

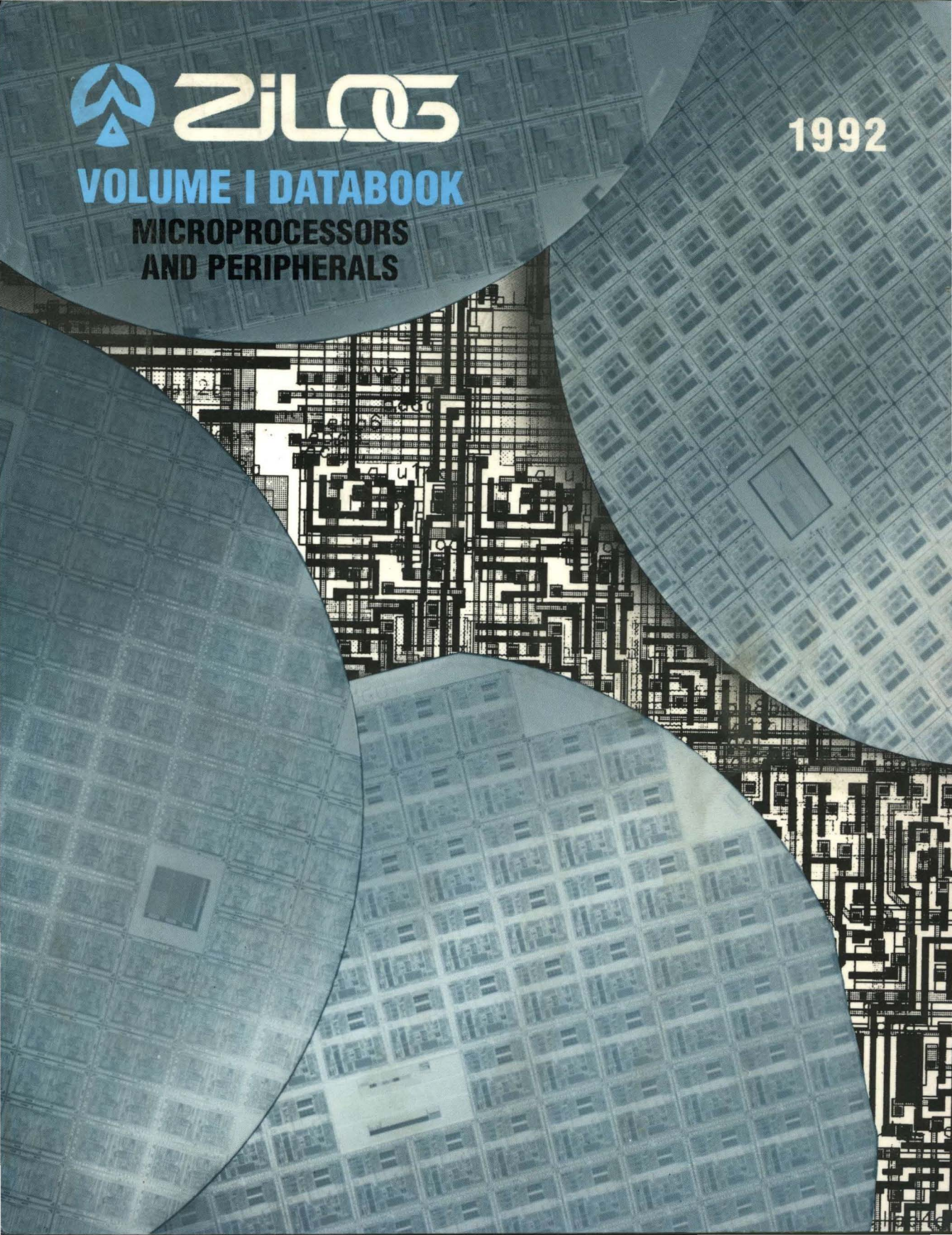


**ZILOG**

**1992**

**VOLUME I DATABOOK**

**MICROPROCESSORS  
AND PERIPHERALS**



---

## Table of Contents

---

<b>Volume I Databook</b>	<b>Page</b>
Introduction .....	1
<b>Discrete Z80® Family .....</b>	<b>3</b>
Z8400/C00 NMOS/CMOS Z80® CPU Product Specification .....	5
Z8410/C10 NMOS/CMOS Z80 DMA Product Specification .....	41
Z8420/C20 NMOS/CMOS Z80 PIO Product Specification .....	67
Z8430/C30 NMOS/CMOS Z80 CTC Product Specification .....	83
Z8440/Z84C40 NMOS/CMOS Z80 SIO Product Specification .....	99
<b>Embedded Controllers .....</b>	<b>123</b>
Z84C01 Z80 CPU with CGC Product Specification .....	125
Z84C50 RAM80™ Preliminary Product Specification .....	157
Z8470 Z80 DART Product Specification .....	195
Z84C90 CMOS Z80 KIO™ Product Specification .....	207
Z84011/C11 PIO Parallel I/O Product Specification .....	231
Z84013/015 Z84C13/C15 IPC/EIPC™ Product Specification .....	293
Z80180/Z8S180 Z180 MPU Product Specification .....	359
Z80181 ZIO™ Controller Product Specification .....	411
Z280™ MPU Preliminary Product Specification .....	483
<b>Serial Communications Controllers .....</b>	<b>547</b>
Z8030/Z8530 Z-BUS® SCC Product Specification .....	549
Z80C30/Z85C30 SCC Product Specification .....	583
Z85230 ESCC™ Product Specification .....	627
Z80230 Z-BUS ESCC Product Specification .....	667
Z16C35 ISCC™ Product Specification .....	709
Z5380 SCSI Product Specification .....	767
Z53C80 SCSI Product Specification .....	803
Z85C80 SCSI/SCC Product Specification .....	841
Z16C30 USC™ Product Specification .....	903
Z16C32 IUSC™ Product Specification .....	979
Z16C33 MUSC™ Product Specification .....	1095
Z16C50 DDPLL™ Product Specification .....	1177

---

## Table of Contents

---

	Page
<b>Technical Articles .....</b>	<b>1183</b>
Z80 Questions and Answers .....	1185
Z180 Questions and Answers .....	1201
SCC Questions and Answers .....	1213
ESCC Questions and Answers .....	1223
ISCC Questions and Answers .....	1225
<b>The Following Application Notes Are Available From Zilog</b>	
Z80® Family Interrupt Structure	
Using the Z80 SIO in Async Communications	
Using the Z80 SIO with SDLC	
Binary Synchronous Communications Using the Z80 SIO	
Serial Communications with the Z80A DART	
Interfacing Z8500 Peripherals to the Z80	
Serial Clock Generation Using the Z8536 CIO	
Timing in Interrupt-Based System with Z80 CTC	
A Z80-Based System Using the DMA with the SIO	
Interfacing Z80 CPUs to the Z8500 Peripheral Family	
Z180™/SCC Serial Communication Controller Interface at 10 MHz	
Local Talk Link Access Protocol Using the Z80181	
Z80 Using the Z84C11/C13/C15 in Place of the Z84011/013/015	
Using SCC with the Z8000® in SDLC Protocol	
SCC in Binary Synchronous Communications	
On-Chip Oscillator Design	
Interfacing the Z8500 Peripherals to the 68000	
ESCC™ Enhancements over the SCC	
Design a Serial Board to Handle Multiple Protocols	
Using the Z16C30 USC Universal Serial Controller with Mil-STD-1553B	
Datacommunications IUSC™/MUSC™ Time Slot Assigner	
Integrating Serial Data and SCSI Peripheral Control on One Chip	
<b>Additional Information .....</b>	<b>1227</b>
Z80® Development Support .....	1229
Superintegration™ Products Guide .....	1232
Support Products Summary .....	1239
Z80®/Z180™/Z280® Hardware and Software Support .....	1299
Military Qualified Products .....	1301
Package Information .....	1303
Ordering Information .....	1311
Quality and Reliability .....	1315
Literature Guide .....	1319



# VOLUME I DATABOOK

## DISCRETE Z80® FAMILY

## EMBEDDED CONTROLLERS

## SERIAL COMMUNICATIONS CONTROLLERS

---

### INTRODUCTION

Zilog is an innovator in the development, design and manufacture of Application Specific Standard Products (ASSPs) for the datacommunications, intelligent peripheral controllers and consumer-product controllers market.

ASSPs are very large-scale integrated circuits designed for a particular market application rather than a single customer. Zilog supplies ASSPs utilizing its Superintegration™ design technology to combine cores and cells from the Company's extensive library of customer familiar microprocessors, microcontrollers, memory and logic circuits to meet the design, cost and time-to-market requirements of its customers.

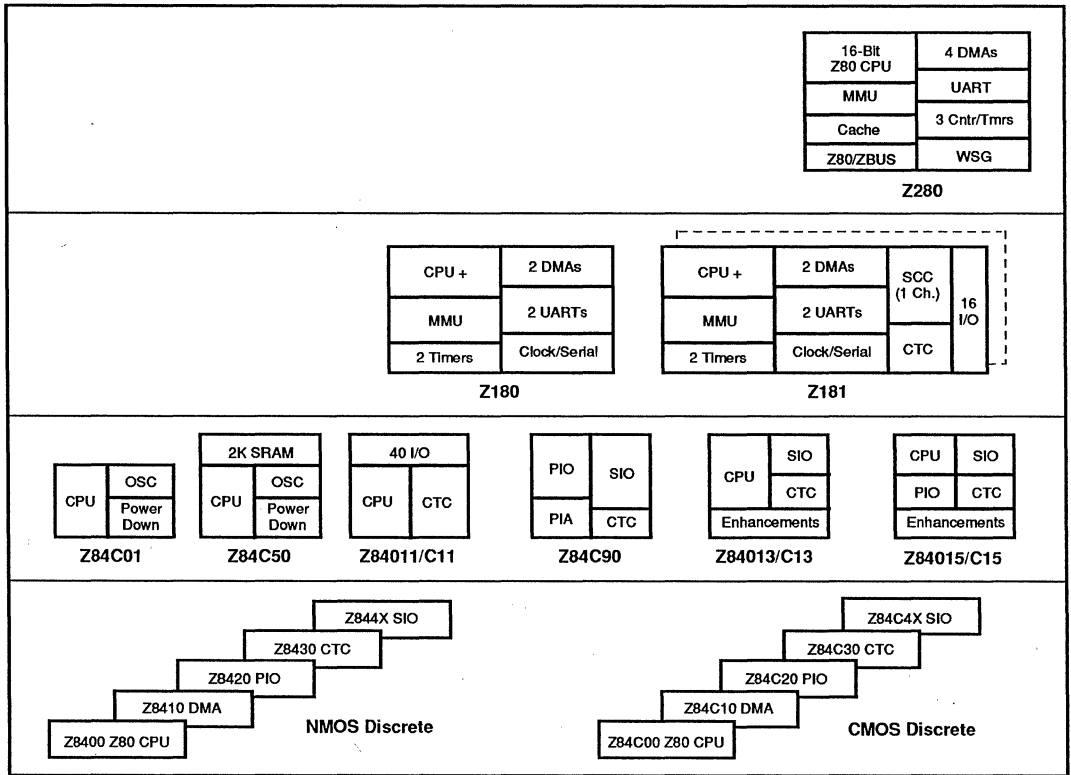
Zilog was an innovator in the addition of intelligence to computer peripheral chips using its popular line of Z80® 8-bit and Z280® 16-bit microprocessors and peripheral circuits. Adding intelligence to computer peripherals frees the central processor from micromanagement tasks and upgrades the performance of the system. Based on the powerful Z8® 8-bit microcontroller, Zilog's family of controllers is used in such diverse products as cellular phones, answering machines, televisions, educational products, and a wide range of computer-related products such as mice, keyboards and mass storage products.

Zilog is rapidly expanding its product offering with more than 19 new products announced in the first three quarters of 1991, yielding more than 40 variations. Zilog's extensive library of cores and cells includes serial communications controllers, 8-bit microcontrollers, 8-, 16- and 32-bit microprocessors and peripheral circuits. The Superintegration library and diverse product portfolio of over 850 items serve three distinct markets. In datacommunications, Zilog holds a leadership position in general purpose, multiprotocol controllers for the local area network markets. The Company offers a range of controllers from the industry standard, medium-speed SCC to the high-performance USC™. Complete families of products are based on these cores.

Included in these introductions is the Z86C94, the industry's first 8-bit microcontroller to add a digital signal processor on chip. The device is used for efficient digital control of servo functions in disk drives. In the consumer area, Zilog introduced a new single-chip closed-caption controller which will position the Company to capture a significant share of the television market.

# Z80 Product Evolution

Performance



Integration



**DISCRETE Z80®  
FAMILY**

# **DISCRETE Z80® FAMILY**



## Z8400/Z84C00 NMOS/CMOS Z80<sup>®</sup> CPU Central Processing Unit

### FEATURES

- The extensive instruction set contains 158 instructions, including the 8080A instruction set as a subset.
- NMOS version for low cost high performance solutions, CMOS version for high performance low power designs.
- NMOS Z0840004 - 4 MHz, Z0840006 - 6.17 MHz, Z0840008 - 8 MHz.
- CMOS Z84C0006 - DC to 6.17 MHz, Z84C008 - DC to 8 MHz, Z84C0010 - DC to 10 MHz, Z84C0020 - DC to 20 MHz
- 6 MHz version can be operated at 6.144 MHz clock.
- The Z80 microprocessors and associated family of peripherals can be linked by a vectored interrupt system. This system can be daisy-chained to allow implementation of a priority interrupt scheme.
- Duplicate set of both general-purpose and flag registers.
- Two sixteen-bit index registers.
- Three modes of maskable interrupts:
  - Mode 0—8080A similar;
  - Mode 1—Non-Z80 environment, location 38H;
  - Mode 2—Z80 family peripherals, vectored interrupts.
- On-chip dynamic memory refresh counter.

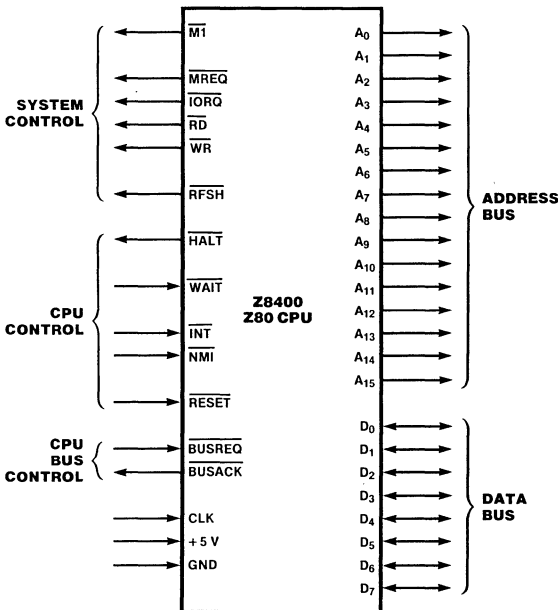


Figure 1. Pin Functions

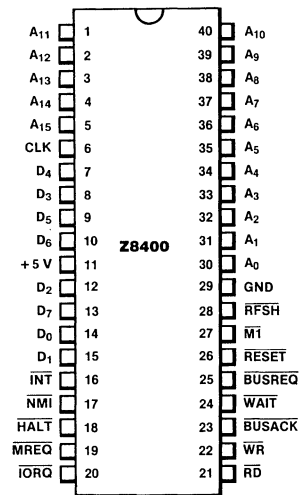
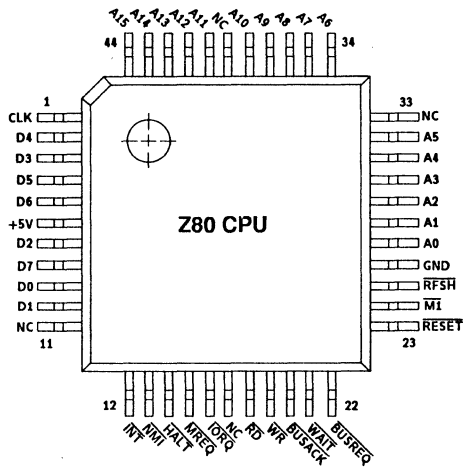
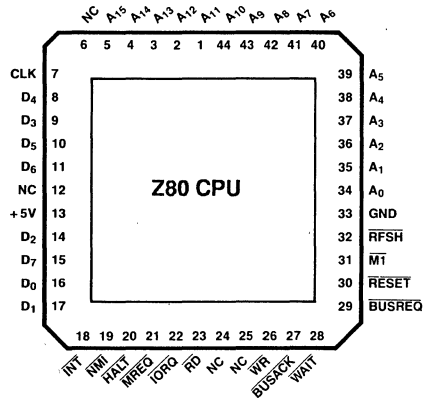


Figure 2. 40-pin Dual-In-Line (DIP), Pin Assignments





**44 pin Quad Flat Pack (QFP), Pin Assignments  
(Only available for 84C00)**

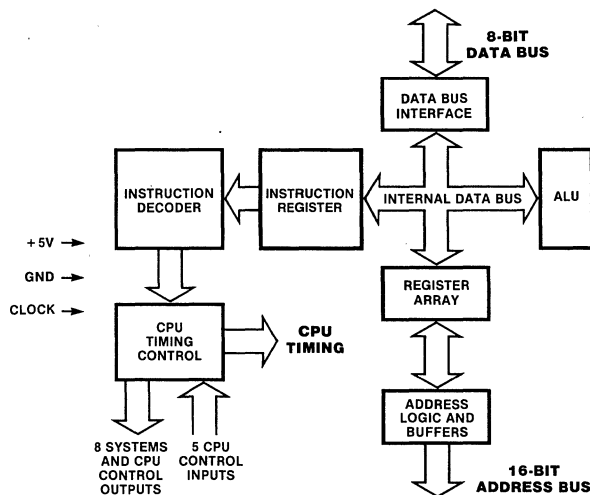


**Figure 2b. 44-Pin Chip Carrier Pin Assignments**

## GENERAL DESCRIPTION

The CPUs are fourth-generation enhanced microprocessors with exceptional computational power. They offer higher system throughput and more efficient memory utilization than comparable second- and third-generation microprocessors. The internal registers contain 208 bits of read/write memory that are accessible to the programmer. These registers include two sets of six general-purpose registers which may be used individually as either 8-bit registers or as 16-bit register pairs. In addition, there are two sets of accumulator and flag registers. A group of "Exchange" instructions makes either set of main or alternate registers accessible to the programmer. The alternate set allows operation in foreground-background mode or it may be reserved for very fast interrupt response.

The CPU also contains a Stack Pointer, Program Counter, two index registers, a Refresh register (counter), and an Interrupt register. The CPU is easy to incorporate into a system since it requires only a single +5V power source. All output signals are fully decoded and timed to control standard memory or peripheral circuits; the CPU is supported by an extensive family of peripheral controllers. The internal block diagram (Figure 3) shows the primary functions of the processors. Subsequent text provides more detail on the I/O controller family, registers, instruction set, interrupts and daisy chaining, and CPU timing.



**Figure 3. Z80C CPU Block Diagram**

Table 1. Z80C CPU Registers

Register	Size (Bits)	Remarks	
A, A'	Accumulator	8	Stores an operand or the results of an operation.
F, F'	Flags	8	See Instruction Set.
B, B'	General Purpose	8	Can be used separately or as a 16-bit register with C.
C, C'	General Purpose	8	Can be used separately or as a 16-bit register with B.
D, D'	General Purpose	8	Can be used separately or as a 16-bit register with E.
E, E'	General Purpose	8	Can be used separately or as a 16-bit register with D.
H, H'	General Purpose	8	Can be used separately or as a 16-bit register with L.
L, L'	General Purpose	8	Can be used separately or as a 16-bit register with H.
Note: The (B,C), (D,E), and (H,L) sets are combined as follows:			
B — High byte    C — Low byte			
D — High byte    E — Low byte			
H — High byte    L — Low byte			
I	Interrupt Register	8	Stores upper eight bits of memory address for vectored interrupt processing.
R	Refresh Register	8	Provides user-transparent dynamic memory refresh. Automatically incremented and placed on the address bus during each instruction fetch cycle.
IX	Index Register	16	Used for indexed addressing.
IY	Index Register	16	Used for indexed addressing
SP	Stack Pointer	16	Holds address of the top of the stack. See Push or Pop in instruction set.
PC	Program Counter	16	Holds address of next instruction.
IFF <sub>1</sub> -IFF <sub>2</sub>	Interrupt Enable	Flip-Flops	Set or reset to indicate interrupt status (see Figure 4).
IMFa-IMFb	Interrupt Mode	Flip-Flops	Reflect Interrupt mode (see Figure 4).

failure has been detected. After recognition of the  $\overline{\text{NMI}}$  signal (providing  $\overline{\text{BUSREQ}}$  is not active), the CPU jumps to restart location 0066H. Normally, software starting at this address contains the interrupt service routine.

**Maskable Interrupt ( $\overline{\text{INT}}$ ).** Regardless of the interrupt mode set by the user, the CPU response to a maskable interrupt input follows a common timing cycle. After the interrupt has been detected by the CPU (provided that interrupts are enabled and  $\overline{\text{BUSREQ}}$  is not active) a special interrupt processing cycle begins. This is a special fetch ( $\overline{\text{M1}}$ ) cycle in which  $\overline{\text{IORQ}}$  becomes active rather than  $\overline{\text{MREQ}}$ , as in a normal  $\overline{\text{M1}}$  cycle. In addition, this special  $\overline{\text{M1}}$  cycle is automatically extended by two  $\overline{\text{WAIT}}$  states, to allow for the time required to acknowledge the interrupt request.

**Mode 0 Interrupt Operation.** This mode is similar to the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus. This is normally a Restart instruction, which will initiate a call

to the selected one of eight restart locations in page zero of memory. Unlike the 8080, the Z80 CPU responds to the Call instruction with only one interrupt acknowledge cycle followed by two memory read cycles.

**Mode 1 Interrupt Operation.** Mode 1 operation is very similar to that for the  $\overline{\text{NMI}}$ . The principal difference is that the Mode 1 interrupt has only one restart location, 0038H.

**Mode 2 Interrupt Operation.** This interrupt mode has been designed to most effectively utilize the capabilities of the Z80 microprocessor and its associated peripheral family. The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit vector on the data bus during the interrupt acknowledge cycle. The CPU forms a pointer using this byte as the lower 8 bits and the contents of the I register as the upper 8 bits. This points to an entry in a table of addresses for interrupt service routines. The CPU then jumps to the routine at that

address. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines. These routines may be located at any available location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 ( $A_0$ ) must be a zero.

**Interrupt Enable/Disable Operation.** Two flip-flops, IFF<sub>1</sub> and IFF<sub>2</sub>, referred to in the register description, are used to signal the CPU interrupt status. Operation of the two flip-flops is described in Table 2. For more details, refer to the *Z80 CPU Technical Manual (03-0029-01)* and *Z80 Assembly Language Programming Manual (03-0002-01)*.

**Table 2. State of Flip-Flops**

Action	IFF <sub>1</sub>	IFF <sub>2</sub>	Comments
CPU Reset	0	0	Maskable interrupt $\overline{\text{INT}}$ disabled
DI instruction execution	0	0	Maskable interrupt $\overline{\text{INT}}$ disabled
EI instruction execution	1	1	Maskable interrupt $\overline{\text{INT}}$ enabled
LD A,I instruction execution	•	•	IFF <sub>2</sub> → Parity flag
LD A,R instruction execution	•	•	IFF <sub>2</sub> → Parity flag
Accept NMI	0	•	Maskable interrupt $\overline{\text{INT}}$ disabled
RETN instruction execution	IFF <sub>2</sub>	•	IFF <sub>2</sub> → IFF <sub>1</sub> at completion of an NMI service routine.

## INSTRUCTION SET

The microprocessor has one of the most powerful and versatile instruction sets available in any 8-bit microprocessor. It includes such unique operations as a block move for fast, efficient data transfers within memory, or between memory and I/O. It also allows operations on any bit in any location in memory.

The following is a summary of the instruction set which shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. For an explanation of flag notations and symbols for mnemonic tables, see the Symbolic Notations section which follows these tables. The *Z80 CPU Technical Manual (03-0029-01)*, the *Programmer's Reference Guide (03-0012-03)*, and *Assembly Language Programming Manual (03-0002-01)* contain significantly more details for programming use.

The instructions are divided into the following categories:

- 8-bit loads
- 16-bit loads
- Exchanges, block transfers, and searches
- 8-bit arithmetic and logic operations
- General-purpose arithmetic and CPU control
- 16-bit arithmetic operations
- Rotates and shifts

- Bit set, reset, and test operations
- Jumps
- Calls, returns, and restarts
- Input and output operations

A variety of addressing modes are implemented to permit efficient and fast data transfer between various registers, memory locations, and input/output devices. These addressing modes include:

- Immediate
- Immediate extended
- Modified page zero
- Relative
- Extended
- Indexed
- Register
- Register indirect
- Implied
- Bit

## 8-BIT LOAD GROUP

Mnemonic	Symbolic Operation	S	Z	Flags				Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments		
				H	P/V	N	C	76	543	210	Hex						
LD r, r'	r ← r'	•	•	X	•	X	•	•	•	01	r	r'	1	1	4	r, r' Reg.	
LD r, n	r ← n	•	•	X	•	X	•	•	•	00	r	110	2	2	7	000 B	
												← n →				001 C	
LD r, (HL)	r ← (HL)	•	•	X	•	X	•	•	•	01	r	110	1	2	7	010 D	
LD r, (IX+d)	r ← (IX+d)	•	•	X	•	X	•	•	•	11	011	101	DD	3	5	19	011 E
												01 r 110				100 H	
												← d →				101 L	
LD r, (IY+d)	r ← (IY+d)	•	•	X	•	X	•	•	•	11	111	101	FD	3	5	19	111 A
												01 r 110					
												← d →					
LD (HL), r	(HL) ← r	•	•	X	•	X	•	•	•	01	110	r	1	2	7		
LD (IX+d), r	(IX+d) ← r	•	•	X	•	X	•	•	•	11	011	101	DD	3	5	19	
												01 110 r					
												← d →					
LD (IY+d), r	(IY+d) ← r	•	•	X	•	X	•	•	•	11	111	101	FD	3	5	19	
												01 110 r					
												← d →					
LD (HL), n	(HL) ← n	•	•	X	•	X	•	•	•	00	110	110	36	2	3	10	
												← n →					
LD (IX+d), n	(IX+d) ← n	•	•	X	•	X	•	•	•	11	011	101	DD	4	5	19	
												00 110 110 36					
												← d →					
												← n →					

## 8-BIT LOAD GROUP (Continued)

Mnemonic	Symbolic Operation	S	Z	Flags			Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments	
				H	P/V	N	C	76	543	210					Hex
LD (IY+d), n	(IY+d) ← n	•	•	X	•	X	•	•	•	11 111 101	FD	4	5	19	
										00 110 110	36				
										← d →					
										← n →					
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	•	•	00 001 010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	•	•	00 011 010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	•	•	00 111 010	3A	3	4	13	
										← n →					
										← n →					
LD (BC), A	(BC) ← A	•	•	X	•	X	•	•	•	00 000 010	02	1	2	7	
LD (DE), A	(DE) ← A	•	•	X	•	X	•	•	•	00 010 010	12	1	2	7	
LD (nn), A	(nn) ← A	•	•	X	•	X	•	•	•	00 110 010	32	3	4	13	
										← n →					
										← n →					
LD A, I	A ← I	‡	‡	X	0	X	IFF	0	•	11 101 101	ED	2	2	9	
										01 010 111	57				
LD A, R	A ← R	‡	‡	X	0	X	IFF	0	•	11 101 101	ED	2	2	9	
										01 011 111	5F				
LD I, A	I ← A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9	
										01 000 111	47				
LD R, A	R ← A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9	
										01 001 111	4F				

NOTE: IFF, the content of the interrupt enable flip-flop, (IFF<sub>2</sub>), is copied into the P/V flag.

## 16-BIT LOAD GROUP

Mnemonic	Symbolic Operation	S	Z	Flags			Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments	
				H	P/V	N	C	76	543	210					Hex
LD dd, nn	dd ← nn	•	•	X	•	X	•	•	•	00 dd0 001		3	3	10	dd Pair
										← n →					00 BC
										← n →					01 DE
LD IX, nn	IX ← nn	•	•	X	•	X	•	•	•	11 011 101	DD	4	4	14	10 HL
										00 100 001	21				11 SP
										← n →					
										← n →					
LD IY, nn	IY ← nn	•	•	X	•	X	•	•	•	11 111 101	FD	4	4	14	
										00 100 001	21				
										← n →					
										← n →					
LD HL, (nn)	H ← (nn+1) L ← (nn)	•	•	X	•	X	•	•	•	00 101 010	2A	3	5	16	
										← n →					
										← n →					
LD dd, (nn)	dd <sub>H</sub> ← (nn+1) dd <sub>L</sub> ← (nn)	•	•	X	•	X	•	•	•	11 101 101	ED	4	6	20	
										01 dd1 011					
										← n →					
										← n →					

NOTE: (PAIR)<sub>H</sub>, (PAIR)<sub>L</sub> refer to high order and low order eight bits of the register pair respectively, e.g., BC<sub>L</sub> = C, AF<sub>H</sub> = A.

## 16-BIT LOAD GROUP (Continued)

Mnemonic	Symbolic Operation	S	Z	Flags				Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments			
				H	P/V	N	C	76	543	210	Hex							
LD IX, (nn)	$IX_H \leftarrow (nn+1)$	•	•	X	•	X	•	•	•	11	011	101	DD	4	6	20		
	$IX_L \leftarrow (nn)$								00	101	010	2A						
										$\leftarrow n \rightarrow$								
										$\leftarrow n \rightarrow$								
LD IY, (nn)	$IY_H \leftarrow (nn+1)$	•	•	X	•	X	•	•	•	11	111	101	FD	4	6	20		
	$IY_L \leftarrow (nn)$								00	101	010	2A						
										$\leftarrow n \rightarrow$								
										$\leftarrow n \rightarrow$								
LD (nn), HL	$(nn+1) \leftarrow H$	•	•	X	•	X	•	•	•	00	100	010	22	3	5	16		
	$(nn) \leftarrow L$									$\leftarrow n \rightarrow$								
										$\leftarrow n \rightarrow$								
LD (nn), dd	$(nn+1) \leftarrow dd_H$	•	•	X	•	X	•	•	•	11	101	101	ED	4	6	20		
	$(nn) \leftarrow dd_L$								01	dd0	011							
										$\leftarrow n \rightarrow$								
										$\leftarrow n \rightarrow$								
LD (nn), IX	$(nn+1) \leftarrow IX_H$	•	•	X	•	X	•	•	•	11	011	101	DD	4	6	20		
	$(nn) \leftarrow IX_L$								00	100	010	22						
										$\leftarrow n \rightarrow$								
										$\leftarrow n \rightarrow$								
LD (nn), IY	$(nn+1) \leftarrow IY_H$	•	•	X	•	X	•	•	•	11	111	101	FD	4	6	20		
	$(nn) \leftarrow IY_L$								00	100	010	22						
										$\leftarrow n \rightarrow$								
										$\leftarrow n \rightarrow$								
LD SP, HL	$SP \leftarrow HL$	•	•	X	•	X	•	•	•	11	111	001	F9	1	1	6		
LD SP, IX	$SP \leftarrow IX$	•	•	X	•	X	•	•	•	11	011	101	DD	2	2	10		
										11	111	001	F9					
LD SP, IY	$SP \leftarrow IY$	•	•	X	•	X	•	•	•	11	111	101	FD	2	2	10		
										11	111	001	F9					
PUSH qq	$(SP-2) \leftarrow qq_L$	•	•	X	•	X	•	•	•	11	qq0	101		1	3	11	00	BC
	$(SP-1) \leftarrow qq_H$																01	DE
	$SP \rightarrow SP-2$																10	HL
PUSH IX	$(SP-2) \leftarrow IX_L$	•	•	X	•	X	•	•	•	11	011	101	DD	2	4	15	11	AF
	$(SP-1) \leftarrow IX_H$									11	100	101	E5					
	$SP \rightarrow SP-2$																	
PUSH IY	$(SP-2) \leftarrow IY_L$	•	•	X	•	X	•	•	•	11	111	101	FD	2	4	15		
	$(SP-1) \leftarrow IY_H$									11	100	101	E5					
	$SP \rightarrow SP-2$																	
POP qq	$qq_H \leftarrow (SP+1)$	•	•	X	•	X	•	•	•	11	qq0	001		1	3	10		
	$qq_L \leftarrow (SP)$																	
	$SP \rightarrow SP+2$																	
POP IX	$IX_H \leftarrow (SP+1)$	•	•	X	•	X	•	•	•	11	011	101	DD	2	4	14		
	$IX_L \leftarrow (SP)$									11	100	001	E1					
	$SP \rightarrow SP+2$																	
POP IY	$IY_H \leftarrow (SP+1)$	•	•	X	•	X	•	•	•	11	111	101	FD	2	4	14		
	$IY_L \leftarrow (SP)$									11	100	001	E1					
	$SP \rightarrow SP+2$																	

NOTE: (PAIR)<sub>H</sub>, (PAIR)<sub>L</sub> refer to high order and low order eight bits of the register pair respectively, e.g., BC<sub>L</sub> = C, AF<sub>H</sub> = A.

## EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS

Mnemonic	Symbolic Operation	Flags					Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments					
		S	Z	H	P/V	N	C	76	543	210					Hex				
EX DE, HL	DE ↔ HL	•	•	X	•	X	•	•	•	•	11	101	011	EB	1	1	4		
EX AF, AF'	AF ↔ AF'	•	•	X	•	X	•	•	•	•	00	001	000	08	1	1	4		
EXX	BC ↔ BC'	•	•	X	•	X	•	•	•	•	11	011	001	D9	1	1	4	Register bank and auxiliary register bank exchange	
	DE ↔ DE'																		
	HL ↔ HL'																		
EX (SP), HL	H ↔ (SP + 1) L ↔ (SP)	•	•	X	•	X	•	•	•	•	11	100	011	E3	1	5	19		
EX (SP), IX	IX <sub>H</sub> ↔ (SP + 1)	•	•	X	•	X	•	•	•	•	11	011	101	DD	2	6	23		
	IX <sub>L</sub> ↔ (SP)										11	100	011	E3					
EX (SP), IY	IY <sub>H</sub> ↔ (SP + 1)	•	•	X	•	X	•	•	•	•	11	111	101	FD	2	6	23		
	IY <sub>L</sub> ↔ (SP)										11	100	011	E3					
LDI	(DE) ← (HL)	•	•	X	0	X	↓	0	•	•	11	101	101	ED	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)	
	DE ← DE + 1										10	100	000	A0					
	HL ← HL + 1																		
	BC ← BC - 1																		
LDIR	(DE) ← (HL)	•	•	X	0	X	0	0	•	•	11	101	101	ED	2	5	21	If BC ≠ 0 If BC = 0	
	DE ← DE + 1										10	110	000	B0	2	4	16		
	HL ← HL + 1																		
	BC ← BC - 1																		
	Repeat until BC = 0																		
LDD	(DE) ← (HL)	•	•	X	0	X	↓	0	•	•	11	101	101	ED	2	4	16		
	DE ← DE - 1										10	101	000	A8					
	HL ← HL - 1																		
	BC ← BC - 1																		
LDDR	(DE) ← (HL)	•	•	X	0	X	0	0	•	•	11	101	101	ED	2	5	21	If BC ≠ 0 If BC = 0	
	DE ← DE - 1										10	111	000	B8	2	4	16		
	HL ← HL - 1																		
	BC ← BC - 1																		
	Repeat until BC = 0																		
CPI	A - (HL)	↓	↓	X	↓	X	↓	1	•	•	11	101	101	ED	2	4	16		
	HL ← HL + 1										10	100	001	A1					
	BC ← BC - 1																		

NOTE: ① P/V flag is 0 if the result of BC - 1 = 0, otherwise P/V = 1.

② P/V flag is 0 only at completion of instruction.

③ Z flag is 1 if A = HL, otherwise Z = 0.

## EXCHANGE, BLOCK TRANSFER, BLOCK SEARCH GROUPS (Continued)

Mnemonic	Symbolic Operation	S Z		Flags				Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments		
		S	Z	H	P/V	N	C	76	543	210	Hex						
CPIR	A ← (HL)	③				①											
		‡	‡	X	‡	X	‡	1	•	11	101	101	ED	2	5	21	If BC ≠ 0 and A ≠ (HL)
	HL ← HL + 1									10	110	001	B1	2	4	16	If BC = 0 or A = (HL)
	BC ← BC - 1																
	Repeat until A = (HL) or BC = 0																
CPD	A ← (HL)	③				①											
		‡	‡	X	‡	X	‡	1	•	11	101	101	ED	2	4	16	
	HL ← HL - 1									10	101	001	A9				
	BC ← BC - 1																
CPDR	A ← (HL)	③				①											
		‡	‡	X	‡	X	‡	1	•	11	101	101	ED	2	5	21	If BC ≠ 0 and A ≠ (HL)
	HL ← HL - 1									10	111	001	B9	2	4	16	If BC = 0 or A = (HL)
	BC ← BC - 1																
	Repeat until A = (HL) or BC = 0																

NOTE: ① P/V flag is 0 if the result of BC - 1 = 0, otherwise P/V = 1.

② P/V flag is 0 only at completion of instruction.

③ Z flag is 1 if A = (HL), otherwise Z = 0.

## 8-BIT ARITHMETIC AND LOGICAL GROUP

Mnemonic	Symbolic Operation	S Z		Flags				Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments											
		S	Z	H	P/V	N	C	76	543	210	Hex															
ADD A, r	A ← A + r	‡	‡	X	‡	X	V	0	‡	10	000	r		1	1	4	r Reg.									
ADD A, n	A ← A + n	‡	‡	X	‡	X	V	0	‡	11	000	110		2	2	7	000 B									
																								001 C		
																										010 D
ADD A, (HL)	A ← A + (HL)	‡	‡	X	‡	X	V	0	‡	10	000	110		1	2	7	100 H									
ADD A, (IX + d)	A ← A + (IX + d)	‡	‡	X	‡	X	V	0	‡	11	011	101	DD	3	5	19	101 L									
																									110	
ADD A, (IY + d)	A ← A + (IY + d)	‡	‡	X	‡	X	V	0	‡	11	111	101	FD	3	5	19										
ADC A, s	A ← A + s + CY	‡	‡	X	‡	X	V	0	‡		001						s is any of r, n, (HL), (IX + d), (IY + d) as shown for ADD instruction. The indicated bits replace the 000 in the ADD set above.									
SUB s	A ← A - s	‡	‡	X	‡	X	V	1	‡		010															
SBC A, s	A ← A - s - CY	‡	‡	X	‡	X	V	1	‡		011															
AND s	A ← A > s	‡	‡	X	1	X	P	0	0		100															
OR s	A ← A > s	‡	‡	X	0	X	P	0	0		110															
XOR s	A ← A ⊕ s	‡	‡	X	0	X	P	0	0		101															
CP s	A - s	‡	‡	X	‡	X	V	1	‡		111															



## 8-BIT ARITHMETIC AND LOGICAL GROUP (Continued)

Mnemonic	Symbolic Operation	Flags						Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments		
		S	Z	H	P/V	N	C	76	543	210	Hex						
INC r	$r \leftarrow r+1$	†	†	X	†	X	V	0	•	00	r	<span style="border: 1px solid black;">100</span>		1	1	4	
INC (HL)	(HL) ← (HL)+1	†	†	X	†	X	V	0	•	00	110	<span style="border: 1px solid black;">100</span>		1	3	11	
INC (IX+d)	(IX+d) ← (IX+d)+1	†	†	X	†	X	V	0	•	11	011	101	DD	3	6	23	
										00	110	<span style="border: 1px solid black;">100</span>					
INC (IY+d)	(IY+d) ← (IY+d)+1	†	†	X	†	X	V	0	•	11	111	101	FD	3	6	23	
										00	110	<span style="border: 1px solid black;">100</span>					
DEC m	$m \leftarrow m-1$	†	†	X	†	X	V	1	•			<span style="border: 1px solid black;">101</span>					

NOTE: m is any of r, (HL), (IX+d), (IY+d) as shown for INC. DEC same format and states as INC. Replace 100 with 101 in opcode.

## GENERAL-PURPOSE ARITHMETIC AND CPU CONTROL GROUPS

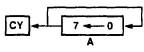
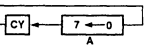
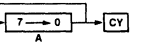
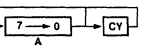
Mnemonic	Symbolic Operation	Flags						Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments		
		S	Z	H	P/V	N	C	76	543	210	Hex						
DAA	@	†	†	X	†	X	P	•	†	00	100	111	27	1	1	4	Decimal adjust accumulator.
CPL	$A \leftarrow A$	•	•	X	1	X	•	1	•	00	101	111	2F	1	1	4	Complement accumulator (one's complement).
NEG	$A \leftarrow 0 - A$	†	†	X	†	X	V	1	†	11	101	101	ED	2	2	8	Negate acc. (two's complement).
										01	000	100	44				
CCF	$CY \leftarrow CY$	•	•	X	X	X	•	0	†	00	111	111	3F	1	1	4	Complement carry flag.
SCF	$CY \leftarrow 1$	•	•	X	0	X	•	0	1	00	110	111	37	1	1	4	Set carry flag.
NOP	No operation	•	•	X	•	X	•	•	•	00	000	000	00	1	1	4	
HALT	CPU halted	•	•	X	•	X	•	•	•	01	110	110	76	1	1	4	
DI ★	$IFF \leftarrow 0$	•	•	X	•	X	•	•	•	11	110	011	F3	1	1	4	
EI ★	$IFF \leftarrow 1$	•	•	X	•	X	•	•	•	11	111	011	FB	1	1	4	
IM 0	Set interrupt mode 0	•	•	X	•	X	•	•	•	11	101	101	ED	2	2	8	
										01	000	110	46				
IM 1	Set interrupt mode 1	•	•	X	•	X	•	•	•	11	101	101	ED	2	2	8	
										01	010	110	56				
IM 2	Set interrupt mode 2	•	•	X	•	X	•	•	•	11	101	101	ED	2	2	8	
										01	011	110	5E				

NOTES: @ converts accumulator content into packed BCD following add or subtract with packed BCD operands.  
 IFF indicates the interrupt enable flip-flop.  
 CY indicates the carry flip-flop.  
 ★ indicates interrupts are not sampled at the end of EI or DI.

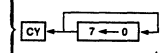
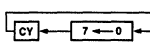
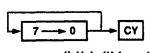
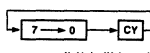
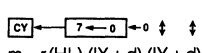
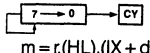
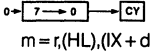
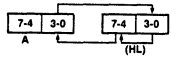
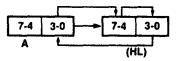
## 16-BIT ARITHMETIC GROUP

Mnemonic	Symbolic Operation	S	Z	Flags			Opcode			Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments			
				H	P/V	N	C	76	543							210	
ADD HL, ss	HL ← HL + ss	•	•	X	X	X	•	0	‡	00	ssl	001	1	3	11	ss Reg. 00 BC	
ADC HL, ss	HL ← HL + ss + CY	‡	‡	X	X	X	V	0	‡	11	101	101	ED	2	4	15	01 DE 10 HL 11 SP
SBC HL, ss	HL ← HL - ss - CY	‡	‡	X	X	X	V	1	‡	11	101	101	ED	2	4	15	01 ss0 010
ADD IX, pp	IX ← IX + pp	•	•	X	X	X	•	0	‡	11	011	101	DD	2	4	15	pp Reg. 00 BC 01 DE 10 IX 11 SP
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	‡	11	111	101	FD	2	4	15	rr Reg. 00 BC
INC ss	ss ← ss + 1	•	•	X	•	X	•	•	•	00	ss0	011		1	1	6	01 DE
INC IX	IX ← IX + 1	•	•	X	•	X	•	•	•	11	011	101	DD	2	2	10	10 IY 11 SP
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11	111	101	FD	2	2	10	00 100 011 23
DEC ss	ss ← ss - 1	•	•	X	•	X	•	•	•	00	ss1	011		1	1	6	
DEC IX	IX ← IX - 1	•	•	X	•	X	•	•	•	11	011	101	DD	2	2	10	00 101 011 2B
DEC IY	IY ← IY - 1	•	•	X	•	X	•	•	•	11	111	101	FD	2	2	10	00 101 011 2B

## ROTATE AND SHIFT GROUP

Mnemonic	Symbolic Operation	S	Z	Flags			Opcode			Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments			
				H	P/V	N	C	76	543							210	
RLCA		•	•	X	0	X	•	0	‡	00	000	111	07	1	1	4	Rotate left circular accumulator.
RLA		•	•	X	0	X	•	0	‡	00	010	111	17	1	1	4	Rotate left accumulator.
RRCA		•	•	X	0	X	•	0	‡	00	001	111	0F	1	1	4	Rotate right circular accumulator.
RRA		•	•	X	0	X	•	0	‡	00	011	111	1F	1	1	4	Rotate right accumulator.

# ROTATE AND SHIFT GROUP (Continued)

Mnemonic	Symbolic Operation	S	Z	Flags				Opcode			Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments		
				H	P/V	N	C	76	543	210							
RLC r		†	†	X	0	X	P	0	•	†	11 001 011	CB	2	2	8	Rotate left circular register r.	
											00 <span style="border: 1px solid black; padding: 2px;">000</span> r						
RLC (HL)		†	†	X	0	X	P	0	†		11 001 011	CB	2	4	15	r	Reg.
											00 000 110					000	B
RLC (IX+d)		†	†	X	0	X	P	0	†		11 011 101	DD	4	6	23	010	D
	r,(HL),(IX+d),(IY+d)										11 001 011	CB				011	E
											← d →					001	H
											00 <span style="border: 1px solid black; padding: 2px;">000</span> 110					101	L
																111	A
RLC (IY+d)		†	†	X	0	X	P	0	†		11 111 101	FD	4	6	23		
											11 001 011	CB					
											← d →						
RL m	 m = r,(HL),(IX+d),(IY+d)	†	†	X	0	X	P	0	†		00 <span style="border: 1px solid black; padding: 2px;">000</span> 110						
											<span style="border: 1px solid black; padding: 2px;">010</span>						
RRC m	 m = r,(HL),(IX+d),(IY+d)	†	†	X	0	X	P	0	†		<span style="border: 1px solid black; padding: 2px;">001</span>						
RR m	 m = r,(HL),(IX+d),(IY+d)	†	†	X	0	X	P	0	†		<span style="border: 1px solid black; padding: 2px;">011</span>						
SLA m	 m = r,(HL),(IX+d),(IY+d)	†	†	X	0	X	P	0	†		<span style="border: 1px solid black; padding: 2px;">100</span>						
SRA m	 m = r,(HL),(IX+d),(IY+d)	†	†	X	0	X	P	0	†		<span style="border: 1px solid black; padding: 2px;">101</span>						
SRL m	 m = r,(HL),(IX+d),(IY+d)	†	†	X	0	X	P	0	†		<span style="border: 1px solid black; padding: 2px;">111</span>						
RLD		†	†	X	0	X	P	0	•	11 101 101	ED	2	5	18		Rotate digit left and right between the accumulator and location (HL).	
										01 101 111	6F						
RRD		†	†	X	0	X	P	0	•	11 101 101	ED	2	5	18		The content of the upper half of the accumulator is unaffected.	
										01 100 111	67						

Instruction format and states are as shown for RLCs. To form new opcode replace 000, or RLCs with shown code.

## BIT SET, RESET AND TEST GROUP

Mnemonic	Symbolic Operation	Flags						Opcode			No. of Bytes	No. of M Cycles	No. of T States	Comments	
		S	Z	H	P/V	N	C	76	543	210					Hex
BIT b, r	$Z \leftarrow r_b$	X	†	X	1	X	X	0	•	11 001 011	CB	2	2	8	r Reg. 000 B
BIT b, (HL)	$Z \leftarrow (HL)_b$	X	†	X	1	X	X	0	•	11 001 011	CB	2	3	12	001 C 010 D
BIT b, (IX+d) <sub>b</sub>	$Z \leftarrow (IX+d)_b$	X	†	X	1	X	X	0	•	11 011 101	DD	4	5	20	011 E 100 H 101 L 111 A
										$\leftarrow d \rightarrow$					
										01 b 110					b Bit Tested
BIT b, (IY+d) <sub>b</sub>	$Z \leftarrow (IY+d)_b$	X	†	X	1	X	X	0	•	11 111 101	FD	4	5	20	000 0 001 1 010 2 011 3
										$\leftarrow d \rightarrow$					
										01 b 110					
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011	CB	2	2	8	100 4 101 5
										<span style="border: 1px solid black; padding: 0 2px;">11</span> b r					
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011	CB	2	4	15	110 6 111 7
										<span style="border: 1px solid black; padding: 0 2px;">11</span> b 110					
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 011 101	DD	4	6	23	
										11 001 011	CB				
										$\leftarrow d \rightarrow$					
										<span style="border: 1px solid black; padding: 0 2px;">11</span> b 110					
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 101	FD	4	6	23	
										11 001 011	CB				
										$\leftarrow d \rightarrow$					
										<span style="border: 1px solid black; padding: 0 2px;">11</span> b 110					
RES b, m	$m_b \leftarrow 0$ $m \equiv r$ (HL), (IX+d), (IY+d)	•	•	X	•	X	•	•	•	<span style="border: 1px solid black; padding: 0 2px;">11</span> <span style="border: 1px solid black; padding: 0 2px;">10</span>	b 110				

To form new opcode replace 11 of SET b, s with 10. Flags and time states for SET instruction.

NOTE: The notation  $m_b$  indicates location m, bit b (0 to 7).

## JUMP GROUP

Mnemonic	Symbolic Operation	Flags					Opcode				No. of Bytes	No. of M Cycles	No. of T States	Comments				
		S	Z	H	P/VN	C	76	543	210	Hex								
JP nn	PC ← nn	•	•	X	•	X	•	•	•	•	11	000	011	C3	3	3	10	cc Condition 000 NZ (non-zero) 001 Z (zero)
												← n →						
												← n →						
JP cc, nn	If condition cc is true PC ← nn, otherwise continue	•	•	X	•	X	•	•	•	•	11	cc	010		3	3	10	010 NC (non-carry) 011 C (carry) 100 PO (parity odd) 101 PE (parity even)
												← n →						
												← n →						
JR e	PC ← PC + e	•	•	X	•	X	•	•	•	•	00	011	000	18	2	3	12	110 P (sign positive) 111 M (sign negative)
												← e - 2 →						
JR C, e	If C = 0, continue If C = 1, PC ← PC + e	•	•	X	•	X	•	•	•	•	00	111	000	38	2	2	7	If condition not met.
												← e - 2 →						
														2	3	12	If condition is met.	
JR NC, e	If C = 1, continue If C = 0, PC ← PC + e	•	•	X	•	X	•	•	•	•	00	110	000	30	2	2	7	If condition not met.
												← e - 2 →						
														2	3	12	If condition is met.	
JP Z, e	If Z = 0, continue If Z = 1, PC ← PC + e	•	•	X	•	X	•	•	•	•	00	101	000	28	2	2	7	If condition not met.
												← e - 2 →						
														2	3	12	If condition is met.	
JR NZ, e	If Z = 1, continue If Z = 0, PC ← PC + e	•	•	X	•	X	•	•	•	•	00	100	000	20	2	2	7	If condition not met.
												← e - 2 →						
														2	3	12	If condition is met.	
JP (HL)	PC ← HL	•	•	X	•	X	•	•	•	•	11	101	001	E9	1	1	4	
JP (IX)	PC ← IX	•	•	X	•	X	•	•	•	•	11	011	101	DD	2	2	8	
												11	101	001	E9			
JP (IY)	PC ← IY	•	•	X	•	X	•	•	•	•	11	111	101	FD	2	2	8	
												11	101	001	E9			
DJNZ, e	B ← B - 1 If B = 0, continue If B ≠ 0, PC ← PC + e	•	•	X	•	X	•	•	•	•	00	010	000	10	2	2	8	If B = 0
												← e - 2 →						
														2	3	13	If B ≠ 0.	

NOTES: e represents the extension in the relative addressing mode.

e is a signal two's complement number in the range < -126, 129 >.

e - 2 in the opcode provides an effective address of pc + e as PC is incremented by 2 prior to the addition of e.

## CALL AND RETURN GROUP

Mnemonic	Symbolic Operation	Flags					Opcode			Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments		
		S	Z	H	P/VN	C	76	543	210							
CALL nn	(SP-1)←PC <sub>H</sub> (SP-2)←PC <sub>L</sub> PC←nn,	•	•	X	•	X	•	•	•	•	11 001 101	CD	3	5	17	
																←n→
																←n→
CALL cc,nn	If condition cc is false continue, otherwise same as CALL nn	•	•	X	•	X	•	•	•	•	11 cc 100		3	3	10	If cc is false.
																←n→
													3	5	17	If cc is true.
RET	PC <sub>L</sub> ←(SP) PC <sub>H</sub> ←(SP+1)	•	•	X	•	X	•	•	•	•	11 001 001	C9	1	3	10	
RET cc	If condition cc is false continue, otherwise same as RET	•	•	X	•	X	•	•	•	•	11 cc 000		1	1	5	If cc is false.
													1	3	11	If cc is true.
																cc Condition
																000 NZ (non-zero)
																001 Z (zero)
																010 NC (non-carry)
																011 C (carry)
																100 PO (parity odd)
																101 PE (parity even)
																110 P (sign positive)
																111 M (sign negative)
RET1	Return from interrupt	•	•	X	•	X	•	•	•	•	11 101 101	ED	2	4	14	
											01 001 101	4D				
RETN <sup>1</sup>	Return from non-maskable interrupt	•	•	X	•	X	•	•	•	•	11 101 101	ED	2	4	14	
											01 000 101	45				
RST p	(SP-1)←PC <sub>H</sub> (SP-2)←PC <sub>L</sub> PC <sub>H</sub> ←0 PC <sub>L</sub> ←p	•	•	X	•	X	•	•	•	•	11 t 111		1	3	11	
																t p
																000 00H
																001 08H
																010 10H
																011 18H
																100 20H
																101 28H
																110 30H
																111 38H

NOTE: <sup>1</sup>RETN loads IFF<sub>2</sub> → IFF<sub>1</sub>

## INPUT AND OUTPUT GROUP

Mnemonic	Symbolic Operation	Flags					Opcode			No. of Bytes	No. of M Cycles	No. of T States	Comments					
		S	Z	H	P/VN	C	76	543	210					Hex				
INA, (n)	A ← (n)	•	•	X	•	X	•	•	•	•	11 011 01	DB	2	3	11	n to A <sub>0</sub> ~ A <sub>7</sub> Acc. to A <sub>8</sub> ~ A <sub>15</sub>		
INr, (C)	r ← (C) if r = 110 only the flags will be affected	‡	‡	X	‡	X	P	0	•	11 101 101	ED	2	3	12	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										01 r 000								
INI	(HL) ← (C) B ← B - 1 HL ← HL + 1	X	‡	X	X	X	X	1	X	11 101 101	ED	2	4	16	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 100 010	A2							
INIR	(HL) ← (C) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101	ED	2	5	21	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 110 010	B2		(If B ≠ 0)					
												2	4	16	(If B = 0)			
IND	(HL) ← (C) B ← B - 1 HL ← HL - 1	X	‡	X	X	X	X	1	X	11 101 101	ED	2	4	16	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 101 010	AA							
INDR	(HL) ← (C) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101	ED	2	5	21	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 111 010	BA		(If B ≠ 0)					
												2	4	16	(If B = 0)			
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	•	•	•	11 010 011	D3	2	3	11	n to A <sub>0</sub> ~ A <sub>7</sub> Acc. to A <sub>8</sub> ~ A <sub>15</sub>
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11 101 101	ED	2	3	12	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										01 r 001								
OUTI	(C) ← (HL) B ← B - 1 HL ← HL + 1	X	‡	X	X	X	X	1	X	11 101 101	ED	2	4	16	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 100 011	A3							
OTIR	(C) ← (HL) B ← B - 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101	ED	2	5	21	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 110 011	B3		(If B ≠ 0)					
												2	4	16	(If B = 0)			
OUTD	(C) ← (HL) B ← B - 1 HL ← HL - 1	X	‡	X	X	X	X	1	X	11 101 101	ED	2	4	16	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 101 011	AB							
OTDR	(C) ← (HL) B ← B - 1 HL ← HL - 1 Repeat until B = 0	X	1	X	X	X	X	1	X	11 101 101	ED	2	5	21	C to A <sub>0</sub> ~ A <sub>7</sub> B to A <sub>8</sub> ~ A <sub>15</sub>			
										10 111 011			(If B ≠ 0)					
												2	4	16	(If B = 0)			

NOTES: ① If the result of B - 1 is zero, the Z flag is set; otherwise it is reset.

② Z flag is set upon instruction completion only.

## SUMMARY OF FLAG OPERATION

Instructions	D <sub>7</sub> S	Z	H	P/V	N	D <sub>0</sub> C	Comments
ADD A, s; ADC A, s	‡	‡	X	‡	X	V 0 ‡	8-bit add or add with carry.
SUB s; SBC A, s; CP s; NEG	‡	‡	X	‡	X	V 1 ‡	8-bit subtract, subtract with carry, compare and negate accumulator.
AND s	‡	‡	X	1	X	P 0 0	Logical operation.
OR s, XOR s	‡	‡	X	0	X	P 0 0	Logical operation.
INC s	‡	‡	X	‡	X	V 0 •	8-bit increment.
DEC s	‡	‡	X	‡	X	V 1 •	8-bit decrement.
ADD DD, ss	•	•	X	X	X	• 0 ‡	16-bit add.
ADC HL, ss	‡	‡	X	X	X	V 0 ‡	16-bit add with carry.
SBC HL, ss	‡	‡	X	X	X	V 1 ‡	16-bit subtract with carry.
RLA; RLCA; RRA; RRCA	•	•	X	0	X	• 0 ‡	Rotate accumulator.
RL m; RLC m; RR m; RRC m; SLA m; SRA m; SRL m	‡	‡	X	0	X	P 0 ‡	Rotate and shift locations.
RLD; RRD	‡	‡	X	0	X	P 0 •	Rotate digit left and right.
DAA	‡	‡	X	‡	X	P • ‡	Decimal adjust accumulator.
CPL	•	•	X	1	X	• 1 •	Complement accumulator.
SCF	•	•	X	0	X	• 0 1	Set carry.
CCF	•	•	X	X	X	• 0 ‡	Complement carry.
IN r (C)	‡	‡	X	0	X	P 0 •	Input register indirect.
INI; IND; OUTI; OUTD	X	‡	X	X	X	X 1 •	Block input and output. Z = 1 if B ≠ 0, otherwise Z = 0.
INIR; INDR; OTIR; OTDR	X	1	X	X	X	X 1 •	Block input and output. Z = 1 if B ≠ 0, otherwise Z = 0.
LDI; LDD	X	X	X	0	X	‡ 0 •	Block transfer instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LDIR; LDDR	X	X	X	0	X	0 0 •	Block transfer instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
CPI; CPIR; CPD; CPDR	X	‡	X	X	X	‡ 1 •	Block search instructions. Z = 1 if A = (HL), otherwise Z = 0. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LD A; I, LD A, R	‡	‡	X	0	X	IFF 0 •	IFF, the content of the interrupt enable flip-flop, (IFF <sub>2</sub> ), is copied into the P/V flag.
BIT b, s	X	‡	X	1	X	X 0 •	The state of bit b of location s is copied into the Z flag.

## SYMBOLIC NOTATION

### Symbol Operation

S	Sign flag. S = 1 if the MSB of the result is 1.
Z	Zero flag. Z = 1 if the result of the operation is 0.
P/V	Parity or overflow flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity: P/V = 1 if the result of the operation is even; P/V = 0 if result is odd. If P/V holds overflow, P/V = 1 if the result of the operation produced an overflow. If P/V does not hold overflow, P/V = 0.
H*	Half-carry flag. H = 1 if the add or subtract operation produced a carry into, or borrow from, bit 4 of the accumulator.
N*	Add/Subtract flag. N = 1 if the previous operation was a subtract.
C	Carry/Link flag. C = 1 if the operation produced a carry from the MSB of the operand or result.

### Symbol Operation

‡	The flag is affected according to the result of the operation.
•	The flag is unchanged by the operation.
0	The flag is reset by the operation.
1	The flag is set by the operation.
X	The flag is indeterminate.
V	P/V flag affected according to the overflow result of the operation.
P	P/V flag affected according to the parity result of the operation.
r	Any one of the CPU registers A, B, C, D, E, H, L.
s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
ss	Any 16-bit location for all the addressing modes allowed for that instruction.
ii	Any one of the two index registers IX or IY.
R	Refresh counter.
n	8-bit value in range < 0, 255 >.
nn	16-bit value in range < 0, 65535 >.

\* H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.



## CPU REGISTERS

Figure 4 shows three groups of registers within the CPU. The first group consists of duplicate sets of 8-bit registers: a principal set and an alternate set [designated by ' (prime), e.g., A']. Both sets consist of the Accumulator register, the Flag register, and six general-purpose registers. Transfer of data between these duplicate sets of registers is accomplished by use of "Exchange" instructions. The result is faster response to interrupts and easy, efficient implementation of such versatile programming techniques

as background-foreground data processing. The second set of registers consists of six registers with assigned functions. These are the I (Interrupt register), the R (Refresh register), the IX and IY (Index registers), the SP (Stack Pointer), and the PC (Program Counter). The third group consists of two interrupt status flip-flops, plus an additional pair of flip-flops which assists in identifying the interrupt mode at any particular time. Table 1 provides further information on these registers.

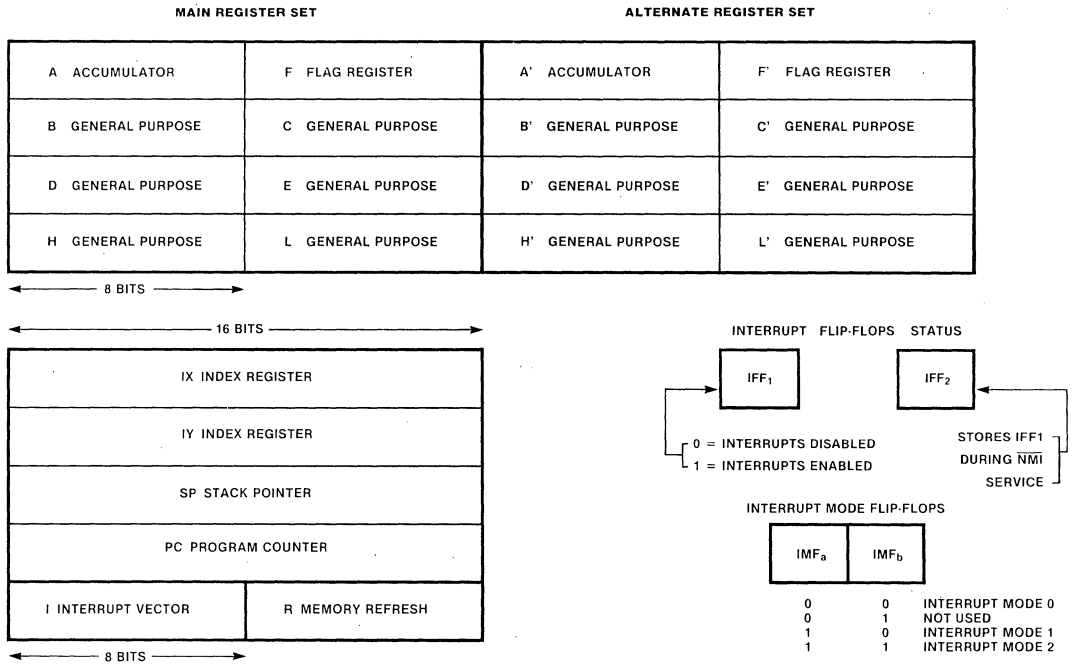


Figure 4. CPU Registers

## INTERRUPTS: GENERAL OPERATION

The CPU accepts two interrupt input signals:  $\overline{\text{NMI}}$  and  $\overline{\text{INT}}$ . The  $\overline{\text{NMI}}$  is a non-maskable interrupt and has the highest priority.  $\overline{\text{INT}}$  is a lower priority interrupt and it requires that interrupts be enabled in software in order to operate.  $\overline{\text{INT}}$  can be connected to multiple peripheral devices in a wired-OR configuration.

The Z80 has a single response mode for interrupt service on the non-maskable interrupt. The maskable interrupt,  $\overline{\text{INT}}$ , has three programmable response modes available. These are:

- Mode 0 — similar to the 8080 microprocessor.
- Mode 1 — Peripheral Interrupt service, for use with non-8080/Z80 systems.

- Mode 2 - a vectored interrupt scheme, usually daisy-chained, for use with the Z80 Family and compatible peripheral devices.

The CPU services interrupts by sampling the  $\overline{\text{NMI}}$  and  $\overline{\text{INT}}$  signals at the rising edge of the last clock of an instruction. Further interrupt service processing depends upon the type of interrupt that was detected. Details on interrupt responses are shown in the CPU Timing Section.

**Non-Maskable Interrupt ( $\overline{\text{NMI}}$ ).** The nonmaskable interrupt cannot be disabled by program control and therefore will be accepted at all times by the CPU.  $\overline{\text{NMI}}$  is usually reserved for servicing only the highest priority type interrupts, such as that for orderly shutdown after power

---

## PIN DESCRIPTIONS

**A<sub>0</sub>-A<sub>15</sub>.** *Address Bus* (output, active High, 3-state). A<sub>0</sub>-A<sub>15</sub> form a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64K bytes) and for I/O device exchanges.

**BUSACK.** *Bus Acknowledge* (output, active Low). Bus Acknowledge indicates to the requesting device that the CPU address bus, data bus, and control signals  $\overline{\text{MREQ}}$ ,  $\overline{\text{IORQ}}$ , RD, and WR have entered their high-impedance states. The external circuitry can now control these lines.

**BUSREQ.** *Bus Request* (input, active Low). Bus Request has a higher priority than NMI and is always recognized at the end of the current machine cycle.  $\overline{\text{BUSREQ}}$  forces the CPU address bus, data bus, and control signals  $\overline{\text{MREQ}}$ ,  $\overline{\text{IORQ}}$ , RD, and WR to go to a high-impedance state so that other devices can control these lines.  $\overline{\text{BUSREQ}}$  is normally wired-OR and requires an external pullup for these applications. Extended  $\overline{\text{BUSREQ}}$  periods due to extensive DMA operations can prevent the CPU from properly refreshing dynamic RAMs.

**D<sub>0</sub>-D<sub>7</sub>.** *Data Bus* (input/output, active High, 3-state). D<sub>0</sub>-D<sub>7</sub> constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.

**HALT.** *Halt State* (output, active Low).  $\overline{\text{HALT}}$  indicates that the CPU has executed a Halt instruction and is awaiting either a nonmaskable or a maskable interrupt (with the mask enabled) before operation can resume. While halted, the CPU executes NOPs to maintain memory refresh.

**INT.** *Interrupt Request* (input, active Low). Interrupt Request is generated by I/O devices. The CPU honors a request at the end of the current instruction if the internal software-controlled interrupt enable flip-flop (IFF) is enabled.  $\overline{\text{INT}}$  is normally wired-OR and requires an external pullup for these applications.

**IORQ.** *Input/Output Request* (output, active Low, 3-state).  $\overline{\text{IORQ}}$  indicates that the lower half of the address bus holds a valid I/O address for an I/O read or write operation.  $\overline{\text{IORQ}}$  is also generated concurrently with  $\overline{\text{M1}}$  during an interrupt acknowledge cycle to indicate that an interrupt response vector can be placed on the data bus.

**M1.** *Machine Cycle One* (output, active Low).  $\overline{\text{M1}}$ , together with  $\overline{\text{MREQ}}$ , indicates that the current machine cycle is the opcode fetch cycle of an instruction execution.  $\overline{\text{M1}}$ , together with  $\overline{\text{IORQ}}$ , indicates an interrupt acknowledge cycle.

**MREQ.** *Memory Request* (output, active Low, 3-state).  $\overline{\text{MREQ}}$  indicates that the address bus holds a valid address for a memory read or memory write operation.

**NMI.** *Non-Maskable Interrupt* (input, negative edge-triggered). NMI has a higher priority than INT. NMI is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop, and automatically forces the CPU to restart at location 0066H.

**RD.** *Read* (output, active Low, 3-state).  $\overline{\text{RD}}$  indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

**RESET.** *Reset* (input, active Low).  $\overline{\text{RESET}}$  initializes the CPU as follows: it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state. Note that  $\overline{\text{RESET}}$  must be active for a minimum of three full clock cycles before the reset operation is complete.

**RFSH.** *Refresh* (output, active Low).  $\overline{\text{RFSH}}$ , together with  $\overline{\text{MREQ}}$ , indicates that the lower seven bits of the system's address bus can be used as a refresh address to the system's dynamic memories.

**WAIT.** *Wait* (input, active Low).  $\overline{\text{WAIT}}$  indicates to the CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter a Wait state as long as this signal is active. Extended  $\overline{\text{WAIT}}$  periods can prevent the CPU from properly refreshing dynamic memory.

**WR.** *Write* (output, active Low, 3-state).  $\overline{\text{WR}}$  indicates that the CPU data bus holds valid data to be stored at the addressed memory or I/O location.

## CPU TIMING

The Z80 CPU executes instructions by proceeding through a specific sequence of operations:

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

The basic clock period is referred to as a T time or cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

**Instruction Opcode Fetch.** The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (Figure 5). Approximately one-half clock cycle later,  $\overline{MREQ}$  goes active. When active,  $\overline{RD}$  indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the  $\overline{WAIT}$  input with the falling edge of clock state  $T_2$ . During clock states  $T_3$  and  $T_4$  of an  $\overline{M1}$  cycle, dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction. When the Refresh Control signal becomes active, refreshing of dynamic memory can take place.

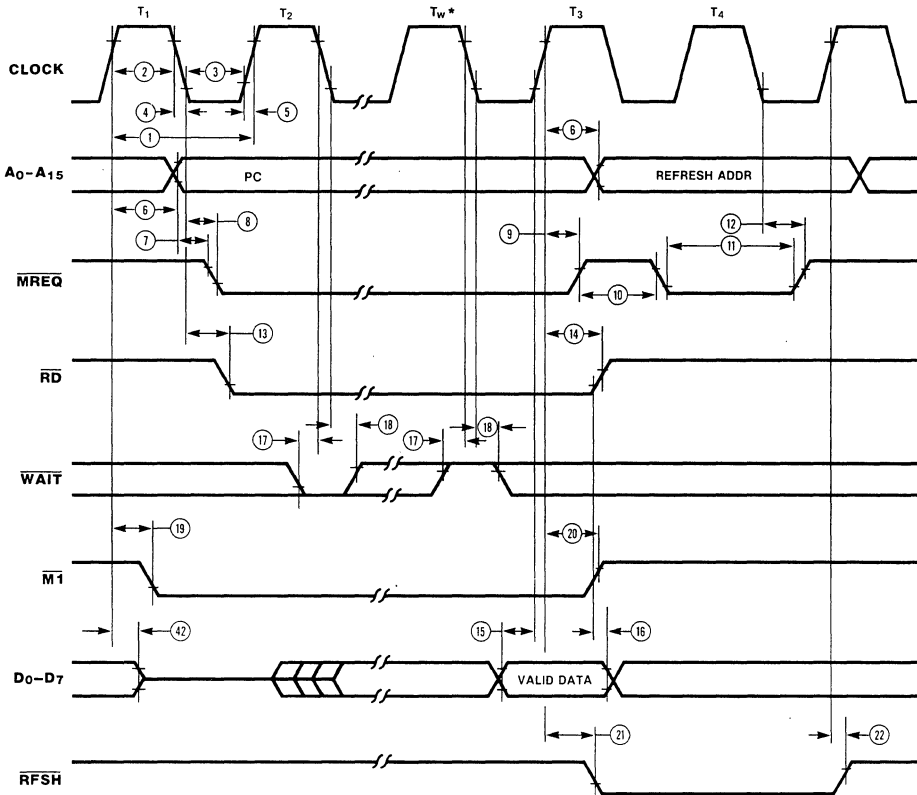


Figure 5. Instruction Opcode Fetch

**Memory Read or Write Cycles.** Figure 6 shows the timing of memory read or write cycles other than an opcode fetch ( $\overline{M1}$ ) cycle. The  $\overline{MREQ}$  and  $\overline{RD}$  signals function exactly as in the fetch cycle. In a memory write cycle,  $\overline{MREQ}$  also

becomes active when the address bus is stable. The  $\overline{WR}$  line is active when the data bus is stable, so that it can be used directly as an  $R/\overline{W}$  pulse to most semiconductor memories.

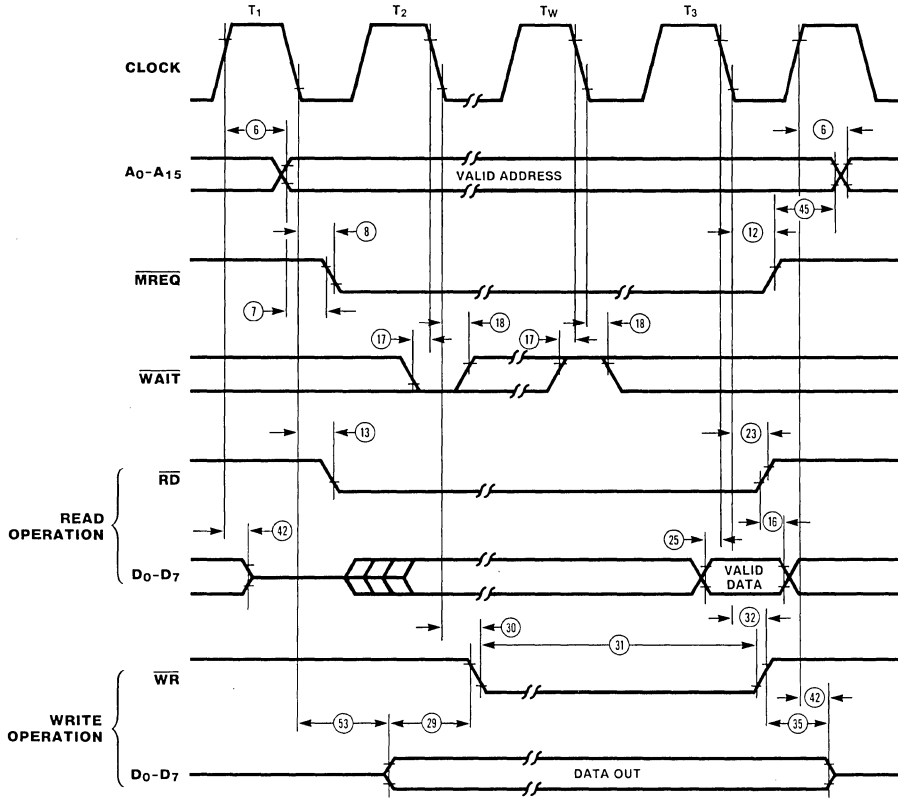
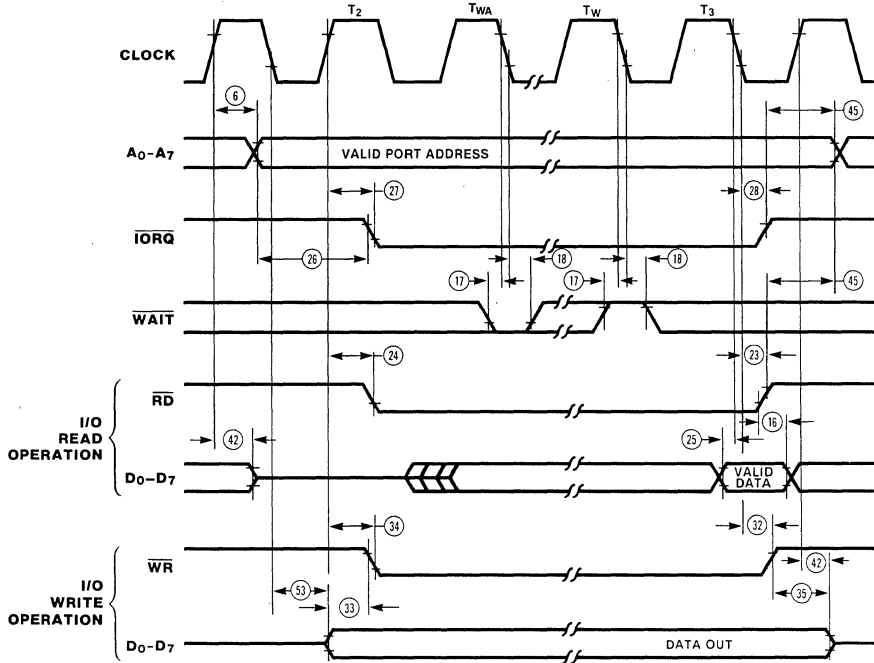


Figure 6. Memory Read or Write Cycles

**Input or Output Cycles.** Figure 7 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single Wait state ( $T_{WA}$ ). This

extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.

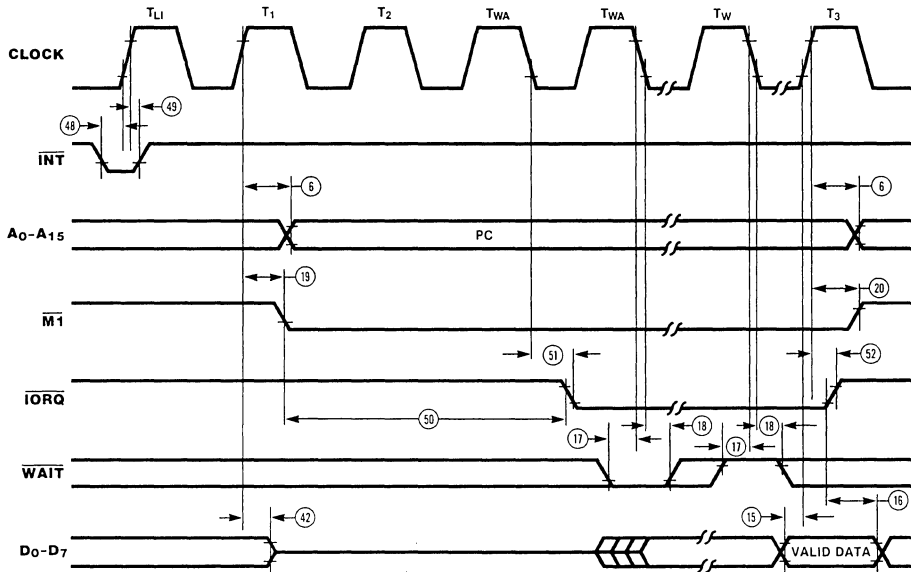


$T_{WA}$  = One wait cycle automatically inserted by CPU.

**Figure 7. Input or Output Cycles**

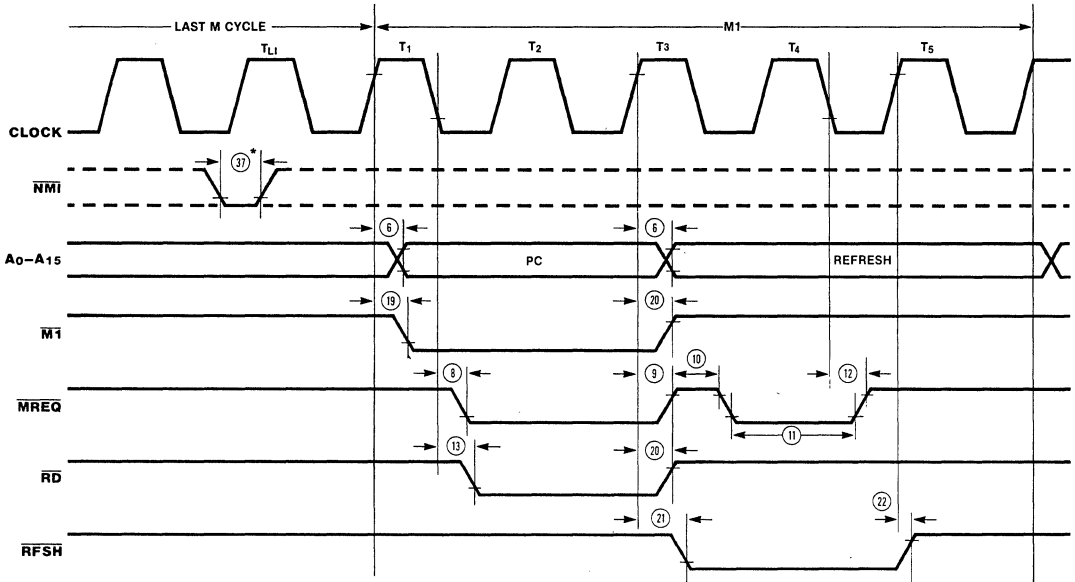
**Interrupt Request/Acknowledge Cycle.** The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 8). When an interrupt is accepted, a special  $\overline{M1}$  cycle is generated.

During this  $\overline{M1}$  cycle,  $\overline{IORQ}$  becomes active (instead of  $\overline{MREQ}$ ) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.



**Non-Maskable Interrupt Request Cycle.**  $\overline{\text{NMI}}$  is sampled at the same time as the maskable interrupt input  $\overline{\text{INT}}$  but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal

memory read operation except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the  $\overline{\text{NMI}}$  service routine located at address 0066H (Figure 9).

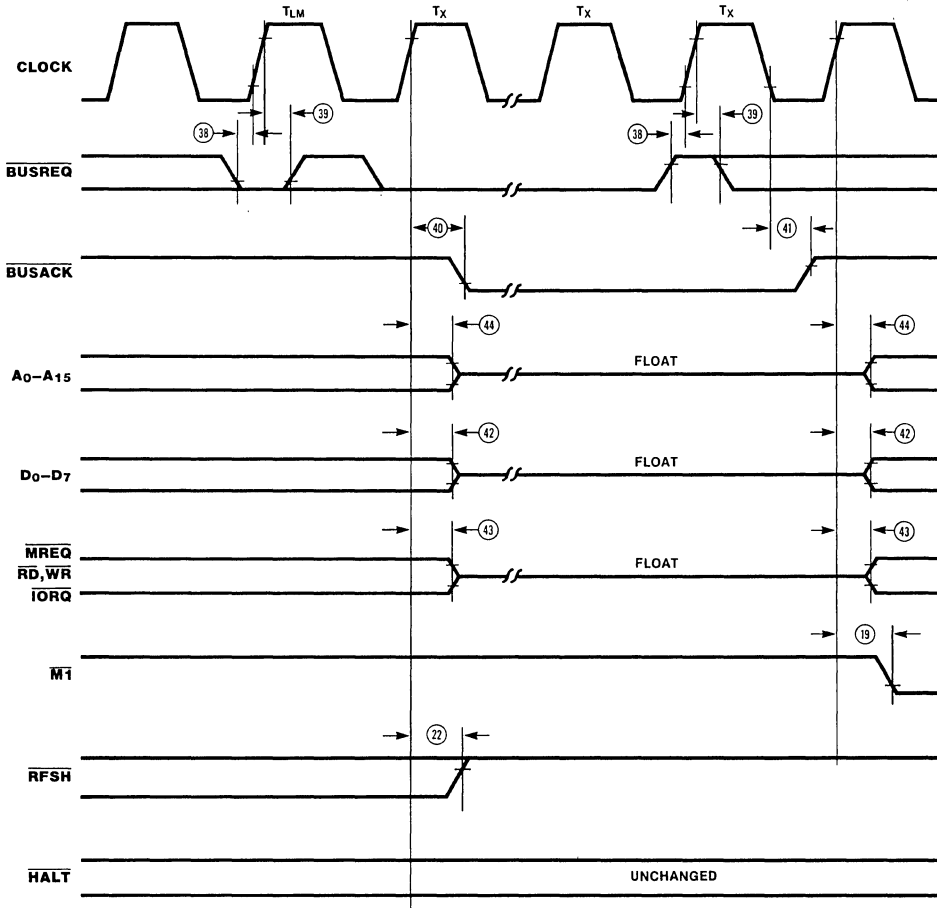


\*Although  $\overline{\text{NMI}}$  is an asynchronous input, to guarantee its being recognized on the following machine cycle,  $\overline{\text{NMI}}$ 's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle ( $T_{L1}$ ).

**Figure 9. Non-Maskable Interrupt Request Operation**

**Bus Request/Acknowledge Cycle.** The CPU samples  $\overline{\text{BUSREQ}}$  with the rising edge of the last clock period of any machine cycle (Figure 10). If  $\overline{\text{BUSREQ}}$  is active, the CPU sets its address, data, and  $\overline{\text{MREQ}}$ ,  $\overline{\text{IORQ}}$ ,  $\overline{\text{RD}}$ , and  $\overline{\text{WR}}$  lines

to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.



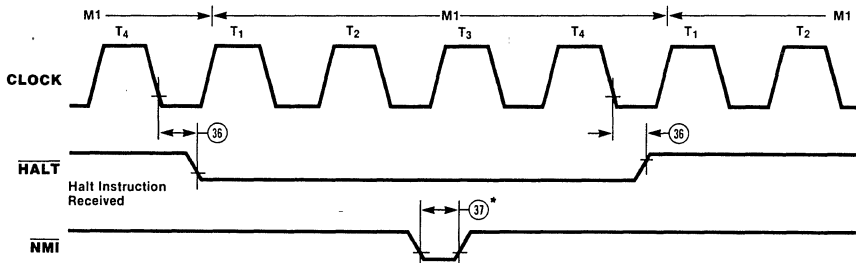
- NOTES: 1)  $T_{LM}$  = Last state of any M cycle.  
 2)  $T_x$  = An arbitrary clock cycle used by requesting device.

Figure 10. BUS Request/Acknowledge Cycle



**Halt Acknowledge Cycle.** When the CPU receives a  $\overline{\text{HALT}}$  instruction, it executes NOP states until either an  $\overline{\text{INT}}$  or  $\overline{\text{NMI}}$  input is received. When in the Halt state, the  $\overline{\text{HALT}}$  output is

active and remains so until an interrupt is received (Figure 11).  $\overline{\text{INT}}$  will also force a Halt exit.



\*Although  $\overline{\text{NMI}}$  is an asynchronous input, to guarantee its being recognized on the following machine cycle,  $\overline{\text{NMI}}$ 's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle ( $T_{1L}$ ).

Figure 11. Halt Acknowledge

**Reset Cycle.**  $\overline{\text{RESET}}$  must be active for at least three clock cycles for the CPU to properly accept it. As long as  $\overline{\text{RESET}}$  remains active, the address and data buses float, and the control outputs are inactive. Once  $\overline{\text{RESET}}$  goes inactive, two

internal T cycles are consumed before the CPU resumes normal processing operation.  $\overline{\text{RESET}}$  clears the PC register, so the first opcode fetch will be to location 0000H (Figure 12).

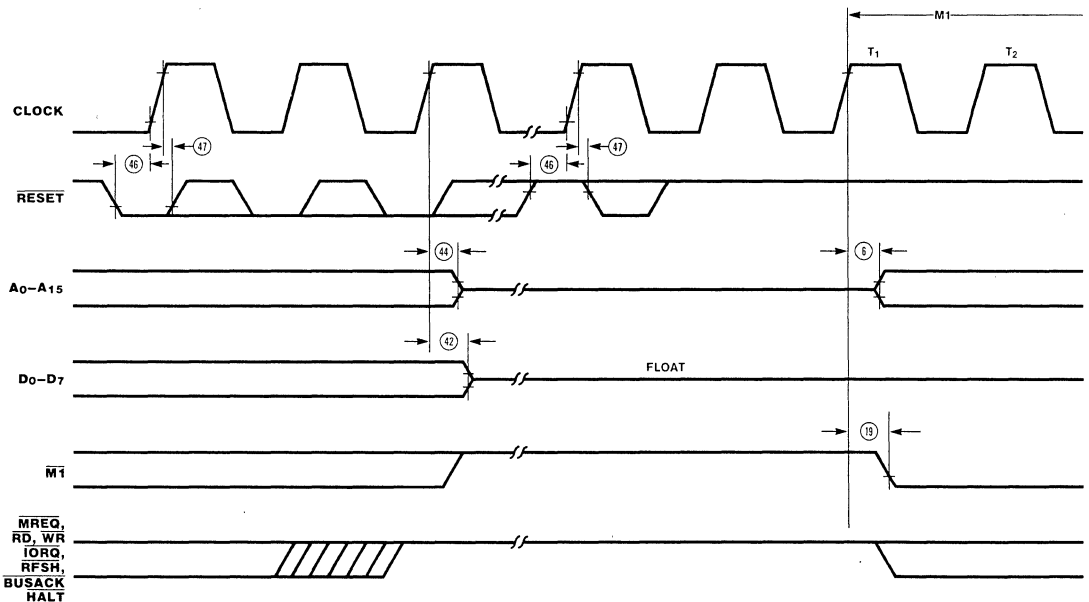


Figure 12. Reset Cycle

---

Power-Down mode of operation (Only applies to CMOS Z80 CPU).  
CMOS Z80 CPU supports Power-Down mode of operation.

This mode is also referred to as the "standby mode", and supply current for the CPU goes down as low as 10 uA (Where specified as  $I_{CC2}$ ).

---

**Power-Down Acknowledge Cycle.** When the clock input to the CPU is stopped at either a High or Low level, the CPU stops its operation and maintains all registers and control signals. However,  $I_{CC2}$  (standby supply current) is guaranteed only when the system clock is stopped at a Low

level during  $T_4$  of the machine cycle following the execution of the HALT instruction. The timing diagram for the power-down function, when implemented with the HALT instruction, is shown in Figure 13.

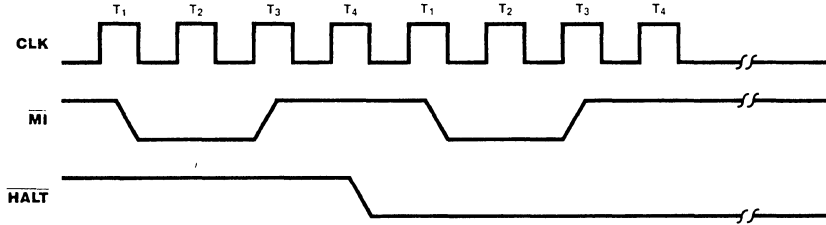


Figure 13. Power-Down Acknowledge

**Power-Down Release Cycle.** The system clock must be supplied to the CPU to release the power-down state. When the system clock is supplied to the CLK input, the CPU restarts operations from the point at which the power-down state was implemented. The timing diagrams for the release from power-down mode are shown in Figure 14.

**NOTES:**

- 1) When the external oscillator has been stopped to enter the power-down state, some warm-up time may be required to obtain a stable clock for the release.
- 2) When the HALT instruction is executed to enter the power-down state, the CPU will also enter the Halt state. An interrupt signal (either  $\overline{\text{NMI}}$  or  $\overline{\text{INT}}$ ) or a  $\overline{\text{RESET}}$  signal must be applied to the CPU after the system clock is supplied in order to release the power-down state.

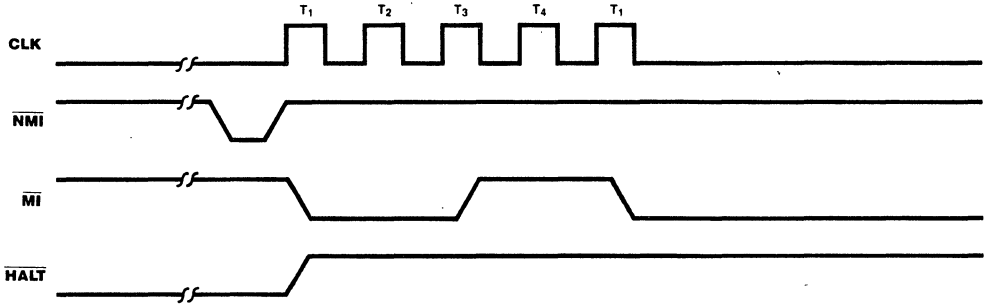


Figure 14a.

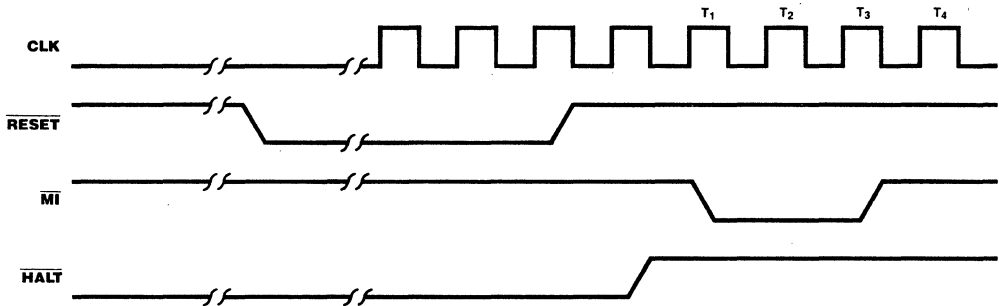


Figure 14b.

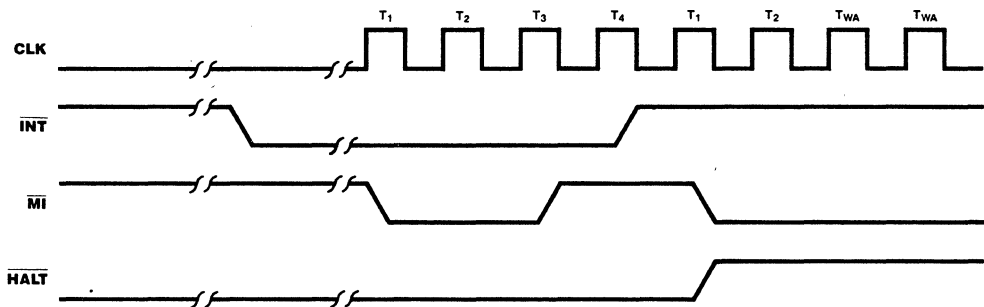


Figure 14c.

Figure 13. Power-Down Release

---

## ABSOLUTE MAXIMUM RATINGS

Voltage on  $V_{CC}$  with respect to  $V_{SS}$  . . . . .  $-0.3V$  to  $+7V$   
Voltages on all inputs with respect  
to  $V_{SS}$  . . . . .  $-0.3V$  to  $V_{CC} + 0.3V$   
Operating Ambient  
Temperature . . . . . See Ordering Information  
Storage Temperature . . . . .  $-65^{\circ}C$  to  $+150^{\circ}C$

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

---

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

The Ordering Information section lists temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.

Available operating temperature ranges are:

■ **S =  $0^{\circ}C$  to  $+70^{\circ}C$**

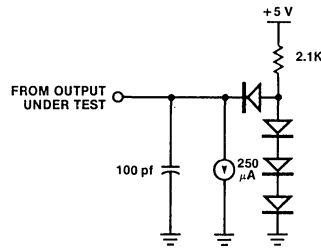
**Voltage Supply Range:**

NMOS:  $+4.75V \leq V_{CC} \leq +5.25V$

CMOS:  $+4.50V \leq V_{CC} \leq +5.50V$

■ **E =  $-40^{\circ}C$  to  $100^{\circ}C$ ,  $+4.50V \leq V_{CC} \leq +5.50V$**

All ac parameters assume a load capacitance of 100 pf. Add 10 ns delay for each 50 pf increase in load up to a maximum of 200 pf for the data bus and 100 pf for address and control lines. AC timing measurements are referenced to 1.5 volts (except for clock, which is referenced to the 10% and 90% points).



## DC CHARACTERISTICS (Z84C00/CMOS Z80 CPU)

Symbol	Parameter	Min	Max	Unit	Condition
V <sub>ILC</sub>	Clock Input Low Voltage	-0.3	0.45	V	
V <sub>IHC</sub>	Clock Input High Voltage	V <sub>CC</sub> - .6	V <sub>CC</sub> + .3	V	
V <sub>IL</sub>	Input Low Voltage	-0.3	0.8	V	
V <sub>IH</sub>	Input High Voltage	2.2	V <sub>CC</sub>	V	
V <sub>OL</sub>	Output Low Voltage		0.4	V	I <sub>OL</sub> = 2.0 mA
V <sub>OH1</sub>	Output High Voltage	2.4		V	I <sub>OH</sub> = -1.6 mA
V <sub>OH2</sub>	Output High Voltage	V <sub>CC</sub> - 0.8		V	I <sub>OH</sub> = -250 μA
I <sub>CC1</sub>	Power Supply Current	4 MHz	20	mA	V <sub>CC</sub> = 5V
		6 MHz	30	mA	V <sub>IH</sub> = V <sub>CC</sub> - 0.2V
		8 MHz	40	mA	V <sub>IL</sub> = 0.2V
		10 MHz	50	mA	
		20 MHz	100	mA	V <sub>CC</sub> = 5V
I <sub>CC2</sub>	Standby Supply Current		10	μA	V <sub>CC</sub> = 5V CLK = (0)
I <sub>LI</sub>	Input Leakage Current	-10	10	μA	V <sub>IH</sub> = V <sub>CC</sub> - 0.2V V <sub>IL</sub> = 0.2V
I <sub>LO</sub>	3-State Output Leakage Current in Float	-10	10 <sup>2</sup>	μA	V <sub>IN</sub> = 0.4 to V <sub>CC</sub> V <sub>OUT</sub> = 0.4 to V <sub>CC</sub>

1. Measurements made with outputs floating.
2. A<sub>15</sub>-A<sub>0</sub>, D<sub>7</sub>-D<sub>0</sub>, MREQ, IORQ, RD, and WR.
3. I<sub>CC2</sub> standby supply current is guaranteed only when the supplied clock is stopped at a low level during T<sub>4</sub> of the machine cycle immediately following the execution of a HALT instruction.

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C <sub>CLOCK</sub>	Clock Capacitance		10	pf
C <sub>IN</sub>	Input Capacitance		5	pf
C <sub>OUT</sub>	Output Capacitance		15	pf

T<sub>A</sub> = 25°C, f = 1 MHz.

Unmeasured pins returned to ground.

## AC CHARACTERISTICS† (Z84C00/CMOS Z80 CPU)

V<sub>cc</sub>=5.0V ± 10%, unless otherwise specified

No	Symbol	Parameter	Z84C0004**		Z84C0006		Z84C0008		Z84C0010		Z84C0020[1]		Unit	Note
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max		
1	TcC	Clock Cycle time	250*	DC	162*	DC	125*	DC	100*	DC	50*	DC	nS	
2	TwCh	Clock Pulse width (high)	110	DC	65	DC	55	DC	40	DC	20	DC	nS	
3	TwCl	Clock Pulse width (low)	110	DC	65	DC	55	DC	40	DC	20	DC	nS	
4	TfC	Clock Fall time	30		20		10		10		10		nS	
5	TrC	Clock Rise time	30		20		10		10		10		nS	
6	TdCr(A)	Address valid from Clock Rise		110		90		80		65		57	nS	[2]
7	TdA(MREQf)	Address valid to /MREQ Fall	65*		35*		20*		5*		-15*		nS	
8	TdCf(MREQf)	Clock Fall to /MREQ Fall delay		85		70		60		55		40	nS	
9	TdCr(MREQr)	Clock Rise to /MREQ Rise delay		85		70		60		55		40	nS	
10	TwMREQh	/MREQ pulse width (High)	110*		65*		45**		30*		10*		nS	[3]
11	TwMREQl	/MREQ pulse width (low)	220*		132*		100*		75*		25*		nS	[3]
12	TdCf(MERQr)	Clock Fall to /MREQ Rise delay		85		70		60		55		40	nS	
13	TdCf(RDf)	Clock Fall to /RD Fall delay		95		80		70		65		40	nS	
14	TdCr(RDr)	Clock Rise to /RD Rise delay		85		70		60		55		40	nS	
15	TsD(Cr)	Data setup time to Clock Rise	35		30		30		25		12		nS	
16	ThD(RDr)	Data hold time after /RD Rise	0		0		0		0		0		nS	
17	TsWAIT(Cf)	/WAIT setup time to Clock Fall	70		60		50		20		7.5		nS	
18	ThWAIT(Cf)	/WAIT hold time after Clock Fall	10		10		10		10		10		nS	
19	TdCr(M1f)	Clock Rise to /M1 Fall delay		100		80		70		65		45	nS	
20	TdCr(M1r)	Clock Rise to /M1 Rise delay		100		80		70		65		45	nS	
21	TdCr(RFSHf)	Clock Rise to /RFSH Fall delay		130		110		95		80		60	nS	
22	TdCr(RFSHr)	Clock Rise to /RFSH Rise delay		120		100		85		80		60	nS	
23	TdCf(RDf)	Clock Fall to /RD Rise delay		85		70		60		55		40	nS	
24	TdCr(RDf)	Clock Rise to /RD Fall delay		85		70		60		55		40	nS	
25	TsD(Cf)	Data setup to Clock Fall during M2, M3, M4 or M5 cycles	50		40		30		25		12		nS	
26	TdA(IORQf)	Address stable prior to /IORQ Fall	180*		107*		75*		50*		0*		nS	
27	TdCr(IORQf)	Clock Rise to /IORQ Fall delay		75		65		55		50		40	nS	
28	TdCf(IORQr)	Clock Fall to /IORQ Rise delay		85		70		60		55		40	nS	
29	TdD(WRf)Mw	Data stable prior to /WR Fall	80*		22*		5*		40*		-10*		nS	
30	TdCf(WRf)	Clock Fall to /WR Fall delay		80		70		60		55		40	nS	
31	TwWR	/WR pulse width	220*		132*		100*		75*		25*		nS	
32	TdCf(WRr)	Clock Fall to /WR Rise delay		80		70		60		55		40	nS	
33	TdD(WRf)O	Data stable prior to /WR Fall	-10*		-55*		-55*		-10*		-30*		nS	
34	TdCr(WRf)	Clock Rise to /WR Fall delay		65		60		60		50		40	nS	
35	TdWRr(D)	Data stable from /WR Rise	60*		30*		15*		10*		0*		nS	
36	TdCf(HALT)	Clock Fall to /HALT 'L' or 'H'		300		260		225		90		70	nS	
37	TwNMI	/NMI pulse width	80		60		60		60		60		nS	
38	TsBUSREQ (Cr)	/BUSREQ setup time to Clock Rise	50		50		40		30		15		nS	

\*For clock periods other than the minimums shown, calculate parameters using the table on the following page.

Calculated values above assumed TrC = TfC = 20 ns.

†Units in nanoseconds (ns).

†† For loading ≥ 50 pf. Decrease width by 10 ns for each additional 50 pf.

\*\*4 MHz CMOS Z80 is obsoleted and replaced by 6 MHz

## AC CHARACTERISTICS† (Z84C00/CMOS Z80 CPU; Continued)

V<sub>CC</sub> = 5.0V ± 10%, unless otherwise specified

No	Symbol	Parameter	Z84C0004**		Z84C0006		Z84C0008		Z84C0010		Z84C0020[1]		Unit	Note
			Min	Max	Min	Max	Min	Max	Min	Max	Min	Max		
39	ThBUSREQ (Cr)	/BUSREQ hold time after Clock Rise	10		10		10		10		10		nS	
40	TdCr (BUSACKf)	Clock Rise to /BASACK Fall delay		100		90		80		75		40	nS	
41	TdCf (BUSACKr)	Clock Fall to /BASACK Rise delay		100		90		80		75		40	nS	
42	TdCr(Dz)	Clock Rise to Data float delay		90		80		70		65		40	nS	
43	TdCr(CTz)	Clock Rise to Control Outputs Float Delay (/MREQ, /IORQ, /RD and /WR)		80		70		60		65		40	nS	
44	TdCr(Az)	Clock Rise to Address float delay		90		80		70		75		40	nS	
45	TdCTr(A)	Address Hold time from /MREQ, /IORQ, /RD or /WR	80*		35*		20*		20*		0*		nS	
46	TsRESET(Cr)	/RESET to Clock Rise setup time	60		60		45		40		15		nS	
47	ThRESET(Cr)	/RESET to Clock Rise Hold time	10		10		10		10		10		nS	
48	TsINTf(Cr)	/INT Fall to Clock Rise Setup Time	80		70		55		50		15		nS	
49	ThINTR(Cr)	/INT Rise to Clock Rise Hold Time	10		10		10		10		10		nS	
50	TdM1f (IORQf)	/M1 Fall to /IORQ Fall delay	565*		359*		270*		220*		100*		nS	
51	TdCf(IORQf)	/Clock Fall to /IORQ Fall delay		85		70		60		55		45	nS	
52	TdCf(IORQr)	Clock Rise to /IORQ Rise delay		85		70		60		55		45	nS	
53	TdCf(D)	Clock Fall to Data Valid delay		150		130		115		110		75	nS	

### Notes:

\* For Clock periods other than the minimum shown, calculate parameters using the following table.

Calculated values above assumed TrC = TfC = maximum.

\*\* 4 MHz CMOS Z80 is obsolete and replaced by 6 MHz

[1] Z84C0020 parameters are guaranteed with 50pF load Capacitance.

[2] If Capacitive Load is other than 50pF, please use Figure 1. to calculate the value.

[3] Increasing delay by 10nS for each 50pF increase in loading, 200pF max for data lines, and 100pF for control lines.

## FOOTNOTES TO AC CHARACTERISTICS

No	Symbol	Parameter	Z84C0004**	Z84C0006	Z84C0008	Z84C0010	Z84C0020
1	TcC	TwCh + TwCl + TrC + TfC					
7	TdA(MREQf)	TwCh + TfC	-65	-50	-45	-45	-45
10	TwMREQh	TwCh + TfC	-20	-20	-20	-20	-20
11	TwMREQl	TcC	-30	-30	-25	-25	-25
26	TdA(IORQf)	TcC	-70	-55	-50	-50	-50
29	TdD(WRf)	TcC	-170	-140	-120	-60	-60
31	TwWR	TcC	-30	-30	-25	-25	-25
33	TdD(WRf)	TwCl + TrC	-140	-140	-120	-60	-60
35	TdWRr(D)	TwCl + TrC	-70	-55	-50	-40	-25
45	TdCTr(A)	TwCl + TrC	-50	-50	-45	-30	-30
50	TdM1f(IORQf)	2TcC + TwCh + TfC	-65	-50	-45	-30	-30

AC Test Conditions: V<sub>IH</sub> = 2.0 V  
V<sub>IL</sub> = 0.8 V

V<sub>OH</sub> = 1.5 V  
V<sub>OL</sub> = 1.5 V

V<sub>IHC</sub> = V<sub>CC</sub> - 0.6 V  
V<sub>ILC</sub> = 0.45 V

FLOAT = ±0.5 V

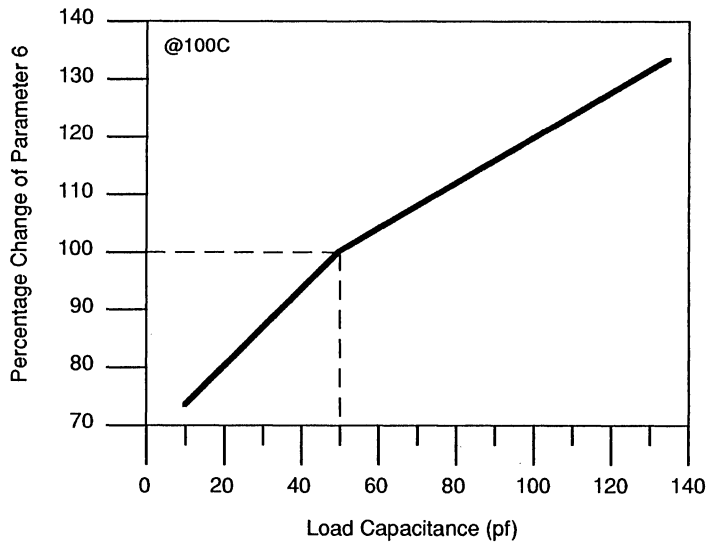


Figure 1. Address Delay Characteristics  
(Parameter 6)

## DC CHARACTERISTICS (Z8400/NMOS Z80 CPU)

All parameters are tested unless otherwise noted.

Symbol	Parameter	Min	Max	Unit	Test Condition
$V_{ILC}$	Clock Input Low Voltage	-0.3	0.45	V	
$V_{IHC}$	Clock Input High Voltage	$V_{CC} - .6$	$V_{CC} + .3$	V	
$V_{IL}$	Input Low Voltage	-0.3	0.8	V	
$V_{IH}$	Input High Voltage	2.0 <sup>1</sup>	$V_{CC}$	V	
$V_{OL}$	Output Low Voltage		0.4	V	$I_{OL} = 2.0 \text{ mA}$
$V_{OH}$	Output High Voltage	2.4 <sup>1</sup>		V	$I_{OH} = -250 \mu\text{A}$
$I_{CC}$	Power Supply Current		200	mA	Note 3
$I_{LI}$	Input Leakage Current		10	$\mu\text{A}$	$V_{IN} = 0 \text{ to } V_{CC}$
$I_{LO}$	3-State Output Leakage Current in Float	-10	10 <sup>2</sup>	$\mu\text{A}$	$V_{OUT} = 0.4 \text{ to } V_{CC}$

1. For military grade parts, refer to the Z80 Military Electrical Specification.

2.  $A_{15}$ - $A_0$ ,  $D_7$ - $D_0$ ,  $MREQ$ ,  $IORQ$ ,  $RD$ , and  $WR$ .

3. Measurements made with outputs floating.

## CAPACITANCE

Guaranteed by design and characterization.

Symbol	Parameter	Min	Max	Unit
$C_{CLOCK}$	Clock Capacitance		35	pf
$C_{IN}$	Input Capacitance		5	pf
$C_{OUT}$	Output Capacitance		15	pf

NOTES:

$T_A = 25^\circ\text{C}$ ,  $f = 1 \text{ MHz}$ .

Unmeasured pins returned to ground.



## AC CHARACTERISTICS† (Z8400/NMOS Z80 CPU)

Number	Symbol	Parameter	Z0840004		Z0840006		Z0840008	
			Min	Max	Min	Max	Min	Max
1	TcC	Clock Cycle Time	250*		162*		125*	
2	TwCh	Clock Pulse Width (High)	110	2000	65	2000	55	2000
3	TwCl	Clock Pulse Width (Low)	110	2000	65	2000	55	2000
4	TfC	Clock Fall Time		30		20		10
5	TrC	Clock Rise Time		30		20		10
6	TdCr(A)	Clock ↑ to Address Valid Delay		110		90		80
7	TdA(MREQf)	Address Valid to $\overline{\text{MREQ}}$ ↓ Delay	65*		35*		20*	
8	TdCf(MREQf)	Clock ↓ to $\overline{\text{MREQ}}$ ↓ Delay		85		70		60
9	TdCr(MREQr)	Clock ↑ to $\overline{\text{MREQ}}$ ↑ Delay		85		70		60
10	TwMREQh	$\overline{\text{MREQ}}$ Pulse Width (High)	110*††		65*††		45*††	
11	TwMREQl	$\overline{\text{MREQ}}$ Pulse Width (Low)	220*††		135*††		100*††	
12	TdCf(MREQr)	Clock ↓ to $\overline{\text{MREQ}}$ ↑ Delay		85		70		60
13	TdCf(RDf)	Clock ↓ to $\overline{\text{RD}}$ ↓ Delay		95		80		70
14	TdCr(RDr)	Clock ↑ to $\overline{\text{RD}}$ ↑ Delay		85		70		60
15	TsD(Cr)	Data Setup Time to Clock ↑	35		30		30	
16	ThD(RDr)	Data Hold Time to $\overline{\text{RD}}$ ↑		0		0		0
17	TsWAIT(Cf)	$\overline{\text{WAIT}}$ Setup Time to Clock ↓	70		60		50	
18	ThWAIT(Cf)	$\overline{\text{WAIT}}$ Hold Time after Clock ↓		0		0		0
19	TdCr(M1f)	Clock ↑ to $\overline{\text{M1}}$ ↓ Delay		100		80		70
20	TdCr(M1r)	Clock ↑ to $\overline{\text{M1}}$ ↑ Delay		100		80		70
21	TdCr(RFSHf)	Clock ↑ to $\overline{\text{RFSH}}$ ↓ Delay		130		110		95
22	TdCr(RFSHr)	Clock ↑ to $\overline{\text{RFSH}}$ ↑ Delay		120		100		85
23	TdCf(RDr)	Clock ↓ to $\overline{\text{RD}}$ ↓ Delay		85		70		60
24	TdCr(RDf)	Clock ↑ to $\overline{\text{RD}}$ ↓ Delay		85		70		60
25	TsD(Cf)	Data Setup to Clock ↓ during M <sub>2</sub> , M <sub>3</sub> , M <sub>4</sub> , or M <sub>5</sub> Cycles	50		40		30	
26	TdA(IORQf)	Address Stable prior to $\overline{\text{IORQ}}$ ↓	180*		110*		75*	
27	TdCr(IORQf)	Clock ↑ to $\overline{\text{IORQ}}$ ↓ Delay		75		65		55
28	TdCf(IORQr)	Clock ↓ to $\overline{\text{IORQ}}$ ↑ Delay		85		70		60
29	TdD(WRf)	Data Stable prior to $\overline{\text{WR}}$ ↓	80*		25*		5*	
30	TdCf(WRf)	Clock ↓ to $\overline{\text{WR}}$ ↓ Delay		80		70		60
31	TwWR	$\overline{\text{WR}}$ Pulse Width	220*		135*		100*	
32	TdCf(WRr)	Clock ↓ to $\overline{\text{WR}}$ ↑ Delay		80		70		60
33	TdD(WRf)	Data Stable prior to $\overline{\text{WR}}$ ↓	-10*		-55*		55*	
34	TdCr(WRf)	Clock ↑ to $\overline{\text{WR}}$ ↓ Delay		65		60		55
35	TdWRr(D)	Data Stable from $\overline{\text{WR}}$ ↑	60*		30*		15*	
36	TdCf(HALT)	Clock ↓ to $\overline{\text{HALT}}$ ↑ or ↓		300		260		225
37	TwNMI	$\overline{\text{NMI}}$ Pulse Width	80		70		60*	
38	TsBUSREQ(Cr)	$\overline{\text{BUSREQ}}$ Setup Time to Clock ↑	50		50		40	

\*For clock periods other than the minimums shown, calculate parameters using the table on the following page. Calculated values above assumed TrC = TfC = 20 ns.

†Units in nanoseconds (ns).

†† For loading  $\geq 50$  pf., Decrease width by 10 ns for each additional 50 pf.

## AC CHARACTERISTICS† (Z8400/NMOS Z80 CPU; Continued)

Number	Symbol	Parameter	Z0840004		Z0840006		Z0840008	
			Min	Max	Min	Max	Min	Max
39	ThBUSREQ(Cr)	BUSREQ Hold Time after Clock ↑	0		0		0	
40	TdCr(BUSACKf)	Clock ↑ to BUSACK ↓ Delay		100		90		80
41	TdCf(BUSACKr)	Clock ↓ to BUSACK ↑ Delay		100		90		80
42	TdCr(Dz)	Clock ↑ to Data Float Delay		90		80		70
43	TdCr(CTz)	Clock ↑ to Control Outputs Float Delay (MREQ, IORQ, RD, and WR)		80		70		60
44	TdCr(Az)	Clock ↑ to Address Float Delay		90		80		70
45	TdCTr(A)	MREQ ↑, IORQ ↑, RD ↑, and WR ↑ to Address Hold Time	80*		35*		20*	
46	TsRESET(Cr)	RESET to Clock ↑ Setup Time	60		60		45	
47	ThRESET(Cr)	RESET to Clock ↑ Hold Time		0		0		0
48	TsINTf(Cr)	INT to Clock ↑ Setup Time	80		70		55	
49	ThINTr(Cr)	INT to Clock ↑ Hold Time		0		0		0
50	TdM1f(IORQf)	M1 ↓ to IORQ ↓ Delay	565*		365*		270*	
51	TdCf(IORQf)	Clock ↓ to IORQ ↓ Delay		85		70		60
52	TdCf(IORQr)	Clock ↑ IORQ ↑ Delay		85		70		60
53	TdCf(D)	Clock ↓ to Data Valid Delay		150		130		115

\*For clock periods other than the minimums shown, calculate parameters using the following table. Calculated values above assumed  $T_C = T_{IC} = 20$  ns.

†Units in nanoseconds (ns).

## FOOTNOTES TO AC CHARACTERISTICS

Number	Symbol	General Parameter	Z0840004	Z0840006	Z0840008
1	TcC	$T_{wCh} + T_{wCl} + T_{rC} + T_{fC}$			
7	TdA(MREQf)	$T_{wCh} + T_{fC}$	- 65	- 50	- 45
10	TwMREQh	$T_{wCh} + T_{fC}$	- 20	- 20	- 20
11	TwMREQl	TcC	- 30	- 30	- 25
26	TdA(IORQf)	TcC	- 70	- 55	- 50
29	TdD(WRf)	TcC	- 170	- 140	- 120
31	TwWR	TcC	- 30	- 30	- 25
33	TdD(WRf)	$T_{wCl} + T_{rC}$	- 140	- 140	- 120
35	TdWRr(D)	$T_{wCl} + T_{rC}$	- 70	- 55	- 50
45	TdCTr(A)	$T_{wCl} + T_{rC}$	- 50	- 50	- 45
50	TdM1f(IORQf)	$2T_{cC} + T_{wCh} + T_{fC}$	- 65	- 50	- 45

AC Test Conditions:

$$V_{IH} = 2.0 \text{ V}$$

$$V_{IL} = 0.8 \text{ V}$$

$$V_{IHC} = V_{CC} - 0.6 \text{ V}$$

$$V_{ILC} = 0.45 \text{ V}$$

$$V_{OH} = 1.5 \text{ V}$$

$$V_{OL} = 1.5 \text{ V}$$

$$\text{FLOAT} = \pm 0.5 \text{ V}$$

---

## Z8410/Z84C10 NMOS/CMOS Z80<sup>®</sup> DMA Direct Memory Access Controller

### FEATURES

- Transfers, searches, and search/transfers in Byte-at-a-Time, Burst, or Continuous modes. Cycle length and edge timing can be programmed to match the speed of any port.
- Dual port addresses (source and destination) generated for memory-to-I/O, memory-to-memory, or I/O-to-I/O operations. Addresses may be fixed or automatically incremented/decremented.
- Next-operation loading without disturbing current operations via buffered starting-address registers. An entire previous sequence can be repeated automatically.
- Extensive programmability of functions. CPU can read complete channel status.
- **NMOS version for cost sensitive performance solutions**
- **CMOS version for the designs requiring low power consumption**
- **NMOS Z0841004 - 4MHz**
- **CMOS Z84C1006 - DC to 6.17 MHz, Z84C1008 - DC to 8 MHz**
- **6 MHz version supports 6.144 MHz CPU clock operation clock.**
- Standard Z80 Family bus-request and prioritized interrupt-request daisy chains implemented without external logic. Sophisticated, internally modifiable interrupt vectoring.
- Direct interfacing to system buses without external logic.

### GENERAL DESCRIPTION

The Z80 DMA (Direct Memory Access), hereafter referred to as Z80 DMA or DMA, is a powerful and versatile device for controlling and processing transfers of data. Its basic

function of managing CPU-independent transfers between two ports is augmented by an array of features that optimize transfer speed and control with little or no external logic in systems using an 8- or 16-bit data bus and a 16-bit address bus.

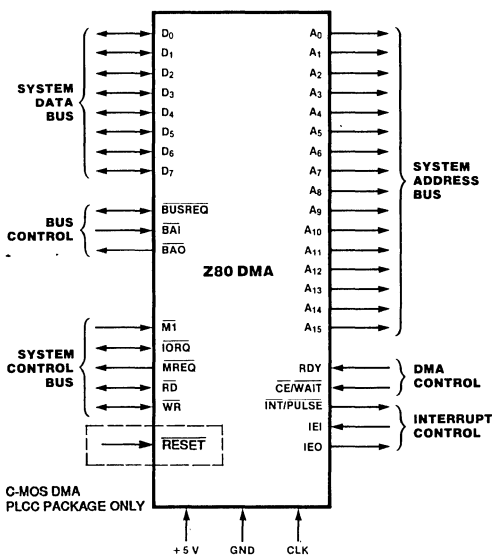


Figure 1. Pin Functions

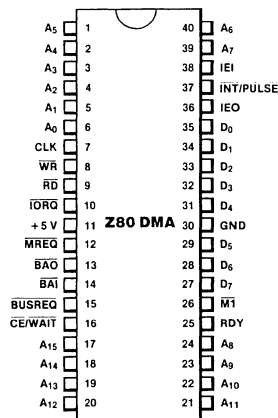


Figure 2a. 40-pin Dual-In-Line Package (DIP), Pin Assignments

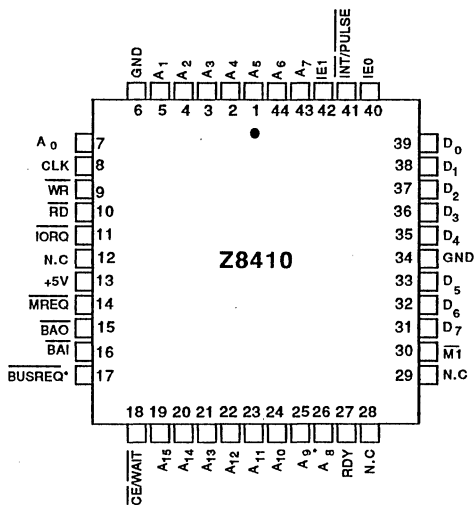


Figure 2b. Z8410 NMOS Z80 DMA  
44-Pin PLCC Pinout

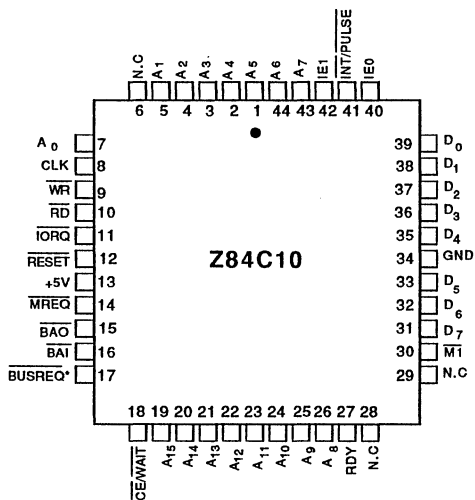


Figure 2c. Z84C10 CMOS Z80 DMA  
PLCC Pinout

Transfers can be done between any two ports (source and destination), including memory-to-I/O, memory-to-memory, and I/O-to-I/O. Dual port addresses are automatically generated for each transaction and may be either fixed or incrementing/decrementing. In addition, bit-maskable byte searches can be performed either concurrently with transfers or as an operation in itself.

The Z80 DMA contains direct interfacing to, and independent control of, system buses, as well as

sophisticated bus and interrupt controls. Many programmable features, including variable cycle timing and auto-restart, minimize CPU software overhead. They are especially useful in adapting this special-purpose transfer processor to a broad variety of memory, I/O and CPU environments.

**The Z80 DMA is packaged in a 40-pin plastic or Cerdip DIP, or 44-pin PCC. It uses a single +5V power supply and the standard Z80 Family single-phase clock.**

## FUNCTIONAL DESCRIPTION

**Classes of Operation.** The Z80 DMA has three basic classes of operation:

- Transfers of data between two ports (memory or I/O peripheral)
- Searches for a particular 8-bit maskable byte at a single port in memory or an I/O peripheral
- Combined transfers with simultaneous search between two ports

Figure 4 illustrates the basic functions served by these classes of operation.

During a transfer, the DMA assumes control of the system address and data buses. Byte by byte, data is read from one addressable port and written to the other addressable port. The ports may be programmed to be either system main memory or peripheral I/O devices. Thus, a block of data may be written from one peripheral to another, from one area of main memory to another, or from a peripheral to main memory and vice versa.

During a search-only operation, data is read from the source port and compared byte by byte with a DMA-internal register containing a programmable match byte. This match byte may optionally be masked so that only certain bits within the match byte are compared. Search rates up to 2M bytes per second can be obtained with the 4 MHz Z80 DMA.

In combined searches and transfers, data is transferred between two ports while simultaneously searching for a bit-maskable byte match.

Data transfers or searches can be programmed to stop, or interrupt, under various conditions. In addition, CPU-readable status bits can be programmed to reflect the condition.

**Modes of Operation.** The Z80 DMA can be programmed to operate in one of three transfer and/or search modes:

- *Byte-at-a-Time:* data operations are performed one byte at a time. Between each byte operation the system buses are released to the CPU. The buses are requested again for each succeeding byte operation.

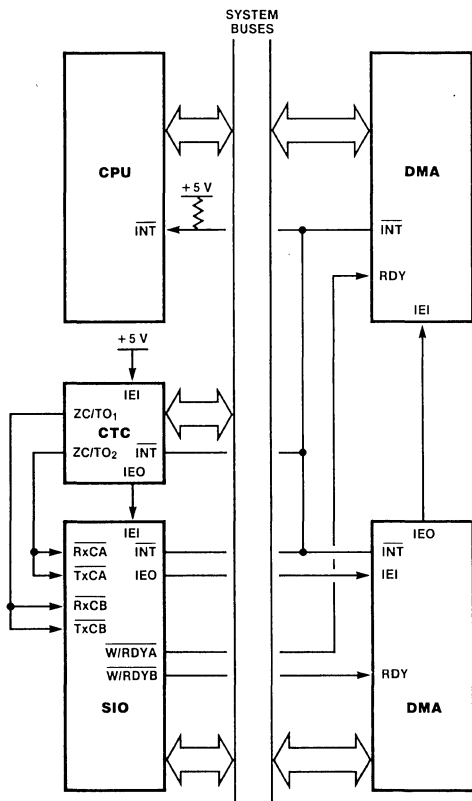


Figure 3. Typical Z80 Environment

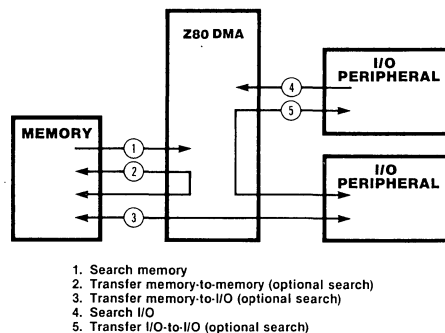


Figure 4. Basic Functions of the Z80 DMA

- **Burst:** data operations continue until a port's Ready line to the DMA goes inactive. The DMA then stops and releases the system buses after completing its current byte operation.
- **Continuous:** data operations continue until the end of the programmed block of data is reached before the system buses are released. If a port's Ready line goes inactive before this occurs, the DMA simply pauses until the Ready line comes active again.

In all modes, once a byte of data is read into the DMA, the operation on the byte will be completed in an orderly fashion, regardless of the state of other signals (including a port's Ready line).

Due to the DMA's high-speed buffered method of reading data, operations on one byte are not completed until the next byte is read in. This means that total transfer or search block lengths must be two or more bytes, and that block lengths programmed into the DMA must be one byte less than the desired block length (count is N-1 where N is the block length).

**Commands and Status.** The Z80 DMA has several writable control registers and readable status registers available to the CPU. Control bytes can be written to the DMA whenever the DMA is not controlling the system buses, but the act of writing a control byte to the DMA disables the DMA until it is again enabled by a specific command. Status bytes can also be read at any such time, but writing the Read Status Byte command or the Initiate Read Sequence command disables the DMA.

Control bytes to the DMA include those which affect immediate command actions such as enable, disable, reset, load starting-address buffers, continue, clear counters, and clear status bits. In addition, many mode-setting control bytes can be written, including mode and class of operation, port configuration, starting addresses, block length, address counting rule, match and match-mask byte, interrupt conditions, interrupt vector, status-affects-vector condition, pulse counting, auto restart, Ready-line and Wait-line rules, and read mask.

Readable status registers include a general status byte reflecting Ready-line, end-of-block, byte-match, and interrupt conditions, as well as 2-byte registers for the current byte count, Port A address, and Port B address.

**Variable Cycle.** The Z80 DMA has the unique feature of programmable operation-cycle length. This is valuable in tailoring the DMA to the particular requirements of other system components (fast or slow) and maximizes the data-transfer rate. It also eliminates external logic for signal conditioning.

There are two aspects to the variable cycle feature. First, the entire read and write cycles (periods) associated with the source and destination ports can be independently programmed as 2, 3, or 4 T-cycles long (more if Wait cycles are used), thereby increasing or decreasing the speed with which all DMA signals change (Figure 5).

Second, the four signals in each port specifically associated with transfers of data (I/O Request, Memory Request, Read and Write) can each have its active trailing edge terminated one-half T-cycle early. This adds a further dimension of flexibility and speed, allowing such things as shorter-than-normal Read or Write signals that go inactive before data starts to change.

**Address Generation.** Two 16-bit addresses are generated by the Z80 DMA for every transfer operation, one address for the source port and another for the destination port. Each address can be either variable or fixed. Variable addresses can increment or decrement from the programmed starting address. The fixed-address capability eliminates the need for separate enabling wires to I/O ports.

Port addresses are multiplexed onto the system address bus, depending on whether the DMA is reading the source port or writing to the destination port. Two readable address counters (2 bytes each) keep the current address of each port.

**Auto Restart.** The starting addresses of either port can be reloaded automatically at the end of a block. This option is selected by the Auto Restart control bit. The byte counter is cleared when the addresses are reloaded.

The Auto Restart feature relieves the CPU of software overhead for repetitive operations such as CRT refresh and many others. Moreover, when the CPU has access to the buses during byte-at-a-time or burst transfers, different starting addresses can be written into buffer registers during transfers, causing the Auto Restart to begin at a new location.

**Interrupts.** The Z80 DMA can be programmed to interrupt the CPU on four conditions:

- Interrupt on Ready (before requesting bus)
- Interrupt on Match
- Interrupt on End of Block
- Interrupt on Match and End of Block

Any of these interrupts causes an interrupt-pending status bit to be set, and each of them can optionally alter the DMA's interrupt vector. Due to the buffered constraint mentioned under "Modes of Operation," interrupts on Match at End of Block are caused by matches to the byte just prior to the last byte in the block.

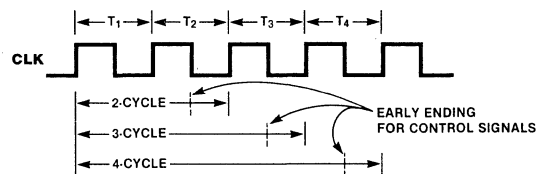


Figure 5. Variable Cycle Length

The DMA shares the Z80 Family's elaborate interrupt scheme, which provides fast interrupt service in real-time applications. In a Z80 CPU environment, the DMA passes its internally modifiable 8-bit interrupt vector to the CPU, which adds an additional eight bits to form the memory address of the interrupt-routine table. This table contains the address of the beginning of the interrupt routine itself. In this process, CPU control is transferred directly to the interrupt routine, so that the next instruction executed after an interrupt acknowledge is the first instruction of the interrupt routine itself.

**Pulse Generation.** External devices can keep track of how many bytes have been transferred by using the DMA's pulse output, which provides a signal at 256-byte intervals. The interval sequence may be offset at the beginning by 1 to 255 bytes.

The Interrupt line outputs the pulse signal in a manner that prevents misinterpretation by the CPU as an interrupt request, since it only appears when the Bus Request and Bus Acknowledge lines are both active.

## PIN DESCRIPTION

**A<sub>0</sub>-A<sub>15</sub>.** *System Address Bus* (output, 3-state). Addresses generated by the DMA are sent to both source and destination ports (main memory or I/O peripherals) on these lines.

**BAI.** *Bus Acknowledge In* (input, active Low). Signals that the system buses have been released for DMA control. In multiple-DMA configurations, the BAI pin of the highest priority DMA is normally connected to the Bus Acknowledge pin of the CPU. Lower-priority DMAs have their BAI connected to the BAO of a higher-priority DMA.

**BAO.** *Bus Acknowledge Out* (output, active Low). In a multiple-DMA configuration, this pin signals that no other higher-priority DMA has requested the system buses. BAI and BAO form a daisy chain for multiple-DMA priority resolution over bus control.

**BUSREQ.** *Bus Request* (bidirectional, active Low, open-drain). As an output, it sends requests for control of the system address bus, data bus, and control bus to the CPU. As an input when multiple DMAs are strung together in a priority daisy chain via BAI and BAO, it senses when another DMA has requested the buses and causes this DMA to refrain from bus requesting until the other DMA is finished. Because it is a bidirectional pin, there cannot be any buffers between this DMA and any other DMA. It can, however, have a buffer between it and the CPU because it is unidirectional into the CPU. A pull-up resistor is connected to this pin.

**CE/WAIT.** *Chip Enable and Wait* (input, active Low). Normally this functions only as a CE line, but it can also be programmed to serve a WAIT function. As a CE line from the CPU, it becomes active when WR or RD and IORQ are active and the I/O port address on the system address bus is the DMA's address, thereby allowing a transfer of control, command bytes from the CPU to the DMA, or status bytes from the DMA to the CPU. As a WAIT line from memory or I/O devices, after the DMA has received a bus-request acknowledge from the CPU, it causes wait states to be inserted in the DMA's operation cycles thereby slowing the DMA to a speed that matches the memory or I/O device.

**CLK.** *System Clock* (input). Standard Z80 single-phase clock.

**D<sub>0</sub>-D<sub>7</sub>.** *System Data Bus* (bidirectional, 3-state). Commands from the CPU, DMA status, and data from memory or I/O

peripherals are transferred on these lines.

**IEI.** *Interrupt Enable In* (input, active High). This is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an interrupt from this DMA. Thus, this signal blocks lower-priority devices from interrupting while a higher-priority device is being serviced by its CPU interrupt service routine.

**INT/PULSE.** *Interrupt Request* (output, active Low, open-drain). While the CPU is the bus master, this output requests a CPU interrupt. The CPU acknowledges the interrupt by pulling its IORQ output Low during an M1 cycle. It is typically connected to the INT pin of the CPU with a pullup resistor and tied to all other INT pins in the system. This pin can also be used to generate periodic pulses to an external device when the DMA is bus master (i.e., the CPU's BUSREQ and BUSACK lines are both Low and the CPU cannot see interrupts). While the DMA is the bus master, this output can be programmed to pulse each time 256 transfers have occurred.

**IORQ.** *Input/Output Request* (bidirectional, active Low, 3-state). As an input, this indicates that the lower half of the address bus holds a valid I/O port address for transfer of control or status bytes from or to the CPU, respectively; this DMA is the addressed port if its CE pin and its WR or RD pins are simultaneously active. As an output, after the DMA has taken control of the system buses, it indicates that the lower half of the address bus holds a valid port address for another I/O device involved in a DMA transfer of data. When IORQ and M1 are both active simultaneously, an interrupt acknowledge is indicated.

**M1.** *Machine Cycle One* (input, active Low). Indicates that the current CPU machine cycle is an instruction fetch. It is used by the DMA to decode the return-from-interrupt instruction (RETI, ED-4D) sent by the CPU. During two-byte instruction fetches, M1 is active as each opcode byte is fetched. An interrupt acknowledge is indicated when both M1 and IORQ are active. On CMOS DMA, M1 signal has another function. When M1 occurs without an active RD or IORQ for at least two clock cycles, the DMA is reset.



**MREQ.** *Memory Request* (output, active Low, 3-state). This indicates that the address bus holds a valid address for a memory read or write operation. After the DMA has taken control of the system buses, it indicates a DMA transfer request from or to memory.

**RD.** *Read* (bidirectional, active Low, 3-state). As an input, this indicates that the CPU wants to read status bytes from the DMA's read registers. As an output, after the DMA has taken control of the system buses, it indicates a DMA-controlled read from a memory or I/O port address.

**RESET.** *Reset* (CMOS PLCC version only: input, active Low). A low on this line resets the DMA.

**RDY.** *Ready* (input, programmable active Low or High). This is monitored by the DMA to determine when a peripheral device associated with a DMA port is ready for a read or write operation. Depending on the mode of DMA operation (Byte, Burst, or Continuous), the RDY line indirectly controls DMA activity by causing the **BUSREQ** line to go Low or High.

**WR.** *Write* (bidirectional, active Low, 3-state). As an input, this indicates that the CPU wants to write control or command bytes to the DMA write registers. As an output, after the DMA has taken control of the system buses, it indicates a DMA-controlled write to a memory or I/O port address.

