SRST, release all SCSI Bus signals, and generate an IRQ.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.
- 6.8.1.2 Parity Error:**  
 Assert Register \$020 bits 5, 4, and 1 (Enable Parity Checking, Enable Parity Interrupt, and DMA Mode—for DMA Receive situation). Parity Error is checked on read of Register 0 and write of Register \$070 and if a parity error occurs, an IRQ will be generated.  
 Read Register \$050 bit 5 (Parity Error—should be 1) to verify.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.
- 6.8.1.3 /EOP (End of Process):**  
 Assert Register \$020 bits 3 and 1 (Enable EOP Interrupt and DMA Mode). If /EOP (internal signal) is asserted and /DACK (internal signal) and either /IOR (internal signal) or /IOW (internal signal) are also asserted for 100ns, an IRQ will be generated.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.
- 6.8.1.4 Loss of Busy:**  
 Assert Register \$020 bit 2 (Monitor Busy). A Loss of Busy Error occurs if /SBSY goes inactive for 400 ns while Register \$020 bit 2 is asserted. If this error occurs, all SCSI Outputs will be tristated and Register \$020 bit 1 (DMA Mode) will be automatically cleared and an IRQ will be generated.  
 Read Register \$050 bit 2 (Busy Error—should be 1) to verify.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.
- 6.8.1.5 Phase Mismatch:**  
 Assert Register \$020 bit 1 (DMA Mode). A Phase Mismatch Error occurs if Register \$030 bits 2-0 do not match Register \$040 bits 4-2. IRQ will be generated on the falling edge of /SREQ if there is a Phase Mismatch Error and Register \$050 (DMA Send) has been written. If this error occurs, all SCSI outputs will be tristated.  
 Read Register \$050 bit 3 (Phase Match—should be 0) to verify.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.
- 6.8.1.6 Chip Selected:**  
 Write Register \$040 with the appropriate device ID. If /SSEL is asserted, /SBSY is deasserted, /SIO is deasserted, and the device ID matches the Register \$040 value for 400 ns, the chip will be selected and will generate an IRQ. Note: if Register \$040 contains all zeros, this interrupt won't be enabled.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.
- 6.8.1.7 Chip Reselected:**  
 Write Register \$040 with appropriate device ID. If /SSEL is asserted, /SBSY is deasserted, /SIO is asserted, and the device ID matches the Register \$040 value for 400 ns, the chip will be reselected and will generate an IRQ. Note: if Register \$040 contains all zeros, this interrupt won't be enabled.  
 Read Register \$050 bit 4 (IRQ active—should be 1) to verify IRQ.



**6.8.1.8 Autoarbitration:**

Follow guidelines from Section 6.7.1 to put the SCSI DMA into autoarbitration. If arbitration is won, IRQ will be asserted.

Read Register \$080 bit 13 (WONARB - should be 1) to verify.

Read Register \$050 bit 4 (IRQ active - should be 1) to verify IRQ.

- 6.8.2 WD Timer Interrupts:** To enable WD Timer Interrupts to appear at the /INT output, one must assert Register \$080 bit 2 (WD Timer Interrupts Enabled). This allows a WD Timer Interrupt request to get to the /INT output. A WD Timer Interrupt is requested whenever the Watchdog Timer counts to zero. A read of Register \$080 bit 6 (WD Timer Interrupt Pending) will indicate whether the WD Timer Interrupt has been posted. To clear a WD-Timer-Interrupt-generated /INT signal, Read or Write Register \$140 (Watchdog Timer Register) or assert the SCSI DMA's /RESET signal.