## INTERNAL STRUCTURE

The internal structure of the Z80 DMA includes driver and receiver circuitry for interfacing with an 8-bit system data bus, a 16-bit system address bus, and system control lines (Figure 6). In a Z80 CPU environment, the DMA can be tied directly to the analogous pins on the CPU (Figure 7) with no additional buffering, except for the **CE/WAIT** line.

The DMA's internal data bus interfaces with the system data bus and services all internal logic and registers. Addresses generated from this logic for Ports A and B (source and destination) of the DMA's single transfer channel are multiplexed onto the system address bus.

Specialized logic circuits in the DMA are dedicated to the various functions of external bus interfacing, internal bus control, byte matching, byte counting, periodic pulse generation, CPU interrupts, bus requests, and address generation. A set of 21 writable control registers and seven readable status registers provides the means by which the CPU governs and monitors the activities of these logic circuits. All registers are eight bits wide, with double-byte information stored in adjacent registers. The two address counters (two bytes each) for Ports A and B are buffered by the two starting addresses.

The 21 writable control registers are organized into seven base-register groups, most of which have multiple registers. The base registers in each writable group contain both

control/command bits and pointer bits that can be set to address other registers within the group. The seven readable status registers have no analogous second-level registers.

The registers are designated as follows, according to their base-register groups:

WR0-WR6—Write Register groups 0 through 6  
(7 base registers plus 14 associated registers)

RR0-RR6—Read Registers 0 through 6

Writing to a register within a write-register group involves first writing to the base register, with the appropriate pointer bits set, then writing to one or more of the other registers within the group. All seven of the readable status registers are accessed sequentially according to a programmable mask contained in one of the writable registers. The section entitled Programming explains this in more detail.

A pipelining scheme is used for reading data in. The programmed block length is the number of bytes compared to the byte counter, which increments at the end of each cycle. In searches, data byte comparisons with the match byte are made during the read cycle of the next byte. Matches are, therefore, discovered only after the next byte is read in.

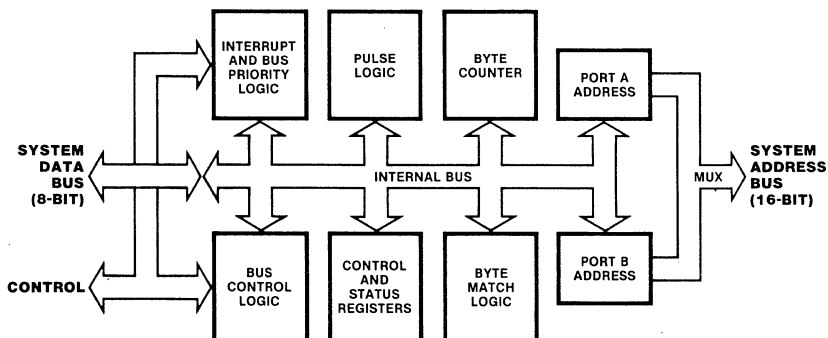


Figure 6. Block Diagram

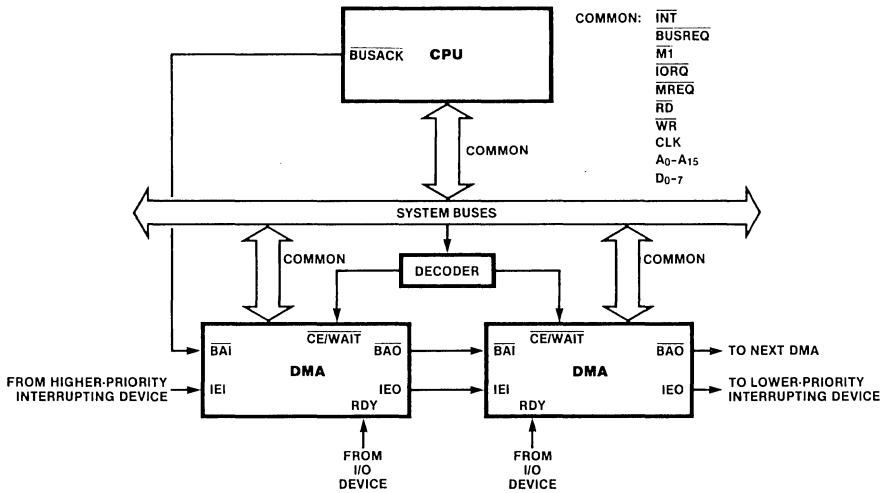


Figure 7. Multiple-DMA Interconnection to the Z80 CPU

In multiple-DMA configurations, interrupt-request daisy chains are prioritized by the order in which their IEI and IEO lines are connected. The system bus, however, may not be pre-empted. Any DMA that gains access to the system buses keeps them until it is finished.

Read Registers	
RR0	Status byte
RR1	Byte counter (low byte)
RR2	Byte counter (high byte)
RR3	Port A address counter (low byte)
RR4	Port A address counter (high byte)
RR5	Port B address counter (low byte)
RR6	Port B address counter (high byte)

Write Registers	
WR0	Base register byte Port A starting address (low byte) Port A starting address (high byte) Block length (low byte) Block length (high byte)
WR1	Base register byte Port A variable-timing byte
WR2	Base register byte Port B variable-timing byte
WR3	Base register byte Mask byte Match byte
WR4	Base register byte Port B starting address (low byte) Port B starting address (high byte) Interrupt control byte Pulse control byte Interrupt vector
WR5	Base register byte
WR6	Base register byte Read mask

## PROGRAMMING

The Z80 DMA has two programmable fundamental states: (1) an enabled state, in which it can gain control of the system buses and direct the transfer of data between ports, and (2) a disabled state, in which it can initiate neither bus requests nor data transfers. When the DMA is powered up or reset by any means, it is automatically placed into the disabled state. Program commands can be written to it by the CPU in either state, but this automatically puts the DMA in the disabled state, which is maintained until an enable command is issued by the CPU. The CPU must program the DMA in advance of any data search or transfer by addressing it as an I/O port and sending a sequence of control bytes using an Output instruction (such as OTIR for the Z80 CPU).

**Reading.** (Figure 8a) The Read Registers (RR0-RR6) are read by the CPU by addressing the DMA as an I/O port using an Input instruction (such as INIR for the Z80 CPU). The readable bytes contain DMA status, byte-counter values, and port addresses since the last DMA reset. The registers are always read in a fixed sequence beginning with RR0 and ending with RR6. However, the register read in this sequence is determined by programming the Read Mask in WR6. The sequence of reading is initialized by writing an Initiate Read Sequence or Set Read Status command to WR6. After a Reset DMA, the sequence must be initialized with the Initiate Read Sequence command or a Read Status command. The sequence of reading all registers that are not excluded by the Read Mask register must be completed before a new Initiate Read Sequence or Read Status command.

**Writing.** Control or command bytes are written into one or more of the Write Register groups (WR0-WR6) by first writing to the base register byte in that group. All groups have base registers and most groups have additional associated registers. The associated registers in a group are sequentially accessed by first writing a byte to the base register containing register-group identification and pointer bits (1's) to one or more of that base register's associated registers.

This is illustrated in Figure 8b. In this figure, the sequence in which associated registers within a group can be written to is shown by the vertical position of the associated registers. For example, if a byte written to the DMA contains the bits that identify WR0 (bits D0, D1 and D7), and also contains 1's in the bit positions that point to the associated "Port A Starting Address (low byte)" and "Port A Starting Address (high byte)," then the next two bytes written to the DMA will be stored in that order in these two registers.

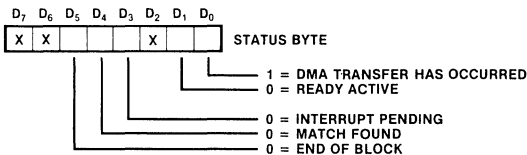
**Fixed-Address Programming.** A special circumstance arises when programming a destination port to have a fixed address. The load command in WR6 only loads a fixed address to a port selected as the source, not to a port selected as the destination. Therefore, a fixed destination address must be loaded by temporarily declaring it a fixed-source address and subsequently declaring the true source as such, thereby implicitly making the other a destination.

The following example illustrates the steps in this procedure, assuming that transfers are to occur from a variable-address source (Port A) to a fixed-address destination (Port B):

1. Temporarily declare Port B as source in WR0.
2. Load Port B address with LOAD command to WR6.
3. Declare Port A as a source in WR0.
4. Load Port A address with LOAD command to WR6.
5. Enable DMA in WR6.

Figure 9 illustrates a program to transfer data from memory (Port A) to a peripheral device (Port B). In this example, the Port A memory starting address is 1050<sub>H</sub> and the Port B peripheral fixed address is 05<sub>H</sub>. Note that the data flow is 1001<sub>H</sub> bytes—one more than specified by the block length. The table of DMA commands may be stored in consecutive memory locations and transferred to the DMA with an output instruction such as the Z80 CPU's OTIR instruction.

### READ REGISTER 0



### READ REGISTER 1



### READ REGISTER 2



### READ REGISTER 3



### READ REGISTER 4



### READ REGISTER 5



### READ REGISTER 6



Figure 8a. Read Registers

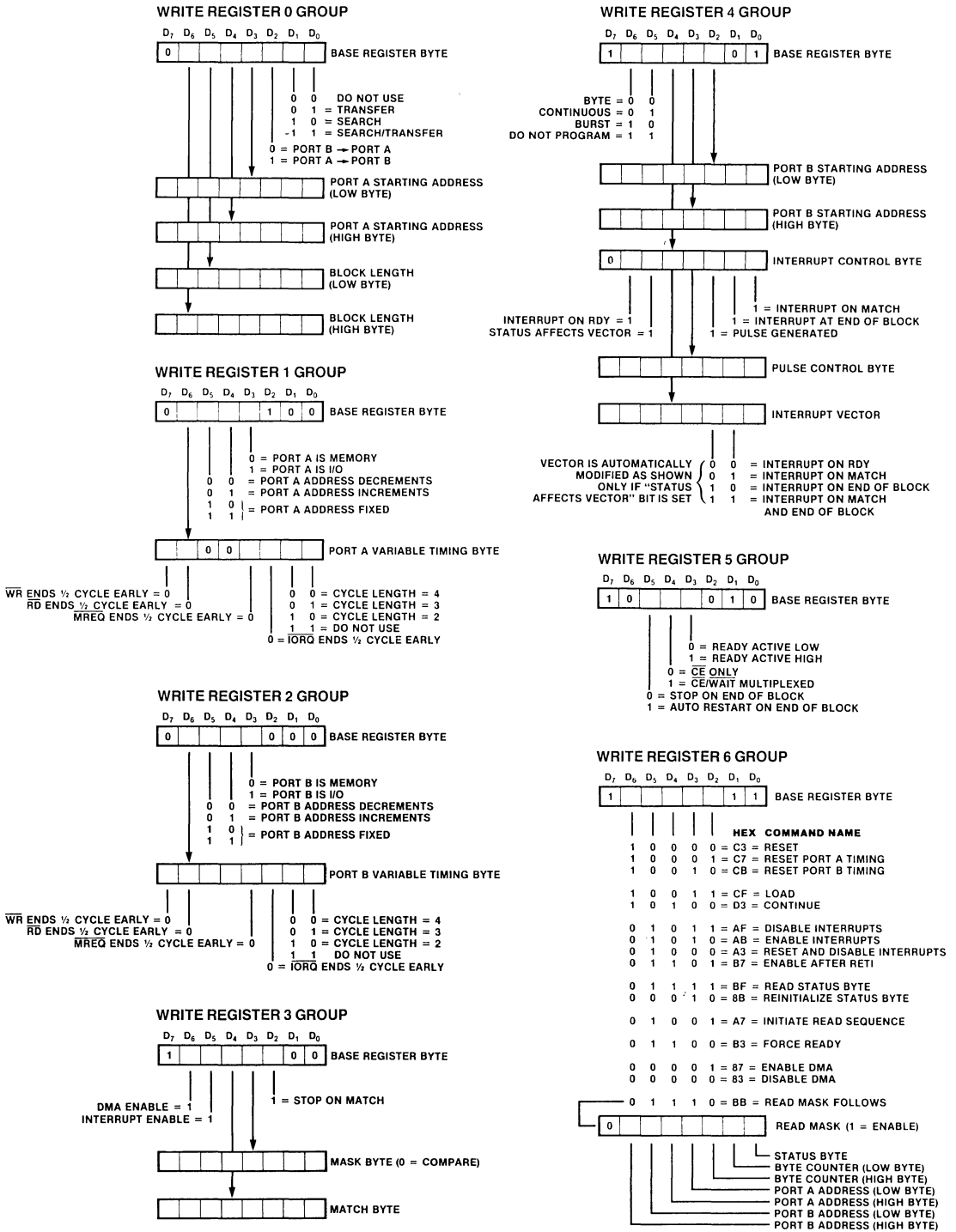


Figure 8b. Write Registers

Comments	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	HEX
WR0 sets DMA to receive block length, Port A starting address and temporarily sets Port B as source.	0	1 Block Length Upper Follows	1 Block Length Lower Follows	1 Port A Upper Address Follows	1 Port A Lower Address Follows	0 B → A Temporary for Loading B Address*	0	1	79
Port A address (lower)	0	1	0	1	0	0	0	0	50
Port A address (upper)	0	0	0	1	0	0	0	0	10
Block length (lower)	0	0	0	0	0	0	0	0	00
Block length (upper)	0	0	0	1	0	0	0	0	10
WR1 defines Port A as memory with fixed incrementing address.	0	0 No Timing Follows	0 Address Changes	1 Address Increments	0 Port is Memory	1	0	0	14
WR2 defines Port B as peripheral with fixed address.	0	0 No Timing Follows	1 Fixed Address	0	1 Port is I/O	0	0	0	28
WR4 sets mode to Burst, sets DMA to expect Port B address.	1	1	0 Burst Mode	0 No Interrupt Control Byte Follows	0 No Upper Address	1 Port B Lower Address Follows	0	1	C5
Port B address (lower)	0	0	0	0	0	1	0	1	05
WR5 sets Ready active High.	1	0	0 No Auto Restart	0 No Wait States	1 RDY Active High	0	1	0	8A
WR6 loads Port B address and resets block counter.*	1	1	0	0	1	1	1	1	CF
WR0 sets Port A as source.*	0	0	0 No Address or Block Length Bytes	0	0	1 A → B	0	1	05
WR6 loads Port A address and resets block counter.	1	1	0	0	1	1	1	1	CF
WR6 enables DMA to start operation.	1	0	0	0	0	1	1	1	87

NOTE. The actual number of bytes transferred is one more than specified by the block length.

\*These entries are necessary only in the case of a fixed destination address.

Figure 9. Sample DMA Program

## INACTIVE STATE TIMING

(DMA as CPU Peripheral)

In its disabled or inactive state, the DMA is addressed by the CPU as an I/O peripheral for write and read (control and status) operations. Write timing is illustrated in Figure 10.

Reading of the DMA's status byte, byte counter, or port address counters is illustrated in Figure 11. These

operations require less than three T-cycles. The  $\overline{CE}$ ,  $\overline{IORQ}$ , and  $\overline{RD}$  lines are made active over two rising edges of CLK, and data appears on the bus approximately one T-cycle after they become active.

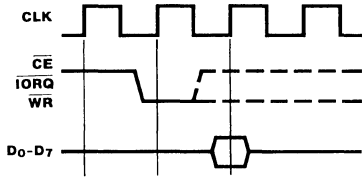


Figure 10. CPU-to-DMA Write Cycle

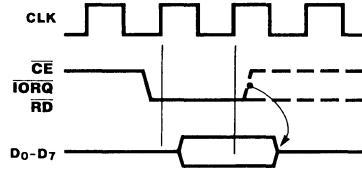


Figure 11. CPU-to-DMA Read Cycle

## ACTIVE STATE TIMING

(DMA as Bus Controller)

**Default Read and Write Cycles.** By default, and after reset, the DMA's timing of read and write operations is exactly the same as the Z80 CPU's timing of read and write cycles for memory and I/O peripherals, with one exception: during a read cycle, data is latched on the falling edge of  $T_3$  and held on the data bus across the boundary between read and write cycles, through the end of the following write cycle.

Figure 12 illustrates the timing for memory-to-I/O port transfers and Figure 13 illustrates I/O-to-memory transfers.

Memory-to-memory and I/O-to-I/O transfer timings are simply permutations of these diagrams.

The default timing uses three T-cycles for memory transactions and four T-cycles for I/O transactions, which include one automatically inserted wait cycle ( $T_{WA}$ ) between  $T_2$  and  $T_3$ . If the  $\overline{CE}/\overline{WAIT}$  line is programmed to act as a  $\overline{WAIT}$  line during the DMA's active state, it is sampled on the falling edge of  $T_2$  for memory transactions and the falling edge of  $T_{WA}$  for I/O transactions. If  $\overline{CE}/\overline{WAIT}$  is Low during this time, another T-cycle is added, during which the

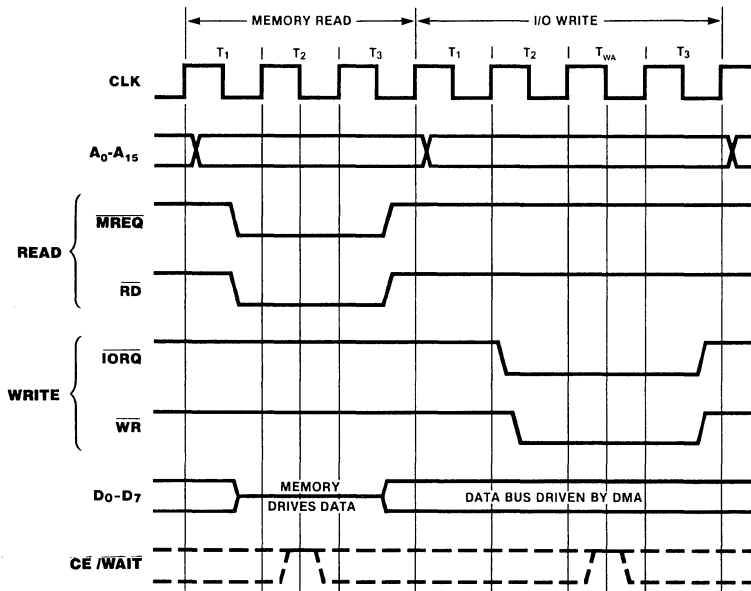


Figure 12. Memory-to-I/O Transfer

$\overline{CE}/\overline{WAIT}$  line will again be sampled. The duration of transactions can thus be indefinitely extended.

**Variable Cycle and Edge Timing.** The Z80 DMA's default operation-cycle length for the source (read) port and destination (write) port can be independently programmed. This variable-cycle feature allows read or write cycles consisting of two, three, or four T-cycles (more if Wait cycles are inserted), thereby increasing or decreasing the speed of all signals generated by the DMA. In addition, the trailing edges of the  $\overline{IORQ}$ ,  $\overline{MREQ}$ ,  $\overline{RD}$ , and  $\overline{WR}$  signals can be independently terminated one-half cycle early. Figure 14 illustrates this.

In the variable-cycle mode, unlike default timing,  $\overline{IORQ}$  comes active one-half cycle before  $\overline{MREQ}$ ,  $\overline{RD}$ , and  $\overline{WR}$ .  $\overline{CE}/\overline{WAIT}$  can be used to extend only the 3 or 4 T-cycle variable memory cycles and only the 4-cycle variable I/O cycle. The  $\overline{CE}/\overline{WAIT}$  line is sampled at the falling edge of  $T_2$  for 3- or 4-cycle memory cycles, and at the falling edge of  $T_3$  for 4-cycle I/O cycles.

During transfers, data is latched on the clock edge causing the rising edge of  $\overline{RD}$  and held until the end of the write cycle.

**Bus Requests.** Figure 15 illustrates the bus request and acceptance timing. The RDY line, which may be programmed active High or Low, is sampled on every rising edge of CLK. If it is found to be active and if the bus is not in use by any other device, the following rising edge of CLK drives  $\overline{BUSREQ}$  Low. After receiving  $\overline{BUSREQ}$ , the CPU acknowledges on the  $\overline{BAI}$  input either directly or through a multiple-DMA daisy chain. When a Low is detected on  $\overline{BAI}$  for two consecutive rising edges of CLK, the DMA will begin transferring data on the next rising edge of CLK.

**Bus Release Byte-at-a-Time.** In Byte-at-a-Time mode,  $\overline{BUSREQ}$  is brought High on the rising edge of CLK prior to the end of each read cycle (search-only) or write cycle (transfer and transfer/search) as illustrated in Figure 16. This is done regardless of the state of RDY. There is no possibility of confusion when a Z80 CPU is used since the CPU cannot

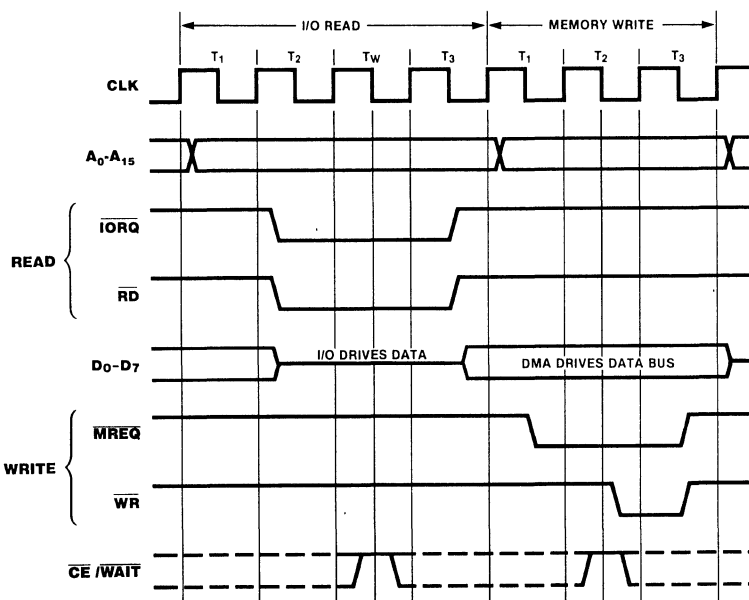


Figure 13. I/O-to-Memory Transfer

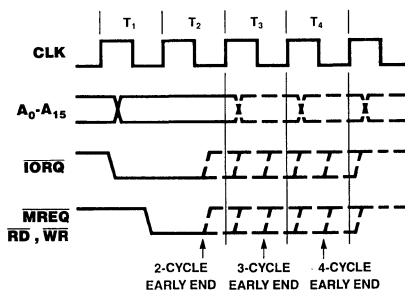


Figure 14. Variable-Cycle and Edge Timing

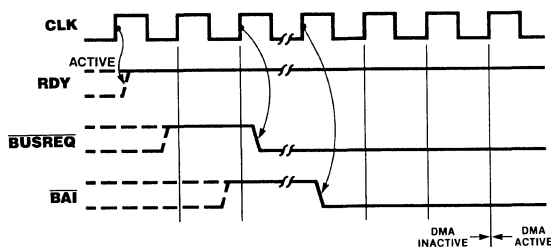


Figure 15. Bus Request and Acceptance

begin an operation until the following T-cycle. Most other CPUs are not bothered by this either, although note should be taken of it. The next bus request for the next byte will come after both  $\overline{\text{BUSREQ}}$  and  $\overline{\text{BAI}}$  have returned High.

**Bus Release at End of Block.** In Burst and Continuous modes, an end of block causes  $\overline{\text{BUSREQ}}$  to go High, usually on the same rising edge of CLK in which the DMA completes the transfer of the data block (Figure 17). The last byte in the block is transferred even if RDY goes inactive before completion of the last byte transfer.

**Bus Release on Not Ready.** In Burst mode, when RDY goes inactive it causes  $\overline{\text{BUSREQ}}$  to go High on the next rising edge of CLK after the completion of its current byte operation (Figure 18). The action on  $\overline{\text{BUSREQ}}$  is thus somewhat delayed from action on the RDY line. The DMA always completes its current byte operation in an orderly fashion before releasing the bus.

By contrast,  $\overline{\text{BUSREQ}}$  is not released in Continuous mode when RDY goes inactive. Instead, the DMA idles after completing the current byte operation, awaiting an active RDY again.

**Bus Release on Match.** If the DMA is programmed to stop on match in Burst or Continuous modes, a match causes

$\overline{\text{BUSREQ}}$  to go inactive on the next DMA operation, i.e., at the end of the next read in a search or at the end of the following write in a transfer (Figure 19). Due to the pipelining scheme, matches are determined while the next DMA read or write is being performed.

The RDY line can go inactive after the matching operation begins without affecting this bus-release timing.

**Interrupts.** Timings for interrupt acknowledge and return from interrupt are the same as for the other Z80 peripherals.

Interrupt on RDY (interrupt before requesting bus) does not directly affect the  $\overline{\text{BUSREQ}}$  line. Instead, the interrupt service routine must handle this by issuing the following commands to WR6:

1. Enable after Return From Interrupt (RETI) Command—Hex B7
2. Enable DMA—Hex 87
3. An RETI instruction that resets the Interrupt Under Service latch in the Z80 DMA.

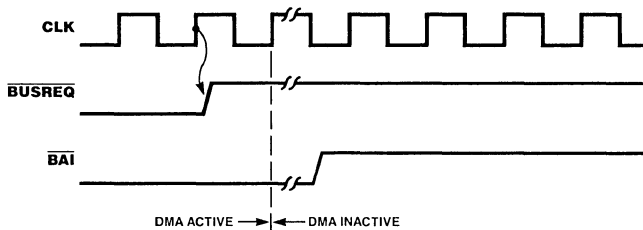


Figure 16. Bus Release (Byte-at-a-Time Mode)

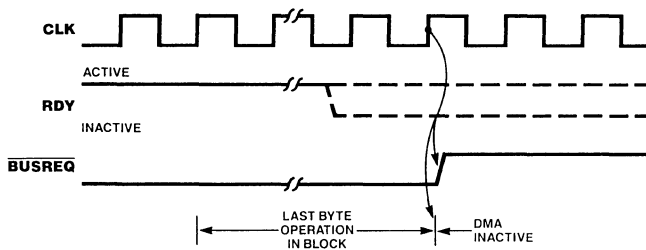


Figure 17. Bus Release at End of Block (Burst and Continuous Modes)



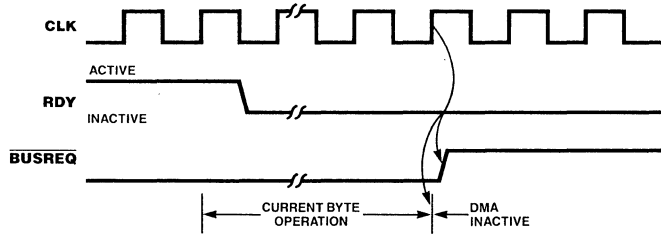


Figure 18. Bus Release When Not Ready  
(Burst Mode)

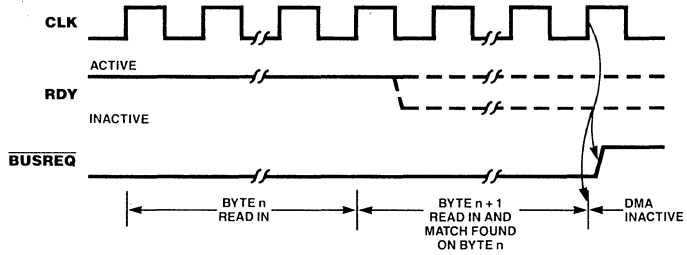


Figure 19. Bus Release on Match  
(Burst and Continuous Modes)

## ABSOLUTE MAXIMUM RATINGS

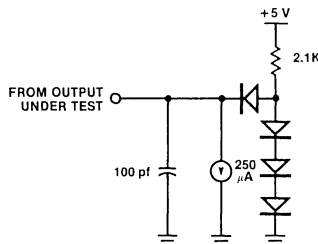
Voltages on  $V_{CC}$  with respect to  $V_{SS}$  . . . . .  $-0.3V$  to  $+7.0V$   
 Voltages on all inputs with respect  
 to  $V_{SS}$  . . . . .  $-0.3V$  to  $V_{CC} + 0.3V$   
 Storage Temperature . . . . .  $-65^{\circ}C$  to  $+150^{\circ}C$

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature range is:

- **S =  $0^{\circ}C$  to  $+70^{\circ}C$ ,  $V_{CC}$  Range**  
 NMOS:  $+4.75V \leq V_{CC} \leq +5.25V$   
 CMOS:  $+4.50V \leq V_{CC} \leq +5.50V$
- **E =  $-40^{\circ}C$  to  $100^{\circ}C$ ,  $+4.50V \leq V_{CC} \leq +5.50V$**



## DC CHARACTERISTICS (Z84C10 / CMOS Z80 DMA)

Symbol	Parameter	Min	Max	Typ	Unit	Test Condition
$V_{ILC}$	Clock Input Low Voltage	-0.3	+0.45		V	
$V_{IHC}$	Clock Input High Voltage	$V_{CC} - 0.6$	$V_{CC} + 0.3$		V	
$V_{IL}$	Input Low Voltage	-0.3	+0.8		V	
$V_{IH}$	Input High Voltage	+2.2	$V_{CC}$		V	
$V_{OL}$	Output Low Voltage		+0.4		V	$I_{OL} = 2.0$ mA
$V_{OH1}$	Output High Voltage	+2.4			V	$I_{OH} = -1.6$ mA
$V_{OH2}$	Output High Voltage	$V_{CC} - 0.8$			V	$I_{OH} = -250$ $\mu$ A
$I_{LI}$	Input Leakage Current		$\pm 10$		$\mu$ A	$V_{IN} = 0.4$ to $V_{CC}$
$I_{LO}$	3-State Output Leakage Current in Float		$\pm 10$		$\mu$ A	$V_{OUT} = 0.4$ to $V_{CC}$
$ICC_1$	Power Supply Current		<b>25/35</b>		mA	$V_{CC} = 5V$ <b>CLK = 6/8MHz</b> $V_{IHC} = V_{IH} = V_{CC} - 0.2V$ $V_{ILC} = 0.2V$
$ICC_2$	Standby Supply Current		10	0.5	$\mu$ A	$V_{CC} = 5V$ CLK = (0) $V_{IHC} = V_{IH} = V_{CC} - 0.2V$ $V_{ILC} = V_{IL} = 0.2V$

Over specified temperature and voltage range.

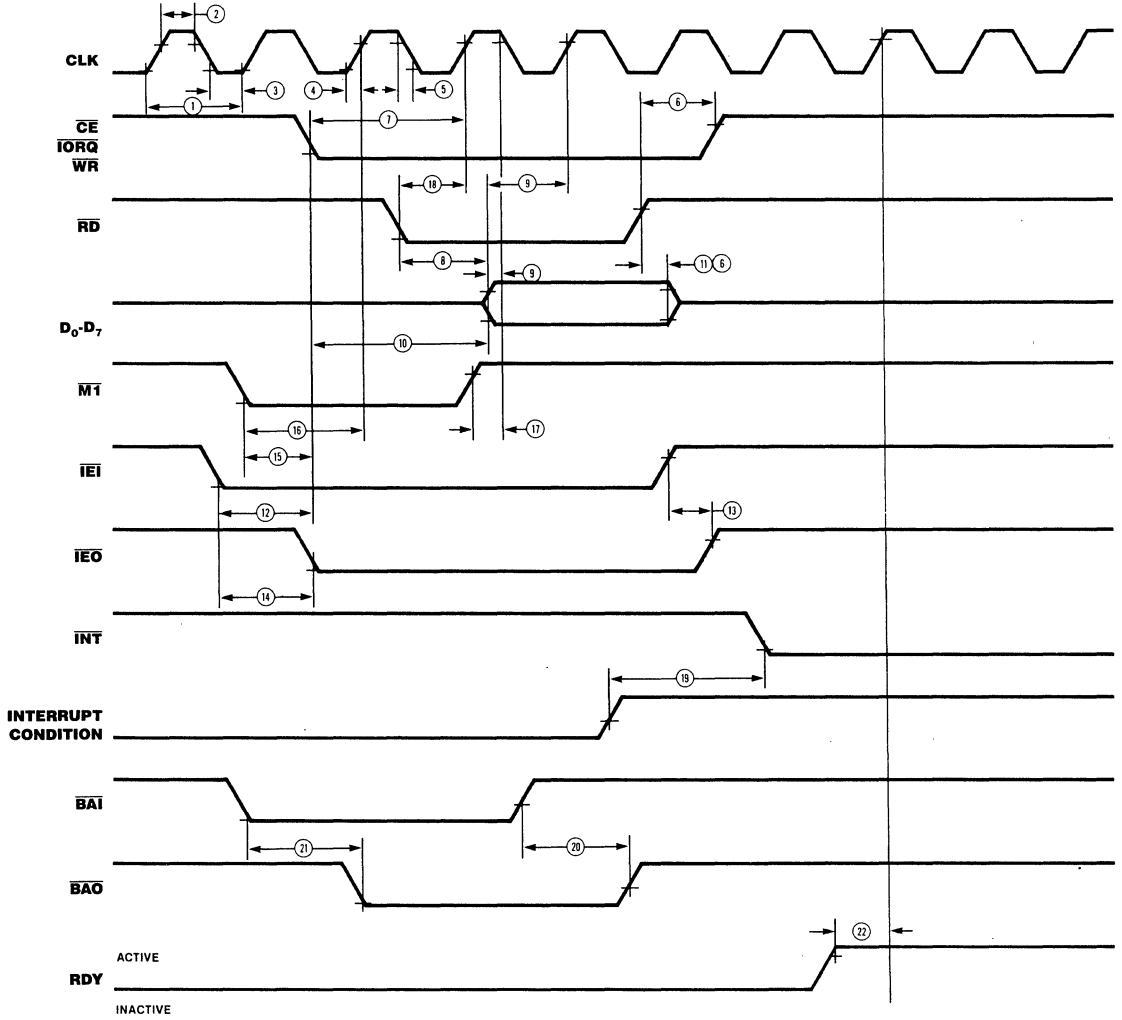
## CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C	Clock Capacitance		5	pf
$C_{IN}$	Input Capacitance		5	pf
$C_{OUT}$	Output Capacitance		10	pf

NOTES: Over specified temperature range; f = MHz.  
 Unmeasured pins returned to ground.

# AC CHARACTERISTICS (Z84C10 / CMOS Z80 DMA)

(Inactive State)



NOTE: Signals in this diagram bear no relation to one another unless specifically noted as a numbered item.

## AC CHARACTERISTICS (Z8410 / NMOS Z80 DMA)

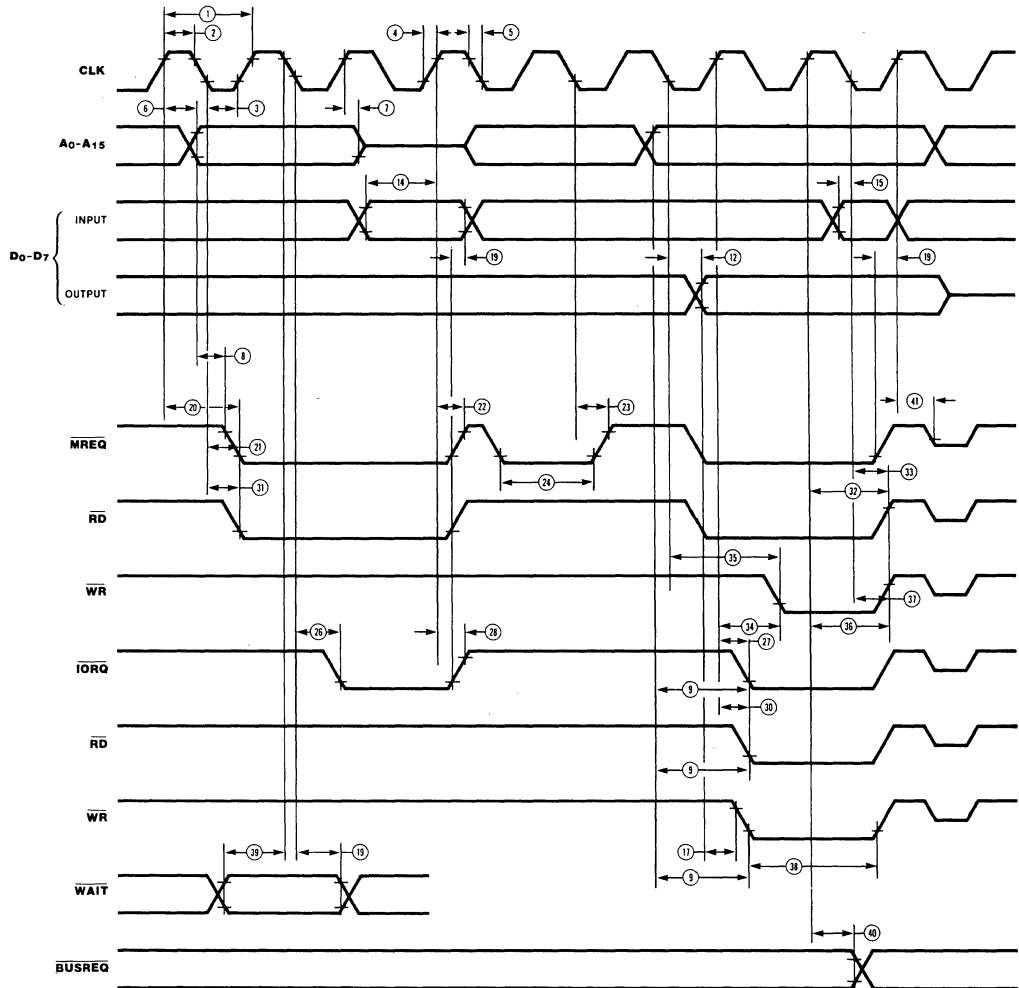
(Inactive State)

Number	Symbol	Parameter	Z0841004		Unit
			Min	Max	
1	TcC	Clock Cycle Time	250	4000	ns
2	TwCh	Clock Width (High)	110	2000	ns
3	TwCl	Clock Width (Low)	110	2000	ns
4	TrC	Clock Rise Time		30	ns
5	TfC	Clock Fall Time		30	ns
6	Th	Hold Time for Any Specified Setup Time	0		ns
7	TsC(Cr)	$\overline{IORQ}$ , $\overline{WR}$ , $\overline{CE}$ ↓ to Clock ↑ Setup	145		ns
8	TdDO(RDf)	$\overline{RD}$ ↓ to Data Output Delay		380	ns
9	TsDI(Cr)	Data In to Clock ↑ Setup ( $\overline{WR}$ or $\overline{M1}$ )	50		ns
10	TdDO(IOf)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)		160	ns
11	TdRD(rDz)	$\overline{RD}$ ↑ to Data Float Delay (output buffer disable)		110	ns
12	TsIE(IORQf)	IEI to $\overline{IORQ}$ ↓ Setup (INTA Cycle)	140		ns
13	TdIEOr(IEIr)	IEI ↑ to IEO ↑ Delay		160	ns
14	TdIEOf(IEIf)	IEI ↓ to IEO ↓ Delay		130	ns
15	TdM1f(IEOf)	$\overline{M1}$ ↓ to IEO ↓ Delay (interrupt just prior to $\overline{M1}$ ↓)		190	ns
16	TsM1f(Cr)	$\overline{M1}$ ↓ to Clock ↑ Setup	90		ns
17	TsM1r(Cf)	$\overline{M1}$ ↑ to Clock Setup	-10		ns
18	TsRDf(Cr)	$\overline{RD}$ ↓ to Clock ↑ Setup ( $\overline{M1}$ Cycle)	115		ns
19	TdI(INTf)	Interrupt Cause to $\overline{INT}$ ↓ Delay ( $\overline{INT}$ generated only when DMA is inactive)		500	ns
20	TdBAlr(BAO r)	$\overline{BAI}$ ↑ to $\overline{BAO}$ ↑ Delay		150	ns
21	TdBAlf(BAO f)	$\overline{BAI}$ ↓ to $\overline{BAO}$ ↓ Delay		150	ns
22	TsRDY(Cr)	RDY Active to Clock ↑ Setup	100		ns

NOTE: Negative minimum setup values mean that the first-mentioned event can come after the second-mentioned event.

# AC CHARACTERISTICS (Z84C10 / CMOS Z80 DMA)

(Active State)



NOTE: Signals in this diagram bear no relation to one another unless specifically noted as a numbered item.

Number	Symbol	Parameter	Z84C1006* ‡ † Z84C1008* ‡ †			
			Min(ns)	Max(ns)		
1	T <sub>cC</sub>	Clock Cycle Time	162	DC	125	DC
2	T <sub>wCh</sub>	Clock Width (High)	65	DC	55	DC
3	T <sub>wCl</sub>	Clock Width (Low)	65	DC	55	DC
4	T <sub>rC</sub>	Clock Rise Time		20		10
5	T <sub>fC</sub>	Clock Fall Time		20		10

NOTES:

° For clock periods other than the minimums shown, calculate parameters using the following table.

‡ Calculated values above assumed T<sub>rC</sub> = T<sub>fC</sub> = 20ns (6 MHz version) or 10ns (8 MHz version).

† Data must be enabled onto data bus when RD is active.

\* Parameter is not illustrated in the AC Timing Diagrams.

## AC CHARACTERISTICS (Z84C10 / CMOS Z80 DMA)

(Active State)

Number	Symbol	Parameter	Z84C1006		Z84C1008	
			Min(ns)	Max(ns)	Min(ns)	Max(ns)
6	TdA	Address Output Delay		90		70
7	TdC(Az)	Clock ↑ to Address Float Delay		80		70
8	TsA(MREQ)	Address to $\overline{\text{MREQ}}$ ↓ Setup (Memory Cycle)	35‡		35‡	
9	TsA(IRW)	Address Stable to $\overline{\text{IORQ}}$ , $\overline{\text{RD}}$ , $\overline{\text{WR}}$ ↓ Setup (I/O Cycle)	110‡		70‡	
*10	TdRW(A)	$\overline{\text{RD}}$ , $\overline{\text{WR}}$ ↑ to Addr. Stable Delay	35‡		15‡	
*11	TdRW(Az)	$\overline{\text{RD}}$ , $\overline{\text{WR}}$ ↑ to Addr. Float	60‡		45‡	
12	TdC(DO)	Clock ↓ to Data Out Delay		130		110
*13	TdCr(Dz)	Clock ↑ to Data Float Delay (Write Cycle)		70		65
14	TsDI(Cr)	Data In to Clock ↑ Setup (Read cycle when rising edge ends read)	30		25	
15	TsDI(Cf)	Data In to Clock ↓ Setup (Read cycle when falling edge ends read)	40		30	
*16	TsDO(WfM)	Data Out to $\overline{\text{WR}}$ ↓ Setup (Memory Cycle)	45*1		40*1	
17	TsDO(WfI)	Data Out to $\overline{\text{WR}}$ ↓ Setup (I/O cycle)	55		40	
*18	TdWr(DO)	$\overline{\text{WR}}$ ↑ to Data Out Delay	30‡		10‡	
19	Th	Hold Time for Any Specified Setup Time	0		0	
20	TdCr(Mf)	Clock ↑ to $\overline{\text{MREQ}}$ ↓ Delay		70		60
21	TdC(Mf)	Clock ↓ to $\overline{\text{MREQ}}$ ↓ Delay		70		60
22	TdCr(Mr)	Clock ↑ to $\overline{\text{MREQ}}$ ↑ Delay		70		60
23	TdC(Mr)	Clock ↓ to $\overline{\text{MREQ}}$ ↑ Delay		70		60
24	TwM1	$\overline{\text{MREQ}}$ Low Pulse Width	135‡		95‡	
*25	TwMh	$\overline{\text{MREQ}}$ High Pulse Width	65‡		45‡	
26	TdC(I <sub>f</sub> )	Clock ↓ to $\overline{\text{IORQ}}$ ↓ Delay		70		60
27	TdCr(I <sub>f</sub> )	Clock ↑ to $\overline{\text{IORQ}}$ ↓ Delay		65		55
28	TdCr(I <sub>r</sub> )	Clock ↑ to $\overline{\text{IORQ}}$ ↑ Delay		70		60
*29	TdC(I <sub>r</sub> )	Clock ↓ to $\overline{\text{IORQ}}$ ↑ Delay		70		60
30	TdCr(R <sub>f</sub> )	Clock ↑ to $\overline{\text{RD}}$ ↓ Delay		70		60
31	TdC(R <sub>f</sub> )	Clock ↓ to $\overline{\text{RD}}$ ↓ Delay		80		70
32	TdCr(R <sub>r</sub> )	Clock ↑ to $\overline{\text{RD}}$ ↑ Delay		70		60
33	TdC(R <sub>r</sub> )	Clock ↓ to $\overline{\text{RD}}$ ↑ Delay		70		60
34	TdCr(W <sub>f</sub> )	Clock ↑ to $\overline{\text{WR}}$ ↓ Delay		60		55
35	TdC(W <sub>f</sub> )	Clock ↓ to $\overline{\text{WR}}$ ↓ Delay		70		60
36	TdCr(W <sub>r</sub> )	Clock ↑ to $\overline{\text{WR}}$ ↑ Delay		70		60
37	TdC(W <sub>r</sub> )	Clock ↓ to $\overline{\text{WR}}$ ↑ Delay		70		60
38	TwWI	$\overline{\text{WR}}$ Low Pulse Width	135‡		95‡	
39	TsWA(Cf)	$\overline{\text{WAIT}}$ to Clock ↓ Setup	60		50	
40	TdCr(B)	Clock ↑ to $\overline{\text{BUSREQ}}$ Delay		90		80
41	TdCr(I <sub>z</sub> )	Clock ↑ to $\overline{\text{IORQ}}$ , $\overline{\text{MREQ}}$ , $\overline{\text{RD}}$ , $\overline{\text{WR}}$ Float Delay		70		70

### NOTES:

- ‡ All AC equations imply DMA default (standard) timing.
- \* Data must be enabled onto data bus when RD is active.
- Parameter is not illustrated in the AC Timing Diagrams.

1. From Figure 13, data is latched onto the bus by the rising edge of  $\overline{\text{RD}}$  signal. These values are the characterized minimum data out set-up time with respect to  $\overline{\text{WR}}$  ↓.

---

**FOOTNOTES TO AC CHARACTERISTICS**

---

Number	Symbol	General Parameter	Z84C1006	Z84C1008
8	TsA(MREQ)	TwCh - TfC	-35	-30
9	TsA(IRW)	TcC	-55	-55
10	TdRW(A)	TwCl - TrC	-50	-50
11	TdRW(Z)	TwCl - TrC	-25	-20
16	TsDO(WfM)	TcC	-140	-120
18	TdWr(DO)	TwCl - TrC	-55	-55
24	TwM1	TcC	-30	-30
25	TwMh	TwCh - TfC	-20	-20
38	TwWI	TcC	-30	-30

---

**DC CHARACTERISTICS (Z8410 NMOS Z80 DMA)** $V_{CC}=5.0V \pm 10\%$ , unless otherwise specified

Sym	Parameter	Min	Max	Unit	Test Condition
$V_{ILC}$	Clock Input Low Voltage	-0.3	0.45	V	
$V_{IHC}$	Clock Input High Voltage	$V_{CC}-.6$	5.5	V	
$V_{IL}$	Input Low Voltage	-0.3	0.8	V	
$V_{IH}$	Input High Voltage	2.0	5.5	V	
$V_{OL}$	Output Low Voltage		0.4	V	$I_{OL}=3.2\text{ mA}$ for /BUSREQ $I_{OL}=2.0\text{ mA}$ for all others
$V_{OH}$	Output High Voltage	2.4		V	$I_{OH}=250\ \mu\text{A}$
$I_{CC}$	Power Supply Current				
	Z-80 DMA		150	mA	
	Z-80A DMA		200	mA	
$I_{LI}$	Input Leakage Current		10	$\mu\text{A}$	$V_{IN}=0$ to $V_{CC}$
$I_{LO}$	3-State Output Leakage Current in Float		$\pm 10$	$\mu\text{A}$	$V_{OUT}=0.4V$ to $V_{CC}$
$I_{LD}$	Data Bus Leakage Current in Input Mode		$\pm 10$	$\mu\text{A}$	$\leq V_{IN} \leq V_{CC}$

**Note:** $V_{CC}=5V \pm 5\%$  unless otherwise specified, over specified temperature range.**CAPACITANCE**

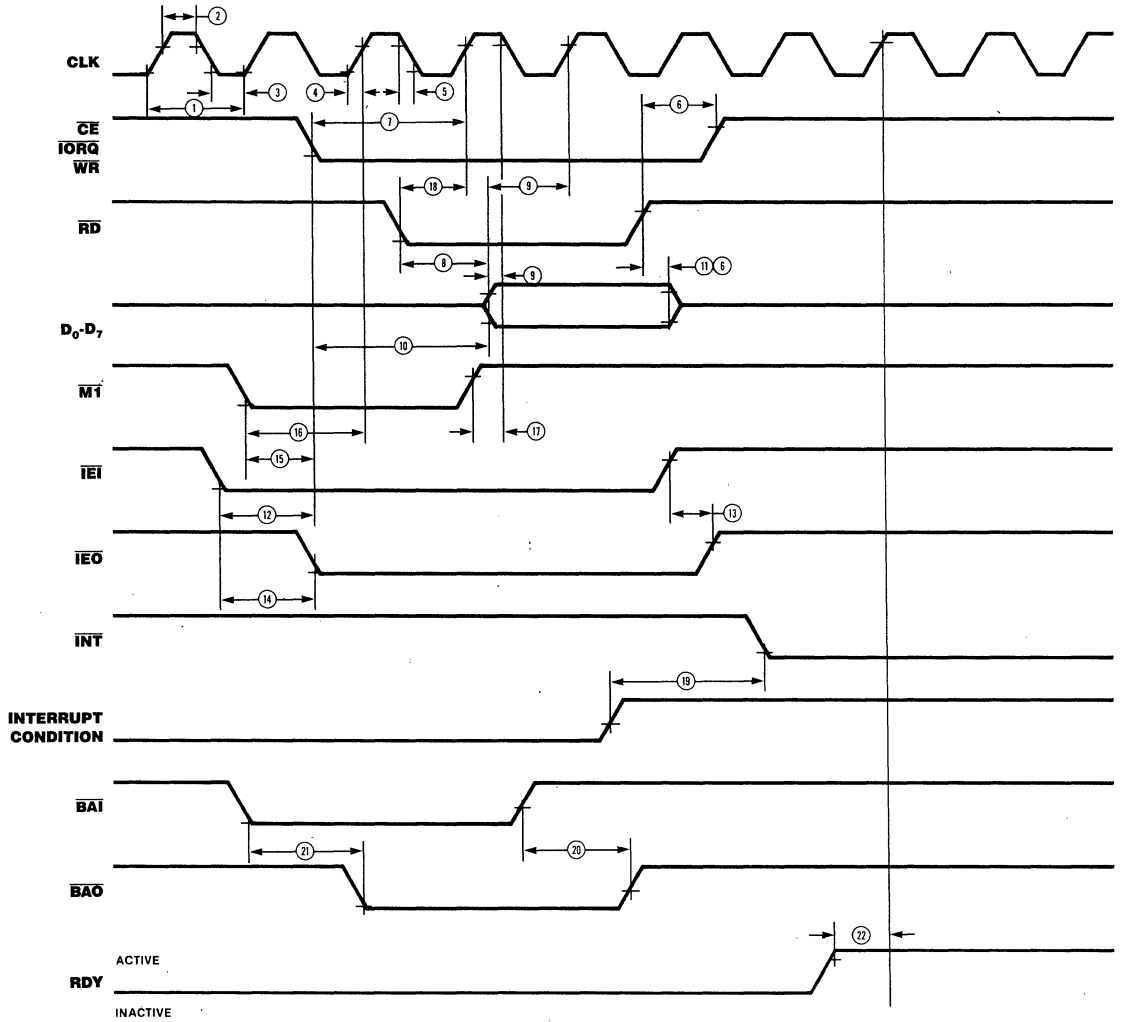
Symbol	Parameter	Min	Max	Unit	Test Condition
C	Clock Capacitance		35	pF	Unmeasured pins returned to ground.
$C_{IN}$	Input Capacitance		10	pF	
$C_{OUT}$	Output Capacitance		10	pF	

**Note:**Over specified temperature range;  $f = 1\text{ MHz}$



# AC CHARACTERISTICS (Z8410 / NMOS Z80 DMA)

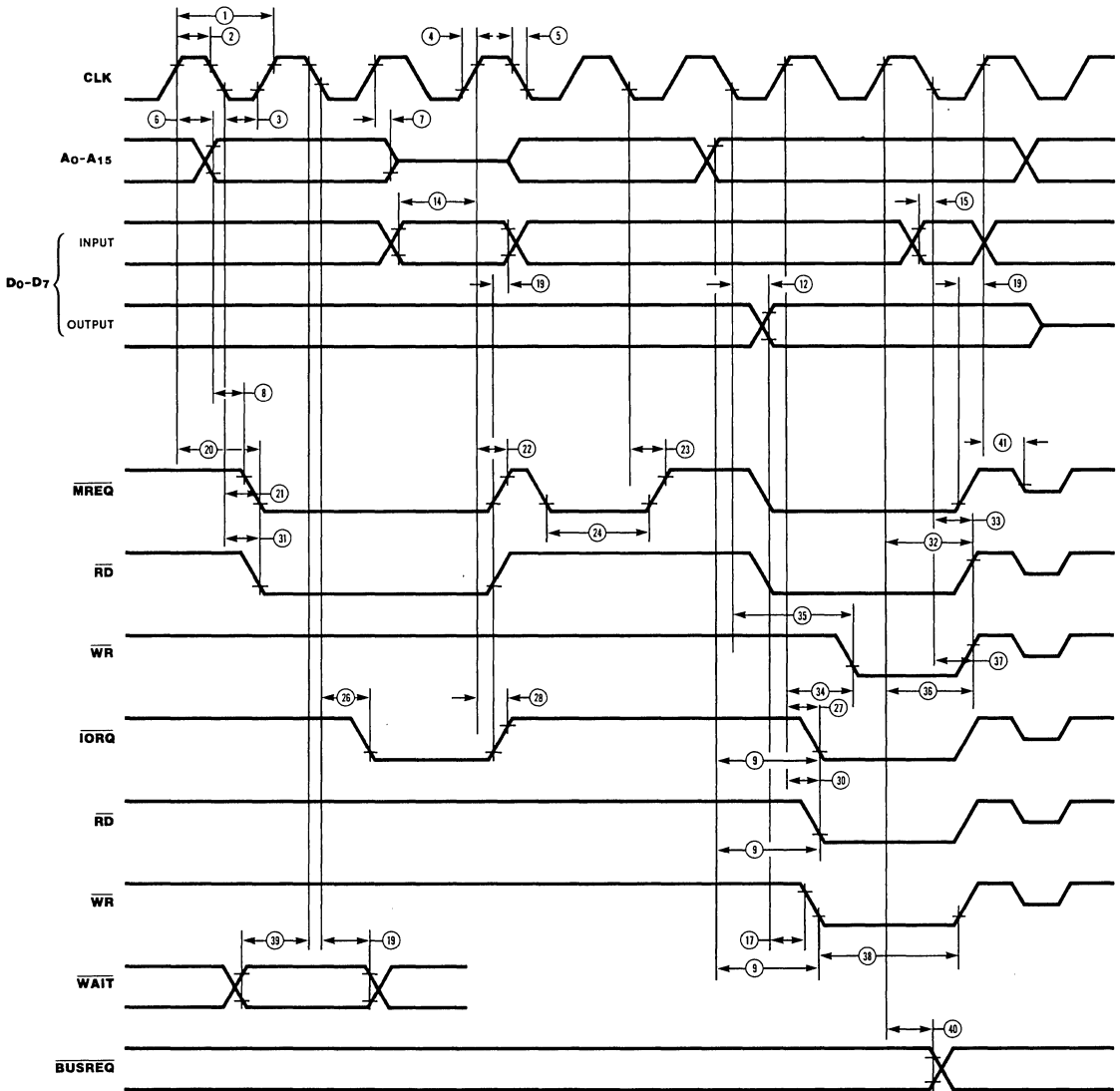
(Inactive State)



NOTE: Signals in this diagram bear no relation to one another unless specifically noted as a numbered item.

# AC CHARACTERISTICS (Z8410 / NMOS Z80 DMA)

(Active State)



NOTE: Signals in this diagram bear no relation to one another unless specifically noted as a numbered item.

			Z841004 <sup>°††</sup>	
Number	Symbol	Parameter	Min(ns)	Max(ns)
1	T <sub>c</sub>	Clock Cycle Time	250	
2	T <sub>wCh</sub>	Clock Width (High)	110	2000
3	T <sub>wCl</sub>	Clock Width (Low)	110	2000
4	T <sub>rC</sub>	Clock Rise Time		30
5	T <sub>fC</sub>	Clock Fall Time		30

**NOTES:**

<sup>°</sup> Numbers in parentheses are other parameter-numbers in this table; their values should be substituted in equations.

‡ All equations imply DMA default (standard) timing.

† Data must be enabled onto data bus when RD is active.

\* Parameter is not illustrated in the AC Timing Diagrams.

## AC CHARACTERISTICS (Z8410 / NMOS Z80 DMA)

(Active State)

Number	Symbol	Parameter	Z0841004 *††	
			Min(ns)	Max(ns)
6	TdA	Address Output Delay		110
7	TdC(Az)	Clock ↑ to Address Float Delay		90
8	TsA(MREQ)	Address to $\overline{\text{MREQ}}$ ↓ Setup (Memory Cycle)	(2) + (5) – 75	
9	TsA(IRW)	Address Stable to $\overline{\text{IORQ}}$ , $\overline{\text{RD}}$ , $\overline{\text{WR}}$ ↓ Setup (I/O Cycle)	(1) – 70	
*10	TdRW(A)	$\overline{\text{RD}}$ , $\overline{\text{WR}}$ ↑ to Addr. Stable Delay	(3) + (4) – 50	
*11	TdRW(Az)	$\overline{\text{RD}}$ , $\overline{\text{WR}}$ ↑ to Addr. Float	(3) + (4) – 45	
12	TdC(DO)	Clock ↓ to Data Out Delay		150
*13	TdCr(Dz)	Clock ↑ to Data Float Delay (Write Cycle)		90
14	TsDI(Cr)	Data In to Clock ↑ Setup (Read cycle when rising edge ends read)	35	
15	TsDI(Cf)	Data In to Clock ↓ Setup (Read cycle when falling edge ends read)	50	
*16	TsDO(WfM)	Data Out to $\overline{\text{WR}}$ ↓ Setup (Memory Cycle)	(1) – 170	
17	TsDO(WfI)	Data Out to $\overline{\text{WR}}$ ↓ Setup (I/O cycle)	100	
*18	TdWr(DO)	$\overline{\text{WR}}$ ↑ to Data Out Delay	(3) + (4) – 70	
19	Th	Hold Time for Any Specified Setup Time	0	
20	TdCr(Mf)	Clock ↑ to $\overline{\text{MREQ}}$ ↓ Delay		85
21	TdCf(Mf)	Clock ↓ to $\overline{\text{MREQ}}$ ↓ Delay		85
22	TdCr(Mr)	Clock ↑ to $\overline{\text{MREQ}}$ ↑ Delay		85
23	TdCf(Mr)	Clock ↓ to $\overline{\text{MREQ}}$ ↑ Delay		85
24	TwM1	$\overline{\text{MREQ}}$ Low Pulse Width	(1) – 30	
*25	TwMh	$\overline{\text{MREQ}}$ High Pulse Width	(2) + (5) – 20	
26	TdCf(Ir)	Clock ↓ to $\overline{\text{IORQ}}$ ↓ Delay		85
27	TdCr(Ir)	Clock ↑ to $\overline{\text{IORQ}}$ ↓ Delay		75
28	TdCr(Ir)	Clock ↑ to $\overline{\text{IORQ}}$ ↑ Delay		85
*29	TdCf(Ir)	Clock ↓ to $\overline{\text{IORQ}}$ ↑ Delay		85
30	TdCr(Rf)	Clock ↑ to $\overline{\text{RD}}$ ↓ Delay		85
31	TdCf(Rf)	Clock ↓ to $\overline{\text{RD}}$ ↓ Delay		95
32	TdCr(Rr)	Clock ↑ to $\overline{\text{RD}}$ ↑ Delay		85
33	TdCf(Rr)	Clock ↓ to $\overline{\text{RD}}$ ↑ Delay		85
34	TdCr(Wf)	Clock ↑ to $\overline{\text{WR}}$ ↓ Delay		65
35	TdCf(Wf)	Clock ↓ to $\overline{\text{WR}}$ ↓ Delay		80
36	TdCr(Wr)	Clock ↑ to $\overline{\text{WR}}$ ↑ Delay		80
37	TdCf(Wr)	Clock ↓ to $\overline{\text{WR}}$ ↑ Delay		80
38	TwWI	$\overline{\text{WR}}$ Low Pulse Width	(1) – 30	
39	TsWA(Cf)	$\overline{\text{WAIT}}$ to Clock ↓ Setup	70	
40	TdCr(B)	Clock ↑ to $\overline{\text{BUSREQ}}$ Delay		100
41	TdCr(Iz)	Clock ↑ to $\overline{\text{IORQ}}$ , $\overline{\text{MREQ}}$ , $\overline{\text{RD}}$ , $\overline{\text{WR}}$ Float Delay		80

NOTES:

‡ All AC equations imply DMA default (standard) timing.

\* Numbers in parentheses are other parameter - numbers in this table; their values should be substituted in equations.

## AC CHARACTERISTICS (Z84C10 / CMOS Z80 DMA)

(Inactive State)

Number	Symbol	Parameter	Z84C1006		Z84C1008		Unit
			Min	Max	Min	Max	
1	TcC	Clock Cycle Time	162	DC	125	DC	
2	TwCh	Clock Width (High)	65	DC	55	DC	
3	TwCl	Clock Width (Low)	65	DC	55	DC	
4	TrC	Clock Rise Time		20		10	
5	TfC	Clock Fall Time		20		10	
6	Th	Hold Time for Any Specified Setup Time	0		0		ns
7	TsC(Cr)	$\overline{\text{IORQ}}$ , $\overline{\text{WR}}$ , $\overline{\text{CE}} \downarrow$ to Clock $\uparrow$ Setup	60		45		ns
8	TdDO(RDf)	$\overline{\text{RD}} \downarrow$ to Data Output Delay		300		220	ns
9	TsDI(Cr)	Data In to Clock $\uparrow$ Setup ( $\overline{\text{WR}}$ or $\overline{\text{MT}}$ )	30		20		ns
10	TdDO(IOf)	$\overline{\text{IORQ}} \downarrow$ to Data Out Delay (INTA Cycle)		110		85	ns
11	TdRDr(Dz)	$\overline{\text{RD}} \uparrow$ to Data Float Delay (output buffer disable)		70		50	ns
12	TsIEI(IORQf)	IEI $\uparrow$ to $\overline{\text{IORQ}} \downarrow$ Setup (INTA Cycle)	100		80		ns
13	TdIEOr(IEIr)	IEI $\uparrow$ to IEO $\uparrow$ Delay		100		70	ns
14	TdIEOf(IEIf)	IEI $\downarrow$ to IEO $\downarrow$ Delay		100		70	ns
15	TdM1f(IEOf)	$\overline{\text{MT}} \downarrow$ to IEO $\downarrow$ Delay (interrupt just prior to $\overline{\text{MT}} \downarrow$ )		100		80	ns
16	TsM1f(Cr)	$\overline{\text{MT}} \downarrow$ to Clock $\uparrow$ Setup	70		45		ns
17	TsM1r(Cf)	$\overline{\text{MT}} \uparrow$ to Clock Setup	-15		-15		ns
18	TsRDf(Cr)	$\overline{\text{RD}} \downarrow$ to Clock $\uparrow$ Setup ( $\overline{\text{MT}}$ Cycle)	60		45		ns
19	TdI(INTf)	Interrupt Cause to $\overline{\text{INT}} \downarrow$ Delay ( $\overline{\text{INT}}$ generated only when DMA is inactive)		450		400	ns
20	TdBAlr(BAOr)	$\overline{\text{BAI}} \uparrow$ to $\overline{\text{BAO}} \downarrow$ Delay		100		70	ns
21	TdBAlf(BAOf)	$\overline{\text{BAI}} \downarrow$ to $\overline{\text{BAO}} \downarrow$ Delay		100		70	ns
22	TsRDY(Cr)	RDY Active to Clock $\uparrow$ Setup	50		50		ns

NOTE: Negative minimum setup values mean that the first-mentioned event can come after the second-mentioned event.

\* **M1 must be active for a minimum of two clock cycles to reset the DMA (This feature is only with C-MOS Z80 DMA).**

---

## Z8420/Z84C20 NMOS/CMOS Z80<sup>®</sup> PIO Parallel Input/Output

### FEATURES

- Provides a direct interface between Z80 microcomputer systems and peripheral devices.
- Two ports with interrupt-driven handshake for fast response.
- Four programmable operating modes: Output, Input, Bidirectional (Port A only), and Bit Control
- Programmable interrupts on peripheral status conditions. (1.5 mV @ 1.5V)
- **NMOS version for cost sensitive performance solutions.**
- **CMOS version for the designs requiring high speed and low power consumption**
- **NMOS Z0842004 - 4 MHz, Z0842006 - 6.17 MHz.**
- **CMOS Z84C2006 - DC to 6.17 MHz, Z84C2008 - DC to 8 MHz**
- Standard Z80 Family bus-request and prioritized interrupt-request daisy chains implemented without external logic.
- The eight Port B outputs can drive Darlington transistors (1.5 mA at 1.5V).
- **6 MHz version supports 6.144 MHz CPU clock operation.**

### GENERAL DESCRIPTION

The Z80 PIO Parallel I/O Circuit (hereinafter referred to as the Z80 PIO or PIO) is a programmable, dual-port device that provides a TTL-compatible interface between peripheral devices and the Z80 CPU (Figures 1 and 2). Note the QFP package is only available in CMOS version. The CPU configures the Z80 PIO to interface with a wide range of

peripheral devices that are compatible with the Z80 PIO include most keyboards, paper tape readers and punches, printers, and PROM programmers.

One characteristic of the Z80 peripheral controllers that separates them from other interface controllers is that all

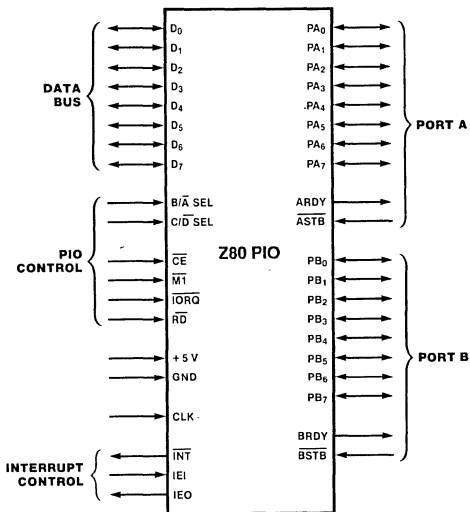


Figure 1. Pin Functions

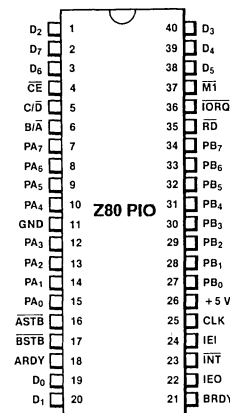


Figure 2a. 40-pin Dual-In-Line Package (DIP), Pin Assignments

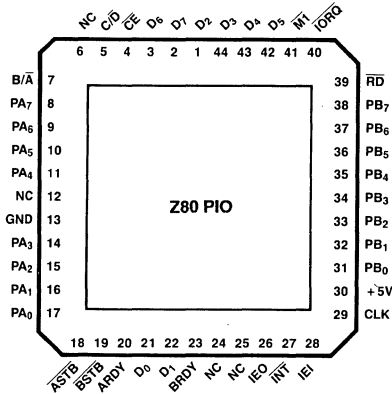


Figure 2b. 44-pin Chip Carrier, Pin Assignments

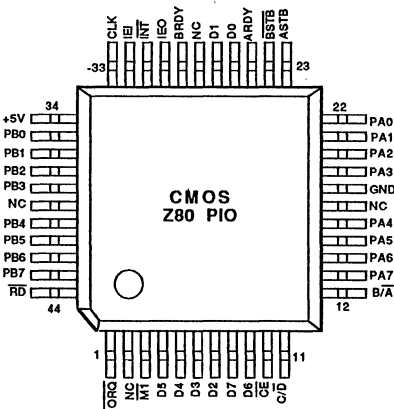


Figure 2c. 44-pin Quad Flat Pack Pin Assignments.

data transfer between the peripheral device and the CPU is accomplished under interrupt control. Thus, the interrupt logic of the PIO permits full use of the efficient interrupt capabilities of the Z80 CPU during I/O transfers. All logic necessary to implement a fully nested interrupt structure is included in the PIO (Figure 3).

Another feature of the PIO is the ability to interrupt the CPU upon occurrence of specified status conditions in the peripheral device. For example, the PIO can be programmed to interrupt if any specified peripheral alarm conditions should occur. This interrupt capability reduces the time the processor must spend in polling peripheral status.

The Z80 PIO interfaces to peripherals via two independent general-purpose I/O ports, designated Port A and Port B. Each port has eight data bits and two handshake signals, Ready and Strobe, which control data transfer. The Ready

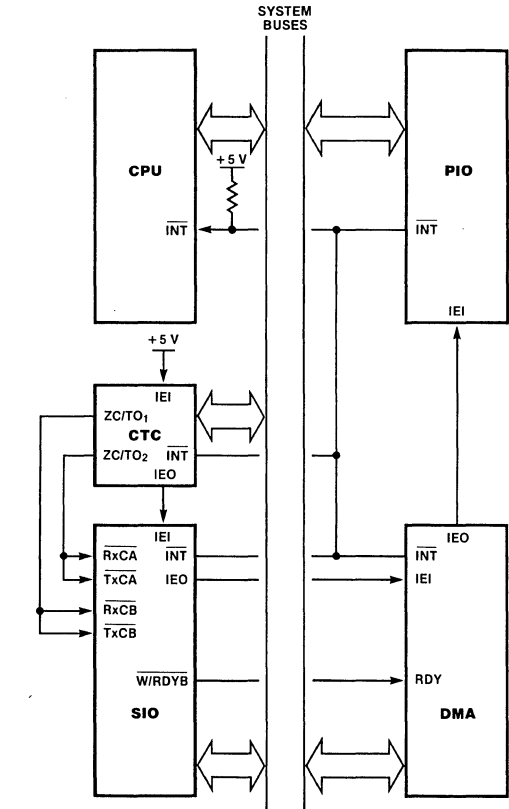


Figure 3. PIO in a Typical Z80 Family Environment

output indicates to the peripheral that the port is ready for a data transfer. Strobe is an input from the peripheral that indicates when a data transfer has occurred.

**Operating Modes.** The Z80 PIO ports can be programmed to operate in four modes: Output (Mode 0), Input (Mode 1), Bidirectional (Mode 2) and Bit Control (Mode 3).

Either Port A or Port B can be programmed to output data in Mode 0. Both ports have output registers that are individually addressed by the CPU; data can be written to either port at any time. When data is written to a port, an active Ready output indicates to the external device that data is available at the associated port and is ready for transfer to the external device. After the data transfer, the external device responds with an active Strobe input, which generates an interrupt, if enabled.

Either Port A or Port B can be programmed to input data in Mode 1. Each port has an input register addressed by the

CPU. When the CPU reads data from a port, the PIO sets the Ready signal, which is detected by the external device. The external device then places data on the I/O lines and strobes the I/O port, which latches the data into the Port Input Register, resets Ready, and triggers the Interrupt Request, if enabled. The CPU can read the input data at any time, which again sets Ready.

Mode 2 is bidirectional and uses only Port A, plus the interrupts and handshake signals from both ports. Port B must be set to Mode 3 and masked off from generating interrupts. In operation, Port A is used for both data input and output. Output operation is similar to Mode 0 except that data is allowed out onto the Port A bus only when  $\overline{ASTB}$  is Low. For input, operation is similar to Mode 1, except that the data input uses the Port B handshake signals and the Port B interrupt, if enabled.

Both ports can be used in Mode 3. In this mode, the individual bits are defined as either input or output bits. This provides up to eight separate, individually defined bits for

each port. During operation, Ready and Strobe are not used. Instead, an interrupt is generated if the condition of one input changes, or if all inputs change. The requirements for generating an interrupt are defined during the programming operation; the active level is specified as either High or Low, and the logic condition is specified as either one input active (OR) or all inputs active (AND). For example, if the port is programmed for active Low inputs and the logic function is AND, then all inputs at the specified port must go Low to generate an interrupt.

Data outputs are controlled by the CPU and can be written or changed at any time.

- Individual bits can be masked off.
- The handshake signals are not used in Mode 3; Ready is held Low, and Strobe is disabled.
- When using the Z80 PIO interrupts, the Z80 CPU interrupt mode must be set to Mode 2.

## INTERNAL STRUCTURE

The internal structure of the Z80 PIO consists of a Z80 CPU bus interface, internal control logic, Port A I/O logic, Port B I/O logic, and interrupt control logic (Figure 4). The CPU bus interface logic allows the Z80 PIO to interface directly to the Z80 CPU with no other external logic. The internal control logic synchronizes the CPU data bus to the peripheral device interfaces (Port A and Port B). The two I/O ports (A and B) are virtually identical and are used to interface directly to peripheral devices.

**Port Logic.** Each port contains separate input and output registers, handshake control logic, and the control registers shown in Figure 5. All data transfers between the peripheral unit and the CPU use the data input and output registers. The handshake logic associated with each port controls the data transfers through the input and the output registers. The mode control register (two bits) selects one of the four programmable operating modes.

The Bit Control mode (Mode 3) uses the remaining registers. The input/output control register specifies which of the eight data bits in the port are to be outputs and enables these bits; the remaining bits are inputs. The mask register and the mask control register govern Mode 3 interrupt conditions. The mask register specifies which of the bits in the port are active and which are masked or inactive.

The mask control register specifies two conditions: first, whether the active state of the input bits is High or Low, and second, whether an interrupt is generated when any one unmasked input bit is active (OR condition) or if the interrupt is generated when *all* unmasked input bits are active (AND condition).

**Interrupt Control Logic.** The interrupt control logic section handles all CPU interrupt protocol for nested-priority interrupt structures. Any device's physical location in a

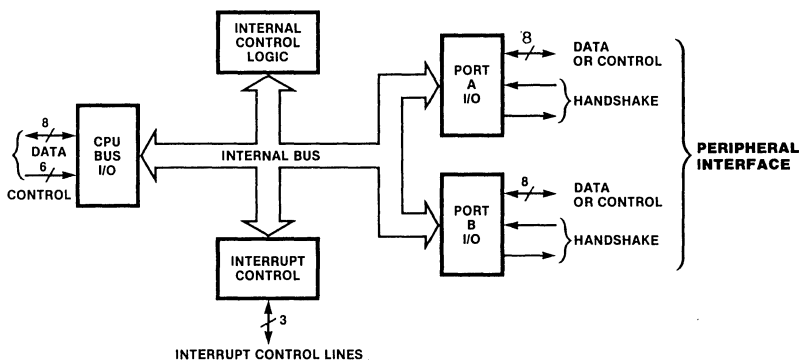


Figure 4. Block Diagram



daisy-chain configuration determines its priority. Two lines (IEI and IEO) are provided in each PIO to form this daisy chain. The device closest to the CPU has the highest priority. Within a PIO, Port A interrupts have higher priority than those of Port B. In the byte input, byte output, or bidirectional modes, an interrupt can be generated whenever the peripheral requests a new byte transfer. In the bit control mode, an interrupt can be generated when the peripheral status matches a programmed value. The PIO provides for complete control of nested interrupts. That is, lower priority devices may not interrupt higher priority devices that have not had their interrupt service routines completed by the CPU. Higher priority devices may interrupt the servicing of lower priority devices.

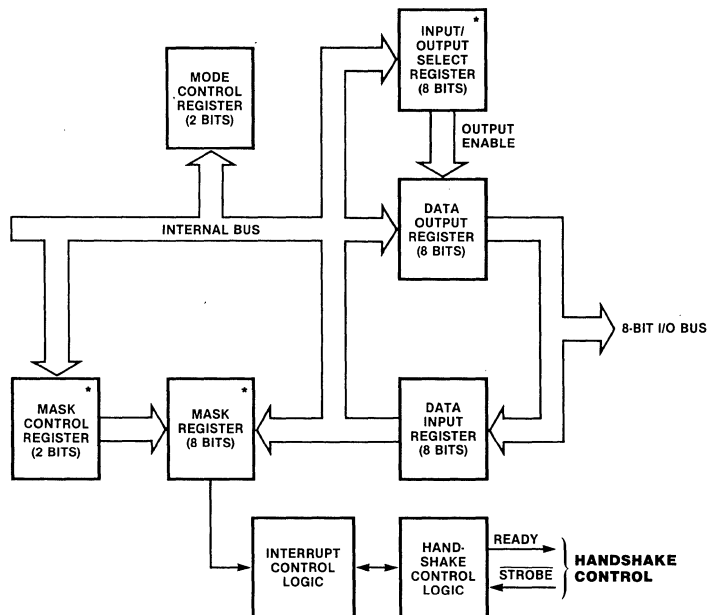
If the CPU (in interrupt Mode 2) accepts an interrupt, the interrupting device must provide an 8-bit interrupt vector for the CPU. This vector forms a pointer to a location in memory where the address of the interrupt service routine is located. The 8-bit vector from the interrupting device forms the least significant eight bits of the indirect pointer while the I Register in the CPU provides the most significant eight bits of the pointer. Each port (A and B) has an independent interrupt vector. The least significant bit of the vector is automatically set to 0 within the PIO because the pointer must point to two adjacent memory locations for a complete 16-bit address.

Unlike the other Z80 peripherals, the PIO does not enable interrupts immediately after programming. It waits until  $\overline{M1}$  goes Low (e.g., during an opcode fetch). This condition is unimportant in the Z80 environment but might not be if another type of CPU is used.

The PIO decodes the RETI (Return From Interrupt) instruction directly from the CPU data bus so that each PIO in the system knows at all times whether it is being serviced by the CPU interrupt service routine. No other communication with the CPU is required.

**CPU Bus I/O Logic.** The CPU bus interface logic interfaces the Z80 PIO directly to the Z80 CPU, so no external logic is necessary. For large systems, however, address decoders and/or buffers may be necessary.

**Internal Control Logic.** This logic receives the control words for each port during programming and, in turn, controls the operating functions of the Z80 PIO. The control logic synchronizes the port operations, controls the port mode, port addressing, selects the read/write function, and issues appropriate commands to the ports and the interrupt logic. The Z80 PIO does not receive a write input from the CPU; instead, the RD, CE, C/D and IORQ signals internally generate the write input.



\* Used in the bit mode only to allow generation of an interrupt if the peripheral I/O pins go to the specified state.

Figure 5. Typical Port I/O Block Diagram

# PROGRAMMING

**Mode 0, 1, or 2.** (Input, Output, or Bidirectional). Programming a port for Mode 0, 1, or 2 requires at least one, and up to three, control words per port. These words are:

**Mode Control Word** (Figure 6). Selects the port operating mode. This word is required and may be written at any time.

**Interrupt Vector Word** (Figure 7). The Z80 PIO is designed for use with the Z80 CPU in interrupt Mode 2. This word must be programmed if interrupts are to be used.

**Interrupt Control Word** (Figure 9) or **Interrupt Disable Word** (Figure 11). Controls the enable or disable of the PIO interrupt function.

**Mode 3** (Bit Control). Programming a port for Mode 3 requires at least two, and up to four, control words.

**Mode Control Word** (Figure 6). Selects the port operating mode. This word is required and may be written at any time.

**I/O Register Control Word** (Figure 8). When Mode 3 is selected, the Mode Control Word must be followed by the I/O Control Word. This word configures the I/O control register, which defines which port lines are inputs or outputs. This word is required.

**Interrupt Vector Word** (Figure 7). The Z80 PIO is designed for use with the Z80 CPU in interrupt Mode 2. This word must be programmed if interrupts are to be used.

**Interrupt Control Word.** In Mode 3, handshake is not used. Interrupts are generated as a logic function of the input signal levels. The interrupt control word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available: AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits changes to the active level, an interrupt is triggered). Bit D<sub>5</sub> sets the logic function, as shown in Figure 9. The active level of the input bits can be set either High or Low. The active level is controlled by Bit D<sub>5</sub>.

**Mask Control Word.** This word sets the mask control register, allowing any unused bits to be masked off. If any bits are to be masked, then D<sub>4</sub> must be set. When D<sub>4</sub> is set, the next word written to the port must be a mask control word (Figure 10).

**Interrupt Disable Word.** This control word can be used to enable or disable a port interrupt. It can be used without changing the rest of the interrupt control word (Figure 11).

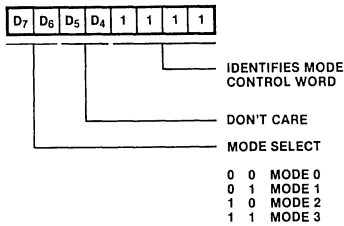


Figure 6. Mode Control Word

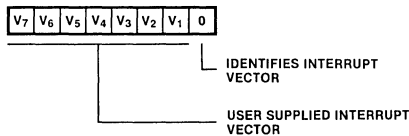


Figure 7. Interrupt Vector Word

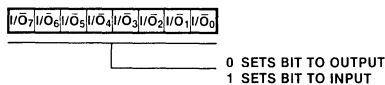
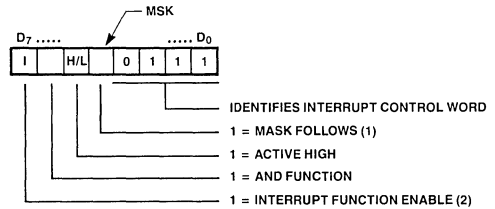


Figure 8. I/O Register Control Word



**\*NOTE:**

1. Regardless of the operating mode, setting Bit D<sub>4</sub> = 1 causes any pending interrupts to be cleared.
2. The port interrupt is not enabled until the interrupt function enable is followed by an active M1.

Figure 9. Interrupt Control Word

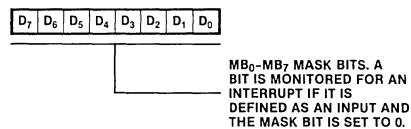


Figure 10. Mask Control Word

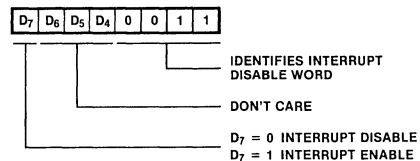


Figure 11. Interrupt Disable Word

---

## PIN DESCRIPTION

**PA<sub>0</sub>-PA<sub>7</sub>.** *Port A Bus* (bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port A of the PIO and a peripheral device. PA<sub>0</sub> is the least significant bit of the Port A data bus.

**ARDY.** *Register A Ready* (output, active High). The meaning of this signal depends on the mode of operation selected for Port A as follows:

**Output Mode.** This signal goes active to indicate that the Port A output register has been loaded and the peripheral data bus is stable and ready for transfer to the peripheral device.

**Input Mode.** This signal is active when the Port A input register is empty and ready to accept data from the peripheral device.

**Bidirectional Mode.** This signal is active when data is available in the Port A output register for transfer to the peripheral device. In this mode, data is not placed on the Port A data bus, unless  $\overline{\text{ASTB}}$  is active.

**Control Mode.** This signal is disabled and forced to a Low state.

**$\overline{\text{ASTB}}$ .** *Port A Strobe Pulse From Peripheral Device* (input, active Low). The meaning of this signal depends on the mode of operation selected for Port A as follows:

**Output Mode.** The positive edge of this strobe is issued by the peripheral to acknowledge the receipt of data made available by the PIO.

**Input Mode.** The strobe is issued by the peripheral to load data from the peripheral into the Port A input register. Data is loaded into the PIO when this signal is active.

**Bidirectional Mode.** When this signal is active, data from the Port A output register is gated onto the Port A bidirectional data bus. The positive edge of the strobe acknowledges the receipt of the data.

**Control Mode.** The strobe is inhibited internally.

**PB<sub>0</sub>-PB<sub>7</sub>.** *Port B Bus* (bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port B and a peripheral device. The Port B data bus can supply 1.5 mA at 1.5V to drive Darlington transistors. PB<sub>0</sub> is the least significant bit of the bus.

**B/ $\overline{\text{A}}$ .** *Port B or A Select* (input, High = B). This pin defines which port is accessed during a data transfer between the CPU and the PIO. A Low on this pin selects Port A; a High selects Port B. Often address bit A<sub>0</sub> from the CPU is used for this selection function.

**BRDY.** *Register B Ready* (output, active High). This signal is similar to ARDY, except that in the Port A bidirectional mode this signal is High when the Port A input register is empty and ready to accept data from the peripheral device.

**$\overline{\text{BSTB}}$ .** *Port B Strobe Pulse From Peripheral Device* (input, active Low). This signal is similar to  $\overline{\text{ASTB}}$ , except that in the Port A bidirectional mode this signal strobes data from the peripheral device into the Port A input register.

**C/ $\overline{\text{D}}$ .** *Control or Data Select* (input, High = C). This pin defines the type of data transfer to be performed between the CPU and the PIO. A High on this pin during a CPU write to the PIO causes the Z80 data bus to be interpreted as a *command* for the port selected by the B/ $\overline{\text{A}}$  Select line. A Low on this pin means that the Z80 data bus is being used to transfer data between the CPU and the PIO. Often address bit A<sub>1</sub> from the CPU is used for this function.

**$\overline{\text{CE}}$ .** *Chip Enable* (input, active Low). A Low on this pin enables the PIO to accept command or data inputs from the CPU during a write cycle or to transmit data to the CPU during a read cycle. This signal is generally decoded from four I/O port numbers for Ports A and B, data, and control.

**CLK.** *System Clock* (input). The Z80 PIO uses the standard single-phase Z80 system clock.

**D<sub>0</sub>-D<sub>7</sub>.** *Z80 CPU Data Bus* (bidirectional, 3-state). This bus is used to transfer all data and commands between the Z80 CPU and the Z80 PIO. D<sub>0</sub> is the least significant bit.

**IEI.** *Interrupt Enable In* (input, active High). This signal is used to form a priority-interrupt daisy chain when more than one interrupt driven device is being used. A High level on this pin indicates that no other devices of higher priority are being serviced by a CPU interrupt service routine.

**IEO.** *Interrupt Enable Out* (output, active High). The IEO signal is the other signal required to form a daisy chain priority scheme. It is High only if IEI is High and the CPU is not servicing an interrupt from this PIO. Thus this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

**$\overline{\text{INT}}$ .** *Interrupt Request* (output, open drain, active Low). When  $\overline{\text{INT}}$  is active the Z80 PIO is requesting an interrupt from the Z80 CPU.

**$\overline{\text{IORQ}}$ .** *Input/Output Request* (input from Z80 CPU, active Low).  $\overline{\text{IORQ}}$  is used in conjunction with B/ $\overline{\text{A}}$ , C/ $\overline{\text{D}}$ ,  $\overline{\text{CE}}$ , and  $\overline{\text{RD}}$  to transfer commands and data between the Z80 CPU and the Z80 PIO. When  $\overline{\text{CE}}$ ,  $\overline{\text{RD}}$ , and  $\overline{\text{IORQ}}$  are active, the port addressed by B/ $\overline{\text{A}}$  transfers data to the CPU (a read operation). Conversely, when  $\overline{\text{CE}}$  and  $\overline{\text{IORQ}}$  are active but  $\overline{\text{RD}}$  is not, the port addressed by B/ $\overline{\text{A}}$  is written into from the CPU with either data or control information, as specified by C/ $\overline{\text{D}}$ . Also, if  $\overline{\text{IORQ}}$  and M1 are active simultaneously, the CPU is acknowledging an interrupt; the interrupting port automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

**$\overline{M1}$ .** *Machine Cycle* (input from CPU, active Low). This signal is used as a sync pulse to control several internal PIO operations. When both the  $\overline{M1}$  and  $\overline{RD}$  signals are active, the Z80 CPU is fetching an instruction from memory. Conversely, when both  $\overline{M1}$  and  $\overline{IORQ}$  are active, the CPU is acknowledging an interrupt. In addition,  $\overline{M1}$  has two other functions within the Z80 PIO: it synchronizes the PIO

interrupt logic; when  $\overline{M1}$  occurs without an active  $\overline{RD}$  or  $\overline{IORQ}$  signal, the PIO is reset.

**$\overline{RD}$ .** *Read Cycle Status* (input from Z80 CPU, active Low). If  $\overline{RD}$  is active, or an I/O operation is in progress,  $\overline{RD}$  is used with  $B/\overline{A}$ ,  $C/\overline{D}$ ,  $\overline{CE}$ , and  $\overline{IORQ}$  to transfer data from the Z80 PIO to the Z80 CPU.

## TIMING

The following timing diagrams show typical timing in a Z80 CPU environment. For more precise specifications refer to the composite ac timing diagram.

**Write Cycle.** Figure 12 illustrates the timing for programming the Z80 PIO or for writing data to one of its ports. The PIO does not receive a specific write signal; it internally generates its own from the lack of an active  $\overline{RD}$  signal.

**Read Cycle.** Figure 13 illustrates the timing for reading the data input from an external device to one of the Z80 PIO ports.

**Output Mode (Mode 0).** An output cycle (Figure 14) is always started by the execution of an output instruction by the CPU. The  $\overline{WR}^*$  pulse from the CPU latches the data from the CPU data bus into the selected port's output register. The  $\overline{WR}^*$  pulse sets the Ready flag after a Low-going edge of CLK, indicating data is available. Ready stays active until the positive edge of the strobe line is received, indicating that data was taken by the peripheral. The positive edge of the strobe pulse generates an  $\overline{INT}$  if the interrupt enable flip-flop has been set and if this device has the highest priority.

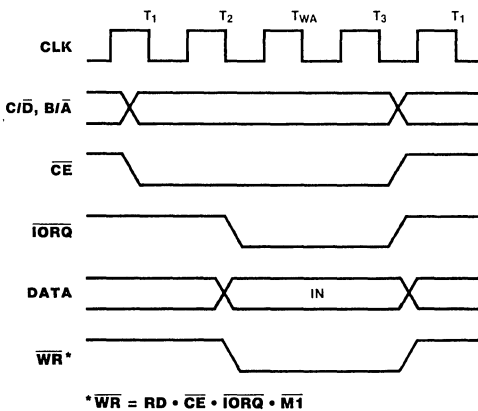


Figure 12. Write Cycle Timing

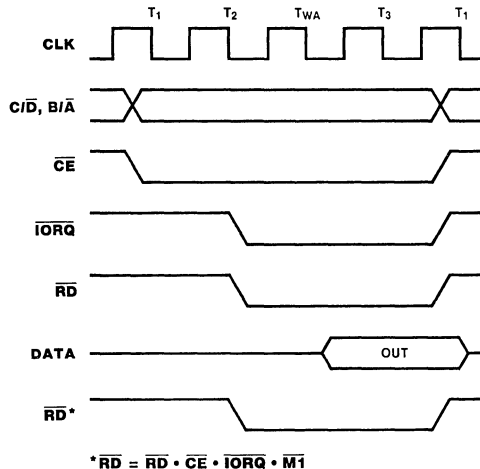


Figure 13. Read Cycle Timing

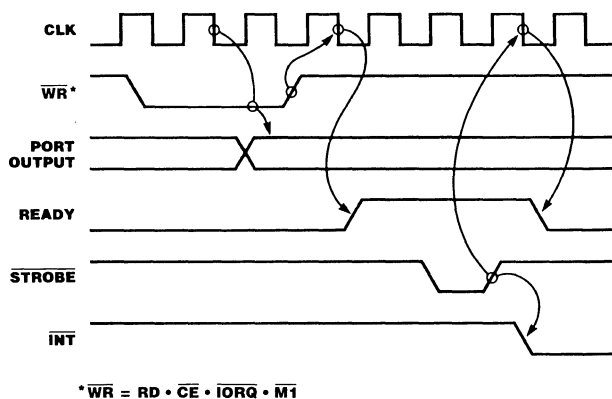


Figure 14. Mode 0 Output Timing

**Input Mode (Mode 1).** When  $\overline{\text{STROBE}}$  goes from Low to High, data is latched into the selected port input register (Figure 15). While  $\overline{\text{STROBE}}$  is Low, the input data latches are transparent. The next rising edge of  $\overline{\text{STROBE}}$  activates  $\overline{\text{INT}}$ , if Interrupt Enable is set and this is the highest-priority requesting device. The following falling edge of CLK resets Ready to an inactive state, indicating that the input register is full and cannot accept any more data until the CPU completes a read. When a read is complete, the positive edge of  $\overline{\text{RD}}$  sets Ready at the next Low-going transition of CLK. At this time new data can be loaded into the PIO.

**Bidirectional Mode (Mode 2).** This is a combination of Modes 0 and 1 using all four handshake lines and the eight Port A I/O lines (Figure 16). Port B must be set to the bit mode and its inputs must be masked. The Port A handshake lines are used for output control and the Port B lines are used for input control. If interrupts occur, Port A's vector will be used during port output and Port B's will be used during port input. Data is allowed out onto the Port A bus only when  $\overline{\text{ASTB}}$  is Low. The rising edge of this strobe can be used to latch the data into the peripheral.

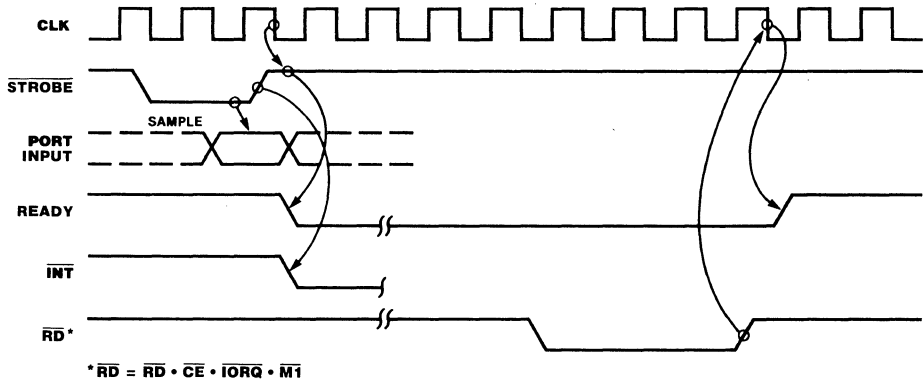


Figure 15. Mode 1 Input Timing

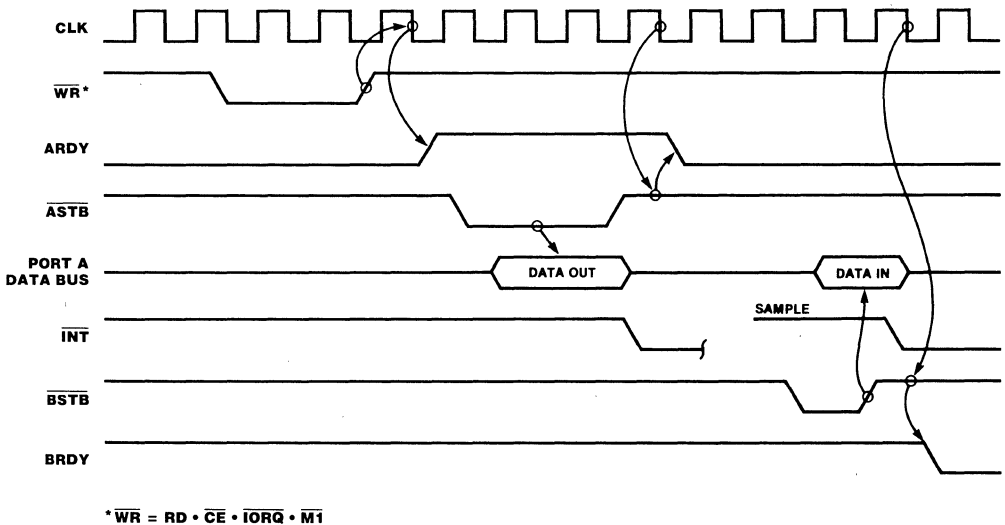


Figure 16. Mode 2 Bidirectional Timing

**Bit Control Mode (Mode 3).** The bit mode does not utilize the handshake signals, and a normal port write or port read can be executed at any time. When writing, the data is latched into the output registers with the same timing as the output mode.

When reading (Figure 17) the PIO, the data returned to the CPU is composed of output register data from those port data lines assigned as outputs and input register data from those port data lines assigned as inputs. The input register contains data that was present immediately prior to the falling edge of  $\overline{RD}$ . An interrupt is generated if interrupts from the port are enabled and the data on the port data lines satisfy the logical equation defined by the 8-bit mask and 2-bit mask control registers. However, if Port A is programmed in bidirectional mode, Port B does not issue an interrupt in bit mode and must therefore be polled.

**Interrupt Acknowledge Timing.** During  $\overline{M1}$  time, peripheral controllers are inhibited from changing their interrupt enable status, permitting the Interrupt Enable signal to ripple through the daisy chain. The peripheral with IEI High and IEO Low during INTACK places a preprogrammed 8-bit interrupt vector on the data bus at this time (Figure 18). IEO is held Low until a Return From

Interrupt (RETI) instruction is executed by the CPU while IEI is High. The 2-byte RETI instruction is decoded internally by the PIO for this purpose.

**Return From Interrupt Cycle.** If a Z80 peripheral has no interrupt pending and is not under service, then its IEO = IEI. If it has an interrupt under service (i.e., it has already interrupted and received an interrupt acknowledge) then its IEO is always Low, inhibiting lower priority devices from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO is Low unless an "ED" is decoded as the first byte of a 2-byte opcode (Figure 19). In this case, IEO goes High until the next opcode byte is decoded, whereupon it goes Low again. If the second byte of the opcode was a "4D," then the opcode was an RETI instruction.

After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service has its IEI High and its IEO Low. This device is the highest-priority device in the daisy chain that has received an interrupt acknowledge. All other peripherals have IEI = IEO. If the next opcode byte decoded is "4D," this peripheral device resets its "interrupt under service" condition.

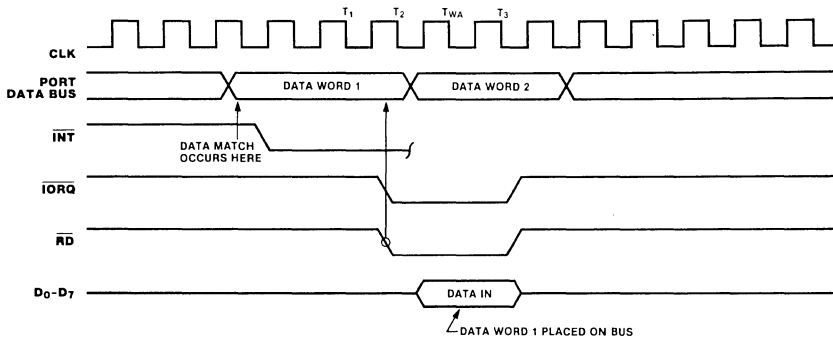


Figure 17. Mode 3 Bit Control Mode Timing, Bit Mode Read

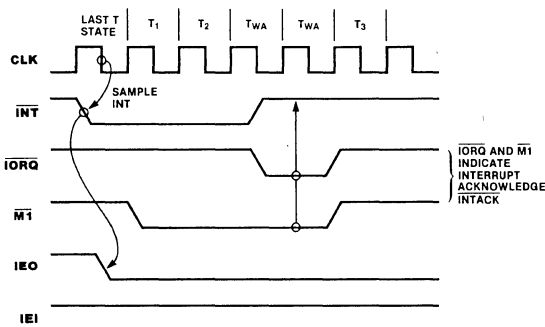


Figure 18. Interrupt Acknowledge Timing

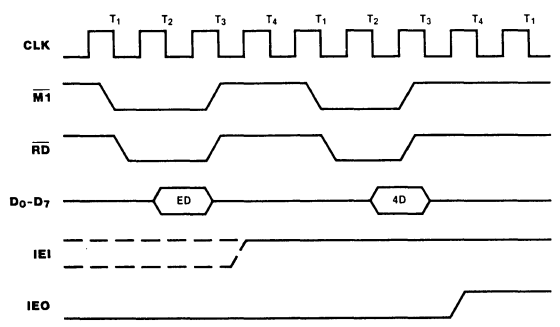


Figure 19. Return From Interrupt

## ABSOLUTE MAXIMUM RATINGS

Voltages on  $V_{CC}$  with respect to  $V_{SS}$  . . . . . -0.3V to +7.0V  
 Voltages on all inputs with respect  
 to  $V_{SS}$  . . . . . -0.3V to  $V_{CC} + 0.3V$   
 Storage Temperature . . . . . -65°C to +150°C

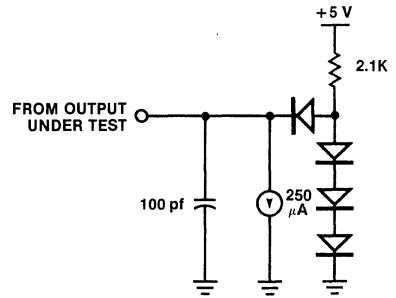
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above these indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature range is:

- **S = 0°C to +70°C,  $V_{CC}$  Range**  
 NMOS:  $+4.75V \leq V_{CC} \leq +5.25V$   
 CMOS:  $+4.50V \leq V_{CC} \leq +5.50V$
- **E = -40°C to 100°C,  $+4.50V \leq V_{CC} \leq +5.50V$**

The Ordering Information section lists package temperature ranges and product numbers. Refer to the Literature List for additional documentation. Package drawings are in the Package Information section.



## CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C	Clock Capacitance		10	pf
$C_{IN}$	Input Capacitance		5	pf
$C_{OUT}$	Output Capacitance		15	pf

Over specified temperature range;  $f = 1$  MHz.  
 Unmeasured pins returned to ground.

**DC CHARACTERISTICS (Z84C20/CMOS Z80 PIO)** $V_{CC}=5.0V \pm 10\%$ , unless otherwise specified

Symbol	Parameter	Min	Max	Typ	Unit	Condition
$V_{ILC}$	Clock Input Low Voltage	-0.3	+0.45		V	
$V_{IHC}$	Clock Input High Voltage	$V_{CC}-0.6$	$V_{CC}+0.3$		V	
$V_{IH}$	Input High Voltage	2.2	$V_{CC}$		V	
$V_{IL}$	Input Low Voltage	-0.3	0.8		V	
$V_{OL}$	Output Low Voltage		0.4		V	$I_{LO}=2.0mA$
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH}=-1.6mA$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$			V	$I_{OH}=-250\mu A$
$I_{LI}$	Input Leakage Current	-10	10		$\mu A$	$V_{IN}=0.4V$ to $V_{CC}$
$I_{LO}$	3-state Output Leakage Current in Float	-10	10		$\mu A$	$V_{OUT}=0.4V$ to $V_{CC}$
$I_{CC1}$	Power Supply Current - 4MHz		5	2	mA	$V_{CC}=5V$
	- 6MHz		6		mA	CLK=4,6,8,10MHz
	- 8MHz		7		mA	$V_{IH}=V_{CC}-0.2V$
	- 10MHz		12		mA	$V_{IL}=0.2V$
$I_{CC2}$	Standby Supply Current		10	0.5	$\mu A$	$V_{CC}=5V$ CLK=(0) $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
$I_{OHID}$	Darlington Drive Current (Port B Only)	-1.5	-5.0		mA	$V_{OH1}=1.5V$ REXT=1.1K ohm

**Note:**

[1] Measurements made with outputs floating.



## AC CHARACTERISTICS (Z84C20/CMOS Z80 PIO)

No	Symbol	Parameter	Z84C2004*		Z84C2006		Z84C2008		Z84C2010		Note
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TcC	Clock Cycle Time	250	[1]	162	[1]	125	[1]	100	[1]	
2	TwCh	Clock Pulse Width (High)	110	DC	65	DC	55	DC	42	DC	
3	TwCl	Clock Pulse Width (Low)	110	DC	65	DC	55	DC	42	DC	
4	TfC	Clock Fall Time		30		20		10		10	
5	TrC	Clock Rise Time		30		20		10		10	
6	TsCS(RI)	/CE,B//A,C//D to /RD, /IORQ Fall Setup Time	50		50		40		35		[6]
7	Th	Any Hold Times for Specified Setup Time	40		35		15		15		
8	TsRI(C)	/RD, /IORQ to Clock Rise Setup Time	115		70		60		40		
9	TdRI(DO)	/RD, /IORQ Fall to Data Out Delay		380		300		200		120	[2]
10	TdRI(DOS)	/RD, /IORQ Rise to Data Out Float Delay		110		70		60		50	
11	TsDI(C)	Data In to Clock Rise Setup Time	50		40		30		20		CL=50pF
12	TdIO(DOI)	/IORQ Falling to Data Out Delay (INTACK Cycle)		160		120		80		80	[3]
13	TsM1(Cr)	/M1 Falling to Clock Rising Setup Time	90		70		50		40		
14	TsM1(Cf)	/M1 Fall to Clock Rise Setup Time (/M1 cycle)	0		0		0		-20		
15	TdM1(IEO)	/M1 Fall to IEO Fall Delay (Interrupt Immediately Preceding /M1 Fall)		190		100		70		70	[5,7]
16	TsIEI(IO)	IEI to /IORQ Falling Setup Time (/INTACK Cycle)	140		100		80		60		[7]
17	TdIEI(IEOf)	IEI Fall to IEO Fall Delay		130		120		70		70	[5]
18	TdIEI(IEOr)	IEI Rise to IEO Rise Delay		160		150		70		70	CL=50pF
19	TcIO(C)	/IORQ Rise to Clock Fall Setup Time (To Activate RDY on Next Clock Cycle)	200		170		140		120		
20	TdC(RDYr)	Clock Fall to RDY Rise Delay		190		170		150		130	[5], CL=50pF
21	TdC(RDYf)	Clock Fall to RDY Fall Delay		140		120		100		85	[5]
22	TwSTB	/STB Pulse Width	150		120		100		80		[4]
23	TsSTB(C)	/STB Rise to Clock Fall Setup Time (To Activate RDY on Next Clock Cycle)	220		150		120		100		[5]
24	TdIO(PD)	/IORQ Rise to Port Data Stable Delay (Mode 0)		180		160		140		120	[5]
25	TsPD(STB)	Port Data to /STB Rise Setup Time (Mode 1)	230		190		140		75		
26	TdSTB(PD)	/STB Fall to Port Data Stable (Mode 2)		210		180		150		120	[5]

\*4 MHz CMOS 84C20 is obsoleted and replaced by 6 MHz.

## Z84C20 AC CHARACTERISTICS (Continued)

No	Symbol	Parameter	Z84C2004*		Z84C2006		Z84C2008		Z84C2010		Note
			Min	Max	Min	Max	Min	Max	Min	Max	
27	TdSTB(PDR)	/STB Rise to Port Data Float Delay (Mode 2)		180		160		140		120	CL=50pF
28	TdPD(INT)	Port Data Match to /INT Fall Delay (Mode 3)		490		430		360		200	
29	TdSTB(INT)	/STB Rise to /INT Fall Delay		440		350		29		220	

\* All parameters in nanosecond, unless otherwise specified.

\* 4 MHz Z84C30 is obsoleted and replaced by 6 MHz

### Notes:

[1]  $T_{cC} = T_{wCh} + T_{wCl} + T_{rC} + T_{fC}$ .

[2] Increase TdRI(DO) by 10nS for each 50pF increasing in Load up to 200pF max.

[3] Increase TdIO(DOI) by 10nS for each 50pF increasing in Load up to 200pF max.

[4] For Mode 2:  $T_{wSTB} > T_{sPD}(STB)$ .

[5] Increase these values by 2nS for 10pF increase in loading up to 100pF Max.

[6]  $T_{sCS}(RI)$  may be reduced. However, the time subtracted from  $T_{sCS}(RI)$  will be added to TdRI(DO).

[7]  $2.5T_{cT} > (N-2)T_{dIEI}(IEOf) + T_{dM1}(IEO) + T_{sEI}(IO) + T_{TL}$  Buffer Delay, if any.

[8] M1 must be active for a minimum of two clock cycles to reset the PIO.

## DC CHARACTERISTICS (Z8420/NMOS Z80 PIO)

Symbol	Parameter	Min	Max	Unit	Test Condition
V <sub>ILC</sub>	Clock Input Low Voltage	-0.3	+0.45	V	
V <sub>IHC</sub>	Clock Input High Voltage	V <sub>CC</sub> -0.6	V <sub>CC</sub> +0.3	V	
V <sub>IL</sub>	Input Low Voltage	-0.3	+0.8	V	
V <sub>IH</sub>	Input High Voltage	+2.0	V <sub>CC</sub>	V	
V <sub>OL</sub>	Output Low Voltage		+0.4	V	I <sub>OL</sub> = 2.0 mA
V <sub>OH</sub>	Output High Voltage	+2.4		V	I <sub>OH</sub> = -250 μA
I <sub>LI</sub>	Input Leakage Current		±10	μA	V <sub>IN</sub> = 0 to V <sub>CC</sub>
I <sub>LO</sub>	3-State Output Leakage Current in Float		±10	μA	V <sub>OUT</sub> = 0.4V to V <sub>CC</sub>
I <sub>CC</sub>	Power Supply Current		100	mA	
I <sub>OHD</sub>	Darlington Drive Current Port B Only	-1.5		mA	V <sub>OH</sub> = 1.5V R <sub>EXT</sub> = 390 Ω

Over specified temperature and voltage range.

## AC CHARACTERISTICS† (Z8420/NMOS Z80 PIO)

Number	Symbol	Parameter	Z0842004		Z0842006		Notes
			Min	Max	Min	Max	
1	TcC	Clock Cycle Time	250	[1]	162	[1]	
2	TwCh	Clock Width (High)	105	2000	65	2000	
3	TwC1	Clock Width (Low)	105	2000	65	2000	
4	TfC	Clock Fall Time		30		20	
5	TrC	Clock Rise Time		30		20	
6	TsCS(RI)	$\overline{CE}$ , B/ $\overline{A}$ , C/ $\overline{D}$ to $\overline{RD}$ , $\overline{IORQ}$ ↓ Setup Time	50		50		[6]
7	Th	Any Hold Times for Specified Setup Time	0		0	0	
8	TsRI(C)	$\overline{RD}$ , $\overline{IORQ}$ to Clock ↑ Setup Time	115		70		
9	TdRI(DO)	$\overline{RD}$ , $\overline{IORQ}$ ↓ to Data Out Delay		380		300	[2]
10	TdRI(DOs)	$\overline{RD}$ , $\overline{IORQ}$ ↑ to Data Out Float Delay		110		70	
11	TsDI(C)	Data In to Clock ↑ Setup Time	50		40		CL = 50 pf
12	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK Cycle)		200		120	[3]
13	TsM1(Cr)	$\overline{M1}$ ↓ to Clock ↑ Setup Time	90		70		
14	TsM1(Cf)	$\overline{M1}$ ↑ to Clock ↓ Setup Time ( $\overline{M1}$ Cycle)	0		0		[8]
15	TdM1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (Interrupt Immediately Preceding $\overline{M1}$ ↓)		190		100	[5,7]
16	TsIEI(IO)	IEI to $\overline{IORQ}$ ↓ Setup Time (INTACK Cycle)	140		100		[7]
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay		130		120	[5] CL = 50 pf
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED Decode)		160		150	[5]
19	TcIO(C)	$\overline{IORQ}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	200		170		
20	TdC(RDYr)	Clock ↓ to READY ↑ Delay		190		170	[5] CL = 50 pf
21	TdC(RDYf)	Clock ↓ to READY ↓ Delay		140		120	[5]
22	TwSTB	$\overline{STROBE}$ Pulse Width	150		120		[4]
23	TsSTB(C)	$\overline{STROBE}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	220		150		[5]
24	TdIO(PD)	$\overline{IORQ}$ ↑ to PORT DATA Stable Delay (Mode 0)		180		160	[5]
25	TsPD(STB)	PORT DATA to $\overline{STROBE}$ ↑ Setup Time (Mode 1)	230		190		
26	TdSTB(PD)	$\overline{STROBE}$ ↓ to PORT DATA Stable (Mode 2)		210		180	[5]
27	TdSTB(PDr)	$\overline{STROBE}$ ↑ to PORT DATA Float Delay (Mode 2)		180		160	CL = 50 pf
28	TdPD(INT)	PORT DATA Match to $\overline{INT}$ ↓ Delay (Mode 3)		490		430	
29	TdSTB(INT)	$\overline{STROBE}$ ↑ to $\overline{INT}$ ↓ Delay		440		350	

### NOTES:

[1] TcC = TwCh + TwCl + TrC + TfC.

[2] Increase TdRI(DO) by 10 ns for each 50 pf increase in load up to 200 pf max.

[3] Increase TdIO(DOI) by 10 ns for each 50 pf, increase in loading up to 200 pf max.

[4] For Mode 2: TwSTB > TsPD(STB).

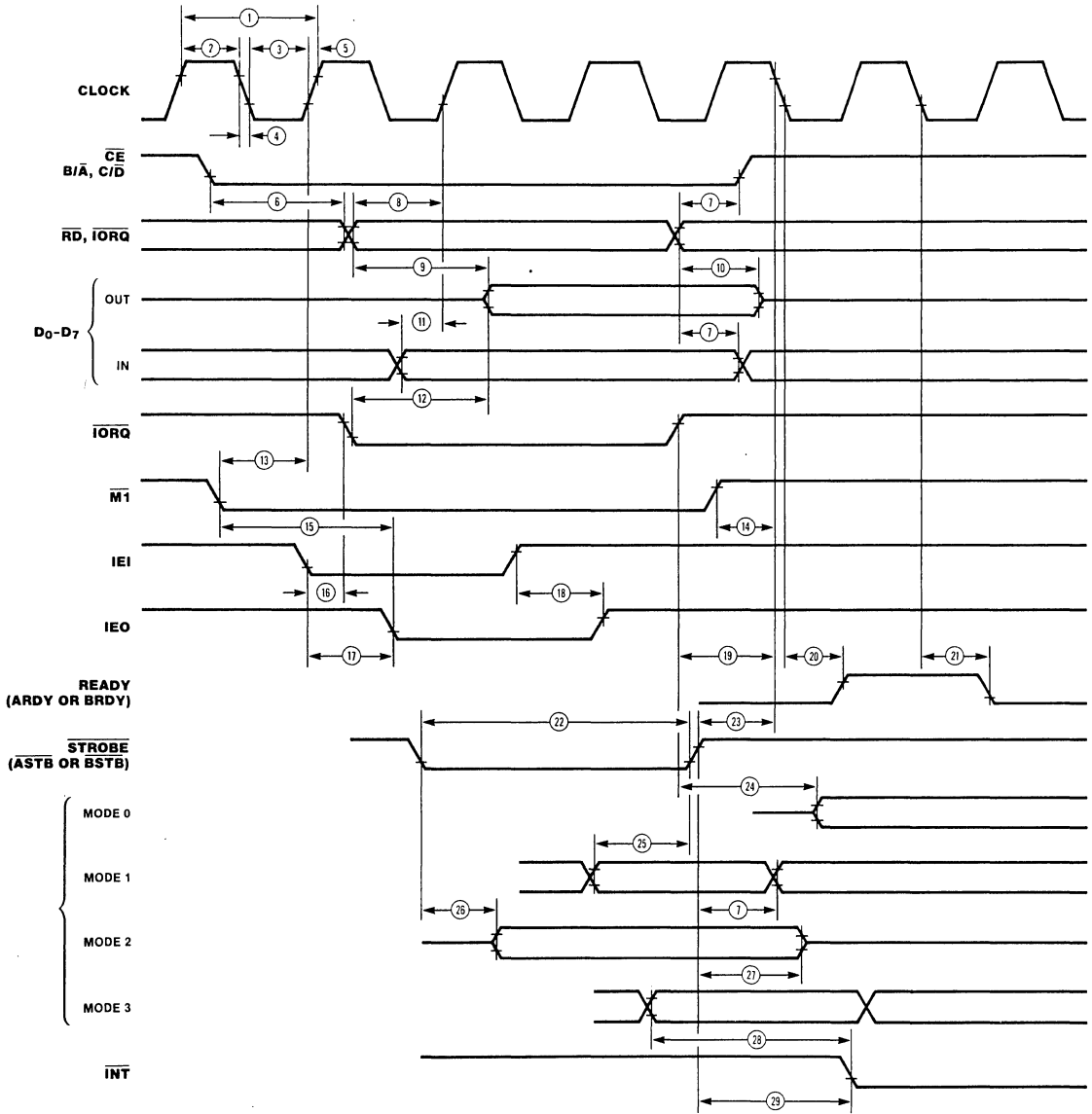
[5] Increase these values by 2 ns for each 10 pf increase in loading up to 100 pf max.

[6] TsCS(RI) may be reduced. However, the time subtracted from TsCS(RI) will be added to TdRI(DO).

\*  $\overline{M1}$  must be active for a minimum of two clock cycles to reset the PIO.

† Units in nanoseconds (ns).

# AC TIMING DIAGRAM





## Z8430/Z84C30 NMOS/CMOS Z80<sup>®</sup> CTC Counter/Timer Circuit

### FEATURES

- Four independently programmable counter/timer channels, each with a readable downcounter and a selectable 16 or 256 prescaler. Downcounters are reloaded automatically at zero count.
- Selectable positive or negative trigger initiates timer operation.
- Three channels have Zero Count/Timeout outputs capable of driving Darlington transistors. (1.5 mV @ 1.5V)
- **NMOS version for cost sensitive performance solutions.**
- **CMOS version for the designs requiring low power consumption**
- **NMOS Z0843004 - 4 MHz, Z0843006 - 6.17 MHz.**
- **CMOS Z84C3006 - DC to 6.17 MHz, Z84C3008 - DC to 8 MHz, Z84C3010 - DC to 10 MHz**
- Interfaces directly to the Z80 CPU or—for baud rate generation—to the Z80 SIO.
- Standard Z80 Family daisy-chain interrupt structure provides fully vectored, prioritized interrupts without external logic. The CTC may also be used as an interrupt controller.
- **6 MHz version supports 6.144 MHz CPU clock operation.**

### GENERAL DESCRIPTION

The Z80 CTC, hereinafter referred to as Z80 CTC or CTC, four-channel counter/timer can be programmed by system software for a broad range of counting and timing applications. The four independently programmable channels of the Z80 CTC satisfy common microcomputer system requirements for event counting, interrupt and interval timing, and general clock rate generation.

System design is simplified because the CTC connects directly to both the Z80 CPU and the Z80 SIO with no additional logic. In larger systems, address decoders and buffers may be required.

Programming the CTC is straightforward: each channel is programmed with two bytes; a third is necessary when interrupts are enabled. Once started, the CTC counts down, automatically reloads its time constant, and resumes counting. Software timing loops are completely eliminated. Interrupt processing is simplified because only one vector need be specified; the CTC internally generates a unique vector for each channel.

**The Z80 CTC requires a single +5%V power supply and the standard Z80 single-phase system clock. It is packaged in 28-pin DIPs, a 44-pin plastic chip carrier, and a 44-pin Quad Flat Pack. (Figures 2a, 2b, and 2c). Note that the QFP package is only available for CMOS versions.**

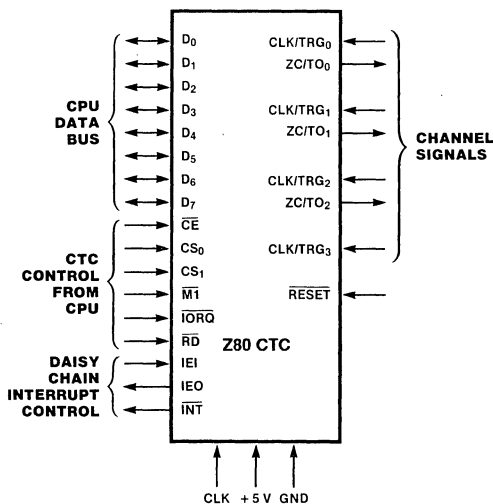


Figure 1. Pin Functions

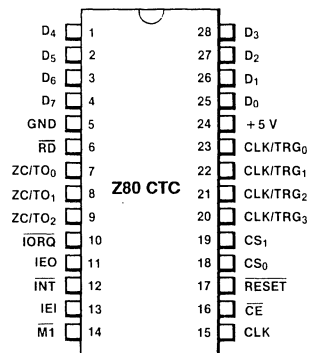


Figure 2a. Pin Assignments

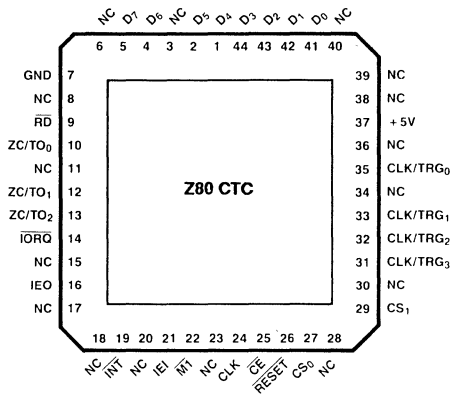


Figure 2b. 44-pin Chip Carrier, Pin Assignments

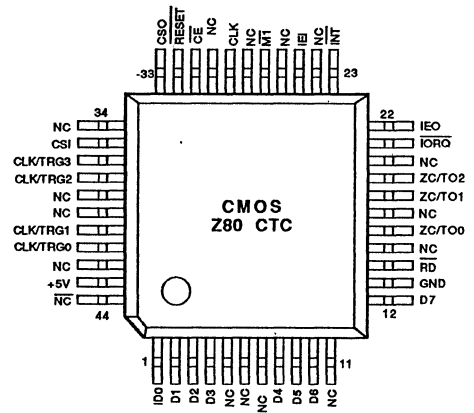


Figure 2c. 44-Pin Quad Flat Pack Pin Assignments

## FUNCTIONAL DESCRIPTION

The Z80 CTC has four independent counter/timer channels. Each channel is individually programmed with two words: a control word and a time-constant word. The control word selects the operating mode (counter or timer), enables or disables the channel interrupt, and selects certain other operating parameters. If the timing mode is selected, the control word also sets a prescaler, which divides the system clock by either 16 or 256. The time-constant word is a value from 1 to 256.

During operation, the individual counter channel counts down from the preset time constant value. In counter mode operation the counter decrements on each of the CLK/TRG input pulses until zero count is reached. Each decrement is synchronized by the system clock. For counts greater than 256, more than one counter can be cascaded. At zero count, the down-counter is automatically reset with the time constant value.

**The timer mode determines time intervals as small as 2  $\mu$ s (8 MHz), 3  $\mu$ s (6 MHz), or 4  $\mu$ s (4MHz) without additional logic or software timing loops. Time intervals are generated by dividing the system clock with a prescaler that decrements a preset down-counter.**

Thus, the time interval is an integral multiple of the clock period, the prescaler value (16 or 256), and the time constant that is preset in the down-counter. A timer is triggered automatically when its time constant value is programmed, or by an external CLK/TRG input.

Three channels have two outputs that occur at zero count. The first output is a zero-count/timeout pulse at the ZC/TO output. The fourth channel (Channel 3) does not have a ZC/TO output; interrupt request is the only output available from Channel 3.

The second output is Interrupt Request ( $\overline{INT}$ ), which occurs if the channel has its interrupt enabled during programming. When the Z80 CPU acknowledges Interrupt Request, the Z80 CTC places an interrupt vector on the data bus.

The four channels of the Z80 CTC are fully prioritized and fit into four contiguous slots in a standard Z80 daisy-chain interrupt structure. Channel 0 is the highest priority and Channel 3 the lowest. Interrupts can be individually enabled (or disabled) for each of the four channels.

## INTERNAL STRUCTURE

The CTC has four major elements, as shown in Figure 3.

- CPU bus I/O
- Channel control logic
- Interrupt logic
- Counter/timer circuits

**CPU Bus I/O.** The CPU bus I/O circuit decodes the address inputs, and interfaces the CPU data and control signals to the CTC for distribution on the internal bus.

**Internal Control Logic.** The CTC internal control logic controls overall chip operating functions such as the chip enable, reset, and read/write logic.

**Interrupt Logic.** The interrupt control logic ensures that the CTC interrupts interface properly with the Z80 CPU interrupt system. The logic controls the interrupt priority of the CTC as a function of the IEI signal. If IEI is High, the CTC has priority. During interrupt processing, the interrupt logic holds IEO Low, which inhibits the interrupt operation on lower priority devices. If the IEI input goes Low, priority is relinquished and the interrupt logic drives IEO Low.

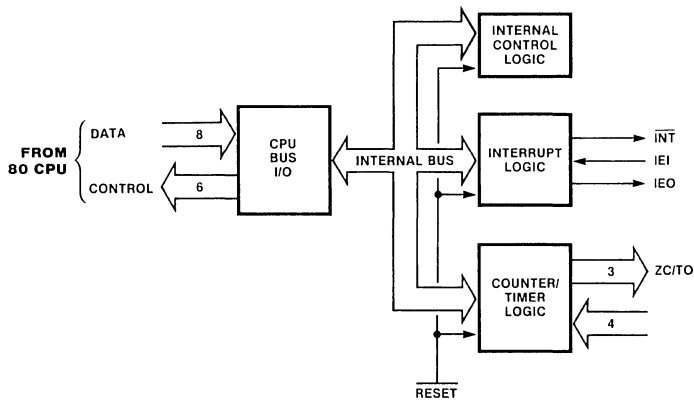


Figure 3. Functional Block Diagram

If a channel is programmed to request an interrupt, the interrupt logic drives IEO Low at the zero count, and generates an  $\overline{\text{INT}}$  signal to the Z80 CPU. When the Z80 CPU responds with interrupt acknowledge ( $\overline{\text{M1}}$  and  $\overline{\text{IORQ}}$ ), then the interrupt logic arbitrates the CTC internal priorities, and the interrupt control logic places a unique interrupt vector on the data bus.

If an interrupt is pending, the interrupt logic holds IEO Low. When the Z80 CPU issues a Return From Interrupt (RETI) instruction, each peripheral device decodes the first byte ( $\text{ED}_{16}$ ). If the device has a pending interrupt, it raises IEO (High) for one  $\overline{\text{M1}}$  cycle. This ensures that all lower priority devices can decode the entire RETI instruction and reset properly.

**Counter/Timer Circuits.** The CTC has four independent counter/timer circuits, each containing the logic shown in Figure 4.

**Channel Control Logic.** The channel control logic receives the 8-bit channel control word when the counter/timer channel is programmed. The channel control logic decodes the control word and sets the following operating conditions:

- Interrupt enable (or disable)
- Operating mode (timer or counter)
- Timer mode prescaler factor (16 or 256)
- Active slope for CLK/TRG input
- Timer mode trigger (automatic or CLK/TRG input)
- Time constant data word to follow
- Software reset

**Time Constant Register.** When the counter/timer channel is programmed, the time constant register receives and stores an 8-bit time constant value, which can be anywhere from 1 to 256 (0 = 256). This constant is automatically loaded into the down-counter when the counter/timer channel is initialized, and subsequently after each zero count.

**Prescaler.** The prescaler, which is used only in timer mode, divides the system clock frequency by a factor of either 16 or 256. The prescaler output clocks the down-counter during timer operation. The effect of the prescaler on the down-counter is a multiplication of the system clock period by 16 or 256. The prescaler factor is programmed by bit 5 of the channel control word.

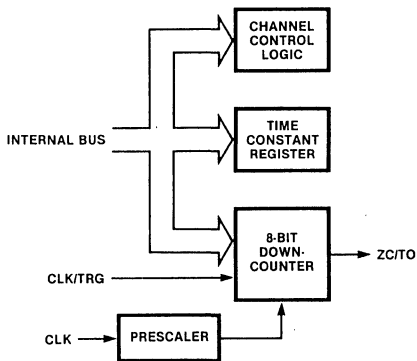


Figure 4. Counter/Timer Block Diagram



**Down-Counter.** Prior to each count cycle, the down-counter is loaded with the time constant register contents. The counter is then decremented one of two ways, depending on operating mode:

- By the prescaler output (timer mode)
- By the trigger pulses into the CLK/TRG input (counter mode)

Without disturbing the down-count, the Z80 CPU can read the count remaining at any time by performing an I/O read operation at the port address assigned to the CTC channel. When the down-counter reaches the zero count, the ZC/TO output generates a positive-going pulse. When the interrupt is enabled, zero count also triggers an interrupt request signal ( $\overline{INT}$ ) from the interrupt logic.

## PROGRAMMING

Each Z80 CTC channel must be programmed prior to operation. Programming consists of writing two words to the I/O port that corresponds to the desired channel. The first word is a control word that selects the operating mode and other parameters; the second word is a time constant, which is a binary data word with a value from 1 to 256. A time constant word must be preceded by a channel control word.

After initialization, channels may be reprogrammed at any time. If updated control and time constant words are written to a channel during the count operation, the count continues to zero before the new time constant is loaded into the counter.

If the interrupt on any Z80 CTC channel is enabled, the programming procedure should also include an interrupt vector. Only one vector is required for all four channels, because the interrupt logic automatically modifies the vector for the channel requesting service.

A control word is identified by a 1 in bit 0. A 1 in bit 2 indicates a time constant word is to follow. Interrupt vectors are always addressed to Channel 0, and identified by a 0 in bit 0.

**Addressing.** During programming, channels are addressed with the channel select pins  $CS_1$  and  $CS_2$ . A 2-bit binary code selects the appropriate channel as shown in the following table.

Channel	$CS_1$	$CS_0$
0	0	0
1	0	1
2	1	0
3	1	1

**Reset.** The CTC has both hardware and software resets. The hardware reset terminates all down-counts and disables all CTC interrupts by resetting the interrupt bits in the control registers. In addition, the ZC/TO and Interrupt outputs go inactive, IEO reflects IEI, and  $D_0$ - $D_7$  go to the high-impedance state. All channels must be completely reprogrammed after a hardware reset.

The software reset is controlled by bit 1 in the channel control word. When a channel receives a software reset, it stops counting. When a software reset is used, the other bits in the control word also change the contents of the channel control register. After a software reset a new time constant word must be written to the same channel.

If the channel control word has both bits  $D_1$  and  $D_2$  set to 1, the addressed channel stops operating, pending a new time constant word. The channel is ready to resume after the new constant is programmed. In timer mode, if  $D_3 = 0$ , operation is triggered automatically when the time constant word is loaded.

**Channel Control Word Programming.** The channel control word is shown in Figure 5. It sets the modes and parameters described below.

**Interrupt Enable.**  $D_7$  enables the interrupt, so that an interrupt output ( $\overline{INT}$ ) is generated at zero count. Interrupts may be programmed in either mode and may be enabled or disabled at any time.

**Mode.**  $D_6$  selects either timer or counter operating mode.

**Prescaler Factor. (Timer Mode Only).**  $D_5$  selects factor—either 16 or 256.

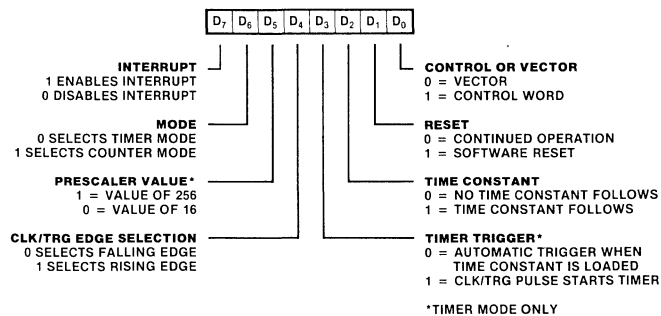


Figure 5. Channel Control Word

**Clock/Trigger Edge Selector.**  $D_4$  selects the active edge or slope of the CLK/TRG input pulses. Note that reprogramming the CLK/TRG slope during operation is equivalent to issuing an active edge. If the trigger slope is changed by a control word update while a channel is pending operation in timer mode, the result is the same as a CLK/TRG pulse and the timer starts. Similarly, if the channel is in counter mode, the counter decrements.

**Timer Trigger (Timer Mode Only).**  $D_3$  selects the trigger mode for timer operation. When  $D_3$  is reset to 0, the timer is triggered automatically. The time constant word is programmed during an I/O write operation, which takes one machine cycle. At the end of the write operation there is a setup delay of one clock period. The timer starts automatically (decrements) on the rising edge of the second clock pulse ( $T_2$ ) of the machine cycle following the write operation. Once started, the timer runs continuously. At zero count the timer reloads automatically and continues counting without interruption or delay, until stopped by a reset.

When  $D_3$  is set to 1, the timer is triggered externally through the CLK/TRG input. The time constant word is programmed during an I/O write operation, which takes one machine cycle. The timer is ready for operation on the rising edge of the second clock pulse ( $T_2$ ) of the following machine cycle. Note that the first timer decrement follows the active edge of the CLK/TRG pulse by a delay time of one clock cycle if a minimum setup time to the rising edge of clock is met. If this minimum is not met, the delay is extended by another clock period. Consequently, for immediate triggering, the CLK/TRG input must precede  $T_2$  by one clock cycle plus its minimum setup time. If the minimum time is not met, the timer will start on the third clock cycle ( $T_3$ ).

Once started the timer operates continuously, without interruption or delay, until stopped by a reset.

**Time Constant.** A 1 in  $D_2$  indicates that the next word addressed to the selected channel is a time constant data word for the time constant register. The time constant word may be written at any time.

A 0 in  $D_2$  indicates no time constant word is to follow. This is ordinarily used when the channel is already in operation and the new channel control word is an update. A channel will

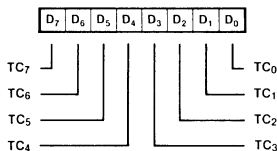


Figure 6. Time Constant Word

not operate without a time constant value. The only way to write a time constant value is to write a control word with  $D_2$  set.

**Software Reset.** Setting  $D_1$  to 1 causes a software reset, which is described in the Reset section.

**Control Word.** Setting  $D_0$  to 0 identifies the word as a control word.

**Time Constant Programming.** Before a channel can start counting it must receive a time constant word from the CPU. During programming or reprogramming, a channel control word in which bit 2 is set must precede the time constant word to indicate that the next word is a time constant. The time constant word can be any value from 1 to 256 (Figure 6). Note that  $00_{16}$  is interpreted as 256.

In timer mode, the time interval is controlled by three factors:

- The system clock period (CLK)
- The prescaler factor (P), which multiplies the interval by either 16 or 256
- The time constant (T), which is programmed into the time constant register

Consequently, the time interval is the product of  $CLK \times P \times T$ . The minimum timer resolution is  $16 \times CLK$  ( $4\mu s$  with a 4MHz clock). The maximum timer interval is  $256 \times CLK \times 256$  (16.4 ms with a 4MHz clock). For longer intervals timers may be cascaded.

**Interrupt Vector Programming.** If the Z80 CTC has one or more interrupts enabled, it can supply interrupt vectors to the Z80 CPU. To do so, the Z80 CTC must be pre-programmed with the most-significant five bits of the interrupt vector. Programming consists of writing a vector word to the I/O port corresponding to the Z80 CTC Channel 0. Note that  $D_0$  of the vector word is always zero, to distinguish the vector from a channel control word.  $D_1$  and  $D_2$  are not used in programming the vector word. These bits are supplied by the interrupt logic to identify the channel requesting interrupt service with a unique interrupt vector (Figure 7). Channel 0 has the highest priority.

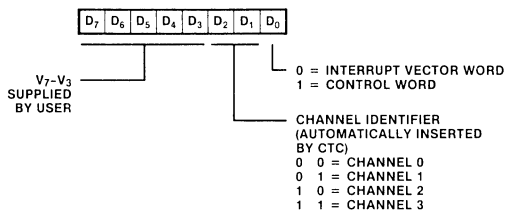


Figure 7. Interrupt Vector Word

## PIN DESCRIPTION

**$\overline{CE}$ .** *Chip Enable* (input, active Low). When enabled the CTC accepts control words, interrupt vectors, or time constant data words from the data bus during an I/O write cycle; or transmits the contents of the downcounter to the CPU during an I/O read cycle. In most applications this signal is decoded from the eight least significant bits of the address bus for any of the four I/O port addresses that are mapped to the four counter-timer channels.

**CLK.** *System Clock* (input). Standard single-phase Z80 system clock.

**CLK/TRG<sub>0</sub>-CLK/TRG<sub>3</sub>.** *External Clock/Timer Trigger* (input, user-selectable active High or Low). Four pins corresponding to the four Z80 CTC channels. In counter mode, every active edge on this pin decrements the downcounter. In timer mode, an active edge starts the timer.

**CS<sub>0</sub>-CS<sub>1</sub>.** *Channel Select* (inputs active High). Two-bit binary address code selects one of the four CTC channels for an I/O write or read (usually connected to A<sub>0</sub> and A<sub>1</sub>).

**D<sub>0</sub>-D<sub>7</sub>.** *System Data Bus* (bidirectional, 3-state). Transfers all data and commands between the Z80 CPU and the Z80 CTC.

**IEI.** *Interrupt Enable In* (input, active High). A High indicates that no other interrupting devices of higher priority in the daisy chain are being serviced by the Z80 CPU.

**IEO.** *Interrupt Enable Out* (output, active High). High only if IEI is High and the Z80 CPU is not servicing an interrupt from any Z80 CTC channel. IEO blocks lower priority devices from interrupting while a higher priority interrupting device is being serviced.

**$\overline{INT}$ .** *Interrupt Request* (output, open drain, active Low). Low when any Z80 CTC channel that has been programmed to enable interrupts as a zero-count condition in its downcounter.

**$\overline{IORQ}$ .** *Input/Output Request* (input from CPU, active Low). Used with  $\overline{CE}$  and  $\overline{RD}$  to transfer data and channel control words between the Z80 CPU and the Z80 CTC. During a write cycle,  $\overline{IORQ}$  and  $\overline{CE}$  are active and  $\overline{RD}$  inactive. The Z80 CTC does not receive a specific write signal; rather, it internally generates its own from the inverse of an active  $\overline{RD}$  signal. In a read cycle,  $\overline{IORQ}$ ,  $\overline{CE}$ , and  $\overline{RD}$  are active; the contents of the downcounter are read by the Z80 CPU. If  $\overline{IORQ}$  and  $\overline{M1}$  are both true, the CPU is acknowledging an interrupt request, and the highest priority interrupting channel places its interrupt vector on the Z80 data bus.

**$\overline{M1}$ .** *Machine Cycle One* (input from CPU, active Low). When  $\overline{M1}$  and  $\overline{IORQ}$  are active, the Z80 CPU is acknowledging an interrupt. The Z80 CTC then places an interrupt vector on the data bus if it has highest priority, and if a channel has requested an interrupt ( $\overline{INT}$ ).

**$\overline{RD}$ .** *Read Cycle Status* (input, active Low). Used in conjunction with  $\overline{IORQ}$  and  $\overline{CE}$  to transfer data and channel control words between the Z80 CPU and the Z80 CTC.

**RESET.** *Reset* (input active Low). Terminates all down-counts and disables all interrupts by resetting the interrupt bits in all control registers; the ZC/TO and the interrupt outputs go inactive; IEO reflects IEI; D<sub>0</sub>-D<sub>7</sub> go to the high-impedance state.

**ZC/TO<sub>0</sub>-ZC/TO<sub>2</sub>.** *Zero Count/Timeout* (output, active High). Three ZC/TO pins corresponding to Z80 CTC channels 2 through 0 (Channel 3 has no ZC/TO pin). In both counter and timer modes the output is an active High pulse when the downcounter decrements to zero.

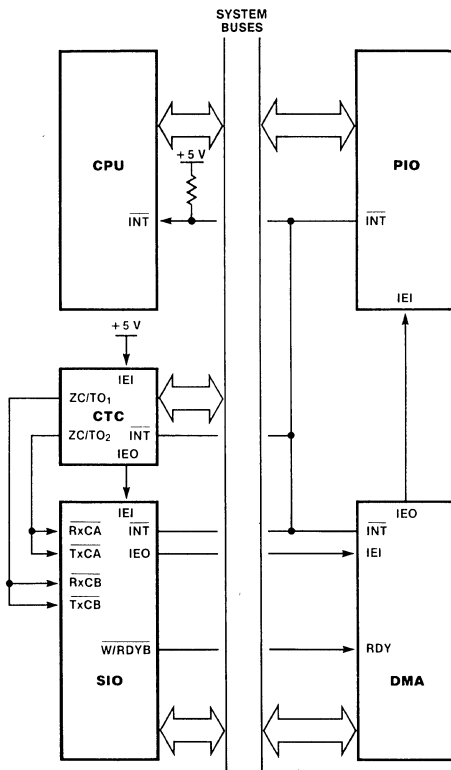


Figure 8. A Typical Z80 Environment

## TIMING

**Read Cycle Timing.** Figure 9 shows read cycle timing. This cycle reads the contents of a down-counter without disturbing the count. During clock cycle  $T_2$ , the Z80 CPU initiates a read cycle by driving the following inputs Low:  $\overline{RD}$ ,  $\overline{IORQ}$ , and  $\overline{CE}$ . A 2-bit binary code at inputs  $CS_1$  and  $CS_0$  selects the channel to be read.  $\overline{M1}$  must be High to distinguish this cycle from an interrupt acknowledge.  $\overline{M1}$  must be High to distinguish this cycle from an interrupt acknowledge.

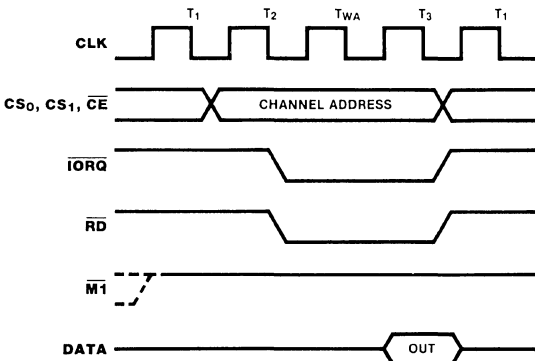


Figure 9. Read Cycle Timing

**Write Cycle Timing.** Figure 10 shows write cycle timing for loading control, time constant, or vector words.

The CTC does not have a write signal input, so it generates one internally when the read ( $\overline{RD}$ ) input is High during  $T_1$ . During  $T_2$   $\overline{IORQ}$  and  $\overline{CE}$  inputs are Low.  $\overline{M1}$  must be High to distinguish a write cycle from an interrupt acknowledge. A 2-bit binary code at inputs  $CS_1$  and  $CS_0$  selects the channel to be addressed, and the word being written is placed on the Z80 data bus. The data word is latched into the appropriate register with the rising edge of clock cycle  $T_3$ .

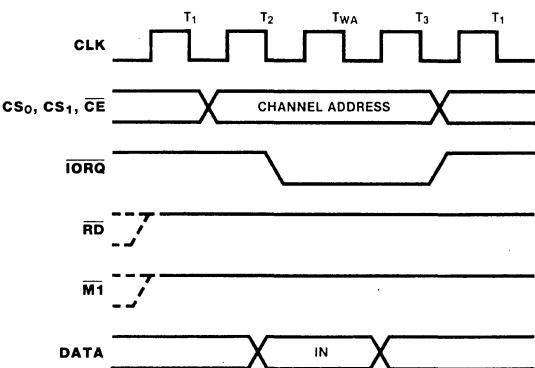


Figure 10. Write Cycle Timing

**Timer Operation.** In the timer mode, a CLK/TRG pulse input starts the timer (Figure 11) on the second succeeding rising edge of CLK. The trigger pulse is asynchronous, and it must have a minimum width. A minimum lead time (210 ns) between the active edge of the CLK/TRG and the next rising edge of CLK to enable the prescaler on the following clock edge. If the CLK/TRG edge occurs closer than this, the initiation of the timer function is delayed one clock cycle. This corresponds to the start-up timing discussed in the programming section. The timer can also be started automatically if so programmed by the channel control word.

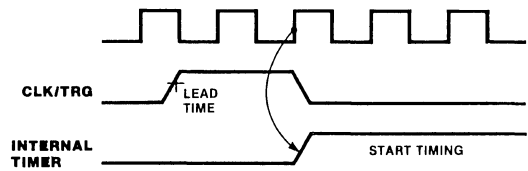


Figure 11. Timer Mode Timing

**Counter Operation.** In the counter mode, the CLK/TRG pulse input decrements the downcounter. The trigger is asynchronous, but the count is synchronized with CLK. For the decrement to occur on the next rising edge of CLK, the trigger edge must precede CLK by a minimum lead time as shown in Figure 12. If the lead time is less than specified, the count is delayed by one clock cycle. The trigger pulse must have a minimum width, and the trigger period must be at least twice the clock period. If the trigger repetition rate is faster than  $1/3$  the clock frequency, then  $T_{sCTR}(Cs)$ , AC Characteristics Specification 26, must be met.

The ZC/TO output occurs immediately after zero count, and follows the rising CLK edge.

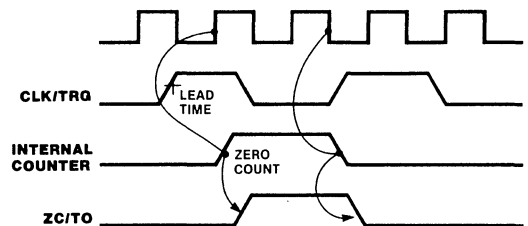


Figure 12. Counter Mode Timing

## INTERRUPT OPERATION

The Z80 CTC follows the Z80 system interrupt protocol for nested priority interrupts and return from interrupt, wherein the interrupt priority of a peripheral is determined by its location in a daisy chain. Two lines—IEI and IEO—in the CTC connect it to the system daisy chain. The device closest to the +5V supply has the highest priority (Figure 13). For additional information on the Z80 interrupt structure, refer to the *Z80 CPU Product Specification* and the *Z80 CPU Technical Manual*.

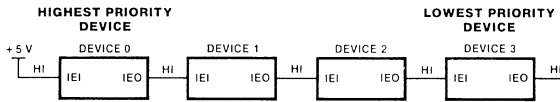


Figure 13. Daisy-Chain Interrupt Priorities

Within the Z80 CTC, interrupt priority is predetermined by channel number: Channel 0 has the highest priority, and Channel 3 the lowest. If a device or channel is being serviced with an interrupt routine, it cannot be interrupted by a device or channel with lower priority until service is complete. Higher priority devices or channels may interrupt the servicing of lower priority devices or channels.

A Z80 CTC channel may be programmed to request an interrupt every time its downcounter reaches zero. Note that the CPU must be programmed for interrupt mode 2. Some time after the interrupt request, the CPU sends an interrupt acknowledge. The CTC interrupt control logic determines the highest priority channel that is requesting an interrupt. Then, if the CTC IEI input is High (indicating that it has priority within the system daisy chain) it places an 8-bit

interrupt vector on the system data bus. The high-order five bits of this vector were written to the CTC during the programming process; the next two bits are provided by the CTC interrupt control logic as a binary code that identifies the highest priority channel requesting an interrupt; the low-order bit is always zero.

**Interrupt Acknowledge Timing.** Figure 14 shows interrupt acknowledge timing. After an interrupt request, the Z80 CPU sends an interrupt acknowledge ( $\overline{M1}$  and  $\overline{IORQ}$ ). All channels are inhibited from changing their interrupt request status when  $\overline{M1}$  is active—about two clock cycles earlier than  $\overline{IORQ}$ .  $\overline{RD}$  is High to distinguish this cycle from an instruction fetch.

The CTC interrupt logic determines the highest priority channel requesting an interrupt. If the CTC interrupt enable input (IEI) is High, the highest priority interrupting channel within the CTC places its interrupt vector on the data bus when  $\overline{IORQ}$  goes Low. Two wait states ( $T_{WA}$ ) are automatically inserted at this time to allow the daisy chain to stabilize. Additional wait states may be added.

**Return from Interrupt Timing.** At the end of an interrupt service routine the RETI (Return From Interrupt) instruction initializes the daisy chain enable lines for proper control of nested priority interrupt handling. The CTC decodes the 2-byte RETI code internally and determines whether it is intended for a channel being serviced. Figure 15 shows RETI timing.

If several Z80 peripherals are in the daisy chain, IEI settles active (High) on the chip currently being serviced when the opcode  $ED_{16}$  is decoded. If the following opcode is  $4D_{16}$ , the peripheral being serviced is released and its IEO becomes active. Additional wait states are allowed.

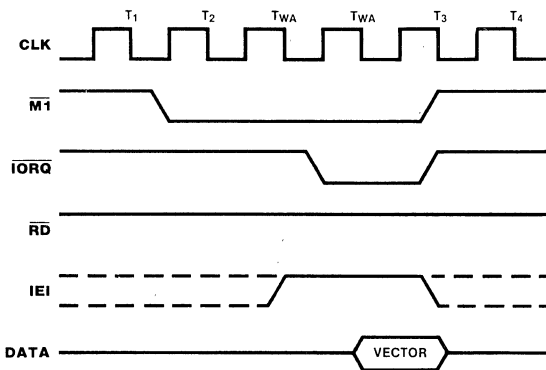


Figure 14. Interrupt Acknowledge Timing

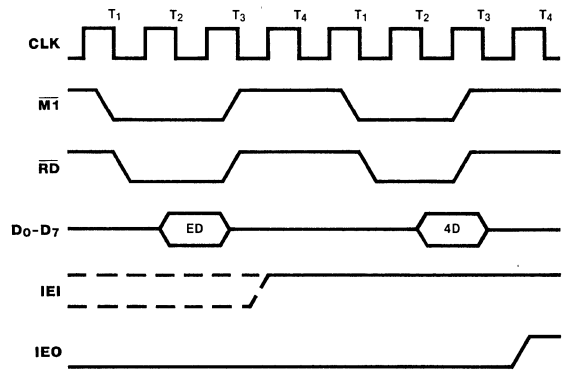


Figure 15. Return From Interrupt Timing

## ABSOLUTE MAXIMUM RATINGS

Voltages on  $V_{CC}$  with respect to  $V_{SS}$  . . . . . -0.3V to +7.0V  
 Voltages on all inputs with respect  
 to  $V_{SS}$  . . . . . -0.3V to  $V_{CC} + 0.3V$   
 Storage Temperature . . . . . -65°C to +150°C

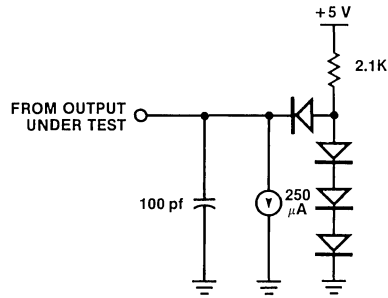
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above these indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature range is:

- **S = 0°C to +70°C,  $V_{CC}$  Range**  
 NMOS:  $+4.75V \leq V_{CC} \leq +5.25V$   
 CMOS:  $+4.50V \leq V_{CC} \leq +5.50V$
- **E = -40°C to 100°C,  $+4.50V \leq V_{CC} \leq +5.50V$**

The Ordering Information section lists package temperature ranges and product numbers. Refer to the Literature List for additional documentation. Package drawings are in the Package Information section.



## DC CHARACTERISTICS (Z84C30/CMOS Z80 CTC)

$V_{CC} = 5.0V \pm 10\%$ , unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Condition
$V_{ILC}$	Clock Input Low Voltage	-0.3	+0.45	V	
$V_{IHIC}$	Clock Input High Voltage	$V_{CC} - 0.6$	$V_{CC} + 0.3$	V	
$V_{IL}$	Input High Voltage	2.2	$V_{CC}$	V	
$V_{IH}$	Input Low Voltage	-0.3	0.8	V	
$V_{OL}$	Output Low Voltage		0.4	V	$I_{LO} = 2.0mA$
$V_{OH1}$	Output High Voltage	2.4		V	$I_{OH} = -1.6mA$
$V_{OH2}$	Output High Voltage	$V_{CC} - 0.8$		V	$I_{OH} = -250\mu A$
$I_{LI}$	Input Leakage Current	-10	10	$\mu A$	$V_{IN} = 0.4V$ to $V_{CC}$
$I_{LO}$	3-state Output Leakage Current in Float	-10	10	$\mu A$	$V_{OUT} = 0.4V$ to $V_{CC}$
$I_{CC1}$	Power Supply Current - 4MHz		7 [1]	mA	$V_{CC} = 5V$
	- 6MHz		8 [1]	mA	CLK = 4, 6, 8, 10MHz
	- 8MHz		10 [1]	mA	$V_{IH} = V_{CC} - 0.2V$
	- 10MHz		12 [1]	mA	$V_{IL} = 0.2V$
$I_{CC2}$	Standby Supply Current		10	$\mu A$	$V_{CC} = 5V$ CLK = (0) $V_{IH} = V_{CC} - 0.2V$ $V_{IL} = 0.2V$
$I_{OHID}$	Darlington Drive Current	-1.5	-5.0	mA	$V_{OH} = 1.5V$ REXT = 1.1K ohm

Note: [1] Measurements made with outputs floating.

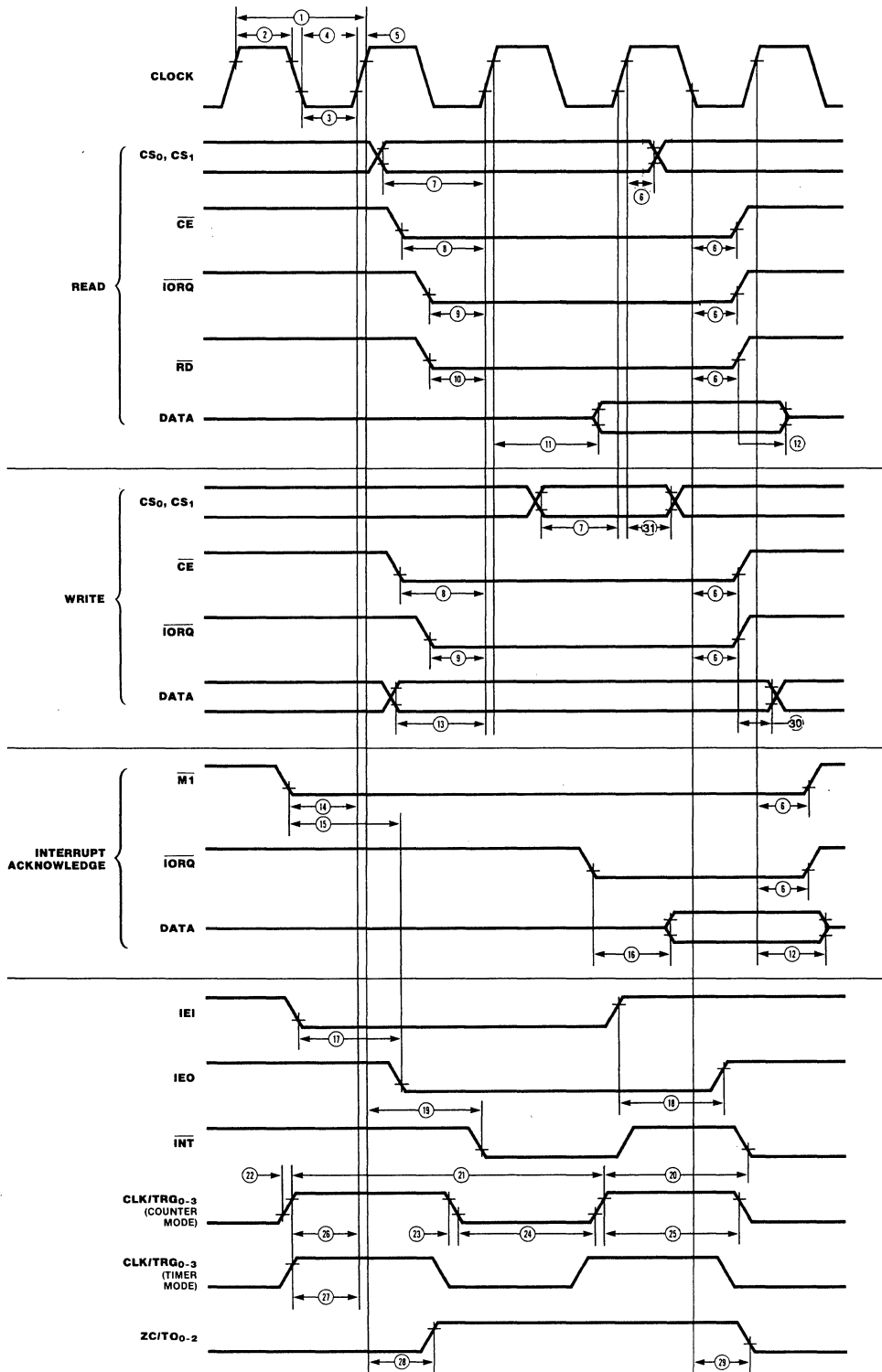
## CAPACITANCE

Symbol	Parameter	Max	Unit
CLK	Clock Capacitance	10	pf
$C_{IN}$	Input Capacitance	10	pf
$C_{OUT}$	Output Capacitance	15	pf

$T_A = 25^\circ C$ ,  $f = 1$  MHz

Unmeasured pins returned to ground.

# AC CHARACTERISTICS (Z84C30/CMOS Z80 CTC)



## AC CHARACTERISTICS (Z84C30/CMOS Z80 CTC Continued)

No	Symbol	Parameter	Z84C3004 *		Z84C3006		Z84C3008		Z84C3010		Note
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TcC	Clock Cycle Time	250	[1]	162	[1]	125	[1]	100	[1]	
2	TwCh	Clock pulse Width (High)	110	DC	65	DC	55	DC	42	DC	
3	TwCl	Clock pulse Width (Low)	110	DC	65	DC	55	DC	42	DC	
4	TfC	Clock Fall Time		30		20		10		10	
5	TrC	Clock Rise Time		30		20		10		10	
6	Th	All Hold Times	0		0		0		0		
7	TsCS(C)	/CS to Clock Rise Setup Time	160		100		50		35		
8	TsCE(C)	/CE to Clock Rise Setup Time	150		100		50		35		
9	TsIO(C)	/IORQ to Clock Rise Setup Time	115		70		40		35		
10	TsRD(C)	/RD Fall to Clock Rise Setup Time	115		70		40		35		
11	TdC(DO)	Clock Rise to Data Out Float Delay	200		130		90		90		[2]
12	TdRlr (DOz)	/RD, /IORQ rising to Data Outtime Float Delay		50		40		40		40	
13	TsDI (C)	Data In to Clock rising set-up	50		40		30		30		
14	TsM1(C)	/M1 to Clock Rise Setup Time	90		70		50		40		
15	TdM1(IEO)	/M1 Fall to IEO Fall Delay (Interrupt Immediately Preceding /M1 Fall)		190		130		90		70	[3]
16	TdIO(DIO)	/IORQ Fall to Data Out Delay (/INTACK Cycle)		160		110		80		80	[2,6]
17	TdIEI(IEOf)	IEI Fall to IEO Fall Delay		130		100		70		70	[3]
18	TdIEI(IEOr)	IEI Rise to IEO Rise Delay (After ED Decode)		160		110		70		70	[3]
19	TdC(INT)	Clock Rise to /INT Fall Delay	(TcC+140)		(TcC+120)		(TcC+100)		(TcC+80)		[4]
20	TdCLK(INT)	CLK/TRG Rise to /INT Fall Delay									
		TsCTR(C) Satisfied	(19)+(26)		(19)+(26)		(19)+(26)		(19)+(26)		[5]
		TsCTR(C) Not Satisfied	(1)+(19)+(26)		(1)+(19)+(26)		(1)+(19)+(26)		(1)+(19)+(26)		[5]
21	TcCTR	CLK/TRG Cycle Time	(2TcC)		(2TcC)		(2TcC)		(2TcC)		[5]
22	TrCTR	CLK/TRG Rise Time		50		40		30		30	
23	TfCTR	CLK/TRG Fall Time		50		40		30		30	
24	TwCTRh	CLK/TRG Width (Low)	200		120		90		90		
25	TwCTRI	CLK/TRG Width (High)	200		120		90		90		
26	TsCTR(Cs)	CLK/TRG Rise to Clock Rise Setup Time for Immediate Count	210		150		110		90		[5]
27	TsCTR(Ct)	CLK/TRG Rise to Clock Rise Setup Time for Enabling of Prescaler On Following Clock Rise	210		150		110		90		[4]

\* 4 MHz Z84C30 is obsolete and replaced by 6 MHz



---

**Z84C30 AC CHARACTERISTICS (Continued)**

---

No	Symbol	Parameter	Z84C3004*		Z84C3006		Z84C3008		Z84C3010		Note
			Min	Max	Min	Max	Min	Max	Min	Max	
28	TdC(ZC/TOr)	Clock Rise to ZC/TO Rise Delay		190		140		100		80	
29	TdC(ZC/TOf)	Clock Fall to ZC/TO Fall Delay		190		140		100		80	
30	ThRlr(D)	/CE, /IORQ Rise to Data Hold	20		20		10		10		
31	ThC(CS)	Clock Rise to /CS Hold	20		20		10		10		

---

\* RESET must be active for a minimum of 3 clock cycles.  
Units in Nanoseconds

**Notes:**

[1]  $T_{cC} = T_{wCh} + T_{wCl} + T_{rC} + T_{fC}$ .

[2] Increasing delay by 10nS for each 50pF increase in loading, 200pF max for data lines, and 100pF for control lines.

[3] Increase delay by 2nS for each 10pF increase in loading, 100pF max.

[4] Timer mode.

[5] Counter mode.

[6]  $2.5T_{cT} > (N-2)T_{dIE}(IEOf) + T_{dM1}(IEO) + T_{sIE}(IO) + TTL \text{ Buffer Delay, if any.}$

---

\* 4 MHz Z84C30 is obsolete and replaced by 6 MHz

## DC CHARACTERISTICS (Z8430/NMOS Z80 CTC)

Symbol	Parameter	Min	Max	Unit	Condition
V <sub>ILC</sub>	Clock Input Low Voltage	-0.3 <sup>c</sup>	+0.45 <sup>a</sup>	V	
V <sub>IHC</sub>	Clock Input High Voltage	V <sub>CC</sub> - 0.6 <sup>a</sup>	V <sub>CC</sub> + 0.3 <sup>b</sup>	V	
V <sub>IL</sub>	Input Low Voltage	-0.3 <sup>c</sup>	+0.8 <sup>a</sup>	V	
V <sub>IH</sub>	Input High Voltage	+2.2 <sup>a</sup>	V <sub>CC</sub> <sup>b</sup>	V	
V <sub>OL</sub>	Output Low Voltage		+0.4 <sup>a</sup>	V	I <sub>OL</sub> = 2.0 mA
V <sub>OH</sub>	Output High Voltage	+2.4 <sup>a</sup>		V	I <sub>OH</sub> = -250 μA
I <sub>CC</sub>	Power Supply Current:		+120 <sup>a</sup>	mA	
I <sub>LI</sub>	Input Leakage Current		±10 <sup>a</sup>	μA	V <sub>IN</sub> = 0.4 to V <sub>CC</sub>
I <sub>LO</sub>	3-State Output Leakage Current in Float		±10 <sup>a</sup>	μA	V <sub>OUT</sub> = 0.4 to V <sub>CC</sub>
I <sub>OHD</sub>	Darlington Drive Current	-1.5 <sup>a</sup>		mA	V <sub>OH</sub> = 1.5V R <sub>EXT</sub> = 390Ω

## CAPACITANCE

Symbol	Parameter	Max	Unit
CLK	Clock Capacitance	20 <sup>c</sup>	pf
C <sub>IN</sub>	Input Capacitance	5 <sup>c</sup>	pf
C <sub>OUT</sub>	Output Capacitance	15 <sup>c</sup>	pf

T<sub>A</sub> = 25°C, f = 1 MHz

Unmeasured pins returned to ground.

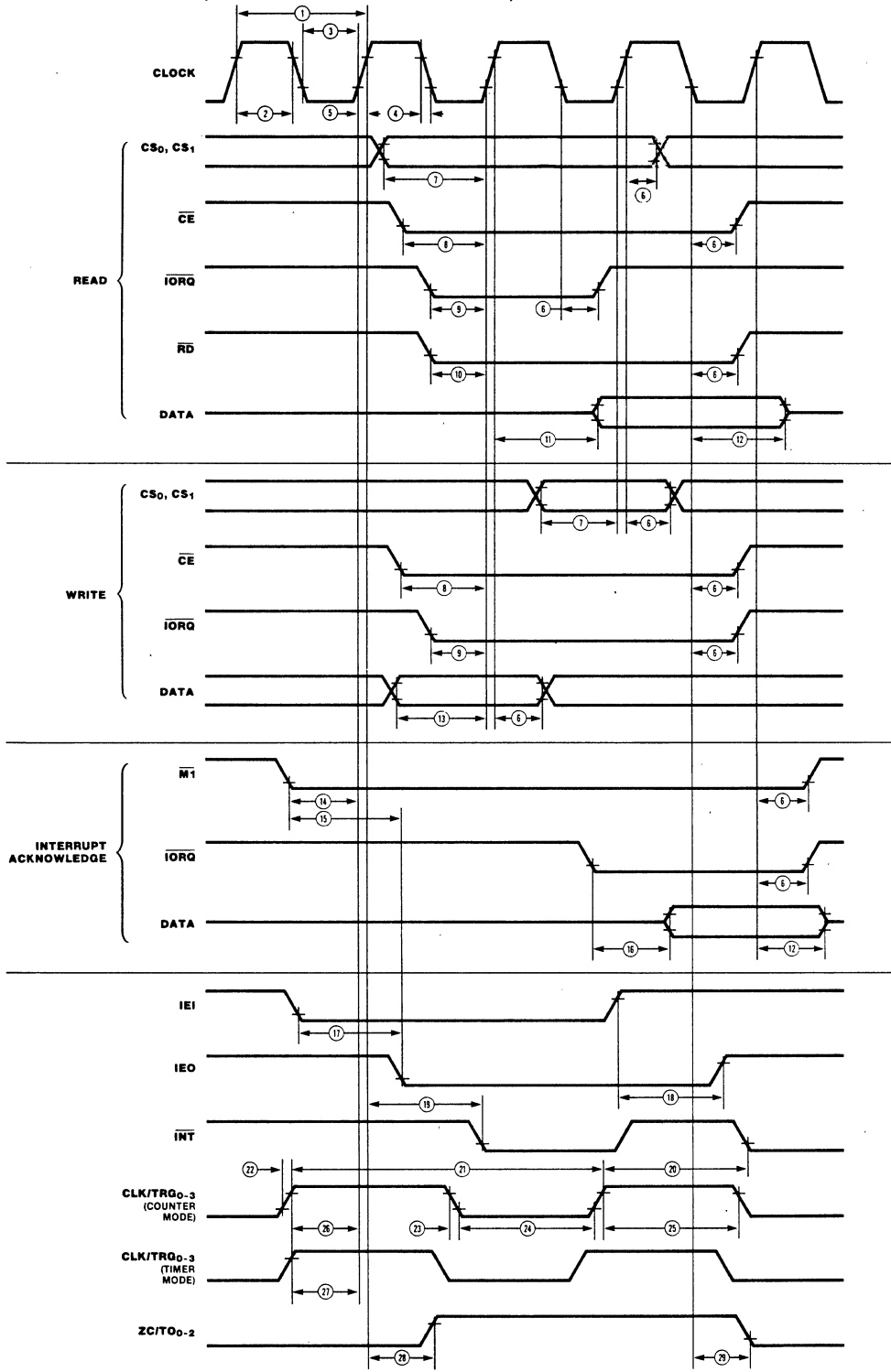
Parameter Test Status:

<sup>a</sup> Tested

<sup>b</sup> Guaranteed

<sup>c</sup> Guaranteed by characterization/design

**AC CHARACTERISTICS (Z8430/NMOS Z80 CTC Continued)**



## AC CHARACTERISTICS (Z8430/NMOS Z80 CTC)

Number	Symbol	Parameter	Z0843004		Z0843006		Notes†
			Min	Max	Min	Max	
1	TcC	Clock Cycle Time	250	[1]	162	[1]	
2	TwCh	Clock Width (High)	105	2000	65	2000	
3	TwCl	Clock Width (Low)	105	2000	65	2000	
4	TfC	Clock Fall Time		30		20	
5	TrC	Clock Rise Time		30		20	
6	Th	All Hold Times	0		0		
7	TsCS(C)	CS to Clock ↑ Setup Time	160		100		
8	TsCE(C)	$\overline{CE}$ to Clock ↑ Setup Time	150		100		
9	TsIO(C)	$\overline{IORQ}$ ↓ to Clock ↑ Setup Time	115		70		
10	TsRD(C)	$\overline{RD}$ ↓ to Clock ↑ Setup Time	115		70		
11	TdC(DO)	Clock ↑ to Data Out Delay		200		130	[2]
12	TdC(DOz)	Clock ↓ to Data Out Float Delay		110		90	
13	TsDI(C)	Data In to Clock ↑ Setup Time	50		40		
14	TsM1(C)	$\overline{M1}$ to Clock ↑ Setup Time	90		70		
15	TdM1(IEO)	$\overline{M1}$ ↓ to IEO ↓ Delay (Interrupt immediately preceding M1)		190		130	[3]
16	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)		160		110	[2]
17	TdEI(IEOf)	IEI ↓ to IEO ↓ Delay		130		100	[3]
18	TdEI(IEOr)	IEI ↑ to IEO ↑ Delay (After ED Decode)		160		110	[3]
19	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay		(1) + 140		(1) + 120	[4,6]
20	TdCLK(INT)	CLK/TRG ↑ to $\overline{INT}$ ↓ tsCTR(C) satisfied		(19) + (26)		(19) + (26)	[5,6]
		tsCTR(C) not satisfied		(1) + (19) + (26)		(1) + (19) + (26)	[5,6]
21	TcCTR	CLK/TRG Cycle Time	2TcC		2TcC		[5]
22	TrCTR	CLK/TRG Rise Time		50		40	
23	TfCTR	CLK/TRG Fall Time		50		40	
24	TwCTRI	CLK/TRG Width (Low)	200		120		
25	TwCTRh	CLK/TRG Width (High)	200		120		

### NOTES:

[1]  $TcC = TwCh + TwCl + TrC + TfC$ .

[2] Increase delay by 10 ns for each 50 pf increase in loading, 200 pf maximum for data lines, and 100 pf for control lines.

[3] Increase delay by 2 ns for each 10 pf increase in loading, 100 pf maximum.

[4] Timer mode

[5] Counter mode.

[6] Parenthetical numbers reference the table number of a parameter. e.g., (1) refers to TcC.

†  $2.5 TcC > (n - 2) TdEI(IEOf) + TdM1(IEO) + TsEI(IO) + TTL$  buffer delay, if any. **RESET** must be active for a minimum of 3 clock cycles. Units are nanoseconds unless otherwise specified.

## AC CHARACTERISTICS (Z8430/NMOS Z80 CTC Continued)

Number	Symbol	Parameter	Z0843004		Z0843006		Notes†
			Min	Max	Min	Max	
26	TsCTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for Immediate Count	210		150		[5]
27	TsCTR(Ct)	CLK/TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following clock ↑	210		150		[4]
28	TdC(ZC/TOr)	Clock ↑ to ZC/TO ↑ Delay		190		140	
29	TdC(ZC/TOf)	Clock ↓ to ZC/TO ↓ Delay		190		140	

NOTES:

- [1]  $T_{cC} = T_{wCh} + T_{wCl} + T_{rC} + T_{fC}$ .
- [2] Increase delay by 10 ns for each 50 pf increase in loading, 200 pf maximum for data lines, and 100 pf for control lines.
- [3] Increase delay by 2 ns for each 10 pf increase in loading, 100 pf maximum.
- [4] Timer mode
- [5] Counter mode.

- [6] Parenthetical numbers reference the table number of a parameter. e.g., (1) refers to  $T_{cC}$ .
- †  $2.5 T_{cC} > (n-2) TDIE(I(EO)) + TDM1(I(EO)) + TslEI(I(O)) + TTL$  buffer delay, if any.  $\overline{RESET}$  must be active for a minimum of 3 clock cycles. Units are nanoseconds unless otherwise specified.



# Z8440/1/2/4, Z84C40/1/2/3/4

## SERIAL INPUT/OUTPUT CONTROLLER

### FEATURES

- Two independent full-duplex channels, with separate control and status lines for modems or other devices.
- Data rate in the x1 clock mode of 0 to 2.0M bits/second with a 10 MHz clock.
- NMOS version for cost sensitive performance solutions, CMOS version for the designs requiring low power consumption
- NMOS Z0844x04 - 4 MHz Z0844x06 - 6.17 MHz (Where x is the designator for the bonding option; 0, 1, 2 or 4)
- CMOS Z84C4x06 - DC to 6.7 MHz, Z84C4x08 - DC to 8 MHz, Z84C4x10 - DC to 10 MHz (Where x is the designator for the bonding option; 0, 1, 2, 3 or 4)
- 6 MHz version supports 6.144 MHz CPU clock operation.
- Asynchronous protocols: everything necessary for complete messages in 5, 6, 7, or 8 bits/character. Includes variable stop bits and several clock-rate multipliers; break generation and detection; parity; overrun and framing error detection.
- Synchronous protocols: everything necessary for complete bit- or byte-oriented messages in 5, 6, 7, or 8 bits/character, including IBM Bisync, SDLC, HDLC, CCITT-X.25 and others. Automatic CRC generation/checking, sync character and zero insertion/deletion, abort generation/detection, and flag insertion.
- Receiver data registers quadruply buffered, transmitter registers doubly buffered.
- Highly sophisticated and flexible daisy-chain interrupt vectoring for interrupts without external logic.

### GENERAL DESCRIPTION

The Z80 SIO (here in after referred to as the Z80 SIO or, SIO). Serial Input/Output Controller is a dual-channel data communication interface with extraordinary versatility and capability. Its basic functions as a serial-to-parallel, parallel-to-serial converter/controller can be programmed by a CPU for a broad range of serial communication applications.

The device supports all common asynchronous and synchronous protocols, byte- or bit-oriented, and performs all of the functions traditionally done by UARTs, USARTs, and synchronous communication controllers combined, plus additional functions traditionally performed by the CPU. Moreover, it does this on two fully-independent

channels, with an exceptionally sophisticated interrupt structure that allows very fast transfers.

Full interfacing is provided for CPU or DMA control. In addition to data communication, the circuit can handle virtually all types of serial I/O with fast, or slow, peripheral devices. While designed primarily as a member of the Z80 family, its versatility makes it well suited to many other CPUs.

The Z80 SIO uses a single +5V power supply and the standard Z80 family single-phase clock. The SIO/0, SIO/1, and SIO/2 are packaged in a 40-pin DIP, the SIO/4 is packaged in a 44-pin PCC and the SIO/3 is packaged in a 44-pin QFP. Note that SIO/3 is only available in CMOS and in QFP package.

### PIN DESCRIPTION

Figures 1 through 6 illustrate the three 40-pin configurations (bonding options) available in the Z80C SIO (hereafter referred to as SIO or Z80 SIO). The constraints of a 40-pin package make it impossible to bring out the Receive Clock ( $\overline{\text{RxC}}$ ), Transmit Clock ( $\overline{\text{TxC}}$ ), Data Terminal Ready ( $\overline{\text{DTR}}$ ) and Sync ( $\overline{\text{SYNC}}$ ) signals for both channels. Therefore, either Channel B lacks a signal or two signals are bonded together:

- Z80 SIO/2 lacks  $\overline{\text{SYNCB}}$
- Z80 SIO/1 lacks  $\overline{\text{DTRB}}$

- Z80 SIO/0 has all four signals, but  $\overline{\text{TxCB}}$  and  $\overline{\text{RxCB}}$  are bonded together

The 44-pin package, the Z80 SIO/4 for PLCC package, and Z80 SIO/3 for QFP, has all options (Figure 7a and 7b).

The first bonding option above (SIO/2) is the preferred version for most applications. The pin descriptions are as follows:

**B/ $\overline{\text{A}}$ .** Channel A or B Select (input, High selects Channel B). This input defines which channel is accessed during a data

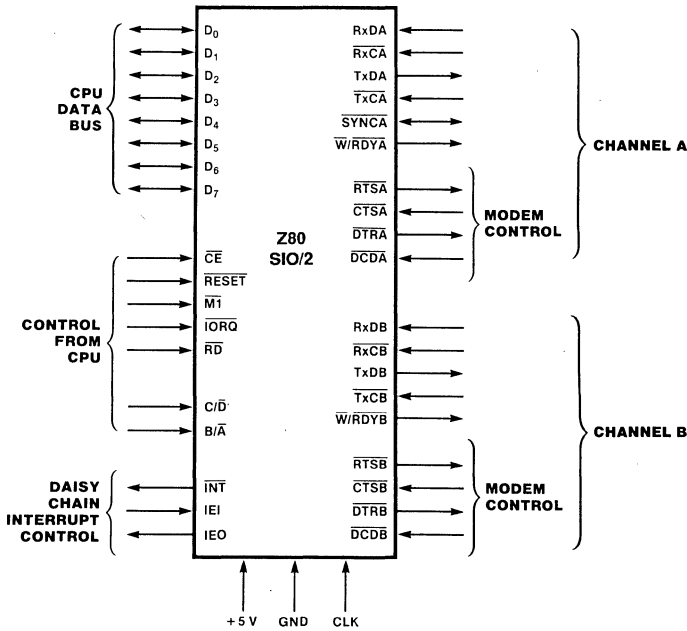


Figure 1. Pin Functions

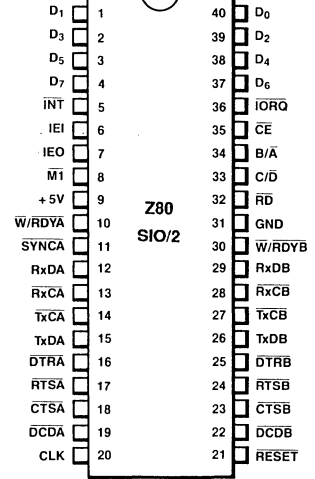


Figure 2. 40-pin Dual-In-Line Package (DIP), Pin Assignments

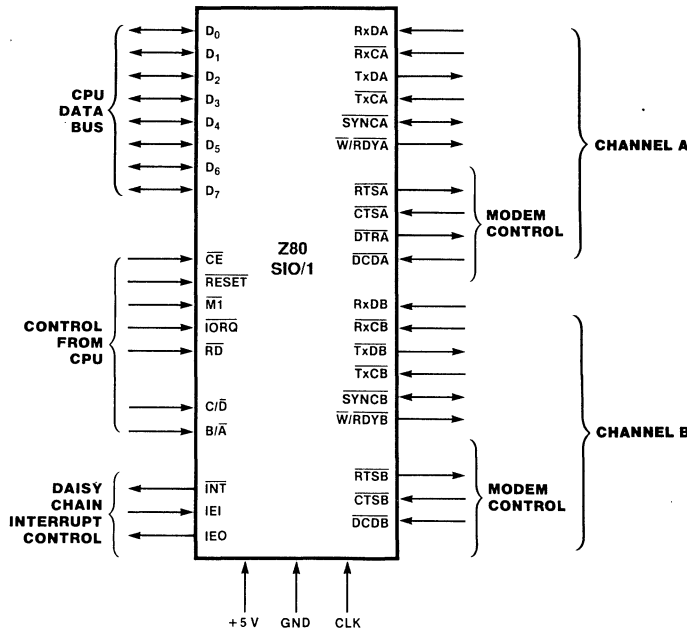


Figure 3. Pin Functions

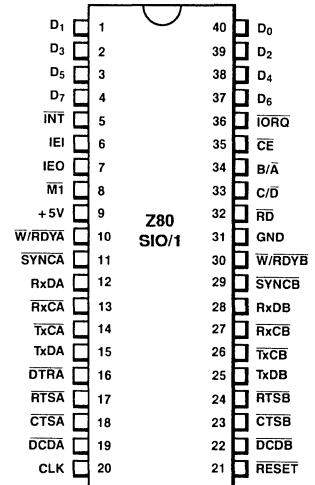


Figure 4. 40-pin Dual-In-Line Package (DIP), Pin Assignments

Note: Power connections follow conventional descriptions below:

Connection	Circuit	Device
Power	V <sub>CC</sub>	V <sub>DD</sub>
Ground	GND	V <sub>SS</sub>

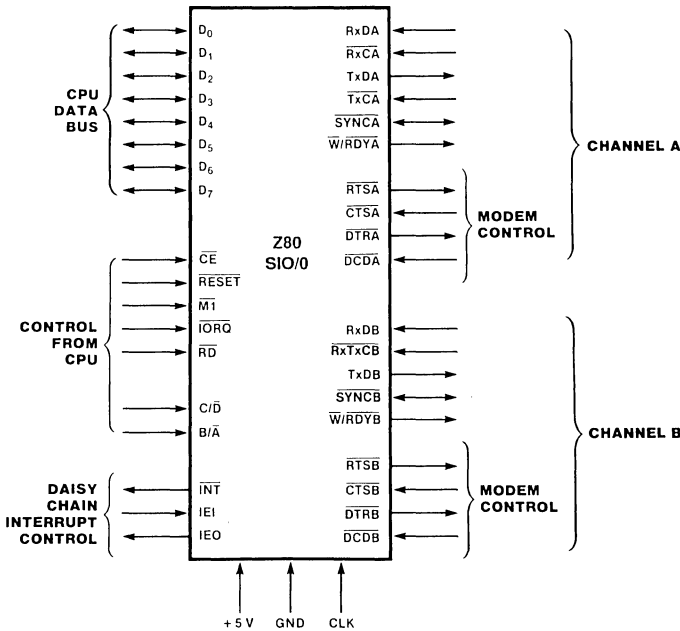


Figure 5. Pin Functions

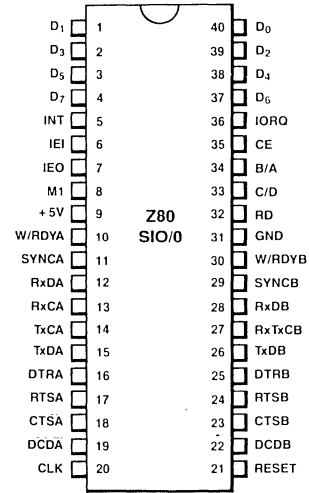


Figure 6. 40-pin Dual-In-Line Package (DIP), Pin Assignments

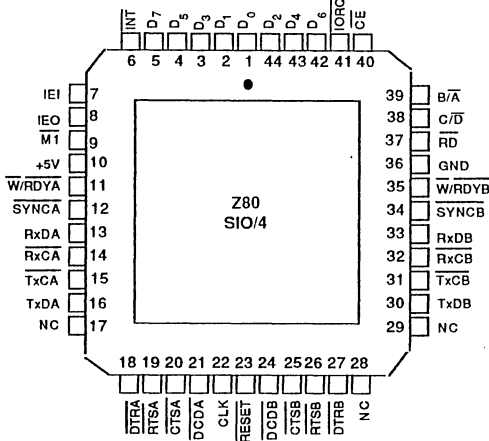


Figure 7a. 44-pin Chip Carrier, Pin Assignments

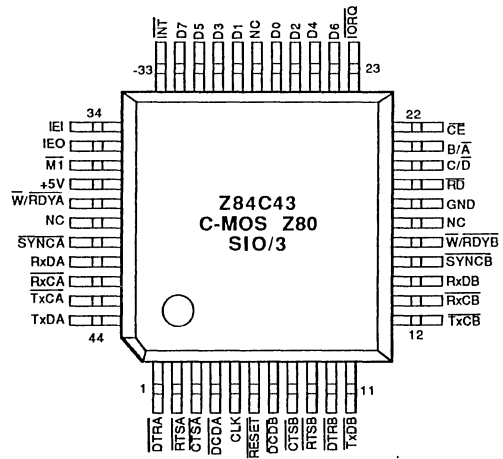


Figure 7b. 44-pin Quad Flat Pack Pin Assignments

transfer between the CPU and the SIO. Address bit  $A_0$  from the CPU is often used for the selection function.

**C/D.** *Control or Data Select* (input, High selects Control). This input defines the type of information transfer performed between the CPU and the SIO. A High at this input during a CPU write to the SIO causes the information on the data bus to be interpreted as a command for the channel selected by B/A. A Low at C/D means that the information on the data bus is data. Address bit  $A_1$  is often used for this function.

**CE.** *Chip Enable* (Input, active Low). A Low level at this input enables the SIO to accept command or data input from the CPU during a write cycle, or to transmit data to the CPU during a read cycle.

**CLK.** *System Clock* (input). The SIO uses the standard Z80 System Clock to synchronize internal signals. This is single-phase clock.



**CTS<sub>A</sub>, CTS<sub>B</sub>.** *Clear To Send* (inputs, active Low). When programmed as Auto Enables, a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow-risetime signals. The SIO detects pulses on these inputs and interrupts the CPU on both logic level transitions. The Schmitt-trigger buffering does not guarantee a specified noise-level margin.

**D<sub>0</sub>-D<sub>7</sub>.** *System Data Bus* (bidirectional, 3-state). The system data bus transfers data and commands between the CPU and the Z80 SIO. D<sub>0</sub> is the least significant bit.

**DCDA, DCDB.** *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if the SIO is programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow-risetime signals. The SIO detects pulses on these pins and interrupts the CPU on both logic level transitions. Schmitt-trigger buffering does not guarantee a specific noise-level margin.

**DTRA, DTRB.** *Data Terminal Ready* (outputs, active Low). These outputs follow the state programmed into the Z80 SIO. They can also be programmed as general-purpose outputs.

In the Z80 SIO/1 bonding option,  $\overline{\text{DTRB}}$  is omitted.

**IEI.** *Interrupt Enable In* (input, active High). This signal is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an interrupt from this SIO. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

**INT.** *Interrupt Request* (output, open drain, active Low). When the SIO is requesting an interrupt, it pulls  $\overline{\text{INT}}$  Low.

**IORQ.** *Input/Output Request* (input from CPU, active Low).  $\overline{\text{IORQ}}$  is used in conjunction with  $\overline{\text{B/A}}$ ,  $\overline{\text{C/D}}$ ,  $\overline{\text{CE}}$ , and  $\overline{\text{RD}}$  to transfer commands and data between the CPU and the SIO. When  $\overline{\text{CE}}$ ,  $\overline{\text{RD}}$ , and  $\overline{\text{IORQ}}$  are all active, the channel selected by  $\overline{\text{B/A}}$  transfers data to the CPU (a read operation). When  $\overline{\text{CE}}$  and  $\overline{\text{IORQ}}$  are active, but  $\overline{\text{RD}}$  is inactive, the channel selected by  $\overline{\text{B/A}}$  is written to by the CPU with either data or control information as specified by  $\overline{\text{C/D}}$ . As mentioned previously, if  $\overline{\text{IORQ}}$  and  $\overline{\text{M1}}$  are active simultaneously, the CPU is acknowledging an interrupt and the SIO automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

**M1.** *Machine Cycle One* (input from Z80 CPU, active Low). When  $\overline{\text{M1}}$  is active and  $\overline{\text{RD}}$  is also active, the Z80 CPU is fetching an instruction from memory; when  $\overline{\text{M1}}$  is active

while  $\overline{\text{IORQ}}$  is active, the SIO accepts  $\overline{\text{M1}}$  and  $\overline{\text{IORQ}}$  as an interrupt acknowledge if the SIO is the highest priority device that has interrupted the Z80 CPU.

**RxCA, RxCB.** *Receiver Clocks* (inputs). Receive data is sampled on the rising edge of  $\overline{\text{RxC}}$ . The Receive Clocks may be 1, 16, 32, or 64 times the data rate in asynchronous modes. These clocks may be driven by the Z80 CTC Counter Timer Circuit for programmable baud rate generation. Both inputs are Schmitt-trigger buffered; no noise level margin is specified.

In the Z80 SIO/0 bonding option,  $\overline{\text{RxCB}}$  is bonded together with  $\overline{\text{TxCB}}$ .

**RD.** *Read Cycle Status* (input from CPU, active Low). If  $\overline{\text{RD}}$  is active, a memory or I/O read operation is in progress.  $\overline{\text{RD}}$  is used with  $\overline{\text{B/A}}$ ,  $\overline{\text{CE}}$ , and  $\overline{\text{IORQ}}$  to transfer data from the SIO to the CPU.

**RxDA, RxDB.** *Receive Data* (inputs, active High). Serial data at TTL levels.

**RESET.** *Reset* (input, active Low). A Low  $\overline{\text{RESET}}$  disables both receivers and transmitters, forces  $\overline{\text{TxDA}}$  and  $\overline{\text{TxDB}}$  marking, forces the modem controls High, and disables all interrupts. The control registers must be rewritten after the SIO is reset and before data is transmitted or received.

**RTSA, RTSB.** *Request To Send* (outputs, active Low). When the RTS bit in Write Register 5 (Figure 14) is set, the  $\overline{\text{RTS}}$  output goes Low. When the RTS bit is reset in the Asynchronous mode, the output goes High after the transmitter is empty. In Synchronous modes, the  $\overline{\text{RTS}}$  pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

**SYNCA, SYNCB.** *Synchronization* (bidirectional, active Low). These pins can act either as inputs or outputs. In the asynchronous receive mode, they are inputs similar to  $\overline{\text{CTS}}$  and  $\overline{\text{DCD}}$ . In this mode, the transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0 (Figure 13), but have no other function. In the External Sync mode, these lines also act as inputs. When external synchronization is achieved,  $\overline{\text{SYNC}}$  must be driven Low on the second rising edge of  $\overline{\text{RxC}}$  after that rising edge of  $\overline{\text{RxC}}$  on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the  $\overline{\text{SYNC}}$  input. Once  $\overline{\text{SYNC}}$  is forced Low, it should be kept Low until the CPU informs the external synchronization detect logic that synchronization has been lost or a new message is about to start. Character assembly begins on the rising edge of  $\overline{\text{RxC}}$  that immediately precedes the falling edge of  $\overline{\text{SYNC}}$  in the External Sync mode.

In the internal synchronization mode (Monosync and Bisync), these pins act as outputs that are active during the part of the receive clock ( $\overline{\text{RxC}}$ ) cycle in which sync characters are recognized. The sync condition is not latched, so these outputs are active each time a sync pattern

---

is recognized, regardless of character boundaries.

In the Z80 SIO/2 bonding option,  $\overline{\text{SYNCB}}$  is omitted.

**$\overline{\text{TxCA}}$ ,  $\overline{\text{TxCB}}$ .** *Transmitter Clocks* (inputs). In asynchronous modes, the Transmitter Clocks may be 1, 16, 32, or 64 times the data rate; however, the clock multiplier must be the same for the transmitter and the receiver. The Transmit Clock inputs are Schmitt-trigger buffered for relaxed rise- and fall-time requirements; no noise level margin is specified. Transmitter Clocks may be driven by the Z80 CTC Counter Timer Circuit for programmable baud rate generation.

In the Z80 SIO/0 bonding option,  $\overline{\text{TxCB}}$  is bonded together with  $\overline{\text{RxCB}}$ .

**$\text{TxDA}$ ,  $\text{TxDB}$ .** *Transmit Data* (outputs, active High). Serial data at TTL levels.  $\text{TxD}$  changes from the falling edge of  $\overline{\text{TxC}}$ .

**$\overline{\text{W/RDYA}}$ ,  $\overline{\text{W/RDYB}}$ .** *Wait/Ready* (outputs, open drain when programmed for Wait function; driven High and Low when programmed for Ready function). These dual-purpose outputs may be programmed as Ready lines for a DMA controller or as Wait lines that synchronize the CPU to the SIO data rate. The reset state is open drain.

## FUNCTIONAL DESCRIPTION

The functional capabilities of the Z80 SIO can be described from two different points of view: as a data communications device, it transmits and receives serial data in a wide variety of data-communication protocols; as a Z80 family peripheral, it interacts with the Z80 CPU and other peripheral circuits, sharing the data, address and control buses, as well as being a part of the Z80 interrupt structure. As a peripheral to other microprocessors, the SIO offers valuable features such as non-vectored interrupts, polling, and simple handshake capability. Figure 8 is a block diagram.

Figure 9 illustrates the conventional devices that the SIO replaces.

The first part of the following discussion covers SIO data-communication capabilities; the second part describes interactions between the CPU and the SIO.

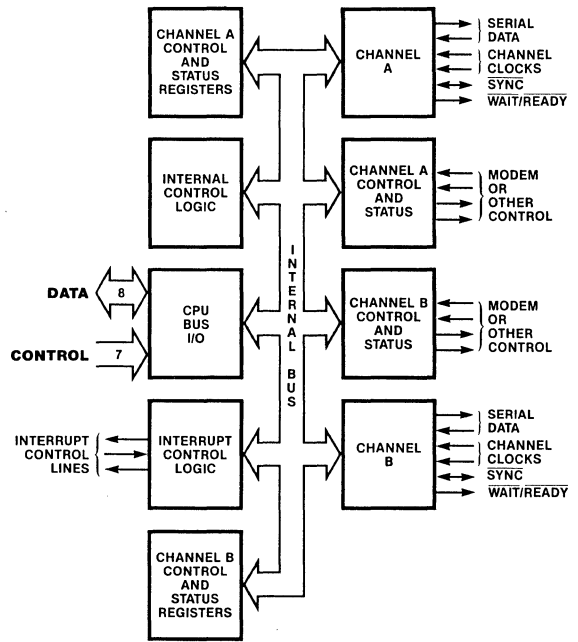


Figure 8. Block Diagram

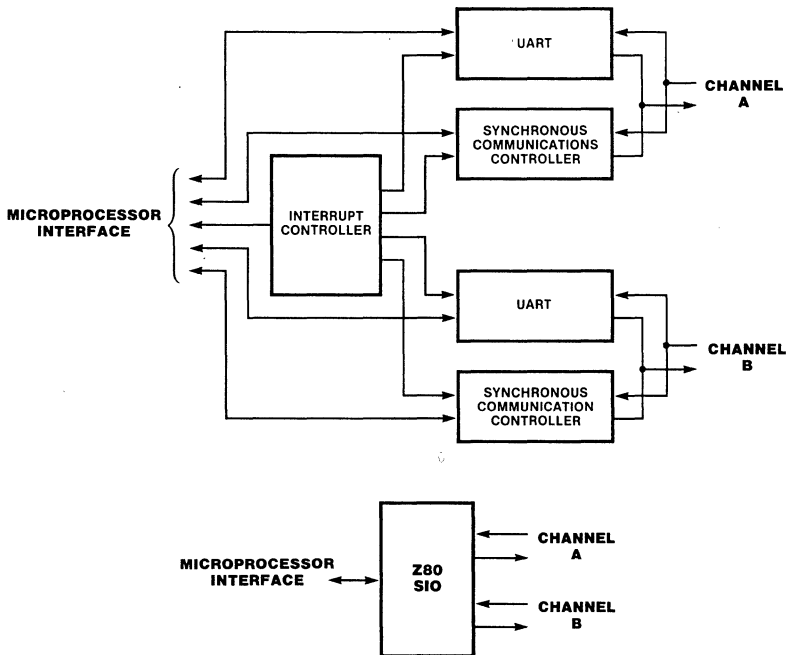


Figure 9. Conventional Devices Replaced by the Z80 SIO

## DATA COMMUNICATION CAPABILITIES

The SIO provides two independent full-duplex channels that can be programmed for use in any common asynchronous, or synchronous data-communication protocol. Figure 10a illustrates some of these protocols. The following is a short description of them. A more detailed explanation of these modes can be found in the *Z80 SIO Technical Manual* (03-3033-01).

**Asynchronous Modes.** Transmission and reception can be done independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half, or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxD A or RxD B in Figure 5). If the Low does not persist, as in the case of a transient, the character assembly process is not started.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occurred. Vectored interrupts allow fast servicing of error conditions using dedicated routines. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The SIO does not require symmetric transmit and receive clock signals, a feature that allows it to be used with a Z80 CTC or many other clock sources. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32, or 1/64 of the clock rate supplied to the receive and transmit clock inputs.

In asynchronous modes, the  $\overline{\text{SYNC}}$  pin may be programmed as an input that can be used for functions such as monitoring a ring indicator.

**Synchronous Modes.** The SIO supports both byte-oriented and bit-oriented synchronous communication.

Synchronous byte-oriented protocols can be handled in several modes that allow character synchronization with an 8-bit sync character (Monosync), any 16-bit sync pattern (Bisync), or with an external sync signal. Leading sync characters can be removed without interrupting the CPU.

Five-, six-, or seven-bit sync characters are detected with 8- or 16-bit patterns in the SIO by overlapping the larger pattern across multiple incoming sync characters, as shown in Figure 10b.

CRC checking for synchronous byte-oriented modes is delayed by one character time so the CPU may disable CRC checking on specific characters. This permits implementation of protocols such as IBM Bisync.

Figure 10a. Some Z80 SIO Protocols

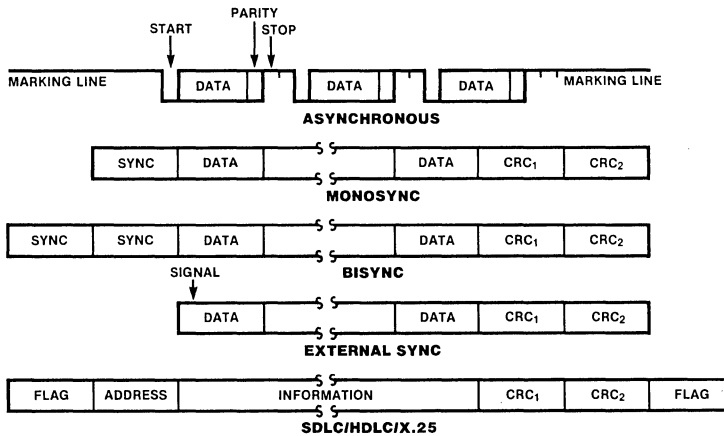


Figure 10b. Six-Bit Sync Character Recognition

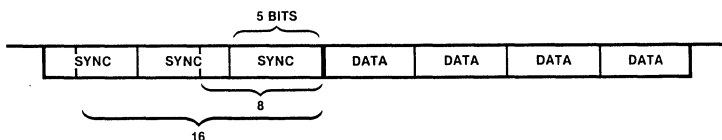


Figure 10. Data Communication

Both CRC-16 ( $X^{16} + X^{15} + X^2 + 1$ ) and CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) error checking polynomials are supported. In all non-SDLC modes, the CRC generator is initialized to 0s; in SDLC modes, it is initialized to 1s. The SIO can be used for interfacing to peripherals such as hard-sectored floppy disks, but it cannot generate or check CRC for IBM-compatible soft-sectored disks. The SIO also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows very high-speed transmissions under DMA control with no need for CPU intervention at the end of a message. When there is no data or CRC to send in synchronous modes, the transmitter inserts 8- or 16-bit sync characters regardless of the programmed character length.

The SIO supports synchronous bit-oriented protocols such as SDLC and HDLC by performing automatic flag sending, zero insertion, and CRC generation. A special command can be used to abort a frame in transmission. At the end of a message the SIO automatically transmits the CRC and trailing flag when the transmit buffer becomes empty. If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. One to eight bits per character can be sent, which allows reception of a message with no prior information about the character structure in the information field of a frame.

---

## I/O INTERFACE CAPABILITIES

The SIO offers the choice of polling, vectored or non-vectored interrupts and block-transfer modes to transfer data, status, and control information to, and from, the CPU. The block-transfer mode can also be implemented under DMA control.

**Polling.** Two status registers are updated at appropriate times for each function being performed (for example, CRC error-status valid at the end of a message). When the CPU is operated in a polling fashion, one of the SIO's two status registers is used to indicate whether the SIO has some data or needs some data. Depending on the contents of this register, the CPU will either write data, read data, or just go on. Two bits in the register indicate that a data transfer is needed. In addition, error and other conditions are indicated. The second status register (special receive conditions) does not have to be read in a polling sequence, until a character has been received. All interrupt modes are disabled when operating the device in a polled environment.

**Interrupts.** The SIO has an elaborate interrupt scheme to provide fast interrupt service in real-time applications. A control register and a status register in Channel B contain the interrupt vector. When programmed to do so, the SIO can modify three bits of the interrupt vector in the status register so that it points directly to one of eight interrupt service routines in memory, thereby servicing conditions in both channels and eliminating most of the needs for a status-analysis routine.

Transmit interrupts, receive interrupts, and external/status interrupts are the main sources of interrupts. Each interrupt

The receiver automatically synchronizes on the leading flag of a frame in SDLC or HDLC, and provides a synchronization signal on the SYNC pin; an interrupt can also be programmed. The receiver can be programmed to search for frames addressed by a single byte to only a specified user-selected address or to a global broadcast address. In this mode, frames that do not match either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For transmitting data, an interrupt on the first received character or on every character can be selected. The receiver automatically deletes all zeroes inserted by the transmitter during character assembly. It also calculates and automatically checks the CRC to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers.

The SIO can be conveniently used under DMA control to provide high-speed reception or transmission. In reception, for example, the SIO can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The SIO then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received.

source is enabled under program control, with Channel A having a higher priority than Channel B, and with receive, transmit, and external/status interrupts prioritized in that order within each channel. When the transmit interrupt is enabled, the CPU is interrupted by the transmit buffer becoming empty. (This implies that the transmitter must have had a data character written into it so it can become empty.) The receiver can interrupt the CPU in one of two ways:

- Interrupt on first received character
- Interrupt on all received characters

Interrupt-on-first-received-character is typically used with the block-transfer mode. Interrupt-on-all-received-characters has the option of modifying the interrupt vector in the event of a parity error. Both of these interrupt modes will also interrupt under special receive conditions on a character or message basis (end-of-frame interrupt in SDLC, for example). This means that the special-receive condition can cause an interrupt only if the interrupt-on-first-received-character or interrupt-on-all-received-characters mode is selected. In interrupt-on-first-received-character, an interrupt can occur from special-receive conditions (except parity error) after the first-received-character interrupt (example: receive-overflow interrupt).

The main function of the external/status interrupt is to monitor the signal transitions of the Clear To Send (CTS), Data Carrier Detect ( $\overline{DCD}$ ), and Synchronization (SYNC) pins (Figures 1 through 7). In addition, an external/status

interrupt is also caused by a CRC-sending condition, or by the detection of a break sequence (asynchronous mode) or abort sequence (SDLC mode) in the data stream. The interrupt caused by the break/abort sequence allows the SIO to interrupt when the break/abort sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the break/abort condition in external logic.

In a Z80 CPU environment (Figure 11), SIO interrupt vectoring is "automatic": the SIO passes its internally-modifiable 8-bit interrupt vector to the CPU, which adds an additional 8 bits from its interrupt-vector (I) register to form the memory address of the interrupt-routine table. This table contains the address of the beginning of the interrupt routine itself. The process entails an indirect transfer of CPU control to the interrupt routine, so that the next instruction executed after an interrupt acknowledge by the CPU is the first instruction of the interrupt routine itself.

**CPU/DMA Block Transfer.** The SIO's block-transfer mode accommodates both CPU block transfers and DMA controllers (Z80 DMA or other designs). The block-transfer mode uses the Wait/Ready output signal, which is selected with three bits in an internal control register. The Wait/Ready output signal can be programmed as a  $\overline{\text{WAIT}}$  line in the CPU block-transfer mode or as a  $\overline{\text{READY}}$  line in the DMA block-transfer mode.

To a DMA controller, the SIO  $\overline{\text{READY}}$  output indicates that the SIO is ready to transfer data to, or from, memory. To the CPU, the  $\overline{\text{WAIT}}$  output indicates that the SIO is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle.

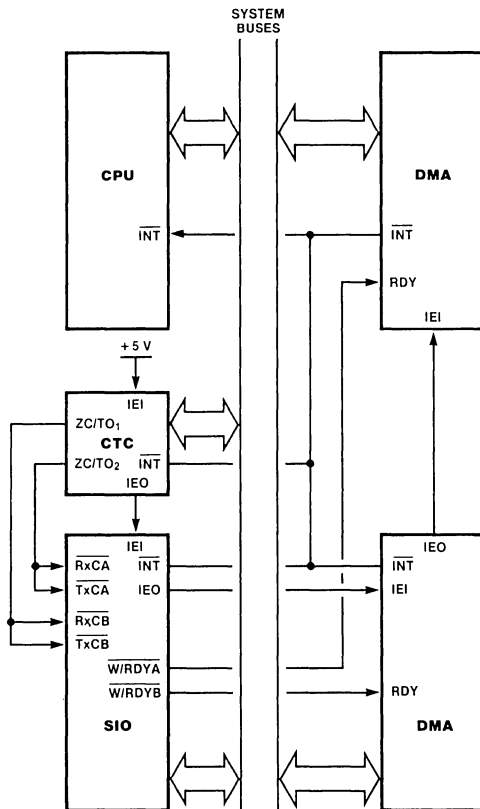


Figure 11. Typical Z80 Environment

## INTERNAL STRUCTURE

The internal structure of the device includes a Z80 CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains its own set of control and status (write and read) registers, and control and status logic that provides the interface to modems or other external devices.

The registers for each channel are designated as follows:

- WR0-WR7 — Write Registers 0 through 7
- RR0-RR2 — Read Registers 0 through 2

The register group includes five 8-bit control registers, two sync-character registers and two status registers. The interrupt vector is written into an additional 8-bit register (Write Register 2) in Channel B that may be read through another 8-bit register (Read Register 2) in Channel B. The bit assignment and functional grouping of each register is configured to simplify and organize the programming process. Table 1 lists the functions assigned to each read or write register.

The logic for both channels provides formats, synchronization, and validation for data transferred to and from the channel interface. The modem control inputs, Clear To Send (CTS) and Data Carrier Detect (DCD), are

Table 1. Register Functions

Read Register Functions	
RR0	Transmit/Receive buffer status, interrupt status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only)
Write Register Functions	
WR0	Register pointers, CRC initialize, and initialization commands for the various modes.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (Channel B only)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls
WR6	Sync character or SDLC address field
WR7	Sync character or SDLC flag

monitored by the external control and status logic under program control. All external control-and-status-logic signals are general-purpose in nature and can be used for functions other than modem control.

**Data Path.** The transmit and receive data path illustrated for Channel A in Figure 12 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement, in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high-speed data.

Incoming data is routed through one of several paths (data or CRC) depending on the selected mode and—in asynchronous modes—the character length.

The transmitter has an 8-bit transmit data buffer register that is loaded from the internal data bus, and a 20-bit transmit shift register that can be loaded from the sync-character buffers or from the transmit data register. Depending on the operational mode, outgoing data is routed through one of four main paths before it is transmitted from the Transmit Data output (TxD).

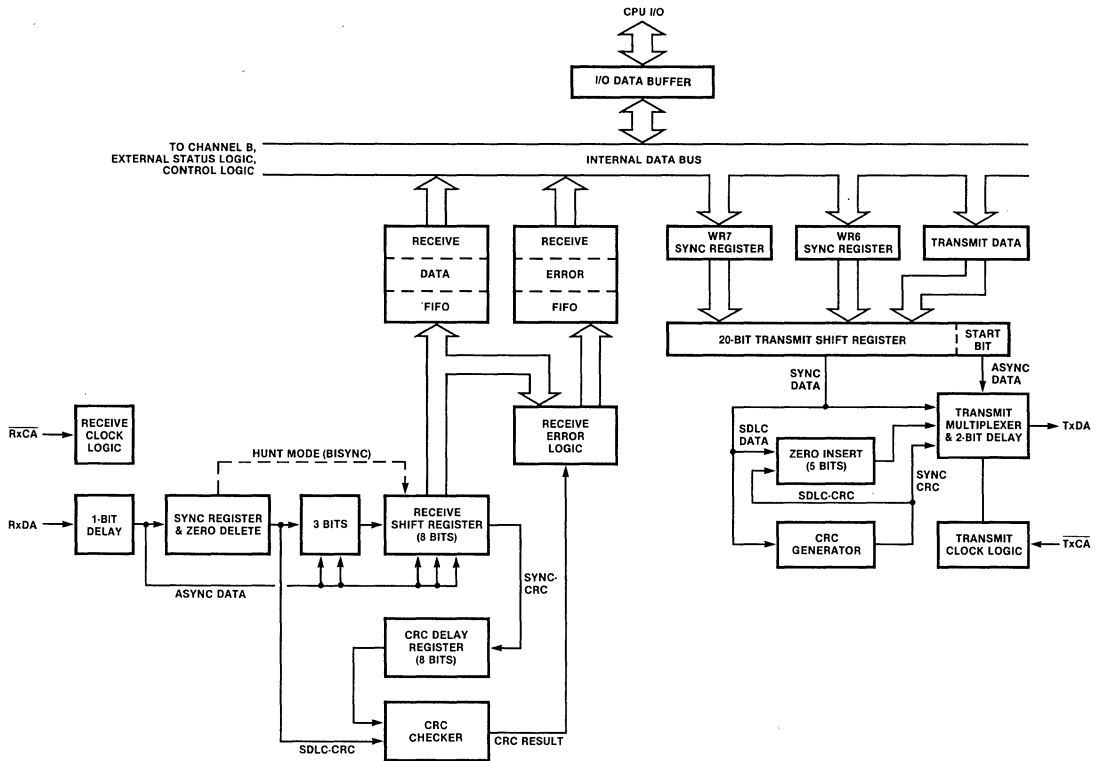


Figure 12. Transmit and Receive Data Path (Channel A)

# PROGRAMMING

The system program first issues a series of commands that initialize the basic mode of operation and then issues other commands that qualify conditions within the selected mode. For example, the asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first; then the interrupt mode; and finally, receiver or transmitter enable.

Both channels contain registers that must be programmed via the system program prior to operation. The channel-select input (B/ $\bar{A}$ ) and the control/data (C/ $\bar{D}$ ) are the command-structure addressing controls, and are normally controlled by the CPU address bus. Figures 15 and 16 illustrate the timing relationships for programming the write registers and transferring data and status.

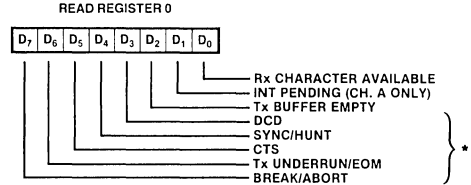
**Read Registers.** The SIO contains three read registers for Channel B and two read registers for Channel A (RR0-RR2 in Figure 13) that can be read to obtain the status information; RR2 contains the internally-modifiable interrupt vector and is only in the Channel B register set. The status information includes error conditions, interrupt vector, and standard communications-interface signals.

To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operation. Then, by executing a read instruction, the contents of the addressed read register can be read by the CPU.

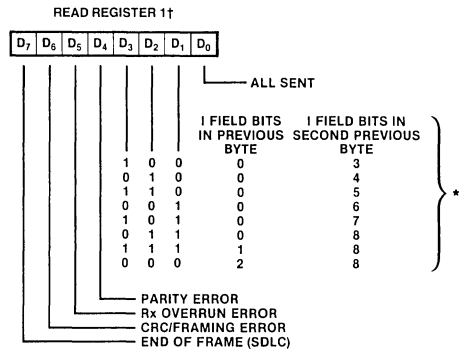
The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Condition interrupt has occurred, all the appropriate error bits can be read from a single register (RR1).

**Write Registers.** The SIO contains eight write registers for Channel B and seven write registers for Channel A (WR0-WR7 in Figure 14) that are programmed separately to configure the functional personality of the channels; WR2 contains the interrupt vector for both channels and is only in the Channel B register set. With the exception of WR0, programming the write registers requires two bytes. The first byte is to WR0 and contains three bits (D<sub>0</sub>-D<sub>2</sub>) that point to the selected register; the second byte is the actual control word that is written into the register to configure the SIO.

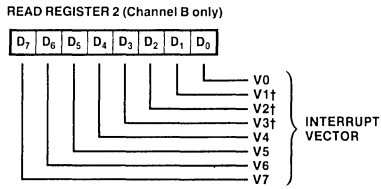
WR0 is a special case in that all of the basic commands can be written to it with a single byte. Reset (internal or external) initializes the pointer bits D<sub>0</sub>-D<sub>2</sub> to point to WR0. This implies that a channel reset must not be combined with the pointing to any register.



\* Used With "External/Status Interrupt" Modes



\*Residue data for eight Rx bits/character programmed  
†Used with special receive condition mode



†Variable if "Status Affects Vector" is programmed

Figure 13. Read Register Bit Functions



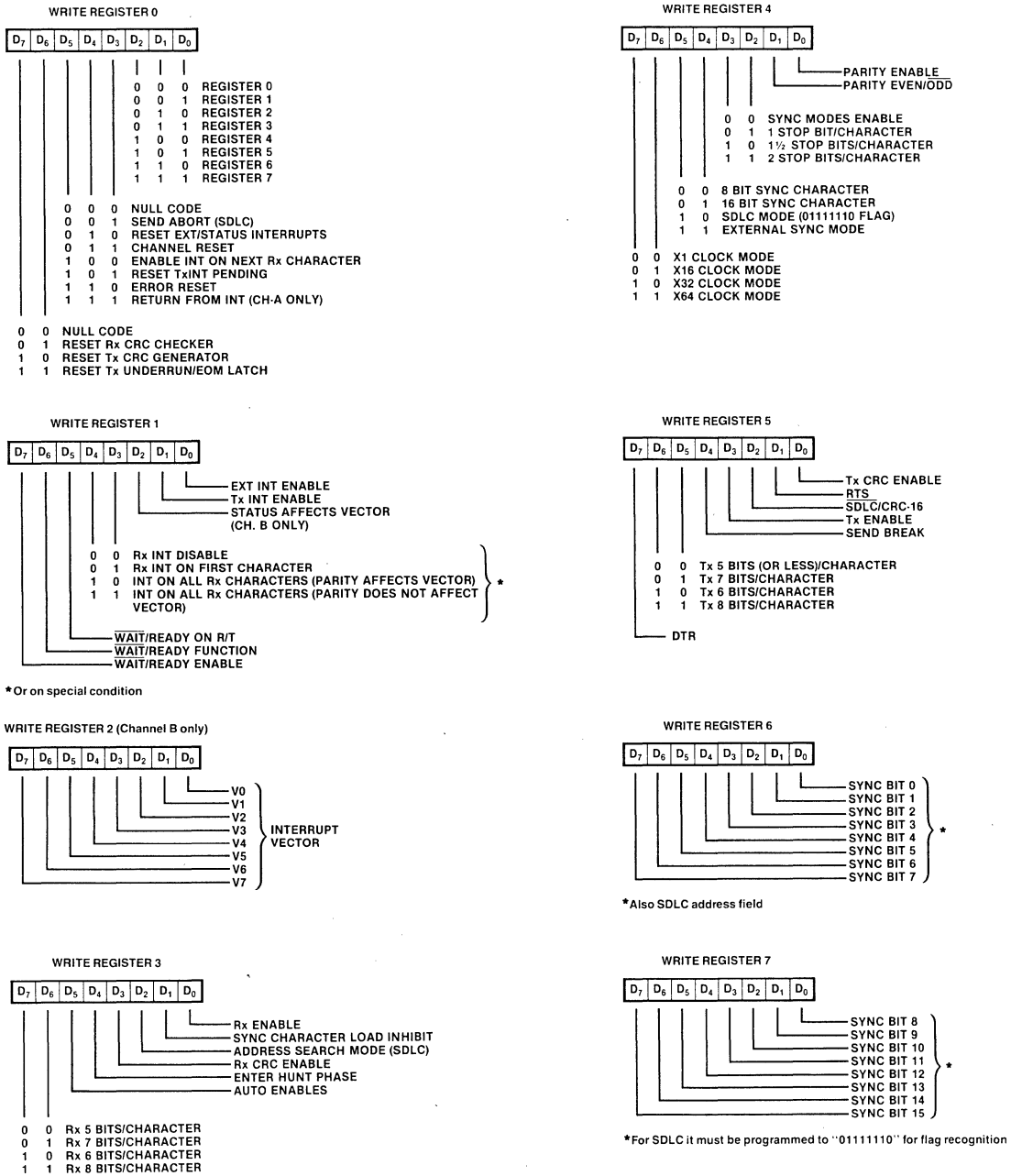


Figure 14. Write Register Bit Functions

## TIMING

The SIO must have the same clock as the CPU (same phase and frequency relationship, not necessarily the same driver).

**Read Cycle.** The timing signals generated by a Z80 CPU input instruction to read a data or status byte from the SIO are illustrated in Figure 15.

**Write Cycle.** Figure 16 illustrates the timing and data signals generated by a Z80 CPU output instruction to write a data or control byte into the SIO.

**Interrupt-Acknowledge Cycle.** After receiving an interrupt-request signal from an SIO (INT pulled Low), the Z80 CPU sends an interrupt-acknowledge sequence,  $\overline{M\overline{I}}$  Low and  $\overline{IORQ}$  Low, a few cycles later (Figure 17).

The SIO contains an internal daisy-chained interrupt structure for prioritizing nested interrupts for the various functions of its two channels, and this structure can be used within an external user-defined daisy chain that prioritizes several peripheral circuits.

The IEI of the highest-priority device is terminated High. A device that has an interrupt pending or under service forces its IEO Low. For devices with no interrupt pending or under service,  $IEO = IEI$ .

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while  $\overline{M\overline{I}}$  is Low. When  $\overline{IORQ}$  is Low, the highest priority interrupt requestor

(the one with IEI High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

**Return From Interrupt Cycle.** Figure 18 illustrates the return from interrupt cycle. Normally, the Z80 CPU issues a Return From Interrupt (RETI) instruction at the end of an interrupt service routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch in the SIO to terminate the interrupt that has just been processed. This is accomplished by manipulating the daisy chain in the following way.

The normal daisy-chain operation can be used to detect a pending interrupt; however, it cannot distinguish between an interrupt under service and a pending unacknowledged interrupt of a higher priority. Whenever ED is decoded, the daisy chain is modified by forcing High the IEO of any interrupt that has not yet been acknowledged. Thus the daisy chain identifies the device presently under service as the only one with an IEI High and an IEO Low. If the next opcode byte is 4D, the interrupt-under-service latch is reset.

The ripple time of the interrupt daisy chain (both the High-to-Low and the Low-to-High transitions) limits the number of devices that can be placed in the daisy chain. Ripple time can be improved with carry-look-ahead, or by extending the interrupt-acknowledge cycle. For further information about techniques for increasing the number of daisy-chained devices, refer to the *Z8400 Z80 CPU Product Specification* (00-2001-04).

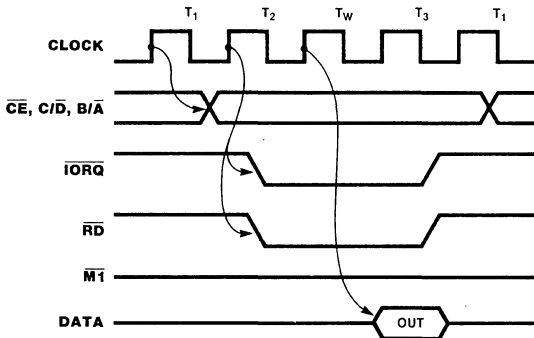


Figure 15. Read Cycle

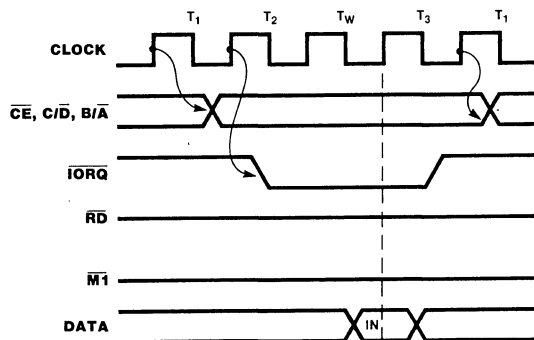


Figure 16. Write Cycle

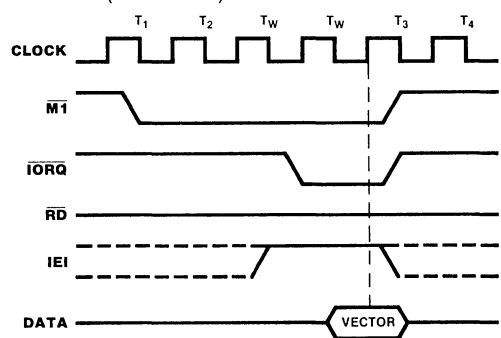


Figure 17. Interrupt Acknowledge Cycle

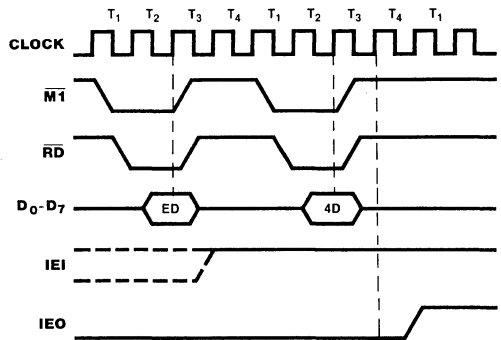


Figure 18. Return from Interrupt Cycle

---

## ABSOLUTE MAXIMUM RATINGS

Voltages in  $V_{CC}$  with respect to  $V_{SS}$  . . . . .  $-0.3V$  to  $+0.7V$   
Voltages on all inputs with respect  
to  $V_{SS}$  . . . . .  $-0.3V$  to  $V_{CC} + 0.3V$   
Storage Temperature . . . . .  $-65^{\circ}C$  to  $+150^{\circ}C$

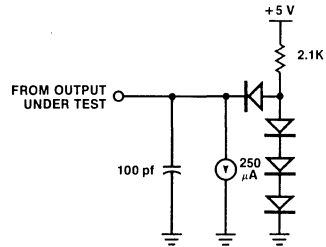
Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above these indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

---

## STANDARD TEST CONDITIONS

The characteristics below apply for the following test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin. Available operating temperature range is:

- **S =  $0^{\circ}C$  to  $+70^{\circ}C$ ,  $V_{CC}$  Range**  
    NMOS:  $+4.75V \leq V_{CC} \leq +5.25V$   
    CMOS:  $+4.50V \leq V_{CC} \leq +5.50V$
- **E =  $-40^{\circ}C$  to  $100^{\circ}C$ ,  $V_{CC} = +4.50V \leq V_{CC} \leq +5.50V$**



## DC CHARACTERISTICS

### Z84C40 CMOS Z80 SIO, Z84C40/41/42/43/44 DC CHARACTERISTICS

$V_{CC}=5.0V \pm 10\%$ , unless otherwise specified

Symbol	Parameter	Min	Max	Typ	Unit	Condition
$V_{ILC}$	Clock Input Low Voltage	-0.3	+0.45		V	
$V_{IHC}$	Clock Input High Voltage	$V_{CC}-0.6$	$V_{CC}+0.3$		V	
$V_{IH}$	Input High Voltage	2.2	$V_{CC}$		V	
$V_{IL}$	Input Low Voltage	-0.3	0.8		V	
$V_{OL}$	Output Low Voltage		0.4		V	$I_{LO}=2.0mA$
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH}=-1.6mA$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$			V	$I_{OH}=-250\mu A$
$I_{LI}$	Input Leakage Current	-10	10		$\mu A$	$V_{IN}=0.4V$ to $V_{CC}$
$I_{LO}$	3-state Output Leakage Current in Float	-10	10		$\mu A$	$V_{OUT}=0.4V$ to $V_{CC}$
$I_{L(SY)}$	SYNC Pin Leakage Current	-40	10		$\mu A$	
$I_{CC1}$	Power Supply Current - 4MHz		10 [1]	7	mA	$V_{CC}=5V$
		- 6MHz		7	mA	CLK=4,6,8,10MHz
		- 8MHz	12 [1]	8	mA	$V_{IH}=V_{CC}-0.2V$
		- 10MHz	15 [1]	8	mA	$V_{IL}=0.2V$
$I_{CC2}$	Standby Supply Current		10		$\mu A$	$V_{CC}=5V$ CLK=(0) $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$

**Note:**

[1] Measurements made with outputs floating.

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C	Clock Capacitance		7	pf
$C_{IN}$	Input Capacitance		5	pf
$C_{OUT}$	Output Capacitance		10	pf

Over specified temperature range; f = 1 MHz.

Unmeasured pins returned to ground.

## AC CHARACTERISTICS\*

### Z84C40/41/42/43/44 AC CHARACTERISTICS

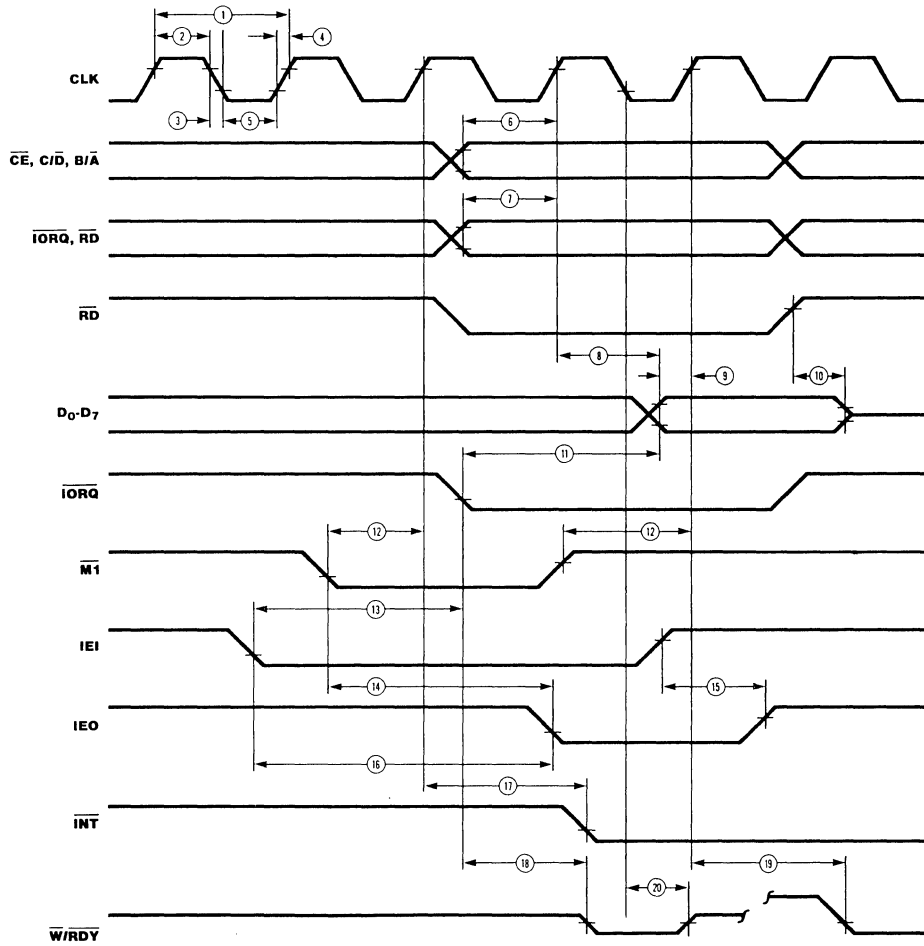
No	Symbol	Parameter	Z84C4X04*		Z84C4X06		Z84C4X08		Z84C4X10		Note
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TcC	Clock Cycle Time	250	DC	162	DC	125	DC	100	DC	
2	TwCh	Clock Pulse Width (High)	105	DC	65	DC	55	DC	42	DC	
3	TfC	Clock Fall Time		30		20		10		10	
4	TrC	Clock Rise Time		30		20		10		10	
5	TwCl	Clock Pulse Width (Low)	105	DC	65	DC	55	DC	42	DC	
6	TsAD	/CE, B//A, C//D to Clock Rise Setup Time	145		60		40		35		
7	TsCS(C)	/IORQ, /RD to Clock Rise	115		60		40		35		
8	TdC(DO)	Clock Rise to Data Out Delay		220		150		100		85	
9	TsDI(C)	Data In to Clock Rise Setup Time (Write or /M1 Cycle)	50		30		20		20		
10	TdRD(DOz)	/RD Rise to Data Out Float Delay		110		90		75		65	
11	TdIO(DOI)	/IORQ Fall to Data Out Delay (/INTACK Cycle)		160		120		90		80	
12	TsM1(C)	/M1 to Clock Rise Setup Time	90		75		55		40		
13	TsIEI(IO)	IEI to /IORQ Fall Setup Time (/INTACK Cycle)	140		120		80		60		
14	TdM1(IEO)	M1 Fall to IEO Fall Delay (Interrupt Before /M1)		190		160		130		100	
15	TdIEI(IEOr)	IEI Rise to IEO Rise Delay (After ED Decode)		100		70		60		50	
16	TdIEI(IEOf)	/M1 Fall to IEO Fall Delay		100		70		60		50	
17	TdC(INT)	Clock Rise to /INT Fall Delay		200		150		120		100	
18	TdIO(W/RWf)	/IORQ or /CE Fall to /W//RDY Delay (Wait Mode)		210		175		130		110	
19	TdC(W/RR)	Clock Rise to /W//RDY Delay (Ready Mode)	120		100		90		85		
20	TdC(W/RWz)	Clock Fall to /W//RDY Float Delay (Wait Mode) When Setup is Specified		130		110		90		80	
21	Th	Any Unspecified Hold	0		0		0		0		

**Note:**

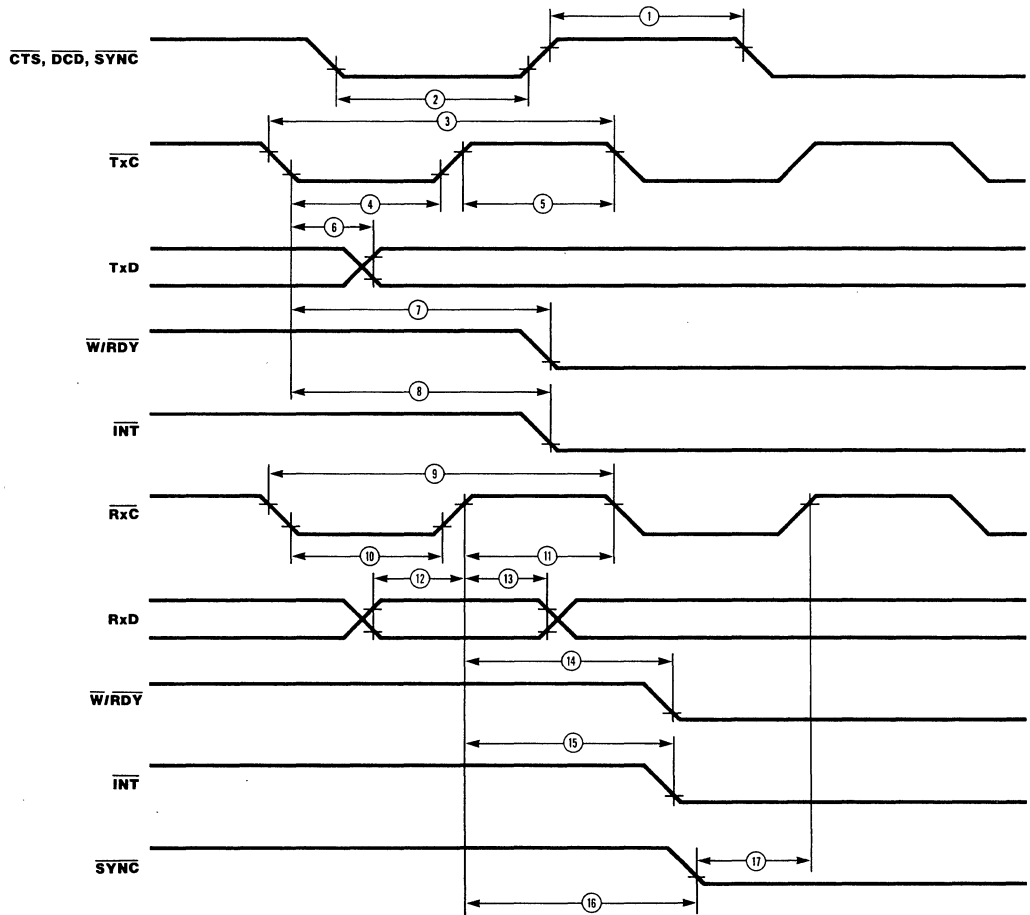
\* Units in nanoseconds (nS).

\* 4 MHz 84C4x is obsoleted and replaced by 6 MHz.

**AC CHARACTERISTICS TIMING (Z84C4X CMOS Z80 SIO)**



**AC CHARACTERISTICS TIMING** (Z84C4X CMOS Z80 SIO; Continued)



## AC CHARACTERISTICS (Z84C4X CMOS Z80 SIO; Continued)

### Z84C40/41/42/43/44 AC CHARACTERISTICS

No	Symbol	Parameter	Z84C4X04*		Z84C4X06		Z84C4X08		Z84C4X10		Note
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TwPh	Pulse Width (High)	200		200		150		150		[2]
2	TwPl	Pulse Width (Low)	200		200		150		150		[2]
3	TcTxC	/TxC Cycle Time	400		330		250		200		[2]
4	TwTxCl	/TxC Width (Low)	180		100		85		80		[2]
5	TwTxCh	/TxC Width (High)	180		100		85		80		[2]
6	TdTxC(TxD)	/TxC Fall to TxD Delay		300		220		160		120	[2]
7	TdTxC(W/RRf)	/TxC Fall to /W//RDY Fall Delay (Ready Mode)	5	9	5	9	5	9	5	9	[1]
8	TdTxC(INT)	/TxC Fall to /INT Fall Delay	5	9	5	9	5	9	5	9	[1]
9	TcRxC	/RxC Cycle Time	400		330		250		200		[2]
10	TwRxCl	/RxC Width (Low)	180		100		85		80		[2]
11	TwRxCh	/RxC Width (High)	180		100		85		80		[2]
12	TsRxD(RxC)	RxD to /RxC Setup Time (X1 Mode)	0		0		0		0		[2]
13	ThRxD(RxC)	/RxC Rise to RxD Hold Time (X1 Mode)	140		100		80		60		[2]
14	TdRxC(W/RRf)	/RxC Rise to /W//RDY Fall Delay (Ready Mode)	10	13	10	13	10	13	10	13	[1]
15	TdRxC(INT)	/RxC Rise to /INT Fall Delay	10	13	10	13	10	13	10	13	[1]
16	TdRxC(SYNC)	/RxC Rise to /SYNC Fall Delay (Output Modes)	4	7	4	7	4	7	4	7	[1]
17	TsSYNC(RxC)	/SYNC Fall to /RxC Rise Setup (External Sync Modes)	-100		-100		-100		-100		[2]

\* In All Modes, the System Clock rate must be at least five times the maximum data rate.  
/RESET must be active a minimum of one complete clock cycle.

#### Notes:

[1] Units equal to System Clock Periods.

[2] Units in nanoseconds (nS).

\* 4 MHz 84C4x is obsolete and replaced by 6 MHz.



## DC CHARACTERISTICS (Z844X / NMOS Z80 SIO)

Symbol	Parameter	Min	Max	Unit	Test Condition
V <sub>IILC</sub>	Clock Input Low Voltage	-0.3	+0.45	V	
V <sub>IHC</sub>	Clock Input High Voltage	V <sub>CC</sub> - 0.6	V <sub>CC</sub> + 0.3	V	
V <sub>IL</sub>	Input Low Voltage	-0.3	+0.8	V	
V <sub>IH</sub>	Input High Voltage	+2.0	V <sub>CC</sub>	V	
V <sub>OL</sub>	Output Low Voltage		+0.4	V	
V <sub>OH1</sub>	Output High Voltage	+2.4		V	I <sub>OL</sub> = 2.0 mA
V <sub>OH2</sub>	Output High Voltage			V	I <sub>OH</sub> = -250 μA
I <sub>LI</sub>	Input Leakage Current		± 10	μA	V <sub>IN</sub> = 0.4 to V <sub>CC</sub>
I <sub>LO</sub>	3-State Output Leakage Current in Float		± 10	μA	V <sub>OUT</sub> = 0.4 to V <sub>CC</sub>
I <sub>L(SY)</sub>	SYNC Pin Leakage Current		+ 10 / - 40	μA	0 < V <sub>IN</sub> < V <sub>CC</sub>
ICC <sub>1</sub>	Power Supply Current		100	mA	

Over specified temperature and voltage range.

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C	Clock Capacitance		40	pf
C <sub>IN</sub>	Input Capacitance		5	pf
C <sub>OUT</sub>	Output Capacitance		15	pf

Over specified temperature range; f = 1 MHz.

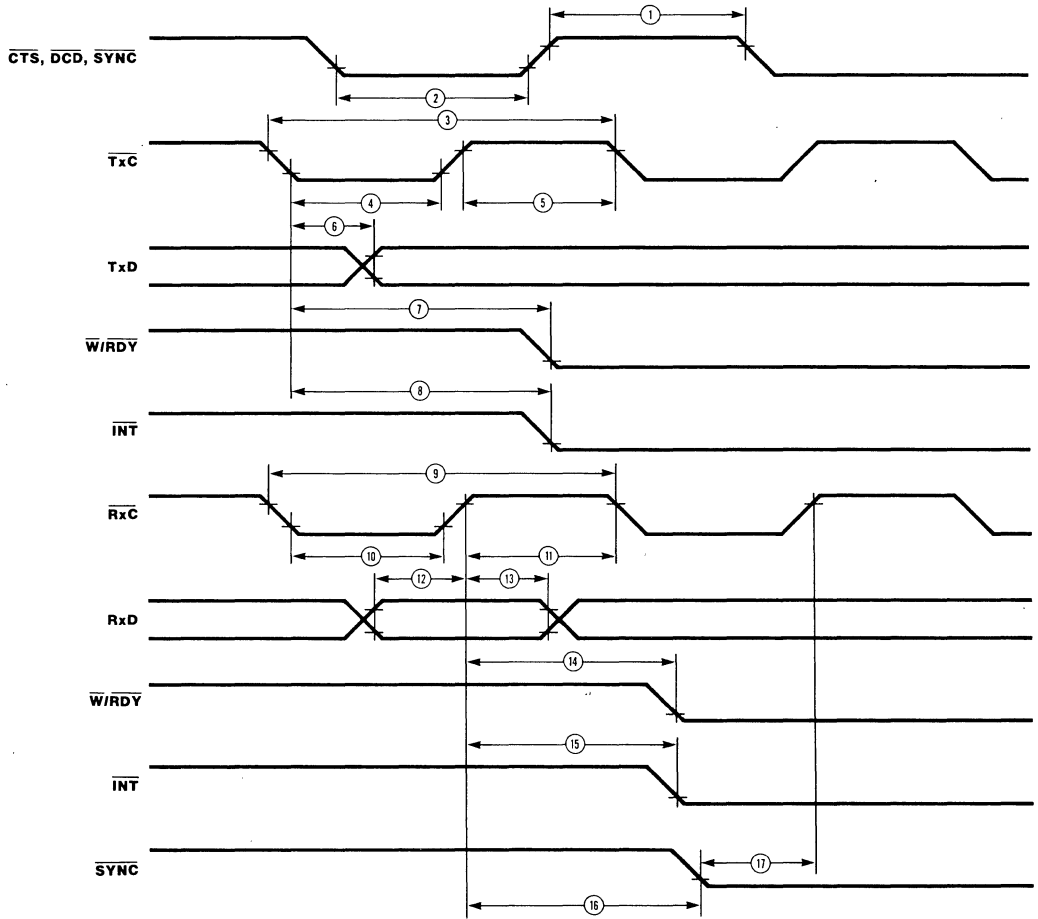
Unmeasured pins returned to ground.

## AC CHARACTERISTICS\* (Z844X / NMOS Z80 SIO)

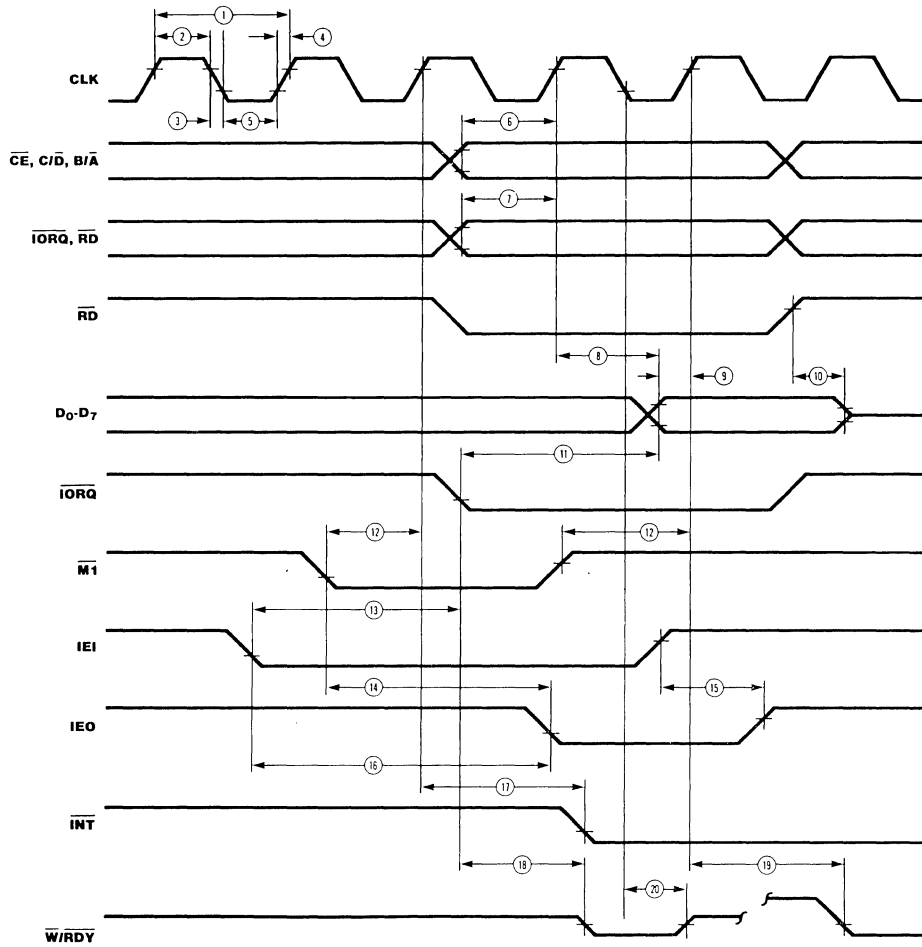
Number	Symbol	Parameter	Z0844X04		Z0844X06	
			Min	Max	Min	Max
1	TcC	Clock Cycle Time	250	4000	162	4000
2	TwCh	Clock Width (High)	105	2000	70	2000
3	TfC	Clock Fall Time		30		15
4	TrC	Clock Rise Time		30		15
5	TwCl	Clock Width (Low)	105	2000	70	2000
6	TsAD(C)	$\overline{CE}$ , $C/\overline{D}$ , $B/\overline{A}$ to Clock $\uparrow$ Setup Time	145		60	
7	TsCS(C)	$\overline{IORQ}$ , $\overline{RD}$ to Clock $\uparrow$ Setup Time	115		60	
8	TdC(DO)	Clock $\uparrow$ to Data Out Delay		220		150
9	TsDI(C)	Data In to Clock $\uparrow$ Setup (Write or $\overline{M1}$ Cycle)	50		30	
10	TdRD(DOz)	$\overline{RD}$ $\uparrow$ to Data Out Float Delay		110		90
11	TdIO(DOI)	$\overline{IORQ}$ $\downarrow$ to Data Out Delay (INTACK Cycle)		160		120
12	TsM1(C)	$\overline{M1}$ to Clock $\uparrow$ Setup Time	90		75	
13	TsIEI(IO)	IEI to $\overline{IORQ}$ $\downarrow$ Setup Time (INTACK Cycle)	140		120	
14	TdM1(IEO)	$\overline{M1}$ $\downarrow$ to IEO $\downarrow$ Delay (interrupt before $\overline{M1}$ )		190		160
15	TdIEI(IEOr)	IEI $\uparrow$ to IEO $\uparrow$ Delay (after ED decode)		100		70
16	TdIEI(IEOf)	IEI $\downarrow$ to IEO $\downarrow$ Delay		100		70
17	TdC(INT)	Clock $\uparrow$ to $\overline{INT}$ $\downarrow$ Delay		200		150
18	TdIO(W/RWf)	$\overline{IORQ}$ $\downarrow$ or $\overline{CE}$ $\downarrow$ to $\overline{W/RDY}$ $\downarrow$ Delay (Wait Mode)		210		175
19	TdC(W/RRf)	Clock $\uparrow$ to $\overline{W/RDY}$ $\downarrow$ Delay (Ready Mode)		120		100
20	TdC(W/RWz)	Clock $\downarrow$ to $\overline{W/RDY}$ Float Delay (Wait Mode)		130		110
21	Th	Any unspecified Hold when Setup is specified	0		0	

\*Units in nanoseconds (ns).

**AC CHARACTERISTICS TIMING (Z844X / NMOS Z80 SIO; Continued)**



AC CHARACTERISTICS TIMING (Z844X / NMOS Z80 SIO)



## AC CHARACTERISTICS (Z844X / NMOS Z80 SIO; Continued)

No.	Symbol	Parameter	Z0844X04		Z0844X06		Notes*
			Min	Max	Min	Max	
1	TwPh	Pulse Width (High)	200		200		2
2	TwPl	Pulse Width (Low)	200		200		2
3	TcTxC	$\overline{\text{Tx}}\overline{\text{C}}$ Cycle Time	400	$\infty$	330	$\infty$	2
4	TwTxCl	$\overline{\text{Tx}}\overline{\text{C}}$ Width (Low)	180	$\infty$	100	$\infty$	2
5	TwTxCh	$\overline{\text{Tx}}\overline{\text{C}}$ Width (High)	180	$\infty$	100	$\infty$	2
6	TdTxC(TxD)	$\overline{\text{Tx}}\overline{\text{C}}$ ↓ to TxD Delay		300		220	2
7	TdTxC(W/RRf)	$\overline{\text{Tx}}\overline{\text{C}}$ ↓ to $\overline{\text{W}}/\overline{\text{RD}}\overline{\text{Y}}$ ↓ Delay (Ready Mode)	5	9	5	9	1
8	TdTxC(INT)	$\overline{\text{Tx}}\overline{\text{C}}$ ↓ to $\overline{\text{INT}}$ ↓ Delay	5	9	5	9	1
9	TcRxC	$\overline{\text{RxC}}$ Cycle Time	400	$\infty$	330	$\infty$	2
10	TwRxCl	$\overline{\text{RxC}}$ Width (Low)	180	$\infty$	100	$\infty$	2
11	TwRxCh	$\overline{\text{RxC}}$ Width (High)	180	$\infty$	100	$\infty$	2
12	TsRxD(RxC)	RxD to $\overline{\text{RxC}}$ ↑ Setup Time (x1 Mode)	0		0		2
13	ThRxD(RxC)	$\overline{\text{RxC}}$ ↑ RxD Hold Time (x1 Mode)	140		100		2
14	TdRxC(W/RRf)	$\overline{\text{RxC}}$ ↑ to $\overline{\text{W}}/\overline{\text{RD}}\overline{\text{Y}}$ ↓ Delay (Ready Mode)	10	13	10	13	1
15	TdRxC(INT)	$\overline{\text{RxC}}$ ↑ to $\overline{\text{INT}}$ ↓ Delay	10	13	10	13	1
16	TdRxC(SYNC)	$\overline{\text{RxC}}$ ↑ to $\overline{\text{SYNC}}$ ↓ Delay (Output Modes)	4	7	4	7	1
17	TsSYNC(RxC)	$\overline{\text{SYNC}}$ ↓ to $\overline{\text{RxC}}$ ↑ Setup (External Sync Modes)	-100		-100		2

\*In all modes, the System Clock rate must be at least five times the maximum data rate.  $\overline{\text{RESET}}$  must be active a minimum of one complete clock cycle.

1. Units equal to System Clock Periods.

2. Units in nanoseconds (ns).

# EMBEDDED CONTROLLERS

---

## Z84C01 Z80® CPU with Clock Generator/Controller

---

### FEATURES:

- Commands compatible with the Zilog Z80 MPU
- Low power consumption
  - 40mA Typ (5V, 10 MHz under RUN mode)
  - 2mA Typ (5V, 10 MHz under IDLE1 mode)
  - 10mA Typ (5V, 10 MHz under IDLE2 mode)
  - 5  $\mu$ A Typ (5V under STOP mode)
- DC to 10 MHz operation (at 5V $\pm$ 10%)
- Single 5V power supply (at 5V $\pm$ 10%)
- Operating temperature (0° C to 70° C)
- On-chip clock generator

In the HALT state, the following 4 modes are selectable:

- RUN mode
- IDLE 1 mode
- IDLE 2 mode
- STOP mode

- Powerful set of 158 instructions
- Powerful interrupt function
  - Non-maskable interrupt terminal ( $\overline{\text{NMI}}$ )
  - Maskable interrupt terminal ( $\overline{\text{INT}}$ )

The following three modes are selectable:

- 8080 compatible interrupt mode (interrupt by Non-Z80 family peripheral LSI) (Mode 0)
- Restart interrupt (Mode 1)
- Daisy-chain structure interrupt using Z80 family peripheral LSI (Mode 2)
- An auxiliary register provided to each of general purpose registers.
- 2 index registers
  - 10 addressing modes
- Built-in refresh circuit for dynamic memory
- 44-Pin PLCC or QFP Package

---

### GENERAL DESCRIPTION:

The Z84C01 is an 8-bit microprocessor (hereinafter referred to as MPU) with a built-in clock generator/controller, which provides low power operation and high performance.

Built into the Z84C01 is a control function and clock generator for the standby function in addition to: six paired general purpose registers, accumulator, flag registers, an arithmetic-and-logic unit, bus control, memory control and timing control circuits.

The Z84C01 is fabricated with Zilog CMOS technology and molded in a 44-pin PLCC or QFP packages.

Further, in the following text and explanations for charts and tables, hexadecimal numbers are directly used without giving an identification to explanation of address, etc. so as not to cause confusions.



## PIN CONNECTIONS AND PIN FUNCTIONS:

The pin connections and I/O pin names and brief functions of the Z84C01 are shown below.

**Pin Names and Functions.** I/O pin names and functions are as shown in Table 1.

**Pin Connections.** The pin connections of the Z84C01 are as shown in Fig. 1.

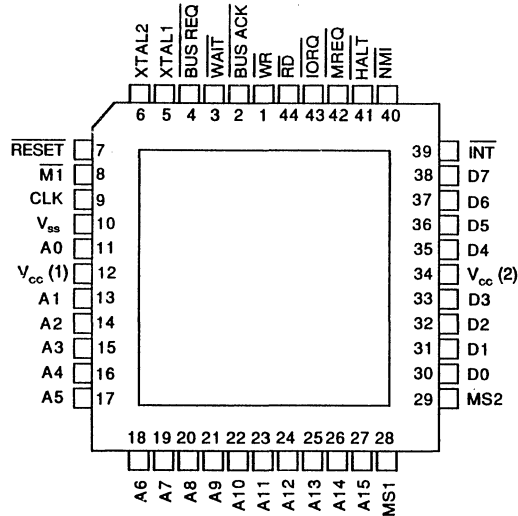


Figure 1. Pin Connections (Top View)

Table 1 Pin Names and Functions

Pin Name	Number of Pin	Input/Output 3-state	Function
A0 - A15	16	Output 3-state	16-bit address bus. Specify addresses of memories and I/O to be accessed. During the refresh period, addresses for refreshing are output.
MS1, MS2	2	Input	Mode selection input. One of 4 modes (Run, IDLE1/2, STOP) is selected according to the state of these 2 pins.
D0 - D7	8	I/O 3-state	8-bit bidirectional data bus.
$\overline{\text{INT}}$	1	Input	Maskable interrupt request signal. Interrupt is generated by peripheral LSI. This signal is accepted if the interrupt enable flip-flop (IFF) is set at "1". $\overline{\text{INT}}$ is normally wired-OR and requires an external pull up for these applications.

**Table 1 Pin Names and Functions (continued)**

Pin Name	Number of Pin	Input/Output 3-state	Function
$\overline{\text{NMI}}$	1	Input	Non-maskable interrupt request signal. This interrupt request has the higher priority than the maskable interrupt request and does not rely upon the state of the interrupt enable flip-flop (IFF).
$\overline{\text{HALT}}$	1	Output	Halt signal. Indicates that the CPU has executed a Halt instruction.
$\overline{\text{MREQ}}$	1	Output 3-state	Memory request signal. When an effective address for memory access is on the address bus, "0" is output.
$\overline{\text{IORQ}}$	1	Output 3-state	I/O request signal. When addresses for I/O are on lower 8 bits (A0 - A7) of the address bus in the I/O operation, "0" is output. In addition, $\overline{\text{IORQ}}$ signal is output together with $\overline{\text{M1}}$ signal at time of interrupt acknowledge cycle to inform peripheral LSI of the state that the interrupt response vector may be put on the data bus.
$\overline{\text{RD}}$	1	Output 3-state	Read signal. "0" signal is output for a period when MPU can receive data from a memory or peripheral LSI. It is possible to put data from a specified peripheral LSI or mamory on the MPU data bus after gating by this signal.
$\overline{\text{WR}}$	1	Output 3-state	Write signal. This signal is output when data to be stored in a specified memory or peripheral LSI is on the MPU data bus.
$\overline{\text{BUSACK}}$	1	Output	Bus acknowledge signal. In response to $\overline{\text{BUSREQ}}$ signal, this signal informs a peripheral LSI of the fact that the address bus, data bus, $\overline{\text{MREQ}}$ , $\overline{\text{IORQ}}$ , $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signals have been placed in the high impedance state.
$\overline{\text{WAIT}}$	1	Input	Wait signal. $\overline{\text{WAIT}}$ signal is a signal to inform MPU of specified memory or peripheral LSI which is not ready for data transfer. As long as $\overline{\text{WAIT}}$ signal as at "0" level, MPU is continuously kept in the wait state.
$\overline{\text{BUSREQ}}$	1	Input	Bus request signal. $\overline{\text{BUSREQ}}$ signal is a signal requesting placement of the address bus, data bus, $\overline{\text{MREQ}}$ , $\overline{\text{IORQ}}$ , $\overline{\text{RD}}$ and $\overline{\text{WR}}$ signals in the high impedance state. $\overline{\text{BUSREQ}}$ signal is normally wired-OR. In this case, a pull-up resistor is externally connected.

**Table 1 Pin Names and Functions (continued)**

Pin Name	Number of Pin	Input/Output 3-state	Function
RESET	1	Input	Reset signal. RESET signal is used for initializing MPU and must be kept in active state ("0") for a period of at least 3 clocks.
M1	1	Output	Signal showing machine cycle 1. "0" is output together with MREQ signal in the operation code fetch cycle. This signal is output for every opcode fetch when 2 byte opcode is executed. In the maskable interrupt acknowledge cycle, this signal is output together with IORQ signal.
XTAL 1 (XIN) XTAL 2 (XOUT)	2	Input Output	Crystal oscillator connecting terminal.
CLK	1	Output	Single-phase clock output. Clock polarity is in-phase with OSC-IN (XTAL 1) so that Z80 users could use OSC-IN as clock input without needing extra inverter on the board. When the HALT instruction in STOP Mode is executed, MPU stops its operation and holds clock output at "0" level.
VCC (1), (2)	2	Power supply	+5V Connect pin 34 and pin 12 externally.
VSS	1	Power supply	0V

## FUNCTIONAL DESCRIPTION:

The system configuration, functions and basic operation of the Z84C01 are described here.

**Block Diagram.** The block diagram of the internal configuration is shown in Fig. 2.

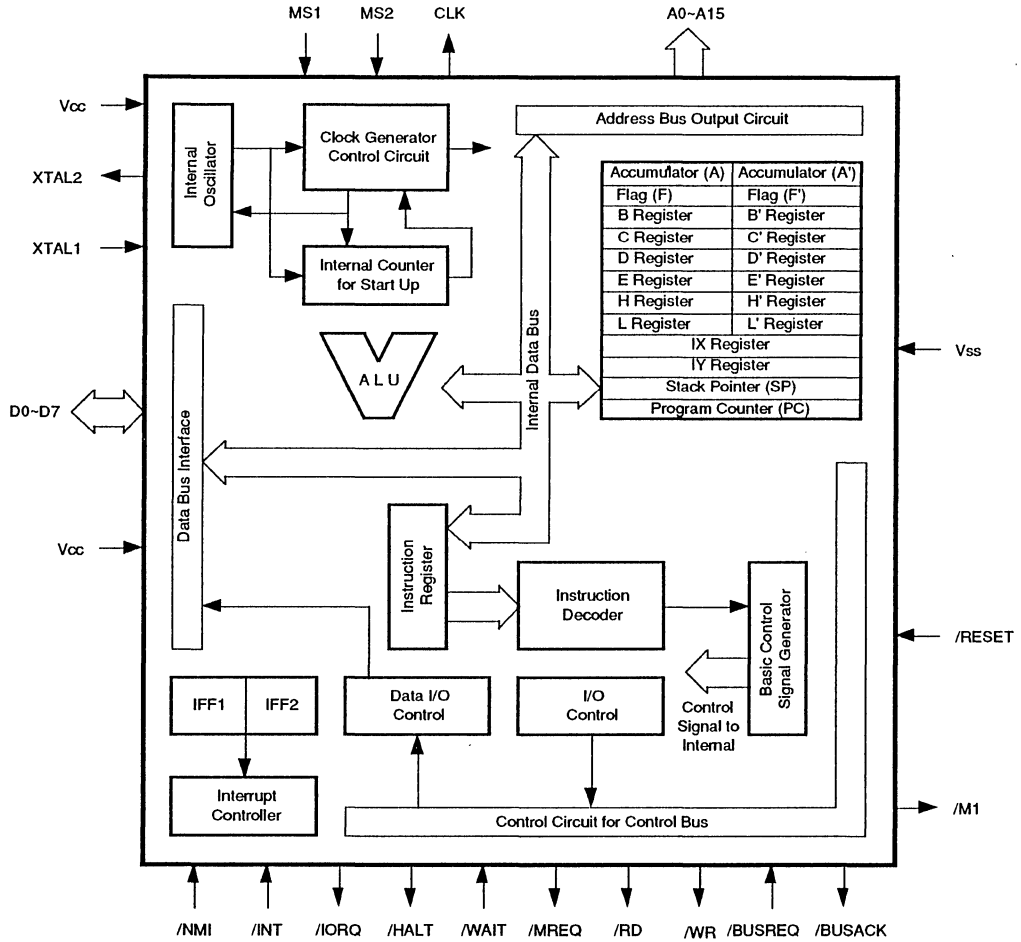


Figure 2. Block Diagram

**System Configuration.** The Z84C01 has a built-in system clock generator for CMOS Z80 in addition to the standard functions of the Z84C00 MPU. The explanation is provided here with emphasis placed on the halt function relative to the clock generator, which is an additional function. The internal register group, reset and interrupt function are identical to those of the Z84C00. For details, please refer to the data sheet for the Z84C00.

In this section, the following principal components and functions will be described:

- (1) Generation of clock
- (2) Operation mode
- (3) Start-up time at time of restart

**Generating the System Clock.** The Z84C01 has a built-in oscillation circuit and required clock can be easily generated by connecting an oscillator to the external terminals (XTAL1, XTAL2). Clock in the same frequency as input oscillation frequency is generated.

Examples of oscillator connection are shown in Figures 3a, 3b.

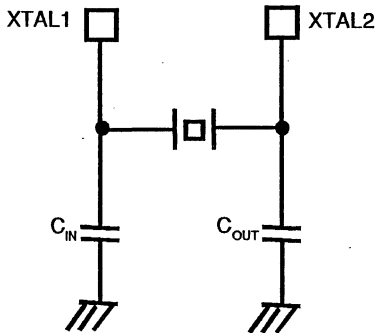


Figure 3a Example of Oscillator Connection and Constant

$C_{IN}$	$C_{OUT}$
22 <sub>PF</sub>	33 <sub>PF</sub>

Figure 3b Example of Oscillator Connection and Constant

**Operation Modes.** There are four kinds of operation modes available for the Z84C01 in connection with generation of clock; RUN Mode, IDLE1/2 Modes and STOP Mode. One of these modes is selected by the mode select inputs (MS1, MS2).

The operation mode is effective when the halt instruction is executed. Restart of MPU from the stopped state under IDLE1/2 Mode or STOP Mode is effected by inputting either RESET signal or interrupt signal (INT or NMI).

Operations of these modes in the halt state are shown in Table 2.

---

**Table 2 Clock Generating Operation Mode**

---

Operation Mode	MS1	MS2	Description at HALT State
RUN Mode	1	1	MPU continues the operation and supplies clock to the outside continuously.
IDLE 1 Mode	0	0	The internal oscillator's operation is continued. Clock (CLK) output as well as internal operations are stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.
IDLE 2 Mode	0	1	The internal oscillator's operation and clock (CLK) output are continued but the internal operations are stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.
STOP Mode	1	0	All operations of the internal oscillator, clock (CLK) output, and internal operation are stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.

---

**Start-up Time at Time of Restart (STOP Mode).**

When MPU is released from the halt state by accepting an interrupt request, MPU, then will execute an interrupt service routine. Therefore, when an interrupt request is accepted, MPU starts generation of internal system clock and clock output after a start-up time by the internal counter ( $2^{14}+2.5$ ) TcC (TcC: Clock Cycle) to obtain a stabilized oscillation for MPU operation.

Further, in case of the restart by  $\overline{\text{RESET}}$  signal, the internal counter does not operate for a quick operation at time of power ON.

**Status Change Flowchart and Basic Timing.** In this section, the status change and basic timing when the Z84C01 is operating are explained.

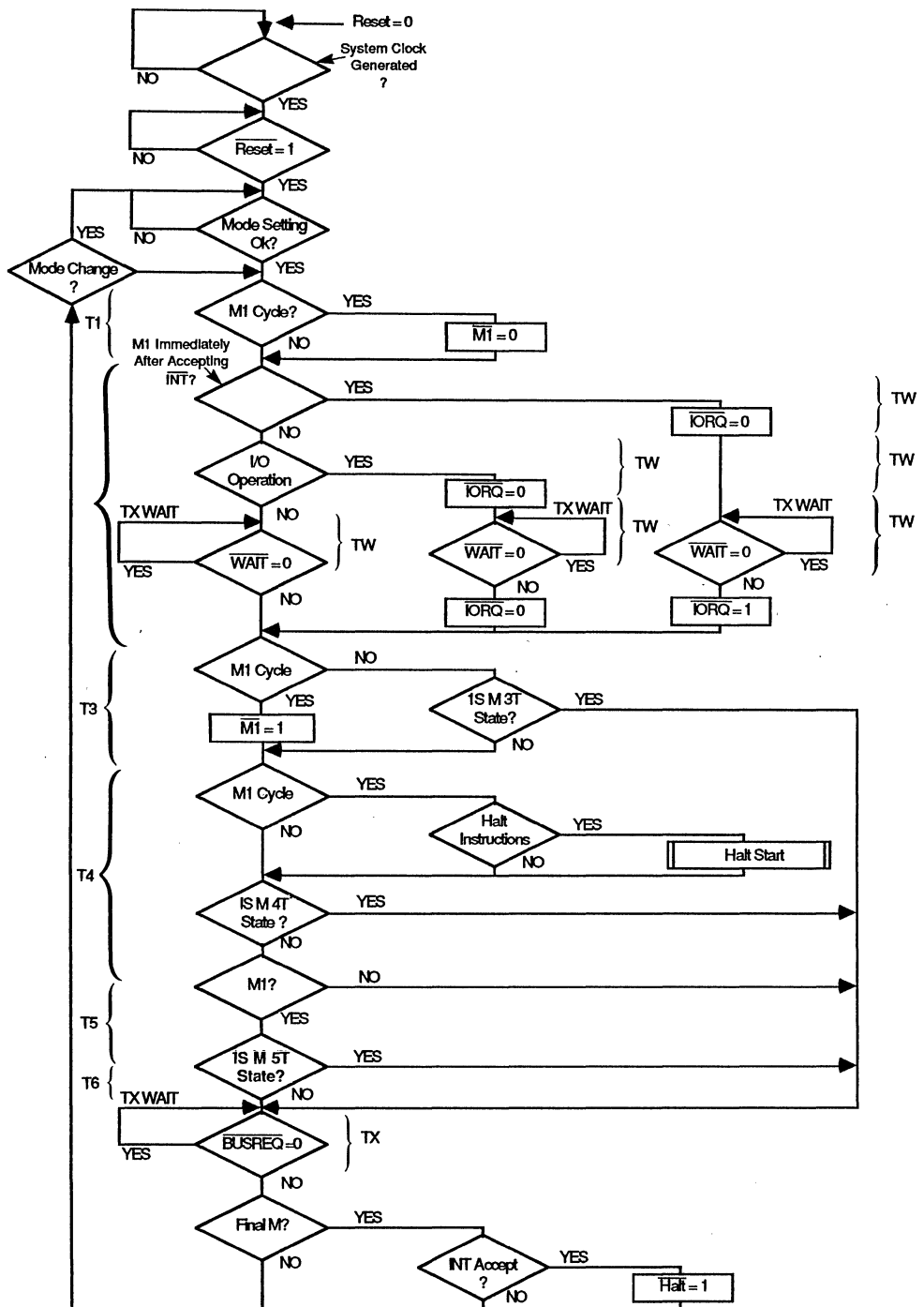


Figure 4 (a) Status Change Flowchart

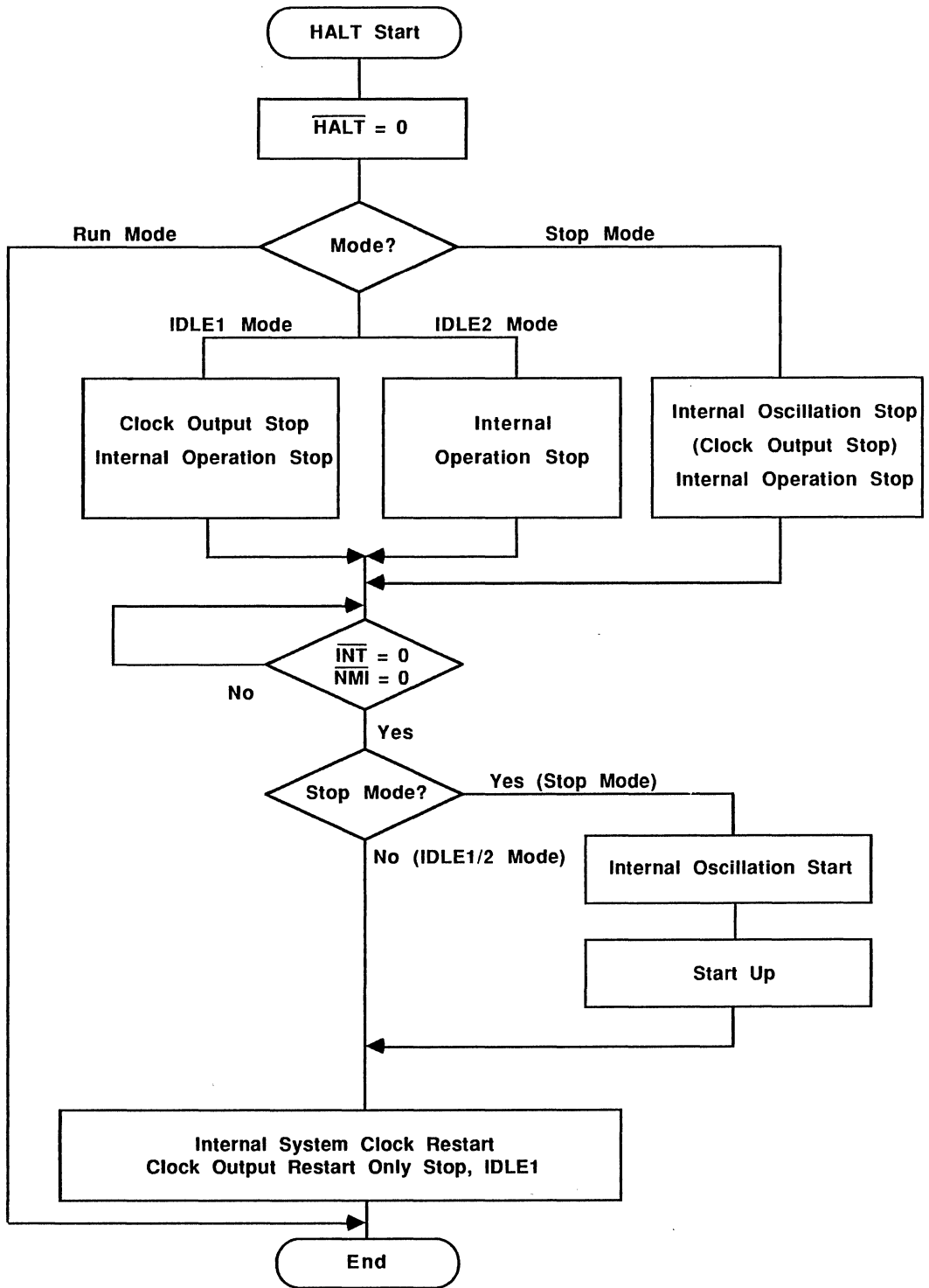


Figure 4 (b) Status Change Flowchart



**Basic Timing.** The basic timing is explained here with emphasis placed on the halt function relative to the clock generator. Except  $\overline{\text{RFSH}}$  signal output, the following items are identical to those for the Z84C00. Refer to the data sheet for the Z84C00.

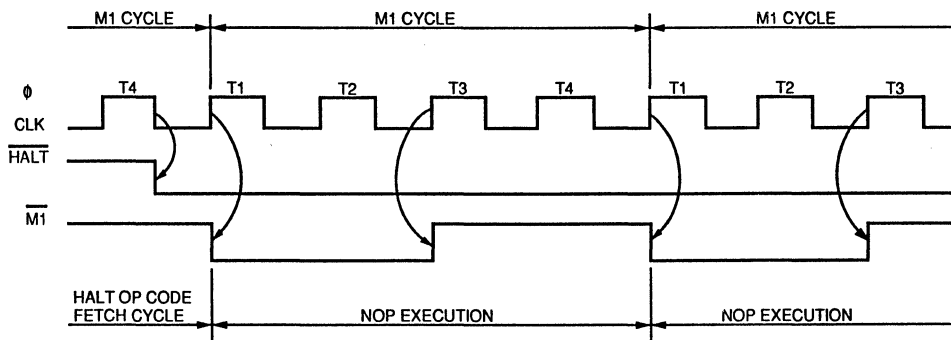
- Operation code fetch cycle
- Memory read/write operation
- Input/output operation
- Bus request/acknowledge operation
- Maskable interrupt request operation
- Non-maskable interrupt request operation
- Reset operation

Note that the Z84C01 does not have the refresh terminal ( $\overline{\text{RFSH}}$ ), but refresh address is output on the address bus in the operation code fetch cycle (M1) as in the Z84C00 since the on-chip refresh control circuit is available.

(1) **Operation When HALT Instruction is Executed**  
When MPU fetches a halt instruction in the operation code fetch cycle, HALT signal goes active (low level) in synchronous with falling edge of T4 state for the peripheral LSI and MPU stops the operation. The system clock generating operation after this differs depending upon the operation mode (RUN Mode, IDLE1/2 Mode or STOP Mode). If the internal system clock is running, MPU continues to execute NOP instruction even in the halt state.

(a) **RUN Mode (MS1=1, MS2=1)**  
Shown in Fig. 5 is the basic timing when the halt instruction is executed in RUN Mode.

In RUN Mode, system clock ( $\phi$ ) in MPU and clock output (CLK) are not stopped, even after the halt instruction is executed. Therefore, until the halt state is released by the interrupt signal ( $\overline{\text{NMI}}$  or  $\overline{\text{INT}}$ ) or  $\overline{\text{RESET}}$  signal, MPU continues to execute NOP instruction.

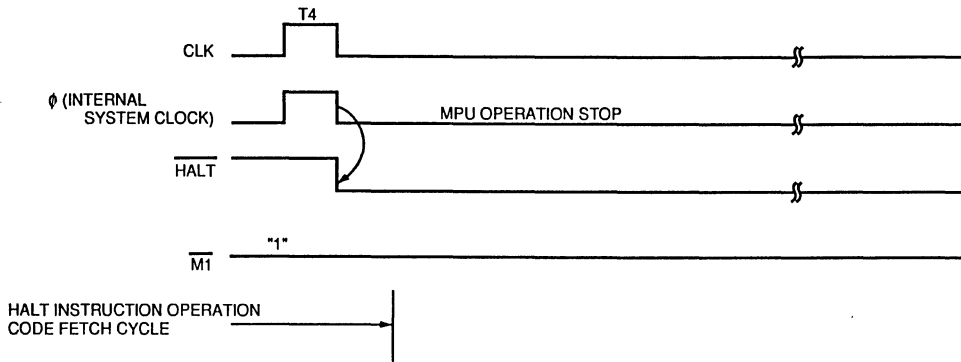


**Figure 5 Timing of RUN Mode  
(at Halt Command Execution)**

**(b) IDLE1 Mode (MS1=0, MS2=0)**

Shown in Fig. 6 is the basic timing when the halt instruction is executed in IDLE1 Mode.

In IDLE1 Mode, system clock ( $\phi$ ) in MPU and clock output (CLK) are stopped and MPU stops its operation after the halt instruction is executed. However, the internal oscillator continues to operate.

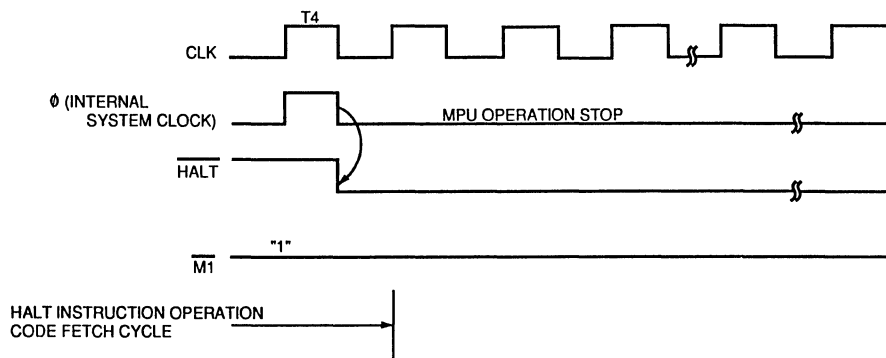


**Figure 6 IDLE1 Mode Timing  
(at Halt Instruction Execution)**

**(c) IDLE2 Mode (MS1=0, MS2=1)**

Shown in Fig. 7 is the basic timing when the halt instruction is executed in IDLE2 Mode.

In IDLE2 Mode, system clock ( $\phi$ ) in MPU is stopped and MPU stops its operation after the halt instruction is executed. However, the internal oscillator and clock output (CLK) to the outside of MPU continues to operate.

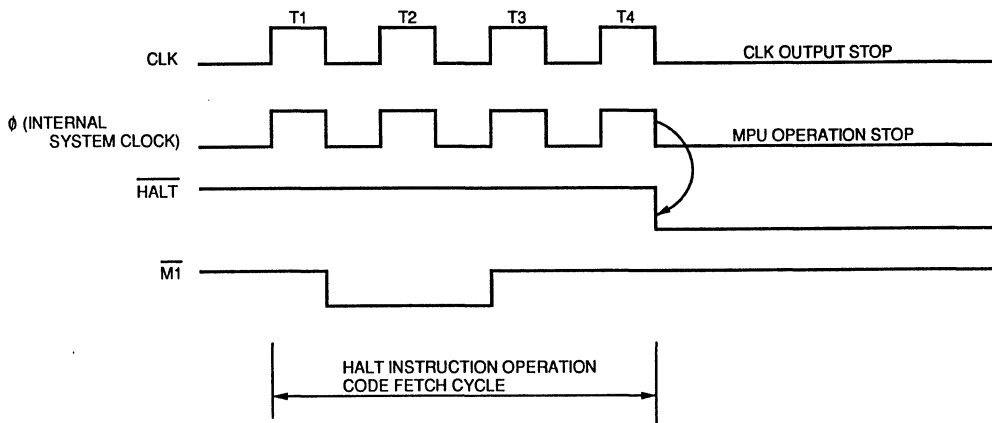


**Figure 7 IDLE2 Mode Timing  
(at Halt Instruction Execution)**

**(d) STOP Mode (MS1=1, MS2=0)**

Shown in Fig. 8 is the basic timing when the halt instruction is executed in STOP Mode.

In STOP Mode, internal operation and internal oscillator are stopped after the halt instruction is executed. Therefore, system clock ( $\phi$ ) in MPU and clock output (CLK) to the outside of MPU are stopped.



**Figure 8 STOP Mode Timing  
(at Halt Instruction Execution)**

**(2) Release from Halt State**

The halt state of MPU is released when "0" is input to RESET signal and MPU is reset or an interrupt request is accepted. An interrupt request signal is sampled at the leading edge of the last clock cycle (T4 state) of NOP instruction. In case of the maskable interrupt, interrupt will be accepted by an active INT signal ("0" level). Also the interrupt enable flip-flop must have been set to "1". The accepted interrupt process is started from next cycle.

Further, when the internal system clock is stopped (IDLE1/2 Mode, STOP Mode), it is necessary first to restart the internal system clock. The internal system clock is restarted when RESET or interrupt signal (NMI or INT) is input.

**(a) RUN Mode (MS1, MS2=1)**

The halt release operation by acceptance of interrupt request in RUN Mode is shown in Fig. 9.

In RUN Mode the internal system clock is not stopped, and therefore, if the interrupt signal is recognized at the rise of T4 state of the continued NOP instruction, MPU will execute the interrupt process from next cycle.

The halt release operation by resetting MPU in RUN Mode is shown in Fig. 10. After reset, MPU will execute an instruction starting from address 0000H. However, in order to reset MPU it is necessary to keep RESET signal at "0" for at least 3 clocks. In addition, if RESET signal becomes "1", after the dummy cycle for at least two T states, MPU executes an instruction from address 0000H.

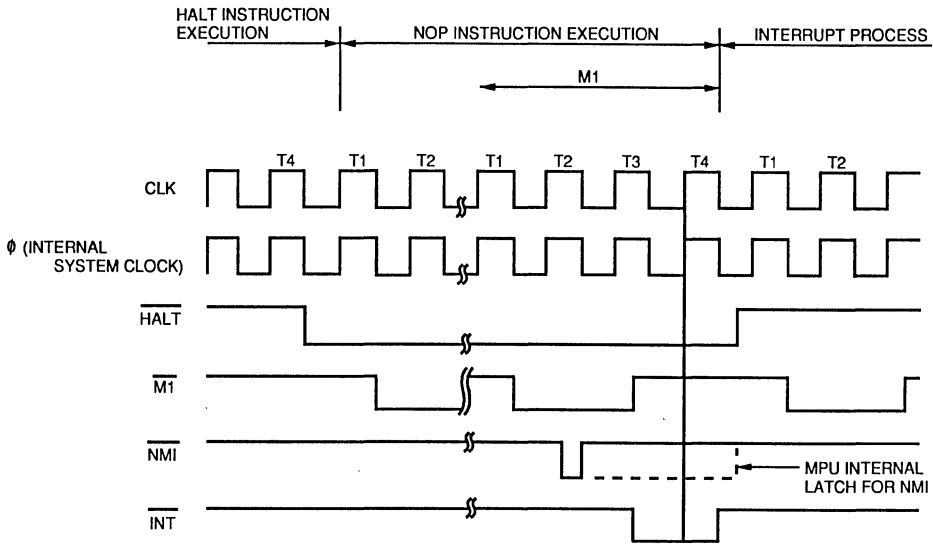


Figure 9 Halt Release Operation Timing by interrupt Request Signal in RUN Mode

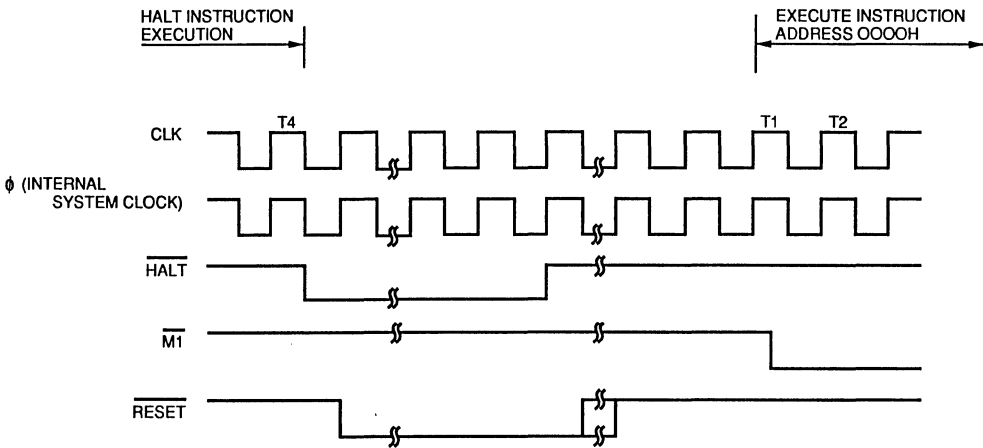


Figure 10 Halt Release Operation Timing by Reset in RUN Mode

(b) IDLE1 Mode (MS1=0, MS2=0), IDLE2 Mode (MS1=0, MS2=1)

The halt release operation by interrupt signal in IDLE1 Mode is shown in Fig. 11 (a) and in IDLE2 Mode in Fig. 11 (b).

When receiving  $\overline{\text{NMI}}$  or  $\overline{\text{INT}}$  signal, MPU starts the internal system clock operation. In IDLE1 Mode, MPU starts clock output to the outside at the same time.

The operation stop of MPU in IDLE1/2 Mode is taking place at "0" level during T4 state in the halt instruction operation code fetch cycle. Therefore, after being re-started by the interruption signal, MPU executes one NOP instruction and samples an interrupt signal at the rise of T4 state during the execution of this NOP instruction, and executes the interrupt process from next cycle.

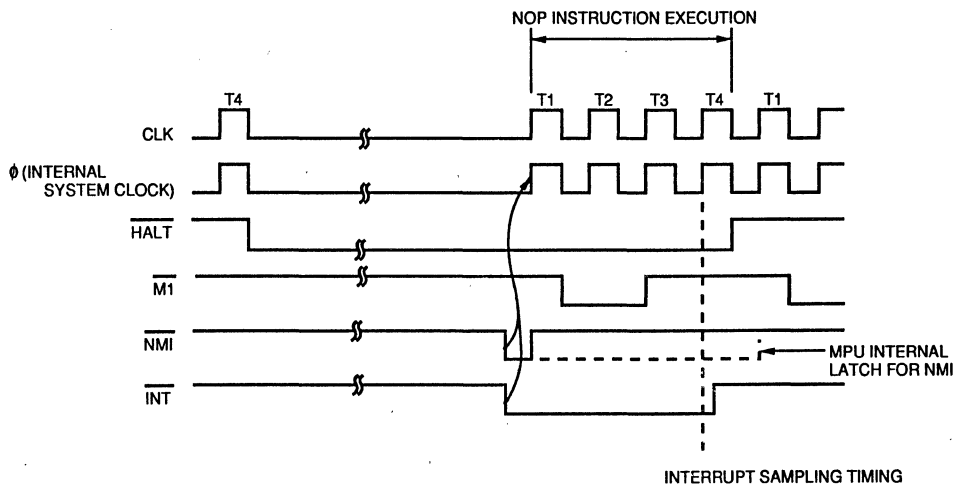


Figure 11 (a) IDLE1 Mode

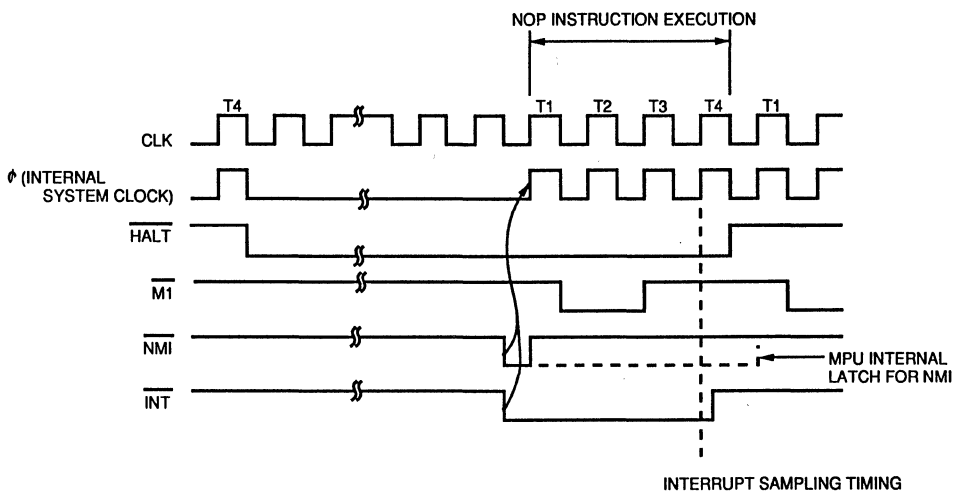


Figure 11 (b) IDLE2 Mode

Figure 11 Halt Release Operation Timing by Interrupt Request Signal in IDLE1/2 Mode

If no interrupt signal is accepted during the execution of the first NOP instruction after the internal system clock is restarted, MPU is not released from the halt state and is placed in IDLE1/2 Mode again at "0" level during T4 state of the NOP instruction, stopping the internal system clock. If  $\overline{INT}$  signal is not at "0" level at the rise of T4 state, no interrupt request is accepted.

When  $\overline{RESET}$  signal at "0" level is input into MPU, the internal system clock is restarted and MPU will execute an instruction stored in address 0000H.

At time of  $\overline{RESET}$  signal input, it is necessary to take the same care as that in resetting MPU in RUN Mode.

The halt release operation by resetting MPU in IDLE1 Mode is shown in Fig. 12 (a) and that in IDLE2 Mode in Fig. 12 (b).

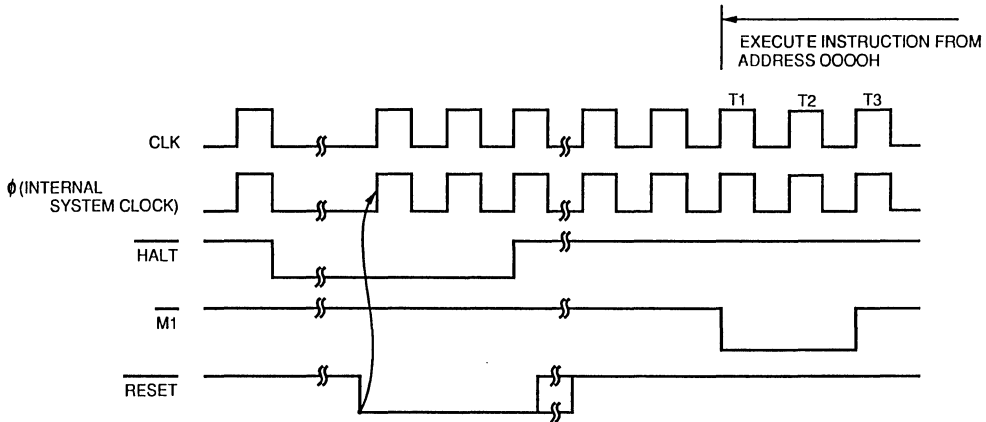


Figure 12 (a) IDLE1 Mode

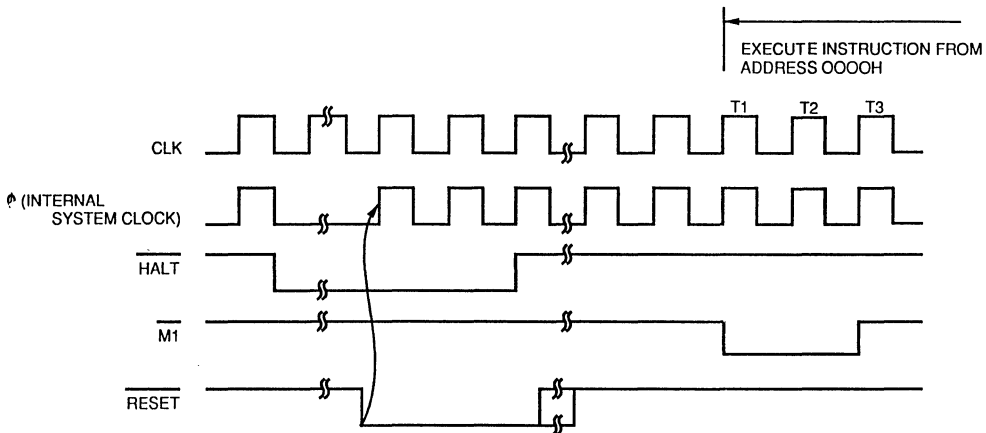


Figure 12 (b) IDLE2 Mode

Figure 12 Halt Release Operation Timing by Reset in IDLE1/2 Mode

**(c) STOP Mode (MS1=1, MS2=0)**

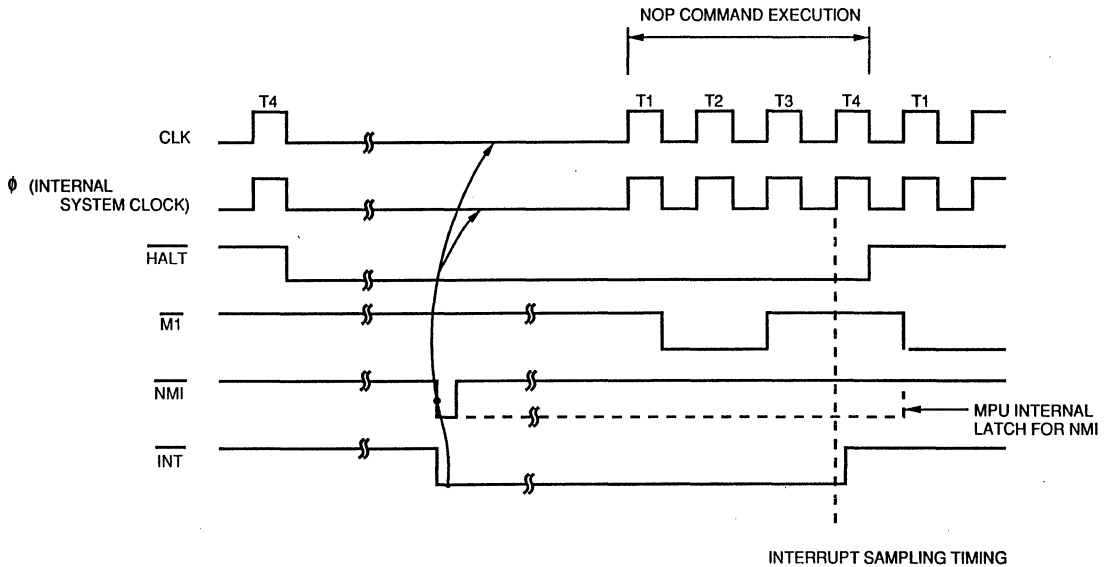
The halt release operation by interrupt signal in STOP Mode is shown in Fig. 13.

When MPU received an interrupt signal, the internal oscillator is restarted. In order to obtain stabilized oscillation, the internal system clock and clock output to the outside are started after a start-up time of  $(2^{14}+2.5) T_{cC}$  ( $T_{cC}$ : Clock Cycle) by the internal counter.

MPU executes one NOP instruction after the internal system clock is restarted and at the same time, sampling an interrupt signal at the rise of T4 state during the execution of this NOP instruction. If the interrupt signal is accepted, MPU executes the interrupt process operation from next cycle.

At time of interrupt signal input, it is necessary to take the same care as that in the interrupt signal input in IDLE1/2 Mode. The halt release operation by MPU resetting in STOP Mode is shown in Fig. 14.

When  $\overline{\text{RESET}}$  signal at "0" level is input into MPU, the internal oscillator is restarted. However, since it performs a quick operation at time of power ON, the internal counter does not operate. Therefore, the operation may not be carried out properly due to unstable clock immediately after the signal in STOP Mode, it is necessary to hold  $\overline{\text{RESET}}$  signal at "0" level for sufficient time. When  $\overline{\text{RESET}}$  signal becomes "1", after the dummy cycle for at least 2T states, MPU starts to execute an execution from address 0000H.



**Figure 13 Halt Release Operation Timing by Interrupt Request Signal in STOP Mode**

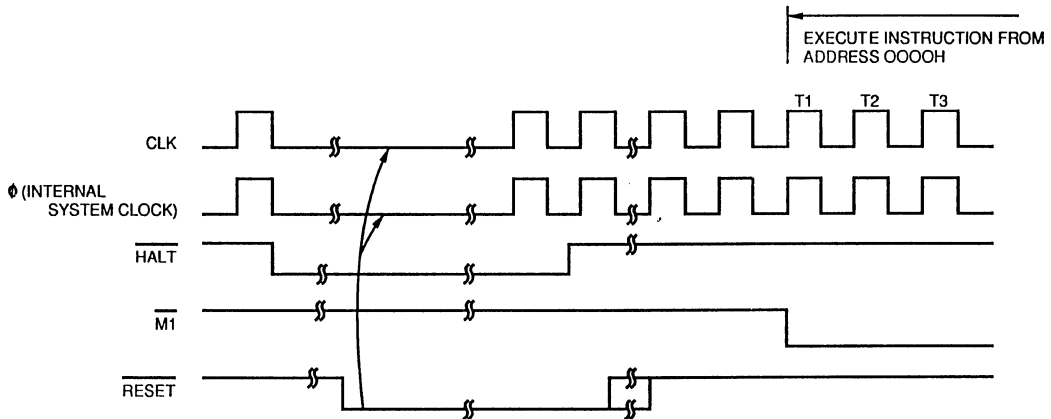


Figure 14 Halt Release Operation Timing by Reset in STOP Mode

**Instruction Set.** Instruction set of the Z84C01 is the same as that for the Z84C00. For details refer to the data sheet for the Z84C00.

**Method of Use.** An example of the Z84C01 with the Z80 family peripheral LSI's is shown in Fig. 15.



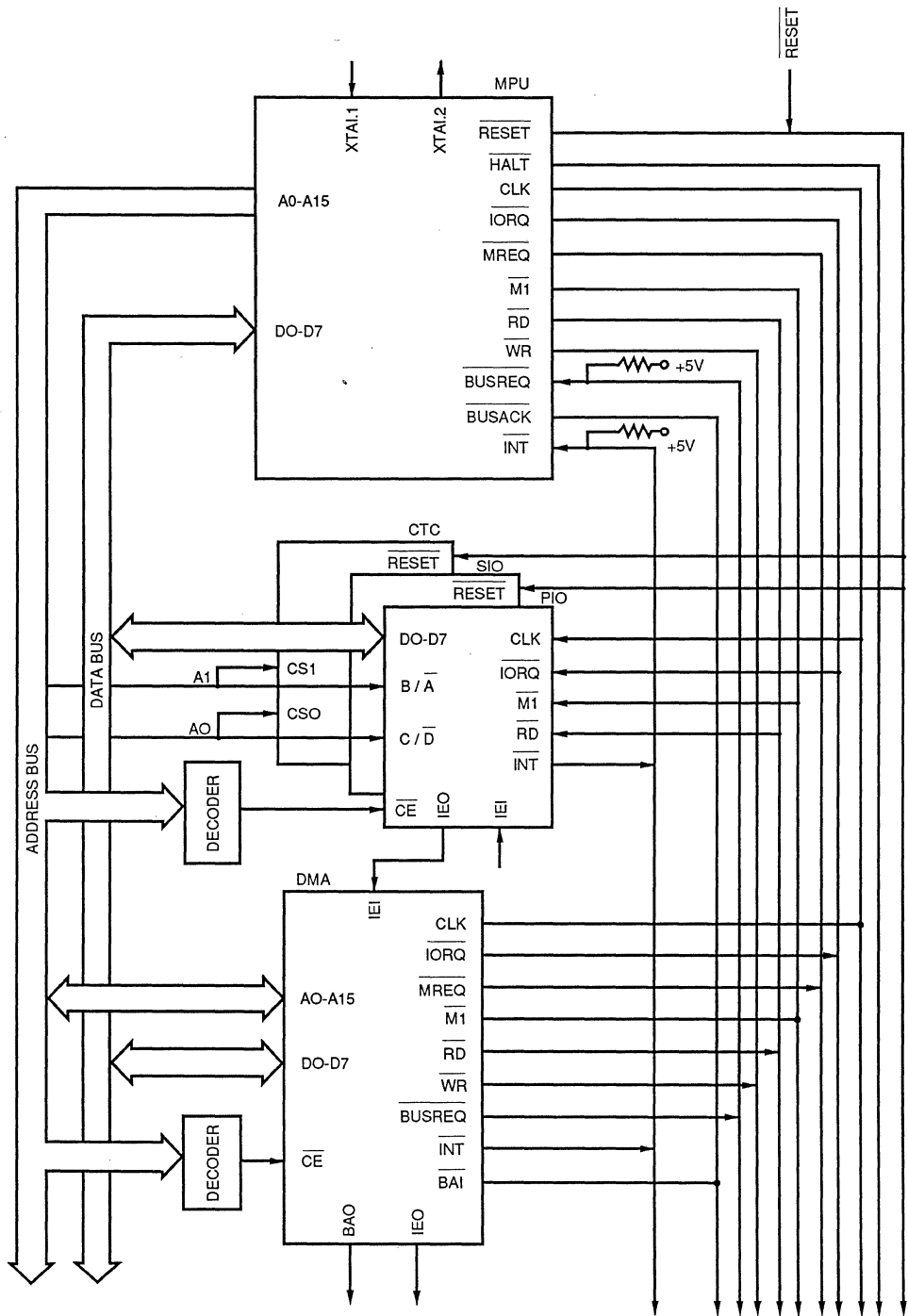


Figure 15 Example of Z80 Family Peripheral LSI

## CPU TIMING

**Timing Diagrams.** The Z84C01 CPU executes instructions by proceeding through a specific sequence of operations:

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

The basic clock period is referred to as a Time or Cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

**Instruction Opcode Fetch.** The CPU places the contents of the Program Counter (PC) on the address bus as the start of the cycle (Figure 16). Approximately one-half clock cycle later,  $\overline{MREQ}$  goes active. When active,  $\overline{RD}$  indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the  $\overline{WAIT}$  input with the falling edge of clock state T2. During clock states T3 and T4 of an  $\overline{M1}$  cycle, dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction.

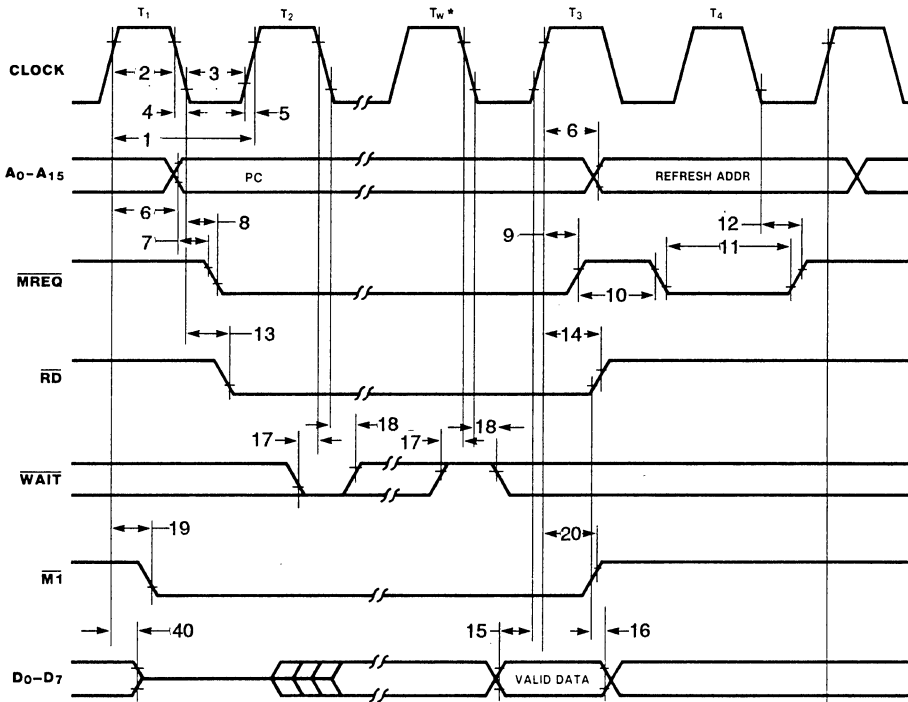


Figure 16 Instruction Opcode Fetch

Memory Read or Write Cycles. Figure 17 shows the timing of memory read or write cycles other than an opcode fetch ( $\overline{M1}$ ) cycle. The  $\overline{MREQ}$  and  $\overline{RD}$  signals function exactly as in the fetch cycle.

In a memory write cycle,  $\overline{MREQ}$  also becomes active when the address bus is stable. The  $\overline{WR}$  line is active when the data bus is stable, so that it can be used directly as an R/W pulse to most semiconductor memories.

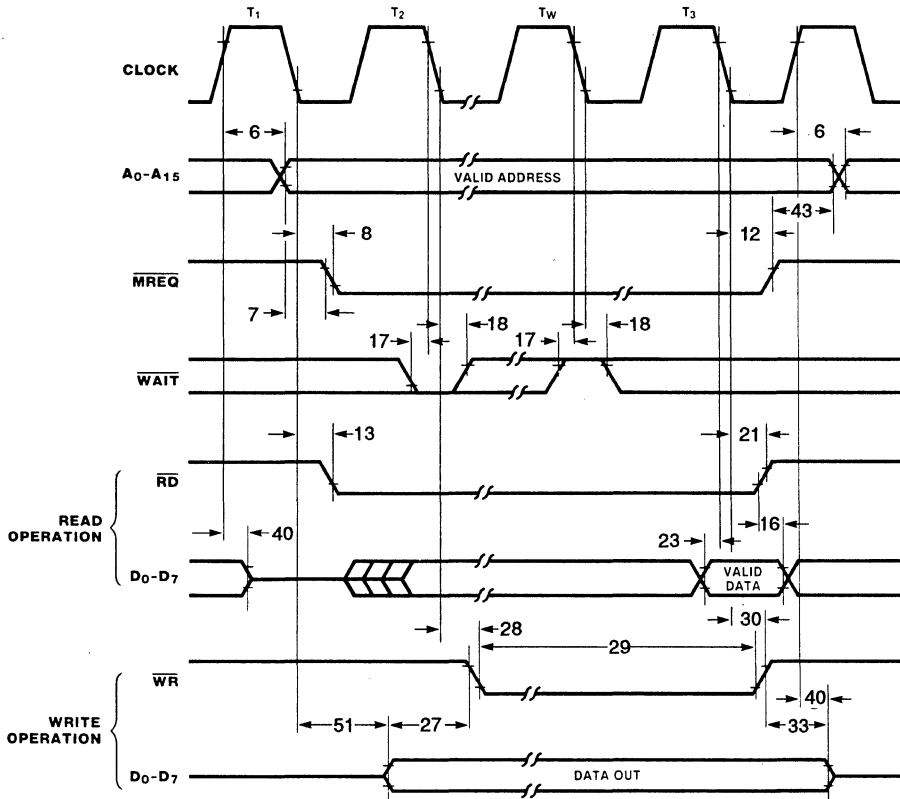
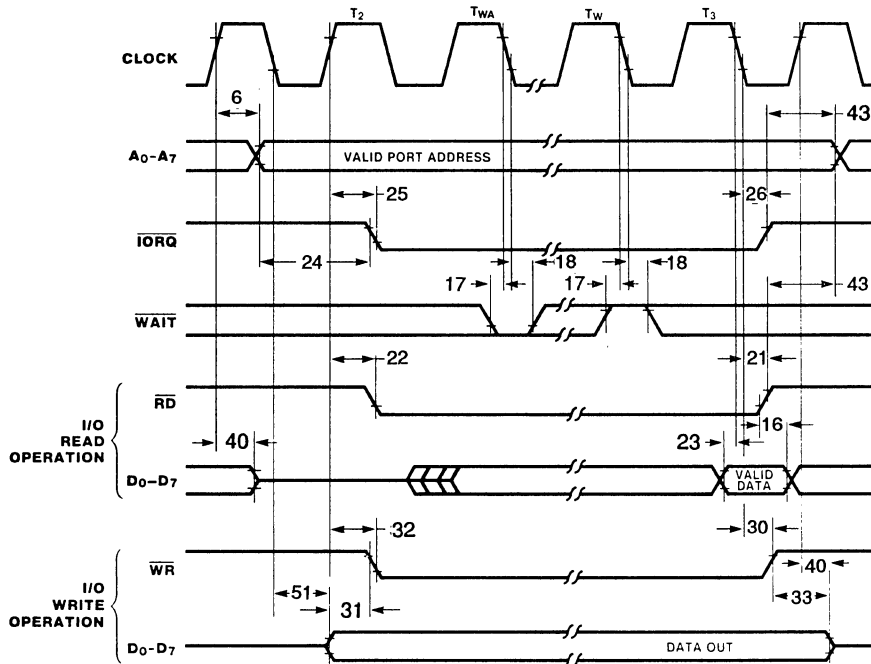


Figure 17 Memory Read or Write Cycles

**Input or Output Cycles.** Fig. 18 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single Wait state ( $T_{WA}$ ).

This extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.

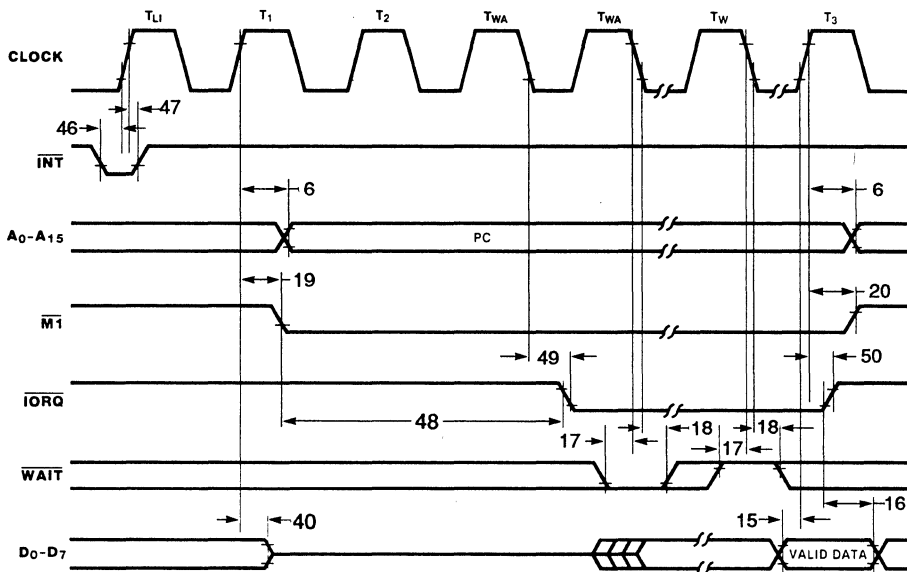


$T_{WA}$  = One wait cycle automatically inserted by CPU.

**Figure 18 Input or Output Cycles**

**Interrupt Request/Acknowledge Cycle.** The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Fig. 19). When an interrupt is accepted, a special  $\overline{M1}$  cycle is generated.

During this  $\overline{M1}$  cycle,  $\overline{IORQ}$  becomes active (instead of  $\overline{MREQ}$ ) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.

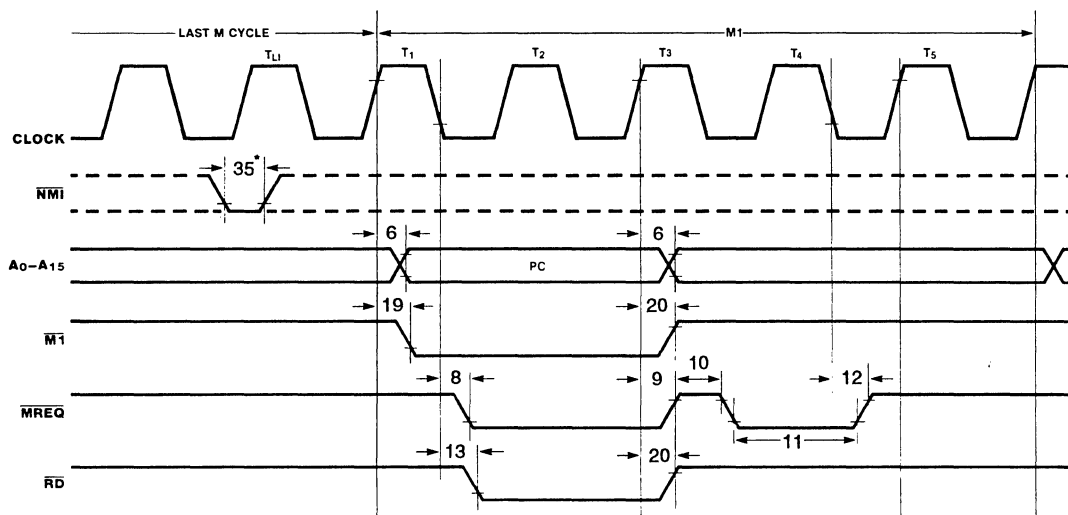


NOTES: 1)  $T_{LI}$  = Last state of any instruction cycle.  
 2)  $T_{WA}$  = Wait cycle automatically inserted by CPU.

Figure 19 Interrupt Request/Acknowledge Cycle

**Non-Maskable Interrupt Request Cycle.**  $\overline{\text{NMI}}$  is sampled at the same time as the maskable interrupt input  $\overline{\text{INT}}$ , but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that

of a normal memory read operation except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the  $\overline{\text{NMI}}$  service routine located at address 0066H (Fig. 20).

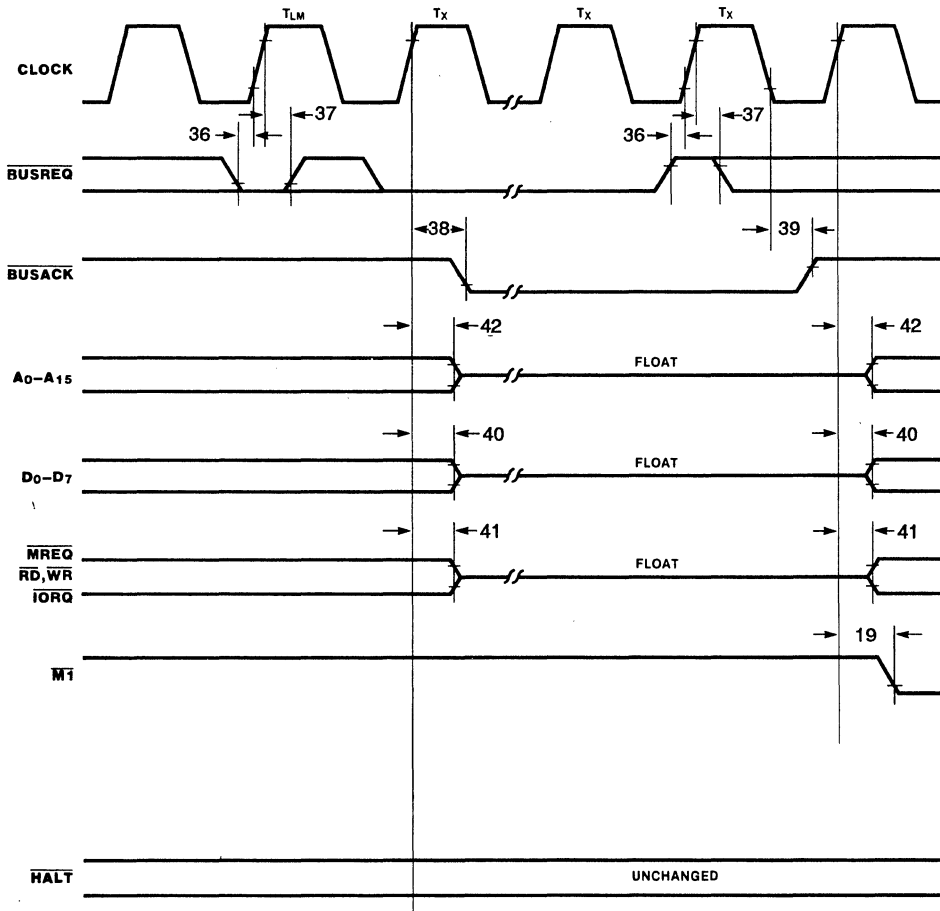


\*Although  $\overline{\text{NMI}}$  is an asynchronous input, to guarantee its being recognized on the following machine cycle,  $\overline{\text{NMI}}$ 's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle ( $T_{L1}$ ).

**Figure 20 Non-Maskable Interrupt Request Operation**

**Bus Request/Acknowledge Cycle.** The CPU samples  $\overline{\text{BUSREQ}}$  with the rising edge of the last clock period of any machine cycle (Fig. 21). If  $\overline{\text{BUSREQ}}$  is active, the CPU sets its address, data, and  $\overline{\text{MREQ}}$ ,  $\overline{\text{IORQ}}$ ,  $\overline{\text{RD}}$ , and

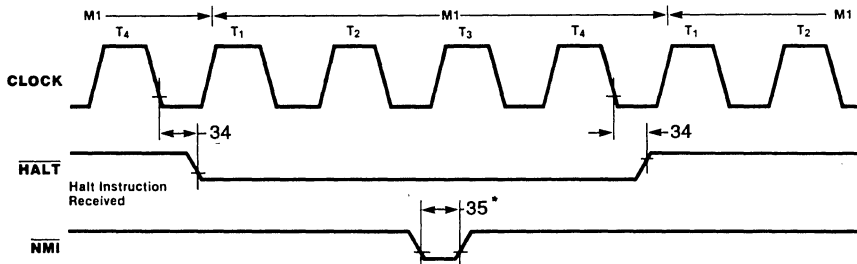
$\overline{\text{WR}}$  lines to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.



NOTES: 1)  $T_{LM}$  = Last state of any M cycle.  
 2)  $T_x$  = An arbitrary clock cycle used by requesting device.

Figure 21 BUS Request/Acknowledge Cycle

## Halt Acknowledge Cycle.



\*Although  $\overline{\text{NMI}}$  is an asynchronous input, to guarantee its being recognized on the following machine cycle,  $\overline{\text{NMI}}$ 's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle ( $T_{L1}$ ).

Figure 22 Halt Acknowledge

**Reset Cycle.**  $\overline{\text{RESET}}$  must be active for at least three clock cycles for the CPU to properly accept it. As long as  $\overline{\text{RESET}}$  remains active, the address and data buses float, and the control outputs are inactive.

Once  $\overline{\text{RESET}}$  goes inactive, two internal T cycles are consumed before the CPU resumes normal processing operation.  $\overline{\text{RESET}}$  clears the PC register, so the first opcode fetch will be location 0000H (Fig. 23).

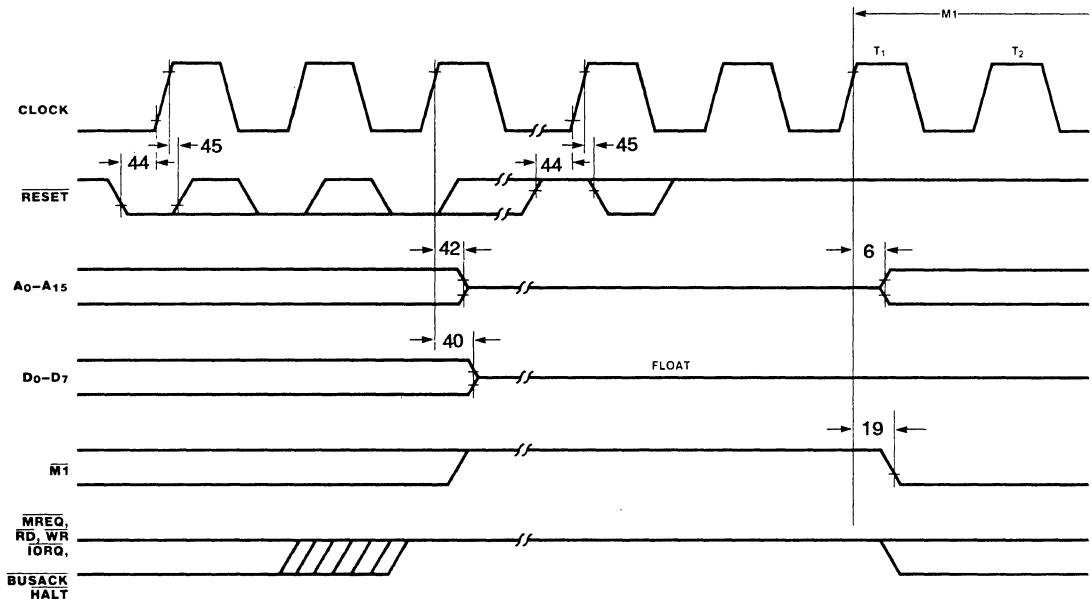
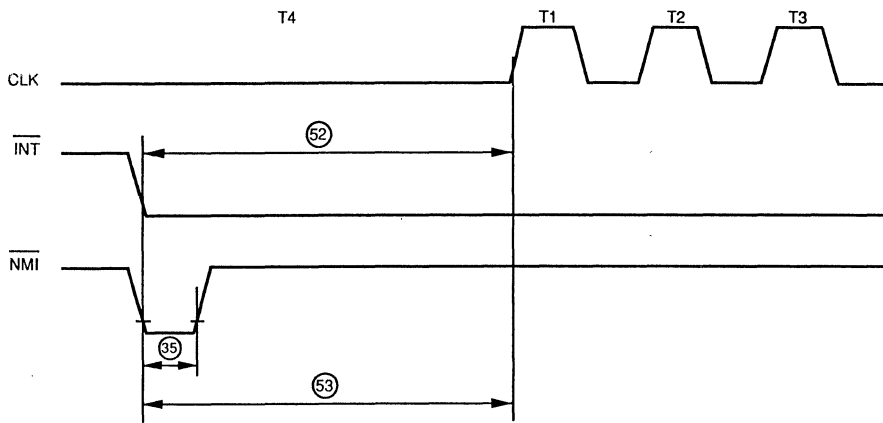
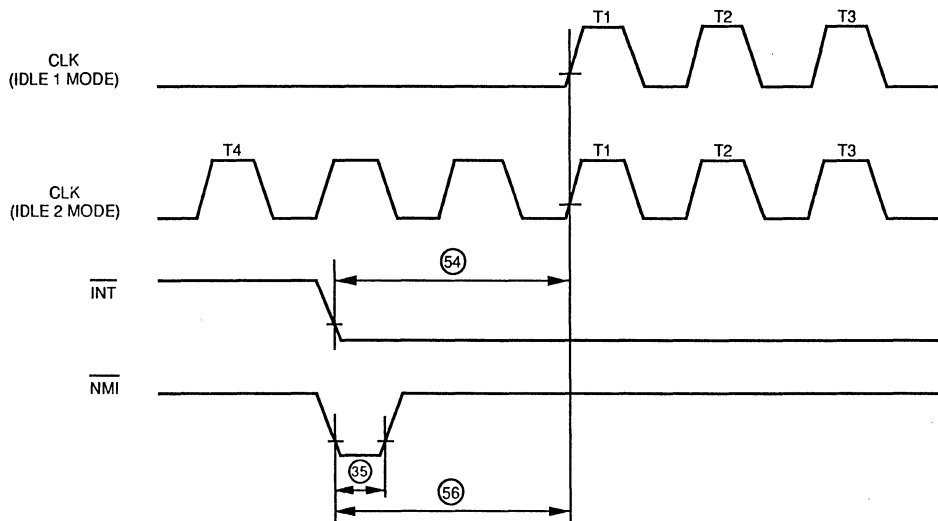


Figure 23 Reset Cycle





**Figure 24 Clock Restart Timing (STOP Mode)**



**Figure 25 Clock Restart Timing (IDLE1/2 Mode)**

---

## PRECAUTIONS:

- (1) To reset MPU, it is necessary to hold  $\overline{\text{RESET}}$  signal input at "0" level for at least three clocks.

In particular, to release the HALT state by  $\overline{\text{RESET}}$  signal in STOP Mode, hold  $\overline{\text{RESET}}$  signal at "0" level for sufficient time in order to stabilize output from the internal oscillator.

- (2) In releasing MPU from the HALT state by interrupt signal in IDLE1/2 Mode and STOP Mode, MPU will not be released from the HALT state and

the internal system clock will stop again unless an interrupt signal is accepted during the execution of NOP instruction even when the internal system clock is restarted by the interrupt signal input. In particular, care must be taken when  $\overline{\text{INT}}$  is used.

Other precautions are identical to those for the Z84C00, except those for  $\overline{\text{RFSH}}$  terminal. Refer to the data sheet for the Z84C00.

## AC CHARACTERISTICS:

No	Symbol	Parameter	Z84C0106		Z84C0110		Unit	Note
			Min	Max	Min	Max		
1	TcC	Clock Cycle Time	162*	DC	100*	DC	nS	
2	TwCh	Clock Pulse Width (High)	65	DC	40	DC	nS	
3	TwCl	Clock Pulse Width (Low)	65	DC	40	DC	nS	
4	TfC	Clock Fall Time		20		10	nS	
5	TrC	Clock Rise Time		20		10	nS	
6	TdCr(A)	Address Valid from Clock Rise		90		60	nS	
7	TdA(MREQf)	Address valid to /MREQ Fall	35*		5*		nS	
8	TdC(MREQf)	Clock Fall to /MREQ Fall Delay		70		40	nS	
9	TdCr(MREQr)	Clock Rise to /MREQ Rise Delay		70		40	nS	
10	TwMREQh	/MREQ Pulse Width (High)	60*		25*		nS	[1]
11	TwMREQl	/MREQ Pulse Width (Low)	132*		70*		nS	[1]
12	TdC(MERQr)	Clock Fall to /MREQ Rise Delay		70		40	nS	
13	TdCl(RDf)	Clock Fall to /RD Fall Delay		80		40	nS	
14	TdCr(RDr)	Clock Rise to /RD Rise Delay		70		40	nS	
15	TsD(Cr)	Data Setup Time to Clock Rise	30		25		nS	
16	ThD(RDr)	Data Hold Time after /RD Rise	0		0		nS	
17	TsWAIT(Cf)	/WAIT Setup Time to Clock Fall	60		30		nS	
18	ThWAIT(Cf)	/WAIT Hold Time after Clock Fall	10		0		nS	
19	TdCr(M1f)	Clock Rise to /M1 Fall Delay		80		40	nS	
20	TdCr(M1r)	Clock Rise to /M1 Rise Delay		80		40	nS	
21	TdCl(RDr)	Clock Fall to /RD Rise Delay		70		40	nS	
22	TdCr(RDf)	Clock Rise to /RD Fall Delay		70		40	nS	
23	TsD(Cf)	Data Setup to Clock Fall During M2, M3, M4 or M5 Cycles	40		25		nS	
24	TdA(IORQf)	Address Stable Prior to /IORQ Fall	107*		50*		nS	
25	TdCr(IORQf)	Clock Rise to /IORQ Fall Delay		65		40	nS	
26	TdCl(IORQr)	Clock Fall to /IORQ Rise Delay		70		40	nS	
27	TdD(WRf)Mw	Data Stable Prior to /WR Fall	22*		0*		nS	
28	TdCl(WRf)	Clock Fall to /WR Fall Delay		70		40	nS	
29	TwWR	/WR Pulse Width	132*		75*		nS	
30	TdCl(WRr)	Clock Fall to /WR Rise Delay		70		40	nS	
31	TdD(WRf)IO	Data Stable Prior to /WR Fall	-55*		-50*		nS	
32	TdCr(WRf)	Clock Rise to /WR Fall Delay		60		40	nS	
33	TdWRr(D)	Data Stable from /WR Fall	30*		0*		nS	
34	TdCl(HALT)	Clock Fall to /HALT 'L' or 'H'		260		100	nS	
35	TwNMI	/NMI Pulse Width	70		60		nS	
36	TsBUSREQ(Cr)	/BUSREQ Setup Time to Clock Rise	50		35		nS	
37	ThBUSREQ(Cr)	/BUSREQ Hold Time After Clock Rise	10		0		nS	
38	TdCr(BUSACKf)	Clock Rise to /BASACK Fall Delay		90		40	nS	
39	TdCl(BUSACKr)	Clock Fall to /BASACK Rise Delay		90		40	nS	

## Z84C01 AC CHARACTERISTICS (Continued)

No	Symbol	Parameter	Z84C0106		Z84C0110		Unit	Note
			Min	Max	Min	Max		
40	TdCr(IJz)	Clock Rise to Data Float Delay		80		40	nS	
41	TdCr(CTz)	Clock Rise to Control Outputs Float Delay (/MREQ, /IORQ, /RD and /WR)		70		40	nS	
42	TdCr(Az)	Clock Rise to Address Float Delay		80		50	nS	
43	TdCTr(A)	Address Hold Time From /MREQ, /IORQ, /RD or /WR	35*		5*		nS	
44	TsRESET(Cr)	/RESET to Clock Rise Setup Time	60		30		nS	
45	ThRESET(Cr)	/RESET to Clock Rise Hold Time	10		0		nS	
46	TsINT1(Cr)	/INT Fall to Clock Rise Setup Time	70		50		nS	
47	ThINT1r(Cr)	/INT Rise to Clock Rise Hold Time	10		0		nS	
48	TdM11(IORQf)	/M1 Fall to /IORQ Fall Delay	359*		205*		nS	
49	TdCl(IORQf)	/Clock Fall to /IORQ Fall Delay		70		40	nS	
50	TdCl(IORQr)	Clock Rise to /IORQ Rise Delay		70		40	nS	
51	TdCl(D)	Clock Fall to Data Valid Delay		150		80	nS	
52	TRST1S	CLK Restart Time by /INT (STOP Mode)	(typ) (2 <sup>14</sup> +2.5)TcC		(typ) (2 <sup>14</sup> +2.5)TcC			
53	TRST2S	CLK Restart Time by /NMI (STOP Mode)	(typ) (2 <sup>14</sup> +2.5)TcC		(typ) (2 <sup>14</sup> +2.5)TcC			
54	TRST1I	CLK Restart Time by /INT (IDLE1/2 Mode)	(typ)2.5TcC		(typ)2.5TcC			
55	TRST2I	CLK Restart Time by /NMI (IDLE1/2 Mode)	(typ)2.5TcC		(typ)2.5TcC			

### Notes:

\* For clock periods other than minimum shown, calculate parameters using following 'Note'.

Calculated values above assumed TrC = TfC = maximum.

[1] Increasing delay by 10nS for each 50pF increase in loading, 200pF max for data lines, and 100pF for control lines.

## Z84C01 AC CHARACTERISTICS

### Footnotes

No	Symbol	Parameter	Z84C0106	Z84C0110
1	TcC	TwCh + TwCl + TrC + TfC		
7	TdA(MREQf)	TwCh + TfC	-50	-45
10	TwMREQh	TwCh + TfC	-25	-25
11	TwMREQl	TcC	-30	-30
24	TdA(IORQf)	TcC	-55	-50
27	TdD(WRf)	TcC	-140	-100
29	TwWR	TcC	-30	-25
31	TdD(WRf)	TwCl + TrC	-140	-100
33	TdWRr(D)	TwCl + TrC	-55	-50
43	TdCTr(A)	TwCl + TrC	-50	-45
48	TdM1f(IORQf)	2TcC + TwCh + TfC	-50	-45

## DC CHARACTERISTICS VCC = 5.0 V ±10%

Symbol	Parameter	Min	Max	Unit	Condition	Note
$V_{OLC}$	Clock Output High Voltage	$V_{CC}-0.6$		V	-2.0mA	
$V_{OHC}$	Clock Output Low Voltage		0.4	V	+2.0mA	
$V_{IH1}$	Input Low Voltage	-0.3	0.8	V		
$V_{IL}$	Input High Voltage	2.2	$V_{CC}$	V		
$V_{OL}$	Output Low Voltage		0.4	V	$I_{LO}=2.0mA$	[5]
$V_{OH1}$	Output High Voltage	2.4		V	$I_{OH}=-1.6mA$	[4]
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$		V	$I_{OH}=-250\mu A$	[5]
$I_{CC1}$	Power Supply Current - 10MHz - 6MHz		50 30	mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$	[1]
$I_{CC2}$	Power Supply Current (STOP Mode)		10	$\mu A$	$V_{CC}=5V$	
$I_{CC3}$	Power Supply Current (IDLE1 Mode) - 10MHz - 6MHz		4 4	mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$	
$I_{CC4}$	Power Supply Current (IDLE2 Mode) - 10MHz - 6MHz		15 13	mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$	[1] [1]
$I_{LI}$	Input Leakage Current	-10	10	$\mu A$	$V_{IN}=0.4V$ to $V_{CC}$	[4]
$I_{LO}$	3-state Output Leakage Current in Float	-10	10	$\mu A$	$V_{OUT}=0.4V$ to $V_{CC}$	[2]

### Notes:

- [1] Measurements made with outputs floating.
- [2] A15-A0, D7-D0, /MREQ, /IORQ, /RD and /WR.
- [3]  $I_{CC2}$  Standby Current is guaranteed when the halt pin is low in STOP mode.
- [4] All Pins except XTAL1, where  $I_{LI} = \pm 25\mu A$ .
- [5] A15-A0, D7-D0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1 and /BUSACK.

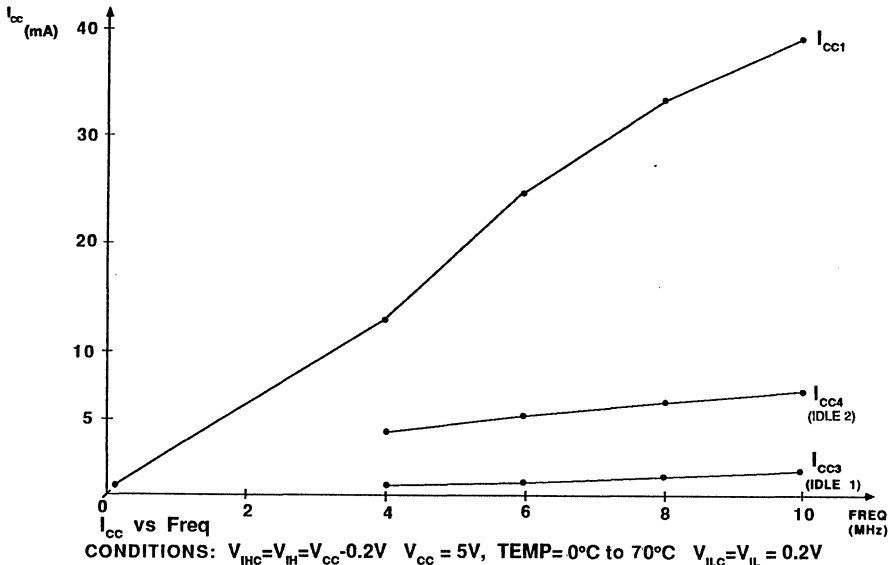


Figure 26 Z84C01 Typical  $I_{CC}$  vs Freq

---

## ELECTRICAL CHARACTERISTICS:

### ABSOLUTE MAXIMUM RATINGS

Voltage on Vcc with respect to Vss.....-0.3V to + 7V  
Voltages on all inputs with respect to Vss..-0.3V to Vcc + 0.3V  
Operating Ambient Temperature.....See Ordering Information  
Storage Temperature.....-65°C to + 150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

---

### Standard Test Conditions

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (OV). Positive current flows into the referenced pin.

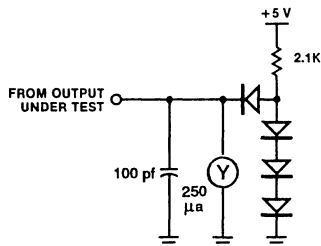
Available operating temperature ranges are:

$$E = -40^{\circ}\text{C to } +100^{\circ}\text{C}$$

Voltage Supply Range:  $+4.50\text{V} \leq V_{cc} \leq +5.50\text{V}$

All AC parameters assume a load capacitance of 100 pf. Add 10 ns delay for each 50 pf increase in load up to a maximum of 150 pf for the data bus and 100 pf for address and control lines. AC timing measurements are referenced to 1.5 volts (except for clock, which is referenced to the 10% and 90% points). Maximum capacitive load for CLK is 125 pf.

The Ordering Information section lists temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.







# Z84C50

Z80 RAM 80  
Z80 CPU/2K SRAM

JUNE 1989

## FEATURES

- Z80 CPU 2K Static RAM
- Wait State Generator for external memory
- Low power consumption
  - (TBD) Typ (5V, 10 MHz under RUN mode)
  - (TBD) Typ (5V, 10 MHz under IDLE1 mode)
  - (TBD) Typ (5V, 10 MHz under IDLE2 mode)
  - (TBD) Typ (5V under STOP mode)
- DC to 10 MHz operation (at  $5V \pm 10\%$ )
- Single 5V power supply (at  $5V \pm 10\%$ )
- Operating Temperature ( $0^{\circ}C$  to  $70^{\circ}C$ )
- On-chip clock generator
- In the HALT state, the following 4 modes are selectable:
  - RUN mode
  - IDLE 1 mode
  - IDLE 2 mode
  - STOP mode
- Powerful set of 158 instructions
- Power Interrupt function
  - Non-Maskable Interrupt terminal (NMI)
  - Maskable Interrupt terminal (INT)
- The following three modes are selectable:
  - 8080 compatible interrupt mode (interrupt by Non-Z80 family peripheral LSI-Mode 0)
  - Restart Interrupt (Mode 1)
  - Daisy-chain structure interrupt using Z80 family peripheral LSI (Mode 2)
- Built-in refresh circuit for dynamic memory
- Available in 40-pin DIP, 44-pin PLCC, and 44-pin QFP packages

## GENERAL DESCRIPTION

The Z84C50 is an 8-bit microprocessor integrated with 2K bytes of static memory and a clock generator/controller. The Z84C50 is targeted for a broad range of applications requiring a small amount of RAM. Additionally, the on-chip RAM can be accessed at a much higher rate than the external memory. This will significantly enhance performance, as the most commonly used and time critical software can be placed in on-chip memory.

Built into the Z84C50 is a control function and clock generator for the standby function in addition to: six paired general purpose registers, accumulator, flag registers, an arithmetic-and-logic unit, bus control, memory control and

timing control circuits. Also, an on-chip wait state generator can be used for automatically inserting wait states for external memory accesses.

The Z84C50 is fabricated with Zilog CMOS technology and molded in 40-pin DIP, 44-pin PLCC, and 44-pin QFP packages.



## PIN CONNECTIONS AND PIN FUNCTIONS

The pin connections and I/O pin names and brief functions of the Z84C50 are shown below.

**Pin Connections.** The pin connections of the Z84C50 are as shown in Figures 1 to 3.

**Pin Names and Functions.** I/O pin names and functions are as shown in Table 1.

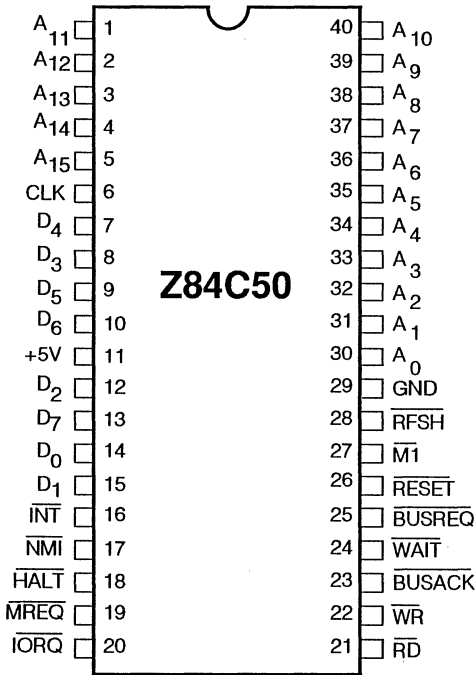


Figure 1. Pin Connections (top view) - DIP Package

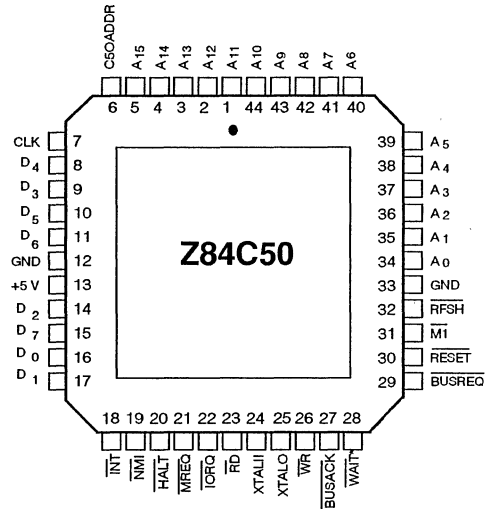


Figure 2. Pin Connections (top view) - PLCC Package

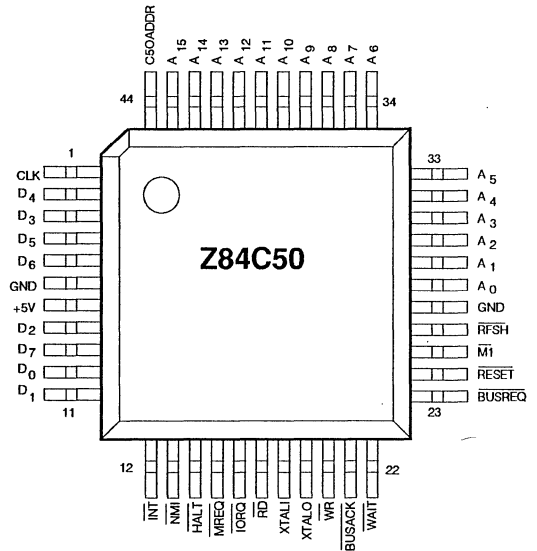


Figure 3. Pin Connections (top view) - QFP Package

**TABLE 1. PIN NAMES AND FUNCTIONS**

Pin	# of Pins	I/O 3-state	Function
A0-A15	16	I/O 3-state	16-bit address bus. Specifies addresses of memories and I/O to be accessed. During the refresh period, addresses (A0-A6) for refreshing are output. The bus is an input when the external master is accessing the on-chip RAM.
D0-D7	8	I/O 3-state	8-bit bidirectional data bus.
$\overline{\text{INT}}$	1	Input	Maskable Interrupt request signal. $\overline{\text{INT}}$ is normally wired-OR and requires an external pull up resistor for these applications.
$\overline{\text{NMI}}$	1	Input	Non-Maskable Interrupt request signal. This interrupt request has the higher priority than the maskable interrupt request.
$\overline{\text{HALT}}$	1	I/O 3-state	Halt signal. Indicates that the CPU has executed a Halt instruction. <i>Input during evaluation mode &amp; BUSACK.</i>
$\overline{\text{MREQ}}$	1	I/O 3-state	Memory Request signal. When an effective address for external memory access is on the address bus, '0' is output. When an external master is accessing the on-chip RAM, it is an input signal.
$\overline{\text{IORQ}}$	1	I/O 3-state	Indicating an I/O operation with I/O address on lower 8-bits (A0-A7) of the address bus. $\overline{\text{IORQ}}$ signal is output together with $\overline{\text{M1}}$ signal at time of interrupt acknowledge cycle to inform peripheral devices that the interrupt response vector may be put on the bus. Input during register accesses by external masters and also to initiate the evaluation mode during reset.
$\overline{\text{RD}}$	1	I/O 3-state	Read signal. Asserted for a period when MPU can receive data from a memory or peripheral LSI. When an external master is accessing the on-chip RAM, it is an input signal.
$\overline{\text{WR}}$	1	I/O 3-state	Write signal. This signal is output when data to be stored in a specified memory or peripheral LSI is on the MPU data bus. When an external master is accessing the on-chip RAM, it is an input signal.
$\overline{\text{BUSACK}}$	1	Output	Bus Acknowledge signal. In response to $\overline{\text{BUSREQ}}$ signal, this signal informs a peripheral LSI that the address bus, data bus, $\overline{\text{MREQ}}$ , $\overline{\text{IORQ}}$ , $\overline{\text{RD}}$ , and $\overline{\text{WR}}$ signals have been placed in the high impedance state.
$\overline{\text{WAIT}}$	1	I/O Wired-OR	Wait signal. $\overline{\text{WAIT}}$ signal is asserted by memory or peripheral LSI that is not ready for data transfer. As long as $\overline{\text{WAIT}}$ signal is active, MPU is continuously kept in the wait state. Driven out when the wait states are inserted by on-chip wait state generator.

**TABLE 1. PIN NAMES AND FUNCTIONS** (Continued)

Pin	# of Pins	I/O 3-state	Function
BUSREQ	1	Input	Bus Request signal. $\overline{\text{BUSREQ}}$ signal is a signal asserted by external masters to request placement of $\overline{\text{M1}}$ , $\overline{\text{HALT}}$ , address bus, data bus, $\overline{\text{MREQ}}$ , $\overline{\text{IORQ}}$ , $\overline{\text{RD}}$ , $\overline{\text{RFSH}}$ , and $\overline{\text{WR}}$ signals in the high impedance state. $\overline{\text{BUSREQ}}$ signal is normally wired-OR. In this case, a pull-up resistor is externally connected.
RESET	1	I/O 3-state	Reset signal. $\overline{\text{RESET}}$ signal is used for initializing MPU and other devices in the system. If it is an input, it must be kept in active state for a period of at least 3 clocks. During the power-up sequence, this input is sampled for 50 to 150 micro-seconds after the power supply passes through approx. 2.2V. If it is not active during this window, Z84C50 will drive the power on reset for 25 to 75 msec.
$\overline{\text{M1}}$	1	Input Output	Signal indicating an op code fetch. Also asserted during the RETI cycle and interrupt acknowledge cycle (along with the $\overline{\text{IORQ}}$ signal). Input during the on-chip accesses by external masters and during evaluation mode. <i>Input during evaluation and bus acknowledge modes.</i>
XTALI	1	I/O	Crystal Oscillator connecting terminals. A parallel resonant crystal is recommended. A crystal presence is automatically detected by the Z84C50. The input signal is divided by 1 or 2, depending upon bit 4 of the Control Register. The oscillator is used in either case. Also, all the power down modes are available in the 'halt' state. Those pins are not available on 40-pin DIP version.
XTALO	1	3-state	
CLK	1	I/O	The single-phase clock generated by the internal oscillator is output on the CLK pin. When this crystal is not installed the pin becomes an input for connecting the external clock. This is always the case for the 40-pin DIP device. The oscillator and clock divider are bypassed. Only the IDLE2 and RUN halt modes are applicable in this case.
C50ADDR	1	Output	This signal is asserted by the Z84C50 when the on-chip 2K static RAM and the I/O registers are accessed by the on-chip CPU or the external master. This signal is not available on 40-pin DIP version.
VCC	1	Power	+5V
VSS	2	GND	0V
RFSH	1	I/O 3-state	$\overline{\text{RFSH}}$ together with the $\overline{\text{MREQ}}$ , indicates that the lower seven bits of system's address bus can be used as a refresh address to the system's dynamic memories. Input signal when the wait-state generator is to be used by external BUS master.

Note: Please make sure that in BUS-acknowledge modes,  $\overline{\text{M1}}$ ,  $\overline{\text{RFSH}}$  and  $\overline{\text{HALT}}$  become input while on Z84C00 these are output pins.

## FUNCTIONAL DESCRIPTION

The system configuration, functions and basic operation of the Z84C50 are described here.

**Block Diagram.** The block diagram of the internal configuration is shown in Figure 4.

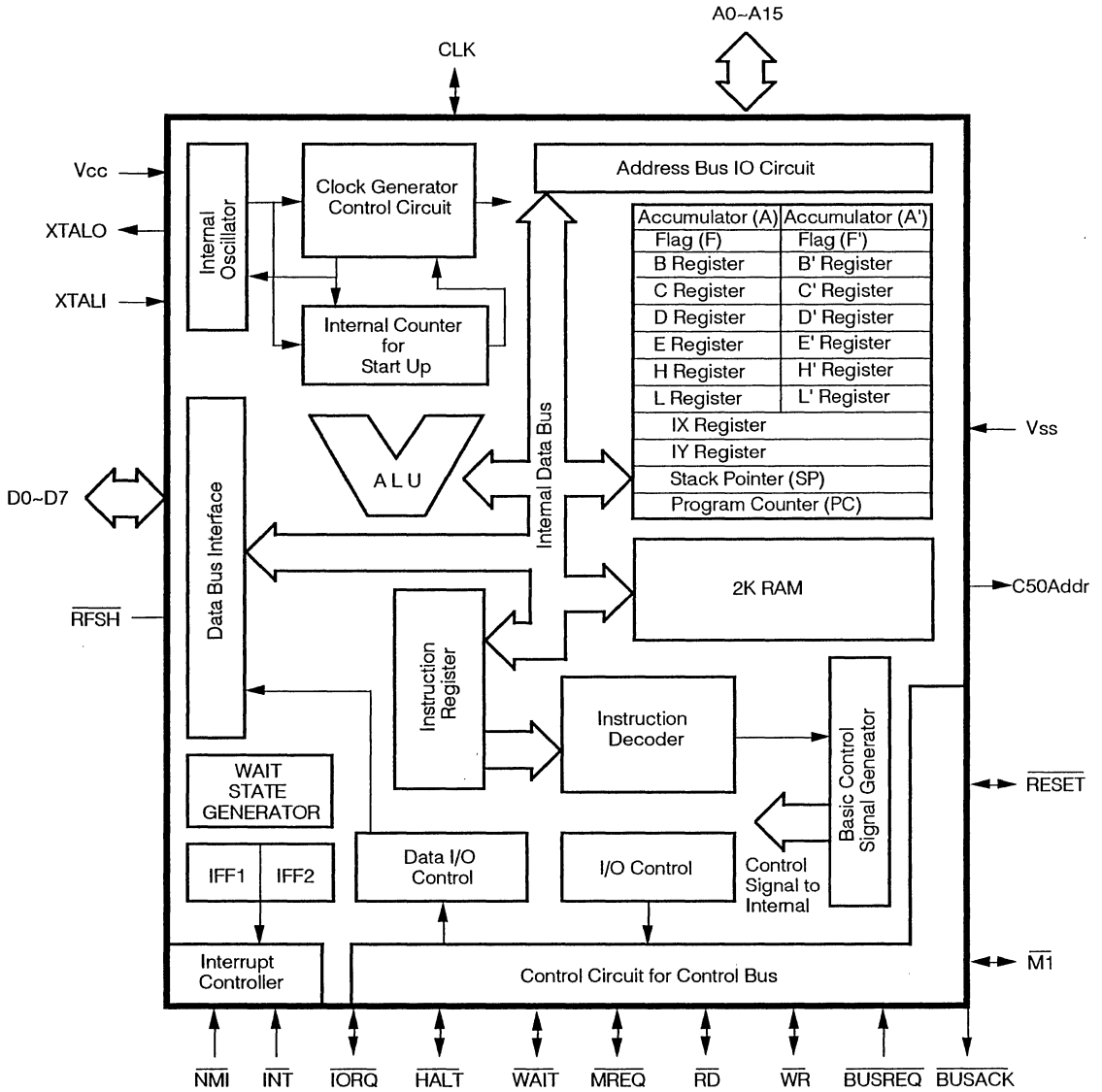


Figure 4. Block Diagram

## On-Chip Memory

The Z84C50 has 2K bytes of on-chip memory. This memory is mappable to any 2K boundary of the Z80 memory space by software control. The memory is accessible from the local CPU or the external masters via the Z80 bus request/bus grant handshake. When this memory is accessed, the C50ADDR output signal is asserted. This signal may be used by external logic to control external memory. The on-chip memory is normally disabled on power-up. On power-up reset, if  $\overline{\text{BUSREQ}}$  precedes the de-assertion of  $\overline{\text{RESET}}$  (rising edge), the RAM is enabled and one NOP instruction is automatically executed. This allows power-up loading of this RAM from external masters. The automatic execution of one NOP instruction prevents any possibility of executing instructions from external memory. For this case, it is recommended that a NOP instruction be at Location 0000H of the program.

The on-chip memory is always accessed with no wait states. When the external master is accessing RAM, the address lines are input to the chip, while the direction of the data lines is reversed from normal external memory accesses by the local CPU. Thus, both the address and data lines are bi-directional.

The base address of this on-chip memory is selectable to any 2K boundary of the Z80 memory space. This is done by programming bits 0-4 of the Memory Page Address Register. If the on-chip RAM is enabled by setting bit 5 of this register, these 5 bits provide the address lines A11-A15 for the on-chip memory accesses.

## Wait State Generator

The Z84C50 has a Wait State Generator for inserting wait states for external memory accesses. The on-chip memory does not require any wait states. The wait state generation is handled separately for the op-code fetches as the transaction timing for the Z80 op-code fetch is tighter. The user can set the "op-code fetch extension" bit (bit 5) of the Control Register (CR) thereby causing the Wait State Generator to insert one additional wait state. The bits 0-1 of the Control Register are set to specify the wait states for the normal external memory accesses. The following table shows the wait states for various cycles:

Table 2. Wait States (Memory)

CR (5)	CR(1)	CR(0)	Op code Fetch	Data
0	0	0	0	0
0	0	1	1	1
0	1	0	2	2
0	1	1	3	3
1	0	0	1	0
1	0	1	2	1
1	1	0	3	2
1	1	1	4	3

## Oscillation Detection Circuit

The PLCC or QFP versions of Z84C50 allows crystal input (XTALI, XTALO) or system clock input. If crystal input is provided, an internal oscillator is used to generate clocks for the Z80 as well as external devices. On power-up, it comes up in divide by 2 mode. If external clock (system clock) is provided on the CLK pin (or on 40-pin DIP version), the oscillator and the divide by 2 circuitry are bypassed. If the external clock or crystal input is provided on the XTAL pins, the internal oscillator is used and the divide-by-2 circuit is activated depending upon bit 4 of the Control Register. Also, the power-down modes of the 84C50 vary based on whether the clock is crystal derived or is the system clock (CLK) Pin. If the clock is crystal derived all of the modes in "halt" state are available. If the external system clock is provided on the CLK pin (and 40-pin DIP version), only the IDLE1 mode is applicable and the internal CPU is stopped if this mode is selected.

## Power-on Reset

The Z84C50 has a power-up reset circuit. If  $\overline{\text{RESET}}$  input is asserted within 150 microseconds of power-up (power supply above 2.2V), it is considered an input signal. Otherwise, an internal  $\overline{\text{RESET}}$  signal is generated after the power is stabilized and output on the  $\overline{\text{RESET}}$  pin for external use. The  $\overline{\text{RESET}}$  signal is asserted for 25 to 75 msec. After termination of the "power-on reset" cycle the pin will revert back to an input. It can also become an output again should the next extension operation not be disabled. If not enabled, it will continue as an input.

The 84C50 registers are initialized on power-up reset as follows: Control Register :x010 1111 and Memory Page Register: xx00 0000. The "x" represents unused bits and thus the values should be masked off. Other bits in the Control Register indicate that on reset the chip comes up with: 3 memory wait states, run mode, divide by two clock, op code extension bit enabled, and reset output enabled. The Memory Page Address Register is initialized to: zero page base address, and RAM disabled.

## Evaluation Mode

The Z84C50 has a built-in evaluation mode feature which allows the users to utilize standard Z80 development systems very conveniently. During the power-up reset sequence, if the  $\overline{\text{IORQ}}$  signal is asserted by the user (input), the Z84C50 enters into an evaluation (or development) mode. The  $\overline{\text{IORQ}}$  signal must meet the setup and hold times with respect to the rising edge of  $\overline{\text{RESET}}$ . In this mode, the internal CPU is immediately disconnected from the internal bus and all 3-state signals are tri-stated. The tri-stated are:  $\overline{\text{A0-A15}}$ ,  $\overline{\text{D0-D7}}$ ,  $\overline{\text{HALT}}$ ,  $\overline{\text{MREQ}}$ ,  $\overline{\text{IORQ}}$ ,  $\overline{\text{RD}}$ ,  $\overline{\text{WR}}$ ,  $\overline{\text{M1}}$ , and  $\overline{\text{RFSH}}$ . This allows the development system CPU to take over and use the internal RAM and I/O registers of the 84C50 like the CPU was on-chip. The wait state generator is utilized as before for external memory accesses only. The 84C50 behaves similarly to the situation where during regular operation, the  $\overline{\text{BUSREQ}}$  signal is asserted by an external master causing all 3-state signals to be tri-stated by the 84C50 after one clock delay. The  $\overline{\text{BUSREQ}}$  approach was not used for the evaluation mode to avoid significant external circuitry, in order to work around the one clock delay before the external CPU can use the bus for 84C50 accesses.

## Clock Generator/Oscillator

The PLCC and QFP versions of the Z84C50 has a built-in system clock generator for CMOS Z80 in addition to the standard functions of the Z84C00 MPU. The explanation is provided in the following section with emphasis placed on the halt function relative to the clock generator, which is an additional function. The internal register group, reset and interrupt function are identical to those of the Z84C00. For details, please refer to the data sheet for the Z84C00.

## PROGRAMMING

The Z84C50 has two internal 8-bit registers to support its functions. These two registers are called the Control Register and the Memory Page Address register. I/O ports "EE" and "EF" of the Z80 I/O space are allocated for the two registers. The following is a brief description of the software model:

### Control Register ("EE")

#### Bits 1-0. Memory Wait States.

- 00 - No Wait State
- 01 - 1 Wait State
- 10 - 2 Wait States
- 11 - 3 Wait States

On power-on reset, these bits are "11" causing 3 wait states until changed by software.

## Generating the System Clock

The PLCC and QFP versions of the Z84C50 have a built-in oscillation circuit and the required clock can be easily generated by connecting a crystal to the external terminals (XTALI, XTALO). Clock output is the same frequency or half the frequency of the external frequency. Examples of oscillator connections are shown in Figure 5. The values will change with crystal frequency.

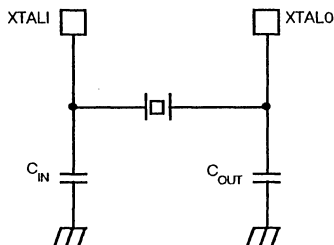


Figure 5a. Example of Oscillator Connection and Constant

$C_{IN}$	$C_{OUT}$
22 PF	33 PF

Figure 5b. Example of Oscillator Connection and Constant

**Bits 3-2. MS2/MS1 Halt Mode Selection inputs.** One of 4 halt modes is selected as follows:

- 00 - IDLE1 Mode: Oscillator running, CLK and CPU operations are stopped.
- 01 - STOP Mode: Oscillator, CLK, and CPU operations are stopped.
- 10 - IDLE2 Mode: Oscillator and CLK running, CPU operations are stopped.
- 11 - RUN Mode: CPU continues the operation and supplies clock to the outside.

The detailed operations and timing diagrams for these modes are given elsewhere in this document.

On-power-on reset these bits are "11".

**Bit 4. Clock Divide-by-One Option.** When set to 1, the external clock input on the XTAL pins is divided by 1 as the divided by 2 circuitry is bypassed.

When cleared to 0, the input clock on the XTAL pins is divided by two. The oscillator is used in either case.

When the external system clock is input on the CLK pin, this bit is ignored. Also the on-chip oscillator and divide by two circuits are bypassed.

On Power-on reset, this bit is "0".

**Bit 5. Op Code Fetch Extension Enable.** When set to 1, this bit causes the generation of an additional wait state for external op code fetches. This wait state is in addition to the wait states specified by bits 1-0 for other external memory accesses. On power-on reset, this bit is 1.

**Bit 6. Reset Output Disable.** This bit controls whether the RESET signal is driven out when reset input is used to take the 84C50 out of the "halt" state. RESET pulse is driven out unless this bit is set.

**Bit 7. Unused.** This bit should be set to "0" when writing into this register. When the register is read, this bit is read as "0".

#### Memory Page Address Register ("EF")

**Bits 4-0. Page Base Address.** These 5 bits select the 2K boundary where the base address of the on-chip 2K RAM is located. This essentially provides the address A15-A11 for the on-chip memory accesses. There are no wait states generated for the on-chip memory accesses. These bits are cleared on reset.

#### Bit 5. RAM Enable

When this bit is set to 1, the on-chip RAM is enabled. This bit is normally "0" on power-up reset except when the BUSREQ precedes the de-assertion of RESET.

#### Bits 6-7. Unused.

These bits should be set to "0" when writing to this register. When the register is read, these bits are read as "0".

## OPERATION MODES

There are four kinds of operation modes available for the Z84C50 in connection with clock generation; RUN Mode, IDLE1/2 Modes and STOP Mode.

The operation mode is effective when the halt instruction is executed. Restart of MPU from the stopped state under IDLE1/2 Mode or STOP Mode is effected by inputting either RESET or interrupt(INT or NMI).

**Table 3. Clock Generating Operation Modes**

Operation Mode	CR(3)	CR(2)	Description at HALT State
Run Mode	1	1	MPU continues the operation and supplies clock to the outside continuously.
IDLE 1 Mode	0	0	The internal oscillator's operation is continued. Clock (CLK) output are as well as internal operations stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.
IDLE 2 Mode	1	0	The internal oscillator's operation and clock (CLK) output are continued but the internal operations are stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.
STOP Mode	0	1	All operations of the internal oscillator, clock (CLK) output, and internal operation are stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.

All of the operating modes listed above are valid with crystal input. For the external clock input on the CLK pin (and 40-pin DIP version); only the IDLE2 and RUN modes are applicable.

### Start-up Time at Time of Restart (STOP Mode)

When the MPU is released from the halt state by accepting an interrupt request, it executes an interrupt service routine. Therefore, when an interrupt request is accepted, the MPU starts generating an internal system clock and clock output after a start-up time by the internal counter  $[(2^{14} + 2.5) T_{cC} (T_{cC}: \text{Clock Cycle})]$  to obtain a stabilized oscillation for MPU operation.

Further, in case of the restart by  $\overline{\text{RESET}}$  signal, the internal counter does not operate for a quick operation at time of power-on.

### Basic Timing

The basic timing is explained here with emphasis placed on the halt function relative to the clock generator. The following items are identical to those for the Z84C00. Refer to the data sheet for the Z84C00.

- Operation code fetch cycle
- Memory read/write operation

- Input/output operation
- Bus request/acknowledge operation
- Maskable interrupt request operation
- Non-maskable interrupt request operation
- Reset operation

### Operation When HALT Instruction is Executed

When the MPU fetches a halt instruction in the operation code fetch cycle, the HALT signal goes active (low) in synchronous with the falling edge of T4 before the peripheral LSI and MPU stops the operation. After this the system clock generating operation after this differs depending upon the operation mode (RUN Mode, IDLE1/2 Mode or STOP Mode). If the internal system clock is running, the MPU continues to execute NOP instructions even in the halt state.

**RUN Mode** (MS1=1, MS2=1). Shown in Figure 6 is the basic timing when the halt instruction is executed in RUN Mode.

In RUN Mode, system clock ( $\emptyset$ ) in MPU and clock output (CLK) are not stopped, even after the halt instruction is executed. Therefore, until the halt state is released by the interrupt signal (NMI or INT) or RESET signal, MPU continues to execute NOP instructions.

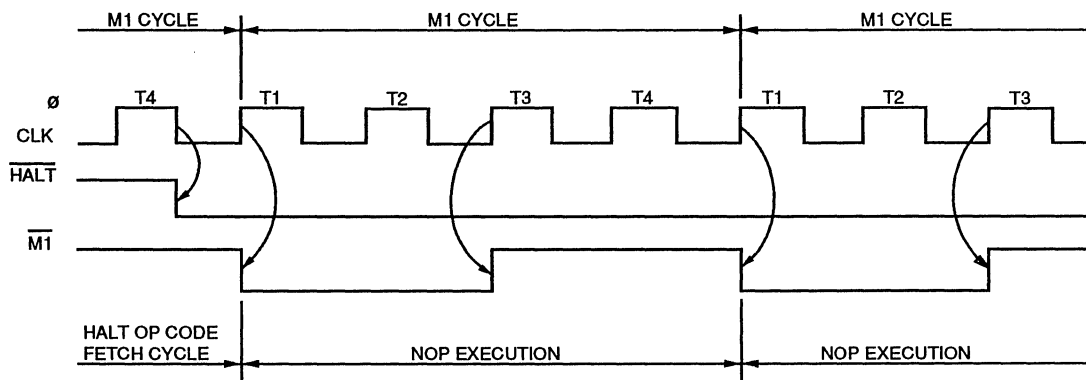
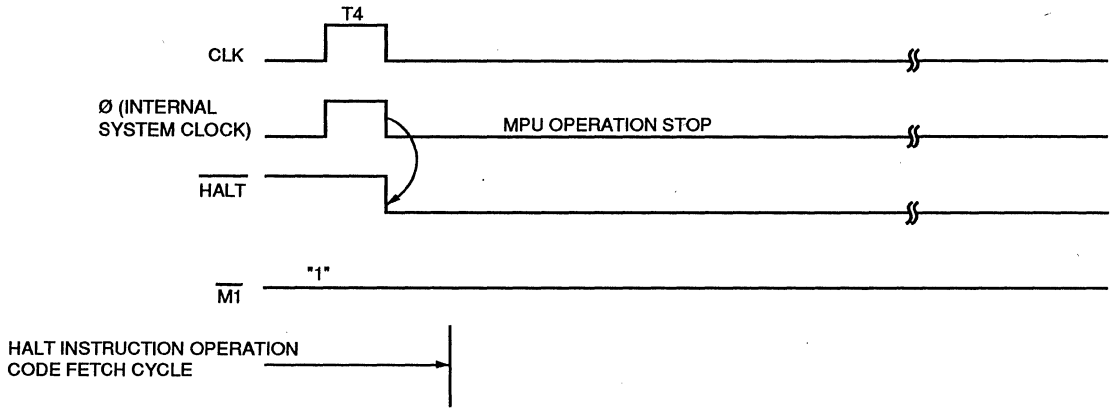


Figure 6. Timing of RUN Mode (at Halt Command Execution)



**IDLE1 Mode** (MS1=0, MS2=0). Shown in Figure 7 is the basic timing when the halt instruction is executed in IDLE1 Mode.

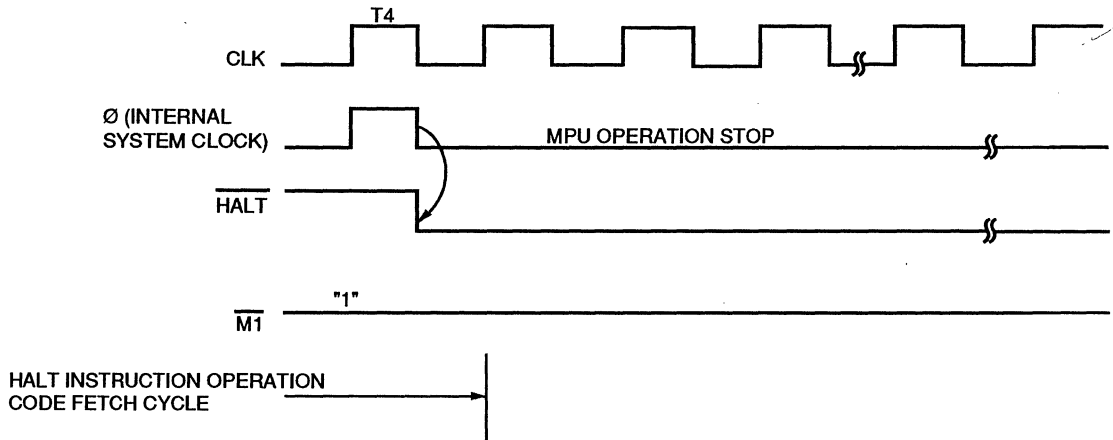
In IDLE1 Mode, system clock ( $\emptyset$ ) in the MPU and clock output (CLK) are stopped, and MPU stops its operation after the halt instruction is executed. However, the internal oscillator continues to operate.



**Figure 7. IDLE1 Mode Timing (at Halt Instruction Execution)**

**IDLE2 Mode** (MS1=0, MS2=1). Shown in Figure 8 is the basic timing when the halt instruction is executed in IDLE2 Mode.

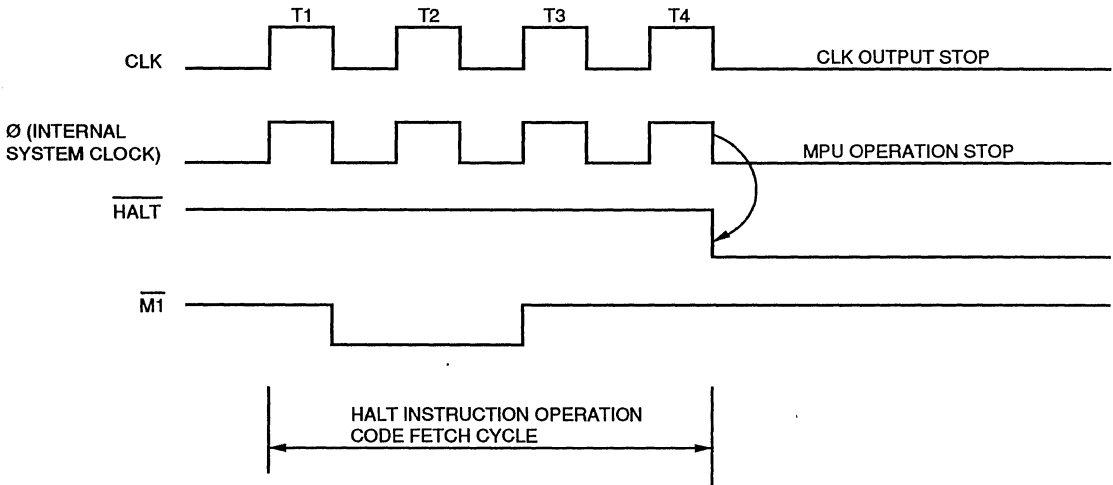
In IDLE2 Mode, system clock ( $\emptyset$ ) in MPU is stopped and MPU stops its operation after the halt instruction is executed. However, the internal oscillator and clock output (CLK) to the outside of MPU continues to operate.



**Figure 8. IDLE2 Mode Timing (at Halt Instruction Execution)**

**STOP Mode** (MS1=1, MS2=0). Shown in Figure 9 is the basic timing when the halt instruction is executed in STOP Mode.

In STOP Mode, internal operation and the internal oscillator are stopped after the halt instruction is executed. Therefore, system clock ( $\emptyset$ ) in the MPU and clock output (CLK) external to the MPU, are stopped.



**Figure 9. Stop Mode Timing (at Halt Instruction Execution)**

**Release from Halt State**

The halt state of MPU is released when "0" is input to RESET signal and MPU is reset or an interrupt request is accepted. An interrupt request signal is sampled at the leading edge of the last clock cycle (T4 state) of NOP instruction. In case of the maskable interrupt, interrupt will be accepted by an active  $\overline{INT}$  signal ("0" level). Also the interrupt enable flip-flop must have been set to "1". The accepted interrupt process is started from the next cycle.

Further, when the internal system clock is stopped (IDLE1/2 Mode, STOP Mode), it is necessary to first restart the internal system clock. The internal system clock is restarted when RESET or the interrupt signal (NMI or INT) is asserted.

**RUN Mode** (MS1, MS2=1). The halt release operation by acceptance of interrupt request in RUN Mode is shown in Figure 10.

In RUN Mode the internal system clock is not stopped, and therefore, if the interrupt signal is recognized at the rise of T4 state of the continued NOP instruction, MPU will execute the interrupt process from the next cycle.

The halt release operation, by resetting MPU in RUN Mode, is shown in Figure 11. After reset, MPU will execute an instruction starting from address 0000H. However, in order to reset MPU, it is necessary to keep RESET signal at "0" for a least 3 clocks. In addition, if RESET signal becomes "1", after the dummy cycle for at least two T states, MPU executes an instruction from address 0000H.

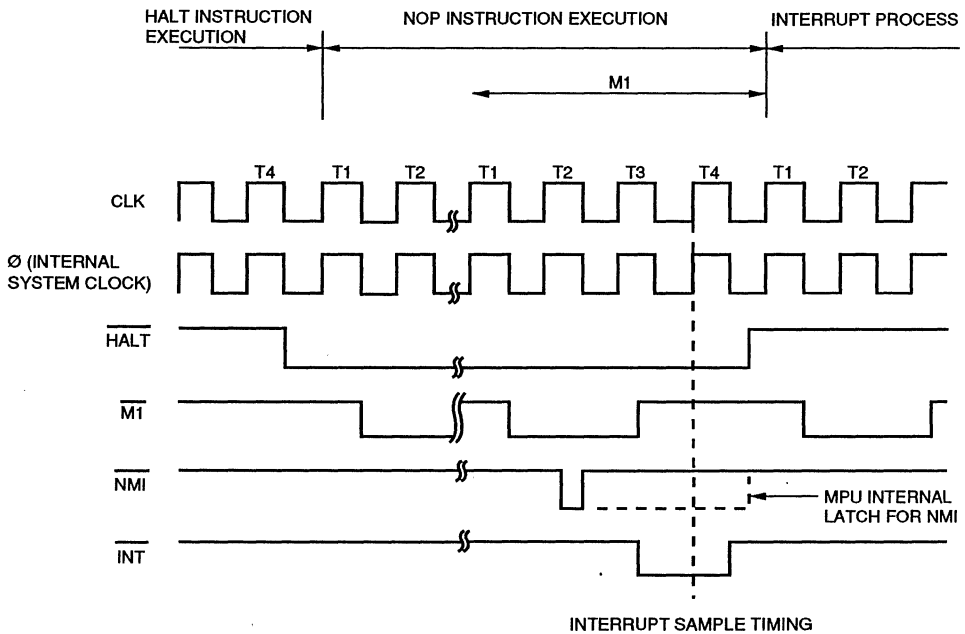


Figure 10. Halt Release Operation Timing by Interrupt Request Signal in RUN Mode

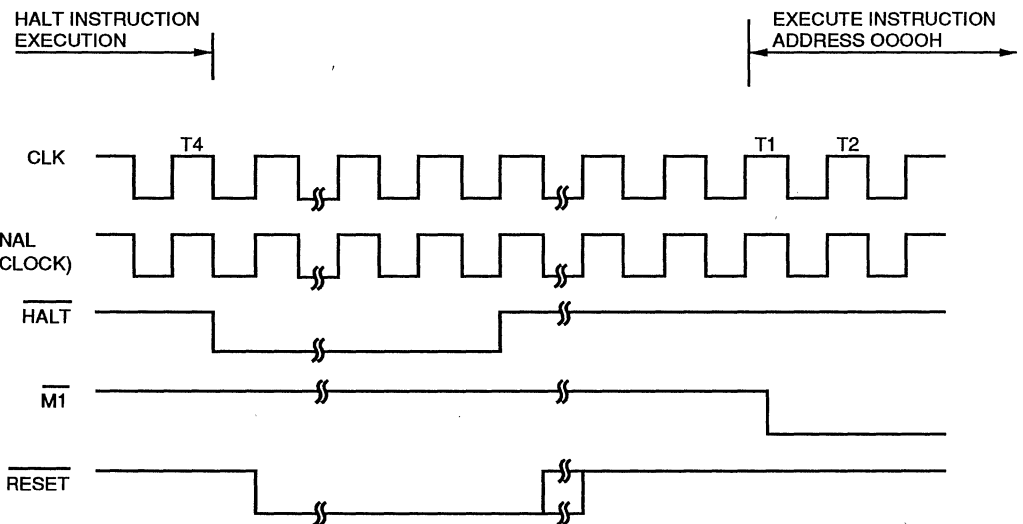


Figure 11. Halt Release Operation Timing by reset in RUN Mode

**IDLE1 Mode** (MS1=0, MS2=0), **IDLE2 Mode** (MS1=0, MS2=1). The halt release operation by interrupt signal in IDLE1 Mode is shown in Figure 12 (a) and in IDLE2 Mode in Figure 12 (b).

When receiving  $\overline{\text{NMI}}$  or  $\overline{\text{INT}}$  signal, MPU starts the internal system clock operation. In IDLE1 Mode, MPU starts clock output to the outside at the same time.

The operation stop of MPU in IDLE/2 Mode is taking place at "0" level during T4 state in the halt instruction operation code fetch cycle. Therefore, after being restarted by the

interruption signal, MPU executes one NOP instruction and samples an interrupt signal at the rise of T4 state during the execution of this NOP instruction, and executes the interrupt process from next cycle.

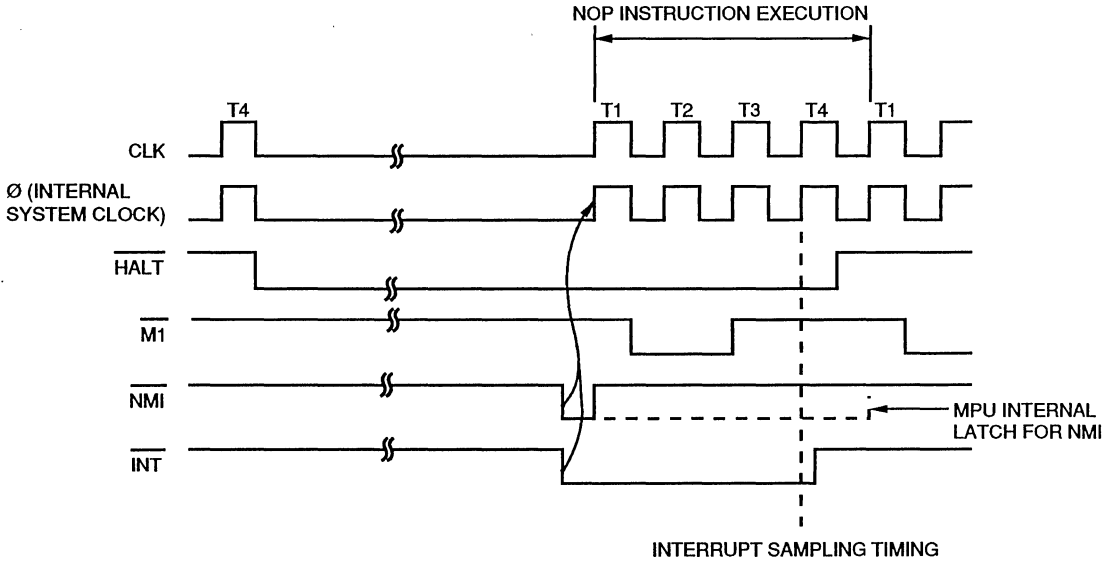


Figure 12. (a) IDLE1 Mode

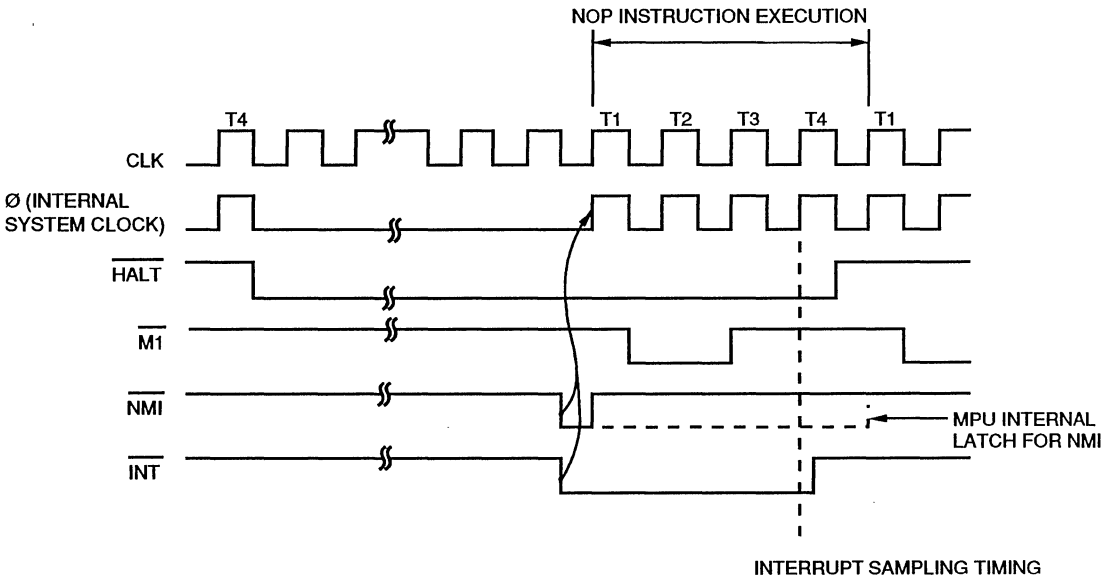


Figure 12. (b) IDLE2 Mode

Figure 12. Halt Release Operation Timing by Interrupt Request Signal in IDLE1/2 Mode

If no interrupt signal is accepted during the execution of the first NOP instruction after the internal system clock is restarted, MPU is not released from the halt state and is placed in IDLE1/2 Mode again at "0" level during T4 state of the NOP instruction, stopping the internal system clock. If INT signal is not at "0" level at the rise of T4 state, no interrupt request is accepted.

When the  $\overline{\text{RESET}}$  signal at "0" level is input into MPU, the internal system clock is restarted and MPU will execute an instruction stored in address 0000H.

At the time of  $\overline{\text{RESET}}$  signal input, it is necessary to take the same care as that in resetting MPU in RUN Mode.

The halt release operation, by resetting MPU in IDLE1 Mode, is shown in Figure 13 (a) and that in IDLE2 Mode in Figure 13 (b).

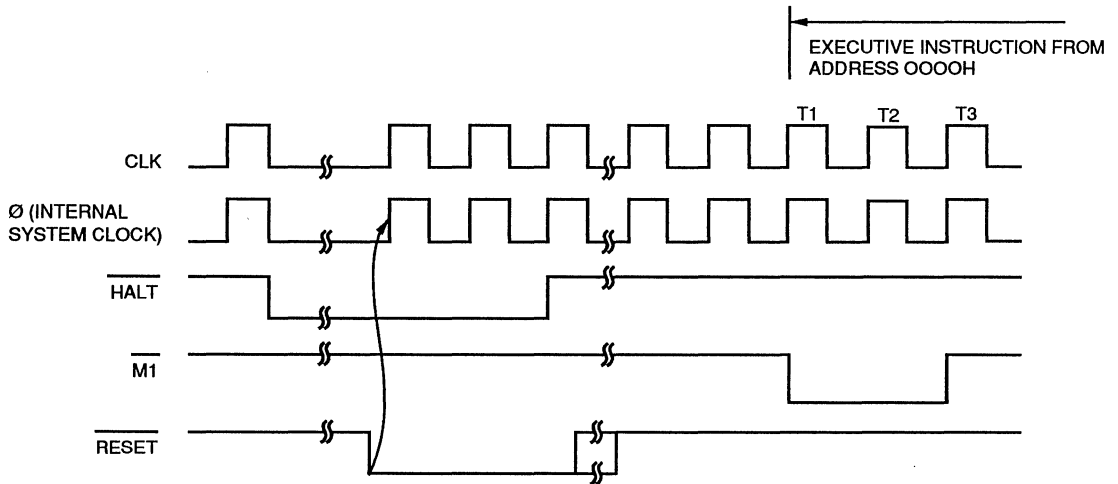


Figure 13. (a) IDLE1 Mode

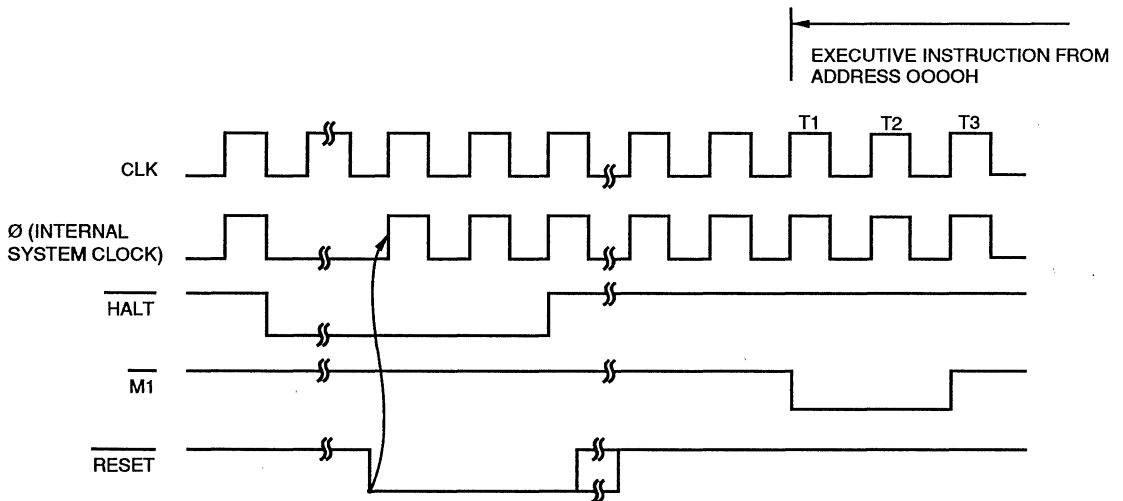


Figure 13. (b) IDLE2 Mode

Figure 13. Halt Release Operation Timing by Reset in IDLE1/2 Mode

**STOP Mode** (MS1=1, MS2=0). The halt release operation by interrupt signal in STOP Mode is shown in Figure 14.

When MPU receives an interrupt signal, the internal oscillator is restarted. In order to obtain stabilized oscillation, the internal system clock and clock output to the outside are started after a start-up time of  $[(2^{14} + 2.5) T_{CC} (T_{CC}: \text{Clock Cycle})]$  by the internal counter.

MPU executes one NOP instruction after the internal system clock is restarted and at the same time, sampling an interrupt signal at the rise of T4 state during the execution of this NOP instruction. If the interrupt signal is accepted, MPU executes the interrupt process operation from next cycle.

At time of interrupt signal input, it is necessary to take the same care as that in the interrupt signal input in IDLE1/2 Mode. The halt release operation by MPU resetting in STOP Mode is shown in Figure 15.

When the  $\overline{\text{RESET}}$  signal at "0" level is input into MPU, the internal oscillator is restarted. However, since it performs a quick operation at time of power-on, the internal counter does not operate. Therefore, the operation may not be carried out properly due to an unstable clock immediately after the signal in STOP Mode. Thus, it is necessary to hold the  $\overline{\text{RESET}}$  signal at "0" level for sufficient time. In the 84C50, the external  $\overline{\text{RESET}}$  signal requirements are the same as in the power-up reset. Internally, the reset pulse is stretched by  $2^{14} + 16$  cycles to allow the oscillator to settle down. This elongated reset signal is driven out of the 84C50 on the reset pin if bit 6 of the control register has not been set. Setting bit 6 disables the driving out of reset.

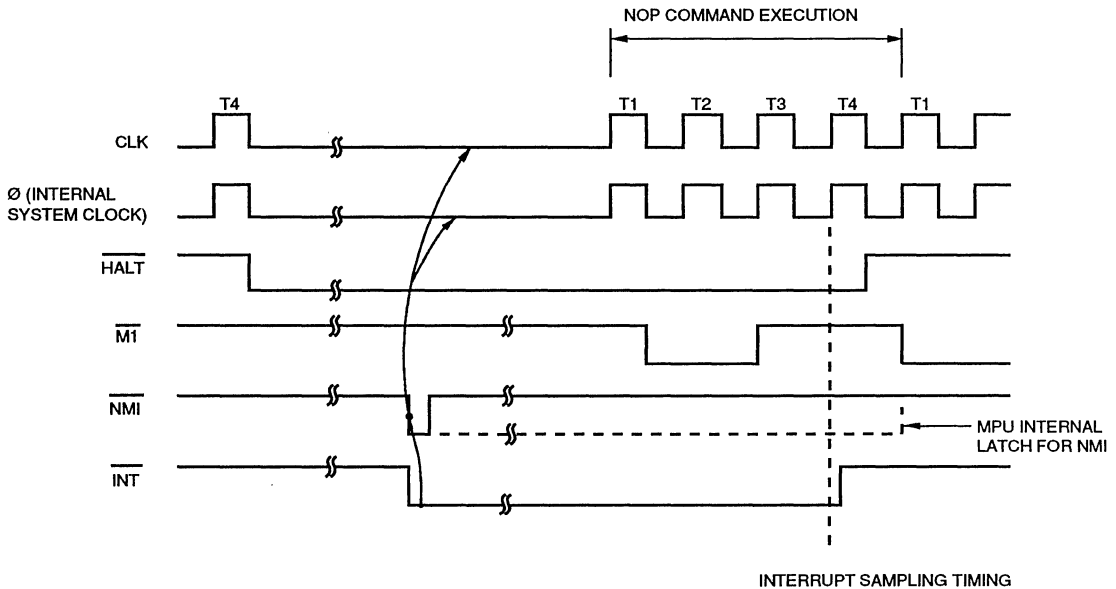


Figure 14. Halt Release Operation Timing by Interrupt Request Signal in STOP Mode

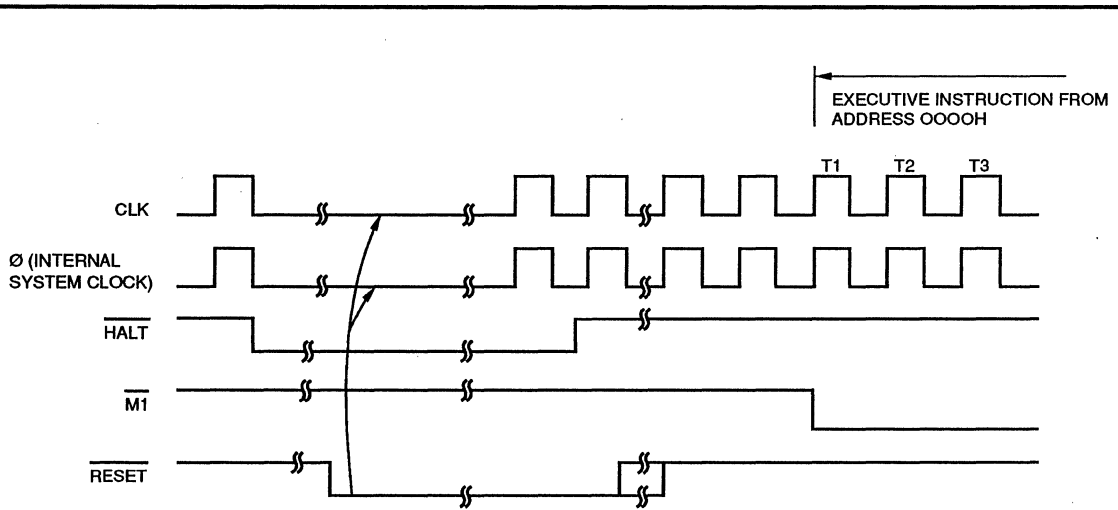


Figure 15. Halt Release Operation Timing by reset in STOP Mode

### Instruction Set

Instruction set of the Z84C50 is the same as that for the Z84C00. For details refer to the data sheet for the Z84C00.

### CPU TIMING

Timing Diagrams. The Z84C50 CPU executes instructions by proceeding through a specific sequence of operations:

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

The basic clock period is referred to as a Time or Cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

Instruction Opcode Fetch. The CPU places the contents of the Program counter (PC) on the address bus at the start of the cycle (Figure 16). Approximately one-half clock cycle later,  $\overline{MREQ}$  goes active. When active,  $\overline{RD}$  indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the  $\overline{WAIT}$  input with the falling edge of clock state T2. During clock states T3 and T4 of an  $\overline{M1}$  cycle, dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction.

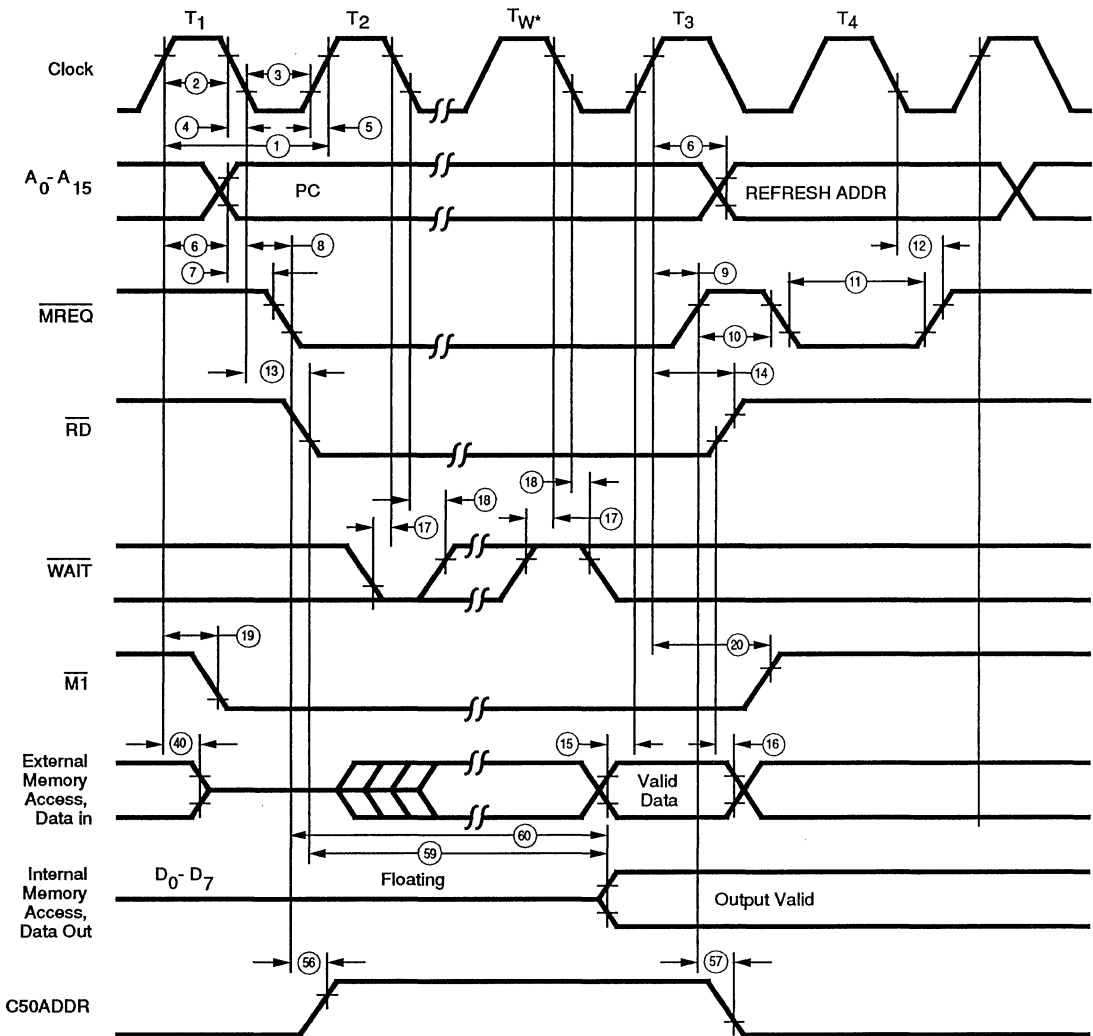


Figure 16. Instruction Op code Fetch



Memory Read or Write Cycles. Figure 17 shows the timing of memory read or write cycles other than an opcode fetch (M1) cycle. The MREQ and RD signals function exactly as in the fetch cycle.

In a memory write cycle,  $\overline{\text{MREQ}}$  also becomes active when the address bus is stable. The  $\overline{\text{WR}}$  line is active when the data bus is stable, so that it can be used directly as an  $\text{R}/\overline{\text{W}}$  pulse to most semiconductor memories.

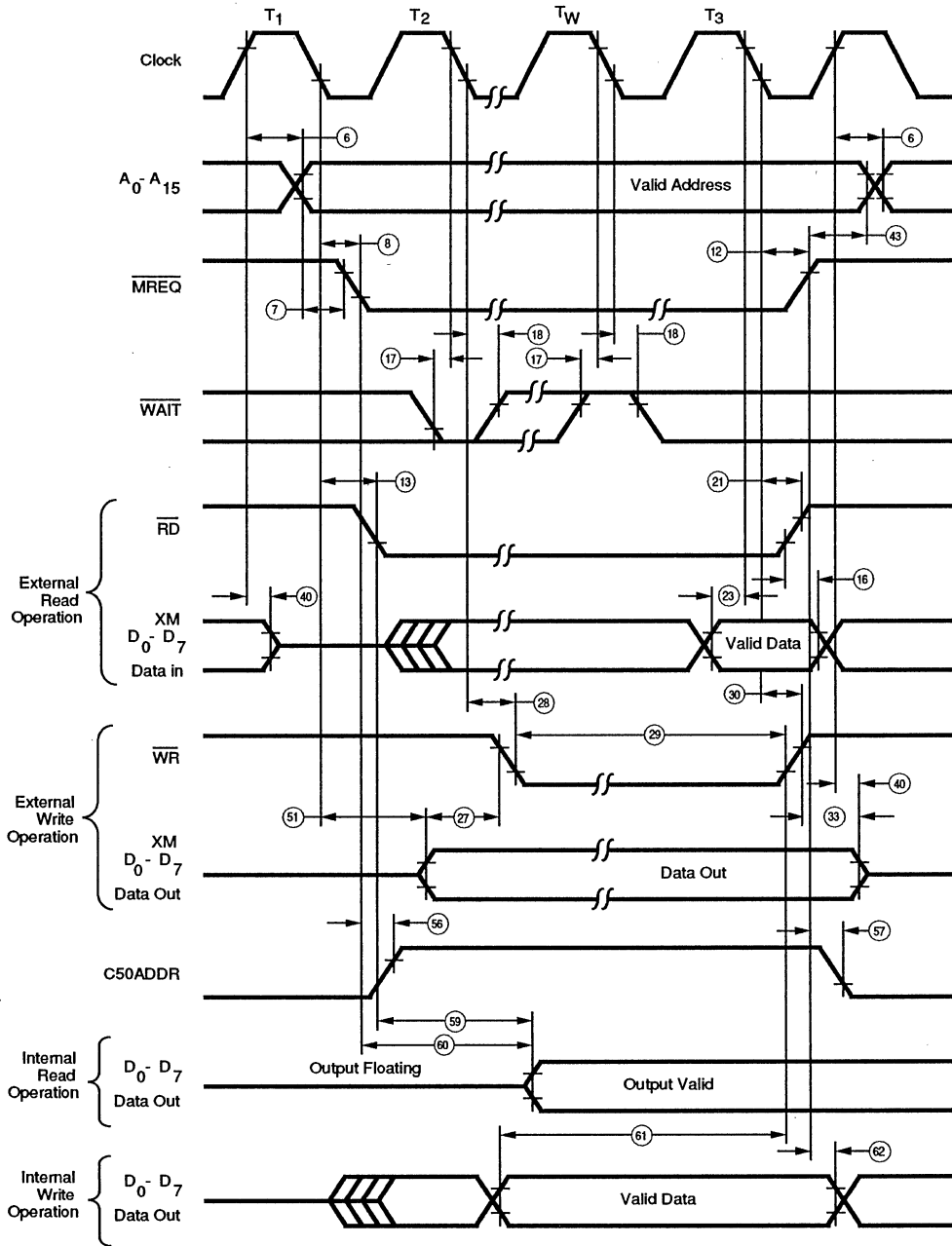
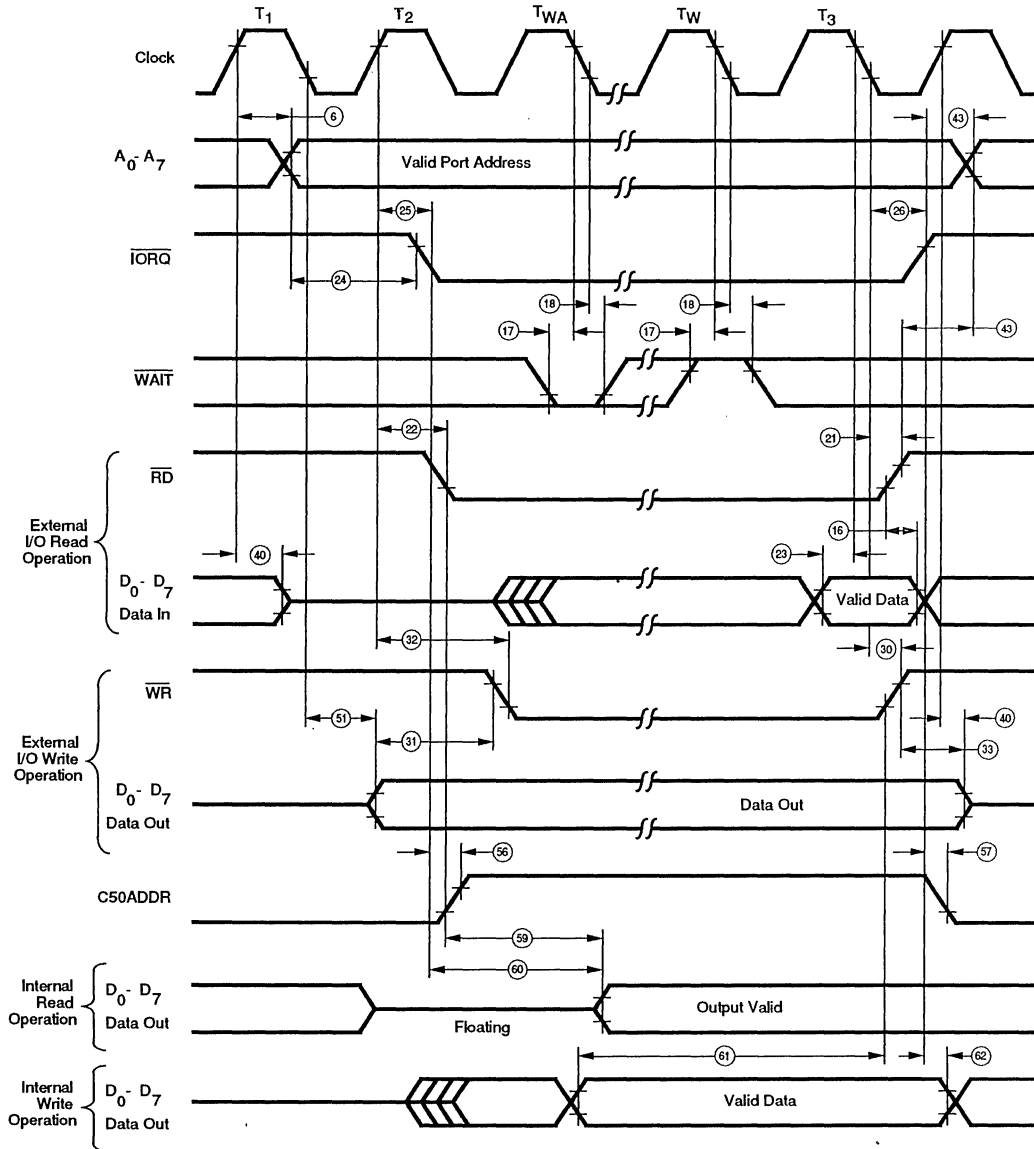


Figure 17. Memory Read or Write Cycles

Input or Output Cycles. Figure 18 shows the timing for an I/O read or I/O write operation. During I/O operation, the CPU automatically inserts a single Wait state ( $T_{WA}$ ). This

extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.

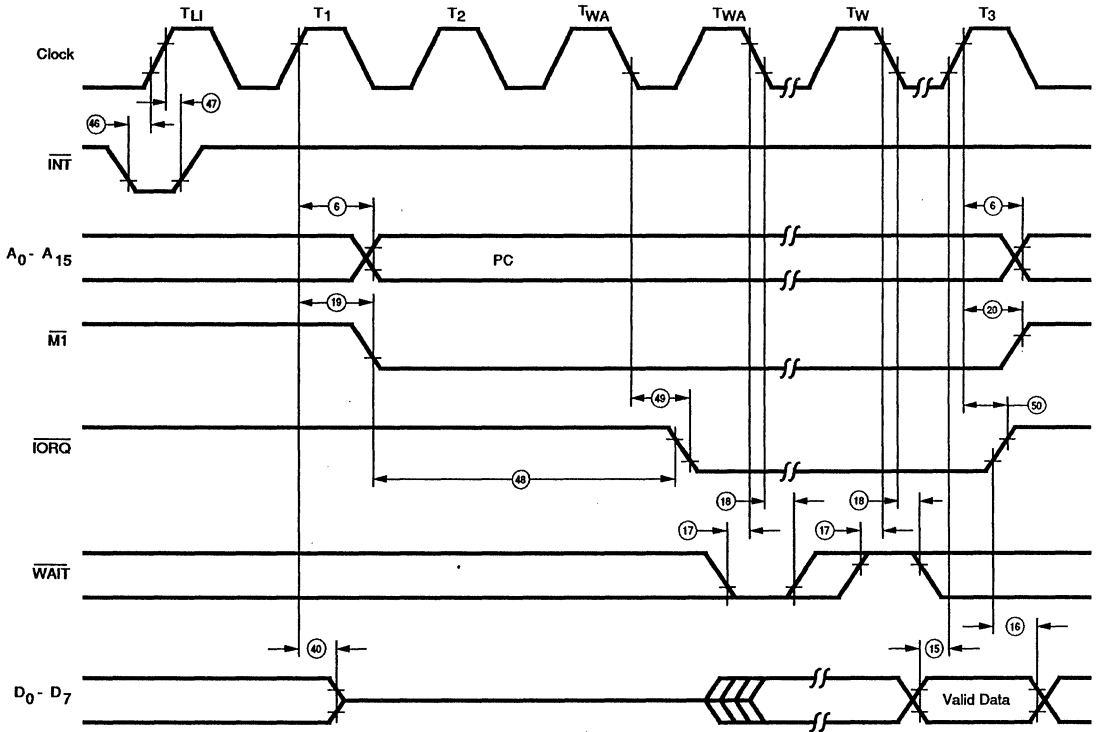


Note:  $T_{WA}$  = One wait cycle automatically inserted by CPU.

Figure 18. Input or Output Cycles

Interrupt Request/Acknowledge Cycle. The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 19). When an interrupt is accepted, a special  $\overline{M1}$  cycle is generated.

During this  $\overline{M1}$  cycle,  $\overline{IORQ}$  becomes active (instead of  $\overline{MREQ}$ ) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.

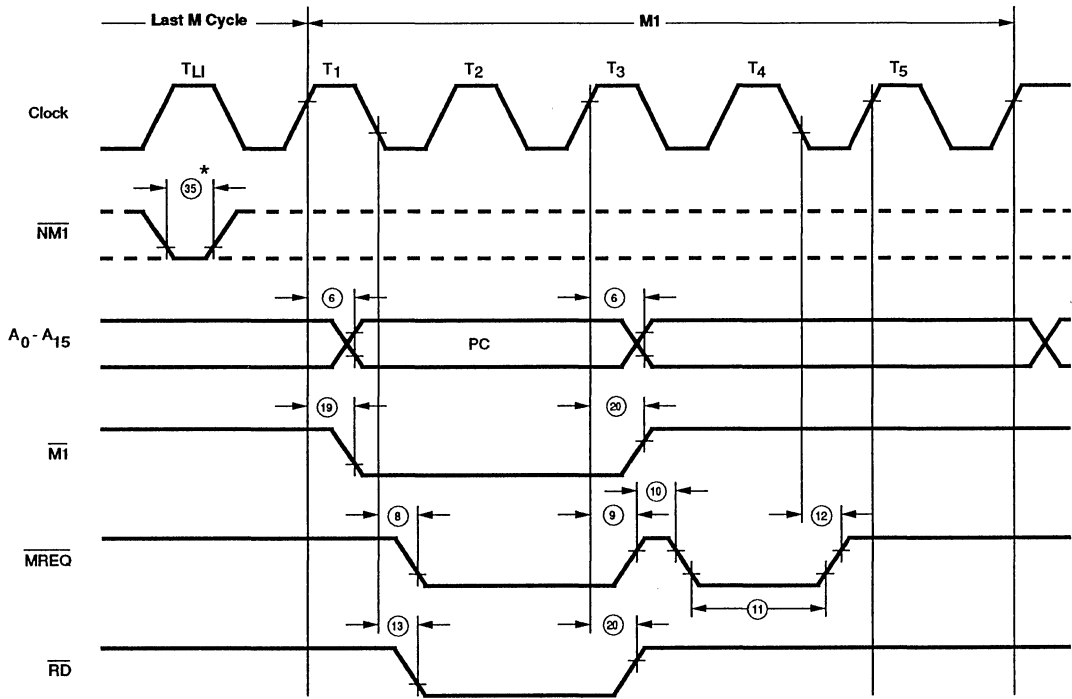


NOTE: 1)  $T_{L1}$  = Last state of any instruction cycle.  
 2)  $T_{WA}$  = Wait cycle automatically inserted by CPU.

Figure 19. Interrupt Request/Acknowledge Cycle

Non-Maskable Interrupt Request Cycle. NMI is sampled at the same time as the maskable interrupt input INT, but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal

memory read operation except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the NMI service routine located at address 0066H (Figure 20).

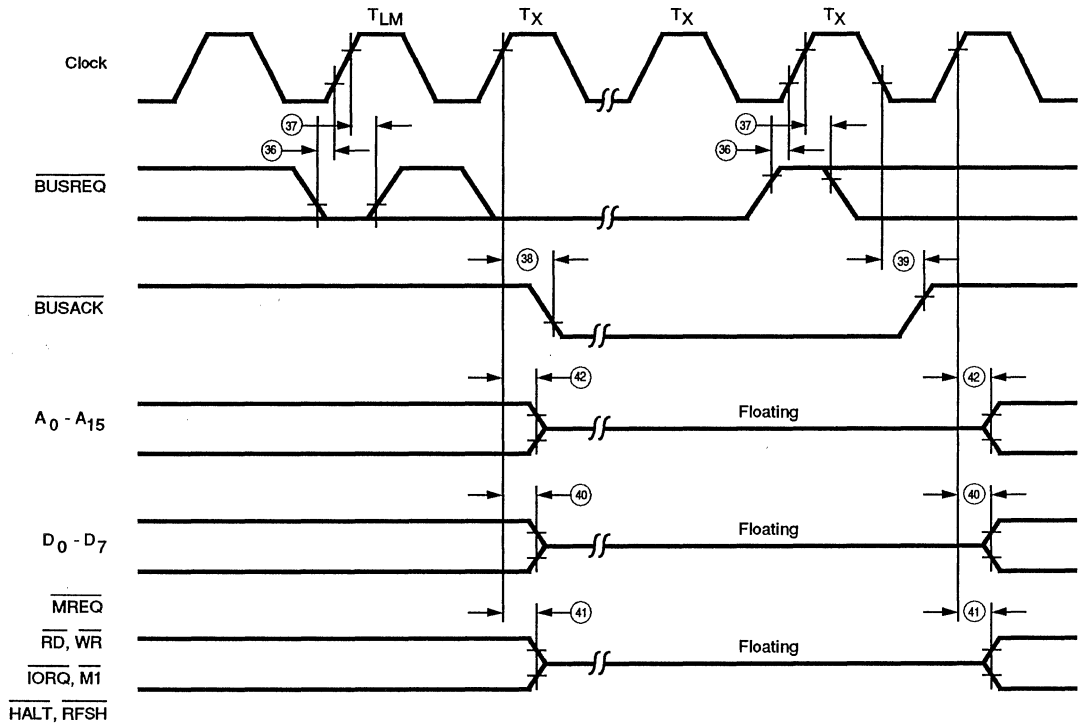


\* Although  $\overline{\text{NMI}}$  is an asynchronous input, to guarantee its being recognized on the following machine cycle,  $\overline{\text{NMI}}$ 's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle (T<sub>LI</sub>).

Figure 20. Non-Maskable Interrupt Request Operation

Bus Request/Acknowledge cycle. The CPU samples  $\overline{\text{BUSREQ}}$  with the rising edge of the last clock period of any machine cycle (Figure 21). If  $\overline{\text{BUSREQ}}$  is active, the CPU sets its address, data, and  $\overline{\text{MREQ}}$ ,  $\overline{\text{IORQ}}$ ,  $\overline{\text{RD}}$ , and  $\overline{\text{WR}}$  lines

to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.



- Notes: 1)  $T_{LM}$  = Last state of any M cycle.  
 2)  $T_X$  = An arbitrary clock cycle used by requesting device.

Figure 21. BUS Request/Acknowledge Cycle

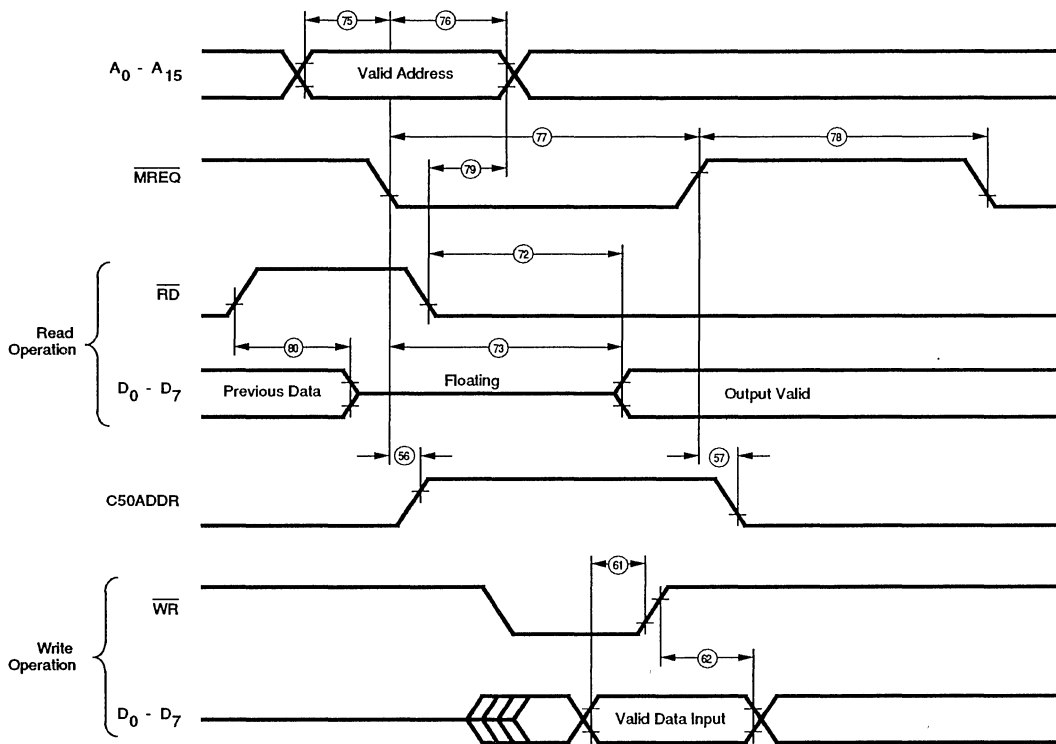


Figure 22(a). Bus Acknowledge Timing for Internal Memory Read or Write Cycles

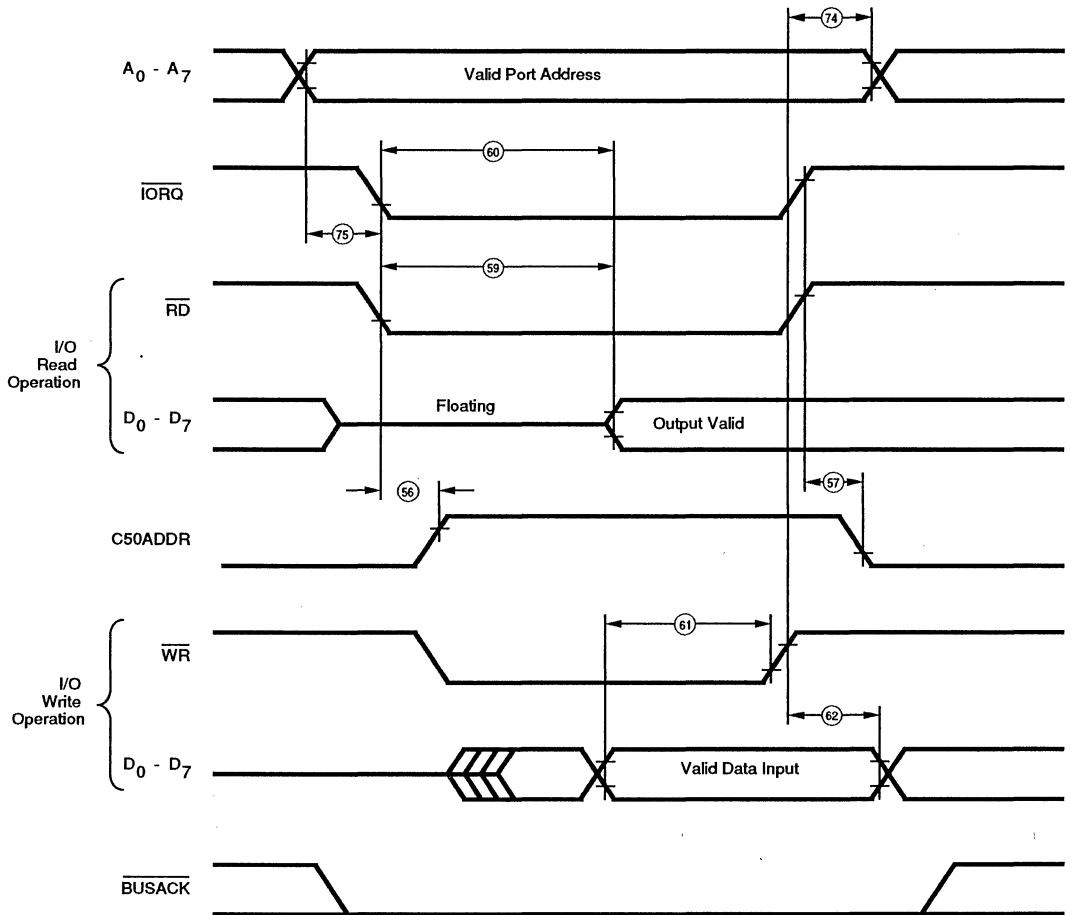
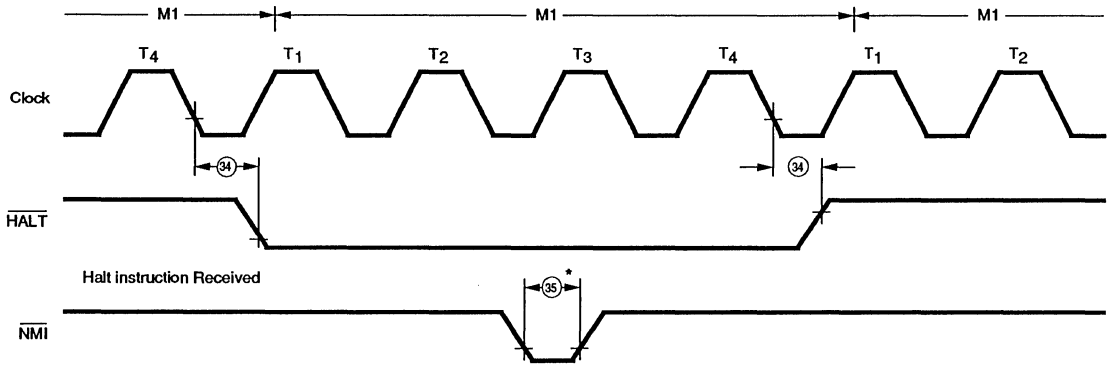


Figure 22(b). Bus Acknowledge Timing for Internal I/O Cycles

Halt Acknowledge Cycle.



\* Although NMI is an asynchronous input, to guarantee its being recognized on the following machine cycle, NMI's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle ( $T_{L1}$ ).

Figure 23. Halt Acknowledge

Reset Cycle.  $\overline{\text{RESET}}$  must be active for at least three clock cycles for the CPU to properly accept it. As long as  $\overline{\text{RESET}}$  remains active, the address and data buses float, and the control outputs are inactive.

Once  $\overline{\text{RESET}}$  goes inactive, two internal T cycles are consumed before the CPU resumes normal processing operation.  $\overline{\text{RESET}}$  clears the PC register, so the first op-code fetch will be location 0000H (Figure 24).

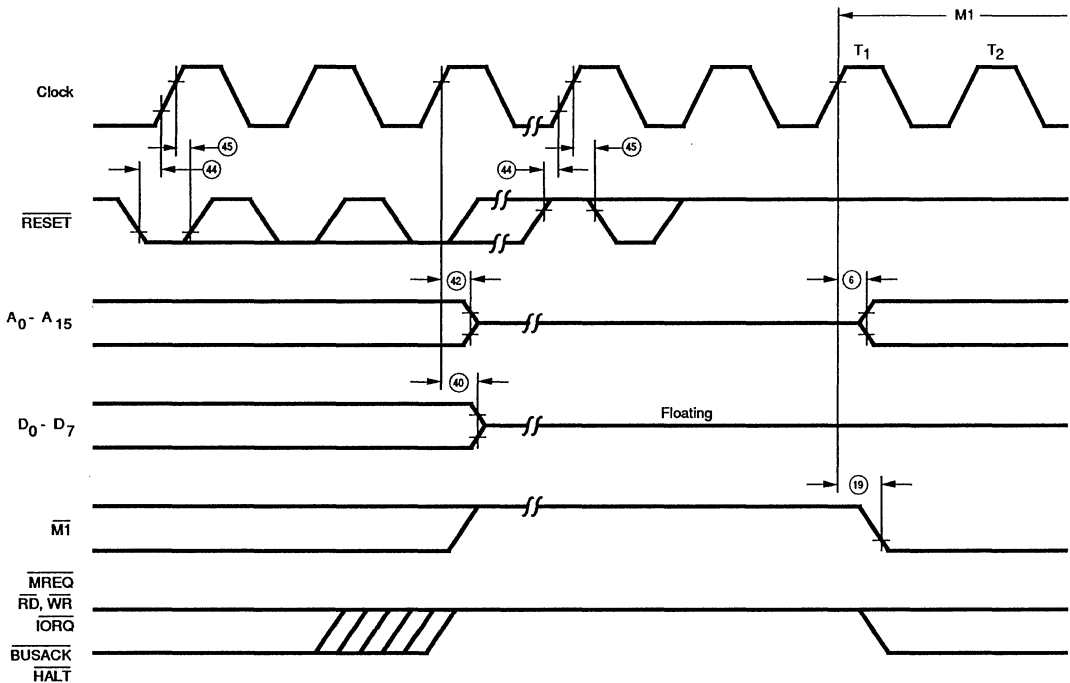


Figure 24. Reset Cycle



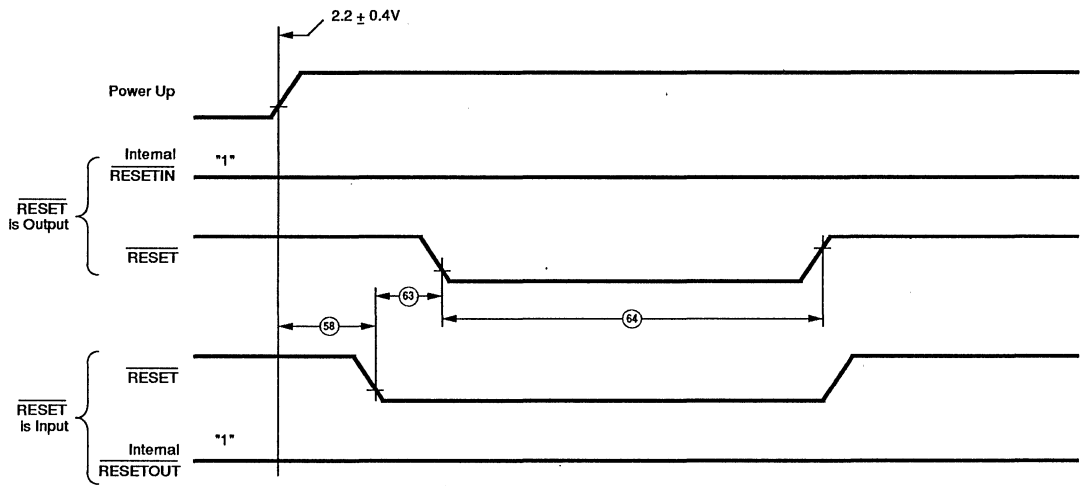


Figure 25. Reset on Power -Up

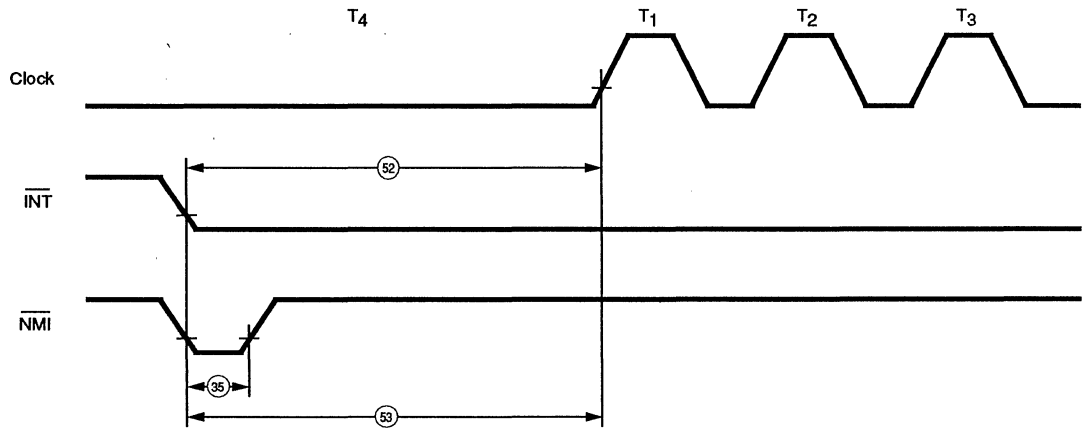
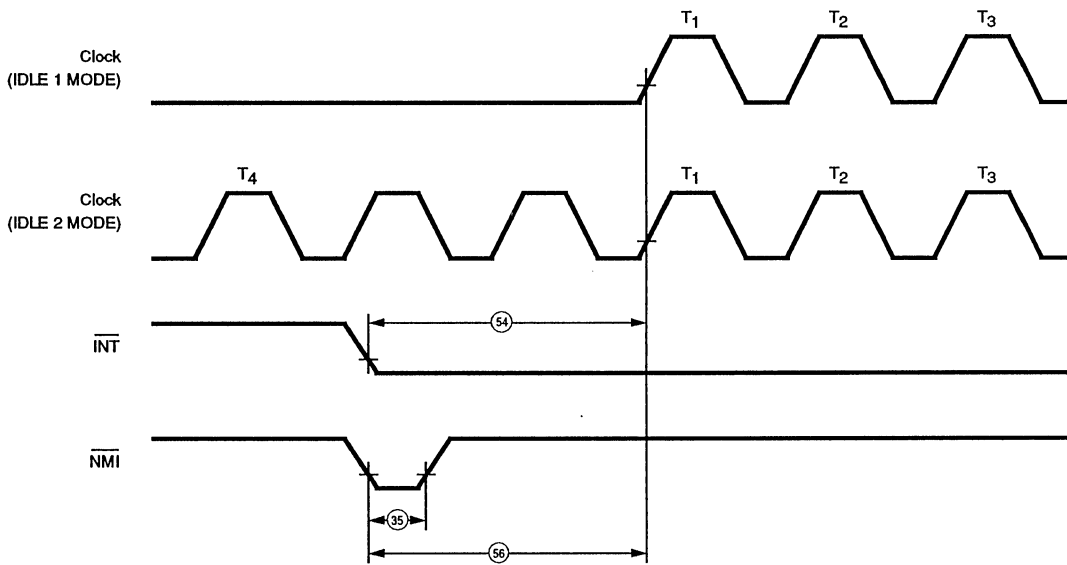


Figure 26. Clock Restart Timing (STOP Mode)



**Figure 27. Clock Restart Timing (IDLE1/2 Mode)**

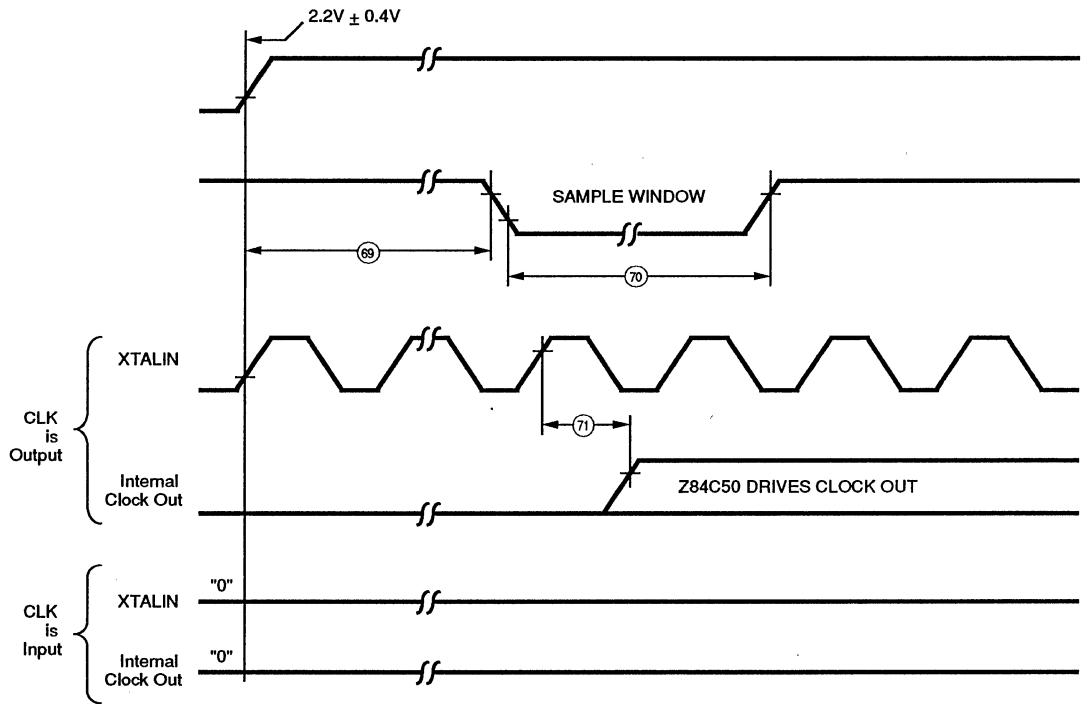


Figure 28. XTALIN Test on Power-up

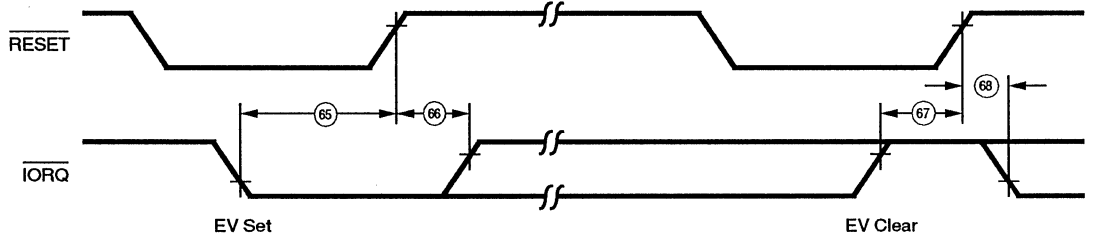


Figure 29. Set/Clear EV During Reset

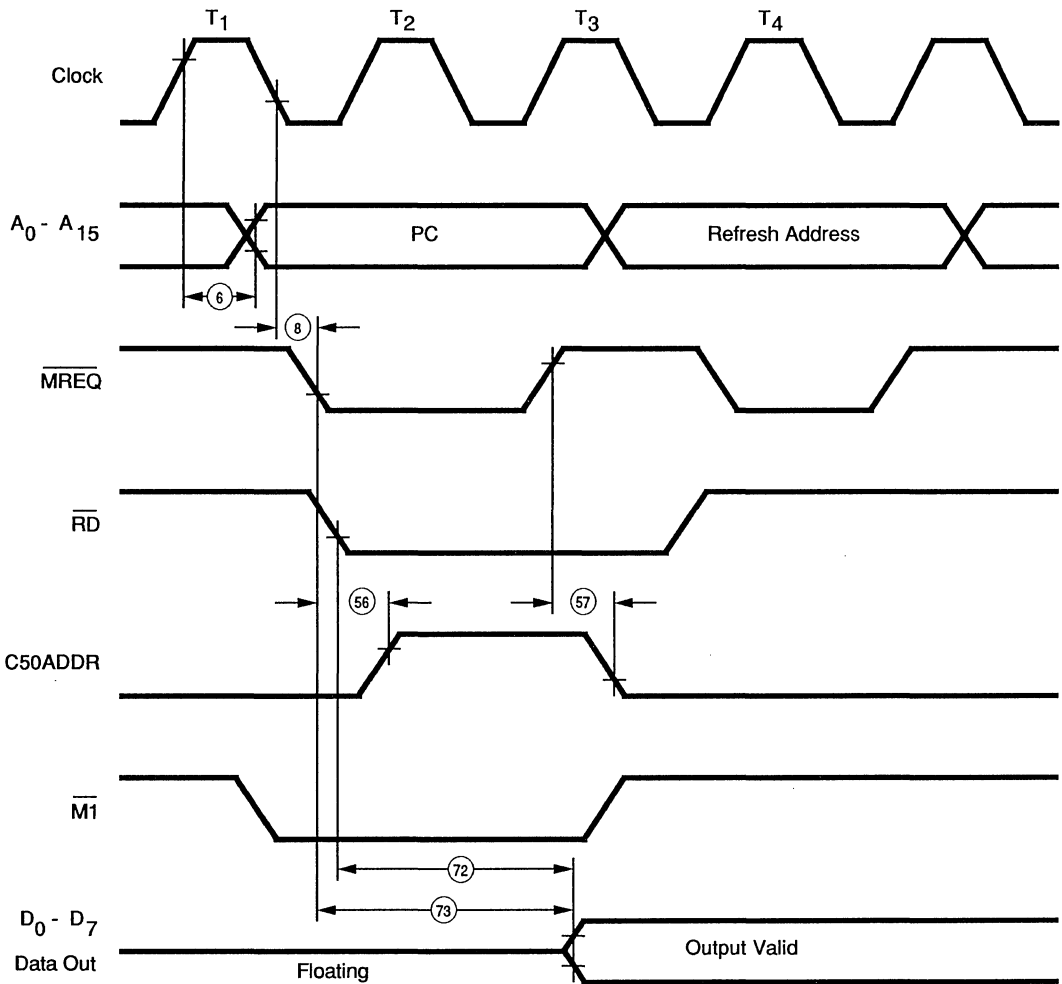


Figure 30. EV Mode Internal Instruction Op-Code Fetch

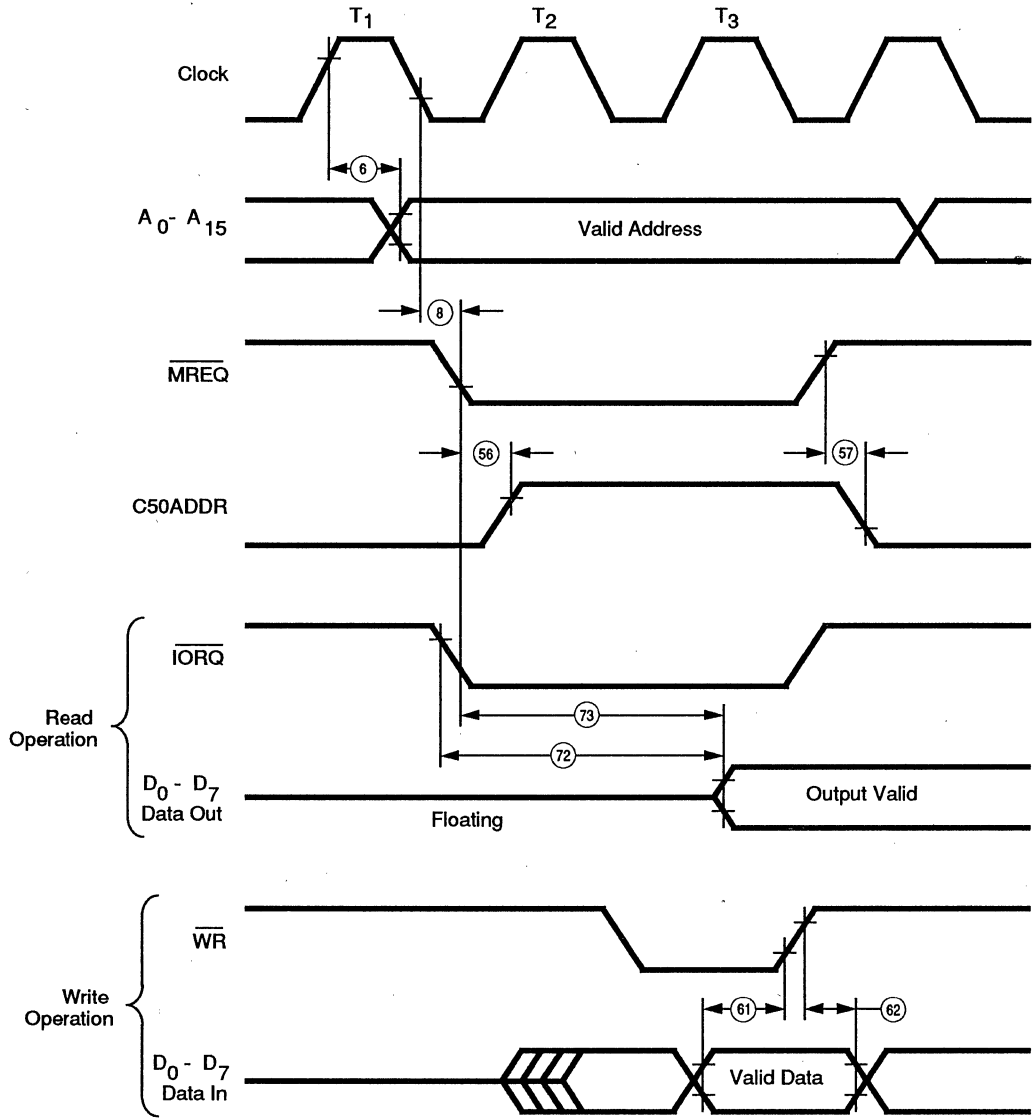


Figure 31. EV Mode Internal Memory Read or Write Cycles

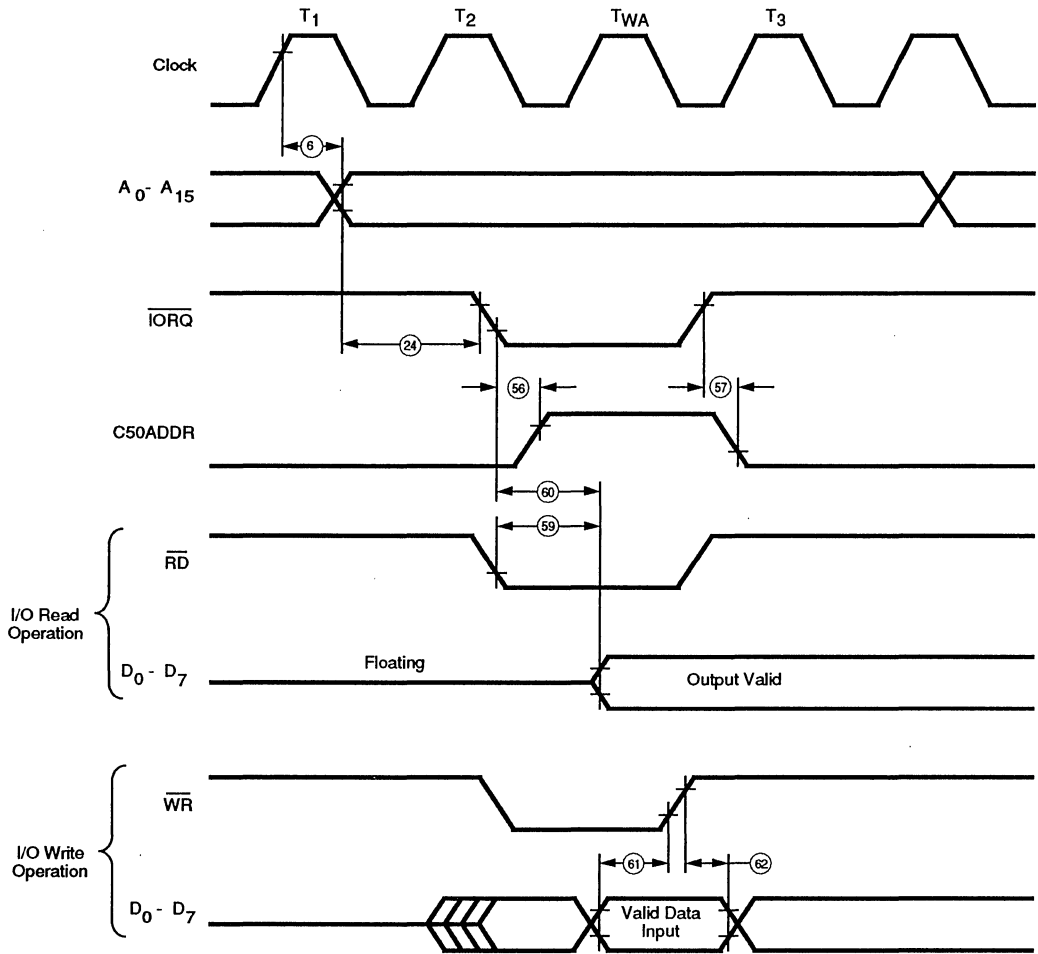


Figure 32. EV Mode Internal I/O Cycles

---

## PRECAUTIONS

To reset the MPU, it is necessary to hold the  $\overline{\text{RESET}}$  signal input at "0" level for at least three clocks.

In particular, to release the HALT state by the  $\overline{\text{RESET}}$  signal in STOP Mode, hold  $\overline{\text{RESET}}$  signal at "0" level for sufficient time in order to stabilize output from the internal oscillator.

In releasing the MPU from the HALT state by interrupt signal in IDLE1/2 Mode and STOP Mode, the MPU will not

be released from the HALT state. Also, the internal system clock will stop again unless an interrupt signal is accepted during the execution of an NOP instruction, even when the internal system clock is restarted by the interrupt signal input. Care must be taken when  $\overline{\text{INT}}$  is used.

Other precautions are identical to those for the Z84C00. Refer to the data sheet for the Z84C00.

## AC CHARACTERISTICS

AC CHARACTERISTICS TA = 0°C to 70°C, VCC = 5V +/- 10%, VSS = 0V.

No	Symbol	Parameter	Min	Max	Unit
1	TcC	Clock cycle time	100	DC	ns
2	TwCh	High clock pulse width	40	DC	ns
3	TwCl	Low clock pulse width	40	DC	ns
4	TfC	Clock falling time		10	ns
5	TrC	Clock rising time		10	ns
6	TdCr (A)	Effective address output delay from clock rise		60	ns
7	TdA (MREQf)	Address output definite time prior to $\overline{\text{MREQ}}$	**		ns
8	TdCf (MREQf)	Delay from clock fall to $\overline{\text{MREQ}} = 'L'$		55	ns
9	TdCf (MREQr)	Delay from clock rise to $\overline{\text{MREQ}} = 'H'$		55	ns
10	TwMREQh	$\overline{\text{MREQ}}$ high level pulse width	**		ns
11	TwMREQl	$\overline{\text{MREQ}}$ low level pulse width	**		ns
12	TdCf (MREQr)	Delay from clock fall to $\overline{\text{MREQ}} = 'H'$		55	ns
13	TdCf (RDf)	Delay from clock fall to $\overline{\text{RD}} = 'L'$		65	ns
14	TdCr (RDr)	Delay from clock rise to $\overline{\text{RD}} = 'H'$		55	ns
15	TsD (Cr)	Data set-up time for Clock rise	25		ns
16	ThD (RDr)	Data hold time for $\overline{\text{RD}}$ rise	0		ns
17	TsWAIT (Cf)	Wait signal set-up time for clock fall	20		ns
18*	ThWAIT (Cf)	Wait hold time after clock fall	10		ns
19	TdCr (Mif)	Delay from clock rise to $\overline{\text{M}}\overline{\text{I}} = 'L'$		65	ns
20	TdCr (Mir)	Delay from clock rise to $\overline{\text{M}}\overline{\text{I}} = 'H'$		65	ns
21	TdCf (RDr)	Delay from clock fall to $\overline{\text{RD}} = 'H'$		55	ns
22	TdCr (RDF)	Delay from clock rise to $\overline{\text{RD}} = 'L'$		55	ns
23	TsD (Cf)	Data set-up time for clock fall	40		ns
24	TdA (IORQf)	Address definite time prior to $\overline{\text{IORQ}}$ fall	**		ns
25	TdCr (IORQf)	Delay from clock rise to $\overline{\text{IORQ}} = 'L'$		50	ns
26	TdCf (IORQr)	Delay from clock fall to $\overline{\text{IORQ}} = 'H'$		55	ns
27	TdD (WRF)	Data definite time prior to $\overline{\text{WR}}$ fall	**		ns
28	TdCf (WRF)	Delay from clock fall to $\overline{\text{WR}} = 'L'$		55	ns
29	TwWR	WR pulse width	**		ns
30	TdCf (WRr)	Delay from clock fall to $\overline{\text{WR}} = 'H'$		55	ns
31	TdD (WRf)	Data definite time prior to $\overline{\text{WR}}$ fall	**		ns
32	TdCr (WRf)	Delay from clock rise to $\overline{\text{WR}} = 'L'$		50	ns
33	TdWRr (D)	Output data holding after $\overline{\text{WR}} = 'H'$	**		ns
34	TdCf (HALT)	Delay from clock fall to $\overline{\text{HALT}} = 'L'$ or $'H'$		100	ns
35	TwNMI	$\overline{\text{NMI}}$ pulse width	60		ns
36	TsBUSREQ (Cr)	Set-up time for clock rise	35		ns
37*	ThBUSREQ (Cr)	$\overline{\text{BUSREQ}}$ hold time after clock rise	15		ns
38	TdCr (BUSACKf)	Time from clock rise to $\overline{\text{BUSACK}} = 'L'$		75	ns
39	TdCf (BUSACKr)	Time from clock fall to $\overline{\text{BUSACK}} = 'H'$		75	ns
40	TdCr (Dz)	Delay from clock rise to data bus float state		65	ns



## AC CHARACTERISTICS (Continued)

No	Symbol	Parameter	Min	Max	Unit
41	TdCr (CTz)	Delay from clock rise to control output float state ( $\overline{MREQ}$ , $\overline{IORQ}$ , $\overline{RD}$ , $\overline{WR}$ )		60	ns
42	TdCr (Az)	Delay from clock rise to address bus float state		75	ns
43	TdCTr (A)	Address hold time from $\overline{MREQ}$ , $\overline{IORQ}$ , $\overline{RD}$ , or $\overline{WR}$	**		ns
44	TsRESET (Cr)	$\overline{RESET}$ set-up time for clock rise	40		ns
45*	ThRESET (Cr)	$\overline{RESET}$ hold time for clock rise	15		ns
46	TsINTf (Cr)	$\overline{INT}$ set-up time for clock rise	50		ns
47*	ThINTR (Cr)	$\overline{INT}$ hold time after clock rise	15		ns
49	TdCf (IORQf)	Delay from clock fall to $\overline{IORQ} = 'L'$		55	ns
50	TdCr (IORQr)	Delay from clock rise to $\overline{IORQ} = 'H'$		55	ns
51	TdCf (D)	Delay from clock fall to data output		110	ns
52	TRST1S	Clock (CLK) restart time by $\overline{INT}$ (STOP mode)	(typ)( $2^{14}+2.5$ )xTcC		ns
53	TRST2S	Clock (CLK) restart time by $\overline{NMI}$ (STOP mode)	(typ)( $2^{14}+2.5$ )xTcC		ns
54	TRST1I	Clock (CLK) restart time by $\overline{INT}$ (IDLE1/2 mode)	(typ) 2.5 TcC		ns
55	TRST2I	Clock (CLK) restart time by $\overline{NMI}$ (IDLE1/2 mode)	(typ) 2.5 TcC		ns
56	TdMlf (C50ADDRy)	Delay from $\overline{MREQ}$ or $\overline{IORQ}$ falling to C50ADDR = 'H'		35	ns
57	TdMlr (C50ADDRf)	Delay from $\overline{MREQ}$ or $\overline{IORQ}$ rising to C50ADDR = 'L'		30	ns
58	TRESEti	$\overline{RESET}$ input test range	50	150	$\mu$ s
59	TRDf (D)	$\overline{RD}$ falling to output data valid		60	ns
60	TMI (D)	$\overline{IORQ}$ or $\overline{MREQ}$ falling to output data valid		70	ns
61	TsD (WRr)	Data set-up time prior to $\overline{WR}$ rising	30		ns
62	ThD (WRr)	Data hold time after $\overline{WR}$ rising	5		ns
63	Td RESET	$\overline{RESETIN}$ falling to $\overline{RESETOUT}$ delay		50	ns
64	Tw RESET	Automatic Power-up pulse width	25	75	ms
65	Td IORQf (RESETr)	$\overline{IORQ}$ set-up time before $\overline{RESET}$ rising to set EV mode	15		ns
66	ThRESETr (IORQf)	$\overline{IORQ}$ hold time from $\overline{RESET}$ rising to EV mode	10		ns
67	Tc IORQr (RESETr)	$\overline{IORQ}$ set-up time before $\overline{RESET}$ rising to clear EV mode	15		ns
68	TcRESETr (IORQf)	$\overline{IORQ}$ hold time from $\overline{RESET}$ rising to clear EV mode	10		ns
69	Ts XTALIN	XTALIN sampling start time	15	45	ms
70	TwXTALIN	XTALIN sampling window duration	10	30	ms
71	TCr (CLK)	Clock rising to 84C50 CLK out delay		30	ns
72	TeRDf (D)	$\overline{RD}$ falling to output data valid for external access of on-chip memory		70	ns
73	TeMREQf (D)	$\overline{MREQ}$ falling to output data valid for external access of on-chip memory		80	ns
74	Th WRr	Address hold time from $\overline{WR}$ rising	15		ns
75	Ts MREQf	Address set-up time to $\overline{MREQ}/\overline{IORQ}$ falling for $\overline{BUSACK}$ cycle	20		ns
76	Th MREQf	Address hold time from $\overline{MREQ}$ falling for $\overline{BUSACK}$ cycle	20		ns
77	TI MREQ	$\overline{MREQ}$ low pulse width for $\overline{BUSACK}$ cycle	40		ns
78	Th MREQ	$\overline{MREQ}$ high pulse width for $\overline{BUSACK}$ cycle	30		ns
79	Th RDf	Address hold time from $\overline{RD}$ falling for $\overline{BUSACK}$ cycle	20		ns
80	Td RDr (Df)	$\overline{RD}$ rising to output data floating for $\overline{BUSACK}$ cycle		10	ns

---

## AC CHARACTERISTICS (Continued)

---

\* Test conditions are: CL = 100 pF

\*\* NOTES: AC Characteristics (per line item number).

---

Number	Symbol	General Parameter
1	TcC	TwCh + TwCl + TrC + TfC
7	TdA (MEREQf)	TwCh + TfC - 45
10	TwMREQh	TwCh + TfC - 25
11	TwMREQl	TcC - 30
24	TdA (IORQf)	TcC - 50
27	TdD (WRf)	TcC - 100
29	TwWR	TcC - 25
31	TdD (WRf)	TwCl + TrC - 100
33	TdWRr (D)	TwCl + TrC - 50
43	TdCTr (A)	TwCl + TrC - 45
48	TdM1f (IORQf)	2TcC + TwCh + TfC - 45

---

AC Test Conditions:

$V_{IH} = 3V$   $V_{OH} = 2V$   $V_{IHC} = V_{CC} - 0.6V$  FLOAT = + -0.5V  
 $V_{IL} = .5V$   $V_{OL} = .8V$   $V_{ILC} = .5V$   $V_{CC} = 4.5$  TO  $5.5V$

## DC CHARACTERISTICS

DC CHARACTERISTICS VCC = 5.0 V +/- 10% unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Condition
V <sub>OHc</sub>	Clock output high voltage	VCC-0.6		V	-2mA
V <sub>OLc</sub>	Clock output low voltage		0.4	V	+2mA
V <sub>IH</sub>	Input high voltage	2.2	VCC	V	
V <sub>IL</sub>	Input low voltage	-0.3	0.8	V	
V <sub>OL</sub>	Output low voltage		0.4 <sup>5</sup>	V	I <sub>OL</sub> = 2.0mA
V <sub>OH1</sub>	Output high voltage	2.4 <sup>5</sup>		V	I <sub>OH</sub> = -1.6mA
V <sub>OH2</sub>	Output high voltage	VCC-0.8 <sup>1,5</sup>		V	I <sub>OH</sub> = -250μA
I <sub>CC1</sub>	Power supply current 10 MHz		50	mA	V <sub>CC</sub> = 5V V <sub>IH</sub> = V <sub>CC</sub> - 0.2V V <sub>IL</sub> = 0.2V XTALIN = 10 MHz
I <sub>CC2</sub>	Power supply current (STOP mode)		10 <sup>1,3</sup>	μA	V <sub>CC</sub> = 5V
I <sub>CC3</sub>	Power supply current (IDLE1 mode)		4 <sup>1</sup>	mA	V <sub>IH</sub> = V <sub>CC</sub> - 0.2V V <sub>IL</sub> = 0.2V XTALIN = 10 MHz
I <sub>CC4</sub>	Power supply current (IDLE2 mode)		15 <sup>1</sup>	mA	V <sub>IH</sub> = V <sub>CC</sub> - 0.2V V <sub>IL</sub> = 0.2V XTALIN = 10 MHz
I <sub>LI</sub>	Input leakage current	-10	10 <sup>4</sup>	μA	V <sub>IN</sub> = 0.4 to V <sub>CC</sub>
I <sub>LO</sub>	3-state output leakage Current in float	-10	10 <sup>2</sup>	μA	V <sub>OUT</sub> = 0.4 to V <sub>CC</sub>

1. Measurements made with outputs floating

2. A<sub>15</sub>-A<sub>0</sub>, D<sub>7</sub>-D<sub>0</sub>, MREQ, IORQ, RD, and WR. Except on RD pin of 40-pin DIP Version, where I<sub>LO</sub> = ± 25 μA.

3. I<sub>CC2</sub> standby current is guaranteed when the halt pin is low in STOP mode.

4. All pins except XTALI, where I<sub>LI</sub> = ± 25 μA.

5. A<sub>15</sub>-A<sub>0</sub>, D<sub>7</sub>-D<sub>0</sub>, MREQ, IORQ, RD, WR, HALT, M1, and BUSACK.

---

## ELECTRICAL CHARACTERISTICS

### Absolute Maximum Ratings

Voltage on Vcc with respect to Vss.....-0.3V to +7V  
Voltages on all inputs with respect to Vss.....-0.3V to Vcc  
+0.3V

Operating Ambient

Temperature.....See Ordering Information

Storage Temperature.....-65° C to 150° C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

---

### Standard Test Conditions

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (OV). Positive current flows into the referenced pin.

The Ordering Information section lists temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.

Available operating temperature ranges are:

E = -40°C TO +100°C

S=0°C to 70°C

Voltage Supply Range:  $+4.50V \leq V_{cc} \leq +5.50V$

All AC parameters assume a load capacitance of 100 pf. Add 10 ns delay for each 50 pf increase in load up to a maximum of 150 pf for the data bus and 100 pf for address and control lines. AC timing measurements are referenced to 1.5 volts (except for clock, which is referenced to the 10% and 90% points). Maximum capacitive load for CLK is 125 pf.



# Z8470 Z80<sup>®</sup> DART Dual Asynchronous Receiver/Transmitter



## Product Specification

### FEATURES

- Two independent full-duplex channels with separate modem controls. Modem status can be monitored.
- In x1 clock mode, data rates are 0 to 800K bits/second with a 4.0 MHz clock, or 0 to 1.2 M bits/second with a 6.0 MHz clock.
- Receiver data registers are quadruply buffered; the transmitter is doubly buffered.
- Programmable options include 1, 1½, or 2 stop bits; even, odd, or no parity; and x1, x16, x32, and x64 clock modes.
- Break generation and detection as well as parity-, overrun-, and framing-error detection are available.
- Interrupt features include a programmable interrupt vector, a "status affects vector" mode for fast interrupt processing, and the standard Z80 peripheral daisy-chain interrupt structure that provides automatic interrupt vectoring with no external logic.
- On-chip logic for ring indication and carrier-detect status.

### GENERAL DESCRIPTION

The Z80 DART (Dual-Channel Asynchronous Receiver/Transmitter) is a dual-channel multifunction peripheral component that satisfies a wide variety of asynchronous serial data communications requirements in microcomputer systems. The Z80 DART is used as a serial-to-parallel,

parallel-to-serial converter/controller in asynchronous applications. In addition, the device also provides modem controls for both channels. In applications where modem controls are not needed, these lines can be used for general-purpose I/O.

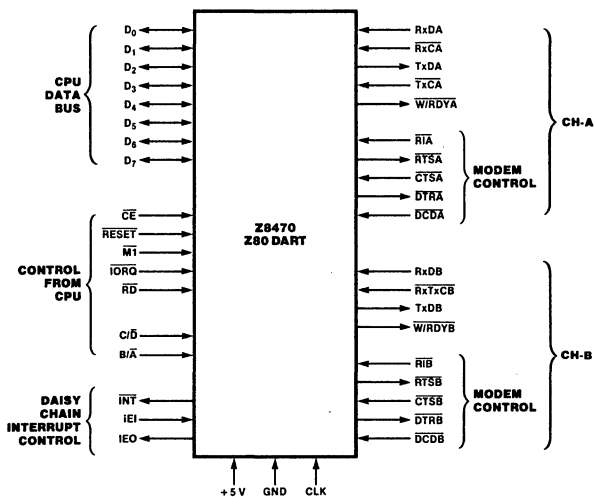


Figure 1. Pin Functions

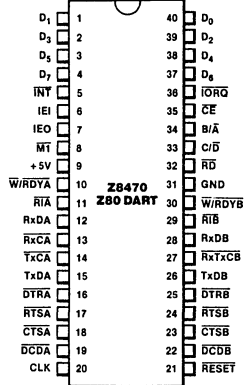


Figure 2. 40-Pin Dual-In-Line Package (DIP),  
Pin Assignments

---

Zilog also offers the Z80 SIO, a more versatile device that provides synchronous (Bisync, HDLC, and SDLC) as well as asynchronous operation.

---

The Z80 DART is fabricated with n-channel silicon-gate depletion-load technology, and is packaged in a 40-pin plastic or ceramic DIP (Figures 1 and 2).

---

## PIN DESCRIPTION

**B/ $\bar{A}$ .** *Channel A or B Select* (input, High selects Channel B). This input defines which channel is accessed during a data transfer between the CPU and the Z80 DART.

**C/ $\bar{D}$ .** *Control or Data Select* (input, High selects Control). This input specifies the type of information (control or data) transferred on the data bus between the CPU and the Z80 DART.

**$\bar{CE}$ .** *Chip Enable* (input, active Low). A Low at this input enables the Z80 DART to accept command or data input from the CPU during a write cycle, or to transmit data to the CPU during a read cycle.

**CLK.** *System Clock* (input). The Z80 DART uses the standard Z80 single-phase system clock to synchronize internal signals.

**$\bar{CTSA}$ ,  $\bar{CTSB}$ .** *Clear To Send* (inputs, active Low). When programmed as Auto Enables, a Low on these inputs enables the respective transmitter. If not programmed as Auto Enables, these inputs may be programmed as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow-risetime signals.

**D<sub>0</sub>-D<sub>7</sub>.** *System Data Bus* (bidirectional, 3-state). This bus transfers data and commands between the CPU and the Z80 DART.

**$\bar{DCDA}$ ,  $\bar{DCDB}$ .** *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if the Z80 DART is programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered.

**$\bar{DTRA}$ ,  $\bar{DTRB}$ .** *Data Terminal Ready* (outputs, active Low). These outputs follow the state programmed into the DTR bit. They can also be programmed as general-purpose outputs.

**IEI.** *Interrupt Enable In* (input, active High). This signal is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A High on this line indicates that no other device of higher priority is being serviced by a CPU interrupt service routine.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an interrupt from this Z80 DART. Thus, this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

**$\bar{INT}$ .** *Interrupt Request* (output, open drain, active Low). When the Z80 DART is requesting an interrupt, it pulls  $\bar{INT}$  Low.

**$\bar{M1}$ .** *Machine Cycle One* (input from Z80 CPU, active Low). When  $\bar{M1}$  and  $\bar{RD}$  are both active, the Z80 CPU is fetching

an instruction from memory; when  $\bar{M1}$  is active while  $\bar{IORQ}$  is active, the Z80 DART accepts  $\bar{M1}$  and  $\bar{IORQ}$  as an interrupt acknowledge if the Z80 DART is the highest priority device that has interrupted the Z80 CPU.

**$\bar{IORQ}$ .** *Input/Output Request* (input from CPU, active Low).  $\bar{IORQ}$  is used in conjunction with B/ $\bar{A}$ , C/ $\bar{D}$ ,  $\bar{CE}$ , and  $\bar{RD}$  to transfer commands and data between the CPU and the Z80 DART. When  $\bar{CE}$ ,  $\bar{RD}$ , and  $\bar{IORQ}$  are all active, the channel selected by B/ $\bar{A}$  transfers data to the CPU (a read operation). When  $\bar{CE}$  and  $\bar{IORQ}$  are active, but  $\bar{RD}$  is inactive, the channel selected by B/ $\bar{A}$  is written to by the CPU with either data or control information as specified by C/ $\bar{D}$ .

**$\bar{RxCA}$ ,  $\bar{RxCB}$ .** *Receiver Clocks* (inputs). Receive data is sampled on the rising edge of RxC. The Receive Clocks may be 1, 16, 32, or 64 times the data rate.

**$\bar{RD}$ .** *Read Cycle Status* (input from CPU, active Low). If  $\bar{RD}$  is active, a memory or I/O read operation is in progress.

**$\bar{RxDA}$ ,  $\bar{RxDB}$ .** *Receive Data* (inputs, active High).

**$\bar{RESET}$ .** *Reset* (input, active Low). Disables both receivers and transmitters, forces TxDA and TxDB marking, forces the modem controls High, and disables all interrupts.

**$\bar{RIA}$ ,  $\bar{RIB}$ .** *Ring Indicator* (inputs, active Low). These inputs are similar to CTS and DCD. The Z80 DART detects both logic level transitions and interrupts the CPU. When not used in switched-line applications, these inputs can be used as general-purpose inputs.

**$\bar{RTSA}$ ,  $\bar{RTSB}$ .** *Request to Send* (outputs, active Low). When the RTS bit is set, the RTS output goes Low. When the  $\bar{RTS}$  bit is reset, the output goes High after the transmitter empties.

**$\bar{TxCA}$ ,  $\bar{TxCB}$ .** *Transmitter Clocks* (inputs). TxD changes on the falling edge of  $\bar{TxC}$ . The Transmitter Clocks may be 1, 16, 32, or 64 times the data rate; however, the clock multiplier for the transmitter and the receiver must be the same. The Transmit Clock inputs are Schmitt-trigger buffered. Both the Receiver and Transmitter Clocks may be driven by the Z80 CTC Counter Time Circuit for programmable baud rate generation.

**TxDA, TxDB.** *Transmit Data* (outputs, active High).

**$\bar{W/RDYA}$ ,  $\bar{W/RDYB}$ .** *Wait/Ready* (outputs, open drain when programmed for Wait function, driven High and Low when programmed for Ready function). These dual-purpose outputs may be programmed as Ready lines for a DMA controller or as Wait lines that synchronize the CPU to the Z80 DART data rate. The reset state is open drain.

## FUNCTIONAL DESCRIPTION

The functional capabilities of the Z80 DART can be described from two different points of view: as a data communications device, it transmits and receives serial data, and meets the requirements of asynchronous data communications protocols; as a Z80 family peripheral, it interacts with the Z80 CPU and other Z80 peripheral circuits, and shares the data, address, and control buses, as well as being a part of the Z80 interrupt structure. As a peripheral to other microprocessors, the Z80 DART offers valuable features such as nonvectored interrupts, polling, and simple handshake capability.

The first part of the following functional description introduces Z80 DART data communications capabilities; the second part describes the interaction between the CPU and the Z80 DART.

The Z80 DART offers RS-232 serial communications support by providing device signals for external modem control. In addition to dual-channel Request To Send, Clear To Send, and Data Carrier Detect ports, the Z80 DART also features a dual channel Ring Indicator (RIA, RIB) input to facilitate local/remote or station-to-station communication capability. Figure 3 is a block diagram.

**Communications Capabilities.** The Z80 DART provides two independent full-duplex channels for use as an asynchronous receiver/transmitter. The following is a short description of receiver/transmitter capabilities. For more details, refer to the Asynchronous Mode section of the *Z80 SIO Technical Manual* (03-3033-01).

The Z80 DART offers transmission and reception of five to eight bits per character, plus optional even or odd parity. The transmitter can supply one, one and a half, or two stop bits per character and can provide a break output at any time. The receiver break detection logic interrupts the CPU both at the start and end of a received break. Reception is protected from spikes by a transient spike rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the Receive Data input. If the Low does not persist—as in the case of a transient—the character assembly process is not started.

Framing errors and overrun errors are detected and buffered together with the character on which they occurred. Vectored interrupts allow fast servicing of interrupting conditions using dedicated routines. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The Z80 DART does not require symmetric Transmit and Receive Clock signals, a feature that allows it to be used with a Z80 CTC or any other clock source. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32, or 1/64 of the clock rate supplied to the Receive and Transmit Clock inputs. When using Channel B, the bit rates for transmit and receive operations must be the same because  $\overline{RxC}$  and  $\overline{TxC}$  are bonded together ( $RxTxCB$ ).

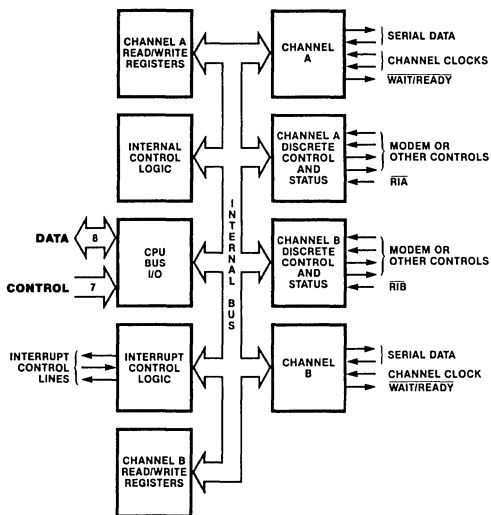


Figure 3. Block Diagram

**I/O Interface Capabilities.** The Z80 DART offers the choice of Polling, Interrupt (vectored or non-vectored) and Block Transfer modes to transfer data, status, and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

**Polling.** There are no interrupts in the Polled mode. Status registers RR0 and RR1 are updated at appropriate times for each function being performed. All the interrupt modes of the Z80 DART must be disabled to operate the device in a Polled environment.

While in its Polling sequence, the CPU examines the status contained in RR0 for each channel; the RR0 status bits serve as an acknowledge to the Poll inquiry. The two RR0 status bits  $D_0$  and  $D_2$  indicate that a data transfer is needed. The status also indicates Error or other special status conditions. The Z80 DART Programming section contains more information. The Special Receive Condition status contained in RR1 does not have to be read in a Polling sequence because the status bits in RR1 are accompanied by a Receive Character Available status in RR0.

**Interrupts.** The Z80 DART offers an elaborate interrupt scheme that provides fast interrupt response in real-time applications. As a member of the Z80 family, the Z80 DART can be daisy-chained along with other Z80 peripherals for peripheral interrupt-priority resolution. In addition, the internal interrupts of the Z80 DART are nested to prioritize the various interrupts generated by Channels A and B. Channel B registers WR2 and RR2 contain the interrupt vector that points to an interrupt service routine in the memory. To eliminate the necessity of writing a status analysis routine, the Z80 DART can modify the interrupt vector in RR2 so it points directly to one of eight interrupt service routines. This is done under program control by setting a program bit (WR1,  $D_2$ ) in Channel B called "Status



---

Affects Vector." When this bit is set, the interrupt vector in RR2 is modified according to the assigned priority of the various interrupting conditions.

Transmit interrupts, Receive interrupts, and External/Status interrupts are the main sources of interrupts. Each interrupt source is enabled under program control with Channel A having a higher priority than Channel B, and with Receiver, Transmit, and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted by the transmit buffer becoming empty. (This implies that the transmitter must have had a data character written into it so it can become empty.) When enabled, the receiver can interrupt the CPU in one of three ways:

- Interrupt on the first received character
- Interrupt on all received characters
- Interrupt on a Special receive condition

Interrupt On First Character is typically used with the Block Transfer mode. Interrupt On All Received Characters can optionally modify the interrupt vector in the event of a parity error. The Special Receive Condition interrupt can occur on a character basis. The Special Receive Condition can cause an interrupt can occur on a character basis. The Special Receive condition can cause an interrupt only if the Interrupt On First Received Character or Interrupt On All Received Characters mode is selected. In Interrupt On First Receive

Character, an interrupt can occur from Special Receive conditions (except Parity Error) after the first Received character interrupt (example: Receive Overrun interrupt).

The main function of the External/Status interrupt is to monitor the signal transitions of the  $\overline{\text{CTS}}$ ,  $\overline{\text{DCD}}$ , and  $\overline{\text{RI}}$  pins; however, an External/Status interrupt is also caused by the detection of a Break sequence in the data stream. The interrupt caused by the Break sequence has a special feature that allows the Z80 DART to interrupt when the Break sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Break condition.

**CPU/DMA Block Transfer.** The Z80 DART provides a Block Transfer mode to accommodate CPU block transfer functions and DMA block transfers (Z80 DMA or other designs). The Block Transfer mode uses the  $\overline{\text{WRDY}}$  output in conjunction with the Wait/Ready bits of Write Register 1. The  $\overline{\text{WRDY}}$  output can be defined under software control as a Wait line in the CPU Block Transfer mode or as a Ready line in the DMA Block Transfer mode.

To a DMA controller, the Z80 DART Ready output indicates that the Z80 DART is ready to transfer data to or from memory. To the CPU, the Wait output indicates that the Z80 DART is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle.

---

## INTERNAL ARCHITECTURE

The device internal structure includes a Z80 CPU interface, internal control and interrupt logic, and two full-duplex channels. Each channel contains read and write registers, and discrete control and status logic that provides the interface to modems or other external devices.

The read and write register group includes five 8-bit control registers and two status registers. The interrupt vector is written into an additional 8-bit register (Write Register 2) in Channel B, that may be read through Read Register 2 in Channel B. The registers for both channels are designated as follows:

- WR0-WR5 Write Registers 0 through 5
- RR0-RR2 Read Registers 0 through 2

The bit assignment and functional grouping of each register is configured to simplify and organize the programming process.

The logic for both channels provides formats, bit synchronization, and validation for data transferred to and from the channel interface. The modem control inputs Clear to Send (CTS), Data Carrier Detect (DCD), and Ring

Indicator ( $\overline{\text{RI}}$ ) are monitored by the control logic under program control. All the modem control signals are general purpose in nature and can be used for functions other than modem control.

For automatic interrupt vectoring, the interrupt control logic determines which channel and which device within the channel has the highest priority. Priority is fixed with Channel A assigned a higher priority than Channel B; Receive, Transmit, and External/Status interrupts are prioritized in that order within each channel.

**Data Path.** The transmit and receive data path illustrated for Channel A in Figure 4 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service a Receive Character Available interrupt in a high-speed data transfer.

The transmitter has an 8-bit transmit data register that is loaded from the internal data bus, and a 9-bit transmit shift register that is loaded from the transmit data register.

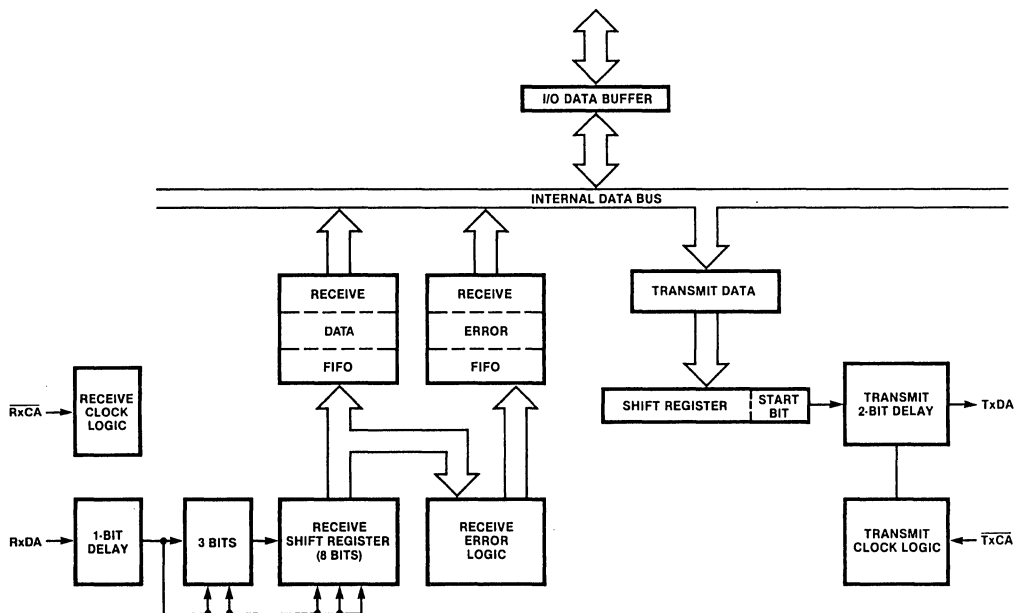


Figure 4. Data Path

## READ, WRITE AND INTERRUPT TIMING

**Read Cycle.** The timing signals generated by a Z80 CPU input instruction to read a Data or Status byte from the Z80 DART are illustrated in Figure 5.

**Write Cycle.** Figure 6 illustrates the timing and data signals generated by a Z80 CPU output instruction to write a Data or Control byte into the Z80 DART.

**Interrupt Acknowledge Cycle.** (Figure 7) After receiving an Interrupt Request signal (INT pulled Low), the Z80 CPU sends an Interrupt Acknowledge signal ( $\overline{M1}$  and  $\overline{IORQ}$  both Low). The daisy-chained interrupt circuits determine the highest priority interrupt requestor. The IEI of the highest priority peripheral is terminated High. For any peripheral

that has no interrupt pending or under service,  $IEO = IEI$ . Any peripheral that does have an interrupt pending or under service forces its  $IEO$  Low.

To insure stable conditions in the daisy chain, all interrupt status signals are prevented from changing while  $\overline{M1}$  is Low. When  $\overline{IORQ}$  is Low, the highest priority interrupt requestor (the one with  $IEI$  High) places its interrupt vector on the data bus and sets its internal interrupt-under-service latch.

Refer to the *Technical Manual* (03-3033-01) for additional details on the interrupt daisy chain and interrupt nesting.

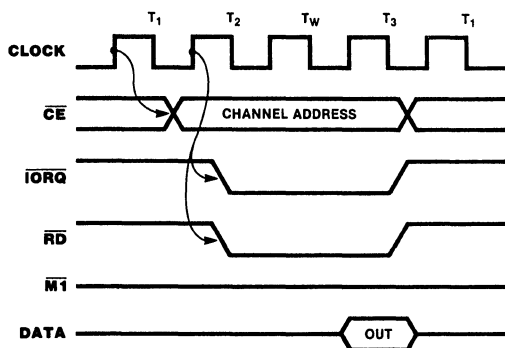


Figure 5. Read Cycle

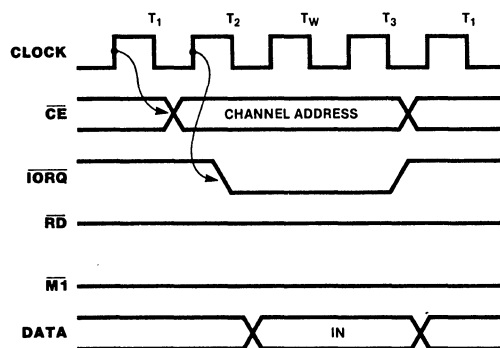


Figure 6. Write Cycle

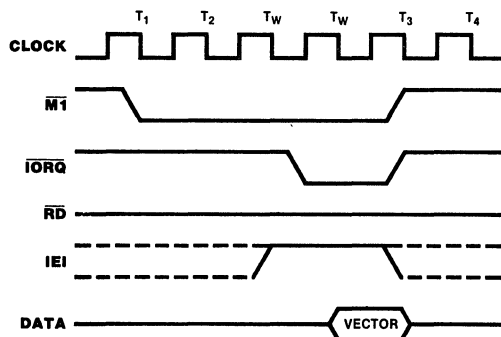


Figure 7. Interrupt Acknowledge Cycle

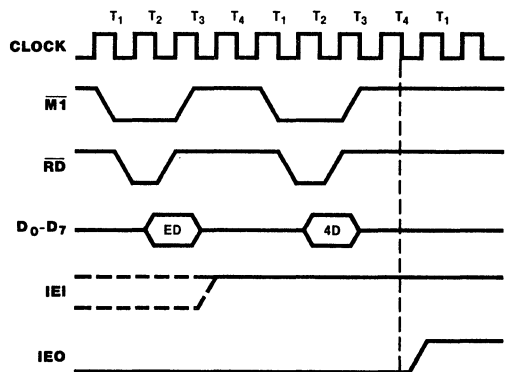


Figure 8. Return from Interrupt Cycle

**Return From Interrupt Cycle.** (Figure 8) Normally, the Z80 CPU issues an RETI (Return From Interrupt) instruction at the end of an interrupt service routine. RETI is a 2-byte opcode (ED-4D) that resets the interrupt-under-service latch to terminate the interrupt that has just been processed.

When used with other CPUs, the Z80 DART allows the user to return from the interrupt cycle with a special command called "Return From Interrupt" in Write Register 0 of Channel A. This command is interpreted by the Z80 DART in exactly the same way it would interpret an RETI command on the data bus.

## Z80 DART PROGRAMMING

To program the Z80 DART, the system program first issues a series of commands that initialize the basic mode and then other commands that qualify conditions within the selected mode. For example, the character length, clock rate, number of stop bits, even or odd parity are first set, then the Interrupt mode and, finally, receiver or transmitter enable.

Both channels contain command registers that must be programmed via the system program prior to operation. The Channel Select input (B/ $\bar{A}$ ) and the Control/Data input (C/ $\bar{D}$ ) are the command structure addressing controls, and are normally controlled by the CPU address bus.

**Write Registers.** The Z80 DART contains six registers (WR0-WR5) in each channel that are programmed separately by the system program to configure the functional personality of the channels (Figure 4). With the exception of WR0, programming the write registers requires two bytes. The first byte contains three bits (D<sub>0</sub>-D<sub>2</sub>) that point to the selected register; the second byte is the actual control word that is written into the register to configure the Z80 DART.

WR0 is a special case in that all the basic commands (CMD<sub>0</sub>-CMD<sub>2</sub>) can be accessed with a single byte. Reset (internal or external) initializes the pointer bits D<sub>0</sub>-D<sub>2</sub> to point to WR0. This means that a register cannot be pointed to in the same operation as a channel reset.

**Read Registers.** The Z80 DART contains three registers (RR0-RR2) that can be read to obtain the status information for each channel (except for RR2, which applies to Channel B only). The status information includes error conditions, interrupt vector, and standard communications-interface signals.

To read the contents of a selected read register other than RR0, the system program must first write the pointer byte to WR0 in exactly the same way as a write register operation. Then, by executing an input instruction, the contents of the addressed read register can be read by the CPU.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring. For example, when the interrupt vector indicates that a Special Receive Condition interrupt has occurred, all the appropriate error bits can be read from a single register (RR1).

### Write Register Functions

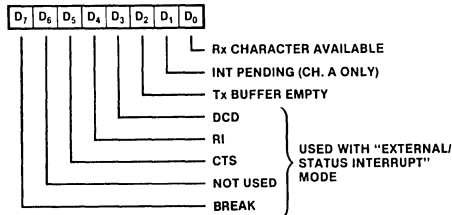
WR0	Register pointers, initialization commands for the various modes
WR1	Transmit/Receive interrupt and data transfer mode definition
WR2	Interrupt vector (Channel B only)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls

### Read Register Functions

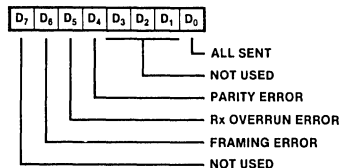
RR0	Transmit/Receive buffer status, interrupt status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only)

# Z80 DART READ AND WRITE REGISTERS

## READ REGISTER 0

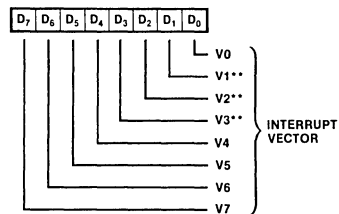


## READ REGISTER 1\*



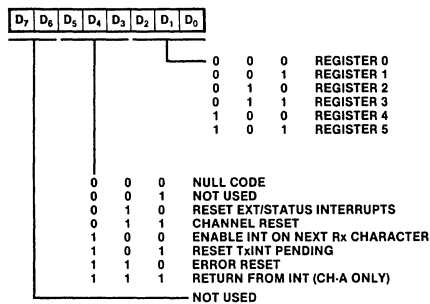
\* Used With Special Receive Condition Mode.

## READ REGISTER 2

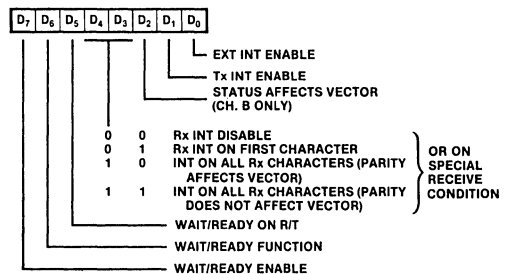


\*\* Variable If "Status Affects Vector" is Programmed.

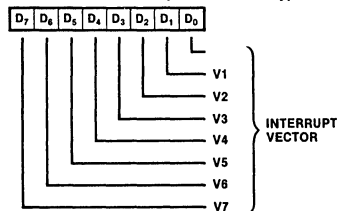
## WRITE REGISTER 0



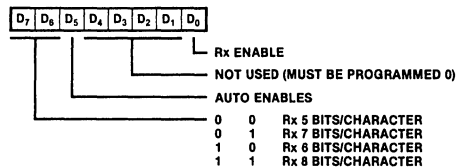
## WRITE REGISTER 1



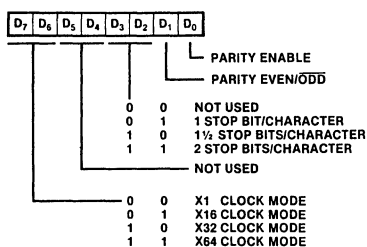
## WRITE REGISTER 2 (Channel B only)



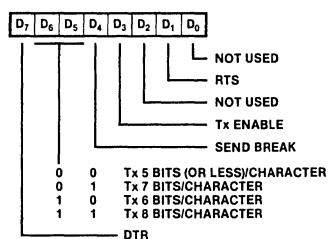
## WRITE REGISTER 3



## WRITE REGISTER 4



## WRITE REGISTER 5



## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins with respect to GND ..... -0.3V to +7V  
 Operating Ambient Temperature ..... See Ordering Information  
 Storage Temperature ..... -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

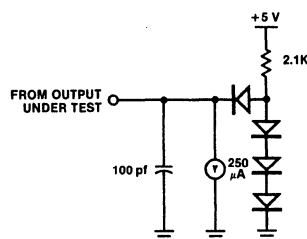
## STANDARD TEST CONDITIONS

The DC characteristics and capacitance sections listed below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

Available operating temperature ranges are:

- S = 0°C to +70°C, +4.75V ≤ V<sub>CC</sub> ≤ +5.25V

The Ordering Information section lists package temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.



## DC CHARACTERISTICS

Symbol	Parameter	Min	Max	Unit	Test Condition
V <sub>ILC</sub>	Clock Input Low Voltage	-0.3	+0.45	V	
V <sub>IHC</sub>	Clock Input High Voltage	V <sub>CC</sub> - 0.6		V	V <sub>CC</sub> + 0.3V
V <sub>IL</sub>	Input Low Voltage	-0.3	+0.8	V	
V <sub>IH</sub>	Input High Voltage	+2.0	+5.5	V	
V <sub>OL</sub>	Output Low Voltage		+0.4	V	I <sub>OL</sub> = 2.0 mA
V <sub>OH</sub>	Output High Voltage	+2.4		V	I <sub>OH</sub> = -250 μA
I <sub>L</sub>	Input/3-State Output Leakage Current	-10	+10	μA	0.4 < V <sub>IN</sub> < 2.4V
I <sub>L(RI)</sub>	RI Pin Leakage Current	-40	+10	μA	0.4 < V <sub>IN</sub> < 2.4V
I <sub>CC</sub>	Power Supply Current		100	mA	

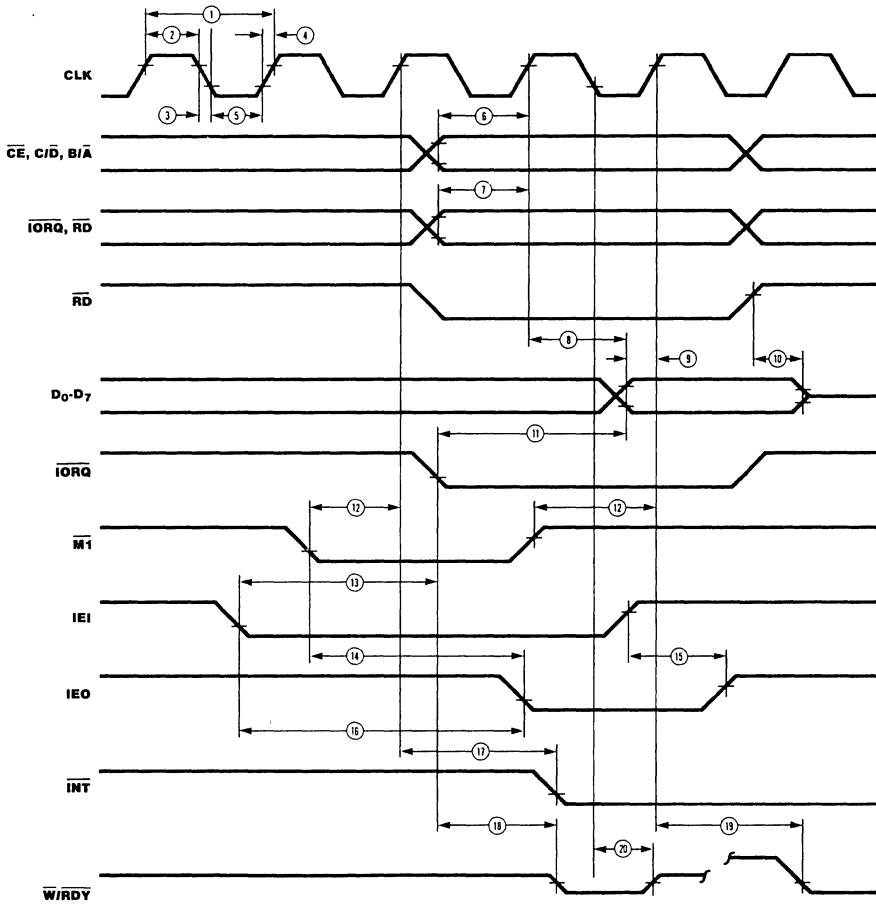
T<sub>A</sub> = 0°C to 70°C, V<sub>CC</sub> = +5V, ±5%.

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit
C	Clock Capacitance		40	pf
C <sub>IN</sub>	Input Capacitance		5	pf
C <sub>OUT</sub>	Output Capacitance		15	pf

Over specified temperature range; f = 1 MHz.  
 Unmeasured pins returned to ground.

# AC CHARACTERISTICS TIMING

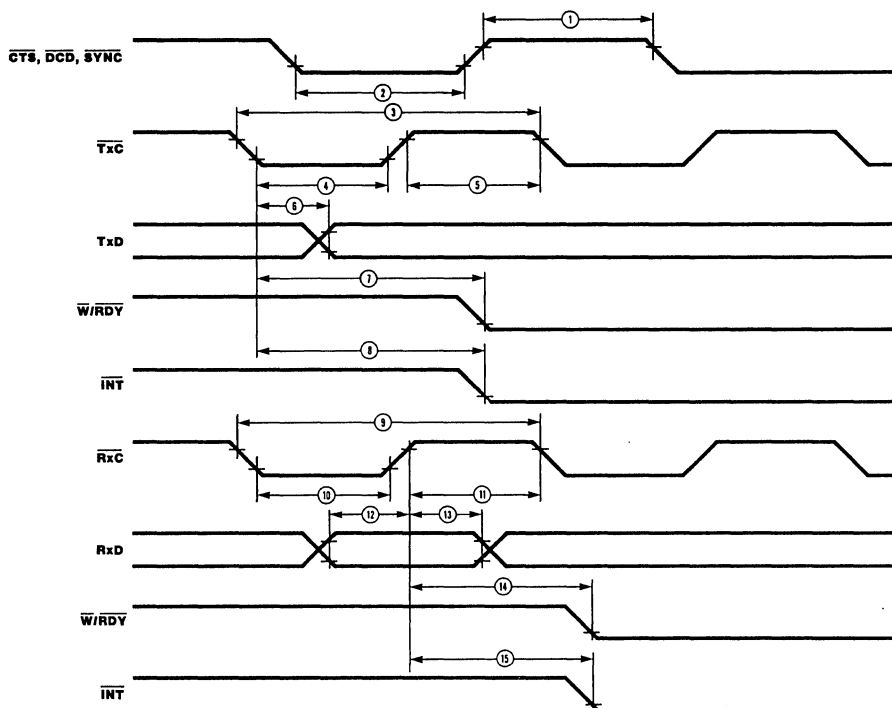


## AC CHARACTERISTICS

Number	Symbol	Parameter	Z0847004		Z0847006	
			Min	Max	Min	Max
1	TcC	Clock Cycle Time	250	4000	165	4000
2	TwCh	Clock Width (High)	105	2000	70	2000
3	TfC	Clock Fall Time		30		15
4	TrC	Clock Rise Time		30		15
5	TwCl	Clock Width (Low)	105	2000	70	2000
6	TsAD(C)	$\overline{CE}$ , $C/\overline{D}$ , $B/\overline{A}$ to Clock $\uparrow$ Setup Time	145		60	
7	TsCS(C)	$\overline{IORQ}$ , $\overline{RD}$ to Clock $\uparrow$ Setup Time	115		60	
8	TdC(DO)	Clock $\uparrow$ to Data Out Delay		220		150
9	TsDI(C)	Data In to Clock $\uparrow$ Setup (Write or M1 Cycle)	50		30	
10	TdRD(DOz)	$\overline{RD}$ $\uparrow$ to Data Out Float Delay		110		90
11	TdIO(DOI)	$\overline{IORQ}$ $\downarrow$ to Data Out Delay (INTACK Cycle)		160		100
12	TsM1(C)	$\overline{M1}$ to Clock $\uparrow$ Setup Time	90		75	
13	TsIEI(IO)	IEI to $\overline{IORQ}$ $\downarrow$ Setup Time (INTACK Cycle)	140		120	
14	TdM1(IEO)	$\overline{M1}$ $\downarrow$ to IEO $\downarrow$ Delay (interrupt before M1)		190		160
15	TdIEI(IEOr)	IEI $\uparrow$ to IEO $\uparrow$ Delay (after ED decode)		100		70
16	TdIEI(IEOf)	IEI $\downarrow$ to IEO $\downarrow$ Delay		100		70
17	TdC(INT)	Clock $\uparrow$ to INT $\downarrow$ Delay		200		150
18	TdIO(W/RWf)	$\overline{IORQ}$ $\downarrow$ or $\overline{CE}$ $\downarrow$ to $\overline{W/RDY}$ $\downarrow$ Delay (Wait Mode)		210		175
19	TdC(W/RR)	Clock $\uparrow$ to $\overline{W/RDY}$ $\downarrow$ Delay (Ready Mode)		120		100
20	TdC(W/RWz)	Clock $\downarrow$ to $\overline{W/RDY}$ Float Delay (Wait Mode)		130		110

\*Units in nanoseconds (ns).

## AC CHARACTERISTICS (Continued)



Number	Symbol	Parameter	Z0847004		Z0847006		Notes*
			Min	Max	Min	Max	
1	TwPh	Pulse Width (High)	200		200		2
2	TwPl	Pulse Width (Low)	200		200		2
3	TcTxC	$\overline{\text{TxC}}$ Cycle Time	400	$\infty$	330	$\infty$	2
4	TwTxCl	$\overline{\text{TxC}}$ Width (Low)	180	$\infty$	100	$\infty$	2
5	TwTxCh	$\overline{\text{TxC}}$ Width (High)	180	$\infty$	100	$\infty$	2
6	TdTxC(TxD)	$\overline{\text{TxC}} \downarrow$ to TxD Delay		300		220	2
7	TdTxC(W/RRf)	$\overline{\text{TxC}} \downarrow$ to W/RDY $\downarrow$ Delay (Ready Mode)	5	9	5	9	1
8	TdTxC(INT)	$\overline{\text{TxC}} \downarrow$ to INT $\downarrow$ Delay	5	9	5	9	1
9	TcRxC	$\overline{\text{RxC}}$ Cycle Time	400	$\infty$	330	$\infty$	2
10	TwRxC	$\overline{\text{RxC}}$ Width (Low)	180	$\infty$	100	$\infty$	2
11	TwRxC	$\overline{\text{RxC}}$ Width (High)	180	$\infty$	100	$\infty$	2
12	TsRxD(RxC)	RxD to $\overline{\text{RxC}} \uparrow$ Setup Time (x1 Mode)	0		0		2
13	ThRxD(RxC)	RxD Hold Time (x1 Mode)	140		100		2
14	TdRxC(W/RRf)	$\overline{\text{RxC}} \uparrow$ to W/RDY $\downarrow$ Delay (Ready Mode)	10	13	10	13	1
15	TdRxC(INT)	$\overline{\text{RxC}} \uparrow$ to INT $\downarrow$ Delay	10	13	10	13	1

\* In all modes, the System Clock rate must be at least five times the maximum data rate. RESET must be active a minimum of one complete clock cycle.

1. Units equal to System Clock Periods.

2. Units in nanoseconds (ns).





## Z84C90 CMOS Z80<sup>®</sup>KIO Serial/Parallel/Counter/Timer

### FEATURES:

- Two independent synchronous/asynchronous serial channels.
- Three 8-bit parallel ports.
- Four independent counter/timer channels.
- On-chip clock oscillator/driver.
- Software/Hardware Resets.
- Designed in CMOS for low power operations.
- Supports Z80 Family interrupt daisy chain.
- Programmable interrupt priorities.
- 8, 10 and 12.5 MHz bus clock frequency.
- Single +5 Volt Power Supply.

### GENERAL DESCRIPTION:

Zilog's Z84C90 Serial/Parallel I/O/Counter/Timer (KIO) is a multi-channel, multi-purpose I/O device designed to provide the end-user with a cost effective and powerful solution to meet his peripheral needs. The Z84C90 combines the features of one Z84C30 CTC, one Z84C4xSIO, one Z84C20 PIO, a byte-wide bit-programmable I/O port, and a crystal oscillator into a single 84-pin PLCC package. The

block diagram for the Z84C90 is shown in Figure 1 while the pinout is shown in Figure 2. Utilizing fifteen internal registers for data and programming information, the KIO can easily be configured to any given system environment. Although the optimum performance is obtained with a Z84C00 CPU, the KIO can just as easily be used with any other CPU.

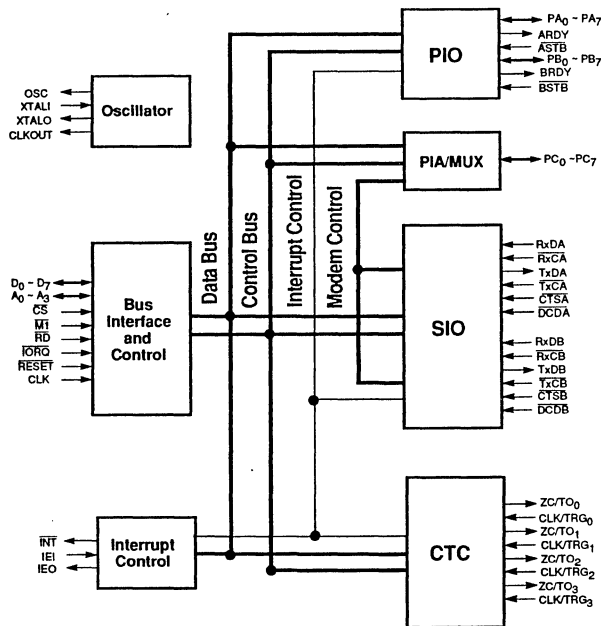


Figure 1: KIO Block Diagram

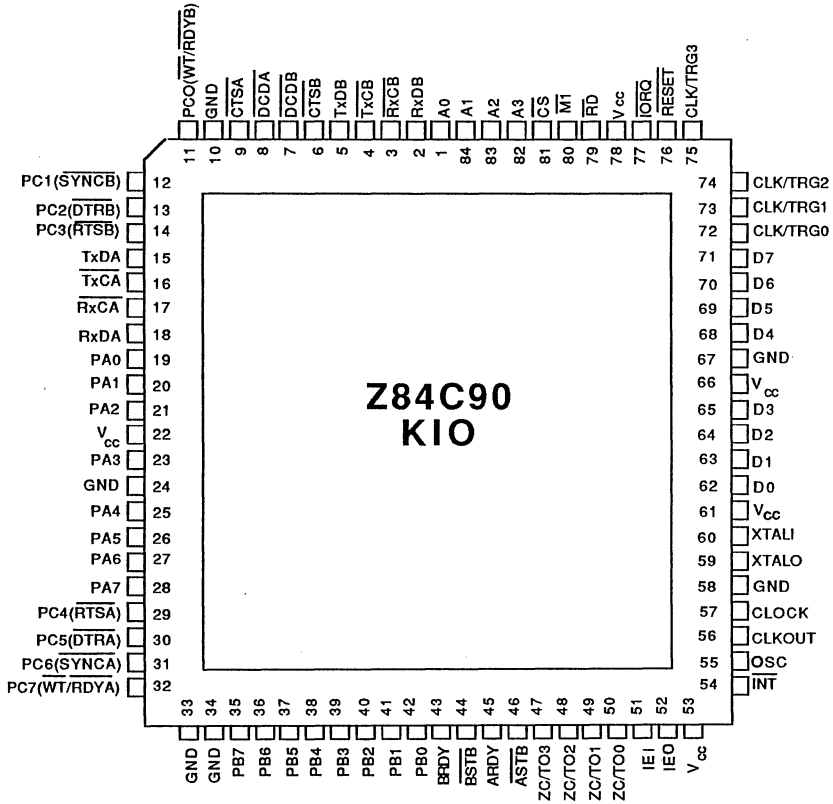


Figure 2: PLCC Pinout

**Z84C20 Parallel Input/Output Logic Unit:** This logic unit provides both TTL- and CMOS-compatible interfaces between peripheral devices and a CPU through the use of two 8-bit parallel ports. The CPU configures the logic to interface to a wide range of peripheral devices with no external logic. Typical devices that are compatible with this interface are keyboards, printers, and EPROM/PAL programmers.

The parallel ports (designated Port A and Port B) are byte-wide and completely compatible with the Z84C20 PIO (see Figure 3.). These two ports have several modes of opera-

tion; input, output, bidirectional, or bit control mode. Each port has two handshake signals (RDY and  $\overline{STB}$ ) which can be used to control data transfers. The RDY (ready) indicates that the port is ready for data transfer while  $\overline{STB}$  (strobe) is an input to the port that indicates when data transfer has occurred. Each of the ports can also be programmed to interrupt the CPU upon the occurrence of specified status conditions and generate unique interrupt vectors when the CPU responds. (For more information on the operation of this portion of the logic, please refer to the Z84C20 PIO Product Specification and Technical Manual.)

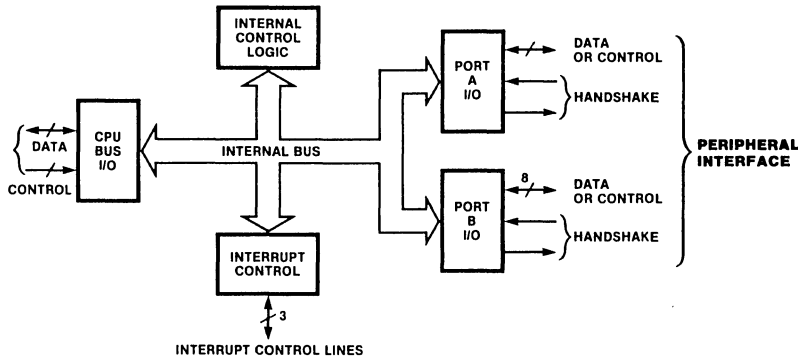


Figure 3: PIO Block Diagram

**Parallel Interface Adapter (PIA) Logic Unit:** This logic also offers an additional 8-bits of I/O, referred to as the PIA port (see Figure 4), to the user. This port, designated as Port C, is bit-programmable for data transfers; each bit can be individually programmed as either an input or an output. Bit direction control is accomplished through the programming of the PIA Control Register. When programmed as outputs, the output data latches are programmed with an I/O write cycle and their state can be read with an I/O read cycle. When programmed as inputs, the state of the external pin is read with the I/O read cycle. This port does not have handshake capabilities and offers no interrupt capabilities. This port is multiplexed to provide, when desired, the additional modem and CPU control signals for the serial I/O logic unit.

When a read from the PIA port is done, input data will be latched when  $\overline{IORQ}$ ,  $\overline{CS}$ , and  $\overline{RD}$  are all detected active. The data bus will display this data as a result of the rising edge of the CLOCK input after this occurrence. When a write to the PIA port is done, data will be written as a result of the rising edge of the CLOCK input after  $\overline{IORQ}$  and  $\overline{CS}$  have been detected active and  $\overline{RD}$  has been detected inactive.

**Counter/Timer Logic Unit:** This logic unit provides the user with four individual 8-bit counter/timer channels that are compatible with the Z84C30 CTC (see Figure 5). The counter/timers can be programmed by the CPU for a broad range of counting and timing applications. Typical applications include event counting, interrupt and interval timing, and serial baud rate clock generation.

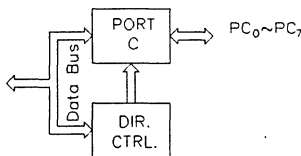


Figure 4: PIA Block Diagram

Each of the counter/timer channels, designated Channels 0 through 3, have an 8-bit prescaler (when used in timer mode) as well as its own 8-bit counter to provide a wide range of count resolution. Each of the channels also have their own clock/trigger input to quantify the counting process and an output to indicate zero crossing/timeout conditions. With only one interrupt vector programmed into this logic unit, each channel can generate a unique interrupt vector in response to the interrupt acknowledge cycle.

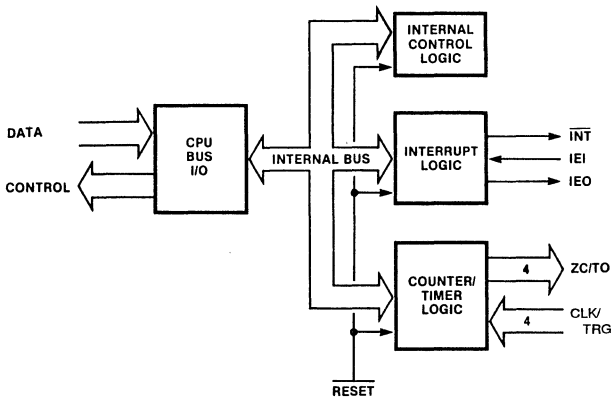


Figure 5: CTC Block Diagram

**Serial I/O Logic Unit:** This logic unit provides the user with two separate serial I/O channels that are completely compatible with the Z84C4x SIO (see Figure 6). Their basic functions as serial-to-parallel and parallel-to-serial converters can be programmed by a CPU for a broad range of serial communications applications. Each channel, desig-

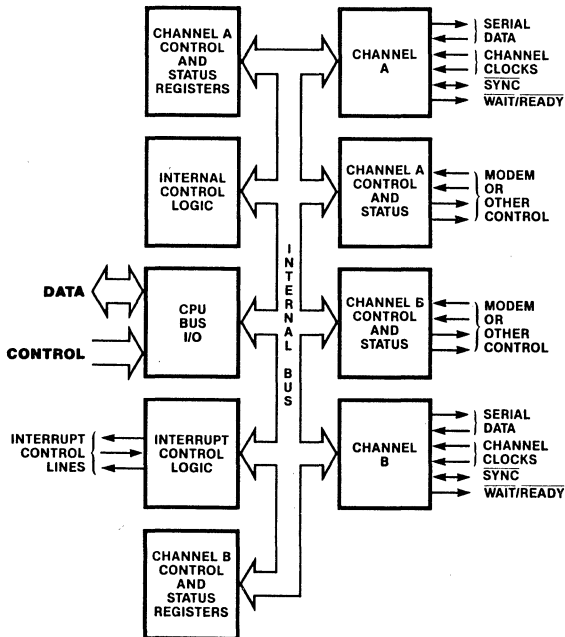


Figure 6: SIO Block Diagram

nated Channel A and Channel B, is capable of supporting all common asynchronous and synchronous protocols (Monosync, Bisync, and SDLC/HDLC), byte- or bit-oriented.

In the default state of the KIO, each serial channel supports full duplex communication with separate transmit and receive data lines, two modem control signals (CTS and DCD), and separate transmit and receive clock inputs. Optionally, additional modem and CPU/DMA control signals can be obtained through the PIA port. (For more information on the operation of this portion of the logic, please refer to the Z84C40 SIO Product Specification and Technical Manual).

**Clock Oscillator/Driver Logic Unit:** A clock oscillator/driver is also available that will allow the user to eliminate that circuitry within his new design, or for use as another oscillator within the system. This logic will accept either a crystal, ceramic resonator, or TTL-compatible clock input and generate a MOS-compatible clock output and also an oscillator reference output. A fundamental parallel resonant crystal (Figure 7) is recommended. The preferred value of the two capacitors - C1 and C2 is 33 pF each.

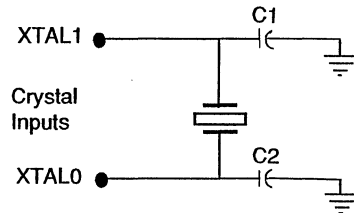


Figure 7: Crystal Connection

**Command Logic Unit:** This logic unit provides for much more than just controlling the interface between the KIO and the CPU. The main function provided by this unit is to allow the user to configure the internal interrupt daisy chain of the KIO into the order in which he would like the peripherals to interrupt. Any one of the three devices (SIO, CTC, PIO) can be the highest priority while another can be second and the remaining one third. The user can even configure the daisy chain such that no internal peripherals are involved in the chain. Programming of the daisy chain configuration is done by programming the Command Register with the appropriate 3-bit pattern in  $D_0$ - $D_2$  and  $D_3$  set to "1".

A second function of this logic unit is to provide software controllable "hardware" resets to each of the individual devices. This allows an individual peripheral to be reset without having to reset the entire KIO. Requiring bit  $D_3$  to be set to a "1" in order to program the daisy chain configuration allows the user to reset the individual devices without changing the daisy chain. The software reset commands for the individual devices still remain available to the user.

A third function of the Command Register allows the user to obtain use of the additional control signals of the SIO logic instead of the PIA Port. This is done by programming bit  $D_7$  of the Command Register with "1".

---

## PIN DESCRIPTIONS:

**A<sub>0</sub>-A<sub>3</sub>.** Address Bus (inputs, active high, 3-state). Use to select which port/register the current transaction cycle is for.

**ARD, BRDY.** Port Ready (outputs, active high). these signals indicate that the port is ready for a data transfer. In mode 0, it indicates that the port has data available for the peripheral device. In mode 1, it indicates that the port is ready to accept data from the peripheral device. In mode 2, ARDY indicates that Port A has available for the peripheral device, but that it will not be placed onto PA<sub>0</sub>-PA<sub>7</sub> until the ASTB signal is active, while BRDY indicates that Port A is able to accept data from a peripheral device. Note that Port B does not support mode 2 operation and can only be used in mode 3 operation when Port A is programmed for mode 2. These signals are not used in mode 3 operation.

**ASTB, BSTB.** Port Strobe (inputs, active low). These signals indicate that the peripheral device has performed a transfer. In mode 0, it indicates that the peripheral device has accepted the data present on the port pins. In mode 1, it causes the data on the port pins to be latched into Port A. In mode 2, the ASTB signal causes the data in the output data latch of Port A to be placed onto the Port A pins while the BSTB signal will cause the data present on the Port A pins to be latched into the Port A input data latch. The end of the current transaction is noted by the rising edge of these signals. Note that Port B does not support mode 2 operation and can only be used in mode 3 operation when Port A is programmed for mode 2. These signals are not used in mode 3 operation.

**CLK/TRG<sub>0</sub>-CLK/TRG<sub>3</sub>.** External Clock/Timer Trigger (inputs, user selectable active high or low). These four pins correspond to the four counter/timer channels of the KIO. In counter mode, each active edge will cause the downcounter to decrement. In timer mode, an active edge will start the timer.

**CLKOUT.** Clock Out (output, active high). This output is a divide-by-two of the oscillator (XTAL) input.

**CLOCK.** System Clock (input, active high). This clock should be the same as (or a derivative of) the CPU clock. If the CLKOUT is to be used as the system clock, then these two pins should be connected together.

**CS.** Chip Select (input, active low). Used to activate the internal register decoding mechanism and allow the KIO to perform a data transfer to/from the CPU.

**CTSA, CTSB.** Clear to Send (inputs, active low). These signals are modem control signals to their serial channels. When programmed for Auto Enables, a low on these pins will enable their respective transmitters. If not programmed as Auto Enables, these pins may be used as general-purpose input signals.

**D<sub>0</sub>-D<sub>7</sub>.** Data Bus (bidirectional, active high, 3-state). Used for data exchanges between the CPU and the KIO for programming and data transfer. The KIO also monitors the data bus during the RETI instruction cycle to resolve its

**DCDA, DCDB.** Data Carrier Detect (inputs, active low). These signals are modem control signals to their serial channels. When programmed for Auto Enables, a low on these pins will enable their respective receivers. If not programmed as Auto Enables, these pins may be used as general-purpose input signals.

**DTRA, DTRB.** Data Terminal Ready (outputs, active low). These signals are modem control signals for their serial channels. They will follow the state programmed into their respective serial channels. They are multiplexed with Port C, bits 5 and 2 respectively.

**IEI.** Interrupt Enable In (input, active high). This signal is used with IEO to form a priority daisy chain when there is more than one interrupt-driven device. A high on this line indicates that no higher priority device is requesting an interrupt.

**IEO.** Interrupt Enable Out (output, active high). This signal is used with IEI to form a priority daisy chain when there is more than one interrupt-driven device. A high on this line indicates that this device and no higher priority device is requesting an interrupt. A low will block any lower priority devices from requesting an interrupt.

**INT.** Interrupt Request (output, active low, open-drain). When any of the devices within the KIO requests interrupt servicing, this line will be active.

**IORQ.** I/O Request (input, active low). IORQ is used with RD, A0-A3, and CS to transfer data between the KIO and the CPU. When IORQ, RD, and CS are all active, the device selected by A0-A3 transfers data to the CPU. When IORQ and CS are active, but RD is inactive, the device selected by A0-A3 is written into by the CPU. When IORQ and M1 are both active the KIO will respond with an interrupt vector from the highest priority interrupting device.

**M1.** Machine Cycle 1 (input, active low). When M1 is active and RD is active, the Z80 CPU is fetching an instruction from memory; the KIO decodes this cycle to determine if the RETI instruction sequence is being executed. When M1 and IORQ are both active, the KIO decodes the cycle to be an interrupt acknowledge and will respond with a vector from the highest priority interrupting device.

**OSC.** Oscillator (output, active high). This output is a reference clock for the oscillator.

**PA<sub>0</sub>-PA<sub>7</sub>.** Port A Bus (bidirectional, active high, 3-state). This 8-bit bus transfers data between the peripheral device and the port. PA<sub>0</sub> is the least significant bit of the bus.

**PB<sub>0</sub>-PB<sub>7</sub>.** Port B Bus (bidirectional, active high, 3-state). This 8-bit bus transfers data between the peripheral device and the port. PB<sub>0</sub> is the least significant bit of the bus. This port can also supply 1.5 mA at 1.5 volts to drive Darlington transistors.

**PC<sub>0</sub>-PC<sub>7</sub>.** Port C Bus (bidirectional, active high, 3-state). This 8-bit bus transfers data between the peripheral device and the port. PC<sub>0</sub> is the least significant bit of the bus. These pins are multiplexed to provide either an 8-bit parallel port or additional modem control signals for the serial channels.

**$\overline{RD}$ .** Read (input, active low). when  $\overline{RD}$  is active, a memory or I/O read operation is in progress.  $\overline{RD}$  is used with A0-A3,  $\overline{CS}$  and  $\overline{IORQ}$  to transfer data between the KIO and CPU.

**$\overline{RESET}$ .** Reset (input, active low). A low on this pin will force the KIO into a reset condition. This signal must be active for a minimum of three CLOCK cycles. The reset state of the KIO is with the PIO ports in Mode 1 operation and handshakes inactive and interrupts disabled; PIA port in input mode and active; CTC channel counting terminated and interrupts disabled; SIO channels disabled and marking with interrupts disabled. All control registers should be rewritten after a hardware reset.

**$\overline{RTSA}$ ,  $\overline{RTSB}$ .** Request to Send (outputs, active low). These signals are modem control signals for their serial channels. They will follow the inverse state programmed into their respective serial channels. They are multiplexed with Port C, bits 4 and 3 respectively.

**$\overline{RxCA}$ ,  $\overline{RxCB}$ .** Receive Clock (inputs, active low). These clock are used to assemble data in the receiver shift register for their serial channels. Data is sampled on the rising edge of the clock.

**RxDA, RxDB.** Receive Data (inputs, active high). These are the input data pins to the receive shift register for their serial channels.

**$\overline{SYNCA}$ ,  $\overline{SYNCB}$ .** Synchronization (bidirectional, active low). In the asynchronous mode of operation, these pins act much like the  $\overline{CTS}$  and  $\overline{DCD}$  pins. Transitions affect the Sync/Hunt status bit for their respective serial channel but serve no other purpose. They are multiplexed with Port C, bits 6 and 1 respectively.

**$\overline{TxCA}$ ,  $\overline{TxCA}$ .** Transmit Clock (inputs, active low). These clocks are used to transmit data from the transmit shift register for their serial channels. Data is transmitted on the falling edge of the clock.

**TxDA, TxDB.** Transmit Data (outputs, active high). These are the output data pins from the transmitter for their serial channels.

**$\overline{WT/RDYA}$ ,  $\overline{WT/RDYB}$ .** Wait/Ready (outputs, open-drain when programmed as Wait, active high when programmed as Ready). These pins may be programmed as Ready lines for a DMA controller or Wait lines for interface to a CPU. As a Ready line, it indicates (when active) that transmitter or receiver is able to perform a transfer between the serial channel and the DMA. As a Wait line, it dictates (when active), that the CPU should wait until the transmitter or receiver can complete the requested transaction. They are multiplexed with Port C, bits 7 and 0 respectively.

**XTALI.** Crystal/Clock Connection (input, active high).

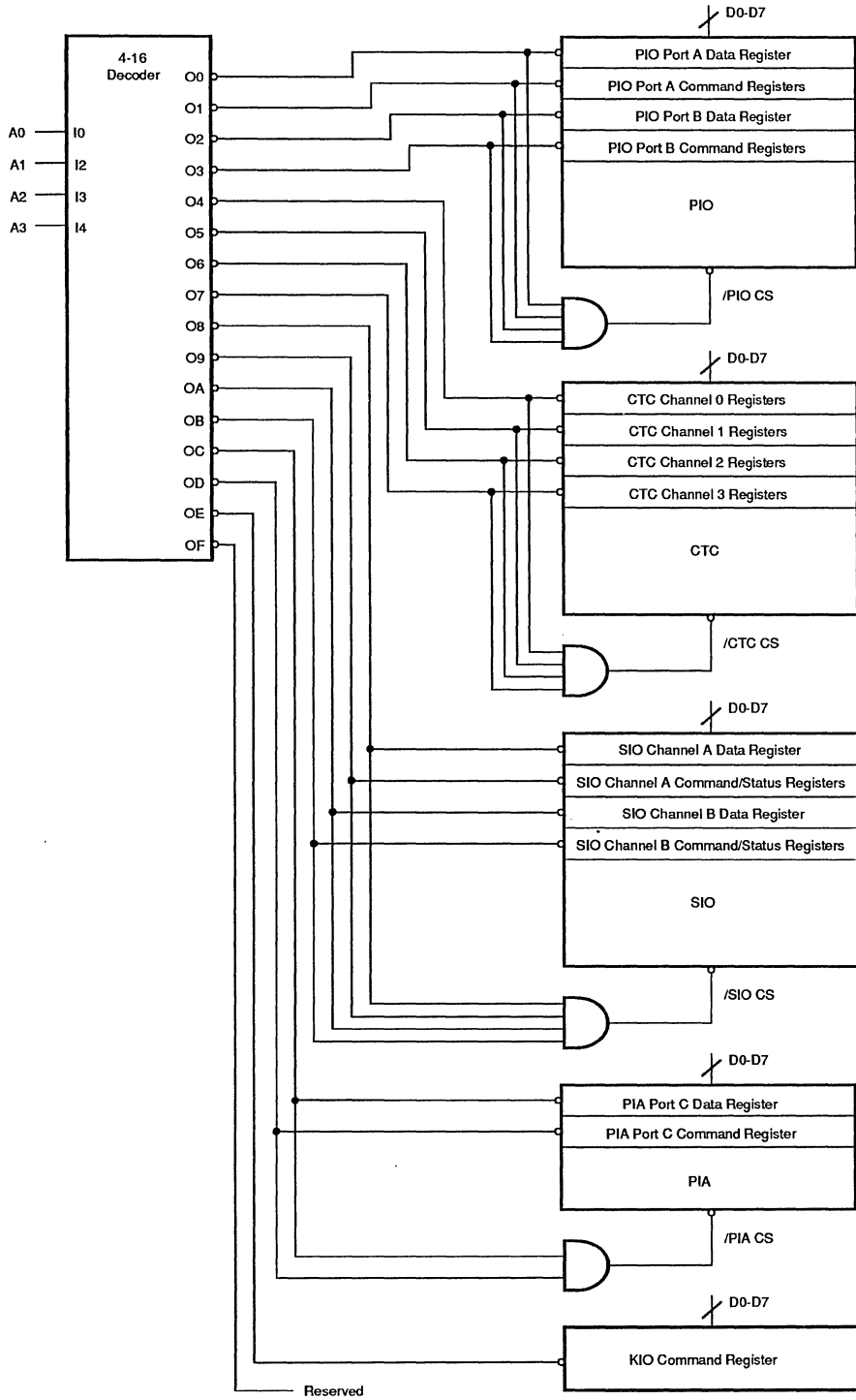
**XTALO.** Crystal Connection (output, active high).

**ZC/TO<sub>0</sub>-ZC/TO<sub>3</sub>.** Zero Count/Timeout (outputs, active high). These four pins correspond to the four counter/timer channels of the KIO. Each pin will become active when its corresponding downcounter reaches a zero count.

Address	A3, A2, A1, A0	$\overline{IORQ}$	$\overline{CS}$	$\overline{RD}$	$\overline{RD}$
				CPU Read	CPU Write
Register 0: PIO Port A Data	0000	0	0	0	1
Register 1: PIO Port A Command	0001	0	0	0	1
Register 2: PIO Port B Data	0010	0	0	0	1
Register 3: PIO Port B Command	0011	0	0	0	1
Register 4: CTC Channel 0	0100	0	0	0	1
Register 5: CTC Channel 1	0101	0	0	0	1
Register 8: SIO Port A Data	1000	0	0	0	1
Register 9: SIO Port A Command/Status	1001	0	0	0	1

Address	A3, A2, A1, A0	$\overline{IORQ}$	$\overline{CS}$	$\overline{RD}$	$\overline{RD}$
				CPU Read	CPU Write
Register 10: SIO Channel B Data	1010	0	0	0	1
Register 11: SIO Channel B Command Status	1011	0	0	0	1
Register 12: PIA Port C Data	1100	0	0	0	1
Register 13: PIA Port C Command	1101	0	0	0	1
Register 14: KIO Command	1110	0	0	0	1
Register 15: Reserved	1111	0	0	0	1

# REGISTER ADDRESS DECODING FOR THE KIO





## REGISTER PROGRAMMING:

**PIO Registers:** For more detailed information, please consult the PIO Technical Manual. These registers apply to channels A and B (see register address decoding).

**Interrupt Vector Word (Figure 8).** The PIO logic unit is designed to work with the Z80 CPU in interrupt Mode 2. This word must be programmed if interrupts are to be used and bit D<sub>0</sub> must be a zero.

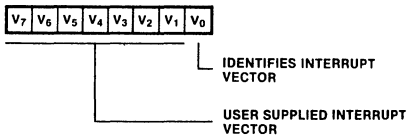


Figure 8: PIO Interrupt Vector Word

**Mode Control Word (Figure 9).** Selects the port operating mode. This word is required and may be written at any time.

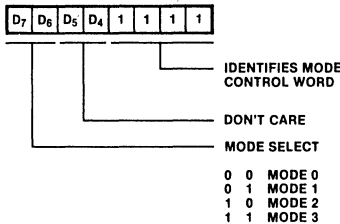


Figure 9: PIO Mode Control Word

**I/O Register Control Word (Figure 10).** When Mode 3 is selected, the Mode Control Word must be followed by the I/O Register Control Word. This word configures the I/O register, which defines which port lines are inputs or outputs. A "1" indicates input while a "0" indicates output. This word is required when in Mode 3.

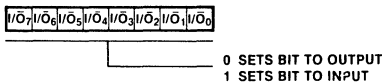
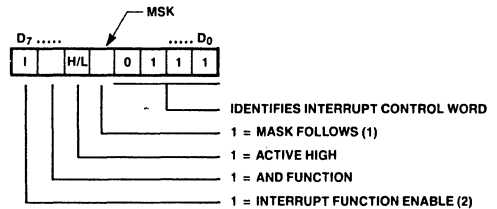


Figure 10: PIO I/O Register Control Word

**Interrupt Control Word (Figure 11).** In Mode 3 operation, handshake signals are not used. Interrupts are generated as a logic function of the input signal levels. The Interrupt Control Word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available: AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits change to the active logic level, an interrupt is triggered). The user can also program which input bits are to be considered as part of this logic function. Bit D<sub>6</sub> sets the logic function, bit D<sub>5</sub> sets the logic level, and bit D<sub>4</sub> specifies a mask control word to follow.



**\*NOTE:**

1. Regardless of the operating mode, setting Bit D<sub>4</sub> = 1 causes any pending interrupts to be cleared.
2. The port interrupt is not enabled until the interrupt function enable is followed by an active M1.

Figure 11: PIO Interrupt Control Word

**Mask Control Word (Figure 12).** This word sets the mask control register, thus allowing any unused bits to be masked off. If any bits are to be masked, then bit D<sub>4</sub> of the interrupt Control Word must be set. When bit D<sub>4</sub> of the Interrupt Control Word is set, then the next word programmed must be the Mask Control Word. To mask an input bit, the corresponding Mask Control Word bit must be a "1".

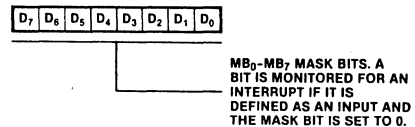


Figure 12: PIO Mask Control Word

**Interrupt Disable Word (Figure 13).** This word can be used to enable or disable a port's interrupts without changing the rest of the port's interrupt conditions.

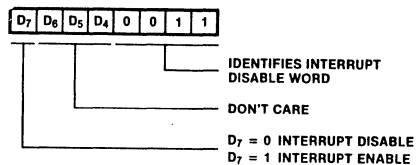


Figure 13: PIO Interrupt Disable Word

**CTC Registers:** For more detailed information, please consult the CTC Technical Manual. These registers apply to channels 0, 1, 2 and 3 (see register address decoding).

**Channel Control Word (Figure 14).** This word sets the operating modes and parameters as described below. Bit D<sub>0</sub> must be a "1" to indicate that this is a Control Word.

**Interrupt Enable.** Bit D<sub>7</sub> enables the interrupt logic so that an interrupt output (INT) can be generated at zero count. Interrupts can be programmed in either mode and may be enabled or disabled at any time.

**Mode.** Bit D<sub>6</sub> selects either Timer Mode or Counter Mode.

**Prescale Factor.** Bit D<sub>5</sub> selects the prescale factor for use in the timer mode. Either divide-by-16 or divide-by-256 is available.

**Clock/Trigger Edge Selector.** Bit D<sub>4</sub> selects the active edge of the CLK/TRG input pulses.

**Timer Trigger.** Bit D<sub>3</sub> selects the trigger mode for timer operation. Either automatic or external trigger may be selected.

**Time Constant.** Bit D<sub>2</sub> indicates that the next word programmed is time constant data for the downcounter.

**Software Reset.** Setting bit D<sub>1</sub> indicates a software reset operation.

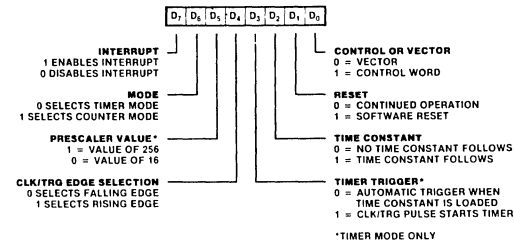


Figure 14: CTC Channel Control Word

**Time Constant Word (Figure 15).** Before a channel can start counting, it must receive a time constant word. The time constant value may be anywhere between 1 and 256, with "0" being accepted as a count of 256.

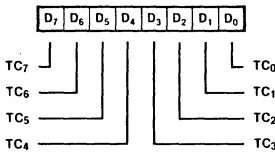


Figure 15: CTC Time Constant Word

**Interrupt Vector Word (Figure 16).** If one or more of the CTC channels have interrupts enabled, then the Interrupt Vector Word must be programmed. Only the five most significant bits of this word are programmed, and bit D<sub>0</sub> must be "0". Bits D<sub>2</sub>-D<sub>1</sub> are automatically modified by the CTC channel when it responds with an interrupt vector.

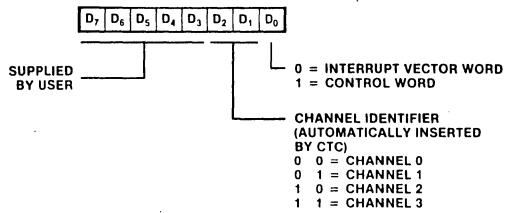
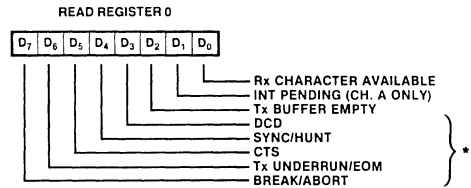


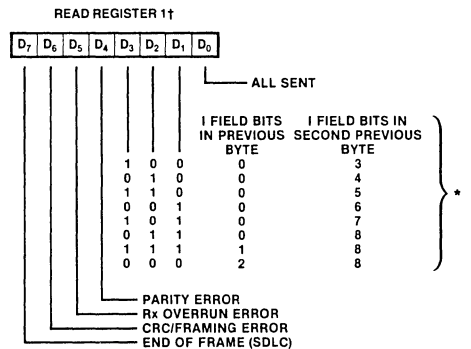
Figure 16: CTC Interrupt Vector Word

**SIO Registers:** For more detailed information, please consult the PIO Technical Manual. These registers apply to channels A and B (see register address decoding).

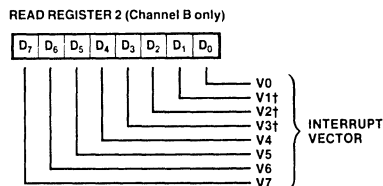
**Read Registers (Figure 17).** The SIO channel B contains three read registers while channel A contains only two that can be read to obtain status information. To read the contents of a register (other than RR<sub>0</sub>), the program must first write a pointer to WR<sub>0</sub> in exactly the same manner as a write register operation. The next I/O read cycle will place the contents of the selected read register onto the data bus.



\* Used With "External/Status Interrupt" Modes



\* Residue data for eight Rx bits/character programmed  
† Used with special receive condition mode



† Variable if "Status Affects Vector" is programmed

Figure 17: SIO Read Registers

**Write Registers (Figure 18).** The SIO channel B contains eight write registers while channel A contains only seven that are programmed to configure the operating modes and characteristics of each channel. With the exception of WR<sub>0</sub>, programming the write registers is a two step opera-

tion. The first operation is a pointer written to WR<sub>0</sub> that point to the selected register. The second operation is the actual control word that is written into the register to configure the SIO channel.

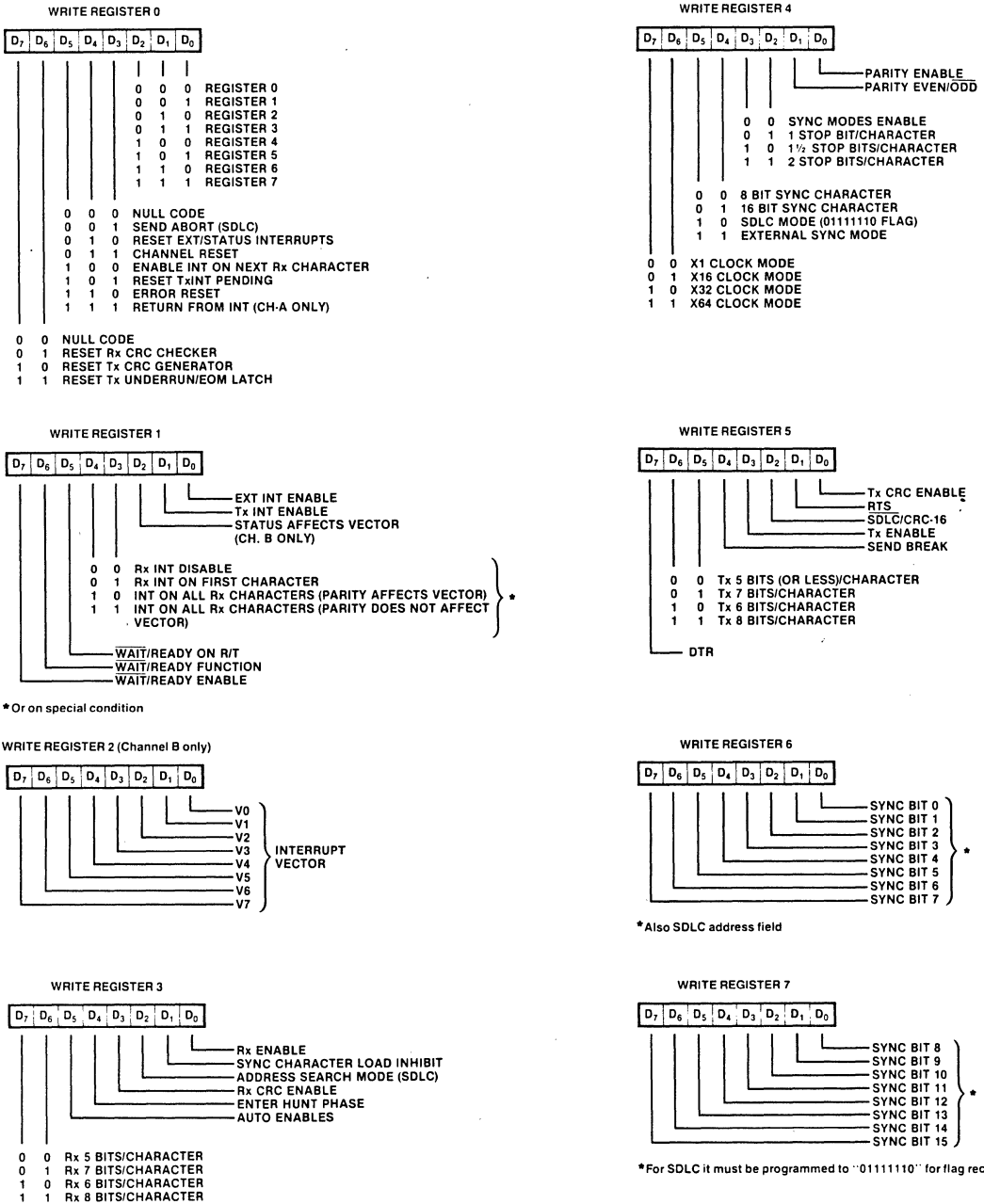


Figure 18: SIO Write Registers

## PIA Registers:

The PIA port can be configured for any combination of input and output bits. The direction is controlled by writing to the PIA Control Register. A "1" written to a bit position will indicate that the respective bit should be an input (Figure 19). All bits are inputs on reset.

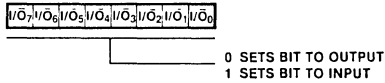


Figure 19: PIA Control Register

## KIO Command Register:

The KIO Command Register is used to program software resets and to configure the internal interrupt daisy chain priority (Figure 20). This register should be programmed before all others. The reset control bits are momentary, writing a "1" will pulse an internal reset signal to the appropriate device.

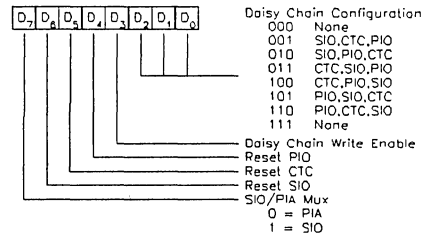


Figure 20: KIO Command Register

## Z84C90 KIO - ENHANCED VERSION

The new revision of the Z84C90 has an enhancement which allows you to 'simulate' the Return From Interrupt sequence, by software. This feature allows interfacing the CPU to other devices in addition to the Z80 CPU (or, Z180/Z280).

### Enhancement - Software RETI.

Writing "1" to a particular command bit location of the KIO, generates a "RETI sequence" internally. Everytime RETI is needed, a "1" is written to this bit.

### Programming of this feature

This revision has one newly assigned register at Register address 15 (this location is "Reserved" on current rev.).

Bit assignment for this register is shown in Figure 21.

Writing "1" to Bit D0 location of this register enables the KIO to simulate a RETI sequence. Writing "0" to this bit has no effect. The upper 7 bits of this register (D7-D1) are reserved and should be programmed as "0". If this register is read, unpredictable data is returned.

### Note:

[1] After writing "1" to this bit, 8 clock cycles of access recovery time is required before making another access to the KIO. If accessing the KIO within this access recovery time, the KIO ignores the transaction on the bus.

[2] When simulating "RETI", the status of the IEI pin is ignored with an internally forced "H". If there are other peripherals on the upper interrupt daisy chain, care must be taken.

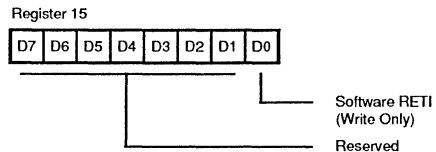


Figure 21. KIO Register 15 - KIO Command Register B

## ABSOLUTE MAXIMUM RATINGS:

Voltage on V <sub>cc</sub> with respect to V <sub>ss</sub>	-0.3V to +7.0V
Voltages on all inputs with respect to V <sub>ss</sub>	-0.3V to V <sub>cc</sub> +0.3V
Operating Ambient Temperature	See Ordering Information
Storage Temperature	-65 C to +150 C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS:

The DC Characteristics and Capacitance sections below apply to the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

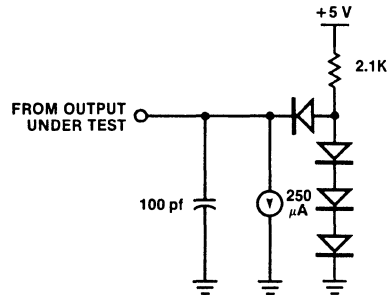
Available operating temperature ranges are:

- S = 0 C to +70 C
- E = -40 C to +100 C

Voltage Supply Range: +5.0V ± 10%

All AC parameters assume a load capacitance of 100 pF. Add 10 ns delay for each 50 pF increase in load up to a maximum of 200pF for the data bus. AC timing measurements are referenced to 1.5 volts (except for CLOCK, which is referenced to the 10% and 90% points).

The Ordering Information section lists temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.



## DC CHARACTERISTICS

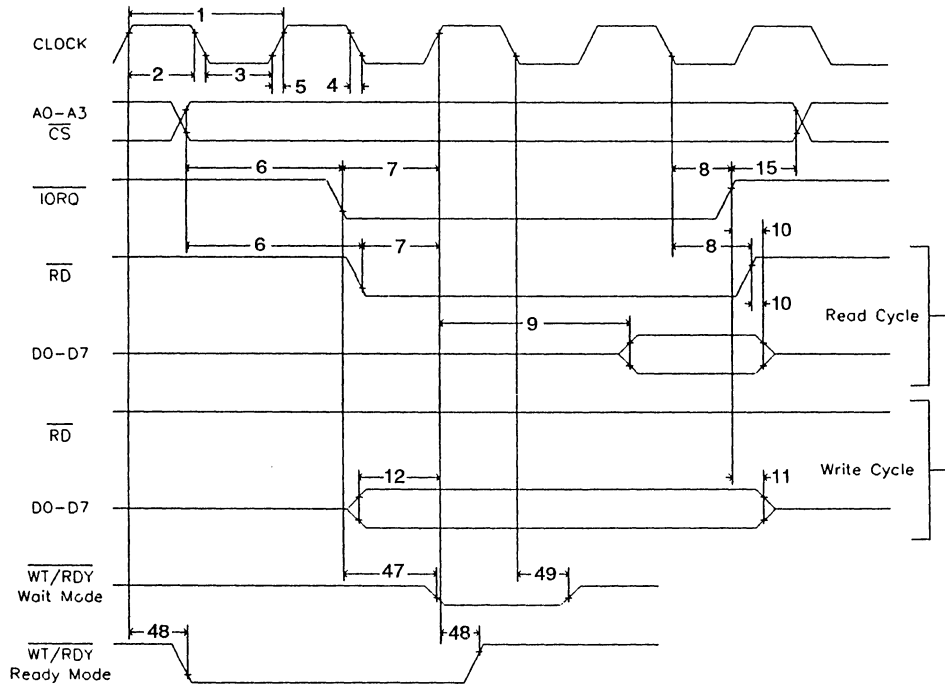
Z84C90 Z80 KIO

Note: The AC parameters #18, 19, 20, 21, 32, 40, 48 for 8 MHz have been changed.

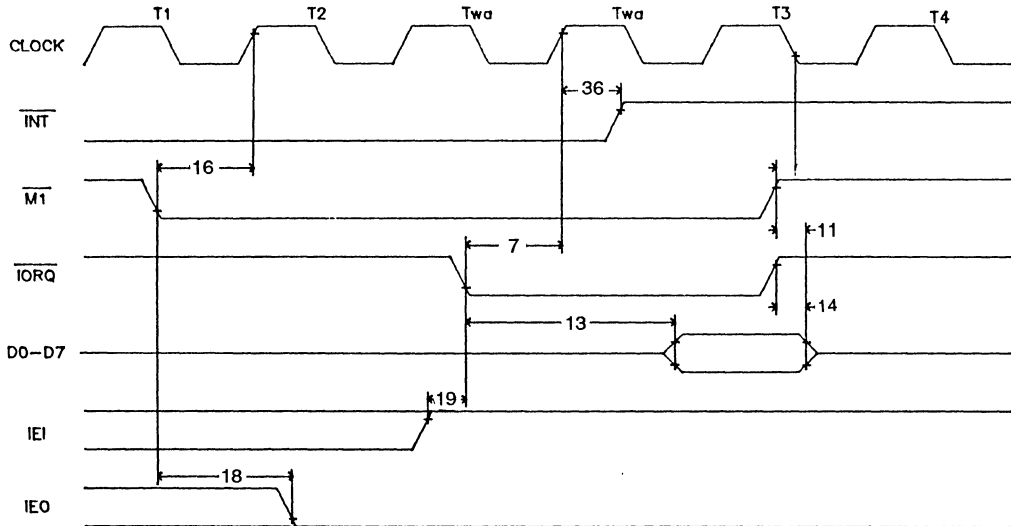
Symbol	Parameter	Min	Max	Typ	Unit	Condition
V <sub>ILC</sub>	Clock Input Low Voltage	-0.3	+0.45		V	
V <sub>IHC</sub>	Clock Input High Voltage	V <sub>cc</sub> -0.6	V <sub>cc</sub> +0.3		V	
V <sub>IL</sub>	Input High Voltage	2.2	V <sub>cc</sub>		V	
V <sub>IH</sub>	Input Low Voltage	-0.3	0.8		V	
V <sub>OL</sub>	Output Low Voltage		0.4		V	I <sub>LO</sub> =2.0 mA
V <sub>OH1</sub>	Output High Voltage	2.4			V	I <sub>OH</sub> =-1.6mA
V <sub>OH2</sub>	Output High Voltage	V <sub>cc</sub> -0.8			V	I <sub>OH</sub> =-250 μA
I <sub>LI</sub>	Input Leakage Current	-10	10		μA	V <sub>IN</sub> =0.4V to V <sub>cc</sub>
I <sub>LO</sub>	3-state Output Leakage Current in Float	-10	10		μA	V <sub>OUT</sub> =0.4V to V <sub>cc</sub>
I <sub>L(SY)</sub>	SYNC Pin Leakage Current	-40	10		μA	V <sub>OUT</sub> =0.4V to V <sub>cc</sub>
I <sub>ODH</sub>	Darlington Drive Current (Port B and ZC/TO0-3)	-1.5				V <sub>OH</sub> =1.5V R <sub>EXT</sub> = 390 Ohms
I <sub>CC1</sub>	Power Supply Current - 8 MHz		15 [1]	7	mA	V <sub>cc</sub> =5V
	- 10 MHz		15 [1]	7	mA	V <sub>IL</sub> =0.2V
	- 12.5 MHz		15	7	mA	V <sub>IH</sub> =V <sub>cc</sub> - 0.2V

Note:

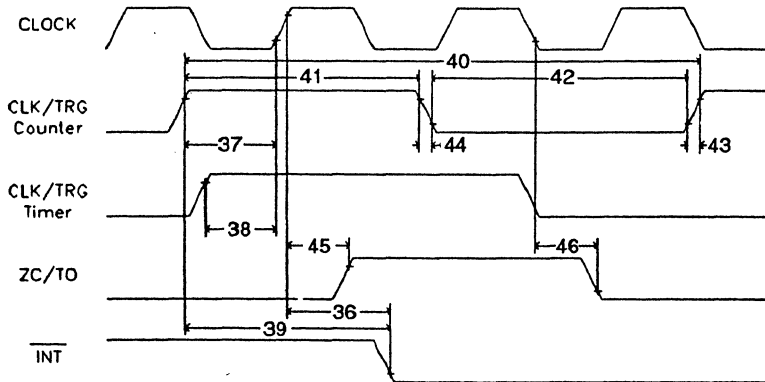
[1] Measurements made with outputs floating and XTAL1 at 0V.  
V<sub>cc</sub>=5.0V ± 10%, unless otherwise specified



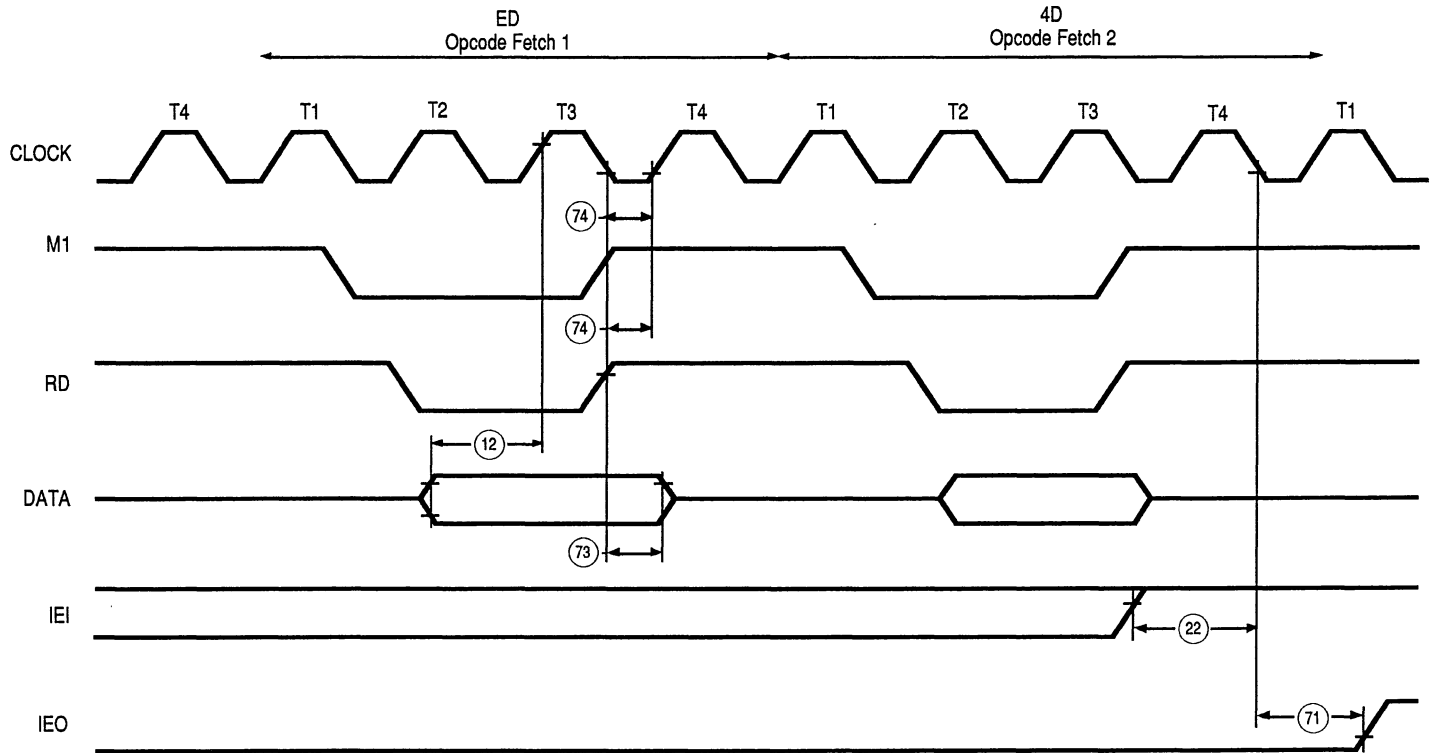
I/O Read/Write Timing ( $\overline{M1} = 1$ )



Interrupt Acknowledge Cycle

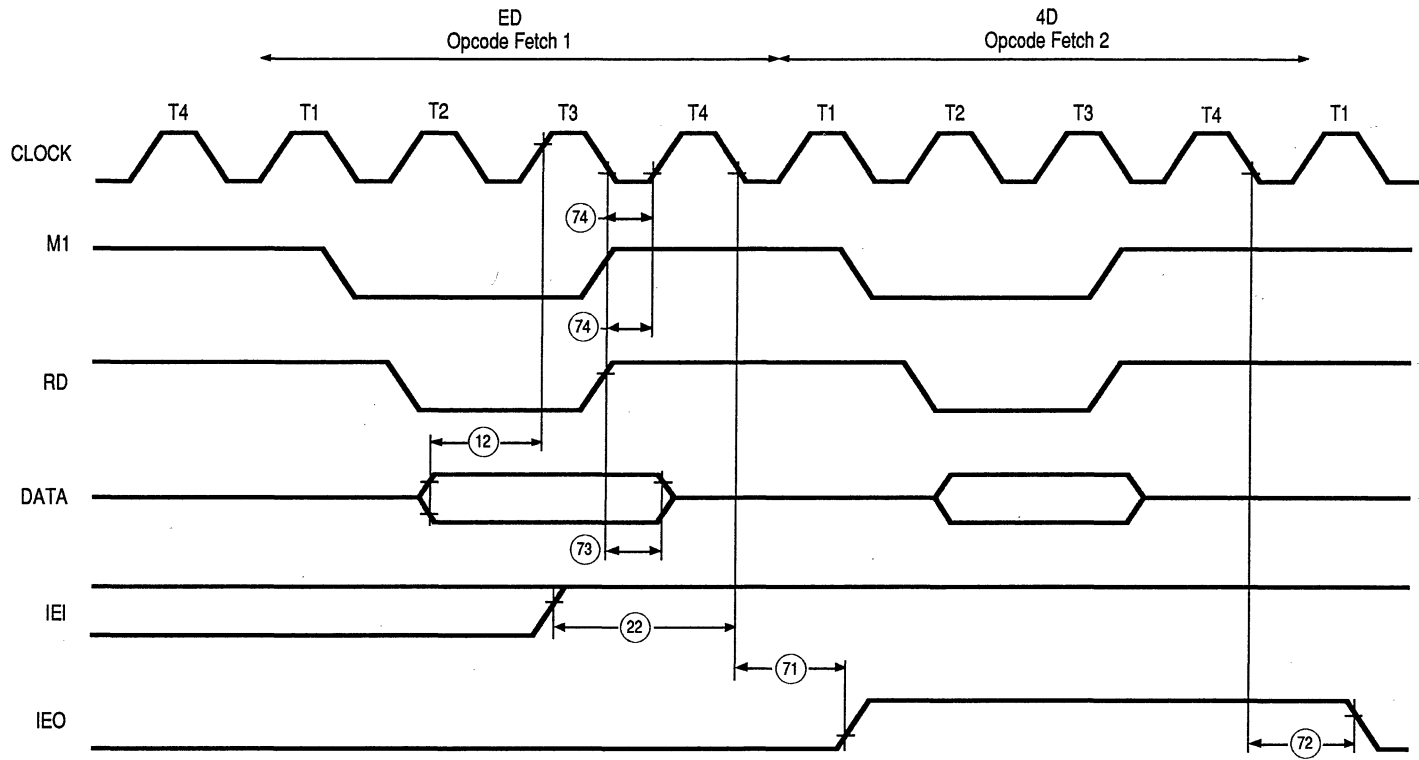


**Counter/Timer Timing**

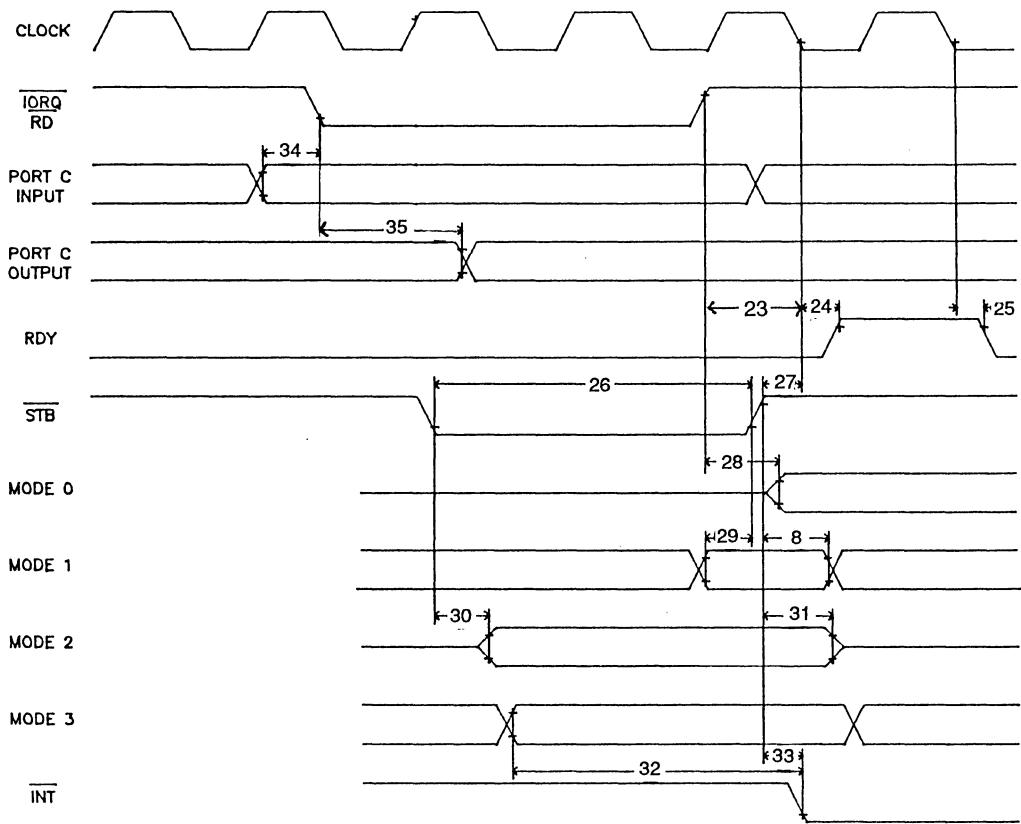


RETI Timing Standard Function

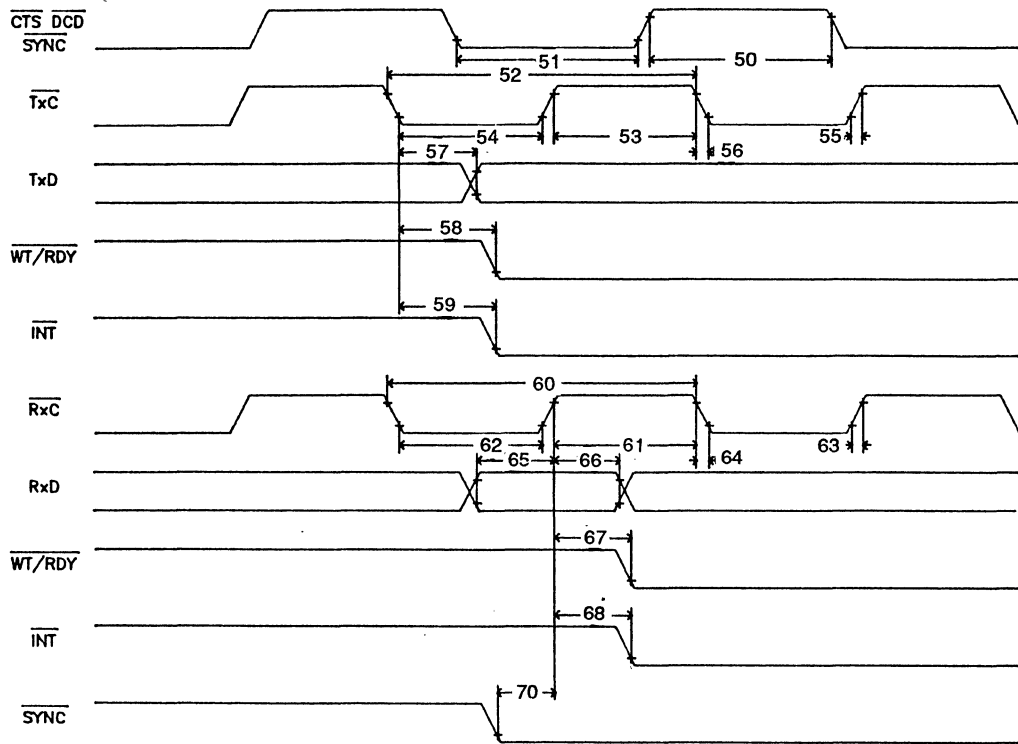




RETI Timing Interrupt Pending



Port I/O Read/Write Timing



Serial I/O Timing

## CAPACITANCE:

Symbol	Parameter	min	max	unit
C <sub>CLOCK</sub>	Clock Capacitance		10	pF
C <sub>IN</sub>	Input Capacitance		10	pF
C <sub>OUT</sub>	Output Capacitance		15	pF

T<sub>A</sub>=25°C, f=1 MHz

## Z84C90 AC CHARACTERISTICS

### BUS Interface Timing

No	Symbol	Parameter	Z84C9008		Z84C9010[5]		Z84C9012[5][6]		Note
			Min	Max	Min	Max	Min	Max	
1	T <sub>c</sub>	Clock Cycle Time	125	DC	100	DC	80	DC	
2	T <sub>wCh</sub>	Clock Pulse Width (High)	55	DC	42	DC	32	DC	
3	T <sub>wCl</sub>	Clock Pulse Width (Low)	55	DC	42	DC	32	DC	
4	T <sub>IC</sub>	Clock Fall Time		10		10		10	
5	T <sub>rC</sub>	Clock Rise Time		10		10		10	
6	T <sub>sA(Rlf)</sub>	Address, /CS Setup Time to /RD, /IORQ Fall	50		40		30		
7	T <sub>sRI(Cr)</sub>	/IORQ, /RD to clock rise setup time	60		50		40		
8	T <sub>h</sub>	Hold Time for Specified Setup	15		15		15		
9	T <sub>dCr(DO)</sub>	Clock Rising to Data Out Delay		100		80		65	
10	T <sub>dRlr(DOz)</sub>	/RD, /IORQ Rise to Data Out Float Delay		75		60		55	
11	T <sub>hRDr(D)</sub>	/M1, /RD, /IORQ Rise to Data <b>M1 cycle</b>	15		15		15		
12	T <sub>sD(Cr)</sub>	Data In to Clock Rise Setup Time	30		25		22		
13	T <sub>dIOl(DOI)</sub>	/IORQ Fall to Data Out Delay (INTACK Cycle)		95		95		95	[1]
14	T <sub>hOr(D)</sub>	/IORQ Rise to Data <b>Float</b>	15		15		15		
15	T <sub>hOr(A)</sub>	/IORQ Rise to Address Hold	15		15		15		
16	T <sub>sM1f(Cr)</sub>	/M1 Fall to Clock Rise Setup Time	40		40		40		
17	T <sub>sM1r(Cf)</sub>	/M1 Rise to Clock <b>Fall</b> Setup Time (M1 Cycle)	-15		-15		-15		
18	T <sub>dM1f(IEOI)</sub>	/M1 Fall to IEO Fall Delay (Interrupt Immediately Preceding M1 Fall)		*		*		*	[4]
19	T <sub>sIEI(IOf)</sub>	IEI to /IORQ Fall Setup Time (INTACK Cycle)		*		*		*	[4]
20	T <sub>dIEIf(IEOI)</sub>	IEI Fall to IEO Fall Delay		160		150		125	[4]
21	T <sub>dIEf(IEOr)</sub>	IEI Rise to IEO Rise Delay (After ED Decode)		160		150		125	[4]
22	T <sub>sIEI(Cf)</sub>	IEI to Clock Fall Setup (For 4D Decode)	50		40		30		
23	T <sub>sOr(Cf)</sub>	/IORQ Rise to Clock Fall Setup Time (To Activate RDY on Next Clock Cycle)	100		100		100		

## AC CHARACTERISTICS:

### PIO Timing

No	Symbol	Parameter	Z84C9008		Z84C9010[5]		Z84C9012[5][6]		Note
			Min	Max	Min	Max	Min	Max	
24	TdCl(RDYr)	Clock Fall to RDY Rise Delay		100		100		100	
25	TdCl(RDYf)	Clock Fall to RDY Fall Delay		100		100		100	
26	TwSTB	/STB Pulse Width	100		80		60		
27	TsSTBr(Cf)	/STB Rise to Clock Fall Setup Time (To Activate RDY On Next Clock Cycle)	100		100		100		
28	TdIOf(PD)	/IORQ Fall to Port Data Stable Delay (Mode 0)		140		120		110	
29	TsPD(STBr)	Port Data to /STB Rise Setup Time (Mode 1)	140		75		75		
30	TdSTBi(PD)	/STB Fall to Port Data Stable (Mode 2)		150		120		110	
31	TdSTBr(PDz)	/STB Rise to Port Data Float Delay (Mode 2)		140		120		110	
32	TdPD(INTf)	Port Data Match to /INT Fall Delay (Mode 3)		250		200		160	
33	TdSTBr(INTf)	/STB Rise to /INT Fall Delay		290		220		190	
34	TsPD(RIf)	PIA Port Data to /RD, /IORQ Fall Setup	TBD		TBD		TBD		
35	TdCr(PD)	Clock Rise to PIA Port Data Valid Delay		80		80		80	
35	TdIO (PD)	IORQ Fall to PIA Port Data Valid							

### CTC Timing

No	Symbol	Parameter	Z84C9008		Z84C9010[5]		Z84C9012[5] [6]		Note
			Min	Max	Min	Max	Min	Max	
36	TdCr(INTf)	Clock Rise to /INT Fall Delay		(TcC+100)		(TcC+80)		(TcC+75)	
37	TsCTR (Cr)†	CLK/TRG to Clock Rise Setup Time for Immediate Count	90		90		75		
38	TsCTR(Ct)	CLK/TRG to Clock Rise Setup Time for Enabling of Prescaler on Following Clock Rise	90		90		75		
39	TdCTR (INTf)	CLK/TRG to /INT Fall Delay							
		TsCTR(C) Satisfied		(36)+(38)		(36)+(38)		(36)+(38)	
		TsCTR(C) Not Satisfied		(1)+(36)+(38)		(1)+(36)+(38)		(1)+(36)+(38)	
40	TcCTR	CLK/TRG Cycle Time	(2TcC)	DC	(2TcC)	DC	(2TcC)	DC	[2]
41	TwCTRh	CLK/TRG Width (High)	90	DC	90	DC	75	DC	
42	TwCTRl	CLK/TRG Width (Low)	90	DC	90	DC	75	DC	
43	TrCTR	CLK/TRG Rise Time		30		30		30	
44	TfCTR	CLK/TRG Fall Time		30		30		30	
45	TdCr(ZCr)	Clock Rise to ZC/TO Rise Delay		80		80		80	
46	TdCl(ZCf)	Clock Fall to ZC/TO Fall Delay		80		80		80	

## Z84C90 AC CHARACTERISTICS

### SIO Timing

No	Symbol	Parameter	Z84C9008		Z84C9010[5]		Z84C9012[5][6]		Note
			Min	Max	Min	Max	Min	Max	
47	TdIO(W/Rf)	/IORQ or /CE Fall to /W//RDY Delay (Wait Mode)		130		110		110	
48	TdCr(W/Rf)	Clock Rise to /W//RDY Delay (Ready Mode)		85		85		85	
49	TdCf(W/Rz)	Clock Fall to /W//RDY Floal Delay (Wait Mode)		90+RC		80+RC		75+RC	[7]
50	TwPh	Pulse Width (High)	150		120			100	
51	TwPl	Pulse Width (Low)	150		120			100	
52	TcTxC	/TxC Cycle Time	250		200			160	
53	TwTxCl	/TxC Width (Low)	85		80			70	
54	TwTxCh	/TxC Width (High)	85		80			70	
55	TrTxC	/TxC Rise Time		60		60		60	
56	TfTxC	/TxC Fall Time		60		60		60	
57	TdTxCl(TxD)	/TxC Fall to TxD Delay		160		120		115	
58	TdTxCl(W/RRf)	/TxC Fall to /W//RDY Fall Delay (Ready Mode)	5	9	5	9	5	9	[3]
59	TdTxCl(INTf)	/TxC Fall to /INT Fall Delay	5	9	5	9	5	9	[3]
60	TcRxC	/RxC Cycle Time	250		200			160	
61	TwRxCh	/RxC Width (High)	85		80			70	
62	TwRxCl	/RxC Width (Low)	85		80			70	
63	TrRxC	/RxC Rise Time		60		60		60	
64	TfRxC	/RxC Fall Time		60		60		60	
65	TsRxD(RxC;)	RxD to /RxC Rise Setup Time (X1 Mode)	0		0			0	
66	ThRxCr(RxD)	/RxC Rise to RxD Hold Time (X1 Mode)	80		60			50	
67	TdRxCr(W/RRf)	/RxC Rise to /W//RDY Fall Delay (Ready Mode)	10	13	10	13	10	13	[3]
68	TdRxCr(INTf)	/RxC Rise to /INT Fall Delay	10	13	10	13	10	13	[3]
69	TdRxCr(SYNCf)	/RxC Rise to /SYNC Fall Delay (Output Modes)	4	7	4	7	4	7	[3]
70	TsSYNCf(RxCr)	/SYNC Fall to /RxC Rise Setup (External Sync Modes)	-100		-100			-100	

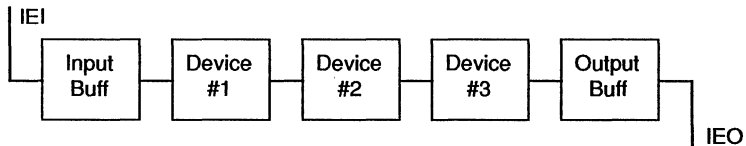
## Z84C90 AC CHARACTERISTICS

### SIO Timing (Continued)

No	Symbol	Parameter	Z84C9008		Z84C9010 [5]		Z84C9012 [5,6]		Note
			Min	Max	Min	Max	Min	Max	
71	TdCf(IEOr)	Clock Fall to IEO Rise Delay		90		75		60	
72	TdCf(IEOf)	Clock Fall to IEO Rise Delay		110		90		75	
73	ThDI(M1r,RDr)	Data Hold Time to /M1 Rise or /RD Rise	0		0		0		
74	TsM1/RD(C)	Setup Time For M1 and RD to Clock Rising (with Data Valid)	20		20		20		

#### Notes:

- [1] If the CPU is Z80 and Clock speed is above 8 MHz, one wait state is required to meet this parameter.
- [2] Counter Mode only; when using a cycle time less than 3TcC, parameter #37 must be met.
- [3] Units equal to System Clock Cycles.
- [4] Parameters #18, 19, 20 and 21. These parameters are daisy-chain timing and calculated value and vary depending on the inside daisy-chain configuration which is specified in the KIO command register. Inside the Z80 KIO, the daisy-chain is figured below.
- [5] If the CPU is a Z80 CPU; and if it is required to have multiple Z80 peripherals in the system. For this case, only one Z80 peripheral other than Z84C90 is the maximum number of peripherals unless the time period between \*M1 active to /IORQ active\* is extended.
- [5] \* In All Modes, the System Clock rate must be at least five times the maximum data rate.
- [6] Timings for 12.5 MHz are preliminary.
- [7] Open drain output add register capacitor time constraint (RC) to spec. value.



**Internal Daisy Chain Configuration**

## Z84C90 AC CHARACTERISTICS (Continued)

Table for Note [4] parameters

No	Parameter	8 MHz		10 MHz		12.5 MHz	
		Min	Max	Min	Max	Min	Max
18	TdM1(IEO) (PIO at #1)		160 ns		150 ns		125 ns
	(CTC at #1)		180 ns		150 ns		125 ns
	(SIO at #1)		230 ns		200 ns		160 ns
19	TsIEI(IO) (PIO at #3)	170 ns		140 ns		115 ns	
	(CTC at #3)	170 ns		160 ns		135 ns	
	(SIO at #3)	180 ns		160 ns		130 ns	
20	TdIEI(IEOf)		160 ns		150 ns		125 ns
21	TdIEI(IEOr)		160 ns		150 ns		125 ns

To calculate Z80 KIO daisy-chain timing, use the Z80 PIO, CTC and SIO with I/O buffers on the chain. The following are calculation formulas:

### Parameter #18

M1 falling to IEO delay  $T_{sM1(IEO)} = T_{dM1(IEO)\#1} + T_{dIEI(IEO)\#2} + T_{dIEI(IEO)\#3} + (\text{Output Buffer Delay})$

### Parameter #19

IEI to IORQ falling setup time  $T_{sIEI(IO)} = T_{dIEI(IEO)\#1} + T_{dIEI(IEO)\#2} + T_{sIEI(IO)\#3} + (\text{Input Buffer delay})$

### Parameter #20

IEI falling to IEO falling delay-  $T_{dIEI(IEOf)} - T_{dIEI(IEOf)PIO} + T_{dIEI(IEOf)CTC} + T_{dIEI(IEOf)SIO} + (\text{Input Buffer delay}) + (\text{output Buffer Delay})$

### Parameter #21

IEI rising to IEO rising delay (After ED decode)-  $T_{dIEI(IEOr)} = T_{dIEI(IEOr)PIO} + T_{dIEI(IEOr)CTC} + T_{dIEI(IEOr)SIO} + (\text{Input Buffer delay}) + (\text{output Buffer Delay})$

\* Where  $T_{dIEI(IEO)}$  is the worst number between  $T_{dIEI(IEOr)}$  and  $T_{dIEI(IEOf)}$ .



The following are numbers for the above calculations:

	8 MHz		10 MHz		12.5 MHz	
	Min	Max	Min	Max	Min	Max
Input Buffer delay	10 ns		10 ns		10 ns	
Output Buffer delay	10 ns		10 ns		10 ns	

8 MHz	PIO part		CTC part		SIO part	
	Min	Max	Min	Max	Min	Max
TdM1(IEO)	60 ns		80 ns		120 ns	
TsIEI(IO)	70 ns		70 ns		70 ns	
TdIEI(IEOf)		50 ns		50 ns		40 ns
TdIEI(IEOr)		50 ns		50 ns		40 ns

10 MHz	PIO part		CTC part		SIO part	
	Min	Max	Min	Max	Min	Max
TdM1(IEO)	60 ns		60 ns		90 ns	
TsIEI(IO)	50 ns		70 ns		50 ns	
TdIEI(IEOf)		50 ns		50 ns		30 ns
TdIEI(IEOr)		50 ns		50 ns		30 ns

12.5 MHz	PIO part		CTC part		SIO part	
	Min	Max	Min	Max	Min	Max
TdM1(IEO)	50 ns		50 ns		70 ns	
TsIEI(IO)	40 ns		60 ns		40 ns	
TdIEI(IEOf)		40 ns		40 ns		25 ns
TdIEI(IEOr)		40 ns		40 ns		25 ns

**Note [4]** (Continued)

When using an interrupt from only a portion of the Z80KIO, these numbers are smaller than the values shown above.

For more details about the "Z80 daisy-chain structure", please refer to the application note "Z80 Family Interrupt Structure," which is included in the Z80 Data Book.

## Z84011/C11

### PARALLEL I/O CONTROLLER

#### FEATURES

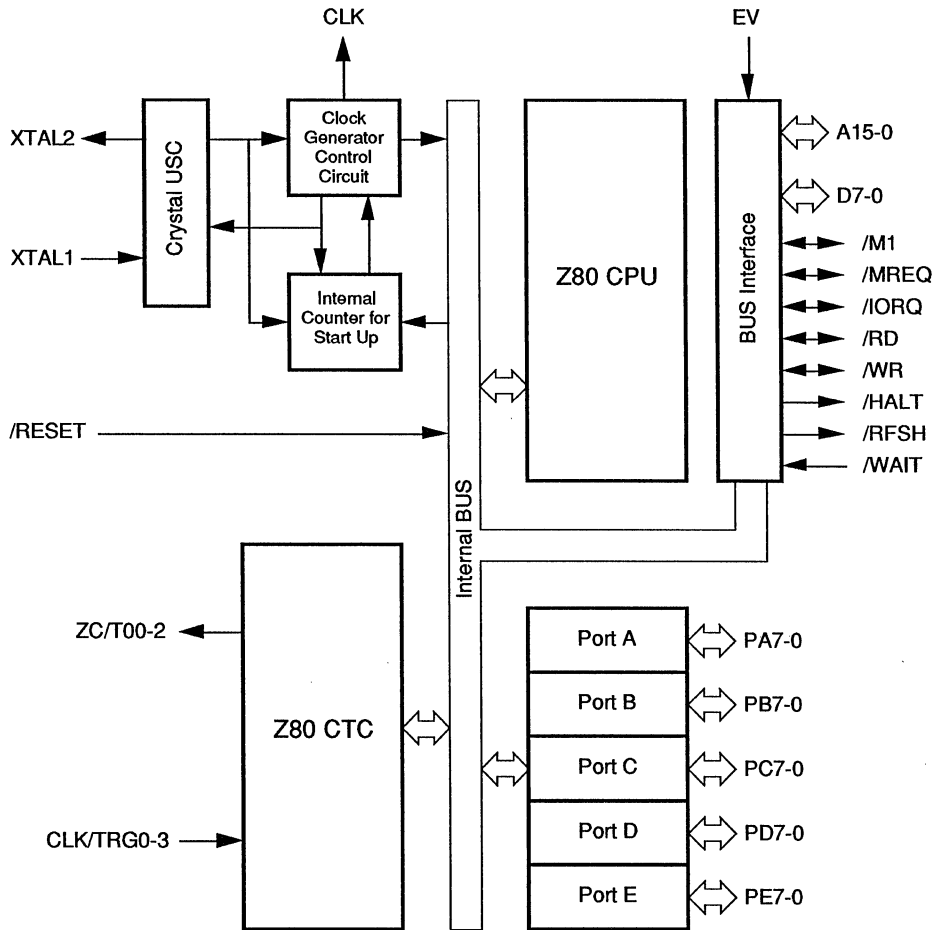
- Z84C00 Z80 CPU with CGC, Z84C30 Z80 CTC, five 8-bit parallel ports.
- High speed operation (6/10 MHz).
- Low power consumption in four operation modes:
  - 45 mA Typ. (Run mode)
  - 6 mA Typ. (Idle1 mode)
  - 9 mA Typ. (Idle2 mode; not applicable to Z84011)
  - 1  $\mu$ A Typ. (Stop mode)
- Wide operational voltage range (5V  $\pm$ 10%).
- TTL/CMOS compatible.
- Z84011 features:
  - Z84C00 Z80 CPU.
  - On-chip four channel Counter Timer Controller(Z80 CTC).
  - Built-in Clock Generator Controller(CGC).
  - Five 8-bit parallel ports.
  - 100-pin QFP Package.
  - Noise filter to CLK/TRG inputs of the Z80 CTC.
- Z84C11 features:
  - All Z84011 features.
  - Support of Idle 2 Mode.
  - Built-in Watch Dog Timer (WDT).
  - Power-on Reset and Reset Extension.
  - Wait State Generator.
  - Simplified EV Mode Selection.
  - Crystal Divide-by-One Option.
  - External Clock Input Option.

#### GENERAL DESCRIPTION

The Z84011 and Z84C11 Parallel I/O Controllers (PIC) are CMOS 8-bit microprocessors. They are integrated with the CTC and five 8-bit parallel ports into a single 100-pin QFP (Quad Flat Pack) package. The Z84C11 is an upward compatible version of the Z84011. Figure 1(a) shows the block diagram of Z84011, and Figure 1(b) shows the block diagram of the Z84C11. Figure 2 has pin assignments for both versions. These high end superintegrated Parallel I/O Controllers are targeted for a broad range of applica-

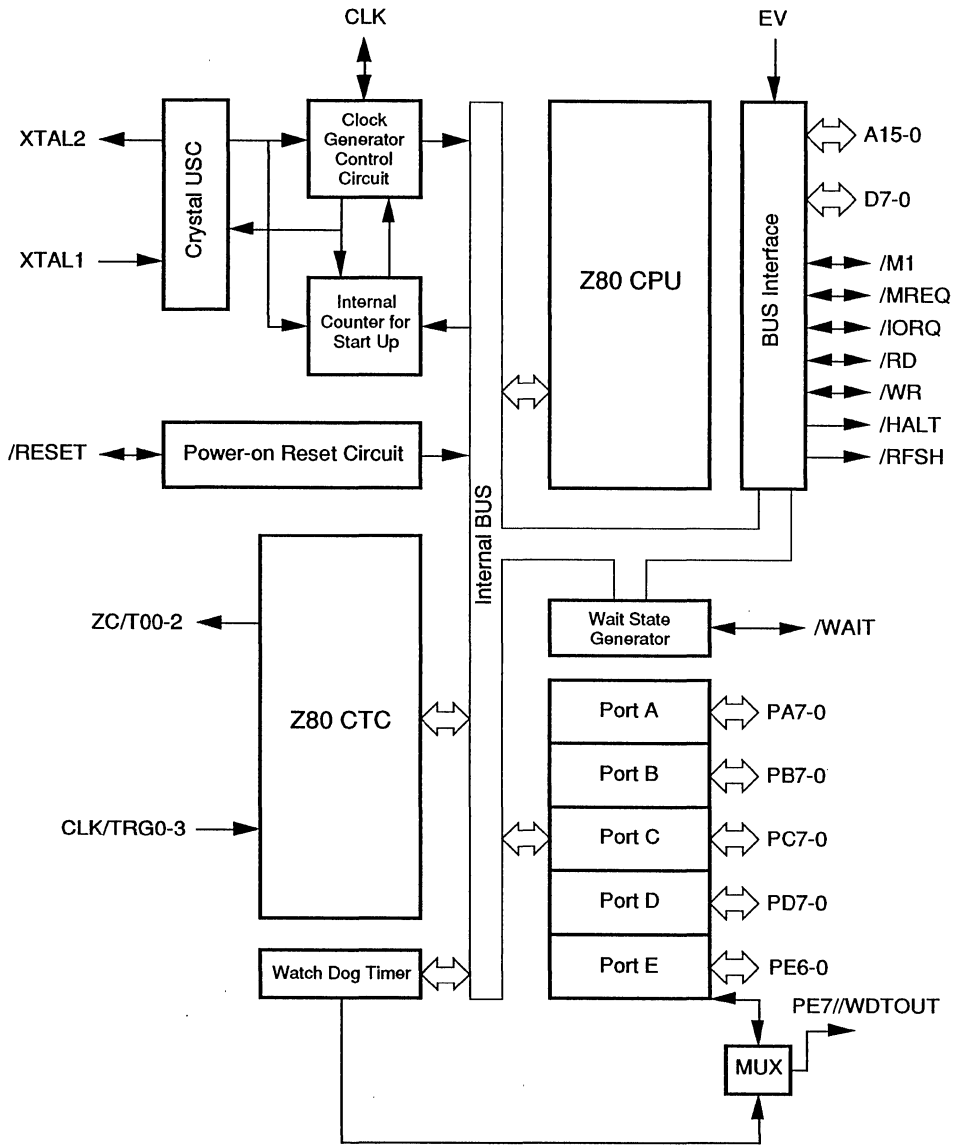
tions ranging from embedded controller to enhancement/ cost reductions of existing hardware using Z80 based discrete peripherals.

Hereinafter, the word PIC on the description covering both versions (Z84C11 and Z84011) is used. Use Z84C11 on the description which applies only to the Z84C11, and use Z84011 which applies only to the Z84011.



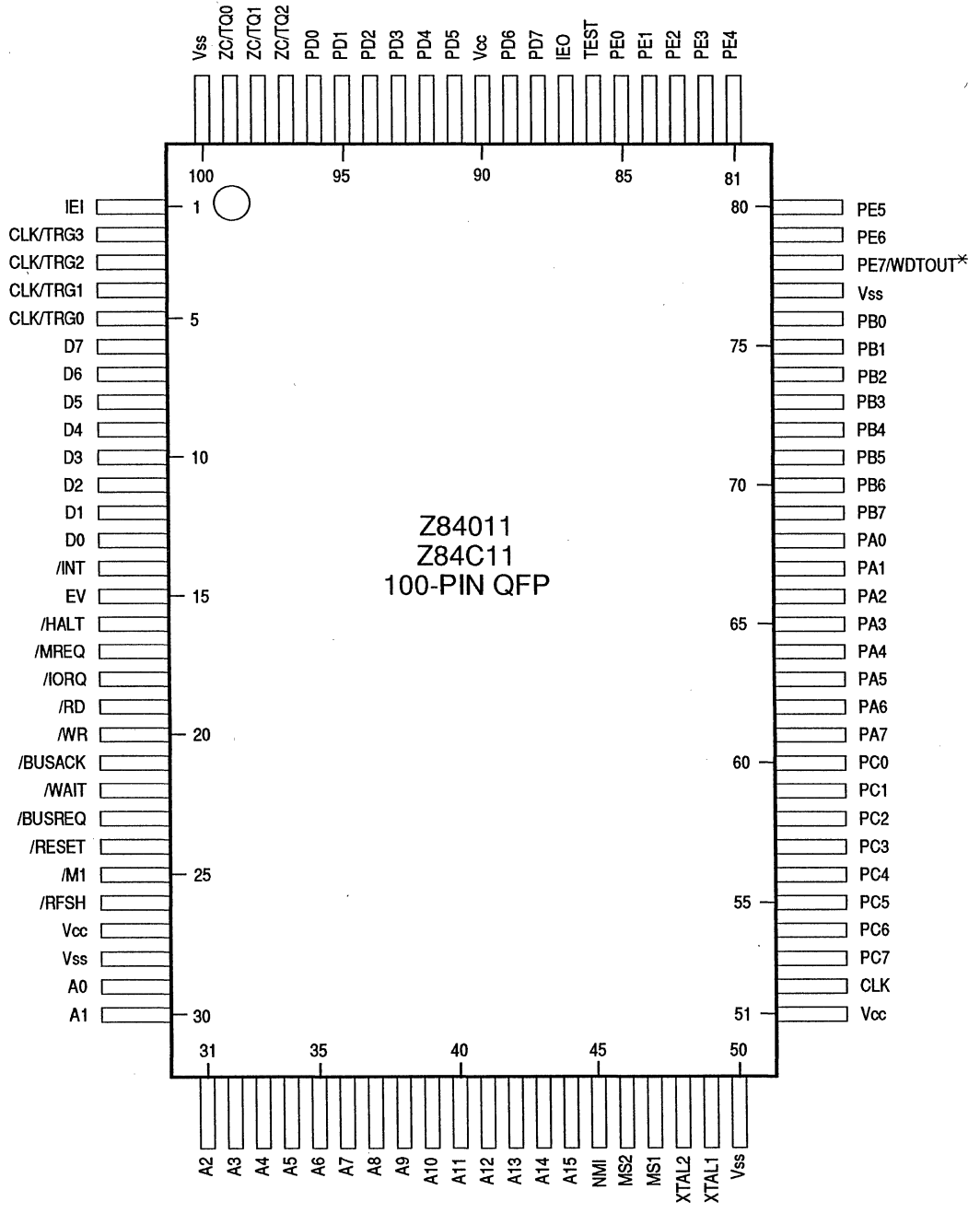
(a) Z84011 Functional Block Diagram

Figure 1. PIC Functional Block Diagram



(b) Z84C11 Functional Block Diagram

Figure 1. PIC Functional Block Diagram (Continued)



\* PE7 for Z84011

Figure 2. PIC Pin Assignment

---

## PIN DESCRIPTION

The pin assignment is shown in Figure 2. Following is the description on each pin. For the description and the pin

number, if stated as "X11", it applies to both the Z84C11/Z84011. Otherwise, C11 for Z84C11 and O11 for Z84011.

---

## CPU SIGNALS

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
A15-A0	16	(44-29)	I/O	16-bit address bus. Specifies I/O and memory addresses to be accessed. During the refresh period, addresses for refreshing are output. The bus is an input when the external bus is accessing the on-chip peripherals.
D7-D0	8	(6-13)	I/O	8-bit bidirectional data bus. When the on-chip CPU is accessing on-chip peripherals, these lines are set to output and hold the data to/from on-chip peripherals.
/RD	1	(19)	I/O	Read signal. CPU read signal for accepting data from a memory or I/O device. When an external master is accessing the on-chip peripherals, it is an input signal.
/WR	1	(20)	I/O	Write Signal. This signal is output when data, to be stored in a specified memory or peripheral LSI, is on the MPU data bus. When an external master is accessing the on-chip peripherals, it is an input signal.
/MREQ	1	(17)	Output, 3-State	Memory request signal. When an effective address for memory access is on the address bus, "0" is output. When an external master controls the bus, this signal is tri-stated.
/IORQ	1	(18)	I/O	I/O request signal. When addresses for I/O are on the lower 8 bits (A7-A0) of the address bus in the I/O operation, "0" is output. In addition, the /IORQ signal is output with the /M1 signal at the time of the interrupt acknowledge cycle. This informs peripheral LSI of the interrupt response vector state when on the data bus. When an external master controls the bus, it is an input signal.
/M1	1	(25)	I/O	Machine Cycle "1". /MREQ and "0" are output together in the operation code fetch cycle. /M1 is output for every op-code fetch when a two byte op-code is executed. In the maskable interrupt acknowledge cycle, this signal is output together with /IORQ. When an external master controls the bus, it is an input signal.

---

---

## CPU SIGNALS

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
/RFSH	1	(26)	Output(011), Output/3-State (C11)	The refresh signal. When the dynamic memory refresh address is on the low order byte on the address bus, this pin goes active along with /MREQ signal.
<b>Note:</b> For the Z84011 the /RFSH is not tri-stated during EV mode.				
<b>Note:</b> For the Z84C11 the /RFSH is tri-stated during EV mode.				
/INT	1	(14)	I/O (Open Drain)	Maskable interrupt request signal. Interrupt is generated by peripheral LSI. This signal is accepted if the Interrupt enable Flip-Flop (IFF) is set to "1". The /INT signal of the CTC is internally wired-OR without pull-up resistors and requires external pull-up. The interrupts from on-chip CTC go out from this pin.
/NMI	1	(45)	Input	Non-maskable interrupt request signal. This interrupt request has a higher priority than the maskable interrupt request and does not rely upon the state of the Interrupt enable Flip-Flop (IFF).
/HALT	1	(16)	Output, 3-State	Halt signal. Indicates that the CPU has executed a HALT instruction. This signal is tri-stated in EV mode.
/BUSREQ	1	(23)	Input	Bus request signal. /BUSREQ requests placement of the address bus, data bus, /MREQ, /IORQ, /RD and /WR signals into the high impedance state. /BUSREQ is normally wired-OR and a pull-up resistor is externally connected.
/BUSACK	1	(21)	Output(011), Output, 3-State (C11)	Bus acknowledge signal. In response to /BUSREQ signal, /BUSACK informs a peripheral LSI that the address bus, data bus, /MREQ, /IORQ, /RD, and /WR signals have been placed in the high impedance state.

**Note:**  
For the Z84011 the /BUSACK will not be tri-stated during EV mode. For the Z84C11 the /BUSACK will be tri-stated during EV mode.

/WAIT	1	(22)	Input(011), I/O(C11)	Wait signal. /WAIT informs the CPU that specified memory or peripheral is not ready for data transfer. As long as /WAIT signal is active, MPU is continuously kept in the wait state.
-------	---	------	-------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Note:**  
For the Z84C11, the /WAIT pin becomes an output to bring out on-chip Wait State Generator during EV mode.

---

## CTC SIGNALS

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
CLK/TRG0 - CLK/TRG3	4	(2-5)	Input.	External Clock/Trigger input. These four CLK/TRG pins correspond to four Counter/Timer Channels. In the counter mode, each active edge causes the downcounter to decrement by one. In timer mode, an active edge starts the timer. It is program selectable whether the active edge is rising or falling.
ZC/TO0 - ZC/TO2	3	(97-99)	Output.	Zero count/timer out signal. In either timer or counter mode, pulses are output when the down counter has reached zero. The Counter/Timer Channel 3 does not have this output.

---

## GENERAL PURPOSE I/O PORT

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
PA7-PA0	8	(61-68)	I/O	General purpose I/O port (Port A). These lines are configured as an input or an output, bit by bit. On Reset, set as "all input."
PB7-PB0	8	(69-76)	I/O	General purpose I/O port (Port B). These lines are configured as an input or an output, bit by bit. On Reset, set as "all input."
PC7-PC0	8	(53-60)	I/O	General purpose I/O port (Port C). These lines are configured as an input or an output, bit by bit. On Reset, set as "all input."
PD7-PD0	8	(88, 89, 91-96)	I/O	General purpose I/O port (Port D). These lines are configured as an input or an output, bit by bit. On Reset, set as "all input."
PE6-PE0	7	(85-79)	I/O	General purpose I/O port (Port E) These lines are configured as an input or an output, bit by bit. On Reset, set as "all input."
PE7 (011 Only)	1	(78)	I/O	General purpose I/O port (Port E 7). This pin is configured as an input or an output. On Reset, set as "Input."
PE7/WDTOUT (C11 Only)	1	(78)	I/O (Open-drain I/O when /WDTOUT)	Port E 7/Watchdog Timer Output (Multiplexed). This pin is configured as a Watch Dog output pin, or as a general purpose input or an output pin. When Watch Dog Timer is enabled, this pin becomes /WDTOUT regardless of the programming as an I/O port, and also becomes an Open-drain output. If /WDTOUT is connected other than a /RESET pin, an external pull-up resistor may be required. On Power-on Reset, this pin is set as PE7 and "Input."

---



---

## SYSTEM CONTROL SIGNALS

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
IEI	1	(1)	Input	Z80 CTC Interrupt enable input signal. This signal is used with the IEO to form an interrupt priority daisy chain when there is more than one interrupt-driven peripheral.
IEO	1	(87)	Output	Z80 CTC interrupt enable output signal. In the daisy chain interrupt control, IEO controls the interrupt of external peripherals. IEO is active when IEI is "1" and the CPU is not servicing an interrupt from the on-chip peripherals.
/RESET	1	(24)	Input (011), I/O (Open Drain) on C11	Reset signal. /RESET signal is used for initializing MPU and other devices in the system. Also used to return from the steady state in the STOP or IDLE modes.

---

**Note:**

For the Z84011 the /RESET is kept in active state for a period of at least three system clock cycles.

**Note:**

For the Z84C11, during the power-up sequence, the /RESET becomes Open-drain output and the Z84C11 will drive this pin to "0" for 25 to 75 msec after the power supply passes through approx. 2.2V and then reverts to input. If it receives the /RESET signal after power-on sequence, it will drive the /RESET pin for 16-processor clock cycles depending on the status of the Reset Output Disable bit in the Watch Dog Timer Master Register. If this Reset output is disabled, it must be kept in an active state for a period of at least three system clock cycles. Note, that if using Z84C11 in the Z84011 socket, modification may be required on the Reset circuit since this pin is a "pure input pin" on the Z84011. The /RESET pin does not have internal pull-up resistors and requires external pull-up. For more details of the function, refer to "Functional Description."

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
XTAL1	1	(49)	Input	Crystal oscillator connecting terminal. A parallel resonant crystal is recommended. If an external clock source is used as an input to the CGC unit, supply clock goes into this terminal.

---

**Note:**

For the Z84C11, a crystal presence is automatically detected by the Z84C11; oscillator and divide-by-two circuits are activated. The single phase clock generated is output on the CLK pin if the external clock is not applied on CLK pin.

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
XTAL2	1	(48)	Output	Crystal oscillator connecting terminal.
CLK	1	(52)	Output (011), I/O(C11)	System Clock

---

**Note:**

For the Z84011, CLK provides Single Phase system clock generated by CGC. For the Z84C11, if the clock is applied on this pin, the internal oscillator and divide-by-two circuits are bypassed. Otherwise, CLK provides System Clock to the system.

---

---

## SYSTEM CONTROL SIGNALS

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
MS1, MS2	2	(47,46)	I	Mode select 1 and 2. The mode select input pins. The status on these pins determine one of four power save modes.(Run, Idle1, Idle2 or STOP).
EV	1	(15)	Input	Evaluator signal. When "1" is applied to this pin, PIC is put in Evaluation mode. For details, refer to "Functional Description" on EV mode.

---

**Note:**

For the Z84011, together with /BUSREQ, the EV signal puts the Z84011 into the evaluation mode. When this signal becomes active, the status of /M1, /HALT and /RFSH change to input. When using Z84011 as an evaluator chip, the CPU is electrically disconnected after one machine cycle is executed with the EV signal "1" and the /BUSREQ signal "0". It follows the instructions from the other CPU (of ICE). Upon receiving /BUSREQ; A15-A0, /MREQ, /IORQ, /RD and /WR are changed to input and D7-D0 changes its direction. /BUSACK is NOT tri-stated so it should be disconnected by an externally connected circuit.

**Note:**

For the Z84C11, to access on-chip resources from the CPU (e.g., ICE CPU), the CPU is electrically disconnected; A15-A0, /MREQ, /IORQ, /RD and /WR are changed to input; D7-D0 changes its direction; /M1, /HALT and /RFSH are put into the high impedance state when the EV pin is set to "1." Also, /BUSACK is tri-stated.

---

Signal Name	# of Pins	Pin #	I/O, 3-State	Description
V <sub>CC</sub>	3	(27, 51, 90)	Power	+5 Volts
V <sub>SS</sub>	4	(28, 50, 77, 100)	GND	0 Volts
TEST	1	(86)	Input	Test pin. This pin should be tied to "0".

---

**Note:**

The following pins have different functions between Z84011 and Z84C11:

Pin Name	Pin #	Function
/RESET	24	Functionality is different.
/WAIT	22	Functionality is different in EV mode.
EV	15	Functionality is different.
PE7	78	(Port E 7) on Z84011; PE7//WDTOUT on Z84C11.
/BUSACK	21	In EV mode, tri-stated on Z84C11; remains active on Z84011.

## FUNCTIONAL DESCRIPTION

As shown in Figure 1(a), the Z84011 has a Z80 CPU, CTC, Clock Generator/Controller and Five 8-bit General Purpose I/Os. In addition to these, the Z84C11 has a Watch Dog Timer, Wait State Generator, and Power-on Reset circuit (Figure 1b).

Functionally, the on-chip Z80 CPU and the Z80 CTC are the same as the discrete devices. Therefore, for detailed description of each individual unit, refer to the Product Specification/Technical Manual of each discrete product.

The following subsections describe each individual functional unit of the PIC.

### Z84C00/01 Logic Unit

The CPU unit provides all the capabilities and pins of the Zilog Z80 CPU. This allows 100% software compatibility with existing Z80 software. Refer to "Z84C01 Z80 CPU with CGC" Product Specification.

### Z84C30 Counter/Timer Logic Unit

This logic unit provides the user with four individual 8-bit Counter/Timer Channels that are compatible with the Z84C30 CTC (Figure 3). The Counter/Timers are programmed by the CPU for a broad range of counting and timing applications. Typical applications include event counting, interrupt and interval counting, and serial baud rate clock generation.

Each of the Counter/Timer Channels, designated Channels 0-3, have an 8-bit prescaler (when used in timer mode) and its own 8-bit counter to provide a wide range of count resolution. Each of the channels have their own Clock/Trigger input to quantify the counting process and an output to indicate zero crossing/timeout conditions. Note that Channel 3 doesn't have its output pin. With only one interrupt vector programmed into the logic unit, each channel can generate a unique interrupt vector in response to the interrupt acknowledge cycle.

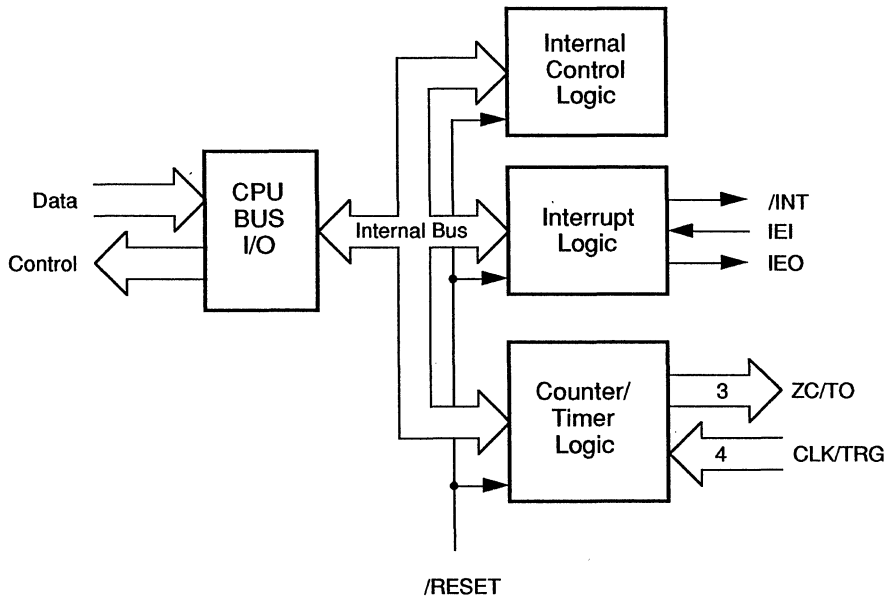


Figure 3. CTC Block Diagram

---

## General Purpose I/O Ports

The PIC has five 8-bit General Purpose I/O ports (a total of 40 I/O lines). Each bit is configured as input or output individually. Figure 4 has the block diagram for General Purpose I/O ports. Each port has 2 associated registers. One is the Port Data Port, which latches the data to the port, and the other is the Data Direction Register, which defines the direction of data flow for the individual bits of its port. While the port bit is assigned as output, the contents of Port Data register can be read back through I/O instructions. For the addresses of these registers, refer to Table 1.

**Note:** For Z84C11, Port 7 bit 7 is multiplexed with Watch Dog Timer Output (WDTOUT). When enabling the Watch Dog Timer, the /WDTOUT is overriding the function as an I/O port. When used as /WDTOUT, a write to Port Data Register has no effect on the PE7/WDTOUT pin, but changes the contents of the Port E data register. A read to this bit returns the status of the /WDTOUT. For more details about Watch Dog Timer, refer to the "Watch Dog Timer" Section.

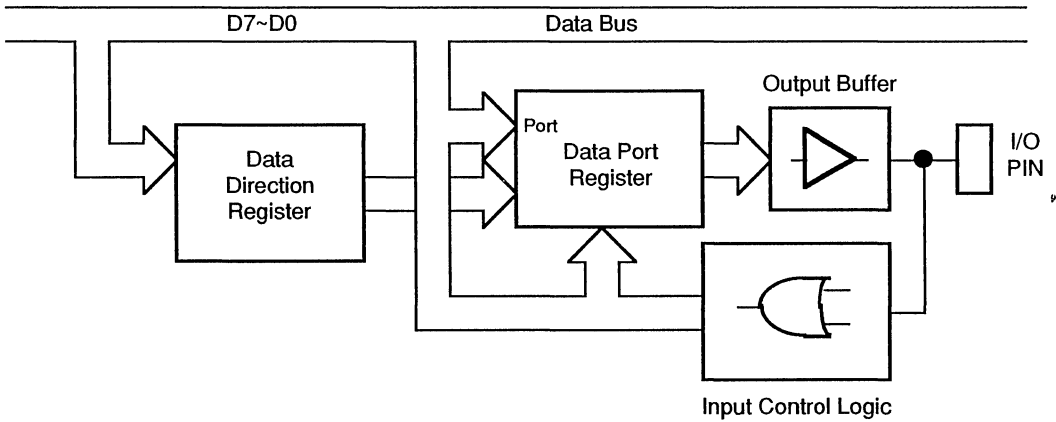


Figure 4. I/O Port Block Diagram

## Watch Dog Timer (WDT) Logic Unit (Z84C11 Only)

This logic unit is being superintegrated into the Z84C11 as an enhanced feature to the Z84011. It detects an operation error, caused by the program runaway, and returns to normal operation. Figure 5, shows the block diagram of the

WDT. While WDT is enabled, the signal PE7//WDTOUT acts as /WDTOUT. During power down mode of operation (either IDLE1/2 or Stop), Watch Dog Timer is halted. Upon Power-on Reset, it is disabled.

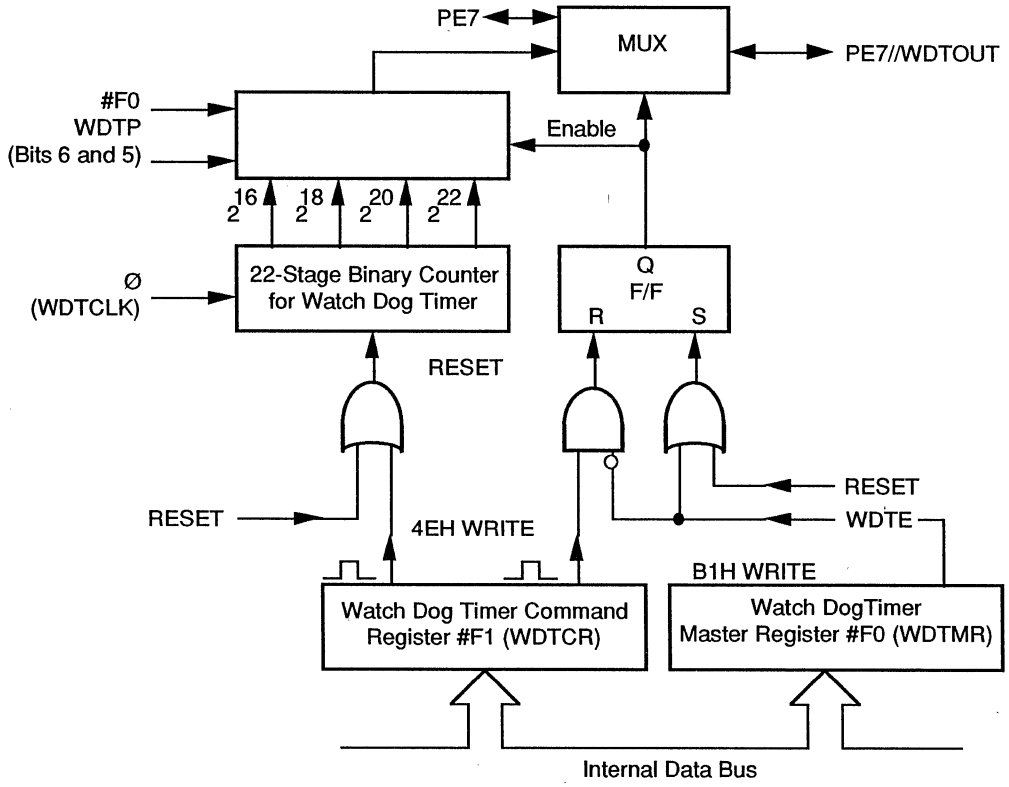


Figure 5. Watch Dog Timer Block Diagram

### WDT Output (PE7//WDTOUT)

Since the Z84C11 doesn't have a dedicated WDT output pin, the WDT output is multiplexed with Port E bit 7. When enabling the Watch Dog Timer, the WDTOUT function overrides PE7 function.

When the WDT is used, the "0" level signal is output from the PE7//WDTOUT pin after a duration of time specified in

the WDTP in the WDTMR. The output pulse width is one of the following, depending on the PE7//WDTOUT pin connection.

The PE7//WDTOUT is connected to the /RESET pin: The "0" level is pulsed for 5T<sub>C</sub> (System Clock cycles).

The PE7/WDTOOUT is connected to a pin other than the /RESET pin: The "0" level is kept until the Watch Dog timer is cleared by software, or reset by /RESET pin.

### CGC Logic Unit

The PIC has a CGC (Clock Generator/Controller) unit. The PIC allows crystal input (XTAL1, XTAL2) or System Clock Input on the XTAL1 pin. It has clock divide-by-two circuits and generates half-speed clock to the input.

**Z84011 Only.** The CGC unit is not supporting "Idle 2" mode of operation.

**Z84C11 Only.** External clock can be also applied from CLK pin. If external clock is provided on the CLK pin, the oscillator and the divide-by-two circuit are bypassed. On Power-on Reset, it comes up in divide-by-two mode. If the external clock or crystal input is provided on the XTAL pins, the internal oscillator is used and the divide-by-two circuit is activated depending upon bit D4 of the WDTMR (See "Programming" section). Power Down modes of the Z84C11 vary based on whether the clock is input on the XTAL1 pin or the CLK pin. If the clock is input on the crystal pin, all of the modes in "halt" state are available. If the system clock is provided from the CLK pin, only the IDLE2 mode is applicable (CTC is kept on running but the internal CPU and Watch Dog Timer are stopped).

### Generating the System Clock

The PIC has a built-in oscillator circuit and the required clock is easily generated by connecting a crystal to the external terminals (XTAL1, XTAL2). Clock output is the half speed of the clock source. Example of an oscillator connection is shown in Figure 6.

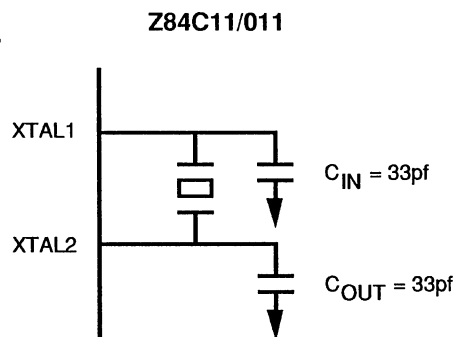


Figure 6. Circuit Configuration for Crystal

**Z84C11.** Clock speed is the same, or half, the frequency of the clock source.

Recommended characteristics of the crystal and the values for the capacitor are as follows, (the values will change with crystal frequency).

- Type of crystal: Fundamental, parallel type crystal (AT cut is recommended).
- Frequency tolerance: Application dependent.
- CL, Load capacitance: Approximately 22pf. (Acceptable range is 20-30pf.)
- Rs, equivalent-series resistance:  $\leq 60$  ohms
- Drive level: 10mW (for  $\leq 10\text{MHz}$  crystal)  
5mW (for  $\geq 10\text{MHz}$  crystal)
- $C_{IN} = C_{OUT} = 33\text{pf}$ .

### Power-on Reset Logic Unit (Z84C11 Only)

The Z84C11 has the enhanced feature of a Power-on Reset circuit. During the power-up sequence, the Open-drain gate of the on-chip Power-on Reset circuit drives /RESET pin to "0" for 25 to 75 msec after the power supply passes through approx. 2.2V. After the termination of the "Power-on Reset" cycle, the Open-drain gate of the on-chip Power-on Reset circuit stops to drive the /RESET pin. It is required to have external pull-up resistor on the /RESET pin.

If it receives /RESET input from outside after the power-on sequence, and while Reset Output Disable bit in Watch Dog Master Register is cleared to "0", it drives the /RESET pin for 16 processor clock cycles from the falling edge of external /RESET input. Otherwise, /RESET pin must be kept in active state for a period of at least 3 system clock cycles.

If there are power-on reset circuits outside of this device, drive this pin with OPEN-DRAIN type gates and pull-up resistors because the /RESET signal is driven low for the period mentioned above during the Power-on sequence. If the external Power-on Reset circuit has push-pull type drivers and they drive the /RESET pin to "1" during that period, it may cause damage. In particular, when using Z84C11 in the Z84011 socket, modification may be required on the external reset circuit.

---

## Wait State Generator Unit (Z84C11 Only)

The Z84C11 has the enhanced feature of a Wait State Generator circuit. It is capable of generating /WAIT signals to the CPU internally. The status of the External /WAIT input line is sampled after the insertion of software wait states, except the wait states insertion for Interrupt Daisy Chain Wait (for this cycle, insertion of a wait state is complex). The Wait State Control Register can be programmed to generate multiple Wait states during different CPU cycles as listed as follows.

### Memory Wait and Op-code Wait

The Wait State Generator can put 0 to 3 wait states in memory accesses. Additionally, one added wait state can be inserted during an /M1 (Op-code fetch) cycle, because /M1 cycles' timing requirement is tighter than memory Read/Write cycles. It generates wait states to the Memory Access in a specified address range, which is programmed in Memory Wait Boundary Register.

### I/O Wait

The Wait State Generator can put 0, 2, 4 or 6 wait states to I/O accesses. Regardless of the programming of this field, no I/O wait states are inserted for accesses to on-chip peripherals.

### Interrupt Vector Wait

During Interrupt acknowledge cycle, the Wait State Generator can insert one wait state after /IORQ goes active, to extend the time between /IORQ fall to vector fetch by CPU. It allows a slow vector response device.

### Interrupt Daisy Chain Wait and RETI Sequence Extension

During Interrupt acknowledge cycle, the Wait State Generator can insert 0, 2, 4 or 6 wait states between /M1 falling to the /IORQ falling edge. This extends the time required to settle the daisy chain. This also allows a longer daisy chain. Further, this field controls the number of wait states inserted during RETI (Return From Interrupt) cycle. If specified to insert 4 or 6 wait states during Interrupt Acknowledge cycle, the Wait State Generator also inserts wait states during the RETI fetch sequence. This sequence is generated with two op-code fetch cycles (Op-code is EDh followed by 4Dh). It inserts 2 or 4 wait states, respectively, if op-code followed by EDh is 4Dh. One wait state if the following op-code is not 4Dh.

### Other Functional Features (Z84C11 Only)

For more system design flexibility, the Z84C11 has the following unique features. These features are controlled by WDTMR (Watch Dog Master Register; Address:FOh) For more details, refer to "Programming section."

- Clock Divide-By-One option
- Reset Output Disable
- Control Register Initialize Option

### Clock Divide-by-One Option

This feature is programmed through Bit D4 of WDTMR. Upon Power-on Reset, the Clock from on-chip CGC is passed through a divide-by-two circuit. By setting this bit to one, the divide-by-two circuit is bypassed so that the system clock is equal to XTAL input. If the clock is applied to the CLK pin from external clock source, the status of this bit is ignored. Upon power-on reset, it is cleared to 0. For details, refer to "Programming" section.

### Reset Output Disable

This feature is programmed by Bit D3 of WDTMR. If this bit is cleared to "0", the /RESET pin is driven to "0" for 16 clock cycles from the falling edge of /RESET input. This feature is for the cases where /RESET is used to get out from the "HALT" state. If this bit has been set to one, the on-chip reset circuit will not drive the /RESET pin except during power-on sequence.

### Control Register Initialize Option

This feature is programmed by Bit D2 of WDTMR. This bit determines whether or not to initialize system control registers to initial value on /RESET. An ideal application for using the Watch Dog Timer.

### Evaluation Mode

The PIC has a built-in evaluation (or development) mode feature which allows the users to utilize standard Z80 development systems conveniently. This mode virtually replaces the on-chip Z80 CPU with the external CPU. In this mode, the on-chip CPU is electrically disconnected from the internal bus and all tri-state signals 15-A0, D7-D0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1, (for C11, /RFSH and /BUSREQ as well) as they are tri-stated, or changed to input. This allows the development system CPU to take over and use the internal I/O registers of the PIC (like the CPU was on-chip).

**Z84011 Only.** When the EV pin is active, the /M1 and /HALT pins are put in the high-impedance state. In using the Z84011 as an evaluator chip, the CPU is electrically disconnected (put in high-impedance state) after one machine cycle is executed with the EV signal being "1" and the /BUSREQ signal being "0". Then, on-chip resources are accessed from outside. /BUSACK and /RFSH are disconnected by an externally connected circuit.

**Z84C11 Only.** If the EV pin is tied to Vcc on power-up, the Z84C11 enters into an evaluation mode. In this mode, the internal CPU is immediately disconnected from the internal bus and all tri-state signals listed above, and /BUSACK and /RFSH signals are tri-stated, or changed to input. Note that the /WAIT pin became the OUTPUT pin in EV mode, and Wait State Generator generates wait states only as programmed. If the target application board has a separate wait state generator, modification of the target may be required.

The Z84C11 acts like regular operation where the /BUSREQ signal is asserted by an external master. This causes all tri-state signals to be tri-stated by the Z84C11 after one clock delay. For this case, /RFSH, /M1, /HALT and /BUSREQ remain active. The /BUSREQ approach was not used for the evaluation mode. This avoided significant external circuitry to work around the time period before the external CPU uses the bus for the Z84C11 accesses.

## PROGRAMMING

### I/O Address Assignment

The PIC 's on-chip peripherals' I/O addresses are listed in Table 1. They are fully decoded from A7-A0 and have no image. The registers with Z84C11 located at I/O Address

EEh, EFh, F0h and F1h control enhanced features to the Z84C11, and are not assigned on Z84011.

**Table 1. I/O Control Register Address**

Address	Device	Channel	Register
10h	CTC	Ch 0	Control Register
11h	CTC	Ch 1	Control Register
12h	CTC	Ch 2	Control Register
13h	CTC	Ch 3	Control Register
50h	PIA	Port A	Port A Data Port (PADP)
54h	PIA	Port A	Port A Data Direction Register (PADR)
51h	PIA	Port B	Port B Data Port (PBDP)
55h	PIA	Port B	Port B Data Direction Register (PBDR)
52h	PIA	Port C	Port C Data Port (PCDP)
56h	PIA	Port C	Port C Data Direction Register (PCDR)
30h	PIA	Port D	Port D Data Port (PDDP)
34h	PIA	Port D	Port D Data Direction Register (PDDR)
40h	PIA	Port E	Port E Data Port (PEDP)
44h	PIA	Port E	Port E Data Direction Register (PEDR)
F0h	WDT		Watch Dog Timer Master Register (WDTMR; Not with Z84011)
F1h	WDT		Watch Dog Timer Control Register (WDTCR; Not with Z84011)
EEh	Misc		System Control Register Pointer (SCRP; Not with Z84011)
EFh	Misc		System Control Data Port (SCDP; Not with Z84011) Through SCRIP and SCDP Control Register 00 - Wait State Control register (WCR) Control Register 01 - Memory Wait state Boundary Register (MWBR)



## CTC Control Registers

For more detailed information, refer to the CTC Technical Manual.

### Channel Control Word

This word sets the operating modes and parameters as described below. Bit D0 is a "1" to indicate that this is a Control Word (Figure 7).

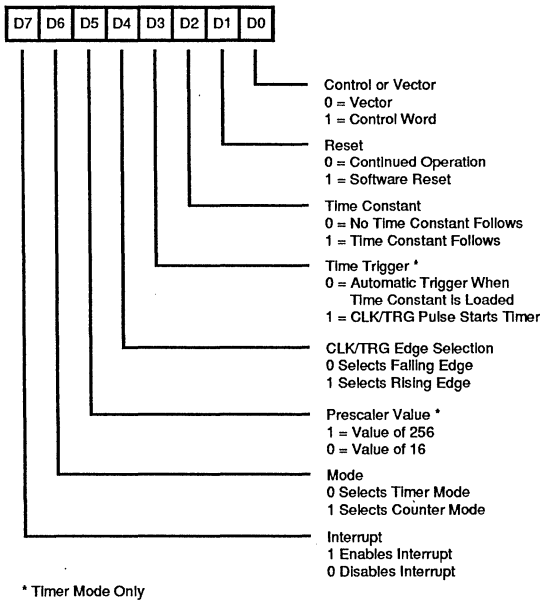


Figure 7. CTC Channel Control Word

**Bit D7. Interrupt Enable.** This bit enables the interrupt logic so that an internal INT can be generated at zero count. Interrupts are programmed in either mode and may be enabled or disabled at any time.

**Bit D6. Mode Bit** This bit selects either Timer Mode or Counter Mode.

**Bit D5. Prescaler Factor.** This bit selects the prescaler factor for use in the timer mode. Either divide-by-16 or divide-by-256 is available.

**Bit D4. Clock/Trigger Edge Selector.** This bit selects the active edge of the CLK/TRG input pulses.

**Bit D3. Timer Trigger.** This bit selects the trigger mode for timer operation. Either automatic or external trigger is selected.

**Bit D2. Time Constant.** This bit indicates that the next word programmed is time constant data for the downcounter.

**Bit D1. Software Reset.** Writing 1 to this bit indicates a software reset operation, which stops counting activities until another time constant word is written.

### Time Constant Word

Before a channel starts counting, it must receive a time constant word. The time constant value is anywhere between 1 and 256, with "0" being accepted as a count of 256 (Figure 8).

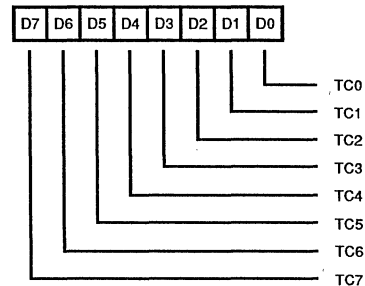


Figure 8. CTC Time Constant Word

### Interrupt Vector Word

If one or more of the CTC channels have interrupt enabled, then the Interrupt Vector Word must be programmed. Only the five most significant bits of this word are programmed, and bit D0 must be "0". Bits D2-D1 are automatically modified by the CTC channels when it responds with an interrupt vector (Figure 9).

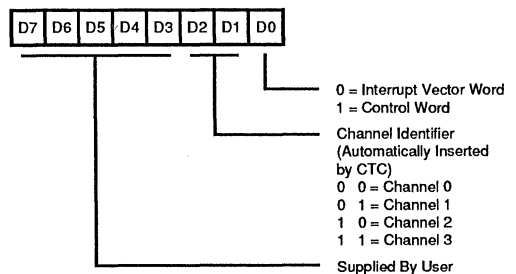


Figure 9. CTC Interrupt Vector Word

---

## PIA Registers

### Port Direction Registers

The PIA ports can be configured for any combination of input and output bits. The direction is controlled by writing to the Port Direction Registers (PADR, PBDR, PCDR, PDDR, PEDR). A "1" written to a bit position indicates that the respective bit is an Output. All bits are inputs on reset. This register is write only (Figure 10).

PDDR, PEDR). A "1" written to a bit position indicates that the respective bit is an Output. All bits are inputs on reset. This register is write only (Figure 10).

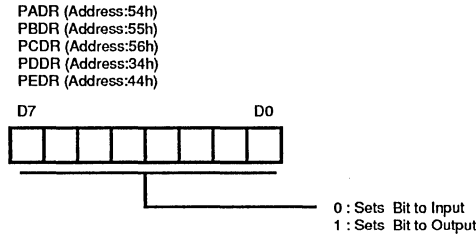


Figure 10. Port Direction Register

---

### Port Data Port

This register holds the data to the port bit assigned as output. It holds the data until modified by the CPU. If the bit is assigned as an output, a read to this register gives the

current value on the port pin, or reads back the contents of this register (Figure 11).

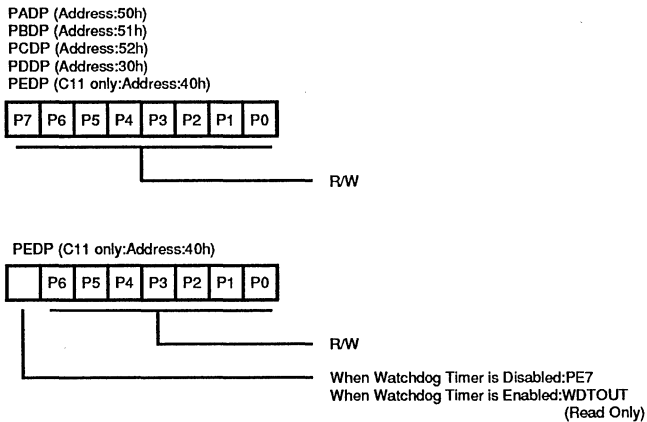


Figure 11. Port Data Port

**Writing a Port Data Port.** If the bit location is assigned as an input, A write to the bit location, assigned as an input, changes the contents of the Port Data Port Register without affecting the port's operation.

If the bit location is assigned as an output. A write to the bit location, assigned as an output, latches the data into the

Port Data Port register, and the content of the register is output on the pin.

**Z84C11 Only.** If Port E bit 7 has been assigned as Watch Dog Timer Output (/WDTOUT), a write to PE7 location will not change the status of the PE7, but changes the bit 7 of PEDR.

**Reading a Port Data Port.** If the bit location is assigned as an input, a read to the bit location, assigned as an input reads the data on the port directly. The contents of the Port Data Port Register are not changed.

If the bit is assigned as an output. A read to the bit location, assigned as an output, reads back the contents of the Port Data Port Register.

**Z84C11 Only.** If Port E bit 7 has been assigned as Watch Dog Timer Output (AWDTOUT), a read to PE7 location returns the status of WDTOUT.

As mentioned above, a write to the bit location assigned as input, will not affect the port's operation. When changing its mode to output from input, write data to be output into port data port before programming Data Direction Register to Output, or there may be a glitch in the port pin.

**Watch Dog Timer Control Registers (Z84C11 Only)**

There are two registers to control Watch Dog Timer operations; Watch Dog Timer Master Register (WDTMR; I/O Address F0h) and the WDT Command Register (WDTCR; I/O Address F1h). Watch Dog Timer Logic has a "double key" structure to prevent the WDT disabling error which could lead to WDT operation stop page due to program runaway. Also, these registers program the power-down mode of operation. The "Second Key" is needed when turning off the Watch Dog Timer.

**Enabling the WDT.** The WDT is enabled by setting the WDT Enable Bit (D7:WDTE) to "1" and the WDT Periodic field (D5,D6:WDTP) to the desired time period. These command bits are in the Watch Dog Timer Master Register (WDTMR; I/O Address F0h).

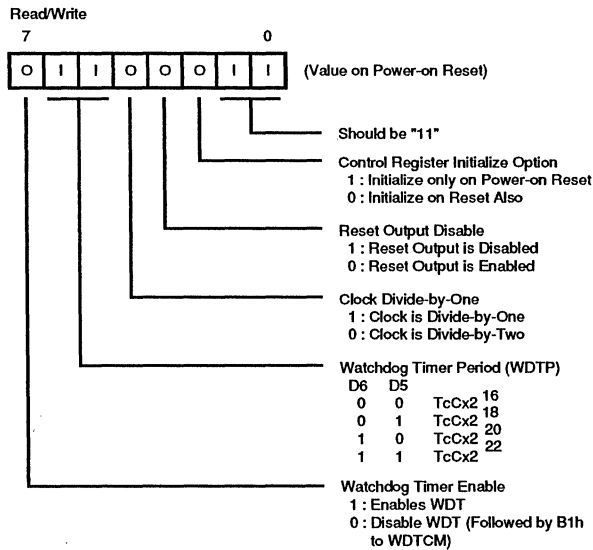
**Disabling the WDT.** The WDT is disabled by clearing WDT Enable bit (WDTE) in the WDTMR to "0" followed by writing "B1h" to the WDT Command Register (WDTCR; I/O Address F1h).

**Clearing the WDT.** The WDT can be cleared by writing "4Eh" into the WDTCR.

**Watch Dog Timer Master Register**

(WDTMR; I/O address F0h)

This register controls the activities of the Watch Dog Timer and system functions (Figure 12).



**Figure 12. Watch Dog Timer Master Register**

**Bit D7. Watch Dog Timer Enable (WDTE).** This bit controls the activities of Watch Dog Timer. The WDT is enabled by setting this bit to "1". To disable WDT, write "0" to this bit followed by writing "B1h" in the WDT Command Register. Watch Dog Timer Logic has a "double key" structure to prevent the WDT disabling error which may lead to WDT operation stop page, due to program runaway. Upon Power-On Reset, this bit is cleared to "0" and the WDT is disabled.

**Bit D6-D5. WDT Periodic field (WDTP).** This two bit field determines the desired time period. Upon Power-on reset, this field is set to "11" and the time period is (TcC x 222).

- 00 - Period is (TcC x 2<sup>16</sup>)
- 01 - Period is (TcC x 2<sup>18</sup>)
- 10 - Period is (TcC x 2<sup>20</sup>)
- 11 - Period is (TcC x 2<sup>22</sup>)

**Bit D4. Clock Divide-by-one option.** "0"-Disable, "1"-enable. On-chip CGC unit has a divide-by-two circuit. By setting this bit to one, this circuit is bypassed and clock on CLK pin is equal to XTAL oscillation frequency (or external clock input on the XTAL1 pin). This bit has no effect when the on-chip CGC unit is not in use and external system clock is fed from CLK pin. Upon Power-on reset, this bit is cleared to "0" and the clock is divided by two.

**Bit D3. Reset Output Disable.** "0"-Reset output is enabled, "1"-Reset output is disabled. This bit controls the /RESET signal and is driven out when /RESET input is used to take the Z84C11 out of the "Halt" state. The reset pulse is driven out for 16-clock cycles from the falling edge of /RESET input, unless this bit is set. Upon Power-on reset, this bit is cleared to "0".

**Bit D2. Control Register Initialize Option.** "0"- Initialize control registers on Reset; "1"- Initialize control registers only on Power-on Reset. D2 determines whether to initialize system control registers to initial values on /RESET. If this bit is cleared to 0, contents of control registers are initialized at /RESET. If this bit is set to 1, contents of control registers are initialized ONLY on Power-on Reset (/RESET will not initialize the control registers). Upon Power-on Reset, this bit is cleared to "0".

**Bit D1-D0. Reserved.** These two bits are reserved and are programmed as "11". A read to these bits returns "11".

### Watch Dog Timer Command Register (WDTCCR; I/O address F1h)

In conjunction with the WDTMR, this register works as a "Second Key" for the Watch Dog Timer. This register is write only (Figure 13).

Write B1h after clearing WDTE to "0" - Disable WDT.  
Write 4Eh - Clear WDT.

WDTCCR (Write Only)

D7	D6	D5	D4	D3	D2	D1	D0	
1	0	1	1	0	0	0	1	(B1h) - Disable WDT (After Clearing WDTE)
0	1	0	0	1	1	1	0	(4Eh) - Clear WDT

Figure 13. Watch Dog Timer Command Register

### Registers for the Wait State Generator

(The following registers are not available on Z84011).

There are two indirectly accessible registers to program wait states; Wait State Control Register (WCR, Control Register 00h) and Memory Wait Boundary Register (MWBR, Control Register 01h). To access these registers, Z84C11 writes "register number to be accessed" to the System Control Register Pointer (SCRCP, I/O address EEh), and then accesses the target register through System Control Data Port (SCDP, I/O address EFh). The pointer which writes into SCRCP is kept until modified.

### System Control Register Pointer (SCRCP, I/O address EEh)

This register stores the pointer to access WCR and MWBR. This register is Read/Write and it holds the pointer value until modified. Upon Power-on reset, all bits are cleared to zero. The pointer value other than 00h and 01h, is reserved and not written. Upon Power-on reset, this register is set to "00h" and points to WCR (Figure 14).

SCRCP (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	0	0	0	0	(Value on Power-on Reset)
0	0	0	0	0	0	0	0	(00h) Point to WCR
0	0	0	0	0	0	0	1	(01h) Point to MWBR

Figure 14. System Control Register Pointer

### System Control Data Port

(SCDP, I/O address EFh)

This register accesses WCR and MWBR (Figure 15).

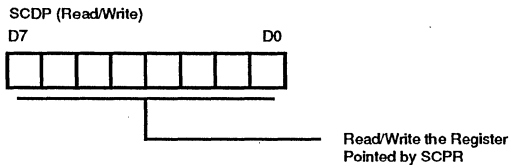


Figure 15. System Control Data Port

### Wait State Control Register

(WCR, register number 00h)

This register accessed through SCDP with the pointer value 00h in SCRP (Figure 16). To maintain compatibility with the Z84011, the Z84C11 inserts the maximum number of wait states (set all bits of this register to one) for sixteen/M1 cycles after Power-on Reset. It automatically clears the contents of this register (move to no-wait state insertion) on the trailing edge of the 16th /M1 signal unless software has programmed a value. If automatic wait state insertion is needed, the wait state is programmed within this time period. A read to WCR during this period will return FFh, unless programmed.

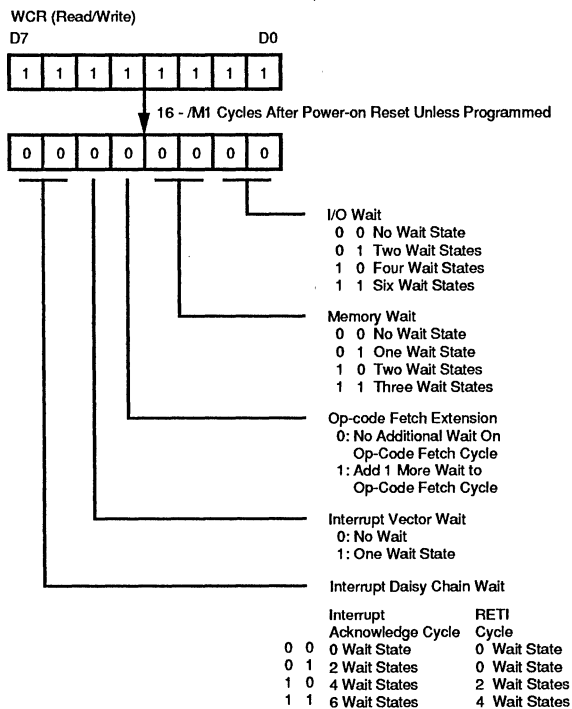


Figure 16. Wait State Control Register

The Wait State Control register has the following fields:

**Bit 7-6. Interrupt Daisy Chain Wait.** This 2-bit field specifies the number of wait states to be inserted during an Interrupt Daisy Chain settle period of the Interrupt Acknowledge cycle. Which means /IORQ goes low after the settling period from /M1 going active. Also, this field controls the number of wait states inserted during the RETI (Return From Interrupt) cycle. If specified to insert four or six wait states during Interrupt Acknowledge cycle, the Wait State Generator also inserts wait states during the RETI fetch sequence. This sequence is formed with two op-code fetch cycles (Op-code is EDh followed by 4Dh). It inserts one wait state if op-code followed by EDh is NOT 4Dh, and inserts two or four wait states, respectively, if the following op-code is 4Dh.

Interrupt Acknowledge	RETI cycle
00 - No Wait states	No wait states
01 - 2 Wait states	No wait states
10 - 4 Wait states	2 Wait states
11 - 6 Wait states	4 Wait states

For sixteen /M1 cycles from Power-on Reset, bits 7-6 are set to "11". They clear to "00" on the trailing edge of the 16th /M1 signal unless programmed.

**Bit 5. Interrupt Vector Wait.** While this bit is set to one, the wait state generator inserts one wait state after the /IORQ signal goes active during the Interrupt Acknowledge cycle. This gives more time for the vector read cycle. While this bit is clear to zero, no wait state is inserted (Standard timing). For sixteen /M1 cycles from Power-on Reset, this bit is set to "1." It then clears to "0" on the trailing edge of the 16th /M1 signal, unless programmed.

**Bit 4. Op-code fetch Extension.** If this bit is set to "1", one additional wait state is inserted during the Op-code fetch cycle in addition to the number of wait states programmed in the Memory Wait field. For sixteen /M1 cycles from Power-on Reset, this bit is set to "1", then clear to "0" on the trailing edge of the 16th /M1 signal, unless programmed.

**Bit 3-2. Memory Wait states.** This 2-bit field specifies the number of wait states inserted during I/O transactions.

- 00 - No Wait states
- 01 - 1 Wait states
- 10 - 2 Wait states
- 11 - 3 Wait states

For sixteen /M1 cycles from Power-on Reset, these bits are set to "11", then cleared to "00" on the trailing edge of the 16th /M1 signal, unless programmed.

**Bit 1-0. I/O Wait states.** This 2-bit field specifies the number of wait states inserted during I/O transactions.

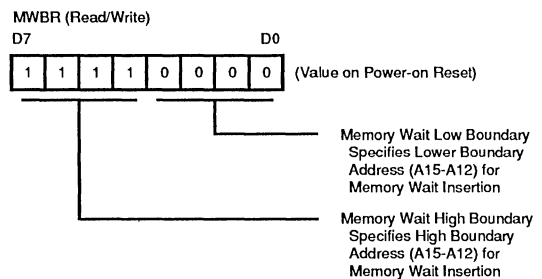
- 00 - No Wait states
- 01 - 2 Wait states
- 10 - 4 Wait states
- 11 - 6 Wait states

For sixteen /M1 cycles from Power-on Reset, these bits are set to "11", then cleared to "00" on the trailing edge of the 16th /M1 signal, unless programmed. For the accesses to the on-chip I/O registers, no Wait states are inserted regardless of the programming of this field.

### Memory Wait Boundary Register

(MWBR, register number 01h)

This register specifies the address range to insert memory wait states. When accessed memory addresses are within this range, the Wait State Generator inserts Memory Wait States specified in the Memory Wait field in WCR (Figure 17).



**Figure 17. Memory Wait Boundary Register**

**Bit D7-D4. Memory Wait High Boundary.** This field specifies A15-A12 of the upper address boundary for Memory wait.

**Bit D3-D0. Memory Wait Low Boundary.** This field specifies A15-A12 of the lower address boundary for Memory wait.

Memory wait states are inserted for the address range:

$$(D7-D4 \text{ of MWBR}) \geq A15-A12 \geq (D3-D0 \text{ of MWBR})$$

This register is set to "F0h" on Power-on Reset, which specifies the address range for Memory wait as "0000h to FFFFh".

---

## OPERATION MODES

There are four kinds of operation modes available for the PIC in connection with clock generation; RUN Mode, IDLE1/2 Modes and STOP Mode.

The Operation mode is effective when the halt instruction executes. Restart of MPU from the stopped state under IDLE1/2 Mode or STOP mode is effected by inputting

either /RESET or interrupt (/NMI or /INT). The mode selection of these power-down modes is made by two external pins (MS1/MS2).

**Note:** Z84011 is not supporting IDLE 2 mode of operation. Do not use the MS1/MS2 combination of 0 1 with Z84011.

---

Operation Mode	MS1	MS2	Description at HALT State
RUN Mode	1	1	The PIC continues the operation. If CLK is an output, it supplies clocks to the outside, continuously.
IDLE1 Mode	0	0	The internal oscillator's operation is continued and supplies clocks to the outside, continuously. Clock output (CLK) (and internal clock to the CTC and the Watch Dog Timer) is stopped at "0" level of T4 state in the halt instruction operation code fetch cycle. This mode is not supported with Z84C11 when external clock is applied to the CLK pin.
IDLE2 Mode (C11 Only)	0	1	The internal oscillator and the CTC's operation continues. If the CLK pin has been selected as output, it supplies clock to the outside continuously. But the internal clock to the CPU and the Watch Dog Timer is stopped at "0" level of the T4 state. This is in the cycle immediately after the halt instruction op-code fetch cycle. This mode is also valid when external clock is applied to the CLK pin.
STOP Mode	1	0	All operations of the internal oscillator, clock (CLK) output, internal clock to the CTC, and the Watch Dog Timer are stopped at the "0" level of the T4 state. This is in the cycle immediately after the halt instruction op-code fetch cycle.

---

**Table 2. Device Status in Halt State**  
(When clock is supplied by on-chip CGC unit)

Mode	CGC	CPU	CTC	WDT	CLK	Note
IDLE1	O	X	X	X	X	
IDLE2	O	X	O	X	O	[1]
STOP	X	X	X	X	X	
RUN	O	O	O	O	O	

O: Operating  
X: Stop

**Note:**  
[1] Not supported on 011.

**011 Only.** All operating modes, except IDLE 2 (Table 2), are valid with Crystal Input (Crystal connected between XTAL1/2 or external Clock input on XTAL1).

**C11 Only.** All the operating modes in Table 3 are valid with crystal input (Crystal connected between XTAL1/2 or external clock input on XTAL1). For the external clock on the CLK pin, only the IDLE2 and RUN modes are applicable.

## TIMING

### Basic Timing

The basic timing is explained here with emphasis placed on the halt function relative to the on-chip Clock Generator. The following items are identical to those for the Z84C00. For details, refer to the data sheet of the Z84C00.

- Operation Code Fetch Cycle
- Memory Read/Write Operation
- Input/Output Operation
- Bus Request/Acknowledge Operation
- Maskable Interrupt Request Operation
- Non-Maskable Interrupt Request Operation
- Reset Operation

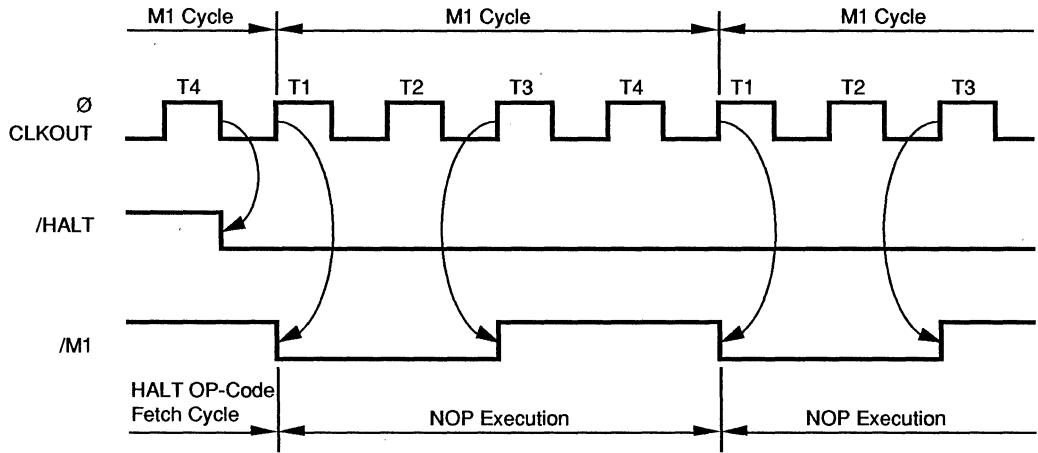
### Operation When HALT Instruction Is Executed

When the CPU fetches a halt instruction in the op-code fetch cycle, /HALT goes active (Low). This is in synchronization with the falling edge of T4 state before the peripheral LSI and CPU stops the operation. After this, the system clock generation differs, depending upon the operation mode (RUN Mode, IDLE1/2 Mode or STOP Mode). If the internal system clock is running, the CPU continues to execute NOP instructions even in the halt state.

#### **RUN Mode (MS1=1, MS2=1)**

Shown in Figure 18 is the basic timing when the halt instruction is executed in RUN Mode.



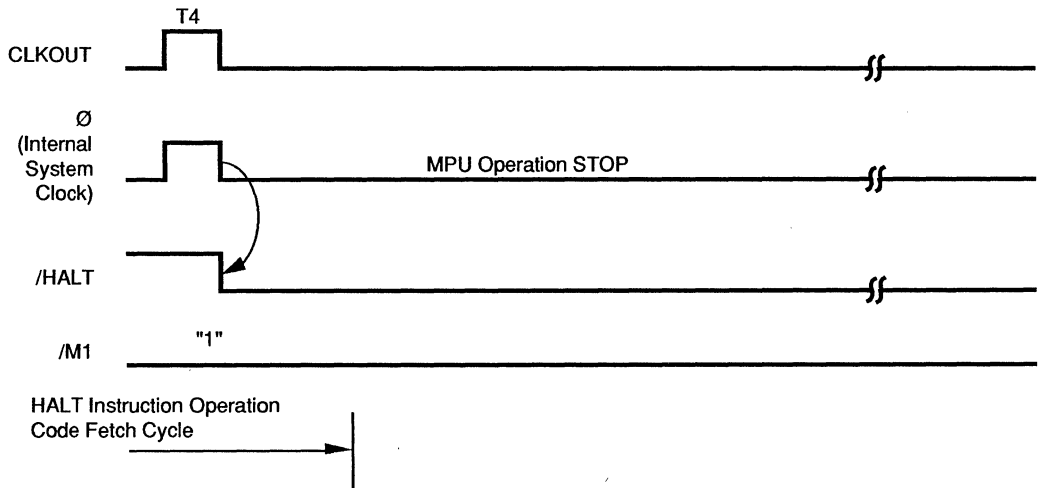


**Figure 18. Timing of RUN Mode**  
(At Halt Instruction Command Execution)

In RUN Mode, internal system clock ( $\emptyset$ ) and clock output (CLK) continues even after the halt instruction is executed. Therefore, until the halt state is released by the interrupt signal (/NMI or /INT) or /RESET signal, MPU continues to execute HALT instruction (internally executing NOP instructions).

**IDLE1 Mode (MS1=0, MS2=0)**

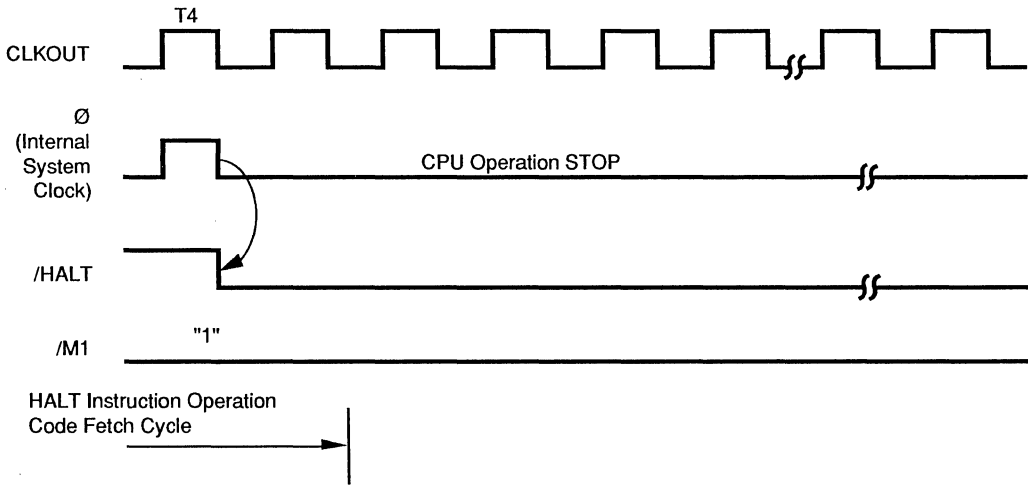
Shown in Figure 19 is the basic timing when the halt instruction is executed in IDLE1 Mode.



**Figure 19. IDLE1 Mode Timing**  
(At Halt Instruction Execution)

In IDLE1 Mode, the internal oscillator continues to operate, but system clock ( $\emptyset$ ) in MPU and clock output (CLK) is stopped at T4 Low state of HALT instruction execution. This mode is not supported when the CGC unit is inactive and the external clock is fed from CLK pin.

**IDLE2 Mode (C11 Only; MS1=0, MS2=1)**  
 Shown in Figure 20 is the basic timing when the halt instruction is executed in IDLE2 Mode. This mode is not supported on O11, and not with C11 when external clock is applied onto the CLK pin.

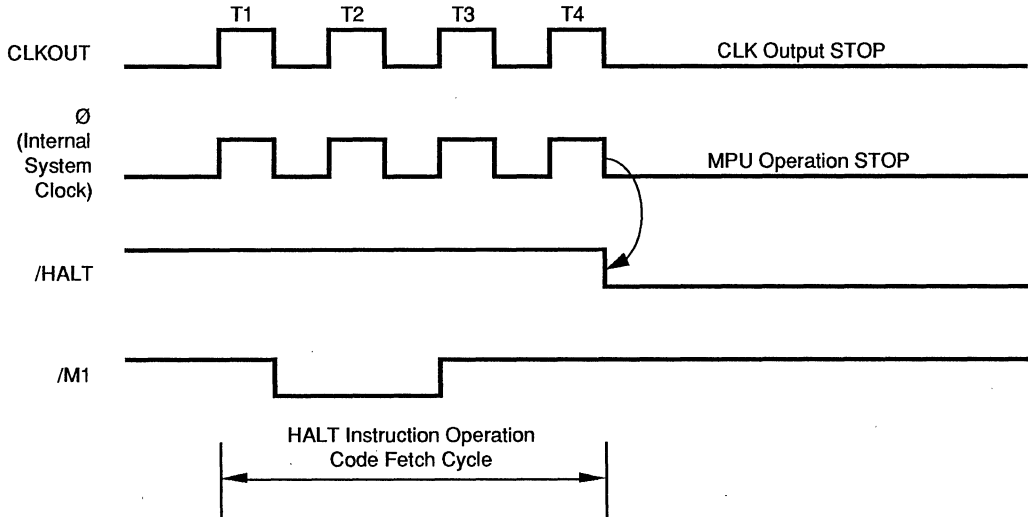


**Figure 20. IDLE2 Mode Timing**  
 (At Halt Instruction Execution)

In IDLE2 Mode, the internal oscillator, clock to CTC and clock output (CLK) to the outside of Z84C11, continues to operate. System Clock ( $\emptyset$ ) in the Z84C11 is stopped at the T4 Low state of HALT instruction execution. Then the CPU and Watch Dog Timer stop their operation.

**STOP Mode (MS1=1, MS2=0)**

Shown in Figure 21 is the basic timing when the halt instruction is executed in STOP Mode.



**Figure 21. STOP Mode Timing**  
(At Halt Instruction Execution)

In STOP Mode, the on-chip CGC unit is stopped at T4 Low state of HALT instruction execution. Therefore, System Clock ( $\emptyset$ ) in the IPC, operation of Watch Dog timer, CPU, CTC and clock output (CLK) to the outside of the IPC are stopped.

### Release From Halt State

The halt state of the CPU is released when "0" is input to the /RESET signal and the MPU is reset or an interrupt request is accepted. An interrupt request signal is sampled at the leading edge of the last clock cycle (T4 state) of NOP instruction. In the case of maskable interrupt, interrupt is

accepted by an active /INT signal ("0" level). Also, the interrupt enable flip-flop is set to "1". The accepted interrupt process is started from the next cycle.

Further, when the internal system clock is stopped (IDLE1/2 Mode, STOP Mode), it is necessary first to restart the internal system clock. The internal system clock is restarted when /RESET or interrupt signal (/NMI or /INT) is asserted.

### RUN Mode (MS1, MS2=1)

The halt release operation by acceptance of interrupt request in RUN Mode is shown in Figure 22.

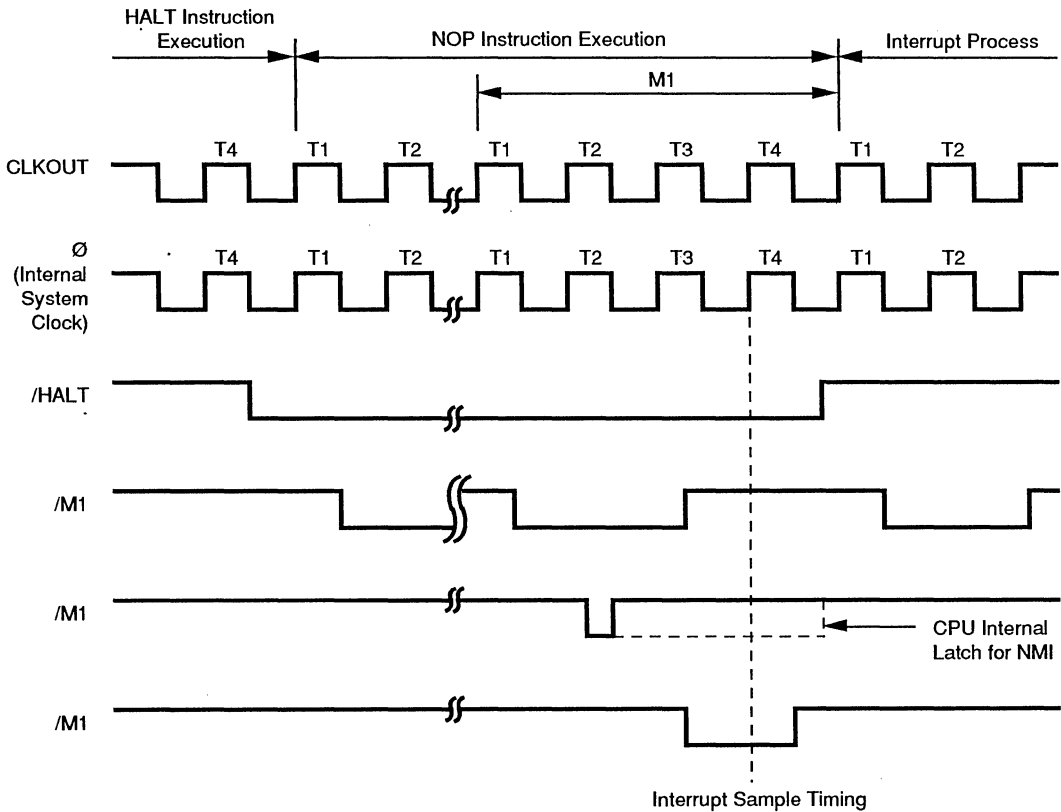


Figure 22. Halt Release Operation Timing By Interrupt Request Signal in RUN Mode

In RUN Mode the internal system clock is not stopped. If the interrupt signal is recognized on the rising clock edge of T4 of the continued NOP instruction, the CPU will execute the interrupt process from the next cycle.

an instruction starting from address 0000H. However, in order to reset CPU, it is necessary to keep /RESET signal at "0" for at least three system clock cycles (For Z84C11: three clock cycles if Reset output is disabled.). In addition, if the /RESET signal becomes "1", after the dummy cycle for at least two T states, the CPU executes an instruction from address 0000H.

The halt release operation by resetting the CPU in RUN Mode is shown in Figure 23. After Reset, the CPU executes

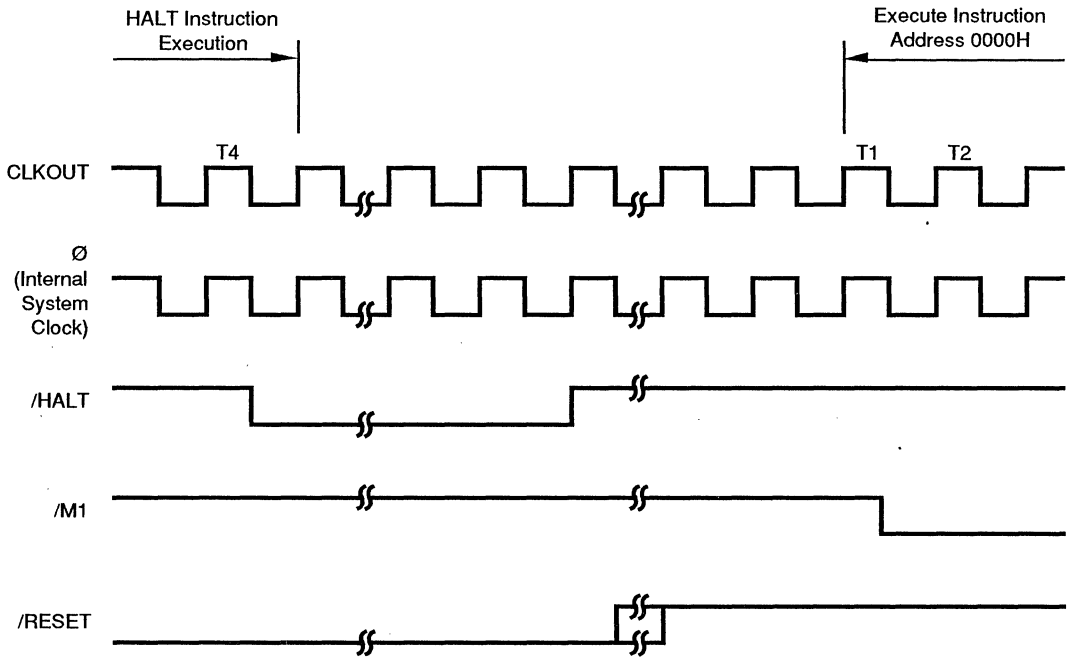
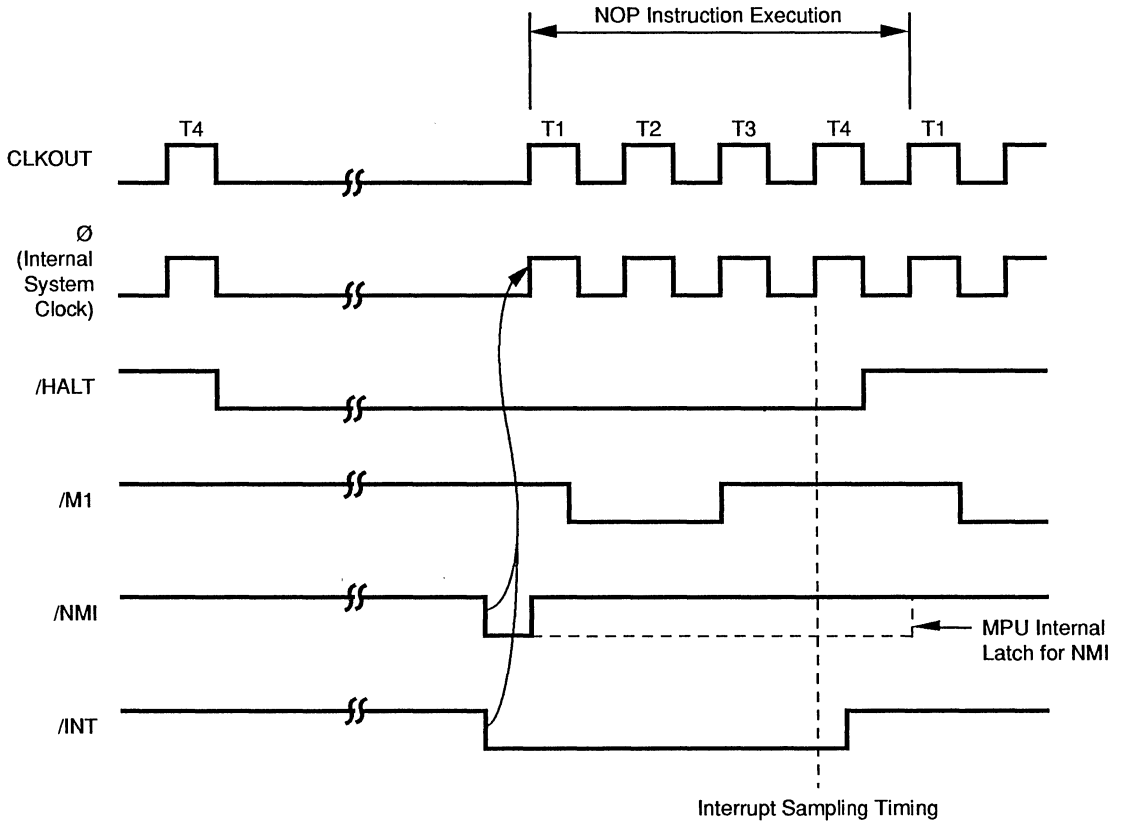


Figure 23. Halt Release Operation Timing By Reset in RUN Mode

IDLE1 Mode (MS1=0, MS2=0)

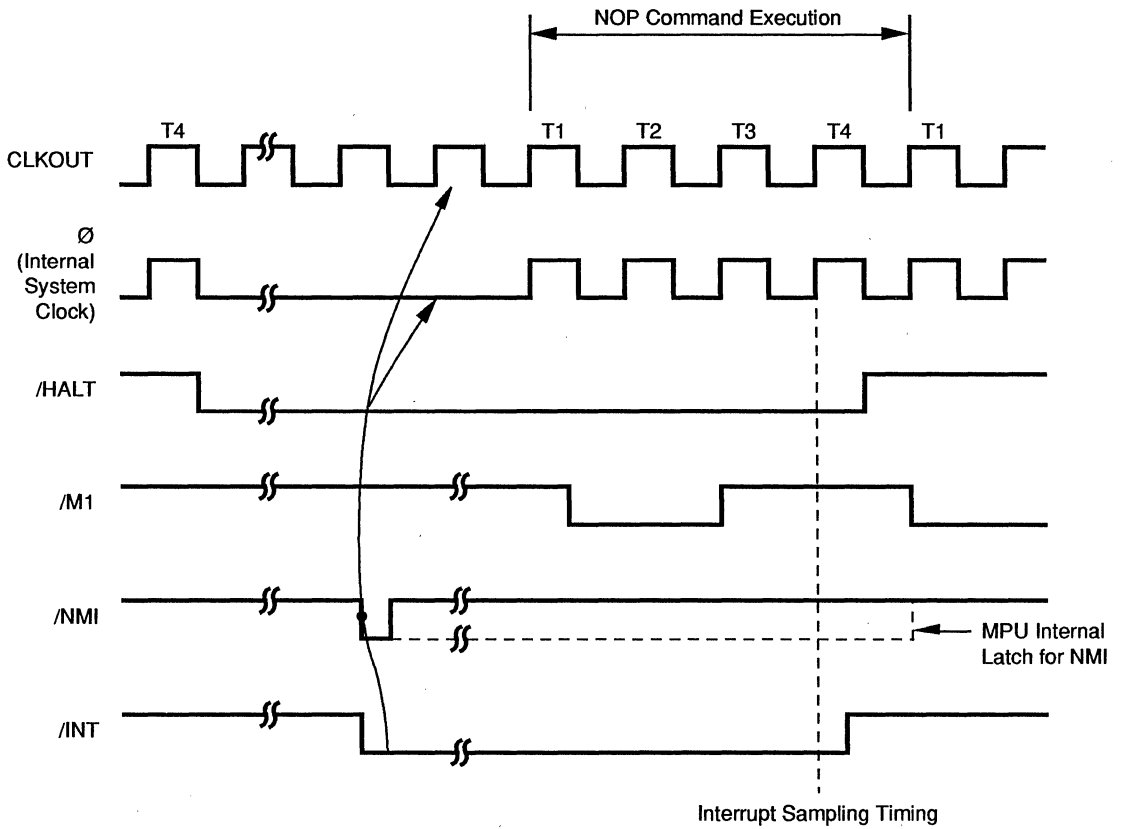
IDLE2 Mode (MS1=0, MS2=1)

The halt release operation by interrupt signal in IDLE1 Mode is shown in Figure 24(a) and in IDLE2 Mode in Figure 24(b).



(a) IDLE1 Mode

Figure 24. Halt Release Operation Timing  
By Interrupt Request Signal in IDLE1/2 Mode



(b) IDLE2 Mode

Figure 24. Halt Release Operation Timing  
By Interrupt Request Signal in IDLE1/2 Mode (Continued)

When receiving /NMI or /INT signal, the stopped internal system clock starts to feed. In IDLE1 Mode, the PIC starts clock output to the outside at the same time.

The operation stop of CPU in IDLE2 mode is taking place at "0" level during T4 state in the halt instruction op-code fetch cycle. Therefore, after being restarted by the interrupt signal, CPU executes one NOP instruction and samples an interrupt signal at the rise of T4 state during the execution of this NOP instruction. It then executes the interrupt process from the next cycle.

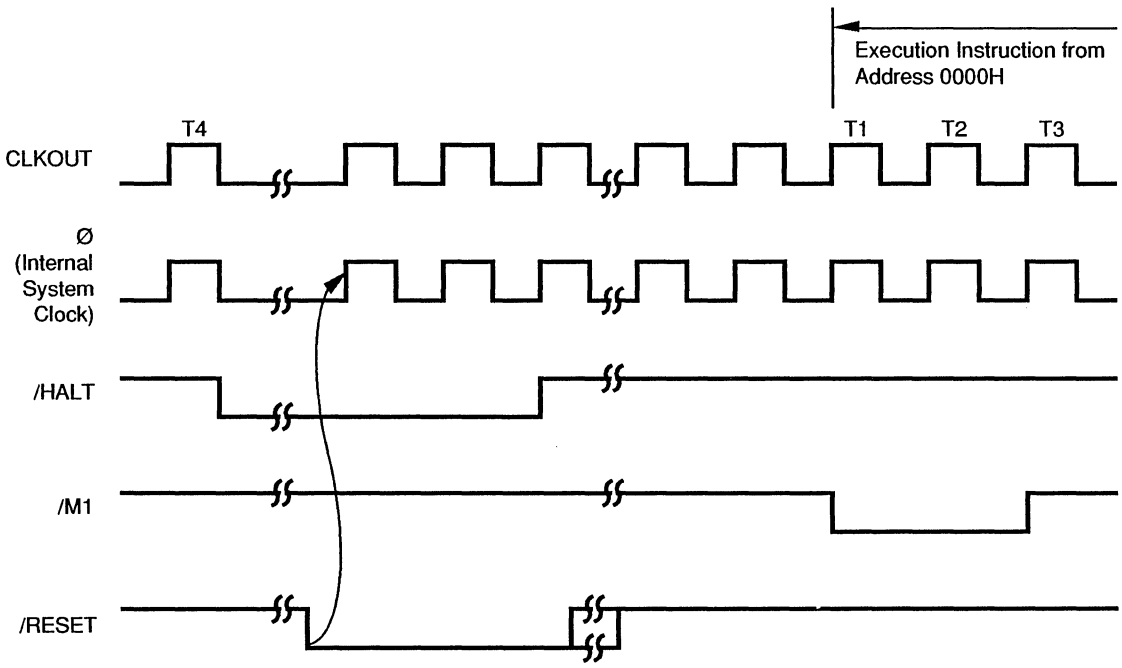
If no interrupt signal is accepted during the execution of the first NOP instruction after the internal system clock is restarted, CPU is not released from the halt state. It is

placed in IDLE1/2 Mode again at "0" level during T4 state of the NOP instruction, stopping the internal system clock. If /INT signal is not at "0" level at the rise of T4 state, no interrupt request is accepted.

**The Halt Release Operation  
By RESET in IDLE1/2 Modes**

When /RESET signal at "0" level is input into the PIC, the internal system clock is restarted and the PIC will execute an instruction stored in address 0000H.

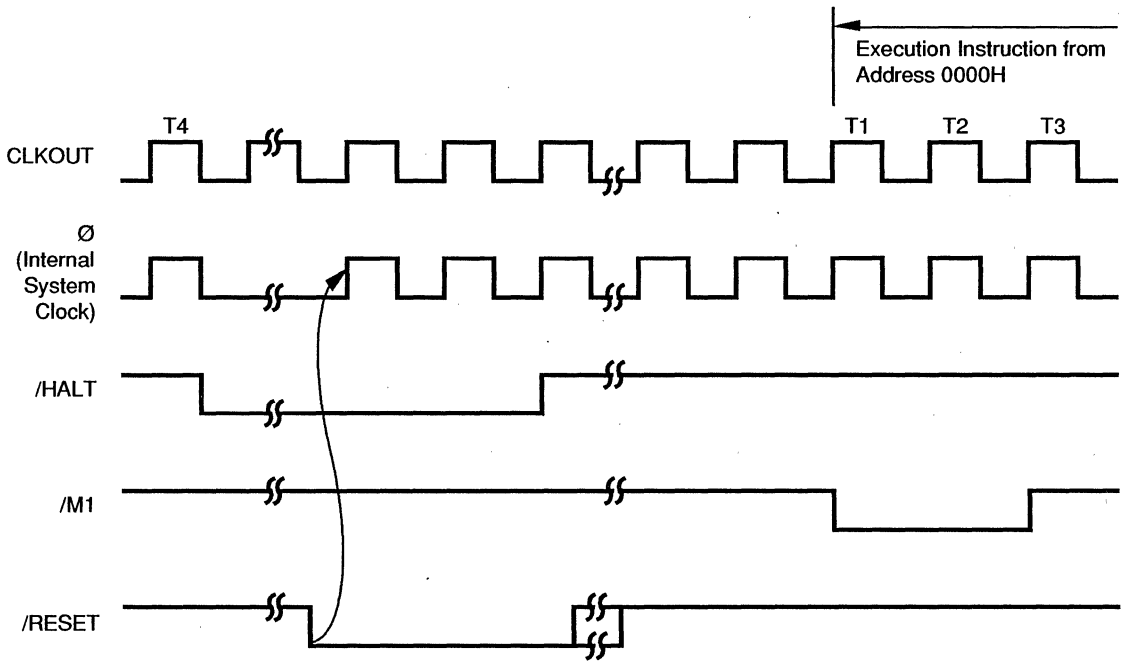
At time of /RESET signal input, it is necessary to take the same care as that in resetting the PIC in RUN Mode (Figures 25a and 25b).



(a) IDLE1 Mode

**Figure 25. Halt Release Operation Timing  
By Reset in IDLE1/2 Mode**



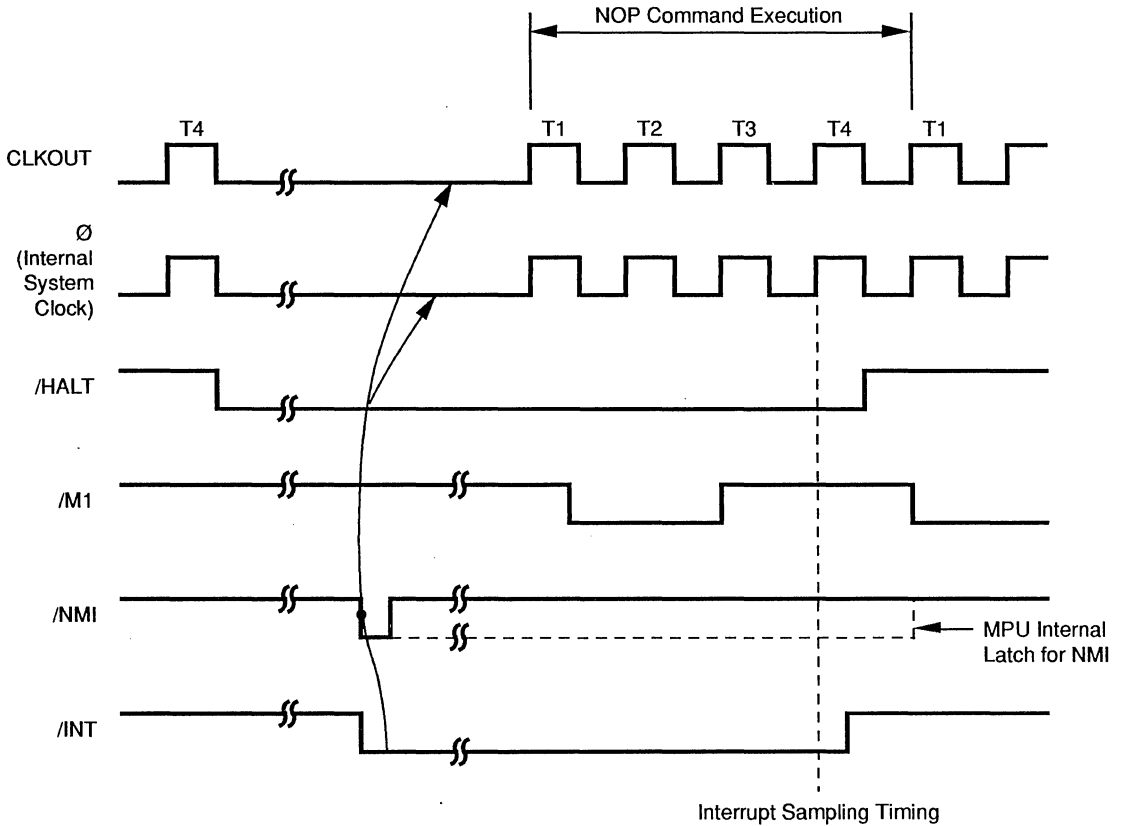


(b) IDLE2 Mode

Figure 25. Halt Release Operation Timing  
By Reset in IDLE1/2 Mode (Continued)

**Halt Release in STOP Mode  
(MS1=1, MS2=0) By Interrupt**

The halt release operation by interrupt signal in STOP Mode is shown in Figure 26.



**Figure 26. Halt Release Operation Timing  
By Interrupt Request Signal in STOP Mode**

When the PIC receives an interrupt signal, the internal oscillator is restarted. To obtain stabilized oscillation, the internal system clock and clock output (CLK) to the outside are started after a start-up time of  $(2^{14}+2.5)$  TcC (TcC: Clock Cycle) by the internal counter.

CPU executes one NOP instruction after the internal system clock is restarted. At the same time, it samples an interrupt signal at the rise of T4 state during the execution of this NOP instruction. If the interrupt signal is accepted, CPU executes the interrupt process operation from the next cycle.

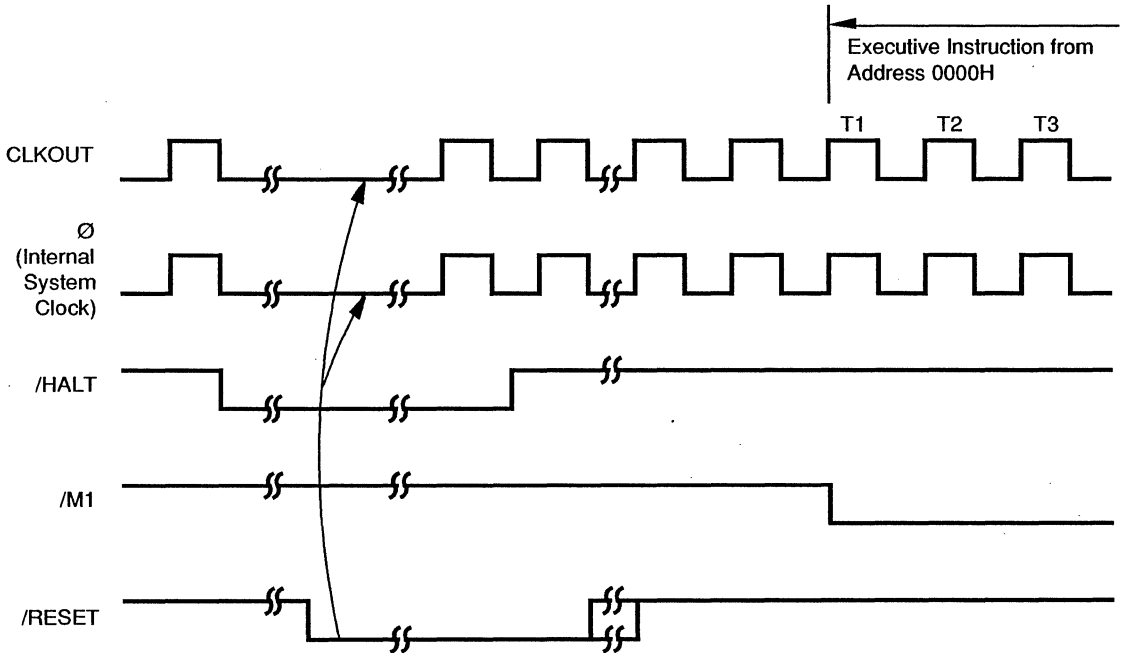
**Note:** During interrupt signal input, care should be taken like the care of the interrupt signal input in IDLE1/2 Mode.

The halt release operation by the Z84C11 resetting in STOP Mode is shown in Figure 27.

**Halt Release in STOP Mode (MS2=0, MS1=1) By /RESET**

When /RESET at "0" level is input into the PIC, the internal oscillator is restarted. However, the internal clock counter for warm-up does not operate. Therefore, the operation is not carried out properly due to unstable clock oscillation. It is necessary to hold the /RESET signal at "0" level for sufficient time. The halt release operation by the PIC resetting in STOP Mode is shown in Figure 27.

**Z84C11 Only.** The /RESET pulse is stretched to a minimum of 16 cycles and driven out of the Z84C11 on the /RESET pin if Reset output is enabled (bit D3 of WDTMR is cleared to "0"). Setting bit D2 disables the driving out of /RESET. If the Control Initialization Option has not been selected (cleared bit D2 of WDTMR), the values programmed in the control registers (WDTMR, SCRIP, WCR and MWBR) are not initialized on /RESET. Otherwise, contents of these registers are initialized to the default value.



**Figure 27. Halt Release Operation Timing By Reset in STOP Mode**

---

**Start-up Time At Time of Restart (STOP Mode).** When the MPU is released from the halt state by accepting an interrupt request, it executes an interrupt service routine. Therefore, when an interrupt request is accepted, it starts the generating clock on the CLK pin (after a start-up time), by the internal counter  $\{(2^{14}+2.5) \text{ TcC (TcC:Clock Cycle)}\}$ . This obtains a stabilized oscillation for operation. Further, in case of restart by the /RESET signal, the internal counter does not operate.

### Evaluation Operation

Each of the CPU signals (15-A0, 7-D0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1) can be tri-stated by activating the EV pin. The Z84C11 enhances the counter part by eliminating the requirement of activating /BUSREQ.

**Instruction set.** The Instruction set of the PIC is the same for the Z84C00. For details, refer to the Data Sheet/Technical Manual of the Z84C00.

---

## AC TIMING

The following section describes the timing of the PIC. The numbers appearing in the figures refer to the parameters on the Table A - F.

M2 or M3). Machine cycles are extended either by the CPU automatically inserting one or more wait states or by the insertion of one or more wait states by the user.

### CPU Timing

The PIC's CPU executes instructions by proceeding through the following specific sequence of operations:

- Memory read or write
- I/O device read or write
- Interrupt Acknowledge

The basic clock period is referred to as a Time or Cycle and three or more T cycles make up a machine cycle (e.g., M1,

### Instruction Op-code Fetch

The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (Figure 28). Approximately one-half clock cycle later, /MREQ goes active. When active, /RD indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the /WAIT input with the falling edge of clock state T2. During clock states T3 and T4 of an M1 cycle, dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction.

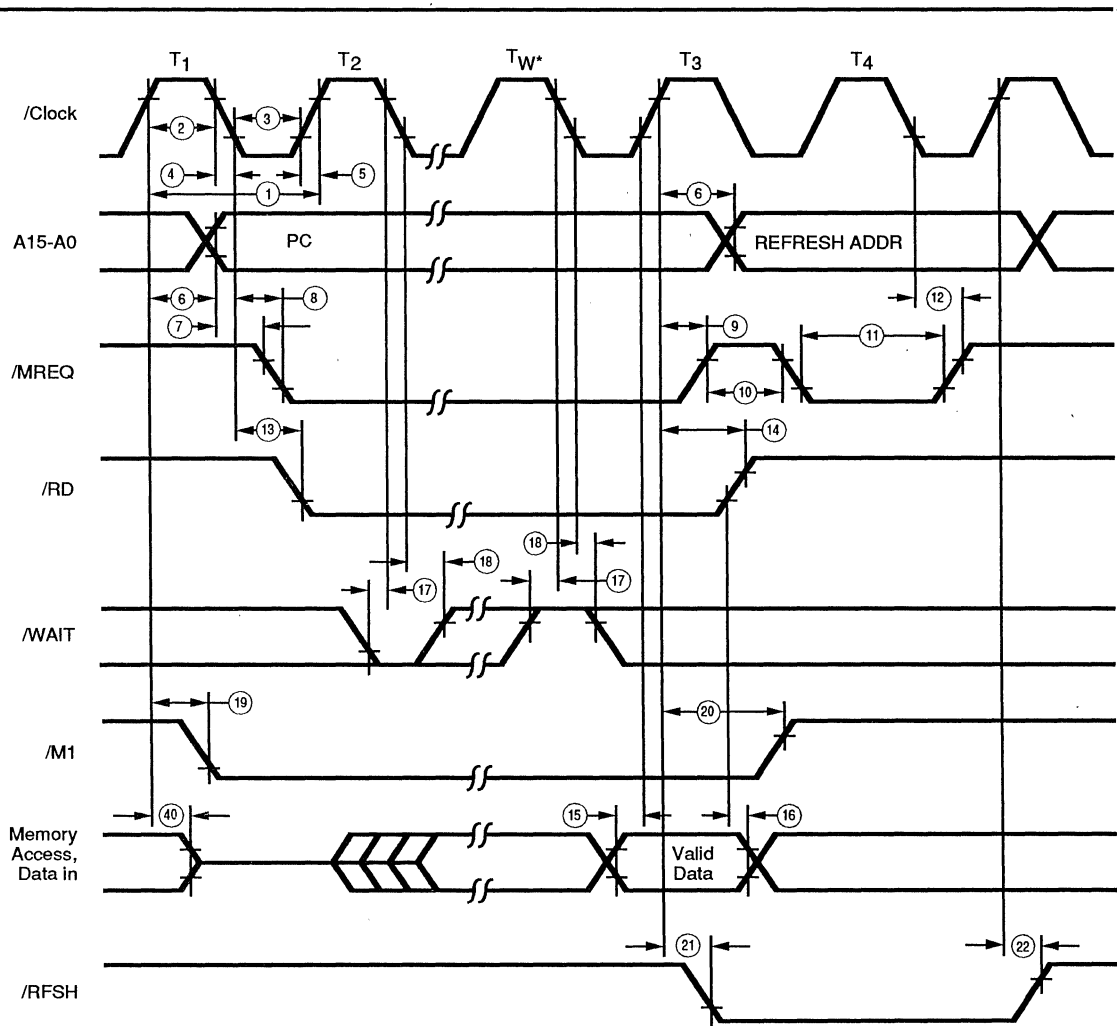


Figure 28. Instruction Op-code Fetch  
(See Table A)

## Memory Read or Write Cycles

Figure 29 shows the timing of memory read or write cycles other than an Op-code fetch (/M1) cycle. The /MREQ and /RD signals function like the Op-code fetch cycle.

In a memory write cycle, /MREQ also becomes active when the Address Bus is stable. The /WR line is active when the Data Bus is stable, so that it is used directly as an R/W pulse to most semiconductor memories.

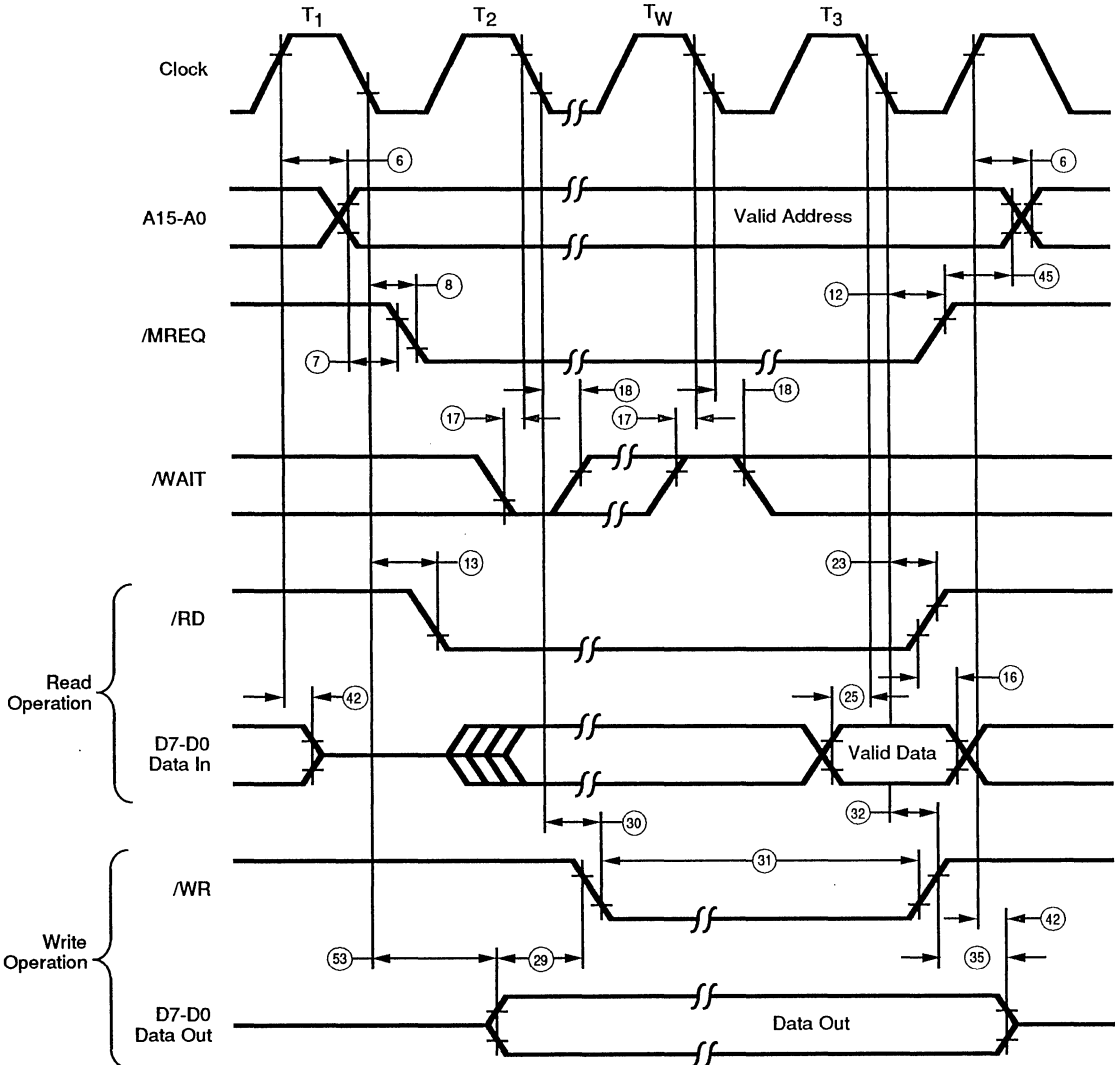
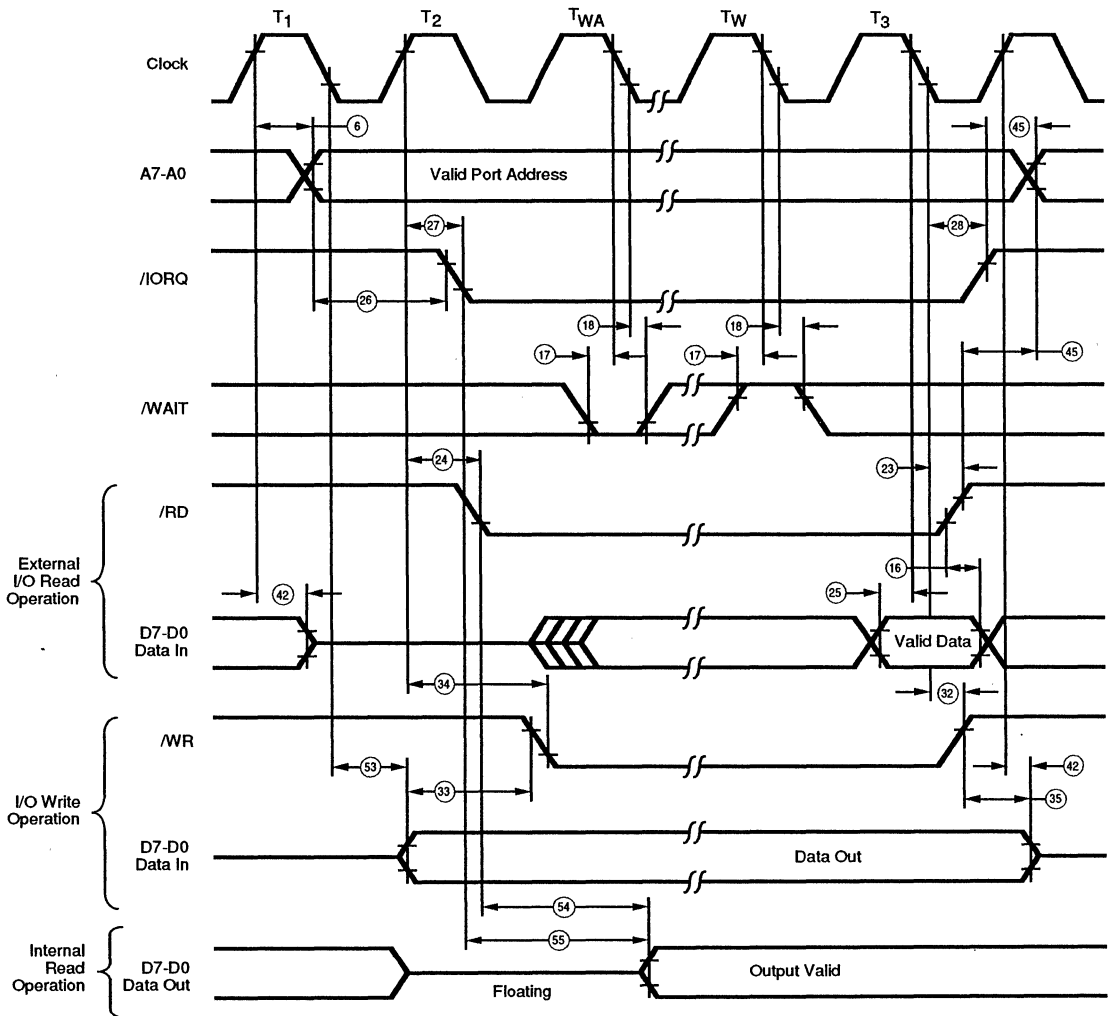


Figure 29. Memory Read or Write Cycle  
(See Table A)

## Input or Output Cycles

Figure 30 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single wait state ( $T_{WA}$ ). This extra wait state allows sufficient time for an I/O port to decode the address from the port address lines.

When the CPU is accessing the on-chip I/O registers (CTC, PIA and system control registers), the data from/to these registers also appears on the data bus, or data bus output during an I/O cycle.



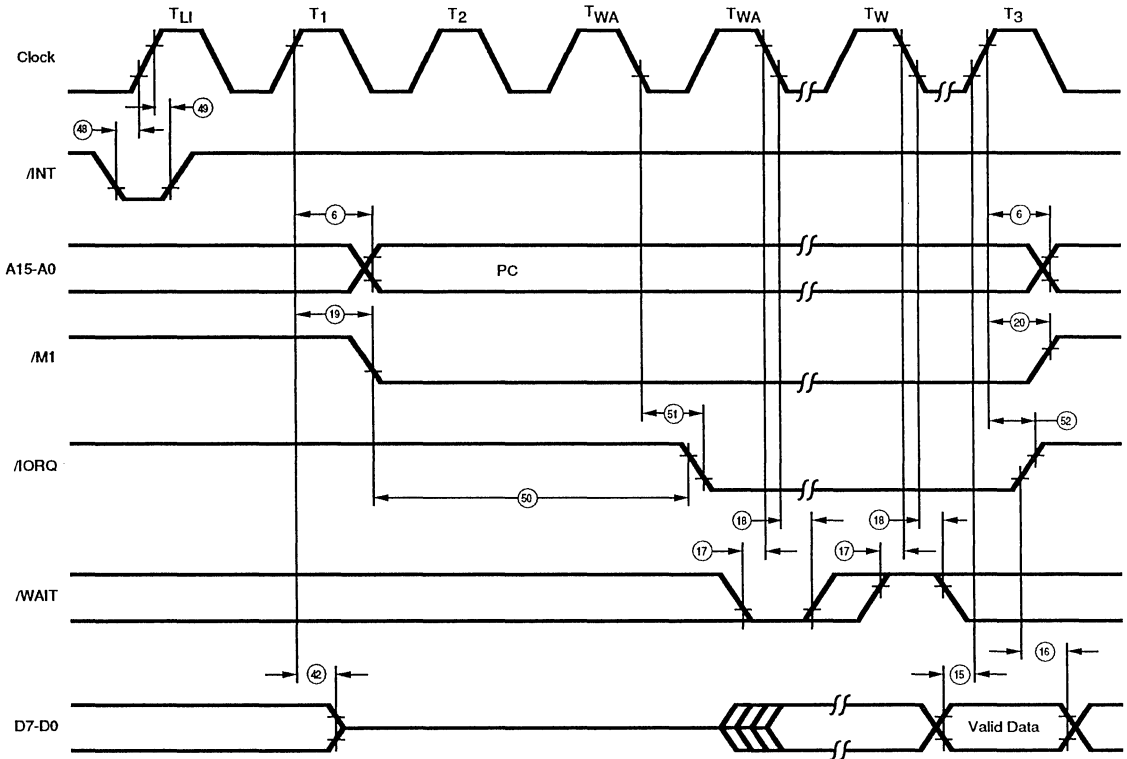
Note:  $T_{WA}$  = One wait cycle automatically inserted by CPU

**Figure 30. Input or Output Cycle**  
(See Table A)

## Interrupt Request/Acknowledge Cycle

The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 31). When an interrupt is accepted, a special /M1 cycle is generated.

During this special /M1 cycle, /IORQ becomes active (instead of /MREQ) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two wait states to this cycle.



NOTE: 1)  $T_{LI}$  = Last state of any instruction cycle

2)  $T_{WA}$  = Wait cycle automatically inserted by CPU

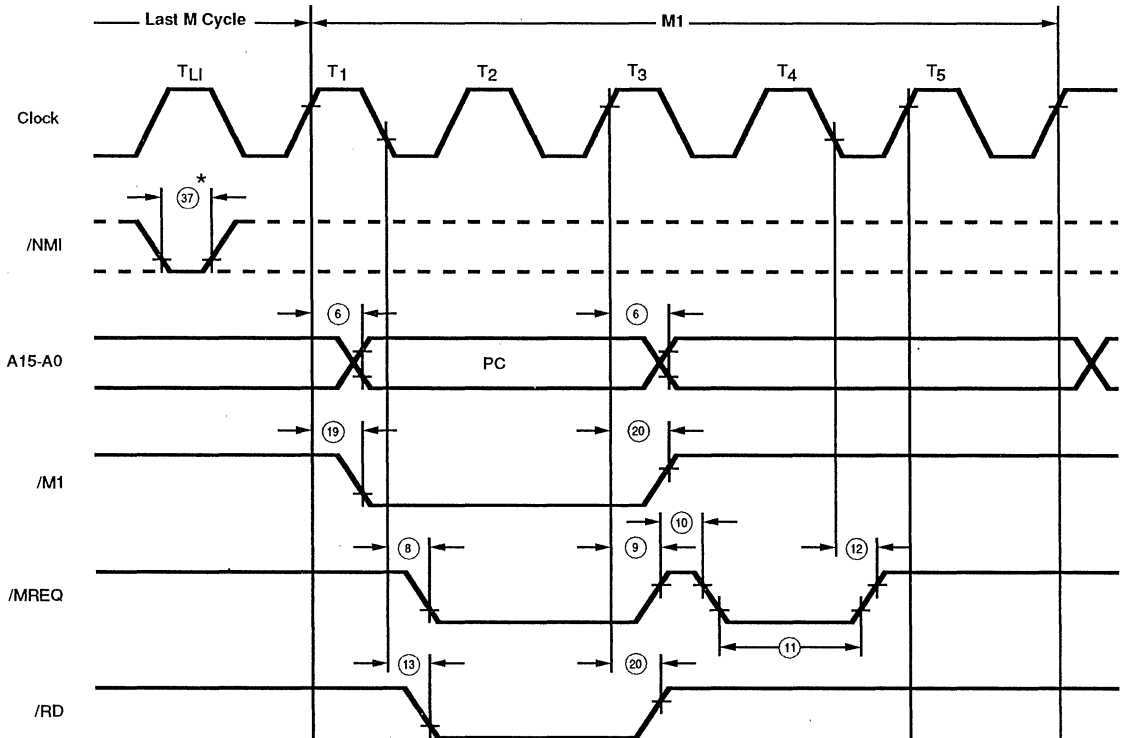
**Figure 31. Interrupt Request/Acknowledge Cycle**  
(See Table A)



## Non-Maskable Interrupt Request Cycle

$\overline{\text{NMI}}$  is sampled at the same time as the maskable interrupt input  $\overline{\text{INT}}$ , but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal memory read operation except that data

put on the bus by the memory is ignored. Instead the CPU executes a restart (RST) operation and jumps to the  $\overline{\text{NMI}}$  service routine located at address 0066H (Figure 32).



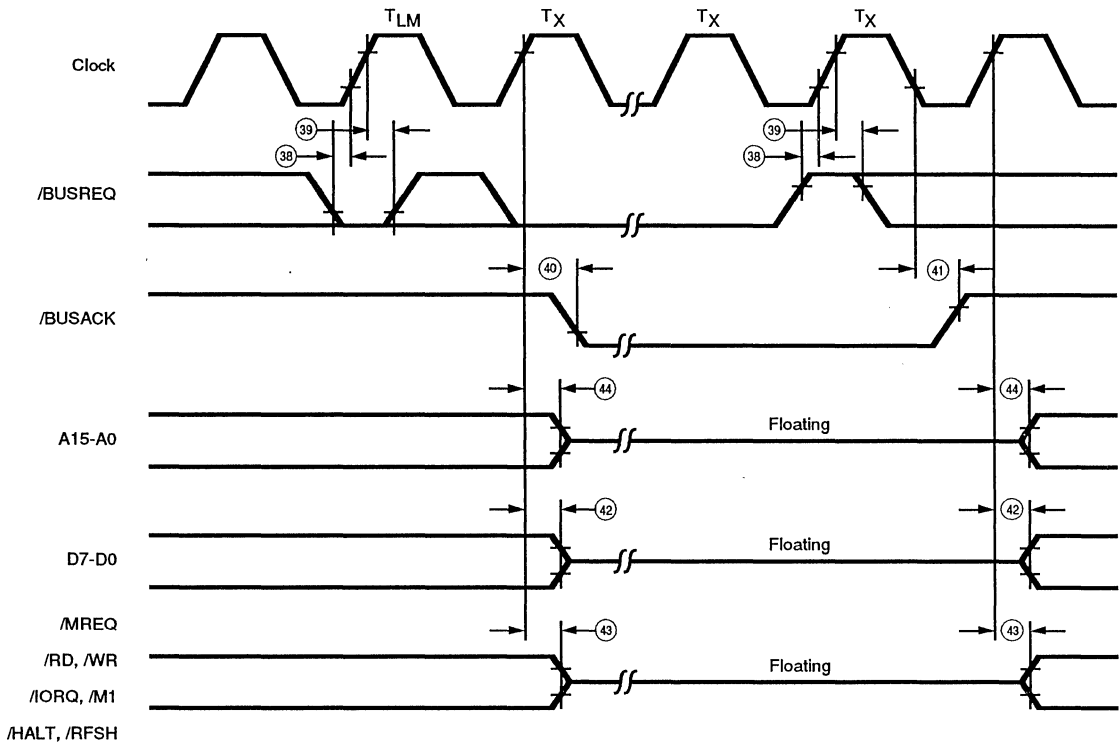
\* Although  $\overline{\text{NMI}}$  is an asynchronous input, to guarantee its being recognized on the following machine cycle,  $\overline{\text{NMI}}$ 's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle (T<sub>LI</sub>).

Figure 32. Non-Maskable Interrupt Request Operation  
(See Table A)

## Bus Request/Acknowledge Cycle

The CPU samples  $\text{/BUSREQ}$  with the rising edge of the last clock period of any machine cycle (Figure 33). If  $\text{/BUSREQ}$  is active, the CPU sets its address, data, and  $\text{/MREQ}$  to a high-impedance state. The  $\text{/IORQ}$ ,  $\text{/RD}$  and  $\text{/WR}$  lines are set to an input for on-chip peripheral access from external

bus master with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.

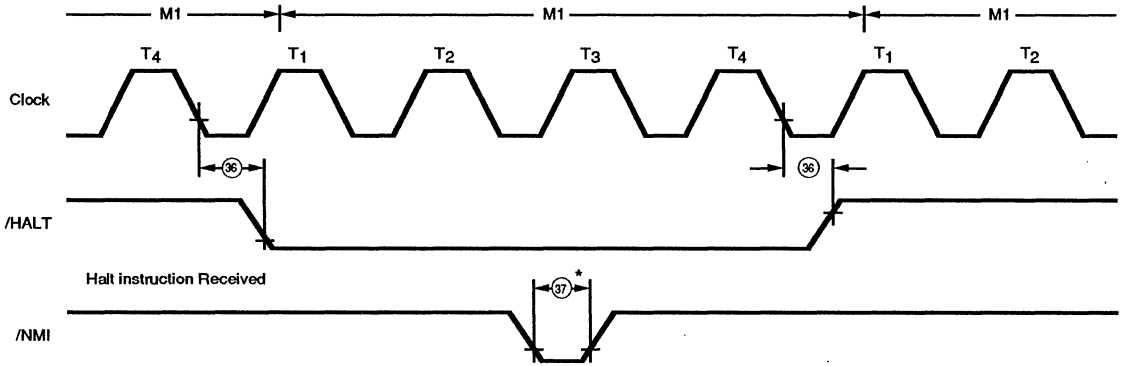


- Notes: 1)  $T_{LM}$  = Last state of any M cycle  
 2)  $T_X$  = An arbitrary clock cycle used by requesting device

**Figure 33. BUS Request/Acknowledge Cycle**  
 (See Table A)

## Halt Acknowledge Cycle

Figure 34 shows the timing for the Halt acknowledge cycle.



\* Although /NMI is an asynchronous input, to guarantee its being recognized on the following machine cycle, /NMI's falling edge must occur no later than the rising edge of the clock preceding the last state of any instruction cycle (T<sub>L1</sub>).

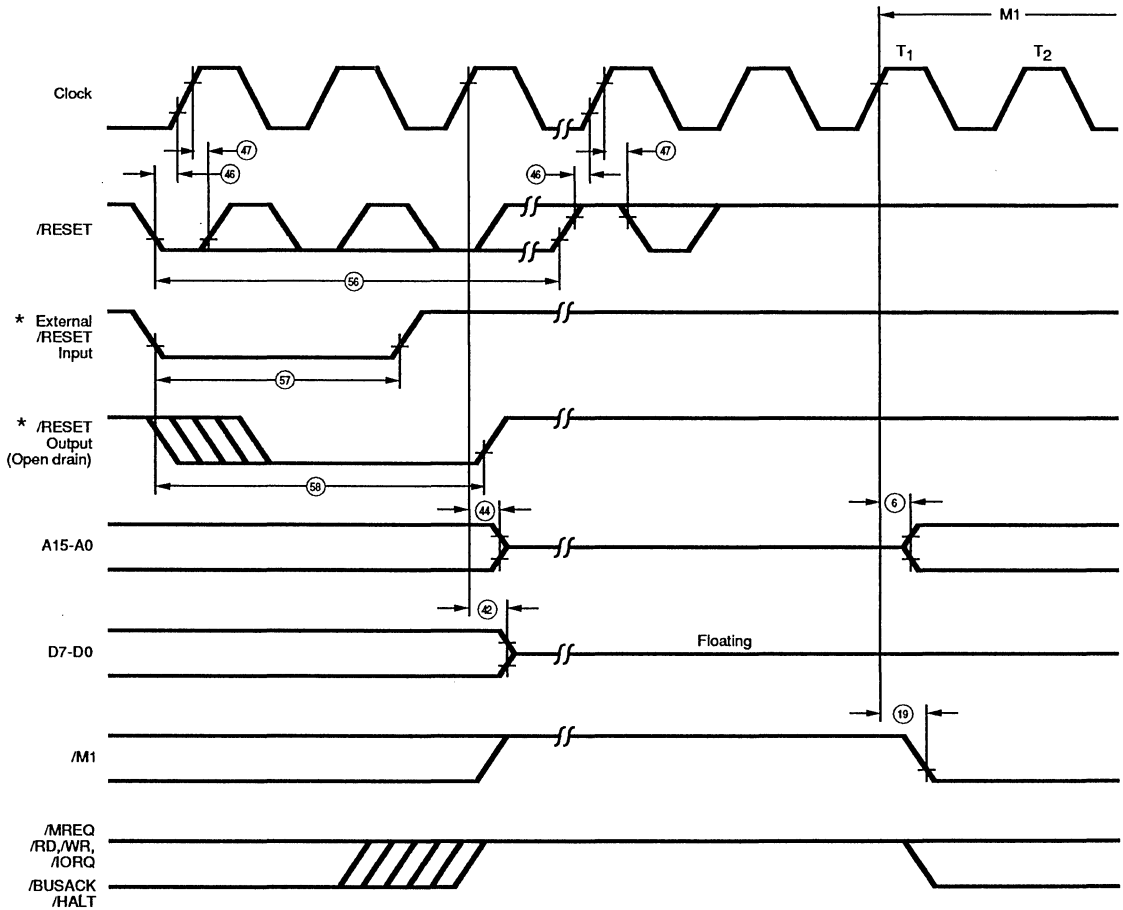
**Figure 34. Halt Acknowledge**  
(See Table A)

## Reset Cycle

$\overline{\text{RESET}}$  must be active for at least three clock cycles for the CPU to properly accept it. As long as  $\overline{\text{RESET}}$  remains active, the address and data buses float, and the control outputs are inactive.

Once  $\overline{\text{RESET}}$  goes inactive, two internal T cycles are consumed before the CPU resumes normal processing operation.  $\overline{\text{RESET}}$  clears the PC register, so the first op-code fetch location is 0000H (Figure 35).

**Z84C11 Only.** If Reset output is disabled,  $\overline{\text{RESET}}$  must be active for at least three clock cycles for the CPU to properly accept it. Otherwise,  $\overline{\text{RESET}}$  must be active for at least two clock cycles and the on-chip reset circuit extends  $\overline{\text{RESET}}$  signal to at least a minimum of 16 clock cycles.



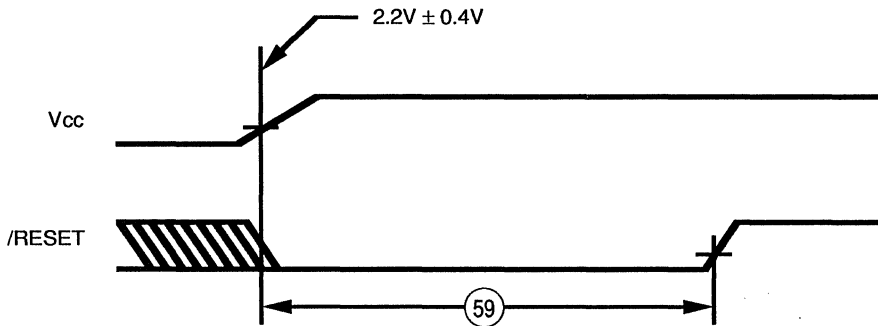
\* 84C11 Only Reset Output is Enabled

**Figure 35. Reset Cycle**  
(See Table A)

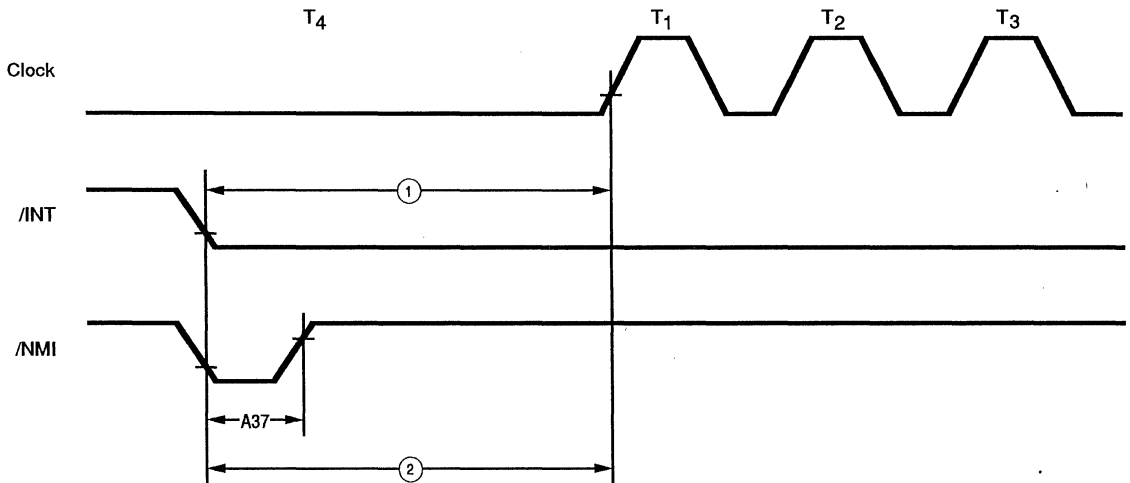
## CGC Timing

Figures 36-39 show the timing related CGC and Power-on Reset circuits. Idle 2 mode of operation is not supported on the Z84011.

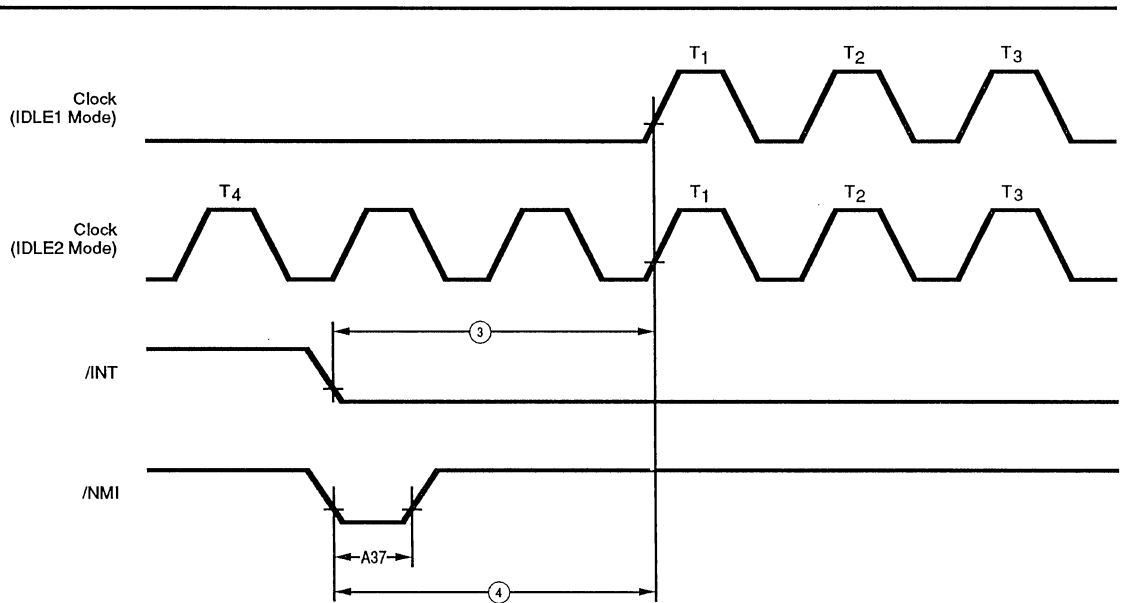
Parameters referenced in Figures 36-39 appear in Table B.



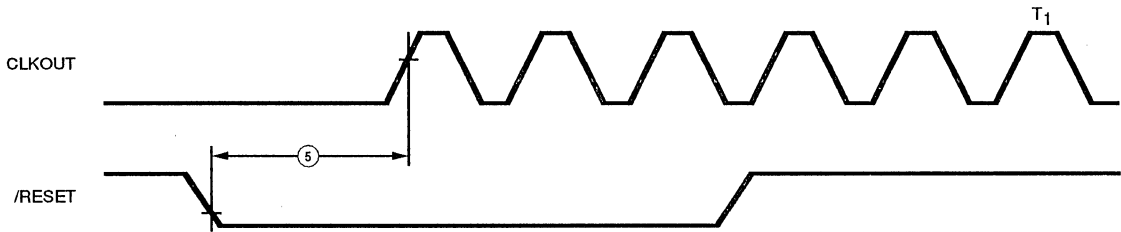
**Figure 36. Reset On Power-up**  
(Applies Only for Z84C11)



**Figure 37. Clock Restart Timing (STOP Mode)**  
(See Table B)

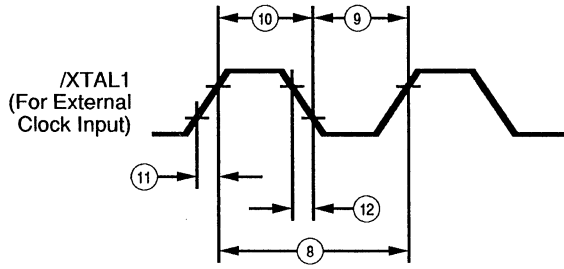


(a) Clock Restart Timing By /INT, /NMI (IDLE1/2 Mode)

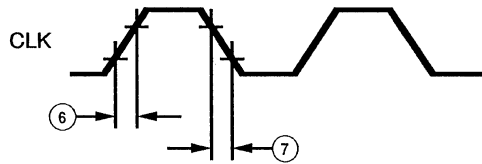


(b) Clock Restart Timing By /RESET (IDLE1/2 Mode)

Figure 38. Clock Restart Timing (IDLE1/2 Mode)  
(See Table B)



(a) XTAL1 Timing for External Clock Input

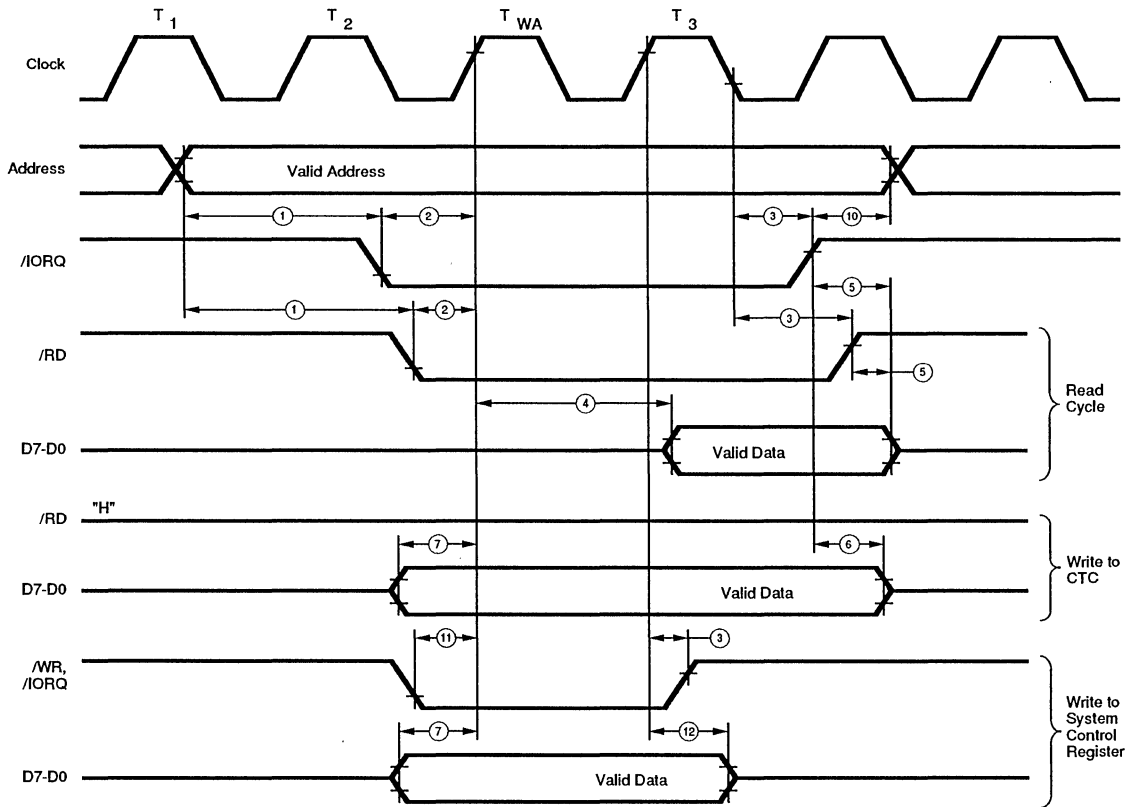


(b) CLK Timing (011 and C11 CLK Pin as Output)

**Figure 39. Clock Timing**  
(See Table B)

**On-chip Peripheral Access  
From External Bus Master**

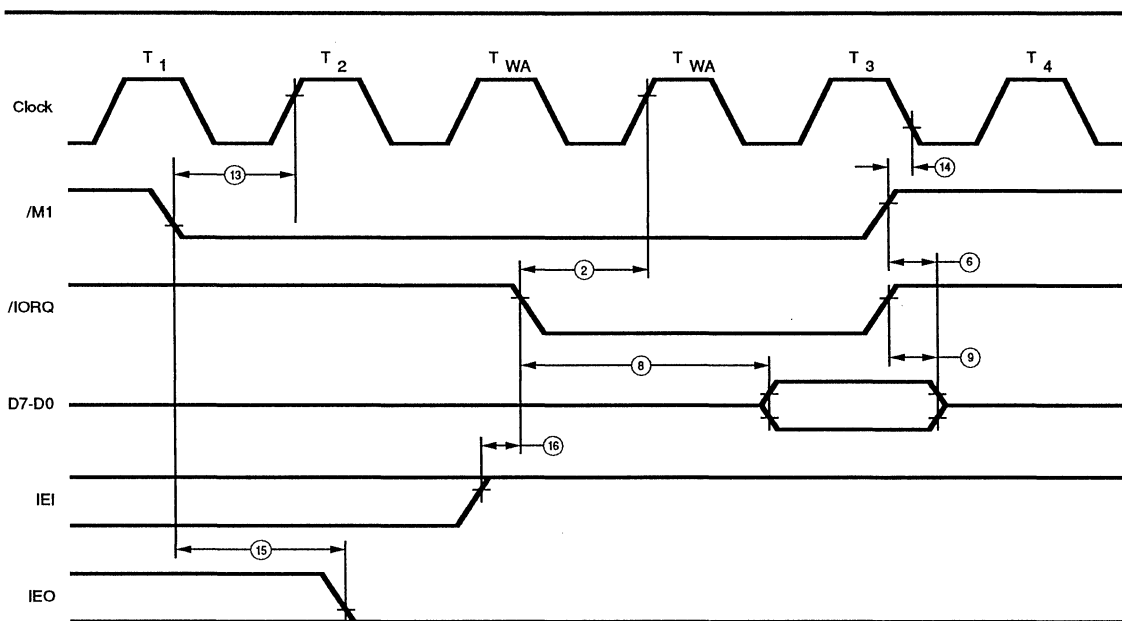
The timing for the on-chip I/O device access from the external bus master is shown in Figure 40. This timing also applies to the timing during EV mode of operation.



**(a) On-chip Peripheral I/O Access From External Bus Master (See Table C)**

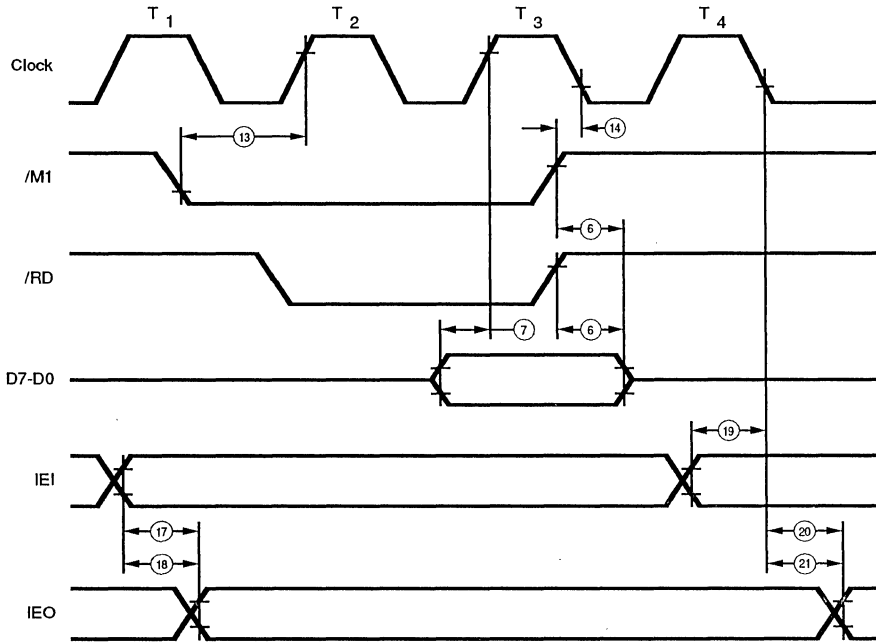
**Figure 40. On-chip Peripheral Timing From External Bus Master**





(b) Interrupt Acknowledge Cycle Timing  
 For On-chip peripheral From External Bus Master  
 (See Table C)

Figure 40. On-chip Peripheral Timing From External Bus Master (Continued)

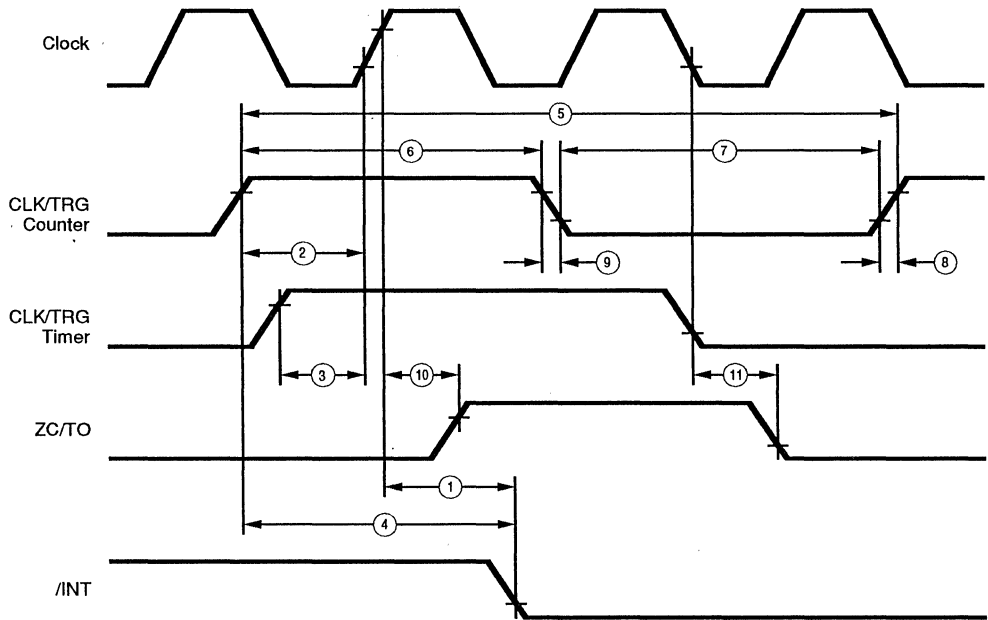


(c) Op-code Fetch Cycle Timing for On-chip Peripheral From External Bus Master (See Table C)

Figure 40. On-chip Peripheral Timing From External Bus Master (Continued)

## CTC Timing

Figure 41 shows the timing for on-chip CTC.

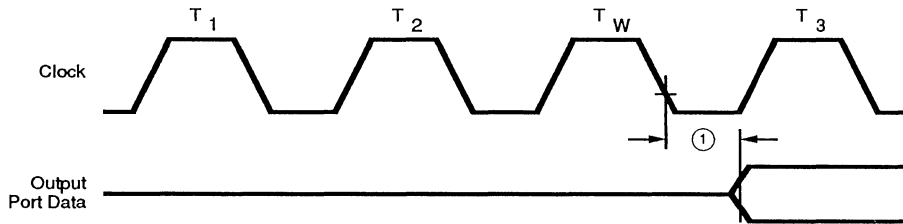


**Figure 41. Counter/Timer Timing**  
(See Table D)

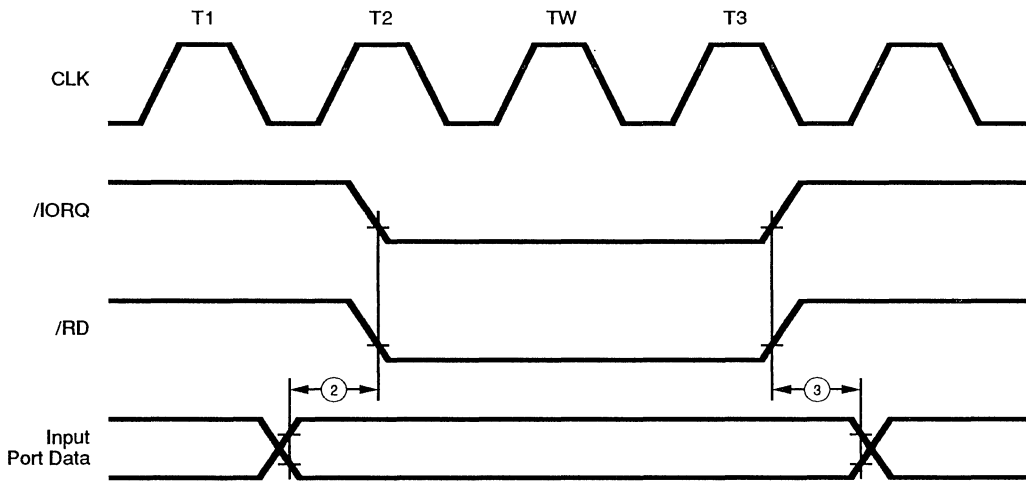
---

## General I/O Port Timing

Figure 42(a) has the Output timing for General I/O port timing while Figure 42(b) has the Input timing.



(a) I/O Port Output Timing  
(See Table E)



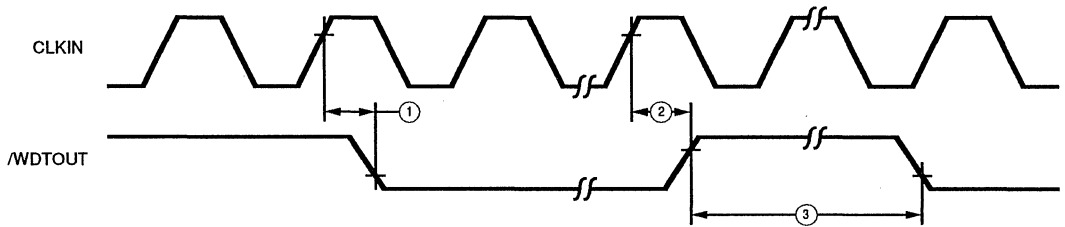
(b) I/O Port Input Timing  
(See Table E)

Figure 42. I/O Port Timing

---

## Watch Dog Timer Timing (Z84C11 Only)

Figure 43 shows the timing for Watch Dog Timer.



**Figure 43. Watch Dog Timer Timing**  
(See Table F)

---

### Precautions

(1) To release the HALT state by /RESET signal in STOP Mode, hold the /RESET signal at "0" until the output from the internal oscillator stabilizes.

**Z84011 Only.** To reset MPU, it is necessary to hold /RESET signal input at "0" level for at least three clocks.

**Z84C11 Only.** If Reset output is disabled, /RESET must be active for at least three clock cycles for the CPU to properly accept it. Otherwise, the on-chip Reset circuit extends /RESET signal to at least a minimum of 16 clock cycles.

(2) Releasing the MPU from the HALT state by an interrupt signal in IDLE1/2 Mode and STOP Mode, does not release the MPU from the HALT state. The internal system clock will stop again unless an interrupt signal is accepted during the execution of a NOP instruction (even when the internal system clock is restarted by the interrupt signal input). Be careful when using /INT.

Other precautions are identical to those for the Z84C00. Refer to the data sheet for the Z84C00.

## ELECTRICAL CHARACTERISTICS

### Absolute Maximum Ratings

Voltage on Vcc with respect to Vss .....-0.3V to +7.0V  
 Voltages on all inputs  
 with respect to Vss .....-0.3V to Vcc +0.3V

### Operating Ambient

Temperature ..... See Ordering Information  
 Storage Temperature .....-65 °C to + 150 °C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

Available operating temperature range is:

E = -40°C to 100°C  
 Voltage Supply Range:  
 $+4.50V \leq V_{cc} \leq +5.50V$

All AC parameters assume a load capacitance of 100 pf. Add 10 ns delay for each 50 pf increase in load up to a maximum of 150 pf for the data bus and 100 pf for address and control lines. AC timing measurements are referenced to 1.5 volts (except for clock, which is referenced to the 10% and 90% points). Maximum capacitive load for CLK is 125 pf.

The Ordering Information section lists temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.

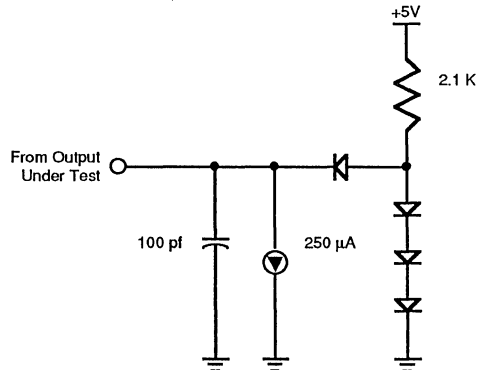


Figure 44. Standard Test Load

## CAPACITANCE

Guaranteed by design and characterization.

Symbol	Parameter	Min	Max	Unit
$C_{\text{clock}}$	Clock Capacitance	35	pf	
$C_{\text{in}}$	Input Capacitance	5	pf	
$C_{\text{out}}$	Output Capacitance	15	pf	

## DC CHARACTERISTICS

Z84011/Z84C11

$V_{CC}=5.0V \pm 10\%$  unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Condition
$V_{OLC}$	Clock Output High Voltage	$V_{CC}-0.6$		V	-2.0mA
$V_{OHC}$	Clock Output Low Voltage		0.4	V	+2.0mA
$V_{IL}$	Input High Voltage	2.2	$V_{CC}$	V	
$V_{IH}$	Input Low Voltage	-0.3	0.8	V	
$V_{OL}$	Output Low Voltage		0.4 [5]	V	$I_{LO}=2.0mA$
$V_{OH1}$	Output High Voltage	2.4 [4]		V	$I_{OH}=-1.6mA$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$ [5]		V	$I_{OH}=-250\mu A$
$I_{CC1}$	Power Supply Current XTALIN=10MHz XTALIN= 6MHz		TBD 30	mA mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
$I_{CC2}$	Power Supply Current (STOP Mode)		50	$\mu A$	$V_{CC}=5V$
$I_{CC3}$	Power Supply Current (IDLE1 Mode) XTALIN=10MHz XTALIN= 6MHz		TBD 4	mA mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
$I_{CC4}$	Power Supply Current (IDLE2 Mode; C11 only) XTALIN=10MHz XTALIN= 6MHz		TBD [1] TBD [1]	mA mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
$I_{LI}$	Input Leakage Current	-10	10 [4]	$\mu A$	$V_{IN}=0.4V$ to $V_{CC}$
$I_{LO}$	Tri-state Output Leakage Current in Float	-10	10 [2]	$\mu A$	$V_{OUT}=0.4V$ to $V_{CC}$
$I_{OHD}$	CTC ZC/TO Darlington Drive Current	-1.5	-5.0	mA	$V_{OH}=1.5V$ $R_{EXT}=1.1K\Omega$

### Notes:

- [1] Measurements made with outputs floating
- [2] A15-A0, D7-D0, /MREQ, /IORQ, /RD and /WR.
- [3]  $I_{CC2}$  Standby Current is guaranteed when the halt pin is low in STOP mode
- [4] All Pins except XTAL1, where  $I_{LI}=\pm 25\mu A$
- [5] A15-A0, D7-D0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1 and /BUSACK.

## AC CHARACTERISTICS

Z84011/Z84C11

Table A. CPU Timing (Figure 28 to 36)

No	Symbol	Parameter	Z84x1106		Z84x1110		Unit	Note
			Min	Max	Min	Max		
1	TcC	Clock Cycle Time	162**	DC	100**	DC	ns	[A1]
2	TwCh	Clock Pulse Width (high)	65	DC	40	DC	ns	[A1]
3	TwCl	Clock Pulse Width (low)	65	DC	40	DC	ns	[A1]
4	TfC	Clock Fall Time		20		10	ns	[A1]
5	TrC	Clock Rise Time		20		10	ns	[A1]
6	TdCr(A)	Address Valid From Clock Rise		90		65	ns	
7	TdA(MREQf)	Address Valid To /MREQ Fall	35**		5**		ns	
8	TdCf(MREQf)	Clock Fall To /MREQ Fall Delay		70		55	ns	
9	TdCr(MREQr)	Clock Rise To /MREQ Rise Delay		70		55	ns	
10	TwMREQh	/MREQ Pulse Width (high)	65**		30**		ns	[A2]
11	TwMREQl	/MREQ Pulse Width (low)	132**		75**		ns	[A2]
12	TdCf(MERQr)	Clock Fall To /MREQ Rise Delay		70		55	ns	
13	TdCf(RDf)	Clock Fall To /RD Fall Delay		80		65	ns	
14	TdCr(RDr)	Clock Rise To /RD Rise Delay		70		55	ns	
15	TsD(Cr)	Data Setup Time To Clock Rise	30		25		ns	
16	ThD(RDr)	Data Hold Time After /RD Rise	0		0		ns	
17	TsWAIT(Cf)	/WAIT Setup Time To Clock Fall	60		20		ns	
18	ThWAIT(Cf)	/WAIT Hold Time After Clock Fall	10		10		ns	
19	TdCr(M1f)	Clock Rise To /M1 Fall Delay		80		65	ns	
20	TdCr(M1r)	Clock Rise To /M1 Rise Delay		80		65	ns	
21	TdCr(RFSHf)	Clock Rise To /RFSH Fall Delay		110		80	ns	
22	TdCr(RFSHr)	Clock Rise To /RFSH Rise Delay		100		80	ns	
23	TdCf(RDr)	Clock Fall To /RD Rise Delay		70		55	ns	
24	TdCr(RDf)	Clock Rise To /RD Fall Delay		70		55	ns	
25	TsD(Cf)	Data Setup To Clock Fall During M2, M3, M4 Or M5 Cycles	40		25		ns	
26	TdA(IORQf)	Address Stable Prior To /IORQ Fall	107**		50**		ns	
27	TdCr(IORQf)	Clock Rise To /IORQ Fall Delay		65		50	ns	
28	TdCf(IORQr)	Clock Fall To /IORQ Rise Delay		70		55	ns	
29	TdD(WRf)Mw	Data Stable Prior To /WR Fall	22**		40**		ns	
30	TdCf(WRf)	Clock Fall To /WR Fall Delay		70		55	ns	
31	TwWR	/WR Pulse Width	132**		75**		ns	
32	TdCf(WRr)	Clock Fall To /WR Rise Delay		70		55	ns	
33	TdD(WRf)IO	Data Stable Prior To /WR Fall	-55**		-10**		ns	
34	TdCr(WRf)	Clock Rise To /WR Fall Delay		60		50	ns	
35	TdWRr(D)	Data Stable From /WR Rise	30**		10**		ns	
36	TdCf(HALT)	Clock Fall to /HALT '0' or '1'		260		90	ns	
37	TwNMI	/NMI Pulse Width	60		60		ns	
38	TsBUSREQ (Cr)	/BUSREQ Setup Time To Clock Rise	50		30		ns	



**AC CHARACTERISTICS** (Continued)  
Z84011/Z84C11

**Table A. CPU Timing (Figure 28 to 36)** (Continued)

No	Symbol	Parameter	Z84x1106		Z84x1110		Unit	Note
			Min	Max	Min	Max		
39	ThBUSREQ (Cr)	/BUSREQ Hold Time After Clock Rise	10		10		ns	
40	TdCr (BUSACKf)	Clock Rise To /BASACK Fall Delay		90		75	ns	
41	TdCf (BUSACKr)	Clock Fall To /BASACK Rise Delay		90		75	ns	
42	TdCr(Dz)	Clock Rise To Data Float Delay		80		65	ns	
43	TdCr(CTz)	Clock Rise To Control Outputs Float Delay (/MREQ, /IORQ, /RD And /WR)		70		65	ns	
44	TdCr(Az)	Clock Rise To Address Float Delay		80		75	ns	
45	TdCTr(A)	Address Hold Time From /MREQ, /IORQ, /RD Or /WR	35**		20**		ns	
46	TsRESET(Cr)	/RESET To Clock Rise Setup Time	60		40		ns	
47	ThRESET(Cr)	/RESET To Clock Rise Hold Time	10		10		ns	
48	TsINTf(Cr)	/INT Fall To Clock Rise Setup Time	70		50		ns	
49	ThINTR(Cr)	/INT Rise To Clock Rise Hold Time	10		10		ns	
50	TdM1f (IORQf)	/M1 Fall To /IORQ Fall Delay	359**		220**		ns	
51	TdCf(IORQf)	Clock Fall To /IORQ Fall Delay		70		55	ns	
52	TdCf(IORQr)	Clock Rise To /IORQ Rise Delay		70		55	ns	
53	TdCf(D)	Clock Fall To Data Valid Delay		130		110	ns	
54	TRDf(D)	/RD Fall To Output Data Valid		TBD		60	ns	
55	TMI(D)	/IORQ Fall To Output Data Valid		TBD		70	ns	
56	TwRESET	/RESET Pulse Width 011, Or C11 With RESET Output Disabled	3TcC		3TcC		ns	[A3]
57	TwRESEToe	/RESET Pulse Width C11 Only; RESET Output Enabled	2TcC		2TcC		ns	[A3]
58	TwRESETdo	/RESET Drive Duration C11 Only; RESET Output Enabled	16TcC		16TcC		ns	[A3]
59	TwRESETpor	/RESET Drive Duration On Power-On Sequence (C11 Only)	25	75	25	75	ms	[A3]

**Note for Table A.**

\*\* For clock period other than the minimum shown, calculate parameters using the formula on Footnotes to Table A.

[A1] These parameters apply to C11 and the external Clock input on CLK pin.

For cases where external Clock is fed from XTAL1, please refer to Table B.

[A2] For loading  $\geq 50$ pf. Decrease width by 10nS for each additional 50pf.

[A3] Apply to Z84C11 only.

**Footnotes to Table A**

Number	Symbol	General Parameter	Z84x1106	Z84x1110
1	TcC	$TwCh + TwCl + TrC + TfC$		
7	TdA(MREQf)	$TwCh + TfC$	-50	-45
10	TwMREQh	$TwCh + TfC$	-20	-20
11	TwMREQl	TcC	-30	-25
26	TdA(IORQf)	TcC	-55	-50
29	TdD(WRf)	TcC	-140	-60
31	TwWR	TcC	-30	-25
33	TdD(WRf)	$TwCl + TrC$	-140	-60
35	TdWRr(D)	$TwCl + TrC$	-55	-40
45	TdCTr(A)	$TwCl + TrC$	-50	-30
50	TdM1f(IORQf)	$2TcC + TwCh + TfC$	-50	-30

**AC CHARACTERISTICS** (Continued)  
Z84011/Z84C11

**Table B. CGC Timing (Figure 36 to 39)**

No	Symbol	Parameter	Z84x1106		Z84X1110		Unit	Note
			Min	Max	Min	Max		
1	TRST(INT)S	Clock Restart Time By /INT (STOP Mode)	(Typ) (2 <sup>14</sup> +2.5) TcC		(Typ) (2 <sup>14</sup> +2.5) TcC		ns	
2	TRST(MNI)S	Clock Restart Time By /NMI (STOP Mode)	(Typ) (2 <sup>14</sup> +2.5) TcC		(Typ) (2 <sup>14</sup> +2.5) TcC		ns	
3	TRST(INT)I	Clock Restart Time By /INT (IDLE Mode)	(Typ) 2.5TcT		(Typ) 2.5TcT		ns	
4	TRST(MNI)I	Clock Restart Time By /NMI (IDLE Mode)	(Typ) 2.5TcT		(Typ) 2.5TcT		ns	
5	TRST (RESET)I	Clock Restart Time By /RESET (IDLE Mode)	(Typ)1TcC		(Typ)1TcC		ns	
6	TfCLKOUT	CLK Rise Time		15		10	ns	[B1]
7	TrCLKOUT	CLK Fall Time		15		10	ns	[B1]
8	TcX1	XTAL1 Cycle Time (For External Clock Input On XTAL1) Divide-By-Two Mode Divide-By-One Mode (C11 Only)	81 162		50 100		ns ns	[B2]
9	TwlX1	XTAL1 Low Pulse Width (For External Clock Input On XTAL1) Divide-By-Two Mode Divide-By-One Mode (C11 Only)	35 65		20 40		ns ns	[B2]
10	TwhX1	XTAL1 High Pulse Width (For External Clock Input On XTAL1) Divide-By-Two Mode Divide-By-One Mode (C11 Only)	35 65		20 40		ns ns	
11	TrX1	XTAL1 Rise Time (For External Clock Input On XTAL1)		25		25	ns	[B3]
12	TfX1	XTAL1 Fall Time (For External Clock Input On XTAL1)		25		25	ns	[B3]

**Note for Table B.**

[B1] These parameters apply for 011 CLK pin (as System Clock Output), and C11 when the CLK pin outputs the system clock.

[B2] Not applicable to Z84011

[B3] If the parameters B8 and B9 are not met, adjust parameters B11 and B12 to satisfy parameters 8 and 9.

**Table C. Timing for On-chip Peripheral Access from External Bus Master  
and Daisy Chain Timing (See Figure 40)**

No	Symbol	Parameter	Z84x1106		Z84X1110		Unit	Note
			Min	Max	Min	Max		
1	TsA(Rlf)	Address Setup Time To /RD, /IORQ Fall	50		40		ns	
2	TsRI(Cr)	/RD, /IORQ Rise To Clock Rise Setup	60		50		ns	
3	Th	Hold Time For Specified Setup	15		15		ns	
4	TdCr(DO)	Clock Rise To Data Out Delay		100		80	ns	
5	TdRIr(DOz)	/RD, /IORQ Rise To Data Out Float Delay		75		60	ns	
6	ThRDr(D)	/M1, /RD, /IORQ Rise To Data Hold	15	40	15	30	ns	[C1]
7	TsD(Cr)	Data In to Clock Rise Setup Time	30		25		ns	
8	TdIOI(DOI)	/IORQ Fall To Data Out Delay (INTACK Cycle)		95		95	ns	
9	ThIOr(D)	/IORQ Rise To Data Hold	15		15		ns	
10	ThIOr(A)	/IORQ Rise To Address Hold	15		15		ns	
11	ThWIf(Cr)	/IORQ, /WR Setup Time To Clock Rise	20		20		ns	[C2]
12	ThWRr(Cr)	Clock Rise To /IORQ, /WR Rise Hold Time	0		0		ns	[C2]
13	TsM1f(Cr)	/M1 Fall To Clock Rise Setup Time	40		40		ns	
14	TsM1r(Cf)	/M1 Rise To Clock Rise Setup Time (/M1 Cycle)	-15		-15		ns	
15	TdM1f(IEOf)	/M1 Fall To IEO Fall Delay (Interrupt Immediately Preceding /M1 Fall)		130		70	ns	
16	TsIEI(IOI)	IEI To /IORQ Fall Setup Time (INTACK Cycle)	100		70		ns	
17	TdIEIf(IEOf)	IEI Fall To IEO Fall Delay		100		70	ns	
18	TdIEIr(IEOr)	IEI Rise To IEO Rise Delay (After ED Decode)		110		70	ns	
19	TsIEI(Cr)	IEI to Clock Fall Setup (For 4D Decode)		160		150	ns	
20	TdCf(IEOr)	Clock Fall To IEO Rise Delay	50		40		ns	
21	TdCf(IEOf)	Clock Fall To IEO Rise Delay		90		75	ns	

**Note to Table C.**

[C1] For I/O write to CTC.

[C2] For I/O Write to system control registers.

**AC CHARACTERISTICS** (Continued)  
Z84011/Z84C11

**Table D. CTC Timing (Figure 42)**

No	Symbol	Parameter	Z84x1106		Z84X1110		Unit	Note
			Min	Max	Min	Max		
1	TdCr(INTf)	Clock Rise To /INT Fall Delay		(TcC+100)		(TcC+80)	ns	[D1]
2	TsCTRr (Cr)c	CLK/TRG Rise To Clock Rise Setup Time For Immediate Count	90		90		ns	[D2]
3	TsCTR(Ct)	CLK/TRG Rise To Clock Rise Setup Time For Enabling Of Prescaler On Following Clock Rise	90		90		ns	[D1]
4	TdCTRr (INTf)	CLK/TRG Rise To /INT Fall Delay		(1)+(2)		(1)+(2)	ns	[D2]
		TsCTR(C) Satisfied TsCTR(C) Not Satisfied		TcC + (1)+(2)		TcC + (1)+(2)	ns	[D2]
5	TcCTR	CLK/TRG Cycle Time	(2TcC)	DC	(2TcC)	DC	ns	[D3]
6	TwCTRh	CLK/TRG Width (low)	90	DC	90	DC	ns	
7	TwCTRl	CLK/TRG Width (high)	90	DC	90	DC	ns	
8	TrCTR	CLK/TRG Rise Time		30		30	ns	
9	TlCTR	CLK/TRG Fall Time		30		30	ns	
10	TdCr(ZCr)	Clock Rise To ZC/TO Rise Delay		80		80	ns	
11	TdC(ZCf)	Clock Fall To ZC/TO Fall Delay		80		80	ns	

**Notes for Table D.**

[D1] Timer Mode

[D2] Counter Mode

[D3] Counter Mode Only; When using a cycle time less than 3TcC, parameter D2 must be met.

**Table E. General Purpose I/O Port Timing (Figure 43)**

No	Symbol	Parameter	Z84x1106		Z84X1110		Unit	Note
			Min	Max	Min	Max		
1	TdCf(Pout)	Clock Fall to Port Data Valid Delay		300		300	ns	
2	TsPin (IORDf)	Port Data to /IORQ and /RD Fall Setup Time	0		0		ns	
3	ThPin	Port Input to /IORQ and /RD Fall Hold Time	0		0		ns	

**Table F. Watchdog Timer Timing (C11 Only; Figure 44)**

No	Symbol	Parameter	Z84x1106		Z84X1110		Unit	Note
			Min	Max	Min	Max		
1	TdC(WDTf)	Clock Rise To /WDTOUT Fall Delay		160		160	ns	
2	TwPI	Clock Rise To /WDTOUT Rise Delay		165		165	ns	
3	TcWDT	/WDTOUT Cycle Time						
		WDTP = 00	(Typ)		(Typ)		ns	
			$2^{16}TcC$		$2^{16}TcC$			
		WDTP = 01	(Typ)		(Typ)		ns	
			$2^{18}TcC$		$2^{18}TcC$			
		WDTP = 10	(Typ)		(Typ)		ns	
			$2^{20}TcC$		$2^{20}TcC$			
		WDTP = 11	(Typ)		(Typ)		ns	
			$2^{22}TcC$		$2^{22}TcC$			

---

# Z84013/015

## Z84C13/Z84C15

### IPC INTELLIGENT PERIPHERAL CONTROLLER

#### FEATURES

- Z84C00 Z80 CPU with Z84C30 CTC, Z84C4X SIO, CGC, Watch Dog Timer(WDT). In addition, Z84C15 and Z84015 have Z84C20 PIO.
  - Built-in Watch Dog Timer (WDT).
  - Noise filter to CLK/TRG inputs of the CTC.
  - 84-pin PLCC package.
- High speed operation 6, 10 MHz
- 16 MHz operation for Z84C15 only.
- Low power consumption in four operation modes:
  - 41 mA Typ. (Run mode)
  - 6 mA Typ. (Idle1 mode)
  - 60  $\mu$ A Typ. (Idle2 mode)
  - 0.5  $\mu$ A Typ. (Stop mode)
- Wide operational voltage range (5V  $\pm$  10%).
- TTL/CMOS compatible.
- Z84013 features:
  - Z84C00 Z80 CPU
  - On-chip two channel SIO (Z80 SIO).
  - On-chip four channel Counter Timer Controller (Z80 CTC).
  - Built-in Clock Generator Controller (CGC).
- Z84015 features:
  - All Z84013 features, plus on-chip two 8-bit ports (Z80 PIO) and 100-pin QFP package.
- Z84C13/Z84C15 enhancements to Z84013/Z84015:
  - Power-on reset.
  - Addition of two chip select pins.
  - 32-bit CRC for Channel A of SIO.
  - Wait state generator.
  - Simplified EV mode selection.
  - Schmitt-trigger inputs to transmit and receive clocks of the SIO.
  - Crystal divide-by-one mode.
  - 100-pin VQFP (Z84C15 only)

#### GENERAL DESCRIPTION

The Intelligent Peripheral Controller (IPC) is a series of highly superintegrated devices with four versions. The Z84C13 and the Z84C15 are upward compatible versions of the Z84013 and the Z84015. The Z84015 is a CMOS 8-bit microprocessor integrated with the CTC, SIO, CGC, WDT and the PIO into a single 100-pin Quad Flat Pack(QFP) package. The Z84013 is the Z84015 without PIO, and is housed in a 84-pin PLCC package. The Z84C13 is the Z84013 with enhancements and the Z84C15 is the Z84015 with enhancements. These high-end superintegrated intelligent peripheral controllers are targeted for a broad

range of applications ranging from error correcting modems to enhancement/cost reductions of existing hardware using Z80-based discrete peripherals. Figures 1 and 2 show the difference between the Z84013/015 and the Z84C13/Z84C15.

Hereinafter, use the word IPC on the description covering all versions (Z84C13/Z84C15 and Z84013/Z84015). Use Z84C13/C15 on the description that applies only to the Z84C13 and Z84C15, and use Z84013/015 on the description that applies only to the Z84013 and Z84015.



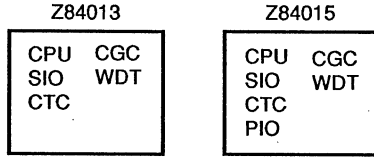


Figure 1. Z84013/015 Version

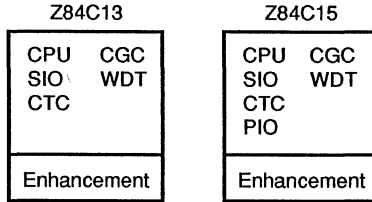
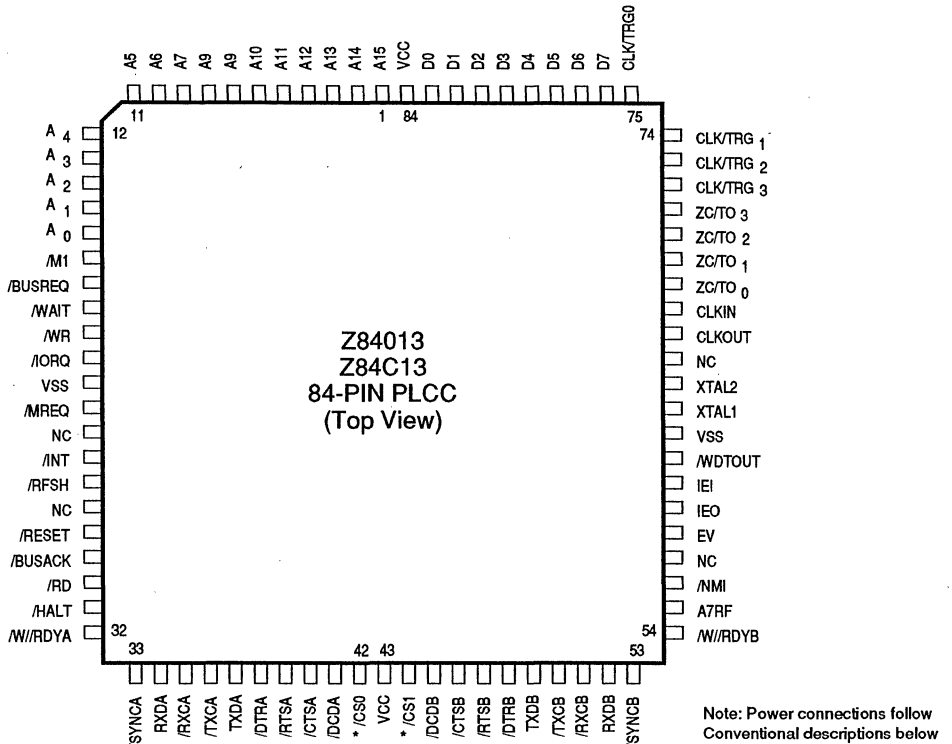
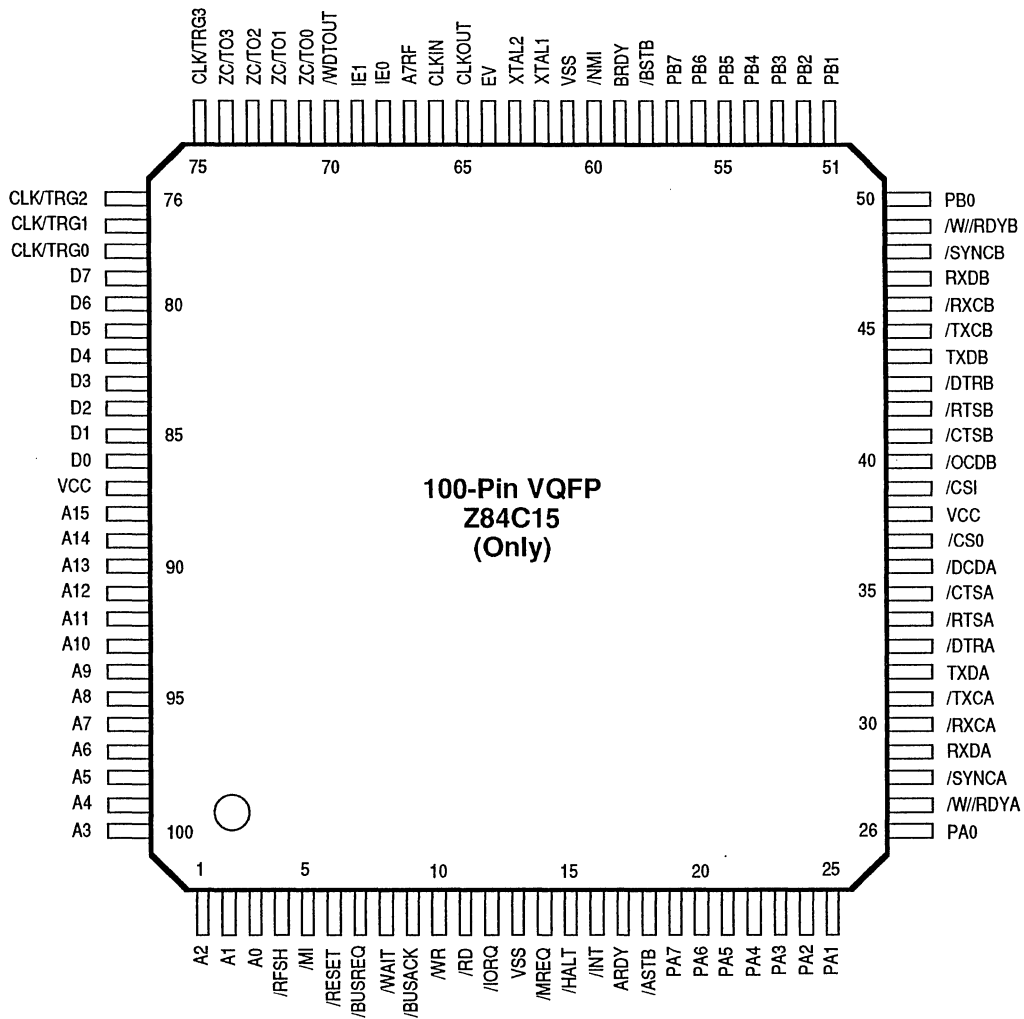


Figure 2. Z84C13/C15 Version

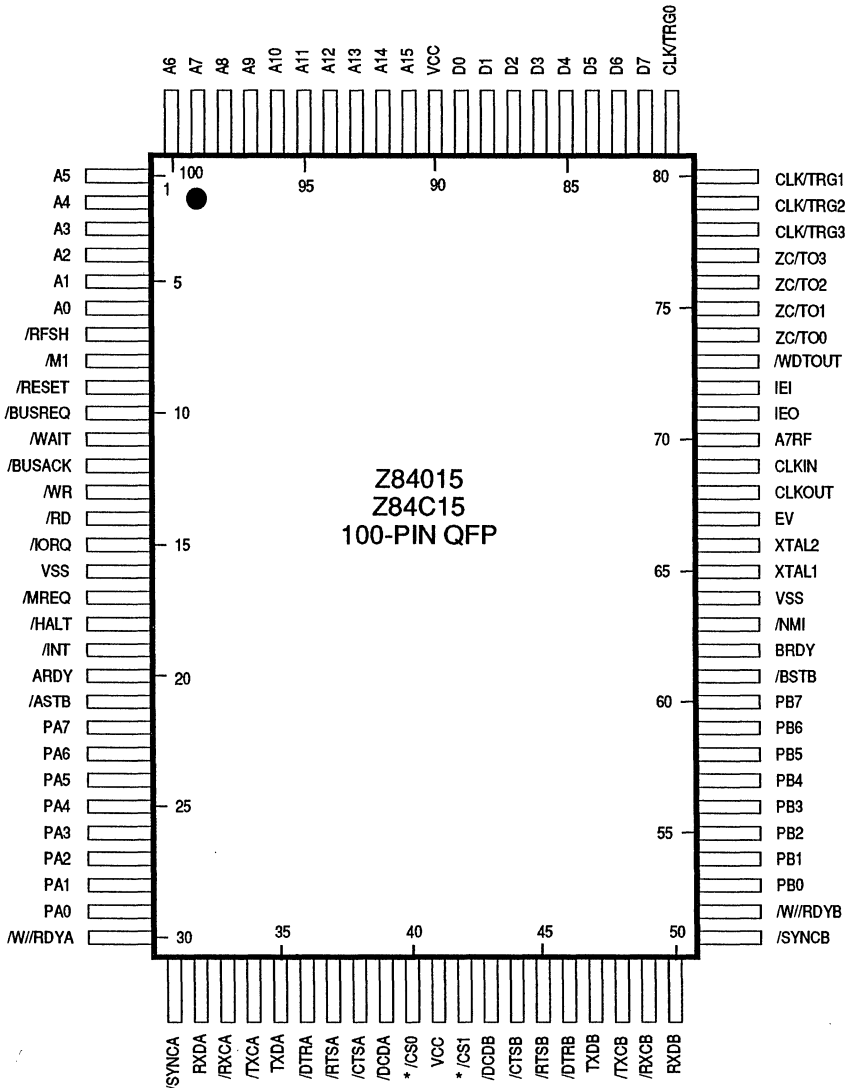


\* ICT for the Z84013

Figure 3. Z84013/Z84C13 Pin-out Assignments



**Z84C15 Pin-out Assignments**



\* ICT for the Z84015

**Figure 4. Z84015/Z84C15 Pin-out Assignments**

## PIN DEFINITIONS

The pin assignment for each device is shown in Figures 3 and 4. Following is the description on each pin. For the description and the pin number, if stated as "x13" or "x15",

that applies to both Z84C13/Z84013 or Z84C15/Z84015. Otherwise, C13 for Z84C13, C15 for Z84C15, 013 for Z84013 and 015 for Z84015.

---

## CPU SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
A0-A15	16-1(x13), 6-1, 100-91(x15)	I/O	16-bit address bus. Specifies I/O and memory addresses to be accessed. During the refresh period, addresses for refreshing are output. The bus is an input when the external master is accessing the on-chip peripherals.
D0-D7	83-76(x13), 89-82(x15)	I/O	8-bit bidirectional data bus. When the on-chip CPU is accessing on-chip peripherals, these lines are set to output and hold the data to/from on-chip peripherals.
/RD	30(x13), 14(x15)	I/O	Read signal. CPU read signal for accepting data from memory or I/O devices. When an external master is accessing the on-chip peripherals, it is an input signal.
/WR	20(x13), 13(x15)	I/O	Write Signal. This signal is output when data, to be stored in a specified memory or peripheral LSI, is on the MPU data bus. When an external master is accessing the on-chip peripherals, it is an input signal.
/MREQ	23(x13), 17(x15)	I/O, 3-State	Memory request signal. When an effective address for memory access is on the address bus, "0" is output. When an external master is accessing the on-chip peripherals, it is a tri-state signal.
/IORQ	21(x13), 15(x15)	I/O	I/O request signal. When addresses for I/O are on the lower 8 bits (A7-A0) of the address bus in the I/O operation, "0" is output. In addition, the /IORQ signal is output with the /M1 signal at the time of interrupt acknowledge cycle to inform peripheral LSI of the state of the interrupt response vector is when put on the data bus. When an external master is accessing the on-chip peripherals, it is an input signal.
/M1	17(x13), 8(x15)	I/O	Machine cycle "1". /MREQ and "0" are output together in the operation code fetch cycle. /M1 is output for every opcode fetch when a two byte opcode is executed. In the maskable interrupt acknowledge cycle, this signal is output together with /IORQ. It is 3-stated in EV mode.

---

---

## CPU SIGNALS (Continued)

Pin Name	Pin Number	Input/Output, 3-State	Function
/RFSH	26(x13), 7(x15)	Out, 3-State	The refresh signal. When the dynamic memory refresh address is on the low order byte of the address bus, /RFSH is active along with /MREQ signal. This pin is 3-stated in EV mode.
/INT	25(x13), 19(x15)	Open drain	Maskable interrupt request signal. Interrupt is generated by peripheral LSI. This signal is accepted if the interrupt enable Flip-Flop (IFF) is set to "1". The /INT signal of on-chip peripherals is internally wired - OR without pull-up resistors and requires external pull-up. Also, interrupts from on-chip peripherals go out from this pin.
/NMI	56(x13), 63(x15)	In	Non-maskable interrupt request signal. This interrupt request has a higher priority than the maskable interrupt request and does not rely upon the state of the interrupt enable Flip-Flop (IFF).
/HALT	31(x13), 81(x15)	Out, 3-State	Halt signal. Indicates that the CPU has executed a HALT instruction. This signal is 3-stated in EV mode.
/BUSREQ	18(x13), 10(x15)	In	BUS request signal. /BUSREQ requests placement of the address bus, data bus, /MREQ, /IORQ, /RD and /WR signals into the high impedance state. /BUSREQ is normally wired-OR and a pull-up resistor is externally connected.
/BUSACK	29(x13), 12(x15)	Out (O13/O15), Out/3-State (C13/C15)	Bus Acknowledge signal. In response to /BUSREQ signal, /BUSACK informs a peripheral LSI that the address bus, data bus, /MREQ, /IORQ, /RD and /WR signals have been placed in the high impedance state.
/WAIT	19(x13), 11(x15)	In(O13/O15), I/O(C13/C15)	Wait signal. /WAIT informs the CPU that specified memory or peripheral is not ready for data transfer. As long as /WAIT signal is active, MPU is continuously kept in the wait state.

**Note:** For the Z84013/015 the /BUSACK signal will not be 3-stated during EV mode. For the Z84C13/C15 the /BUSACK will be 3-stated during EV mode.

**Note:** For the Z84C13/C15, the /WAIT pin becomes an output to bring out on-chip wait state generator during the EV mode.

---

---

## CPU SIGNALS (Continued)

---

Pin Name	Pin Number	Input/Output, 3-State	Function
A7RF	55(x13), 70(x15)	Out	1-bit auxiliary address bus. Output is the same as bit-7 (A7) of the address bus. However, during a refresh cycle, this pin outputs the address which is the most significant bit of the 8-bit refresh address signal linked to the low order 7 bits of the address bus.

---

## CTC SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
CLK/TRG0 - CLK/TRG3	75-72(x13), 81-78(x15)	In	External clock/trigger input. These four CLK/TRG pins correspond to four Counter/Timer Channels. In the counter mode, each active edge will cause the downcounter to decrement by one. In timer mode, an active edge will start the timer. It is program selectable whether the active edge is rising or falling.
ZC/TO0 - ZC/TO3	68-71(x13), 74-77(x15)	Out	Zero count/timer out signal. In either timer or counter mode, pulses are output when the down-counter has reached zero.

---

## SIO SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
/W//RDYA, /W//RDYB	32,54(x13), 30,52(x15)	Out	Wait/Ready signal A and Wait/Ready signal B. Used as /WAIT or /READY depending upon SIO programming. When programmed as /WAIT they go active at "0", alerting the CPU that addressed memory or I/O devices are not ready by requesting the CPU to wait. When programmed as /READY, they are active at "0" which determines when a peripheral device associated with a DMA port is for read/write data.
/SYNCA, /SYNCB	33,53(x13), 31,51(x15)	I/O	Synchronous signals. In asynchronous receive mode, they act as /CTS and /CDC. In external sync mode, these signals act as inputs. In internal sync mode, they act as outputs.
RxDA, RxDB	34,52(x13), 32,50(x15)	In	Serial receive data signal.

---

---

## SIO SIGNALS (Continued)

---

Pin Name	Pin Number	Input/Output, 3-State	Function
/RxCA, /RxCB	35,51(x13), 33,49(x15)	In	Receive clock signal. In the asynchronous mode, the receive clocks can be 1, 16, 32, or 64 times the data transfer rate.
/TxCA, /TxCB	36,50(x13), 34,48(x15)	In	Transmitter clock signal. In the asynchronous mode, the transmitter clocks can be 1, 16, 32, or 64 times the data transfer rate.
TxDA, TxDB	37,49(x13), 35,47(x15)	Out	Serial transmit data signal.
/DTRA, /DTRB	38,48(x13), 36,46(x15)	Out	Data terminal ready signal. When ready, these signals go active to enable the terminal transmitter. When not ready they go inactive to disable the transfer from the terminal.
/RTSA, /RTSB	39,47(x13), 37,45(x15)	Out	Request to send signal. "0" when transmitting serial data. They are active when enabling their receivers to transmit data.
/CTSA, /CTSB	40,46(x13), 38,44(x15)	In	Clear to send signal. When "0", after transmitting these signals the modem is ready to receive serial data. When ready, these signals go active to enable terminal transmitter. When not ready, these signals go inactive to disable transfer from the terminal.
/DCDA, /DCDB	41,45(x13), 39,43(x15)	In	Data carrier detect signal. When "0", serial data can be received. These signals are active to enable receivers to transmit.

---

## SYSTEM CONTROL SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
IEI	60(x13), 72(x15)	In	Interrupt enable input signal. IEI is used with the IEO to form a priority daisy chain when there is more than one interrupt-driven peripheral.
IEO	59(x13), 71(x15)	Out	The interrupt enable output signal. In the daisy chain interrupt control, IEO controls the interrupt of external peripherals. IEO is active when IEI is "1" and the CPU is not servicing an interrupt from the on-chip peripherals.
/CS0 (C13/C15 only)	42(C13), 40(C15)	Out	Chip Select 0. Used to access external memory or I/O devices. This pin has been assigned to "ICT" pin on Z84013/015. This signal is decoded only from A15-A12 without control signals. Refer to "Functional Description" on-chip select signals for further explanation.

---

## SYSTEM CONTROL SIGNALS (Continued)

Pin Name	Pin Number	Input/Output, 3-State	Function
/CS1 (C13/C15 only)	40(x13), 42(x15)	Out	Chip Select 1. Used to access external memory or I/O devices. This pin has been assigned to "ICT" pin on Z84013/015. This signal is decoded only from A15-A12 without control signals. Refer to "Functional Description" on-chip select signals for further explanation.
/WDTOUT	61(x13), 73(x15)	Out(013/015), Open Drain(C13/C15)	Watch Dog Timer Output signal. Output pulse width depends on the externally connected pin.
/RESET	28(x13), 9(x15)	Input(013/015), I/O (Open Drain) (C13/C15)	Reset signal. /RESET signal is used for initializing MPU and other devices in the system. Also used to return from the steady state in the STOP or IDLE modes.

**Note:** For the Z84013/Z84015 the /RESET must be kept in active state for a period of at least three system clock cycles.

**Note:** For the Z84C13/Z84C15, during the power-up sequence, the /RESET becomes an Open drain output and the Z84C13/C15 will drive this pin to "0" for 25 to 75 msec after the power supply passes through approx. 2.2V and then reverts to input. If it receives the /RESET signal after power-on sequence, it will drive /RESET pin for 16-processor clock cycles depending on the status of Reset Output Disable bit in Misc Control Register. If this Reset output is disabled, it must be kept in active state for a period of at least three system clock cycles. Note, that if using Z84C13/C15 in a Z84013/015 socket, modification may be required on the reset circuit since this pin is "pure input pin" on the Z84013/015. Also, the /RESET pin doesn't have internal pull-up resistors and therefore requires external pull-ups. For more details on the device, please refer to "Functional Description."

XTAL1	63(x13), 65(x15)	In	Crystal oscillator connecting terminal. A parallel resonant crystal is recommended. If external clock source is used as an input to the CGC unit, supply clock goes into this terminal. If external clock is supply to CLKIN pin (without CGC unit), this terminal must be connected to "0" or "1".
XTAL2	63(x13), 66(x15)	Out	Crystal oscillator connecting terminal.
CLKIN	67(x13), 69(x15)	In	Single-phase System Clock Input.
CLKOUT	66(x13), 68(x15)	Out	Single-phase clock output from on-chip Clock Generator/Controller.
EV	58(x13), 67(x15)	In	Evaluator signal. When "1" is applied to this pin, IPC is put in Evaluation mode.

**Note:** For the Z84013/015, together with /BUSREQ, the EV signal puts the IPC into the evaluation mode. When this signal becomes active, the status of /M1, /HALT and /RFSH change to input. When using Z84013/015 as an evaluator chip, the CPU is electrically disconnected after one machine cycle is executed with the EV signal "1" and the /BUSREQ signal "0". It follows the instructions from the other CPU (of ICE). Upon receiving /BUSREQ; A15-A0, /MREQ, /IORQ, /RD and /WR are changed to input and D7-D0 changes its direction. /BUSACK is NOT 3-stated so it should be disconnected by an externally connected circuit. For details, please refer to "Functional Description" on EV mode.



---

## SYSTEM CONTROL SIGNALS (Continued)

---

**Note:** For the Z84C13/C15, to access on-chip resources from the CPU (e.g., ICE CPU), the CPU is electrically disconnected; A15-A0, /MREQ, /IORQ, /RD and /WR are changed to input; D7-D0 changes its direction; /M1, /HALT and /RFSH are put into the high impedance state when the EV pin is set to "1". Also, /BUSACK is 3-stated. For details, please refer to "Functional Description" on EV mode.

---

Pin Name	Pin Number	Input/Output, 3-State	Function
ICT	42,44(013), 40,42(015), Not with C13/C15	Out	Test pins. Used in the open state.
NC	24,27,57,65(x13), Not with x15		Not connected.
VCC	43,84(x13), 41,90(x15)	Power Supply	+5 Volts
VSS	22, 62(x13), 16,64(x15)	Power Supply	0 Volts

---

## PIO SIGNALS (for the Z84x15 only)

---

---

Pin Name	Pin Number	Input/Output, 3-State	Function
/ASTB	21(x15)	In	Port A strobe pulse from a peripheral device. The signal is used as the handshake between Port A and external circuits. The meaning of this signal depends on the mode of operation selected for Port A (see "PIO Basic Timing").
/BSTB	61(x15)	In	Port B strobe pulse from a peripheral device. This signal is used as the handshake between Port B and external circuits. The meaning of this signal is the same as /ASTB, except when Port A is in mode 2 (see "PIO Basic Timing").
ARDY	20(x15)	Out	Register A ready signal. Used as the handshake between Port A and external circuits. The meaning of this signal depends on the mode of operation selected for Port A (see "PIO Basic Timing").
BRDY	62(x15)	Out	Register B ready signal. Used as the handshake between Port B and external circuits. The meaning of this signal is the same as ARDY except when Port A is in mode 2 (see "PIO Basic Timing").
PA7-PA0	22-29(x15)	I/O, 3-State	Port A data signals. Used for data transfer between Port A and external circuits.
PB7-PB0	53-60(x15)	I/O, 3-State	Port B data signals. Used for transfer between Port B and external circuits.

---

---

## The following pins have different functions between 013/015 and C13/C15

Pin Name	Pin # X13	Pin # X15	Function
/RESET	28	9	Functionality is different.
/WAIT	19	15	Functionality is different.
EV	58	67	Functionality is different.
/WDTOUT	61	73	Push-pull output on Z84013/015, Open drain on Z84 C13/C15
ICT	40, 42	42, 40	(Test pin) on Z84013/015; /CS0 and /CS1 on Z84C13/15.
TxCA, TxCB, RxCA and RxCB	35, 36, 50, 51	33, 34, 48, 49	On Z84C13/15; these signals have Schmitt-triggered inputs.
/BUSACK	29	12	In EV mode, 3-stated on Z84C13/15; remains active on Z84013/015.

---

## FUNCTIONAL DESCRIPTION

Figure 5(a) shows the functional block diagram of the Z84013/015 and Figure 5(b) shows the functional block diagram of the Z84C13/C15. As described earlier, the only difference between the Z84x13 and the Z84x15 is the PIO not being available on the Z84x13.

Functionally, the on-chip SIO, PIO (not available on Z84x13), CTC, and the Z80 CPU are the same as the discrete devices. Therefore, for detailed description of each individual unit, refer to the Product Specification/Technical Manual of each discrete product.

The following subsections describe each individual functional unit of the IPC.

### Z84C00/01 Logic Unit

The CPU provides all the capabilities and pins of the Zilog Z80 CPU. This allows 100% software compatibility with existing Z80 software. In addition, it has the pin called "A7RF" to extend DRAM refresh address to 8-bits. Refer to "Z84C01 Z80 CPU with CGC" Product Specification.

### Z84C20 Parallel Input/Output Logic Unit (Z84x15 Only)

This logic unit provides both TTL- and CMOS-compatible interfaces between peripheral devices and a CPU through the use of two 8-bit parallel ports (Figure 6). The CPU configures the logic to interface to a wide range of peripheral devices with no external logic. Typical devices that are compatible with this interface are keyboards, printers, and EPROM/PAL programmers.

The parallel ports (designated Port A and Port B) are byte wide and completely compatible with the Z84C20 PIO.

These two ports have several modes of operation; input, output, bi-directional, or bit control mode. Each port has two handshake signals (RDY and /STB) which are used to control data transfers. The RDY (ready) indicates that the port is ready for data transfer while /STB (strobe) is an input to the port that indicates when data transfer has occurred. Each of the ports can be programmed to interrupt the CPU upon the occurrence of specified status conditions, and generate unique interrupt vectors when the CPU responds (for more information on the operation of this portion of the logic, please refer to the Z84C20 PIO Product Specification and Technical Manual).

### Z84C30 Counter/Timer Logic Unit

This logic unit provides the user with four individual 8-bit Counter/Timer Channels that are compatible with the Z84C30 CTC (Figure 7). The Counter/Timers can be programmed by the CPU for a broad range of counting and timing applications. Typical applications include event counting, interrupt and interval counting, and serial baud rate clock generation.

Each of the Counter/Timer Channels, designated Channels 0-3, have an 8-bit prescaler (when used in timer mode) and its own 8-bit counter to provide a wide range of count resolution. Each of the channels have their own Clock/Trigger input to quantify the counting process and an output to indicate zero crossing/timeout conditions. With only one interrupt vector programmed into the logic unit, each channel can generate a unique interrupt vector in response to the interrupt acknowledge cycle.

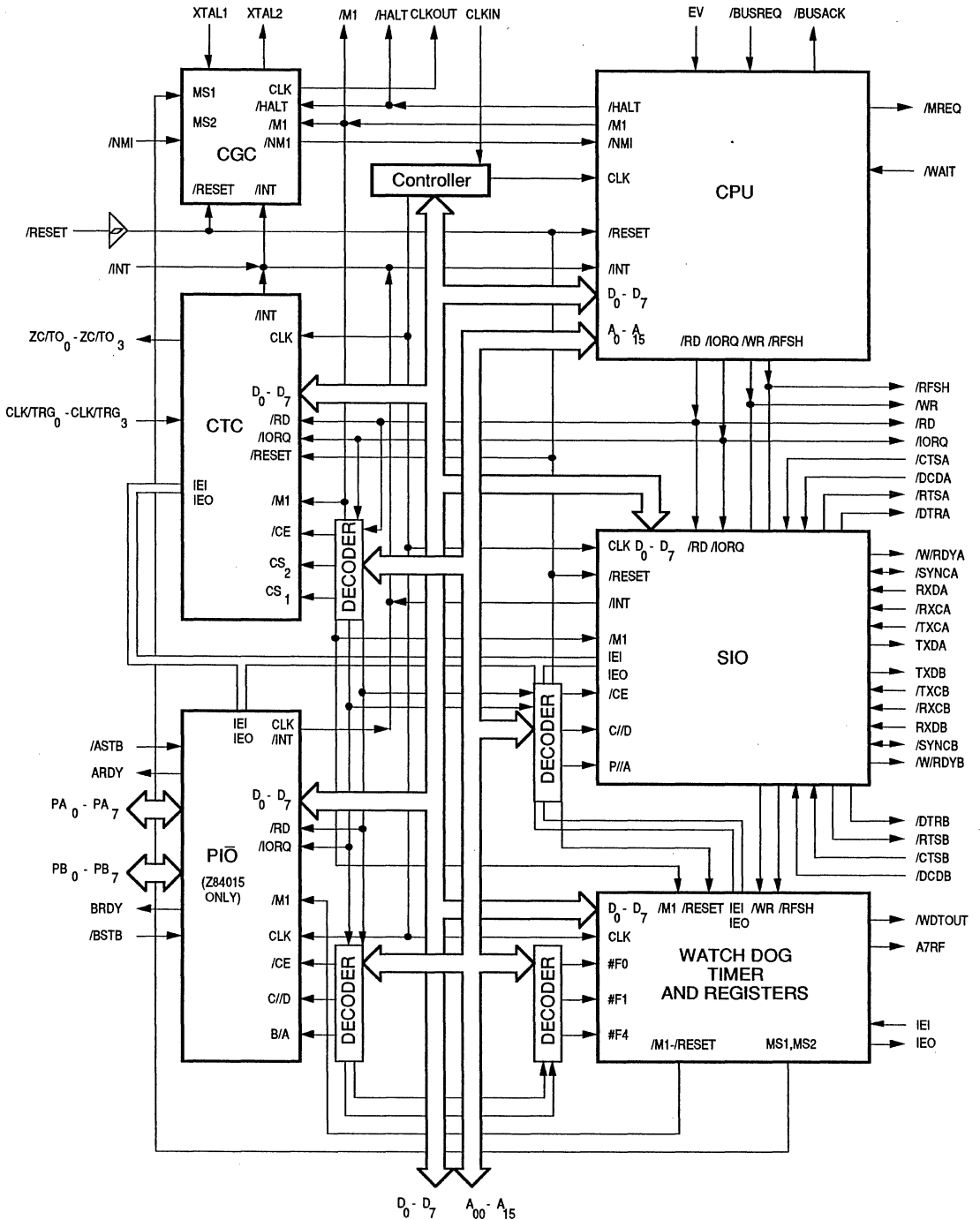


Figure 5(a). Block Diagram for 84013/015 IPC

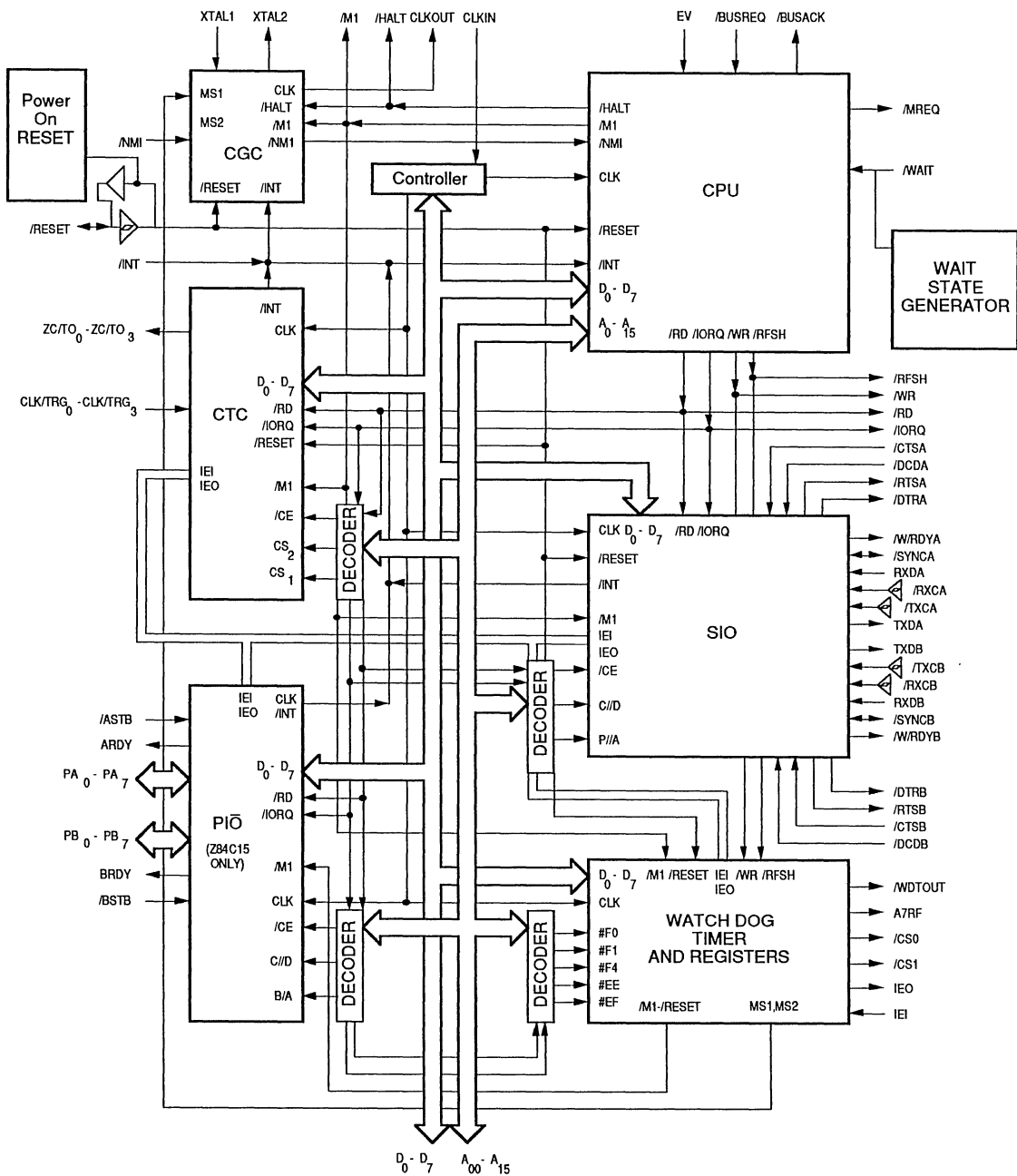


Figure 5(b). Block Diagram for 84C13/C15 IPC

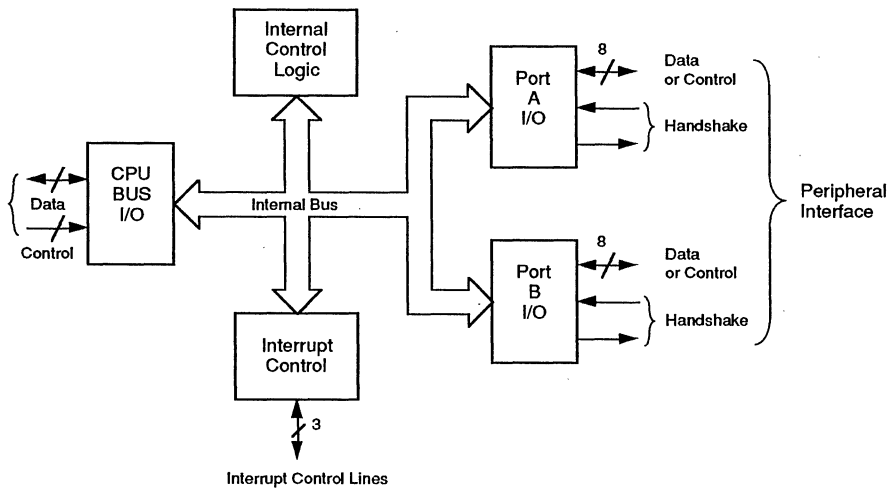


Figure 6. PIO Block Diagram

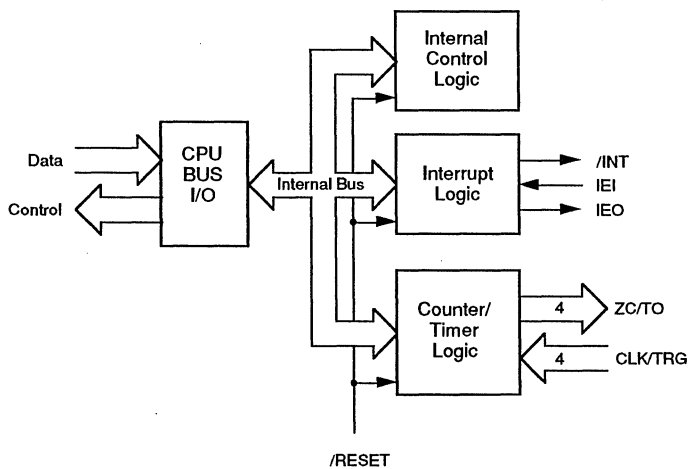


Figure 7. CTC Block Diagram

### Z84C4x Serial I/O Logic Unit

This logic unit provides the user with two separate multi-protocol serial I/O channels that are completely compatible with the Z84C4x SIO. Their basic functions as serial-to-parallel and parallel-to-serial converters can be programmed by a CPU for a broad range of serial communications applications. Each channel, designated Channel A and Channel B, is capable of supporting all common

asynchronous and synchronous protocols (Monosync, Bisync, and SDLC/HDLC, byte or bit oriented - Figure 8).

**Z84C13/C15 Only.** As an enhancement to the Z84013/015, the Z84C13/C15 can handle a 32-bit CRC on Channel A and Schmitt-trigger inputs on the /TxC and /RxC pins of both channels.

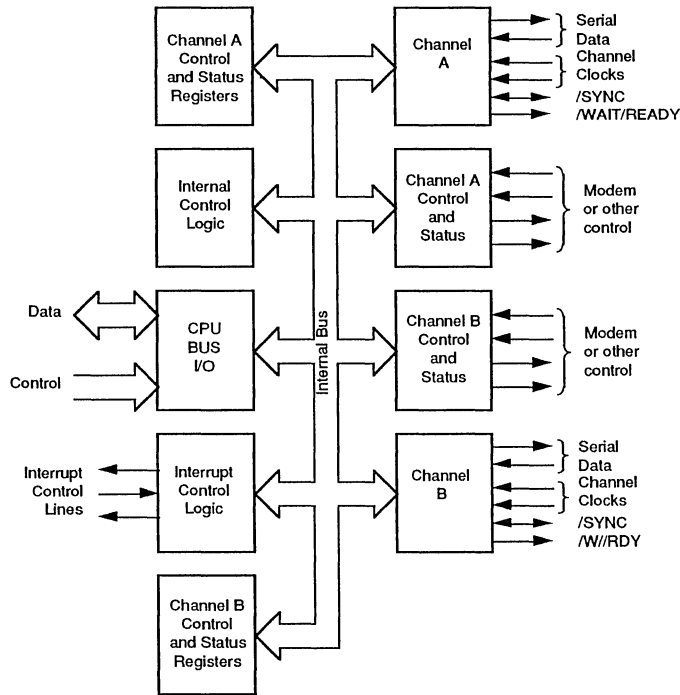


Figure 8. SIO Block Diagram

### Watch Dog Timer (WDT) Logic Unit

This logic unit has been superintegrated into the IPC. It detects an operation error, caused by the program run-away, and returns to normal operation. Figure 9, shows the block diagram of the WDT. Upon Power-On Reset, this unit is enabled. If WDT is not required, but /WDTOUT is connected to /RESET or any other circuit, it has to be disabled. During the power-down mode of operation (either IDLE1/2 or Stop), the Watch Dog Timer is halted.

**WDT Output (/WDTOUT pin).** When the WDT is used, the "0" level signal is output from the /WDTOUT pin after a duration of time specified in the WDTP or in the WDTMR. The output pulse width is one of the following, depending on the /WDTOUT pin connection.

- The /WDTOUT is connected to the /RESET pin: The "0" level is pulsed for 5T<sub>c</sub> (System clock cycles).
- The /WDTOUT is connected to a pin other than the /RESET pin: The "0" level is kept until the Watch Dog timer is cleared by software, or reset by /RESET pin.

**CGC Logic Unit.** The IPC has CGC (Clock Generator/Controller) unit. This unit is identical to the one with the Z84C01 and the Z84C50, and supports power-down modes of operation. The output from this unit is on the pin called CLKOUT, and is not connected to the system clock internally. The CLKIN pin is the system clock input. The user can connect CLKOUT to CLKIN to utilize this CGC unit, or supply external clock from CLKIN pin.

The CGC unit allows crystal input (XTAL1, XTAL2) or External Clock input on the XTAL1 pin. It has clock divide-by-two circuits and generates a half-speed clock to the input.

**Z84C13/C15.** The power-down modes of the IPC vary depending upon whether the system clock is fed from the CGC unit (tie CLKOUT to CLKIN) or the external clock source on the CLKIN pin. They also have divide-by-one Mode. If the clock is supplied by this CGC unit, all of the modes in "halt" state are available. When external clock is provided on the CLKIN pin, XTAL1 is not left open (tied to "0" or "1") to avoid meta-stable conditions to minimize power consumption.

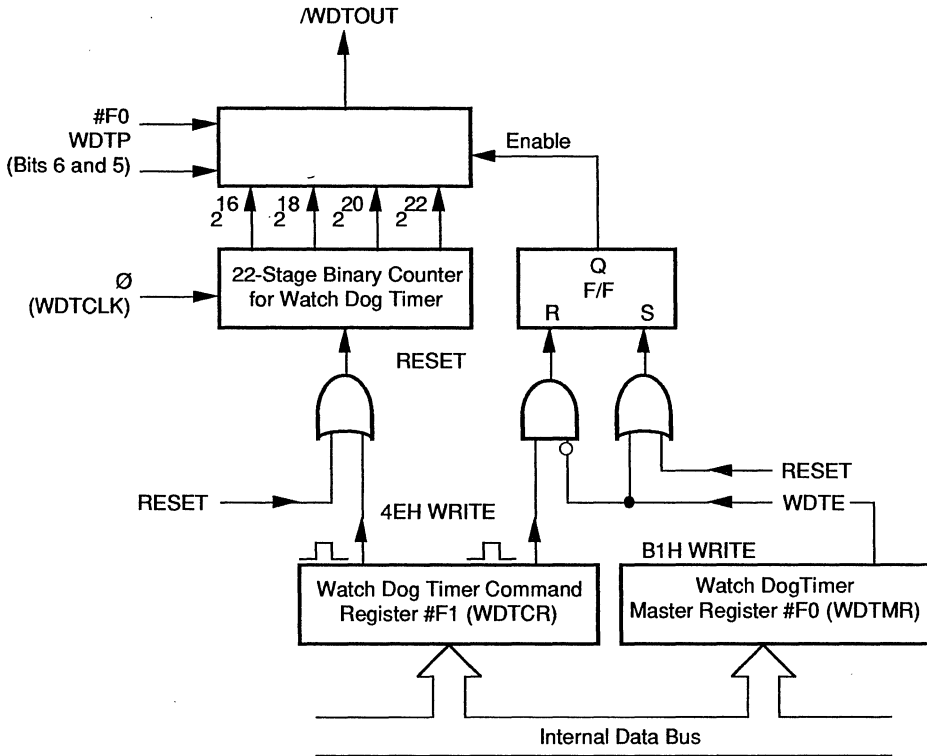


Figure 9. Block Diagram of Watch Dog Timer

**Z84013/015 Only.** If the system clock is provided on the CLKIN pin, none of the power-down mode (except RUN mode) is supported.

**Z84C13/C15.** Clock output is the same, or half, of the external frequency.

**Z84C13/C15 Only.** If the system clock is provided on the CLKIN pin, only the IDLE2 mode is applicable. In this mode, if the HALT instruction is executed, internal clock to the CTC is kept on "Continue", but the clock to the other components (CPU, PIO, SIO and Watch Dog Timer) are stopped. The divide-by-two circuit of the CGC unit can be skipped by programming bit D4 of the WDTMR (see "Programming" section). Upon Power-on Reset, it comes up in divide by two mode.

#### System Clock Generation

The IPC has a built-in oscillator circuit and the required clock can be easily generated by connecting a crystal to the external terminals (XTAL1, XTAL2). Clock output is the same frequency as half the speed of the crystal frequency. Example of oscillator connections are shown in Figure 10.

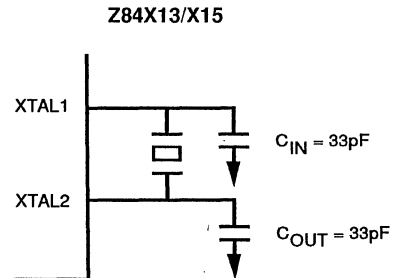


Figure 10. Circuit Configuration For Crystal

---

Recommended characteristics of the crystal and the values for the capacitor are as follows (the values will change with crystal frequency).

- Type of crystal: Fundamental, parallel type crystal (AT cut is recommended).
- Frequency tolerance: Application dependent.
- CL, Load capacitance: Approximately 22pf (acceptable range is 20-30pf).
- Rs, equivalent-series resistance:  $\leq 150$  ohms.
- Drive level: 10mW (for  $\leq 10$ MHz crystal); 5mW (for  $\geq 10$ MHz crystal).
- $C_{IN} = C_{OUT} = 33$ pF.

#### Power-On Reset Logic Unit (Z84C13/C15 Only)

The Z84C13/C15 has the enhanced feature of a Power-on Reset Circuit. During the power-up sequence, the open-drain gate of the on-chip power-on Reset circuit drives /RESET pin to "0" for 25 to 75 msec after the power supply passes through approx. 2.2V. After the termination of the "Power-on Reset" cycle, the open-drain gate of the on-chip Power-on Reset circuit stops to drive the /RESET pin. It is required to have external pull-up register on the /RESET pin.

If it receives /RESET input from outside after the power-on sequence and while the Reset Output Disable bit in Misc Control Register is cleared to "0", it will drive the /RESET pin for 16-processor clock cycles from the falling edge of the external /RESET input. Otherwise, the /RESET pin must be kept in the active state for a period of at least 3 system clock cycles.

If there are power-on reset circuits outside of this device, drive this pin with OPEN-DRAIN type gates with pull-up resistors because /RESET signal is driven low for the period mentioned above during the Power-on sequence. If the external Power-on Reset circuit has push-pull type drivers and they drive the /RESET pin to "1" during that period, it may cause damage. In particular, when using Z84C13/C15 in the Z84013/015 socket, modification may be required on the external reset circuit.

#### Wait State Generator Unit (Z84C13/C15 Only)

The Z84C13/C15 has the enhanced feature of a Wait State Generator circuit. It is capable of generating /WAIT signals to the CPU internally. The status of the External /WAIT input line is sampled after the insertion of software wait states, except for the wait state's insertion of Interrupt Daisy Chain Wait (for this cycle, insertion of a wait state is not simple).

The Wait State Control Register can be programmed to generate multiple Wait states during different CPU cycles listed as follows.

**Memory Wait and Opcode wait.** The Wait State Generator can put 0 to 3 wait states in memory accesses. Additionally, one added wait state can be inserted during an /M1 (Opcode fetch) cycle, because /M1 cycle's timing requirement is tighter than memory Read/Write cycles. It generates wait states to the Memory Access in a specified address range, which is programmed in the Memory Wait Boundary Register.

**I/O Wait.** The Wait State generator can put 0, 2, 4 or 6 wait states in I/O accesses. Regardless of the programming of this field, no I/O wait states are inserted for accesses to on-chip peripherals.

**Interrupt Vector Wait.** During Interrupt acknowledge cycle, the Wait State Generator can insert one wait state after /IORQ goes active, to extend the time between /IORQ fall to vector fetch by CPU. It allows a slow vector response device.

**Interrupt Daisy Chain Wait and RETI sequence extension.** During Interrupt acknowledge cycle, the Wait State Generator can insert 0, 2, 4 or 6 wait states between /M1 falling to /IORQ falling edge, to extend the time required to settle daisy chain. This allows a longer daisy chain. Also, this field controls the number of wait states inserted during RETI (Return From Interrupt) cycle. If specified to insert 4 or 6 wait states during Interrupt Acknowledge cycle, Wait State Generator also inserts wait states during RETI fetch sequence. This sequence is generated with two op-code fetch cycles (Op-code is EDh followed by 4Dh). It inserts 2 or 4 wait states, respectively, if op-code followed by EDh is 4Dh. One wait state if the following op-code is not 4Dh.

#### Chip Select Signals (Z84C13/C15 Only)

The Z84C13/C15 has an enhanced feature of adding two chip select (/CS0, /CS1) pins. Both signals are originally IC test pins (ICT) on the Z84013/015. The boundary value for each Chip Select Signal is 4 bits wide, and compare with A15-A12 of the address. Each Chip Select Signal goes active when:

$$\begin{aligned} /CS0: (D3-D0 \text{ of CSBR}) \geq A15-A12 \geq 0 \\ /CS1: (D7-D4 \text{ of CSBR}) \geq A15-A12 > (D3-D0 \text{ of CSBR}) \end{aligned}$$

(Where CSBR is the contents of Chip Select Boundary Register.)

There is also a separate /CS enable bit. /CS0 is enabled on power-up with a boundary value of "F" causing /CS0 to go active for all memory accesses. /CS1 is disabled on



---

power-up, and boundary address is undefined. These features are controlled via the I/O control registers located at I/O address EEh and EFh. **Note that a glitch may be observed on these pins because address decode logic is decoding only A15-A12, without any control signals.** For more detail, please refer to the "Programming section."

#### Other functional features (Z84C13/C15 Only)

For more system design flexibility, the Z84C13/C15 has the following unique features. These features are controlled by MCR (Misc. Control Register) which is indirectly accessed via the System Control Register Pointer (SCR\_P, I/O address EEh), and System Control Data Port (SCDP, I/O address EFh). For more details, please refer to the "Programming" section.

- Clock Divide-by-one option
- Reset Output Disable
- 32-bit CRC Generation/Checking

**Clock Divide-by-One Option.** This feature is programmed through Bit D4 of MCR. Upon Power-On reset, the Clock from on-chip CGC is passed through a divide-by-two circuit. By setting this bit to one, the divide-by-two circuit is bypassed so the clock on the CLKOUT pin is equal to X'tal input. If the clock is applied to the CLKIN pin from external clock source, the status of this bit is ignored. Upon Power-on Reset, it is cleared to 0. For details, please refer to "Programming" section.

**Reset Output Disable.** This feature is programmed by Bit D3 of MCR. If this bit is cleared to "0", the /RESET pin becomes "Open-drain output" and is driven to "0" for 16-clock cycles from the falling edge of /RESET input. This feature is for the cases where /RESET is used to get out from the "HALT" state. If this bit is set to one, the on-chip reset circuit will not drive /RESET pin.

**32-bit CRC Generation/Checking.** This feature is programmed by Bit D2 of MCR. By setting this bit to one, Channel A of SIO is set to use the 32-bit CRC generator/checker instead of the original 16-bit CRC generator/checker in synchronous communication modes. The polynomial to be used in this mode is the one for the protocols

such as V.42, and is  $(X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1)$ . Upon Power-on Reset, this bit is cleared to 0.

#### Evaluation Mode

The IPC has a built evaluation (or development) mode feature which allows the users to utilize standard Z80 development systems conveniently. This mode virtually replaces the on-chip Z80 CPU with the external CPU. In this mode, the on-chip CPU is electrically disconnected from internal bus and all 3-state signals (A15-0, D7-0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1 and /RFSH; for C13/C15, /BUSREQ as well) are tri-stated, or changed to input. This allows the development system CPU to take over and use the internal I/O registers of the IPC exactly as if the CPU was on-chip.

**Z84013/015 Only.** When this signal is active, the /M1, /HALT and /RFSH pins are put in the high-impedance state. In using the Z84013/015 as an evaluator chip, the CPU is electrically disconnected (put in high-impedance state) after one machine cycle is executed with the EV signal being "1" and the /BUSREQ signal being "0". Then, on-chip resources can be accessed from the outside. /BUSACK is disconnected by an externally connected circuit.

**Z84C13/C15 Only.** If the EV pin is tied to Vcc on Power-up, the Z84C13/C15 enters into an evaluation mode. In this mode, the internal CPU is immediately disconnected from the internal bus and all 3-state signals mentioned above are tri-stated, or changed to input. Note that the /WAIT pin became the OUTPUT pin in EV mode, and the Wait State Generator generates wait states only as programmed. If the target application board has a separate wait state generator, modification of the target may be required. /BUSACK is 3-stated in this mode.

The Z84C13/C15 behaves similarly to the situation where in regular operation, the /BUSREQ signal is asserted by an external master causing all 3-state signals to be tri-stated by the Z84C13/C15 during T1 of the following machine cycle. The /BUSREQ approach was not used for the evaluation mode to avoid significant external circuitry to work around the time period before the external CPU uses the bus for Z84C13/C15 accesses.

---

## PROGRAMMING

### I/O address assignment

The IPC's on-chip peripherals' I/O addresses are listed in Table 1. They are fully decoded from A7-A0 and have no image. The registers with Z84C13/C15 located at I/O Ad-

dress EEh and EFh are the registers to control enhanced features to Z84013/015, and not assigned on Z84C013/015.

**Table 1. I/O Control Register Address**

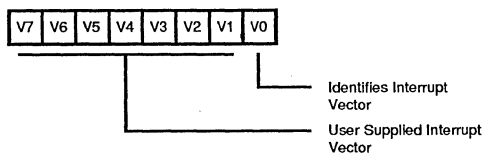
Address	Device	Channel	Register
10h	CTC	Ch 0	Control Register
11h	CTC	Ch 1	Control Register
12h	CTC	Ch 2	Control Register
13h	CTC	Ch 3	Control Register
18h	SIO	Ch. A	Data Register
19h	SIO	Ch. A	Control Register
1Ah	SIO	Ch. B	Data Register
1Bh	SIO	Ch. B	Control Register
1Ch	PIO	Port A	Data Register (Not with Z84x13)
1Dh	PIO	Port A	Command Register (Not with Z84x13)
1Eh	PIO	Port B	Data Register (Not with Z84x13)
1Fh	PIO	Port B	Command Register (Not with Z84x13)
F0h	Watch-Dog Timer		Master Register (WDTMR)
F1h	Watch-Dog Timer		Control Register (WDTCR)
F4h	Interrupt Priority Register		
EEh			System Control Register Pointer (SCRP) (Not with Z84013/015)
EFh			System Control Data Port (SCDP) (Not with Z84013/015)
Through SCRП and SCDP			Control Register 00 - Wait State Control register (WCR) Control Register 01 - Memory Wait state Boundary Register (MWBR)
			Control Register 02 - Chip Select Boundary Register (CSBR) Control Register 03 - Misc. Control Register (MCR)

## PIO REGISTERS

For more detailed information, please refer to the PIO Technical Manual. These registers are not in the Z84x13.

### Interrupt Vector Word

The PIO logic unit is designed to work with the Z80 CPU in interrupt Mode 2. The interrupt word must be programmed if interrupts are used. Bit D0 must be a zero (Figure 11).



**Figure 11. PIO Interrupt Vector Word**

### Mode Control Word

Selects the port operating mode. This word is required and is written at any time (Figure 12).

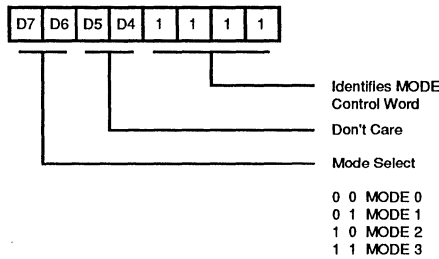


Figure 12. PIO Mode Control Word

### I/O Register Control Word

When Mode 3 is selected, the Mode Control Word is followed by the I/O Register Control Word. This word configures the I/O register, which defines which port lines are inputs or outputs. A "1" indicates input while a "0" indicates output. This word is required when in Mode 3 (Figure 13).

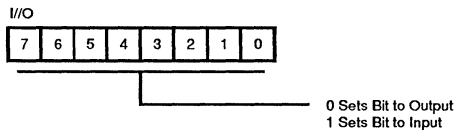
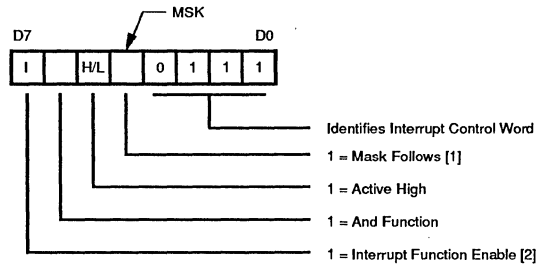


Figure 13. I/O Register Control Word

### Interrupt Control Word

In Mode 3 operation, handshake signals are not used. Interrupts are generated as a logic function of the input signal levels. The Interrupt Control Word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available: AND (if all input bits change to the active level, an interrupt is triggered), OR (if any one of the input bits change to the active logic level, an interrupt is triggered). The user can program which input bits are to be considered as part of

this logic function. Bit D6 sets the logic function, bit D5 sets the logic level, and bit D4 specifies a mask control word to follow (Figure 14).



Note:

- [1] Regardless of the operating mode, setting Bit D4 = 1 causes any pending interrupts to be cleared.
- [2] The port interrupt is not enabled until the interrupt function enable is followed by an active /M1.

Figure 14. Interrupt Control Word

### Mask Control Word

This word sets the mask control register, thus allowing any unused bits to be masked off. If any bits are to be masked, then bit D4 of the interrupt Control Word is set. When bit D4 of the interrupt Control Word is set, then the next word programmed is the Mask Control Word. To mask an input bit, the corresponding Mask Control Word bit is a "1" (Figure 15).

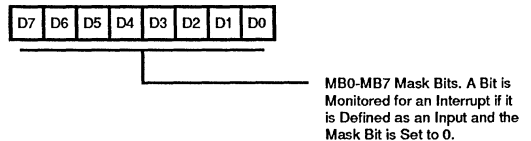


Figure 15. Mask Control Word

### Interrupt Disable Word

This word can be used to enable or disable a port's interrupts without changing the rest of the port's interrupt conditions (Figure 16).

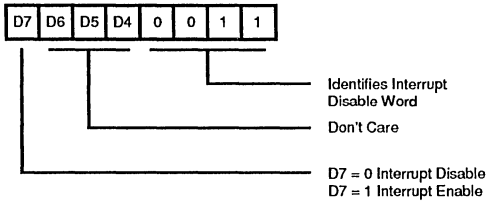


Figure 16. Interrupt Disable Word

**Bit D6. Mode Bit.** This bit selects either Timer Mode or Counter Mode.

**Bit D5. Prescaler Factor.** This bit selects the prescaler factor for use in the timer mode. Either divide-by-16 or divide-by-256 is available.

**Bit D4. Clock/Trigger Edge Selector.** This bit selects the active edge of the CLK/TRG input pulses.

**Bit D3. Timer Trigger.** This bit selects the trigger mode for timer operation. Either automatic or external trigger may be selected.

**Bit D2. Time Constant.** This bit indicates that the next word programmed is time constant data for the downcounter.

**Bit D1. Software Reset.** Writing 1 to this bit indicates a software reset operation, which stops counting activities until another time constant word is written.

**Time Constant Word**

Before a channel starts counting, it must receive a time constant word. The time constant value is anywhere between 1 and 256, with "0" being accepted as a count of 256 (Figure 18).

**CTC CONTROL REGISTERS**

For more detailed information, refer to the CTC Technical Manual.

**Channel Control Word**

This word sets the operating modes and parameters as described below. Bit D0 is a "1" to indicate that this is a Control Word (Figure 17).

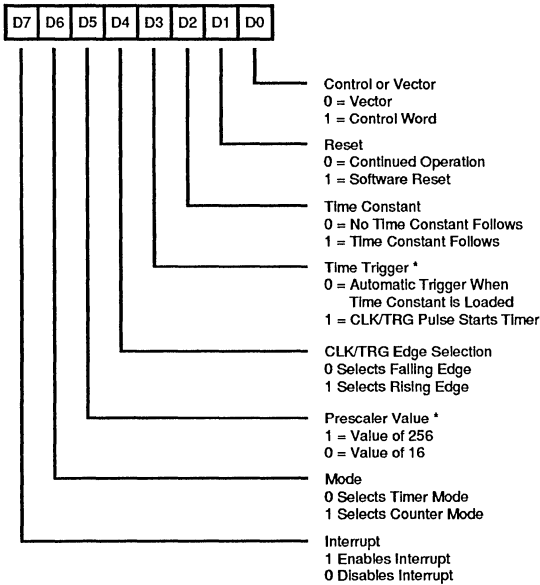


Figure 17. CTC Channel Control Word

**Bit D7. Interrupt Enable.** This bit enables the interrupt logic so that an internal INT can be generated at zero count. Interrupts are programmed in either mode and may be enabled or disabled at any time.

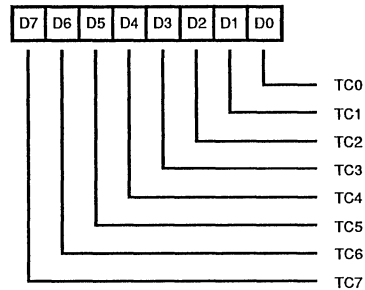


Figure 18. CTC Time Constant Word

**Interrupt Vector Word**

If one or more of the CTC channels have interrupt enabled, then the Interrupt Vector Word must be programmed. Only the five most significant bits of this word are programmed, and bit D0 must be "0". Bits D2-D1 are automatically modified by the CTC channels when it responds with an interrupt vector (Figure 19).

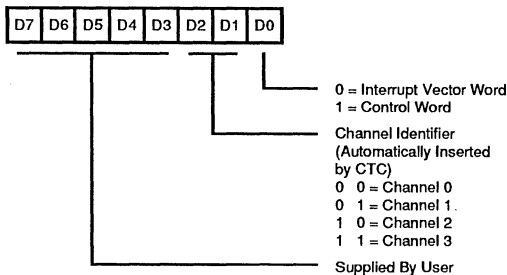
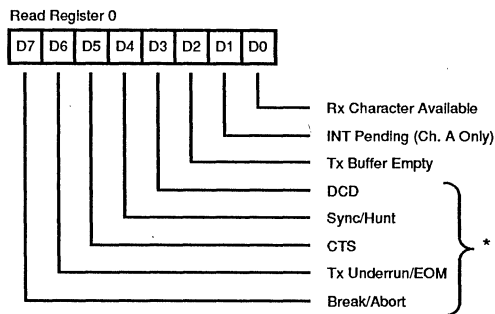


Figure 19. CTC Interrupt Vector Word

## SIO REGISTERS

For more detailed information, refer to the SIO Technical Manual.

**Read Registers.** The SIO channel B contains three read registers while channel A contains only two that are read to obtain status information. To read the contents of a register (rather than RRO), the program must first write a pointer to WRO in exactly the same manner as a write operation. The next I/O read cycle will place the contents of the selected read registers onto the data bus (Figure 20a, b, c).



\* Used With "External/Status Interrupt" Modes

Figure 20a. SIO Read Register 0

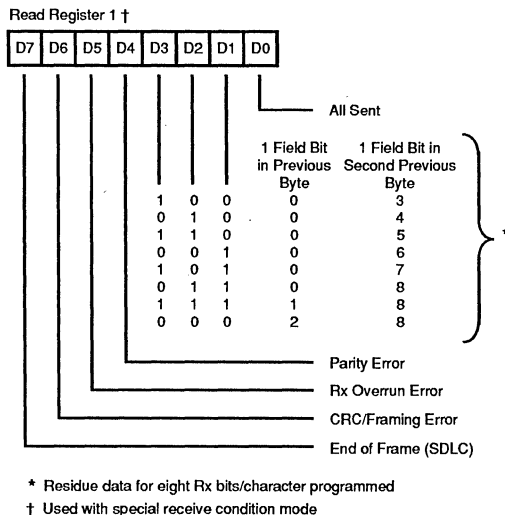


Figure 20b. SIO Read Register 1

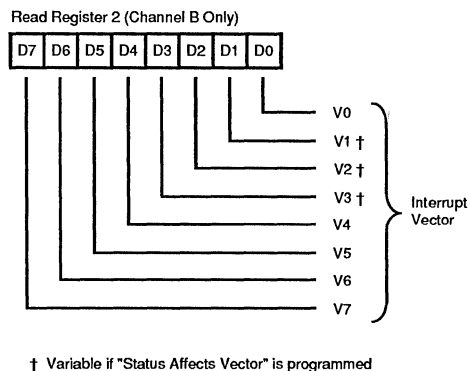
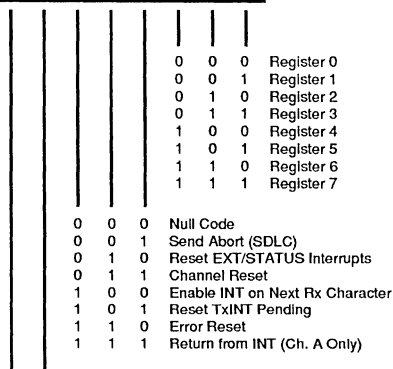
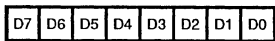


Figure 20c. SIO Read Register 2

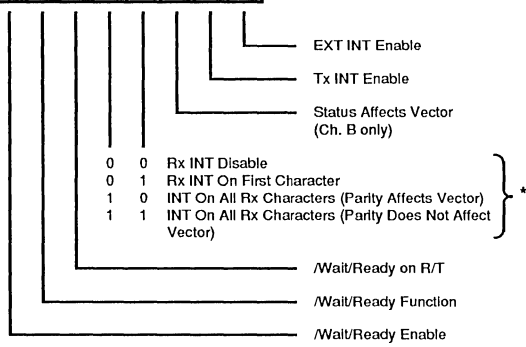
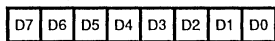
**Write Registers.** The SIO Channel B contains eight write registers while Channel A contains only seven that are programmed to configure the operating mode characteristics of each channel. With the exception of WRO, programming the write registers is a two step operation. The first operation is a pointer written to WRO which points to the selected control register. The second operation is the actual control word that is written into the register to configure the SIO channel (Figure 21).

Write Register 0



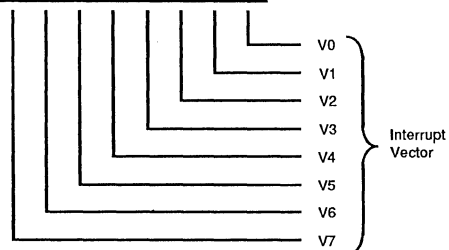
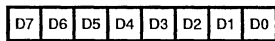
- 0 0 Null Code
- 0 1 Reset Rx CRC Checker
- 1 0 Reset Tx CRC Generator
- 1 1 Reset Tx Underrun/EOM Latch

Write Register 1



\* Or on special condition

Write Register 2 (Channel B Only)



Write Register 3

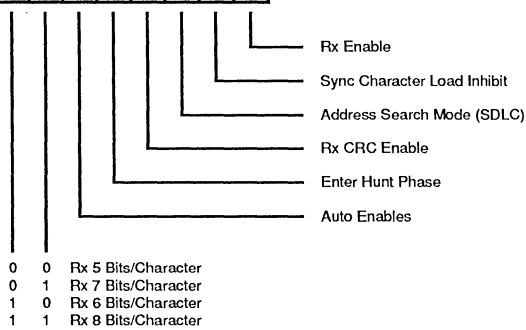
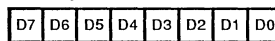
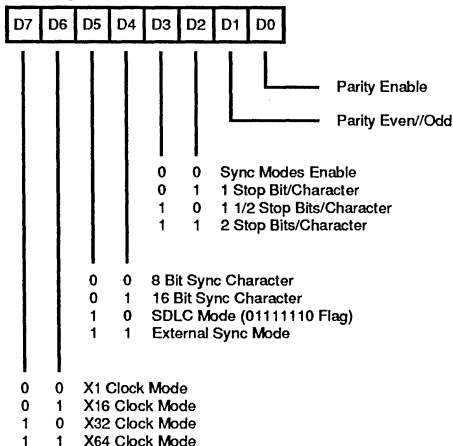
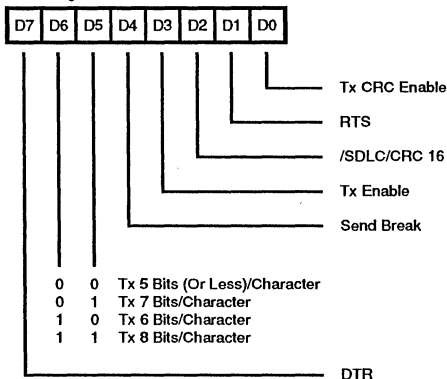


Figure 21. SIO Write Registers

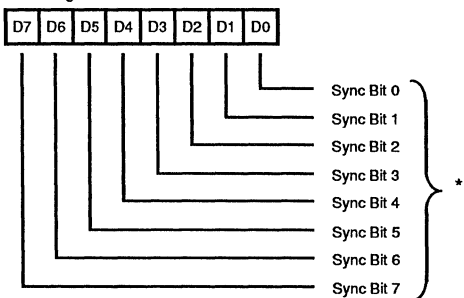
Write Register 4



Write Register 5



Write Register 6



Write Register 7

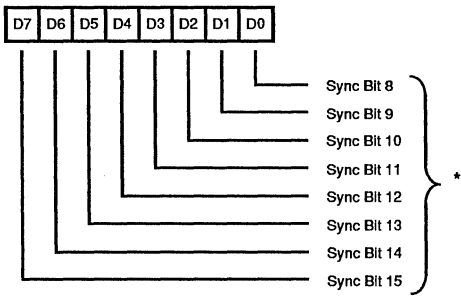


Figure 21. SIO Write Registers (Continued)

## WATCH DOG CONTROL REGISTERS

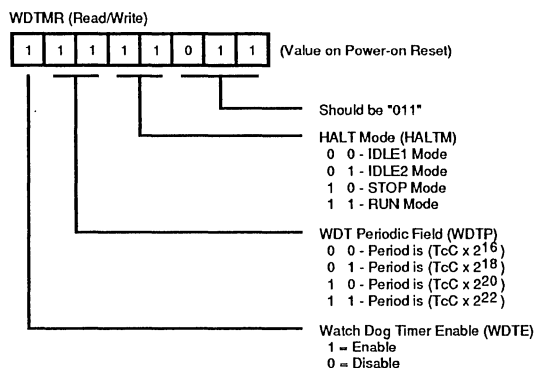
There are two registers to control Watch Dog Timer operations. These are Watch Dog Timer Master Register (WDTMR; I/O Address F0h) and the WDT Command Register (WDTCR; I/O Address F1h). Watch Dog Timer Logic has a "double key" structure to prevent the WDT disabling error, which may lead to the WDT operation to stop due to program runaway. Programming the WDT follows this procedure. Also, these registers program the power-down mode of operation. The "Second key" is needed when turning off the Watch Dog Timer.

**Enabling the WDT.** The WDT is enabled by setting the WDT Enable Bit (D7:WDTE) to "1" and the WDT Periodic field (D5,D6:WDTP) to the desired time period. These command bits are in the Watch Dog Timer Master Register (WDTMR; I/O Address F0h).

**Disabling the WDT.** The WDT is disabled by clearing WDT Enable bit (WDTE) in the WDTMR to "0" followed by writing "B1h" to the WDT Command Register (WDTCR; I/O Address F1h).

**Clearing the WDT.** The WDT can be cleared by writing "4Eh" into the WDTCR.

**Watch Dog Timer Master Register (WDTMR; I/O address F0h).** This register controls the activities of the Watch Dog Timer and selects power-down mode of operation (Figure 22).



**Figure 22. Watch Dog Timer Master Register**

**Bit D7. Watch Dog Timer Enable (WDTE).** This bit controls the activities of Watch Dog Timer. The WDT can be enabled by setting this bit to "1". To disable WDT, write "0" to this bit followed by writing "B1h" in the WDT Command Register. Watch Dog Timer Logic has a "double key" structure to prevent the WDT disabling error, which may lead to the WDT operation to stop, due to program run-away. Upon Power-on reset, this bit is set to "1" and the WDT is enabled.

**Bit D6-D5. WDT Periodic field (WDTP).** This two bit field determines the desired time period. Upon Power-on reset, this field sets to "11".

- 00 - Period is (TcC \* 2<sup>16</sup>)
- 01 - Period is (TcC \* 2<sup>18</sup>)
- 10 - Period is (TcC \* 2<sup>20</sup>)
- 11 - Period is (TcC \* 2<sup>22</sup>)

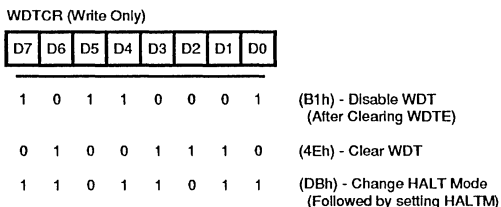
**Bit D4-D3. HALT mode (HALTM).** This two bit field specifies one of four power-down modes. To change this field, write "DBh" to the WDT command register, followed by a write to this register. For detailed descriptions of this field, please refer to the section "Mode of operations." Upon Power-on Reset, this field is set to 11, which specifies "RUN mode."

- 00 - IDLE 1 Mode
- 01 - IDLE 2 Mode
- 10 - STOP Mode
- 11 - RUN Mode

**Bit D2-D0. Reserved.** These three bits are reserved and should always be programmed as "011". A read to these bit returns "011".

**Watch Dog Timer Command Register (WDTCR; I/O address F1h).** In conjunction with the WDTMR, this register works as a "Second key" for the Watch Dog Timer. This register is write only (Figure 23).

- Write B1h after clearing WDTE to "0" - Disable WDT.
- Write 4Eh - Clear WDT.
- Write DBh followed by a write to HALTM - Change Power-down mode.



**Figure 23. Watch Dog Timer Command Register**



## INTERRUPT PRIORITY REGISTER

(INTPR; I/O address F4h)

This register (write only) is provided to determine the interrupt priority for the CTC, SIO and the PIO (Figure 24).

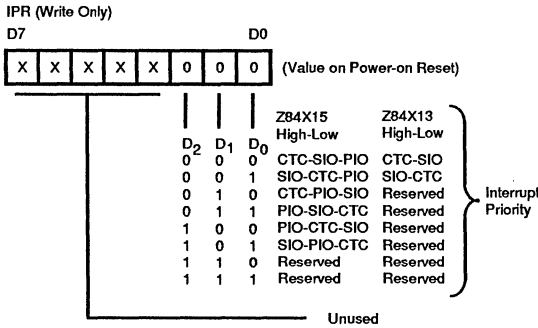


Figure 24. Interrupt Priority Register

Bit D7-D3. Unused

Bit D2-D0. This field specifies the order of the interrupt daisy chain. Upon Power-on Reset, this field is set to "000".

	Z84C15 High - Low	Z84C13 High - Low
000	CTC-SIO-PIO	CTC-SIO
001	SIO-CTC-PIO	SIO-CTC
010	CTC-PIO-SIO	Reserved
011	PIO-SIO-CTC	Reserved
100	PIO-CTC-SIO	Reserved
101	SIO-PIO-CTC	Reserved
110	Reserved	Reserved
111	Reserved	Reserved

## REGISTERS FOR SYSTEM CONFIGURATION

(The following registers are not available on Z84013/015.) There are four indirectly accessible registers to determine System configuration with the Z84C13/C15. These indirectly accessible registers are: Wait State Control Register (WCR, Control Register 00h), Memory Wait Boundary Register (MWBR, Control Register 01h), Chip Select Boundary Register (CSBR, Control Register 02h) and Misc. Control Register (MCR, Control Register 03h). To access these registers, Z84C13/C15 writes "register number to be accessed" to the System Control Register Pointer (SCRIP,

I/O address EEh), and then accesses the target register through the System Control Data Port (SCDP, I/O address EFh). The pointer which writes into SCRIP is kept until modified.

### System Control Register Pointer (SCRIP, I/O address EEh)

This register stores the pointer to access System Control Registers (WCR, MWBR, CSBR and MCR). This register is Read/Write and it holds the pointer value until modified. Upon Power-on Reset, all bits are cleared to zero. The pointer value, other than 00h to 03h is reserved and is not written. Upon Power-on Reset, this register is set to "00h" (Figure 25).

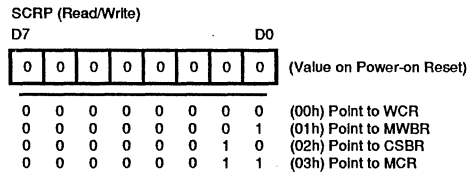


Figure 25. System Control Register Pointer

### System Control Data Port (SCDP, I/O address EFh)

This register is to access WCR, MWBR, CSBR and MCR (Figure 26).

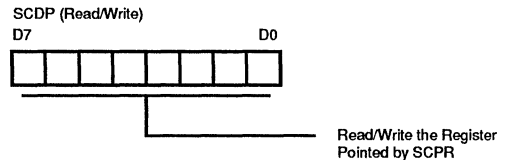


Figure 26. System Control Data Port

### Wait State Control Register (WCR, Control Register 00h)

This register can be accessed through SCDP with the pointer value 00h in SCRIP (Figure 27). To maintain compatibility with the Z84013/015, the Z84C13/C15 inserts the maximum number of wait states (set all bits of this register to one) for fifteen /M1 cycles after Power-on Reset. It automatically clears the contents of this register (move to no-wait state insertion) on the trailing edge of the 16th /M1 signal unless software has programmed a value. If automatic wait state insertion is needed, the wait state is programmed within this time period. A read to WCR during this period will return FFh, unless programmed.

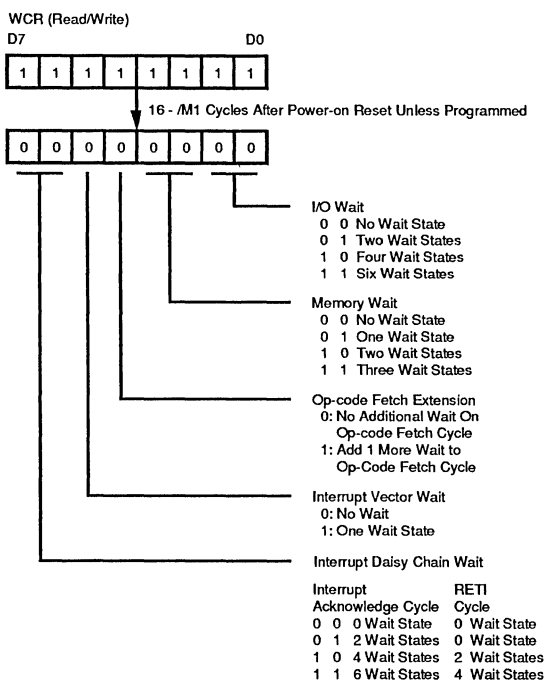


Figure 27. Wait State Control Register

This register has the following fields:

**Bit 7-6. Interrupt Daisy Chain Wait.** This 2-bit field specifies the number of wait states to be inserted during an Interrupt Daisy Chain settle period of the Interrupt Acknowledge cycle, which is /IORQ falls after the settling period from /M1 going active "0". Also, this field controls the number of wait states inserted during the RETI (Return From Interrupt) cycle. If specified to insert 4 or 6 wait states during Interrupt Acknowledge cycle, the Wait state generator also inserts wait states during RETI fetch sequence. This sequence is formed with two op-code fetch cycles (Op-code is EDh followed by 4Dh). It inserts 1 wait state if op-code followed by EDh is NOT 4Dh, and inserts 2 or 4 wait states, respectively, if the following op-code is 4Dh.

Interrupt Acknowledge	RETI cycle
00 - No Wait states	No Wait states
01 - 2 Wait states	No Wait states
10 - 4 Wait states	2 Wait states
11 - 6 Wait states	4 Wait states

For fifteen /M1 cycles from Power-on Reset, bits 7-6 are set to "11". They clear to "00" on the trailing edge of the 16th /M1 signal unless programmed.

**Bit 5. Interrupt Vector Wait.** While this bit is set to one, the wait state generator inserts one wait state after the /IORQ signal goes active during the Interrupt acknowledge cycle. This gives more time for the vector read cycle. While this bit is cleared to zero, no wait state is inserted (standard timing). For fifteen /M1 cycles from Power-on Reset, this bit is set to "1", then cleared to "0" on the trailing edge of the 16th /M1 signal, unless programmed.

**Bit 4. Opcode Fetch Extension.** If this bit is set to "1", one additional wait state is inserted during the Op-code fetch cycle in addition to the number of wait states programmed in the Memory Wait field. For fifteen /M1 cycles from Power-on Reset, this bit is set to "1", then cleared to "0" on the trailing edge of the 16th /M1 signal, unless programmed.

**Bit 3-2. Memory Wait States.** This 2-bit field specifies the number of wait states to be inserted during memory Read/Write transactions.

- 00 - No Wait states
- 01 - 1 Wait states
- 10 - 2 Wait states
- 11 - 3 Wait states

For fifteen /M1 cycles from Power-on Reset, these bits are set to "11", then cleared to "00" on the trailing edge of the 16th /M1 signal, unless programmed.

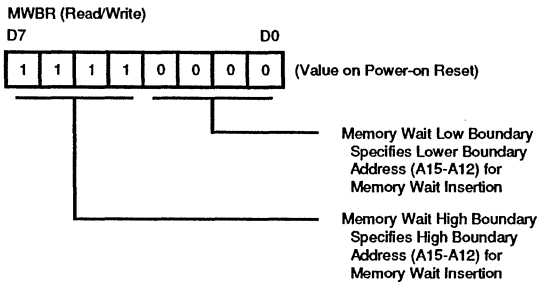
**Bit 1-0. I/O Wait states.** This 2-bit field specifies the number of wait states to be inserted during I/O transactions.

- 00 - No Wait states
- 01 - 2 Wait states
- 10 - 4 Wait states
- 11 - 6 Wait states

For fifteen /M1 cycles from Power-on Reset, these bits are set to "11", then cleared to "00" on the trailing edge of the 16th /M1 signal, unless programmed. For the accesses to the on-chip I/O registers, no Wait states are inserted regardless of the programming of this field.

### Memory Wait Boundary Register (MWBR, Control Register 01h)

This register specifies the address range to insert memory wait states. When accessed memory addresses are within this range, the Memory Wait State generator inserts Memory Wait States specified in the Memory Wait field of WCR (Figure 28).



**Figure 28. Memory Wait Boundary Register**

**Bit D7-D4. Memory Wait High Boundary.** This field specifies A15-A12 of the upper address boundary for Memory Wait.

**Bit D3-D0. Memory Wait Low Boundary.** This field specifies A15-12 of the lower address boundary for Memory Wait.

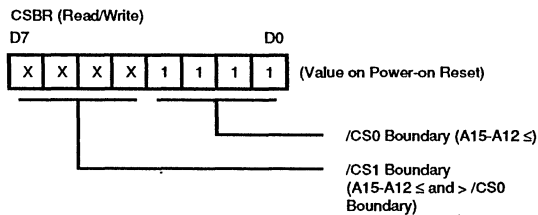
Memory Wait states are inserted for the address range:

$$(D7-D4 \text{ of MWBR}) \geq A15-A12 \geq (D3-D0 \text{ of MWBR})$$

This register is set to "F0h" on Power-on Reset, which specifies the address range for Memory Wait as "0000h to FFFFh".

**Chip Select Boundary Register (CSBR, Control Register 02h)**

This register specifies the address range for each chip select signal. When accessed memory addresses are within this range, chip select signals are active (Figure 29).



**Figure 29. Chip Select Boundary Register**

**D7-D4. /CS1 Boundary Address.** These bits specify the boundary address range for /CS1. The bit values are ignored on power-up as the /CS1 enable bit is off. The /CS1 is asserted if the address lines A15-12 have an address value greater than the programmed value for /CS0, and less than or equal to the programmed value in these bits.

**D3-D0. /CS0 Boundary Address.** These bits specify the boundary address range for /CS0. /CS0 is asserted if the address lines A15-12 have an address value less than or equal to the programmed boundary value. The /CS0 enable bit in the MCR must be set to 1. Upon Power-up reset, these bits come up as all 1's so that /CS0 is asserted for all addresses.

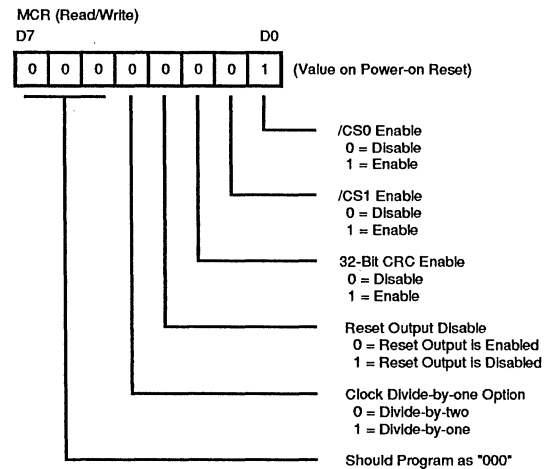
Chip Select signals are active for the address range:

$$\begin{aligned} /CS0: (D3-D0 \text{ of CSBR}) \geq A15-A12 \geq 0 \\ /CS1: (D7-D4 \text{ of CSBR}) \geq A15-A12 > (D3-D0 \text{ of CSBR}) \end{aligned}$$

This register is set to "xxxx1111b" on Power-on Reset, which specifies the address range of /CS0 for "0000h to FFFFh" (all Memory location) and /CS1 "undefined."

**Misc Control Register (MCR, Control Register 03h)**

This register specifies miscellaneous options on this device (Figure 30).



**Figure 30. Misc Control Register**

**Bit D7-D5. Reserved.** These three bits are reserved and are always programmed as "000".

**Bit D4. Clock Divide-by-one option.** "0"-Disable, "1"-enable. On-chip CGC unit has divide-by-two circuit. By setting this bit to one, this circuit is bypassed and CLKOUT is equal to X'tal oscillator frequency (or external clock input on the XTAL1 pin). This bit has no effect when the on-chip CGC unit is not in use and the external system clock is fed from CLKIN pin. Upon Power-on Reset, this bit is cleared to 0 and the clock is divided by two.

**Bit D3. Reset Output Disable.** "0"-Reset output is enabled, "1"-Reset output is disabled. This bit controls the /RESET signal and is driven out when reset input is used to take the Z84C13/C15 out of the "Halt" state. The reset pulse is driven out for 16-clock cycles from the falling edge of /RESET input, unless this bit is set. Upon Power-on reset, this bit is cleared to 0.

**Bit D2. 32-Bit CRC enable.** "0"-Normal mode (16-bit CRC) "1"-32-bit CRC generation/Checking is enabled on SIO Channel A. This bit determines if the 32-bit CRC feature is enabled on Channel A of the SIO. If this bit is 0, the SIO is in a normal mode of operation. If this bit is set to 1, a normal CRC generator/checker is replaced with a 32-bit CRC generator/checker. Upon Power-on Reset, this bit is clear to "0".

**Bit D1. /CS1 Enable.** "0"-Disable, "1"-Enable. This bit enables /CS1 output. While this bit is "0", /CS1 is forced to "1". While this bit is "1", /CS1 carries the address range specified in the CSBR. Upon Power-on Reset, this bit is cleared to "0".

**Bit D0. /CS0 Enable.** "0"-Disable, "1"-Enable. This bit enables /CS0 output. While this bit is "0", /CS1 pin is forced

to "1". While this bit is "1", the /CS0 carries address range specified in the CSBR. Upon Power-on Reset, this bit is set to "1".

#### Operation modes

There are four kinds of operation modes available for the IPC in connection with clock generation; RUN Mode, IDLE1/2 Modes and STOP Mode.

The Operation mode is effective when the HALT instruction is executed. Restart of the MPU from the stopped state under IDLE1/2 Mode or STOP mode is affected by inputting either /RESET or interrupt (/NMI or /INT). The mode selection of these power-down modes is made by programming the HALTM field (Bit D4-3) of WDTMR.

#### Setting Halt Mode

Duplicate control is provided to prevent the stopping of the WDT operation caused by the halt mode setting an error due to program runaway. As described in the programming section, changing the Halt Mode field of WDTMR is in two steps. First, write "DBh" to WDTCR followed by a write to the WDTMR with the value in HALTM. Table 2 has descriptions of each mode, and Table 3 has device status in the Halt state.

**Table 2. Power-down Modes**  
(When using on-chip CGC unit; CLKOUT and CLKIN are tied together)

Operation Mode	WDTMR Bit D4	Bit D3	Description at HALT State
RUN Mode	1	1	The IPC continues the operation and continuously supplies a clock to the outside.
IDLE1 Mode	0	0	The internal oscillator's operation is continued. Clock output (CLKOUT) as well as internal clock to the CPU, PIO, SIO, CTC and the Watch Dog Timer is stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.
IDLE2 Mode	0	1	The internal oscillator and the CTC's operation continues and supplies clock to the outside on the CLKOUT pin continuously. But the internal clock to the CPU, PIO, SIO and the Watch Dog Timer is stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.
STOP Mode	1	0	All operations of the internal oscillator, clock (CLK) output, internal clock to the CPU, PIO, CTC, SIO and the Watch Dog Timer are stopped at "0" level of T4 state in the halt instruction operation code fetch cycle.

**Table 3. Device status in Halt state**  
(When using on-chip CGC unit; CLKOUT and CLKIN are tied together)

Mode	CGC	CPU	CTC	PIO	SIO	WDT	CLKOUT
IDLE1	O	X	X	X	X	X	X
IDLE2	O	X	O	X	X	X	O
STOP	X	X	X	X	X	X	X
RUN	O	O	O	O	O	O	O

O: Operating  
X: Stop

All of the operating modes listed here are valid with crystal input (Crystal connected between XTAL1/2 or external clock input on XTAL1). For the external clock on the CLKIN pin, only the IDLE2 and RUN modes are applicable.

## TIMING

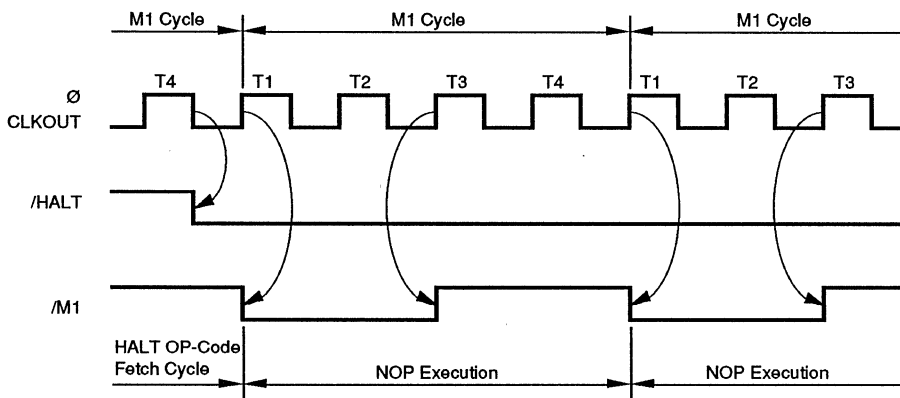
### Basic Timing

The basic timing is explained here with emphasis placed on the halt function relative to the clock generator. The following items are identical to those for the Z84C00. Refer to the data sheet for the Z84C00.

- Operation code fetch cycle
- Memory Read/Write operation
- Input/Output operation
- Bus request/acknowledge operation
- Maskable interrupt request operation
- Non-Maskable interrupt request operation
- Reset Operation

**Operation When HALT Instruction is Executed.** When the CPU fetches a halt instruction in the operation code fetch cycle, /HALT goes active (Low) in synch with the falling edge of T4 state before the peripheral LSI and CPU stops the operation. After this, the system clock generation differs depending upon the operation mode (RUN Mode, IDLE1/2 Mode or STOP Mode). If the internal system clock is running, the CPU continues to execute NOP instruction even in the halt state.

**RUN Mode (HALTM = 11).** Shown in Figure 31 is the basic timing when the halt instruction is executed in RUN Mode.

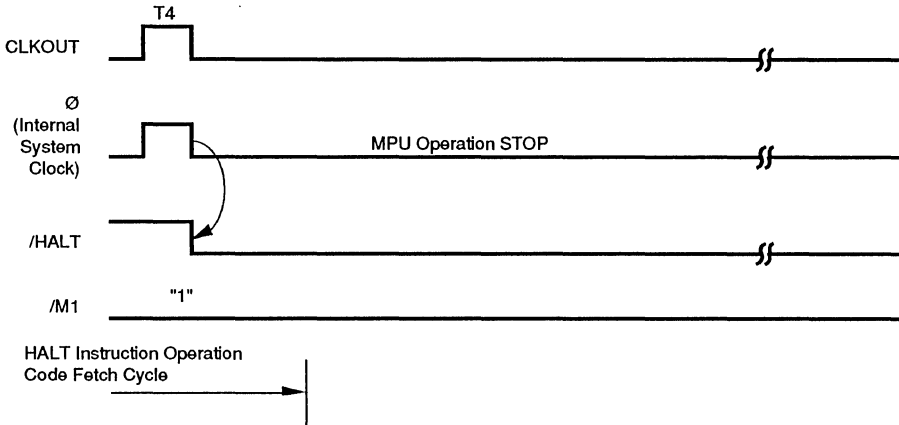


**Figure 31. Timing of RUN Mode**  
(at Halt Instruction Command Execution)

In RUN Mode, output from the CGC unit (CLKOUT) is not stopped and the internal system clock ( $\emptyset$ ) continues even after the halt instruction is executed. Therefore, until the halt state is released by the interrupt signal (/NMI or /INT)

or /RESET signal, MPU continues to execute HALT instructions (internally executing NOP instructions).

**IDLE1 Mode (HALTM=00).** Shown in Figure 32 is the basic timing when the halt instruction is executed in IDLE1 Mode.

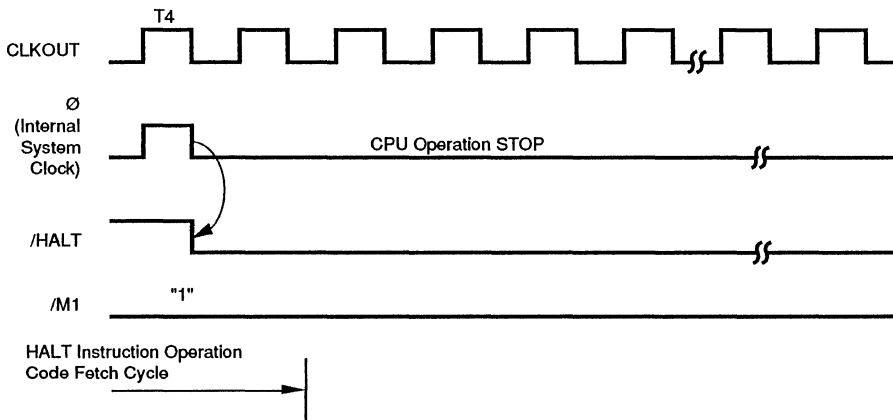


**Figure 32. IDLE1 Mode Timing**  
(At Halt Instruction Execution)

In IDLE1 Mode, the internal oscillator continues to operate, but clock output (CLKOUT) is stopped at T4 Low state of HALT instruction execution. Then all components in the MPU stop their operation. This mode is not supported

when the CGC unit is inactive and the external clock is fed from CLKIN pin; CLKOUT should be connected to CLKIN.

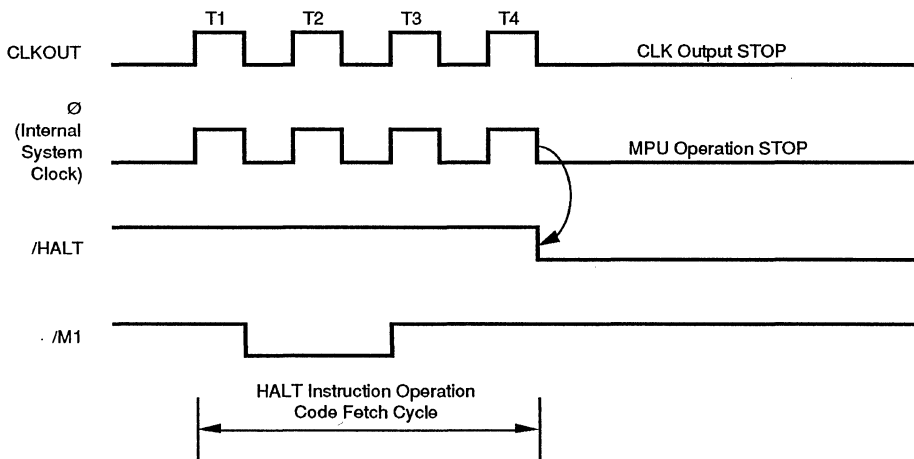
**IDLE2 Mode (HALTM=01).** Shown in Figure 33 is the basic timing when the halt instruction is executed in IDLE2 Mode.



**Figure 33. IDLE2 Mode Timing**  
(At Halt Instruction Execution)

In IDLE2 Mode, the internal oscillator and clock output (CLKOUT) continue to operate. The internal system clock, fed from CLKIN to the components other than CTC is stopped at the T4 Low state of HALT instruction execution.

**STOP Mode (HALTM=10).** Shown in Figure 34 is the basic timing when the halt instruction is executed in STOP Mode.



**Figure 34. STOP Mode Timing**  
(At Halt Instruction Execution)

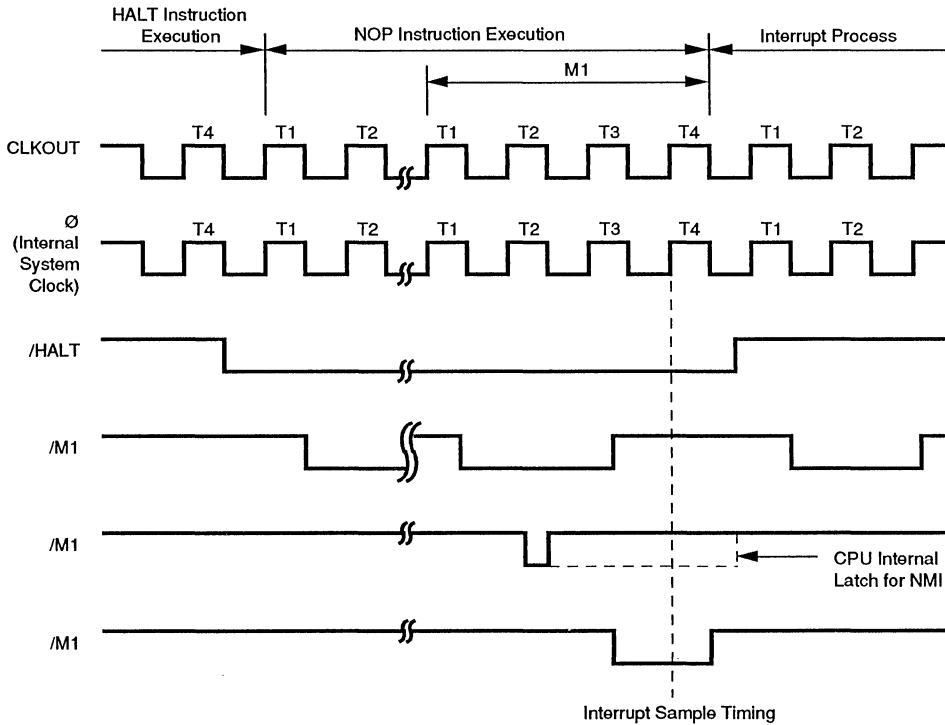
In STOP Mode, the on-chip CGC unit is stopped at T4 Low state of HALT instruction execution. Therefore, clock output (CLKOUT), operation of Watch Dog Timer, CPU, PIO, CTC, SIO are stopped.

**Release from Halt State.** The halt state of the CPU is released when "0" is input to the /RESET signal and the MPU is reset or an interrupt request is accepted. An interrupt request signal is sampled at the leading edge of the last clock cycle (T4 state) of NOP instruction. In case of the maskable interrupt, interrupt will be accepted by an active /INT signal ("0" level). Also, the interrupt enable flip-

flop is set to "1". The accepted interrupt process is started from the next cycle.

Further, when the internal system clock is stopped (IDLE1/2 Mode, STOP Mode), it is necessary first to restart the internal system clock. The internal system clock is restarted when /RESET or interrupt signal (/NMI or /INT) is asserted.

**RUN Mode (HALTM=11).** The halt release operation is enabled by interrupt request in RUN Mode (Figure 35).



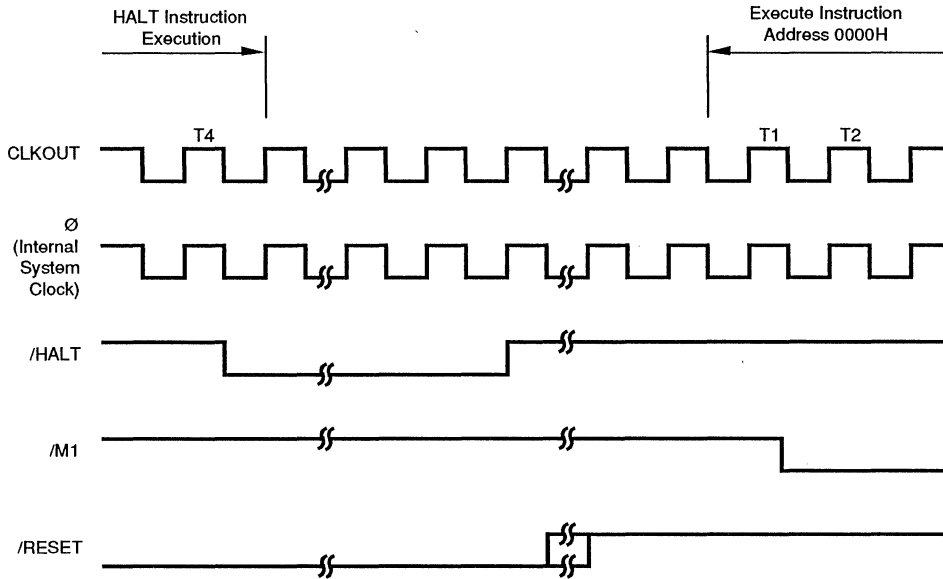
**Figure 35. Halt Release Operation Timing  
By Interrupt Request Signal in RUN Mode**

In RUN Mode the internal system clock is not stopped. If the interrupt signal is recognized on the rising clock edge of T4 of the continued NOP instruction, CPU will execute the interrupt process from the next cycle.

The halt release resets CPU in RUN Mode (Figure 36). After reset, CPU will execute an instruction starting from address 0000H. However, in order to reset the CPU it is

necessary to keep /RESET signal at "0" for at least 3 system clock cycles. (For Z84C13/C15: 3 clock cycles if Reset output is disabled.) In addition, if /RESET signal becomes "1", after the dummy cycle for at least two T states, CPU executes an instruction from address 0000H.





**Figure 36. Halt Release Operation Timing By Reset in RUN Mode**

IDLE1 Mode (HALTM=00), IDLE2 Mode (HALTM=01). The halt release operation by interrupt signal in IDLE1 Mode is shown in Figure 37 (a) and in IDLE2 Mode in Figure 37 (b).

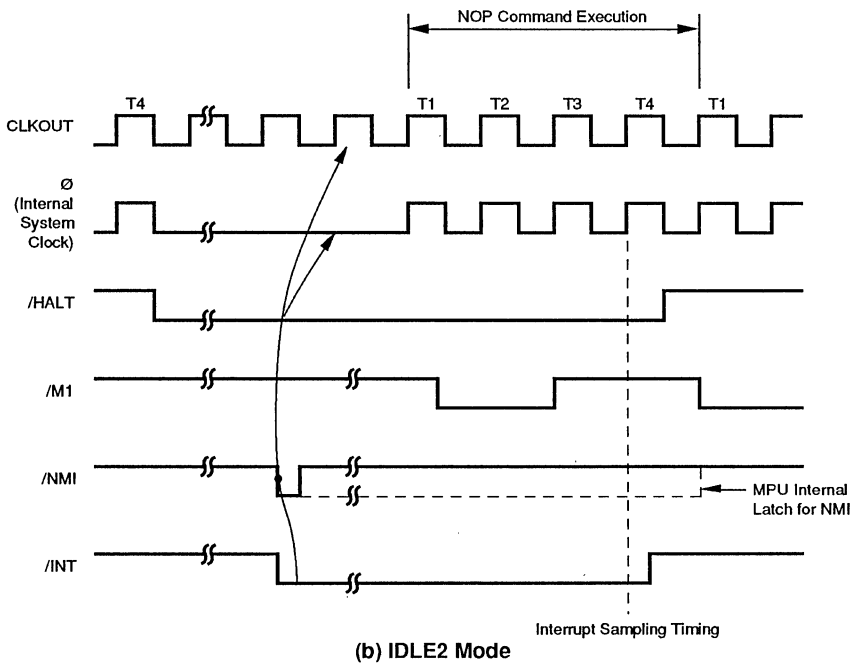
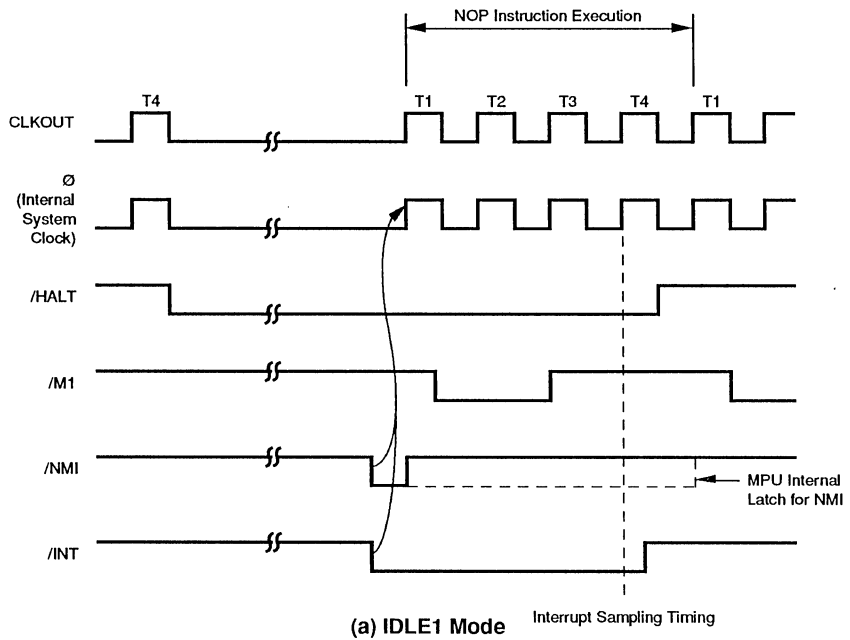


Figure 37. Halt Release Operation Timing By Interrupt Request Signal in IDLE1/2 Mode

When receiving /NMI or /INT signals, the stopped internal system clock starts to feed. In IDLE1 Mode, the IPC starts clock output on CLKOUT at the same time.

The operation stop of CPU in IDLE2 mode is taking place at "0" level during T4 state in the halt instruction op-code fetch cycle. Therefore, after being restarted by the interrupt signal, CPU executes one NOP instruction and samples an interrupt signal at the rise of T4 state during the execution of this NOP instruction, and executes the interrupt process from next cycle.

If no interrupt signal is accepted during the execution of the first NOP instruction after the internal system clock is restarted, CPU is not released from the halt state. It is placed in IDLE1/2 Mode again at "0" level during T4 state of the NOP instruction, stopping the internal system clock. If /INT signal is not at "0" level at the rise of T4 state, no interrupt request is accepted.

The halt release operation resets the IPC in IDLE1 Mode (Figure 38a) and in IDLE2 Mode (Figure 38b).

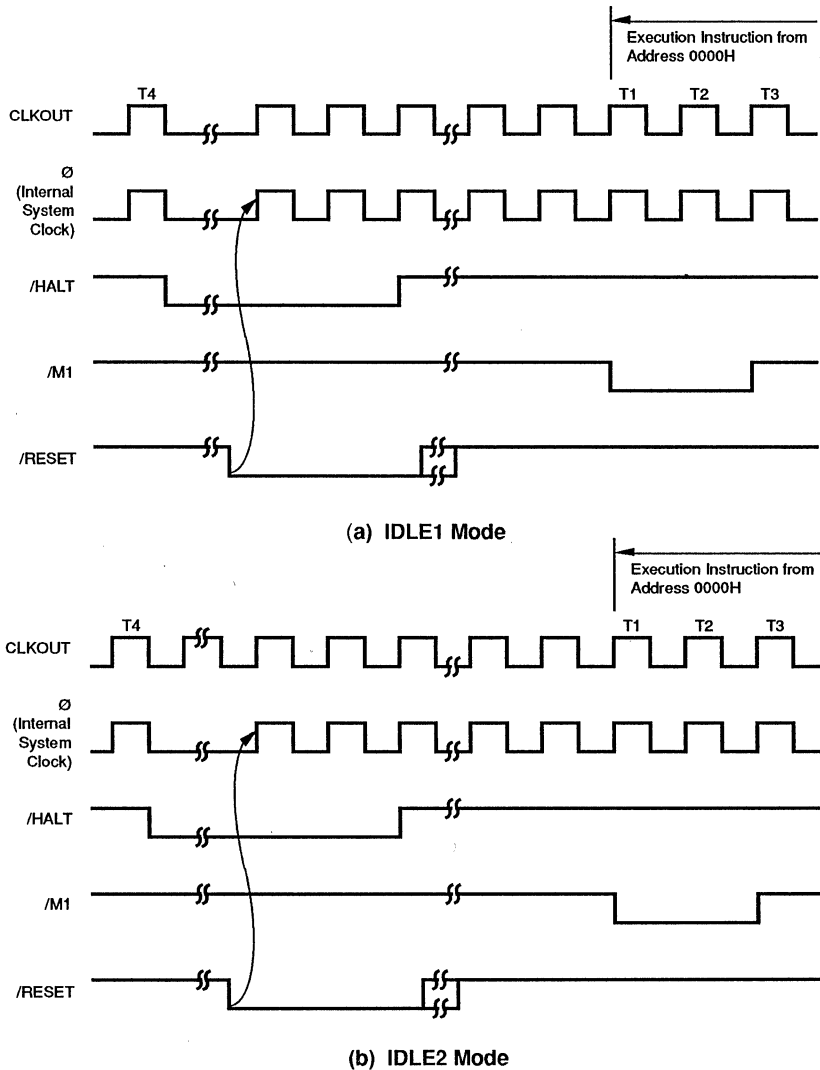
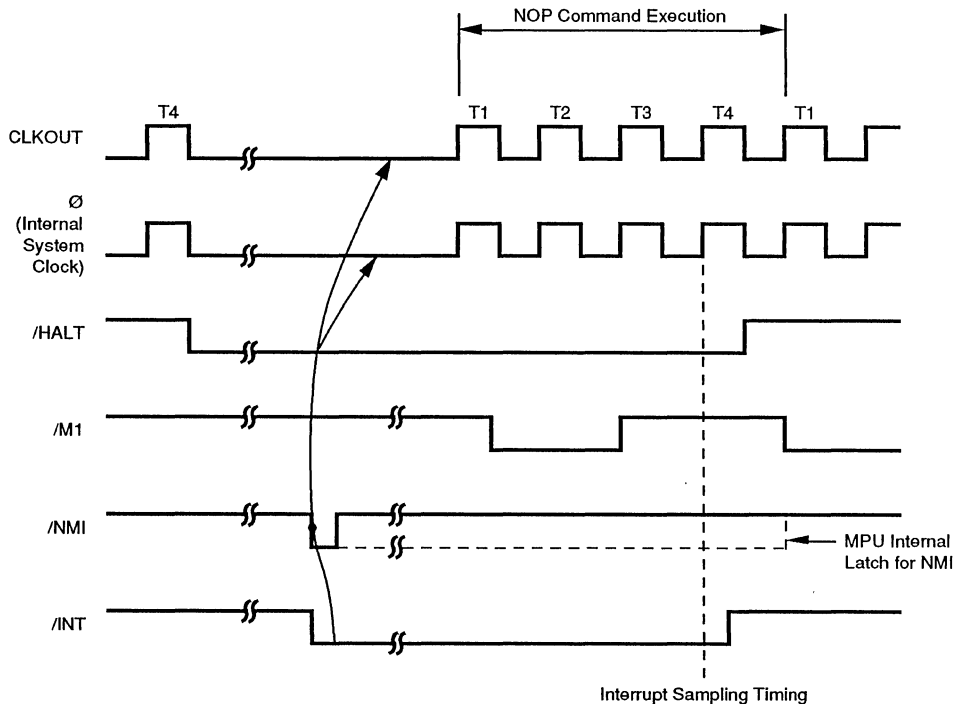


Figure 38. Halt Release Operation Timing By Reset in IDLE1/2 Mode

When /RESET signal at "0" level is input into the IPC, the internal system clock is restarted and the IPC will execute an instruction stored in address 0000H.

**Halt release in STOP Mode (HALTM=10) by interrupt.** The halt release operation by interrupt signal in STOP Mode is shown in Figure 39.

At time of /RESET signal input, it is necessary to take the same care as that in resetting the IPC in RUN Mode.



**Figure 39. Halt Release Operation Timing By Interrupt Request Signal in STOP Mode**

When the IPC receives an interrupt signal, the internal oscillator is restarted. To obtain stabilized oscillation, CLKOUT (and the internal system clock) are started after a start-up time of  $(2^{14}+2.5)$  TcC (TcC: Clock Cycle) by the internal counter.

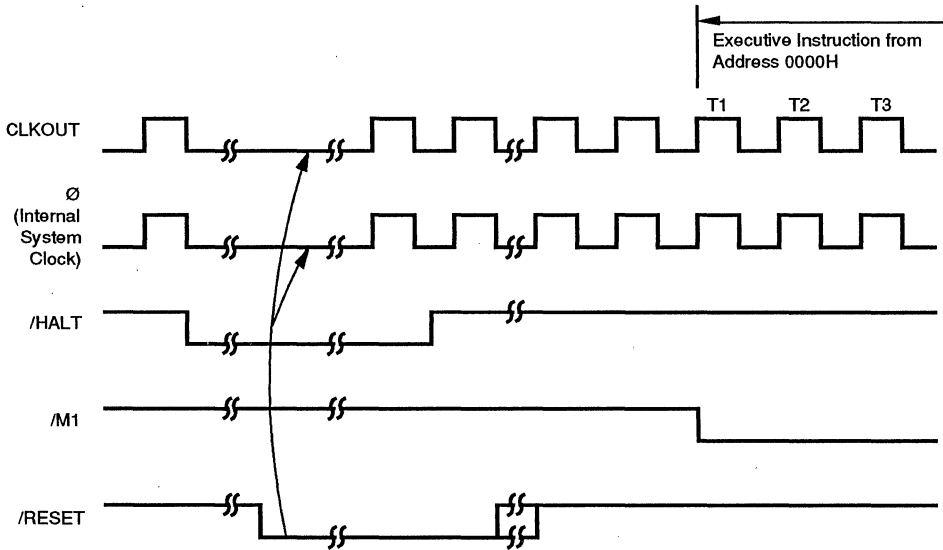
During interrupt signal input, it is necessary to take the same care as the interrupt signal input in IDLE1/2 Mode.

CPU executes one NOP instruction after the internal system clock is restarted. At the same time, it samples an interrupt signal at the rise of T4 state during the execution of this NOP instruction. If the interrupt signal is accepted, CPU executes the interrupt process operation from the next cycle.

**Halt release in STOP Mode (HALTM=10) by /RESET.** When /RESET at "0" level is input into the IPC, the internal oscillator is restarted. However, the internal clock counter for warm-up does not operate. Therefore, the operation is not carried out properly due to unstable clock oscillation. It is necessary to hold /RESET at "0" level for sufficient time. The halt release operation by the IPC resetting in STOP Mode is shown in Figure 40.

**Z84C13/C15 Only.** The /RESET pulse is stretched to a minimum of 16 cycles and driven out of the Z84C13/C15 on the /RESET pin if Reset output is enabled (bit D3 of MCR is cleared to "0"). Setting bit D3 disables the driving out of

/RESET. The values in the control registers (WDTMR, SCRIP, WCR, MWBR, CSBR and MCR) are initialized to the default value on /RESET.



**Figure 40. Halt Release Operation Timing By Reset in STOP Mode**

**Start-up Time at Time of Restart (STOP Mode).** When the MPU is released from the halt state by accepting an interrupt request, it executes an interrupt service routine. Therefore, when an interrupt request is accepted, it starts generating clock on the CLKOUT pin, after a start-up time, by the internal counter  $[(2^{14}+2.5) T_{cC} (T_{cC}: \text{Clock Cycle})]$ . This obtains a stabilized oscillation for operation.

Further, in case of restart by the /RESET signal, the internal counter does not operate.

**Evaluation operation.** Each of the CPU signals (A15-0, D7-0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1, /RFSH) can be 3-stated by activating the EV pin. The Z84C13/C15 enhances the counter part by eliminating the requirement of /BUSREQ to go active.

**Instruction set.** The instruction set of the IPC is the same for the Z84C00. For details, refer to the data sheet of the Z84C00 Technical Manual.

## AC TIMING

The following section describes the timing of the IPC. The numbers appearing in the figures refer to the parameters on Table A - F.

### CPU Timing

Parameters referenced in Figure 41 through Figure 48 appear in Table A.

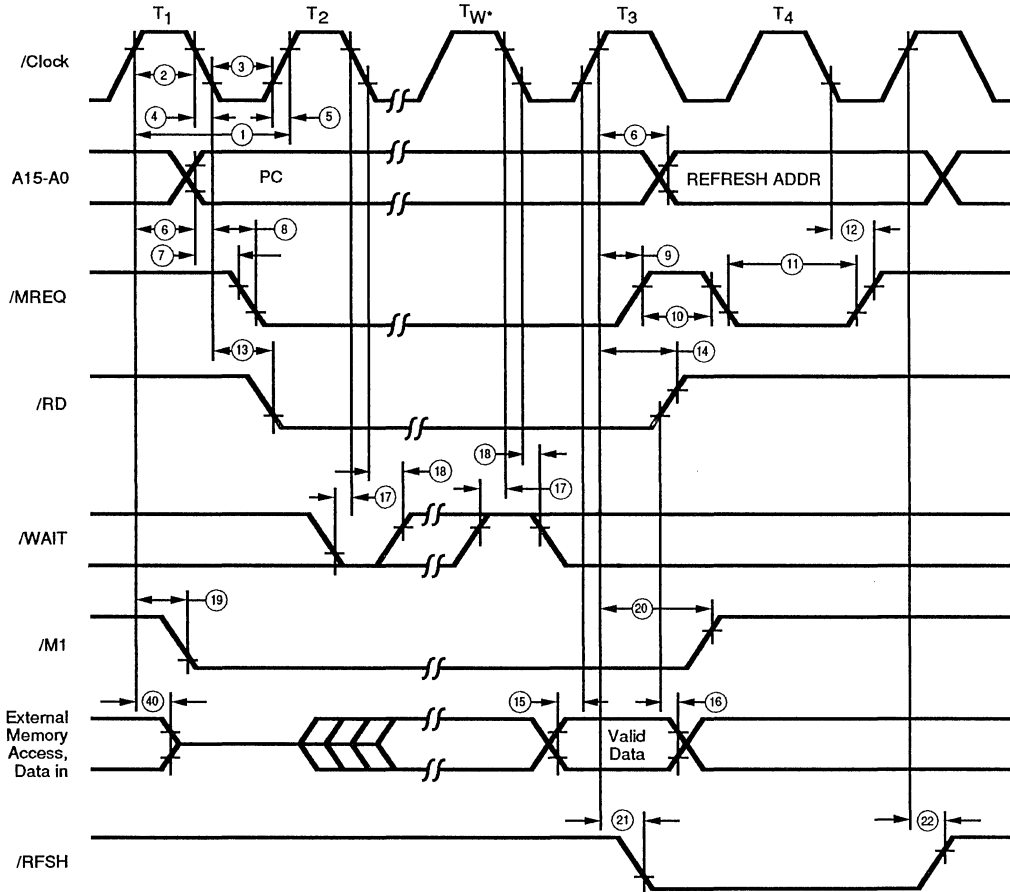
The IPC's CPU executes instructions by proceeding through the following specific sequence of operations:

Memory read or write  
I/O device read or write  
Interrupt acknowledge

The basic clock period is referred to as a Time or Cycle and three or more T cycles make up a machine cycle (e.g., M1, M2 or M3). Machine cycles are extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

**Instruction Op-code Fetch.** The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (Figure 41). Approximately one-half clock cycle later, /MREQ goes active. When active, /RD indicates that the memory data can be enabled onto the CPU data bus.

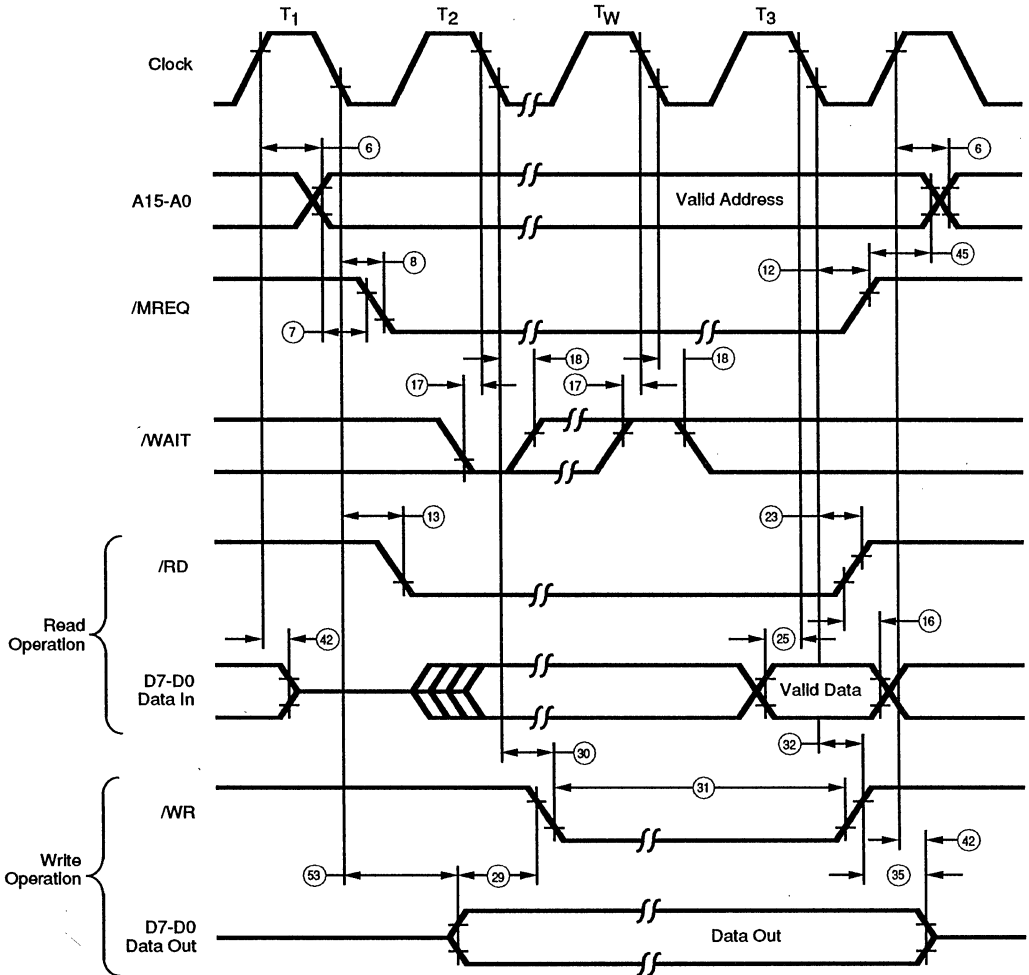
The CPU samples the /WAIT input with the falling edge of clock state T2. During clock states T3 and T4 of an M1 cycle later, dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction.



**Figure 41. Instruction Op-code Fetch**  
(See Table A)

**Memory Read or Write Cycles.** Figure 42 shows the timing of memory read or write cycles other than an Op-code fetch (/M1) cycle. The /MREQ and /RD signals function like the Op-code fetch cycle.

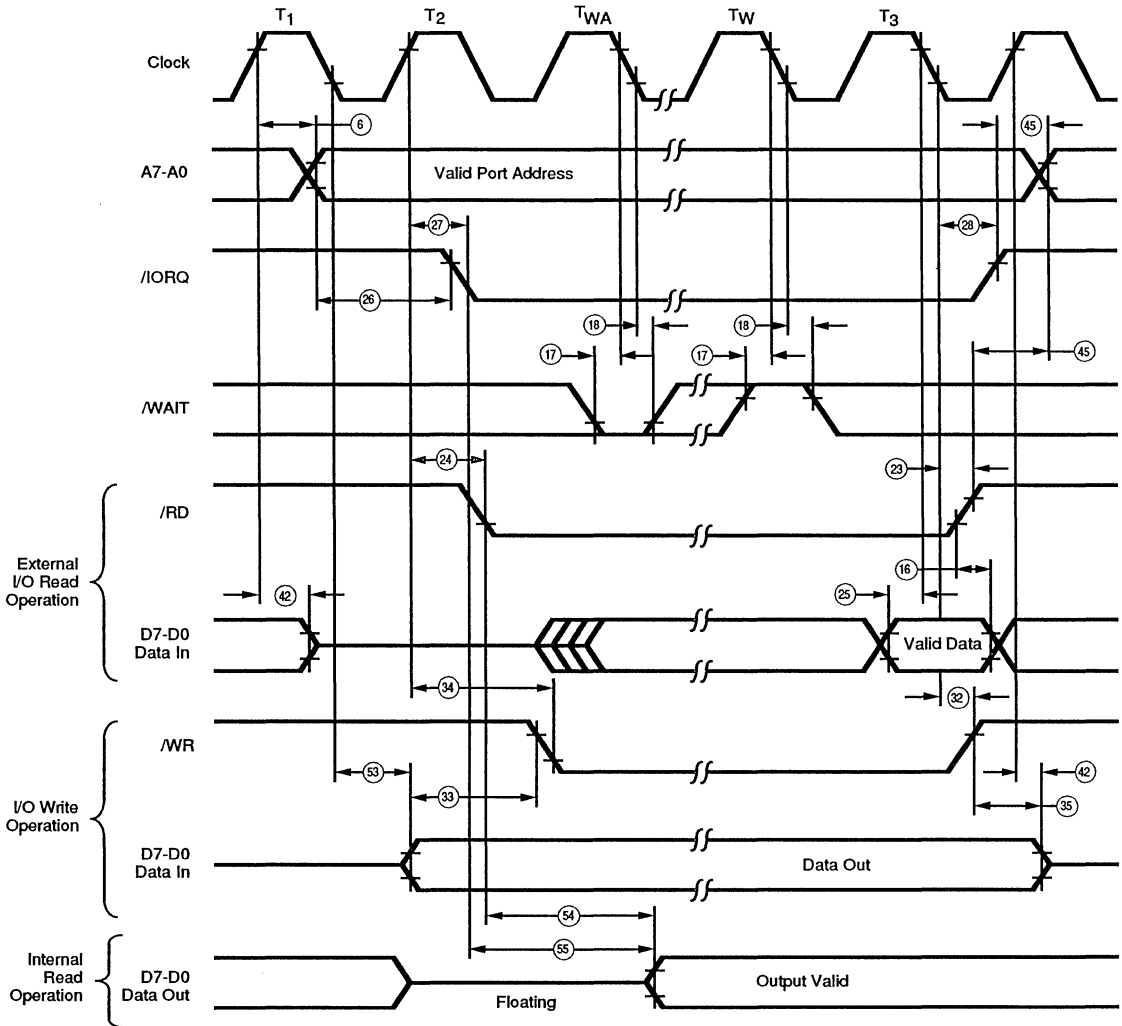
In a memory write cycle, /MREQ also becomes active when the Address Bus is stable. The /WR line is active when the Data Bus is stable, so that it can be used directly as an R/W pulse to most semiconductor memories.



**Figure 42. Memory Read or Write Cycle**  
(See Table A)

**Input or Output Cycles.** Figure 43 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single Wait state ( $T_{WA}$ ). This extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.

When the CPU is accessing the on-chip I/O registers (PIO, CTC, SIO and system control registers), the data from/to these registers also appears on the data bus, or data bus is output during I/O cycle.



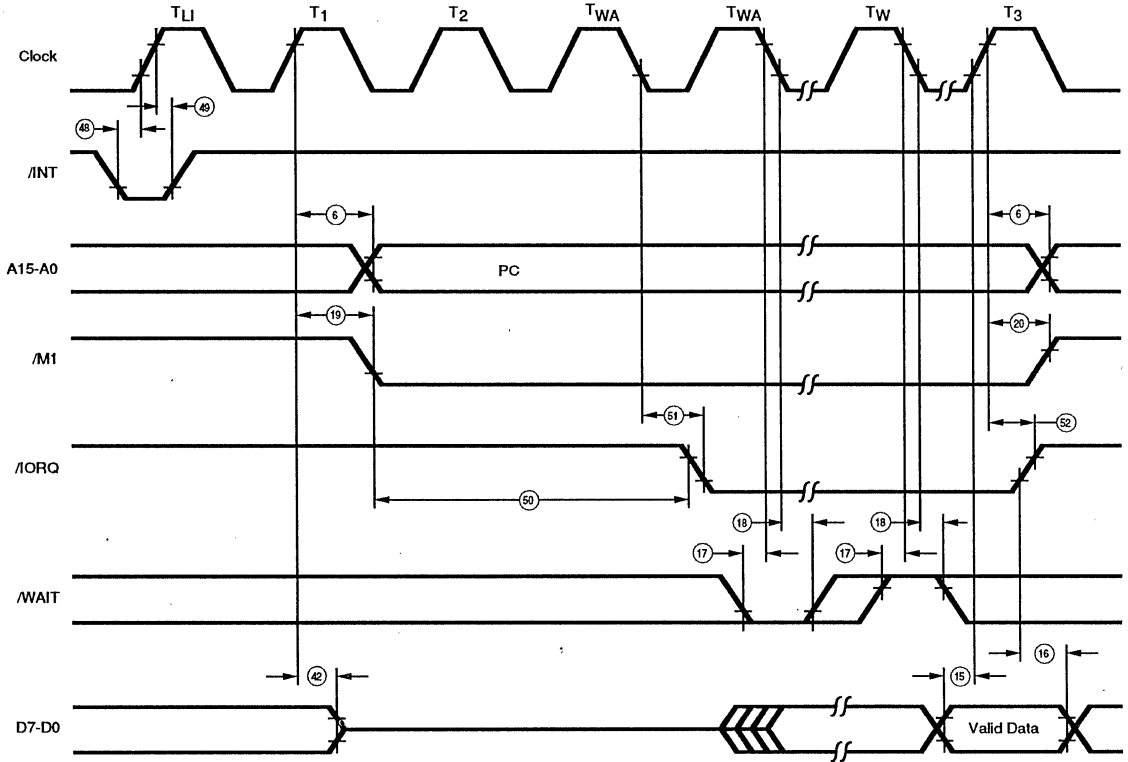
Note:  $T_{WA}$  = One wait cycle automatically inserted by CPU

**Figure 43. Input or Output Cycle**  
(See Table A)



**Interrupt Request/Acknowledge Cycle.** The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 44). When an interrupt is accepted, a special /M1 cycle is generated.

During this /M1 cycle, /IORQ becomes active (instead of /MREQ) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.

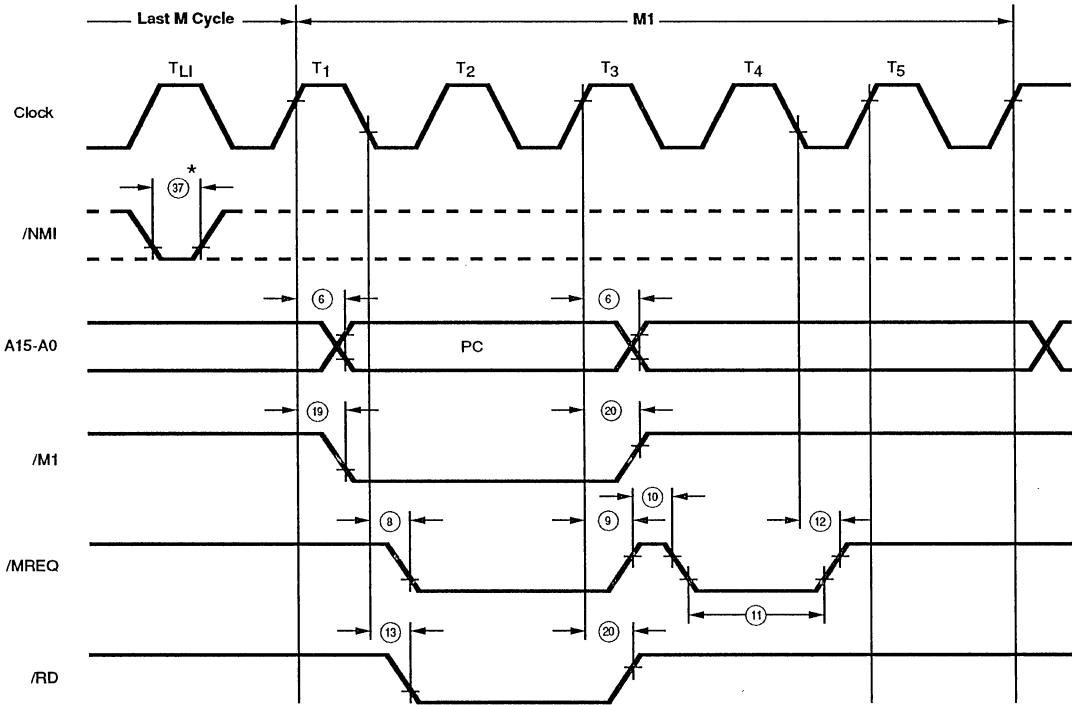


NOTE: 1)  $T_{L1}$  = Last state of any instruction cycle  
 2)  $T_{WA}$  = Wait cycle automatically inserted by CPU

**Figure 44. Interrupt Request/Acknowledge Cycle**  
 (See Table A)

**Non-Maskable Interrupt Request Cycle.** /NMI is sampled at the same time as the maskable interrupt input /INT, but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal

memory read operation except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the /NMI service routine located at the address 0066H (Figure 45).

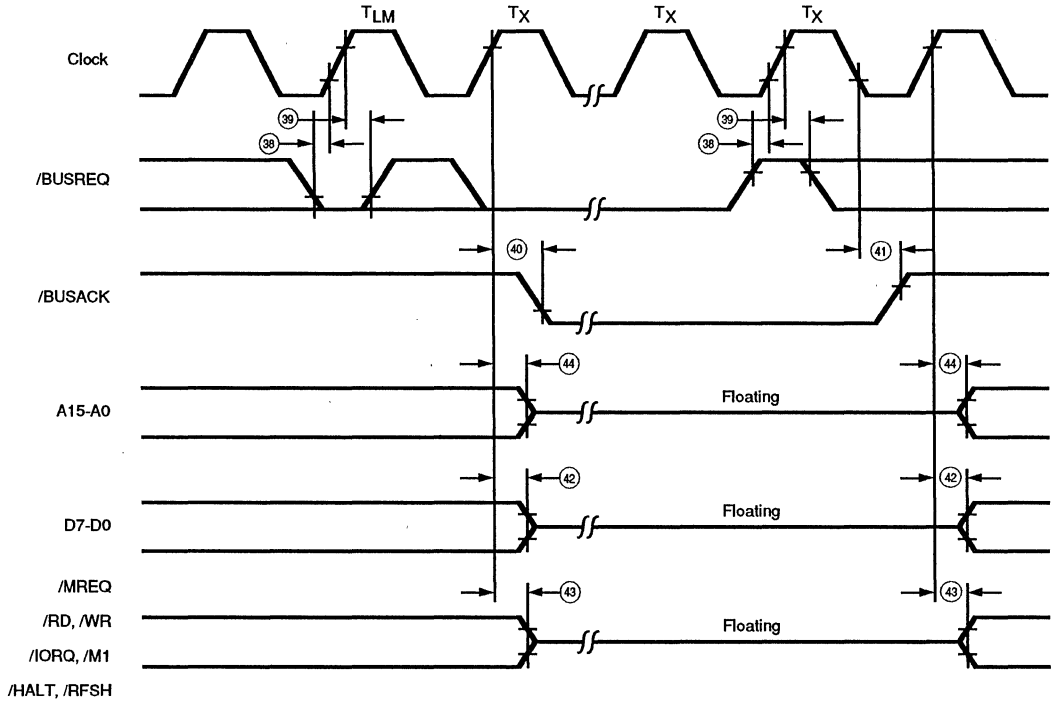


\* Although /NMI is an asynchronous input, to guarantee its being recognized on the following machine cycle, /NMI's falling edge must occur no later than the rising edge of the clock cycle preceding the last state of any instruction cycle (T<sub>L1</sub>).

**Figure 45. Non-Maskable Interrupt Request Operation**  
(See Table A)

**Bus Request/Acknowledge Cycle.** The CPU samples /BUSREQ with the rising edge of the last clock period of any machine cycle (Figure 46). If /BUSREQ is active, the CPU sets its address, data, and /MREQ to inputs, and /IORQ, /RD and /WR lines set to an input for on-chip

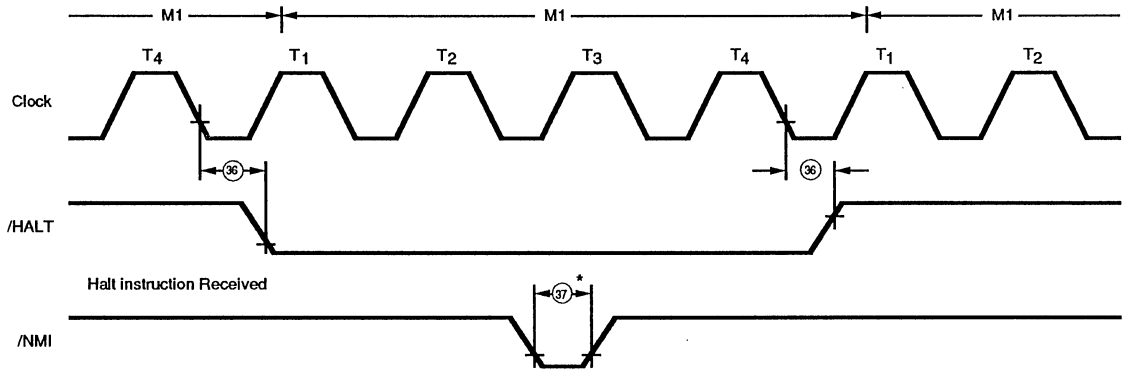
peripheral access from an external bus master with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.



- Notes: 1)  $T_{LM}$  = Last state of any M cycle  
 2)  $T_X$  = An arbitrary clock cycle used by requesting device

**Figure 46. BUS Request/Acknowledge Cycle**  
 (See Table A)

**Halt acknowledge cycle.** Figure 47 shows the timing for Halt acknowledge cycle.



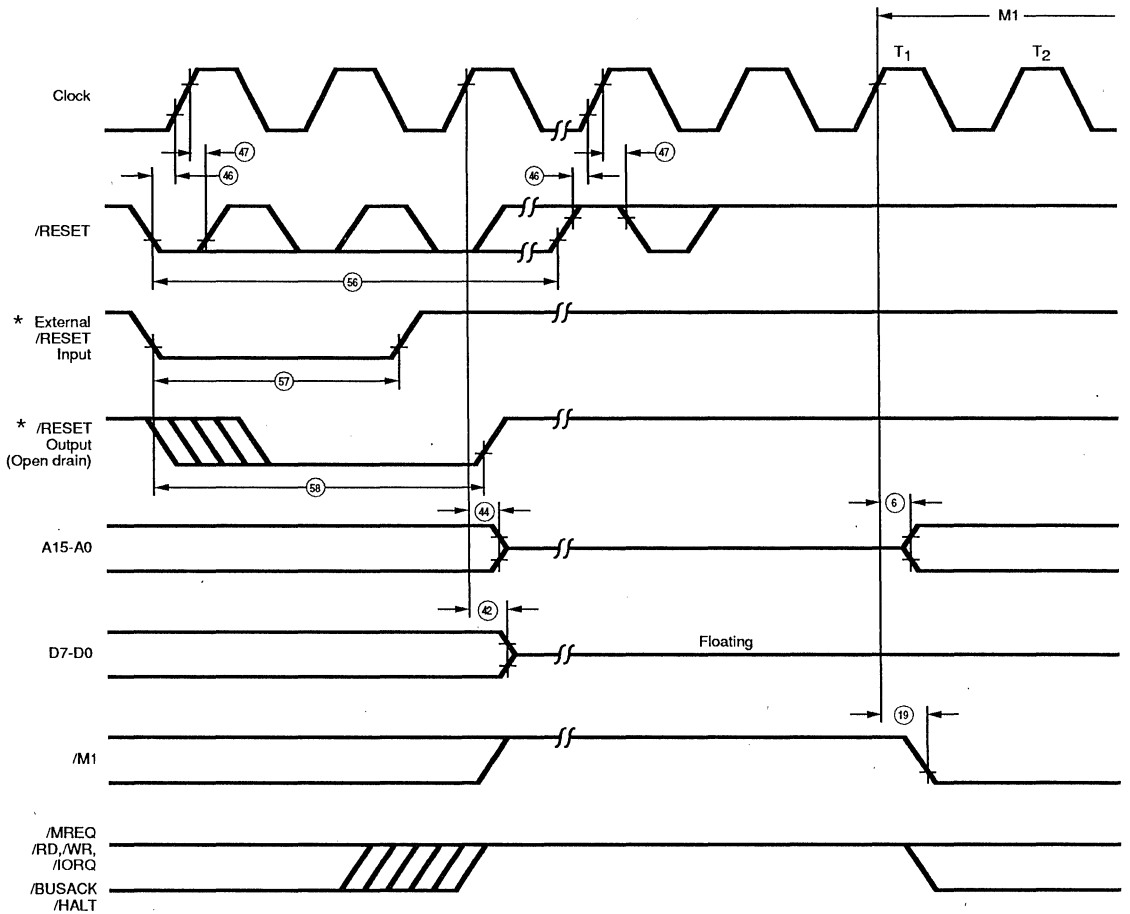
\* Although /NMI is an asynchronous input, to guarantee its being recognized on the following machine cycle, /NMI's falling edge must occur no later than the rising edge of the clock preceding the last state of any instruction cycle ( $T_{L1}$ ).

**Figure 47. Halt Acknowledge**  
(See Table A)

**Reset Cycle.** /RESET must be active for at least three clock cycles for the CPU to properly accept it. As long as /RESET remains active, the address and data buses float, and the control outputs are inactive.

Once /RESET goes inactive, two internal T cycles are consumed before the CPU resumes normal processing operation. /RESET clears the PC register, so the first op-code fetch location is 0000H (Figure 48).

**Z84C13/C15 Only.** If Reset output is disabled, /RESET must be active for at least three clock cycles for the CPU to properly accept it. Otherwise, /RESET must be active for at least two clock cycles and the on-chip reset circuit extends /RESET signal to at least a minimum of 16-clock cycles.



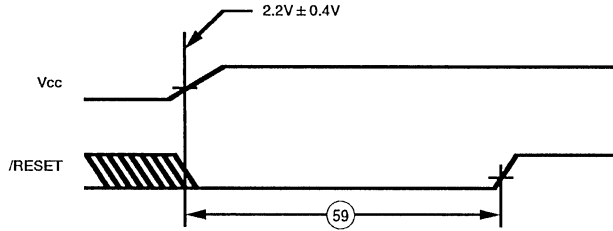
\* 84C13/15 Only Reset Output is Enabled

Figure 48. Reset Cycle  
(See Table A)

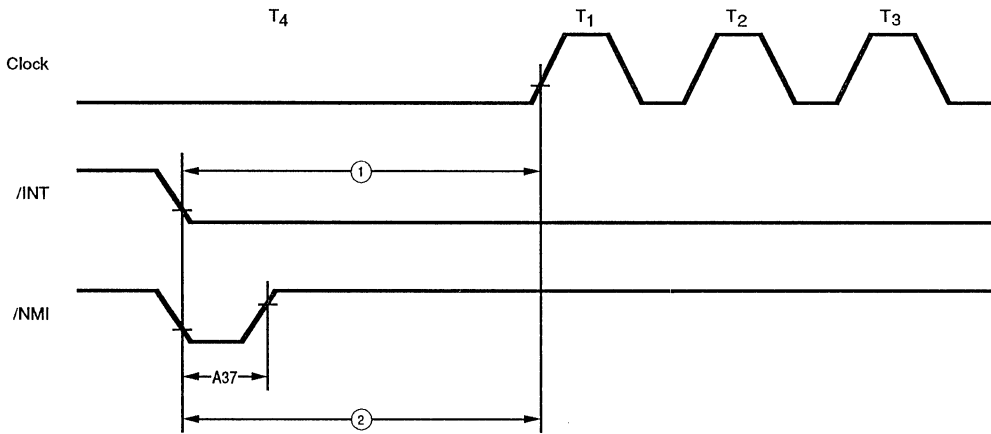
## CGC TIMING

Figure 49 to Figure 52 shows the timing related CGC and Power-On Reset circuit.

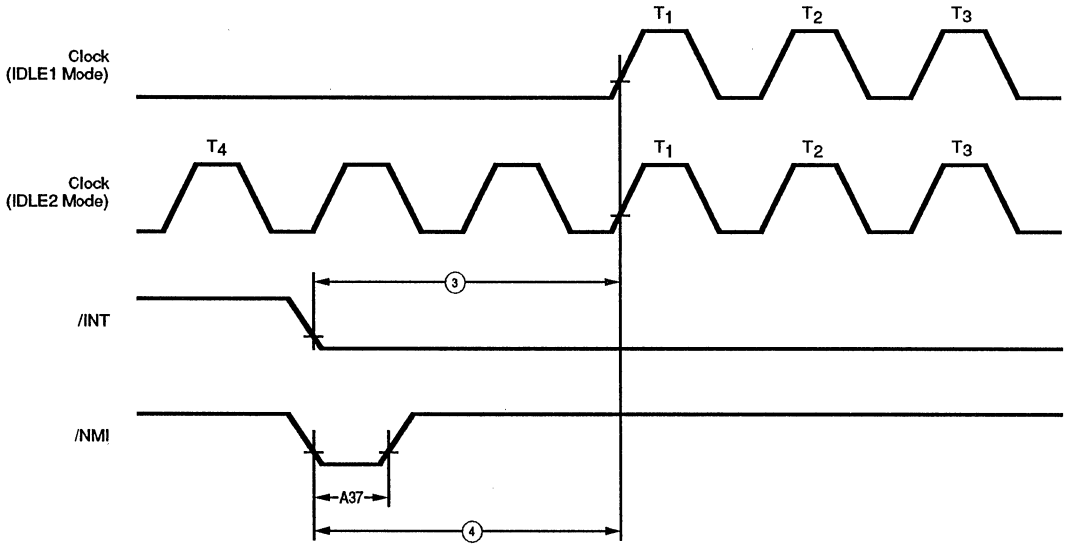
Parameters referenced in Figure 49 thru Figure 52 appear in Table B.



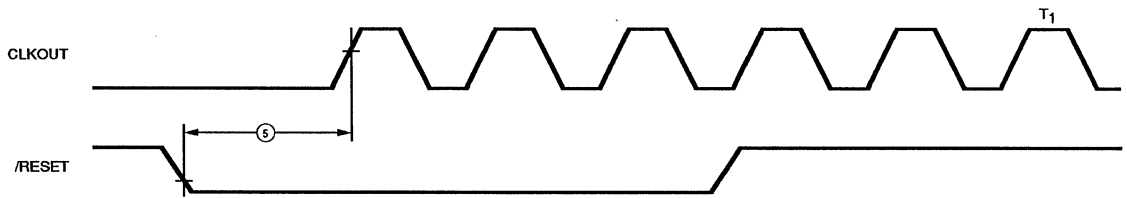
**Figure 49. Reset on Power-up (Applies only for Z84C13/C15)**  
(See Table B)



**Figure 50. Clock Restart Timing by /INT, /NMI (STOP Mode)**  
(See Table B)

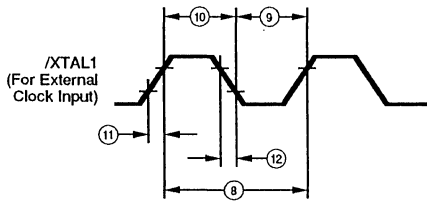


(a) Clock Restart Timing by /INT, /NMI (IDLE1/2 Mode)

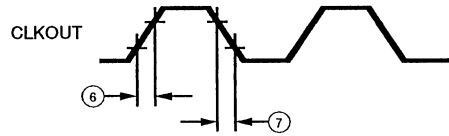


(b) Clock Restart Timing by /RESET (IDLE 1/2 Mode)

Figure 51. Clock Restart Timing (IDLE1/2 Mode)  
(See Table B)



(a) XTAL1 Timing for External Clock Input

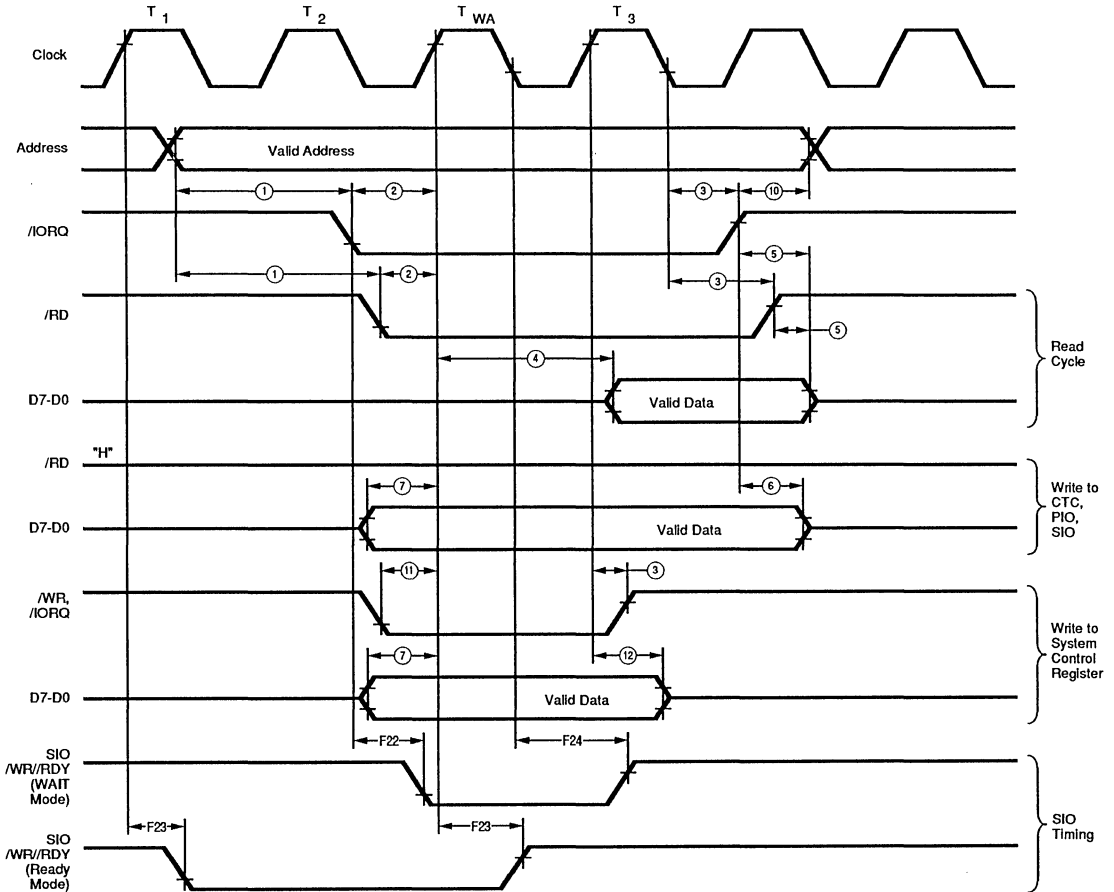


(b) CLKOUT Timing

**Figure 52. Clock Timing**  
(See Table B)

On-chip peripheral access from External Bus master. The timing for the on-chip I/O device access from the external

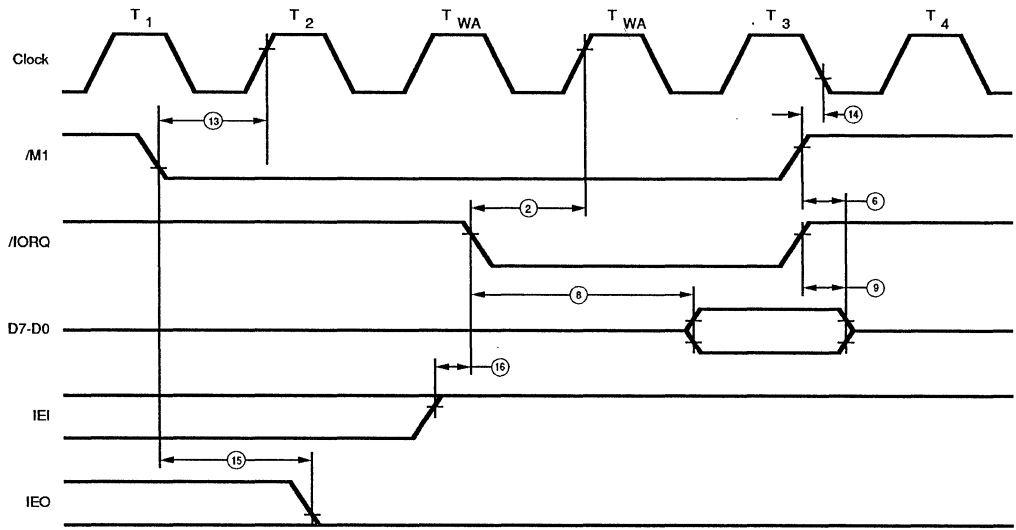
bus master is shown in Figure 53. This timing also applies to the timing during EV mode of operation.



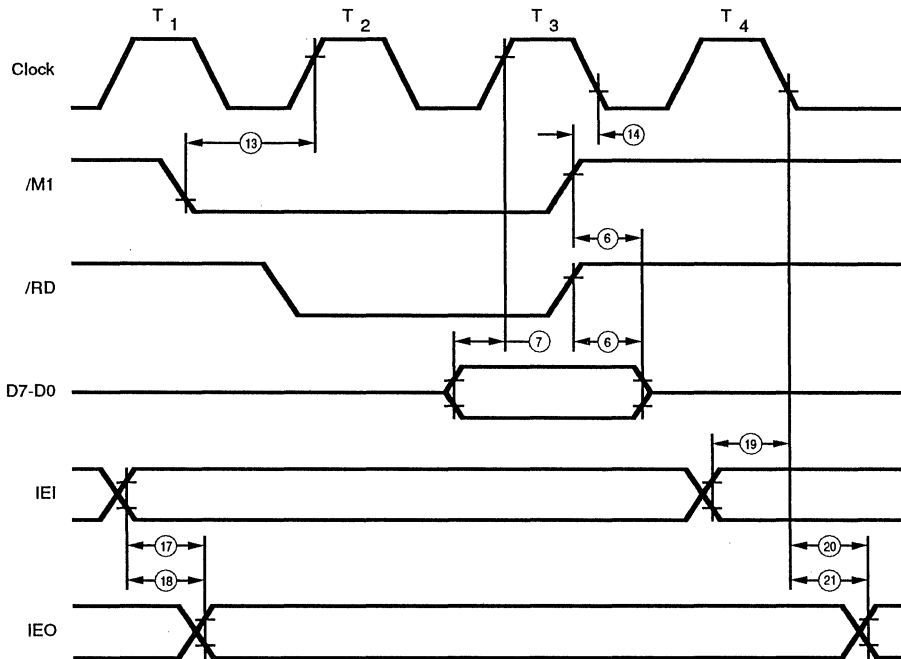
(a) On-chip peripheral I/O access from External Bus master  
(See Tables C and F)

**Figure 53. On-chip Peripheral Timing from External Bus master**





(b) Interrupt Acknowledge Cycle Timing for On-chip peripheral from External Bus master  
(See Table C)

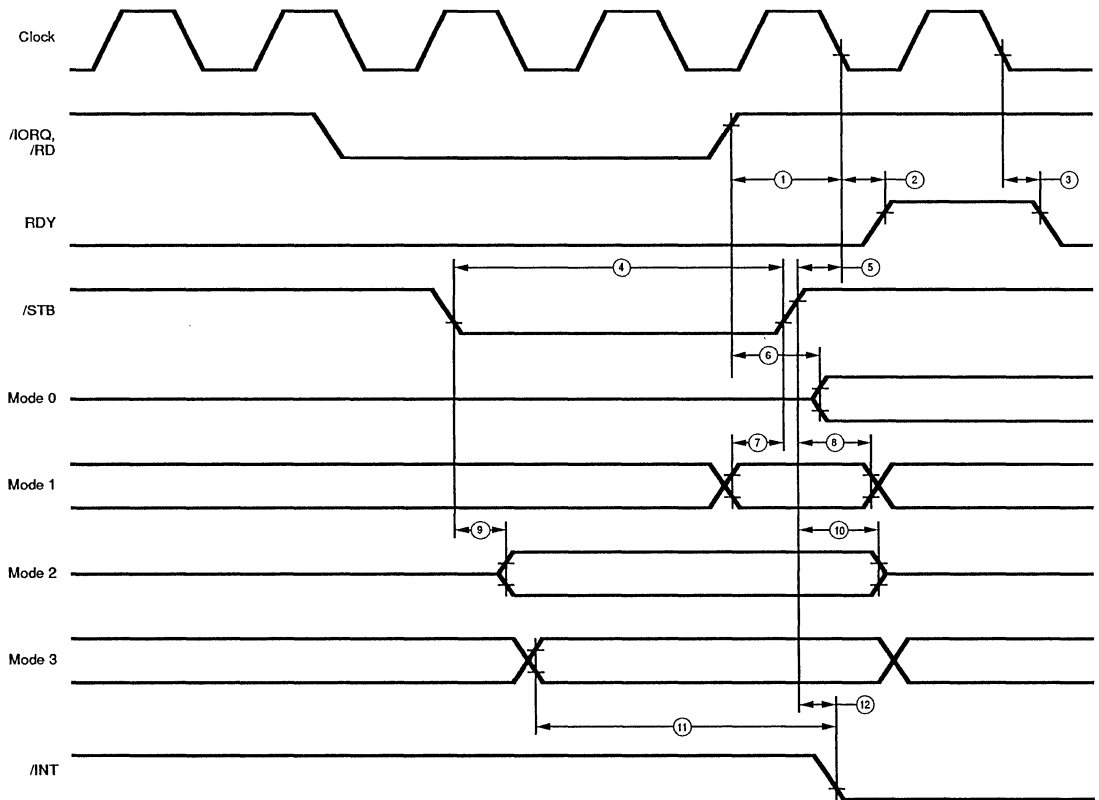


(c) Op-code fetch Cycle Timing for On-chip peripheral from External Bus master  
(See Table C)

Figure 53. On-chip Peripheral Timing from External Bus master (Continued)

### PIO timing

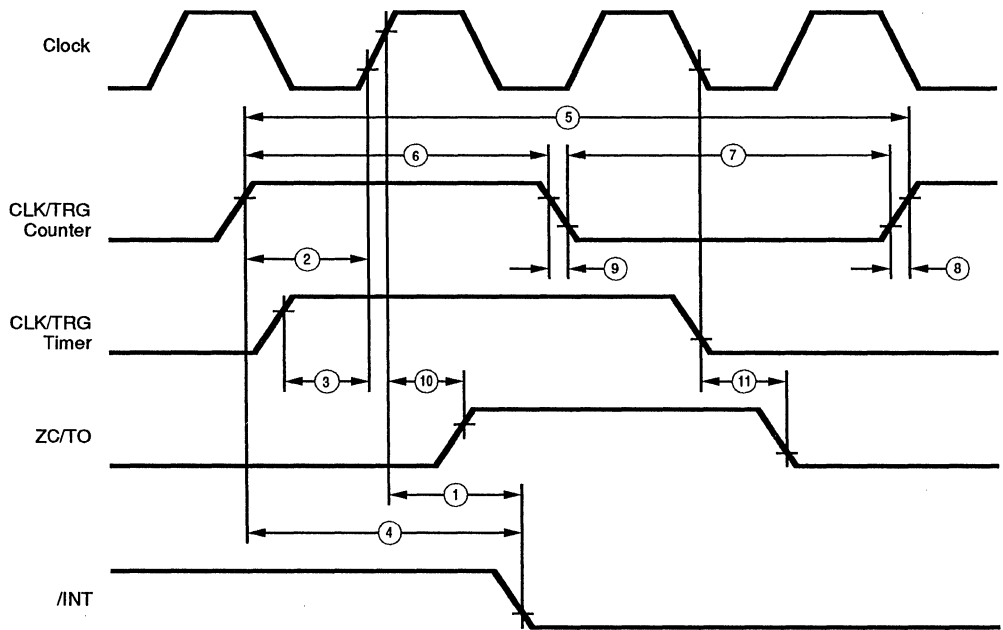
(Not applicable on Z84x13) Figure 54 shows the timing for on-chip PIO.



**Figure 54. PIO Timing**  
(See Table D)

### CTC Timing

Figure 55 shows the timing for on-chip CTC.



**Figure 55. Counter/Timer Timing**  
(See Table E)

## SIO Timing

Figure 56 shows the timing for on-chip SIO.

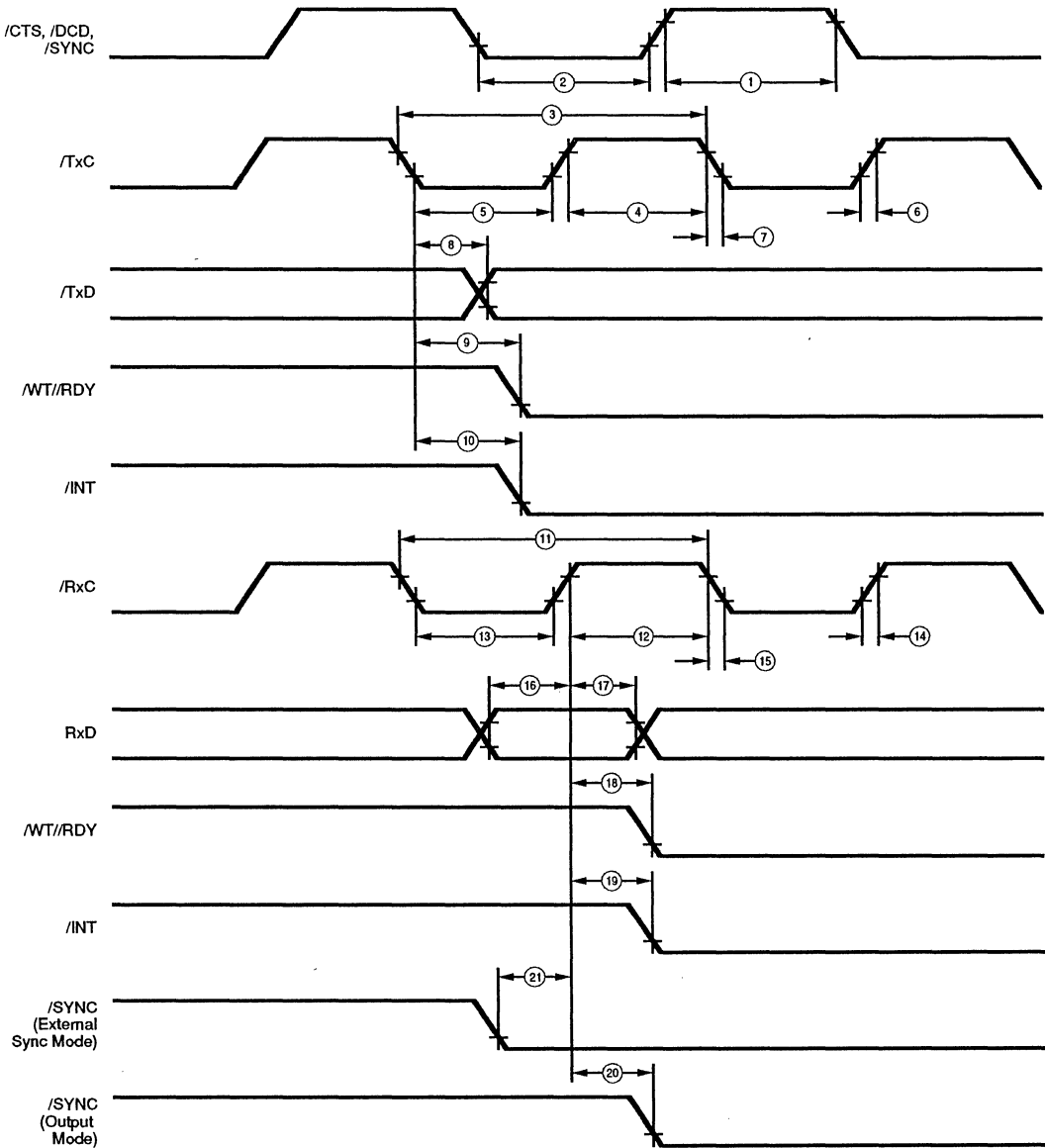


Figure 56. SIO Timing  
(See Table F)

---

## Watch-Dog Timer Timing

Figure 57 shows the timing for Watch-dog Timer.

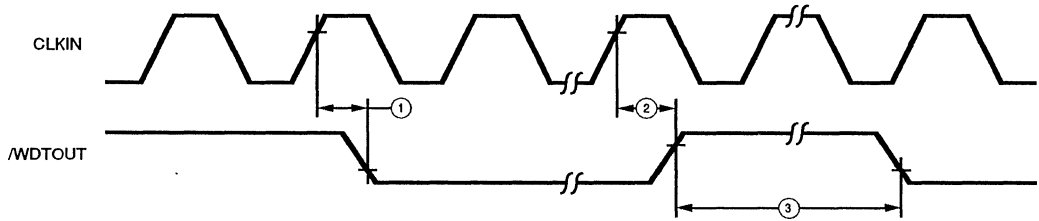


Figure 57. Watch-dog Timer Timing  
(See Table H)

---

## PRECAUTIONS

(1) To release the HALT state by /RESET signal in STOP Mode, hold the /RESET signal at "0" until the output from the internal oscillator stabilizes.

**Z84013/015 Only.** To reset MPU, it is necessary to hold /RESET signal input at "0" level for at least three clocks.

**Z84C13/C15 Only.** If Reset output is disabled, /RESET must be active for at least three clock cycles for the CPU to properly accept it. Otherwise, the on-chip reset circuit extends /RESET signal to at least a minimum of 16-clock cycles.

(2) Releasing the MPU from the HALT state by the interrupt signal in IDLE1/2 Mode and STOP Mode, depends upon the HALT state and the internal system clock. They will stop unless an interrupt signal is accepted during the execution of NOP instruction, even when the internal system clock is restarted by the interrupt signal input. In particular, care must be taken when /INT is used.

Other precautions are identical to those for the Z84C00. Refer to the data sheet for the Z84C00.

---

## ELECTRICAL CHARACTERISTICS

### Absolute Maximum Ratings

Voltage on Vcc with respect to Vss .....	-0.3V to +7.0V
Voltages on all inputs	
with respect to Vss .....	-0.3V to Vcc+0.3V
Operating Ambient	
Temperature .....	See Ordering Information
Storage Temperature .....	-65 °C to + 150 °C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

---

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

Available operating temperature range is:

E = -40°C to 100°C

Voltage Supply Range:

$+4.50V \leq V_{CC} \leq +5.50V$

All AC parameters assume a load capacitance of 100 pF. Add 10 ns delay for each 50 pF increase in load up to a maximum of 150 pF for the data bus and 100 pF for address and control lines. AC timing measurements are referenced to 1.5 volts (except for clock, which is referenced to the 10% and 90% points). Maximum capacitive load for CLK is 125 pF.

The Ordering Information section lists temperature ranges and product numbers. Package drawings are in the Package Information section. Refer to the Literature List for additional documentation.

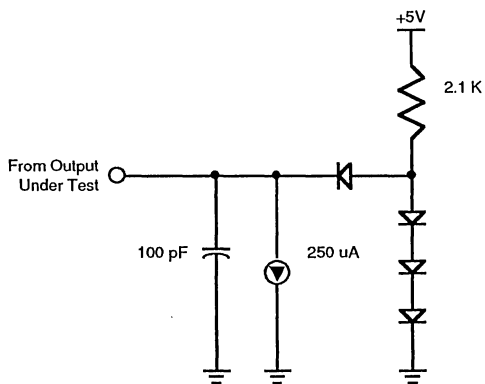


Figure 58. Standard Test Load

---

## CAPACITANCE

Guaranteed by design and characterization

Symbol	Parameter	Min	Max	Unit
$C_{\text{clock}}$	Clock Capacitance	35	pF	
$C_{\text{IN}}$	Input Capacitance	5	pF	
$C_{\text{OUT}}$	Output Capacitance	15	pF	

---

## DC CHARACTERISTICS

$V_{CC}=5.0V \pm 10\%$ , unless otherwise specified

Symbol	Parameter	Min	Max	Unit	Condition
$V_{OLC}$	Clock Output High Voltage	$V_{CC}-0.6$		V	-2.0mA
$V_{OHC}$	Clock Output Low Voltage		0.4	V	+2.0mA
$V_{IHC}$	Clock Input High Voltage	$V_{CC}-0.6$		V	
$V_{ILC}$	Clock Input Low Voltage		0.4	V	
$V_{IH}$	Input High Voltage	2.2	$V_{CC}$	V	
$V_{IL}$	Input Low Voltage	-0.3	0.8	V	
$V_{OL}$	Output Low Voltage		0.4 [5]	V	$I_{LO}=2.0mA$
$V_{OH1}$	Output High Voltage	2.4		V	$I_{OH}=-1.6mA$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$ [5]		V	$I_{OH}=-250\mu A$
$I_{CC1}$	Power Supply Current XTALIN = 10MHz		50	mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
	XTALIN = 6MHz		30	mA	
$I_{CC2}$	Power Supply Current (STOP Mode)		50	$\mu A$	$V_{CC}=5V$
$I_{CC3}$	Power Supply Current (IDLE1 Mode) XTALIN = 10MHz		6	mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
	XTALIN = 6MHz		4	mA	
$I_{CC4}$	Power Supply Current (IDLE2 Mode) XTALIN = 10MHz		TBD [1]	mA	$V_{CC}=5V$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
	XTALIN = 6MHz		TBD [1]	mA	
$I_{LI}$	Input Leakage Current	-10	10 [4]	$\mu A$	$V_{IN}=0.4V$ to $V_{CC}$
$I_{L(SY)}$	SYNC pin Leakage Current	-40	10	$\mu A$	$V_{OUT}=0.4V$ to $V_{CC}$
$I_{LO}$	3-state Output Leakage Current in Float	-10	10 [2]	$\mu A$	$V_{OUT}=0.4V$ to $V_{CC}$
$I_{OHD}$	Darlington Drive Current (Port B and CTC ZC/TO)	-1.5		mA	$V_{OH}=1.5V$ REXT = 390 Ohms

### Notes:

- [1] Measurements made with outputs floating.
- [2] A15-A0, D7-D0, /MREQ, /IORQ, /RD and /WR.
- [3]  $I_{CC2}$  Standby Current is guaranteed when the /HALT pin is low in STOP mode.
- [4] All Pins except XTALI, where  $I_{LI}=\pm 25\mu A$ .
- [5] A15-A0, D7-D0, /MREQ, /IORQ, /RD, /WR, /HALT, /M1 and /BUSACK.

## AC CHARACTERISTICS

Table A. CPU Timing (See Figure 41 to 48)

No	Symbol	Parameter	Z84X1306 Z84X1506		Z84X1310 Z84X1510		Z84C1316* Z84C1516		Unit	Note
			Min	Max	Min	Max	Min	Max		
1	TcC	Clock Cycle time	162**	DC	100**	DC	61	DC	nS	[A1]
2	TwCh	Clock Pulse Width (High)	65	DC	40	DC	20	DC	nS	[A1]
3	TwCl	Clock Pulse Width (Low)	65	DC	40	DC	20	DC	ns	[A1]
4	TfC	Clock Fall time		20		10		6	ns	[A1]
5	TrC	Clock Rise time		20		10		6	ns	[A1]
6	TdCr(A)	Address Valid from Clock Rise		90		65		55	ns	
7	TdA(MREQf)	Address Valid to /MREQ Fall	35**		0**		-15		ns	
8	TdCf(MREQf)	Clock Fall to /MREQ Fall Delay		70		55		40	ns	
9	TdCr(MREQr)	Clock Rise to /MREQ Rise Delay		70		55		40	ns	
10	TwMREQh	/MREQ Pulse Width (High)	65**		30**		10		ns	[A2]
11	TwMREQl	/MREQ Pulse Width (Low)	132**		75**		25		ns	[A2]
12	TdCf(MERQr)	Clock Fall to /MREQ Rise Delay		70		55		40	ns	
13	TdCf(RDf)	Clock Fall to /RD Fall Delay		80		65		40	ns	
14	TdCr(RDr)	Clock Rise to /RD Rise Delay		70		55		40	ns	
15	TsD(Cr)	Data Setup Time to Clock Rise	30		25		10		ns	
16	ThD(RDr)	Data Hold Time After /RD Rise	0		0		0		ns	
17	TsWAIT(Cf)	/WAIT Setup Time to Clock Fall	60		20		75		ns	
18	ThWAIT(Cf)	/WAIT Hold Time After Clock Fall	10		10		10		ns	
19	TdCr(M1f)	Clock Rise to /M1 Fall Delay		80		65		40	ns	
20	TdCr(M1r)	Clock Rise to /M1 Rise Delay		80		65		40	ns	
21	TdCr(RFSHf)	Clock Rise to /RFSH Fall Delay		110		80		60	ns	
22	TdCr(RFSHr)	Clock Rise to /RFSH Rise Delay		100		80		60	ns	
23	TdCf(RDr)	Clock Fall to /RD Rise Delay		70		55		40	ns	
24	TdCr(RDf)	Clock Rise to /RD Fall Delay		70		55		40	ns	
25	TsD(Cf)	Data Setup to Clock Fall During M2, M3, M4 or M5 Cycles	40		25		12		ns	
26	TdA(IORQf)	Address Stable Prior to /IORQ Fall	107**		50**		0		ns	
27	TdCr(IORQf)	Clock Rise to /IORQ Fall Delay		65		50		40	ns	
28	TdCf(IORQr)	Clock Fall to /IORQ Rise Delay		70		55		40	ns	
29	TdD(WRf)	Data Stable Prior to /WR Fall	22**		40**		-10		ns	
30	TdCr(WRf)	Clock Fall to /WR Fall Delay		70		55		40	ns	
31	TwWR	/WR Pulse Width	132**		75**		25		ns	
32	TdCf(WRr)	Clock Fall to /WR Rise Delay		70		55		40	ns	
33	TdD(WRf)IO	Data Stable Prior to /WR Fall	-55**		-10**		-30		ns	
34	TdCr(WRf)	Clock Rise to /WR Fall Delay		60		50		40	ns	
35	TdWRr(D)	Data Stable from /WR Fall	30**		10**		0	0	ns	
36	TdCf(HALT)	Clock Fall to /HALT 0 or 1		260		90		70	ns	
37	TwNMI	/NMI pulse Width	60		60		60		ns	
38	TsBUSREQ(Cr)	/BUSREQ Setup Time to Clock Rise	50		30		15		ns	
39	ThBUSREQ(Cr)	/BUSREQ Hold Time after Clock Rise	10		10		10		ns	
40	TdCr(BUSACKf)	Clock Rise to /BASACK Fall Delay		90		75		40	ns	



## AC CHARACTERISTICS (Continued)

**Table A. CPU Timing (Continued)**

No	Symbol	Parameter	Z84X1306		Z84X1310		Z84C1316*		Unit	Note
			Z84X1506	Min	Max	Z84X1510	Min	Max		
41	TdCf(BUSACKr)	Clock Fall to /BASACK Rise Delay		90		75		40	ns	
42	TdCr(Dz)	Clock Rise to Data Float Delay		80		65		40	ns	
43	TdCr(CTz)	Clock Rise to Control Outputs Float Delay								
		(/MREQ, /IORQ, /RD and /WR)		70		65		40	ns	
44	TdCr(Az)	Clock Rise to Address Float Delay		80		75		40	ns	
45	TdCTr(A)	Address Hold Time from	35**			20**		0	ns	
		/MREQ, /IORQ, /RD or /WR								
46	TsRESET(Cr)	/RESET to Clock Rise Setup Time	60			40		15	ns	
47	ThRESET(Cr)	/RESET to Clock Rise Hold Time	10			10		10	ns	
48	TsINTf(Cr)	/INT Fall to Clock Rise Setup Time	70			50		15	ns	
49	ThINTR(Cr)	/INT Rise to Clock Rise Hold Time	10			10		10	ns	
50	TdM1f(IORQf)	/M1 Fall to /IORQ Fall Delay	359**			220**		100	ns	
51	TdCf(IORQf)	Clock Fall to /IORQ Fall Delay		70		55		45	ns	
52	TdCf(IORQr)	Clock Rise to /IORQ Rise Delay		70		55		45	ns	
53	TdCf(D)	Clock Fall to Data Valid Delay		130		110		75	ns	
54	TRDf(D)	/RD Fall to Output Data Valid		TBD		60		40	ns	
55	TdIORQ(D)	/IORQ Fall to Output Data Valid		TBD		70		45	ns	
56	TwRESET	/RESET Pulse Width								
		013/015, or C13/C15 with RESET	3TcC			3TcC		3TcC	ns	
		Output Disabled								
57	TwRESEToe	/RESET Pulse Width								
		RESET Output Enabled	2TcC			2TcC		2TcC	ns	
58	TwRESETdo	/RESET Drive Duration								
		RESET Output Enabled	16TcC			16TcC		16TcC	ns	
59	TwRESETpor	/RESET drive duration on								
		Power-On Sequence	10	75		10	75	10	75	ms

**Notes:**

\* 16 MHz Timings are preliminary and subject to change. Only C version

\*\* For clock period other than the minimum shown, calculate parameters using the formula on Table H.

[A1] These parameters apply to the external Clock input on CLKIN pin. For the cases where external Clock is fed from XTAL1, please refer to Table B.

[A2] For loading  $\geq 50$  pF, decrease width by 10 ns for each additional 50 pF.

**Table H. Footnote to Table A.**

No	Symbol	Parameter	Z84X1306	Z84X1310	Z84C1316*
			Z84X1506	Z84X1510	Z84C1516
1	TcC	TwCh + TwCl + TrC + TIC			
7	TdA(MREQf)	TwCh + TIC	-50	-50	-45
10	TwMREQh	TwCh + TIC	-20	-20	-20
11	TwMREQl	TcC	-30	-25	-25
26	TdA(IORQf)	TcC	-55	-50	-50
29	TdD(WRf)	TcC	-140	-60	-60
31	TwWR	TcC	-30	-25	-25
33	TdD(WRf)	TwCl + TrC	-140	-60	-60
35	TdWRr(D)	TwCl + TrC	-55	-40	-25
45	TdCTr(A)	TwCl + TrC	-50	-30	-30
50	TdM1f(IORQf)	2TcC + TwCh + TIC	-50	-30	-30

## AC CHARACTERISTICS (Continued)

**Table B. CGC Timing** (See Figure 49 to 52)

No	Symbol	Parameter	Z84C1306 Z84C1506		Z84C1310 Z84C1510		Z84C1316* Z84C1516		Unit	Note
			Min	Max	Min	Max	Min	Max		
1	TRST(INT)S	Clock Restart Time by /INT (STOP Mode)	(Typ)2 <sup>14</sup> +2.5TcC		(Typ)2 <sup>14</sup> +2.5TcC		(Typ)2 <sup>14</sup> +2.5TcC		ns	
2	TRST(MNI)S	Clock Restart Time by /NMI (STOP Mode)	(Typ)2 <sup>14</sup> +2.5TcC		(Typ)2 <sup>14</sup> +2.5TcC		(Typ)2 <sup>14</sup> +2.5TcC		ns	
3	TRST(INT)I	Clock Restart Time by /INT (IDLE Mode)	2.5TcT		2.5TcT		2.5TcT		ns	
4	TRST(Nmi)I	Clock Restart Time by /NMI (IDLE Mode)	2.5TcT		2.5TcT		2.5TcT		ns	
5	TRST(RESET)I	Clock Restart Time by /RESET (IDLE Mode)	1TcC		1TcC		1TcC		ns	
6	TfCLKOUT	CLKOUT Rise Time	15		10		6		ns	
7	TrCLKOUT	CLKOUT Fall time	15		10		6		ns	
8	TcX1	XTAL1 Cycle Time (for External Clock Input on XTAL1) Divide-by-Two Mode Divide-by-One Mode	81 162		50 100		31 61		ns ns	
9	TwlX1	XTAL1 Low Pulse Width (for External Clock Input on XTAL1) Divide-by-Two Mode Divide-by-One Mode	35 65		15 40		10 25		ns ns	
10	TwhX1	XTAL1 High Pulse Width (for External Clock input on XTAL1) Divide-by-Two mode Divide-by-One mode	35 65		15 40		10 25		ns ns	
11	TrX1	XTAL1 Rise Time (for External Clock Input on XTAL1)		25		25		15	ns	[B1]
12	TfX1	XTAL1 Fall Time (for External Clock Input on XTAL1)		25		25		15	ns	[B1]

**Note:**

[B1] If parameters 8 and 9 are not met, adjust parameters 11 and 12 to satisfy parameters 8 and 9.

**Table C. Timing for on-chip peripheral access from external bus master  
and daisy chain timing (See Figure 53(a))**

No	Symbol	Parameter	Z84C1306		Z84C1310		Z84C1316*		Unit	Note
			Z84C1506	Min	Max	Z84C1510	Min	Max		
1	TsA(Rf)	Address Setup Time to /RD, /IORQ Fall	50		40		30		ns	
2	TsRl(Cr)	/RD, /IORQ Rise to Clock Rise Setup	60		50		40		ns	
3	Th	Hold time for Specified Setup	15		15		10		ns	
4	TdCr(DO)	Clock Rise to Data out delay		100		80		60	ns	
5	TdRlr(DOz)	/RD, /IORQ Rise to Data Out Float Delay		75		60		50	ns	
6	ThRDr(D)	/M1, /RD, /IORQ Rise to Data Hold	15	40	15	30		20	ns	[C1]
7	TsD(Cr)	Data In to Clock Rise Setup Time	30		25		15		ns	
8	TdIOl(DOI)	/IORQ Fall to Data Out Delay (INTACK cycle)		95		95		70	ns	
9	ThIOr(D)	/IORQ Rise to Data Hold	15		15		10		ns	
10	ThIOr(A)	/IORQ Rise to Address Hold	15		15		10		ns	
11	TsWllf(Cr)	/IORQ, /WR setup time to Clock Rise New parameter	20		20		15		ns	[C2]
12	ThWRr(Cr)	Clock Rise to /IORQ, /WR Rise hold time	0		0		0		ns	[C2]
13	TsM1f(Cr)	/M1 Fall to Clock Rise Setup Time	40		40		15		ns	
14	TsM1r(Cf)	/M1 Rise to Clock Rise Setup Time (/M1 cycle)	-15		-15		-10		ns	
15	TdM1f(IEOf)	/M1 Fall to IEO Fall delay (Interrupt Immediately Preceding /M1 Fall)		140		80		60	ns	
20	TdCf(IEOr)	Clock Fall to IEO Rise Delay	50		40		30		ns	
21	TdCf(IEOf)	Clock Fall to IEO Rise Delay		90		75		50	ns	

**Notes:**

[C1] For I/O write to PIO, CTC and SIO.

[C2] For I/O Write to system control registers.

[C3] For daisy-chain timing, please refer to the note on Page 356.

## AC CHARACTERISTICS (Continued)

**Table D. PIO Timing (Z84x15 only)** (See Figure 54)

No	Symbol	Parameter	Z84C1506		Z84C1510		Z84C1516*		Unit	Note
			Min	Max	Min	Max	Min	Max		
1	TsIOr(Cr)	/IORQ Rise to Clock Fall Setup Time (To Activate RDY on Next Clock Cycle)	100		100		100		ns	
2	TdCf(RDYr)	Clock Fall to RDY Rise Delay		100		115		30	ns	[D2]
3	TdCf(RDYf)	Clock Fall to RDY Fall Delay		100		115		30	ns	[D2]
4	TwSTB	/STB Pulse Width	100		80		50		ns	[D1]
5	TsSTBr(Cr)	/STB Rise to Clock Fall Setup Time (To Activate RDY on Next Clock Cycle)	100		100		70		ns	[D2]
6	TdIOr(PD)	/IORQ Rise to Port Data Stable Delay (Mode 0)		140		120		100	ns	[D2]
7	TsPD(STBr)	Port Data to /STB Rise Setup Time (Mode 1)	140		75		30		ns	
8	ThPD(STBr)	Port Data to /STB Rise Hold Time (Mode 1)	15		15		15		ns	
9	TdSTBr(PD)	/STB Fall to Port Data Stable (Mode 2)		150		120		30	ns	[D2]
10	TdSTBr(PDz)	/STB Rise to Port Data Float Delay (Mode 2)		140		120		50	ns	
11	TdPD(INTf)	Port Data Match to /INT Fall Delay (Mode 3)		250		200		40	ns	
12	TdSTBr(INTf)	/STB Rise to /INT Fall Delay		290		220		75	ns	

**Notes:**

[D1] For Mode 2: TwSTB > TsPD(STB).

[D2] Increase these values by 2 ns for 10 pF increase in loading up to 100 pF Max.

**Table E. CTC Timing** (Figure 55)

No	Symbol	Parameter	Z84C1306		Z84C1310		Z84C1316*		Unit	Note
			Min	Max	Min	Max	Min	Max		
1	TdCr(INTf)	Clock Rise to /INT Fall Delay		(TcC+100)		(TcC+80)		(TcC+30)		[E1]
2	TsCTR(Cc)	CLK/TRG to Clock Rise Setup Time for Immediate Count	90		90		40		ns	[E2]
3	TsCTR(Ct)	CLK/TRG to Clock Rise Setup Time for Enabling of Prescaler on Following Clock Rise	90		90		40		ns	[E1]
4	TdCTR(INTf)	CLK/TRG to /INT Fall Delay								
		TsCTR(C) Satisfied		(1)+(3)		(1)+(3)		(1)+(3)	ns	[E2]
		TsCTR(C) not Satisfied		TcC+(1)+(3)		TcC+(1)+(3)		TcC+(1)+(3)	ns	[E2]
5	TcCTR	CLK/TRG Cycle time	(2TcC)	DC	(2TcC)	DC	(2TcC)	DC	ns	[E3]
6	TwCTRh	CLK/TRG Width (Low)	90	DC	90	DC	25	DC	ns	
7	TwCTRl	CLK/TRG Width (High)	90	DC	90	DC	25	DC	ns	
8	TrCTR	CLK/TRG Rise Time		30		30		15	ns	
9	TfCTR	CLK/TRG Fall Time		30		30		15	ns	
10	TdCr(ZCr)	Clock Rise to ZC/TO Rise Delay		80		80		25	ns	
11	TdCf(ZCf)	Clock Fall to ZC/TO Fall Delay		80		80		25	ns	

**Notes:**

[E1] Timer Mode.

[E2] Counter Mode.

[E3] Counter Mode only; when using a cycle time less than 3TcC, parameter #2 must be met.

**Table F. SIO Timing** (See Figures 53(a) and 56)

No	Symbol	Parameter	Z84C1306 Z84C1506		Z84C1310 Z84C1510		Z84C1316* Z84C1516		Unit	Note
			Min	Max	Min	Max	Min	Max		
1	TwPh	Pulse Width (High)	150		120		80		ns	
2	TwPl	Pulse Width (Low)	150		120		80		ns	
3	TcTxC	/TxC Cycle Time	250		200		120		ns	[F1]
4	TwTxCH	/TxC Width (High)	85		80		55		ns	
5	TwTxCL	/TxC Width (Low)	85		80		55		ns	
6	TrTxC	/TxC Rise Time		60		60		60	ns	
7	TfTxC	/TxC Fall Time		60		60		60	ns	
8	TdTxCf(TxD)	/TxC Fall to TxD Delay		160		120		40	ns	
9	TdTxCf(W/RRf) (Ready Mode)	/TxC Fall to /W//RDY Fall Delay	5	9	5	9	5	8	TcC	
10	TdTxCf(INTf)	/TxC Fall to /INT Fall Delay	5	9	5	9	5	9	TcC	
11	TcRxC	/RxC Cycle Time	250		200		120		ns	[F1]
12	TwRxCh	/RxC Width (High)	85		80		55		ns	
13	TwRxCl	/RxC Width (Low)	85		80		55		ns	
14	TrRxC	/RxC Rise Time		60		60		60	ns	
15	TfRxC	/RxC Fall Time		60		60		60	ns	
16	TsRxD(RxCr)	RxD to /RxC Rise Setup Time (X1 Mode)	0		0		0		ns	
17	ThRxCr(RxD)	/RxC Rise to RxD Hold Time (X1 Mode)	80		60			40	ns	
18	TdRxCr(W/RRf)	/RxC Rise to /W//RDY Fall Delay (Ready Mode)	10	13	10	13	10	13	TcC	
19	TdRxCr(INTf)	/RxC Rise to /INT Fall Delay	10	13	10	13	10	13	TcC	
20	TdRxCr(SYNCf)	/RxC Rise to /SYNC Fall Delay (Output Modes)	4	7	4	7	4	7	TcC	
21	TsSYNCf(RxCr)	/SYNC Fall to /RxC Rise Setup (External Sync Modes)	-100		-100		-100		ns	[F2]
22	TdIOf(W/RRf)	/IORQ Fall or Valid Address to /W//RDY Delay (Wait Mode)		130		110		40	ns	[F2]
23	TdCr(W/RRf)	Clock Rise to /W//RDY Delay (Ready Mode)		85		85		40	ns	[F2]
24	TdCf(W/Rz)	Clock Fall to /W//RDY Float Delay (Wait Mode)		90		80		40	ns	[F2]

**Notes:**

[F1] In all modes, the System Clock rate must be at least five times the maximum data rate.

[F2] Parameters 22 to 24 are on Figure 53a.

## AC CHARACTERISTICS (Continued)

**Table G. Watch Dog Timer Timing** (See Figure 57)

No	Symbol	Parameter	Z84C1306 Z84C1506		Z84C1310 Z84C1510		Z84C1316* Z84C1516		Units
			Min	Max	Min	Max	Min	Max	
1	TdC(WDTf)	Clock Rise to /WDTOUT Fall Delay		160		160		160	ns
2	TdCr(WbTc)	Clock Rise to /WDTOUT Rise Delay		165		165		160	ns
3	TcWDT	/WDTOUT Cycle Time							
		WDTP = 00	(Typ)2 <sup>16</sup> TcC		(Typ)2 <sup>16</sup> TcC		(Typ)2 <sup>16</sup> TcC		ns
		WDTP = 01	(Typ)2 <sup>18</sup> TcC		(Typ)2 <sup>18</sup> TcC		(Typ)2 <sup>18</sup> TcC		ns
		WDTP = 10	(Typ)2 <sup>20</sup> TcC		(Typ)2 <sup>20</sup> TcC		(Typ)2 <sup>20</sup> TcC		ns
		WDTP = 11	(Typ)2 <sup>22</sup> TcC		(Typ)2 <sup>22</sup> TcC		(Typ)2 <sup>22</sup> TcC		ns

### Notes:

\* In all modes, the System Clock rate must be at least five times the maximum data rate.

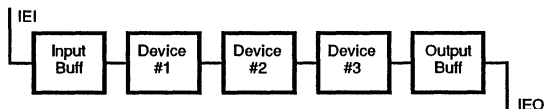
RESET must be active a minimum of one complete clock cycle.

[1] Units equal to System Clock Periods.

[2] Units in nanoseconds (ns).

### Additional information for note [C3]

Parameter #15, 16, 17 and 18 of Table C. These parameters are daisy-chain timing and calculated values, and vary depending on the inside daisy-chain configuration, which is specified in the Interrupt Priority Register. Inside the IPC, the daisy chain can be figured as follows:



**Internal Daisy Chain Configuration**

No	Parameter	6 MHz		10 MHz		16 MHz*		Units
		Min	Max	Min	Max	Min	Max	
15	TdM1(IEO)		160		100		100	ns
16	TsIEI(IO) (PIO at #3)	160		100		100		ns
	(CTC at #3)	160		100		100		ns
	(SIO at #3)	160		100		100		ns
17	TdIEI(IEOf)		120		70		100	ns
18	TdIEI(IEOr)		120		70		100	ns

To calculate IPC daisy-chain timing, it can be treated as if there are Z80 PIO, CTC and SIO with Input buffer and look ahead circuit on the chain. Following are the calculation formulas:

Parameter Table C, #15, /M1 falling to IEO delay  
 $TsM1(IEO) = \text{Max}[TdM1(IO)\#1, TdM1(IO)\#2, TdM1(IO)\#3] + \text{(look-ahead gate Delay)}$

Parameter Table C, #16, IEI to /IORQ falling setup time  
 $TsIEI(IO) = TdIEI(IEO)\#1 + TdIEI(IEO)\#2 + TsIEI(IO)\#3 + \text{(Input Buffer delay)}$

Parameter Table C, #17, IEI falling to IEO falling delay  
 $TdIEI(IEOf) = \text{Max}[TdIEI(IEOf)PIO, TdIEI(IEOf)CTC, TdIEI(IEOf)SIO] + \text{(Input Buffer delay)} + \text{(look-ahead gate Delay)}$

Parameter Table C, #18, IEI rising to IEO rising delay (After ED decode)  
 $TdIEI(IEOr) = TdIEI(IEOr)PIO + TdIEI(IEOr)CTC + TdIEI(IEOr)SIO + \text{(Input Buffer delay)} + \text{(look-ahead gate Delay)}$

\* Where TdIEI(IEO) is worse number between TdIEI(IEOr) and TdIEI(IEOf)

	6MHz Min	Max	10MHz Min	Max	16MHz Min	Max
Input Buffer Delay	10nS		10nS		10 nS	
Look ahead gate delay	10nS		10nS		10 nS	

6MHz	PIO part Min	Max	CTC part Min	Max	SIO part Min	Max
TdM1(IEO)		90nS		130nS		150nS
TsIEI(IO)		90nS		100nS		70nS
TdIEI(IEOf)		100nS		90nS		50nS
TdIEI(IEOr)		130nS		90nS		50nS

10MHz	PIO part Min	Max	CTC part Min	Max	SIO part Min	Max
TdM1(IEO)		60nS		60nS		90nS
TsIEI(IO)		50nS		70nS		50nS
TdIEI(IEOf)		50nS		50nS		30nS
TdIEI(IEOr)		50nS		50nS		30nS

**Preliminary**

16MHz*	PIO part Min	Max	CTC part Min	Max	SIO part Min	Max
TdM1(IEO)		55nS		55nS		90nS
TsIEI(IO)		45nS		65nS		45nS
TdIEI(IEOf)		45nS		45nS		30nS
TdIEI(IEOr)		45nS		45nS		30nS

\* Note:  
16MHz is for C15 only.

If using an interrupt from only a portion of the IPC, these numbers are smaller than the values shown above. For more details about the "Z80 Daisy Chain Structure," please refer to the Application Note "Z80 Family Interrupt Structure" included in the Z80 Data book.



---



## Z80180/Z8S180\*

### Z180® MICROPROCESSOR (\*Z8S180 - PRELIMINARY)

#### FEATURES

- Z80180 Supports Operating Frequency to 10 MHz
- Z8S180 Supports Operating Frequency to 20 MHz
- On-Chip MMU Supports Extended Address Space
- Two DMA Channels
- On-Chip Wait State Generators
- Two UART Channels
- Two 16-Bit Timer Channels
- On-Chip Interrupt Controller
- On-Chip Clock Oscillator/Generator
- Clocked Serial I/O Port
- CPU Control Register (Z8S180 only) provides:
  - Programmable execution speed (to 20 MHz)
  - Programmable EMI noise reduction
  - Reduced operating current (to 20% of Z80180)
  - Fully static operation - low STANDBY current (10  $\mu$ A max.)
- Code Compatible with Zilog Z80 CPU Extended Instructions
- 6 MHz Version Supports 6.144 MHz CPU Clock Operation.

#### GENERAL DESCRIPTION

Based on a microcoded execution unit and an advanced CMOS manufacturing technology, the Z80180 is an 8-bit MPU which provides the benefits of reduced system costs and low power operation while offering higher performance and maintaining compatibility with a large base of industry standard software written around the Zilog Z80® CPU.

The Z8S180 is a fully static version of Z80180 which provides complete software and hardware compatibility to the Z80180 as well as enhancements for doubling execution speeds, power savings and EMI noise reduction.

Higher performance is obtained by virtue of higher operating frequencies, reduced instruction execution times, an enhanced instruction set, and an on-chip memory management unit (MMU) with the capability of addressing up to 1 Mbyte of memory.

Reduced system costs are obtained by incorporating several key system functions on-chip with the CPU. These key functions include I/O devices such as DMA, UART, and timer channels. Also included on-chip are several "glue" functions such as dynamic RAM refresh control, wait state generators, clock oscillator, and interrupt controller.

Not only does the Z80180 consume a low amount of power during normal operation, but it also provides two operating modes that are designed to drastically reduce the power consumption even further. The SLEEP mode reduces power by placing the CPU into a "stopped" state, thereby consuming less current, while the on-chip I/O device is still operating. The SYSTEM STOP mode places both the CPU and the on-chip peripherals into a "stopped" mode, thereby reducing power consumption even further.

In addition to the two operating modes offered by the Z80180, the Z8S180 provides a STANDBY mode. The STANDBY mode consumes less than 10  $\mu$ A by stopping the external oscillators and internal clock.

When combined with other CMOS VLSI devices and memories, the Z80180 provides an excellent solution to system applications requiring high performance, and low power operation.

For the rest of this document, descriptions of Z180 operation applies to both Z80180 and Z8S180.

# GENERAL DESCRIPTION (Continued)

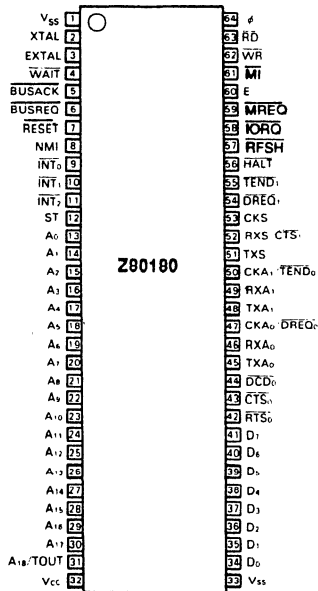


Figure 1a. Z80180 64-Pin DIP

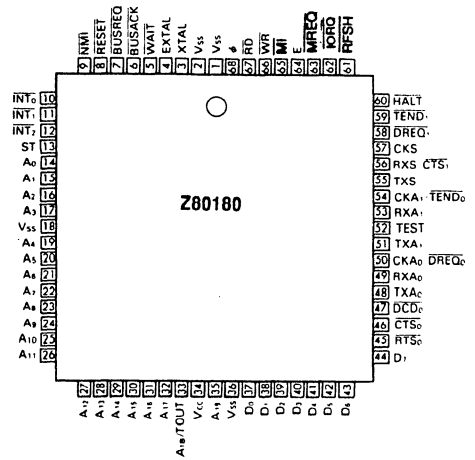


Figure 1b. Z80180 68-Pin PLCC

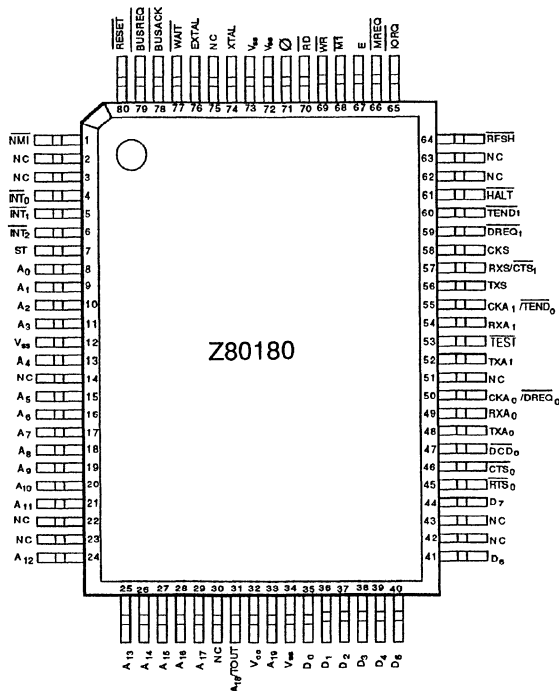


Figure 1c. Z80180 80-Pin QFP

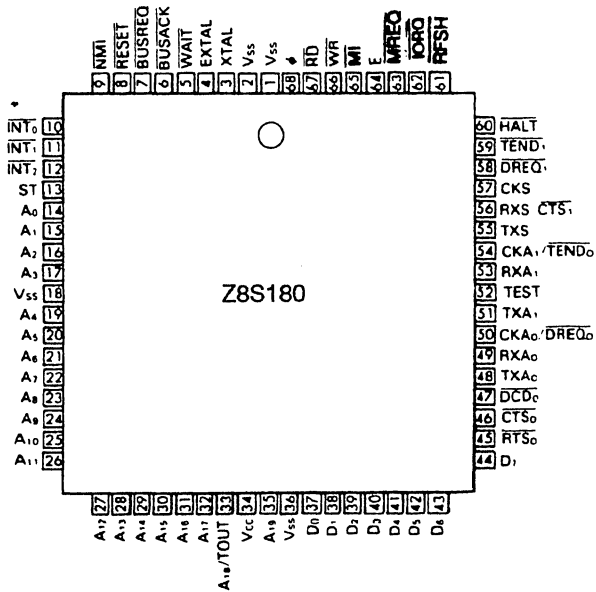


Figure 2a. Z8S180 68-Pin PLCC

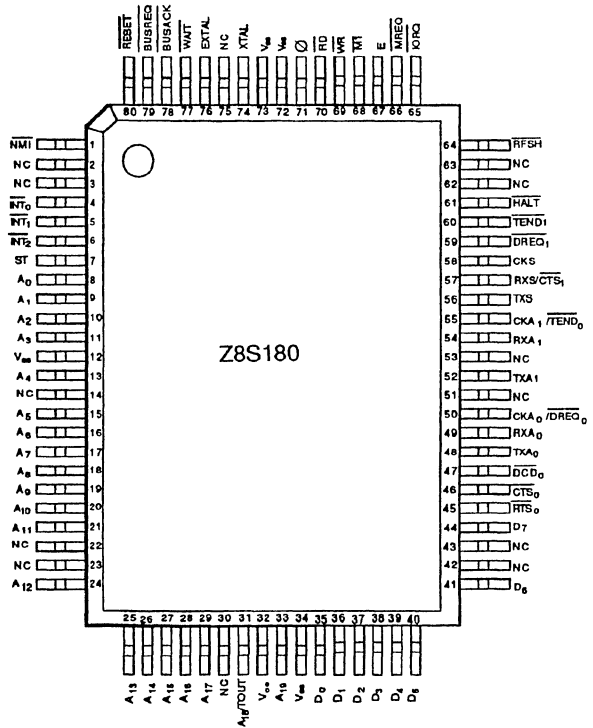


Figure 2b. Z8S180 80-Pin QFP

GENERAL DESCRIPTION (Continued)

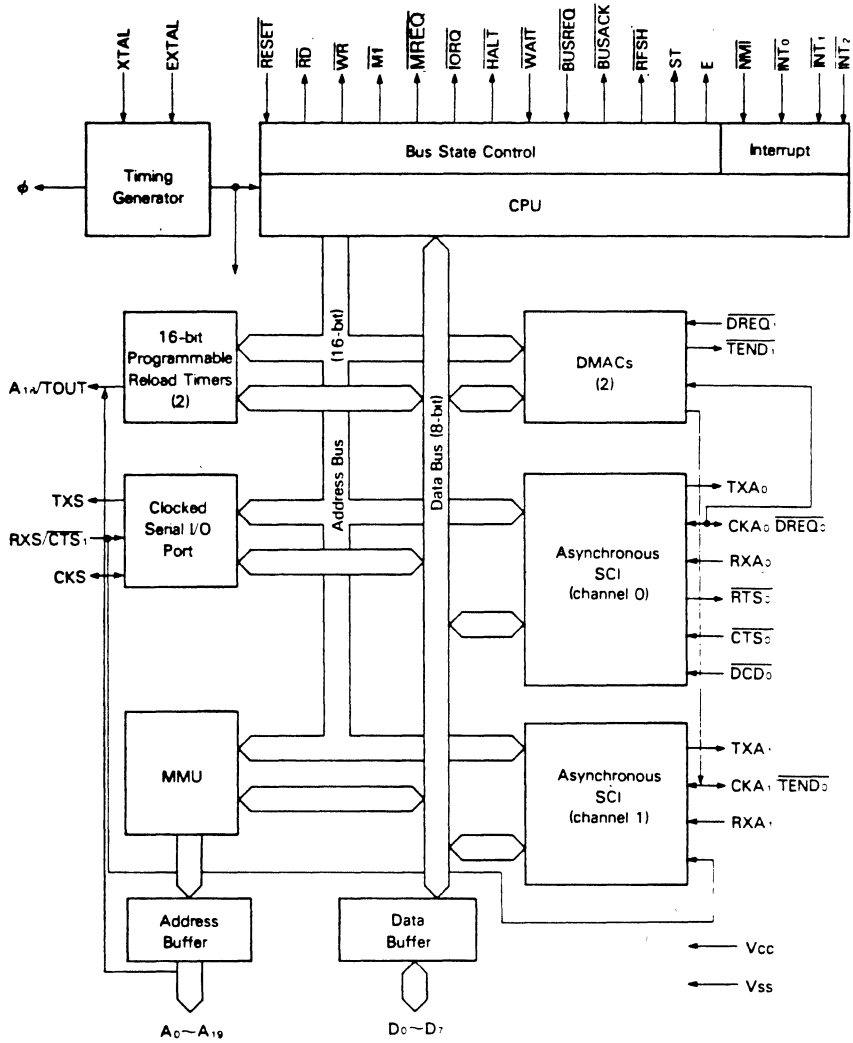


Figure 3. Z80180 Block Diagram

Note:

The Z8S180 contains an additional control register. For details see I/O Register Section.

---

## PIN DESCRIPTION:

**A<sub>0</sub>-A<sub>19</sub>. Address Bus (Output, active High, 3-state).** A<sub>0</sub>-A<sub>19</sub> form a 20-bit address bus. The Address Bus provides the address for memory data bus exchanges, up to 1 Mbyte, and I/O data bus exchanges, up to 64K. The address bus enters a high impedance state during reset and external bus acknowledge cycles. Address line A<sub>18</sub> is multiplexed with the output of PRT channel 1 (TOUT, selected as address output on reset) and address line A<sub>19</sub> is not available in DIP versions of the Z80180.

**$\overline{\text{BUSACK}}$ . Bus Acknowledge (Output, active Low).**  $\overline{\text{BUSACK}}$  indicates the requesting device, the MPU address and data bus, and some control signals, have entered their high impedance state.

**$\overline{\text{BUSREQ}}$ . Bus Request (Input, active Low).** This input is used by external devices (such as DMA controllers) to request access to the system bus. This request has a higher priority than  $\overline{\text{NMI}}$  and is always recognized at the end of the current machine cycle. This signal will stop the CPU from executing further instructions and places the address and data buses, and other control signals, into the high impedance state.

**CKA<sub>0</sub>, CKA<sub>1</sub>. Asynchronous Clock 0 and 1 (Bidirectional, active High).** These pins are the transmit and receive clocks for the synchronous channels. CKA<sub>0</sub> is multiplexed with  $\overline{\text{DREQ}}_0$  and CKA<sub>1</sub> is multiplexed with  $\overline{\text{TEND}}_0$ .

**CKS. Serial Clock (Bidirectional, active High).** This line is clock for the CSIO channel.

**CLOCK. System Clock (Output, active High).** The output is used as a reference clock for the MPU and the external system. The frequency of this output is equal to one-half that of the crystal or input clock frequency.

**$\overline{\text{CTS}}_0$ - $\overline{\text{CTS}}_1$ . Clear to Send 0 and 1 (Inputs, active Low).** These lines are modem control signals for the ASCII channels.  $\overline{\text{CTS}}_1$  is multiplexed with RXS.

**D<sub>0</sub>-D<sub>7</sub>. Data Bus (Bidirectional, active High, 3-state).** D<sub>0</sub>-D<sub>7</sub> constitute an 8-bit bidirectional data bus, used for the transfer of information to and from I/O and memory devices. The data bus enters the high impedance state during reset and external bus acknowledge cycles.

**$\overline{\text{DCD}}_0$ . Data Carrier Detect 0 (Input, active Low).** This is a programmable modem control signal for ASCII channel 0.

**$\overline{\text{DREQ}}_0$ ,  $\overline{\text{DREQ}}_1$ . DMA Request 0 and 1 (Input, active Low).**  $\overline{\text{DREQ}}$  is used to request a DMA transfer from one of the on-chip DMA channels. The DMA channels monitor these inputs to determine when an external device is ready for a read or write operation. These inputs can be programmed to be either level or edge sensed.  $\overline{\text{DREQ}}_0$  is multiplexed with CKA<sub>0</sub>.

**E. Enable Clock (Output, active High).** Synchronous machine cycle clock output during bus transactions.

**EXTAL. External Clock/Crystal (Input, active High).** Crystal oscillator connection. An external clock can be input to the Z80180 on this pin when a crystal is not used. This input is Schmitt triggered.

**$\overline{\text{HALT}}$ . Halt/Sleep Status (Output, active Low).** This output is asserted after the CPU has executed either the  $\overline{\text{HALT}}$  or SLP instruction, and is waiting for either non-maskable or maskable interrupt before operation can resume. It is also used with the  $\overline{\text{M}}_1$  and ST signals to decode status of the CPU machine cycle.

**$\overline{\text{INT}}_0$ . Maskable Interrupt Request 0 (Input, active Low).** This signal is generated by external I/O devices. The CPU will honor this request at the end of the current instruction cycle as long as the  $\overline{\text{NMI}}$  and  $\overline{\text{BUSREQ}}$  signals are inactive. The CPU acknowledges this interrupt request with an interrupt acknowledge cycle. During this cycle, both the  $\overline{\text{M}}_1$  and  $\overline{\text{IORQ}}$  signals will become active.

**$\overline{\text{INT}}_1$ ,  $\overline{\text{INT}}_2$ . Maskable Interrupt Requests 1 and 2 (Inputs, active Low).** This signal is generated by external I/O devices. The CPU will honor these requests at the end of the current instruction cycle as long as the  $\overline{\text{NMI}}$ ,  $\overline{\text{BUSREQ}}$ , and  $\overline{\text{INT}}_0$  signals are inactive. The CPU will acknowledge these interrupt requests with an interrupt acknowledge cycle. Unlike the acknowledgement for  $\overline{\text{INT}}_0$ , during this cycle neither the  $\overline{\text{M}}_1$  or  $\overline{\text{IORQ}}$  signals will become active.

**$\overline{\text{IORQ}}$ . I/O Request (Output, active Low, 3-state).**  $\overline{\text{IORQ}}$  indicates that the address bus contains a valid I/O address for an I/O read or I/O write operation.  $\overline{\text{IORQ}}$  is also generated, along with  $\overline{\text{M}}_1$ , during the acknowledgement of the  $\overline{\text{INT}}_0$  input signal to indicate that an interrupt response vector can be placed onto the data bus. This signal is analogous to the  $\overline{\text{IOE}}$  signal of the Z64180.

**$\overline{\text{M}}_1$ . Machine Cycle 1 (Output, active Low).** Together with  $\overline{\text{MREQ}}$ ,  $\overline{\text{M}}_1$  indicates that the current cycle is the opcode fetch cycle of an instruction execution. Together with  $\overline{\text{IORQ}}$ ,  $\overline{\text{M}}_1$  indicates that the current cycle is for an interrupt acknowledge. It is also used with the  $\overline{\text{HALT}}$  and ST signal to decode status of the CPU machine cycle. This signal is analogous to the  $\overline{\text{LIR}}$  signal of the Z64180.

**$\overline{\text{MREQ}}$ . Memory Request (Output, active Low, 3-state).**  $\overline{\text{MREQ}}$  indicates that the address bus holds a valid address for a memory read or memory write operation. This signal is analogous to the  $\overline{\text{ME}}$  signal of the Z64180.

**$\overline{\text{NMI}}$ . Non-maskable Interrupt (Input, negative edge triggered).**  $\overline{\text{NMI}}$  has a higher priority than  $\overline{\text{INT}}$  and is always recognized at the end of an instruction, regardless of the state of the interrupt enable flip-flops. This signal forces CPU execution to continue at location 0066H.

$\overline{RD}$ . *Read (Output, active Low, 3-state)*.  $\overline{RD}$  indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O or memory device should use this signal to gate data onto the CPU data bus.

$\overline{RFSH}$ . *Refresh (Output, active Low)*. Together with  $\overline{MREQ}$ ,  $\overline{RFSH}$  indicates that the current CPU machine cycle and the contents of the address bus should be used for refresh of dynamic memories. The low order 8 bits of the address bus (A<sub>7</sub>-A<sub>0</sub>) contain the refresh address.

This signal is analogous to the  $\overline{REF}$  signal of the Z64180.

$\overline{RTS_0}$ . *Request to Send 0 (Output, active Low)*. This is a programmable modem control signal for ASCII channel 0.

RXA<sub>0</sub>, RXA<sub>1</sub>. *Receive Data 0 and 1 (Inputs, active High)*. These signals are the receive data to the ASCII channels.

RXS. *Clocked Serial Receive Data (Input, active High)*. This line is the receiver data for the CSIO channel. RXS is multiplexed with the  $\overline{CTS_1}$  signal for ASCII channel 1.

ST. *Status (Output, active High)*. This signal is used with the M1 and HALT output to decode the status of the CPU machine cycle.

Note that the output from M1 is affected by the status of the M1E bit in OMCR register. Table 1 shows the status while M1E = 1.

ST	$\overline{HALT}$	$\overline{M1}$	Operation
0	1	0	CPU operation (1st op-code fetch)
1	1	0	CPU operation (2nd op-code and 3rd op-code fetch)
1	1	1	CPU operation (MC except for op-code fetch)
0	X	1	DMA operation
0	0	0	HALT mode
1	0	1	SLEEP mode (including SYSTEM STOP mode)

NOTE X: Don't care  
MC: Machine cycle

**Table 1. Status Summary**

$\overline{TEND_0}$ ,  $\overline{TEND_1}$ . *Transfer End 0 and 1 (Outputs, active Low)*. This output is asserted active during the last write cycle of a DMA operation. It is used to indicate the end of the block transfer.  $\overline{TEND_0}$  is multiplexed with CKA<sub>1</sub>.

TEST (Output): This pin is for test and is left open.

TOUT. *Timer Out (Output, active High)*. TOUT is the pulse output from PRT channel 1. This line is multiplexed with A<sub>18</sub> of the address bus.

TXA<sub>0</sub>, TXA<sub>1</sub>. *Transmit Data 0 and 1 (Outputs, active High)*. These signals are the transmitted data from the ASCII channels. Transmitted data changes are with respect to the falling edge of the transmit clock.

TXS. *Clocked Serial Transmit Data (Output, active High)*. This line is the transmitted data from the CSIO channel.

$\overline{WAIT}$ . *Wait (Input, active Low)*.  $\overline{WAIT}$  indicates to the MPU that the addressed memory or I/O devices are not ready for a data transfer. This input is used to induce additional clock cycles into the current machine cycle. The  $\overline{WAIT}$  input is sampled on the falling edge of T<sub>2</sub> (and subsequent wait states). If the input is sampled low, then additional wait states are inserted until the  $\overline{WAIT}$  input is sampled high, at which time execution will continue.

$\overline{WR}$ . *Write (Output, active Low, 3-state)*.  $\overline{WR}$  indicates that the CPU data bus holds valid data to be stored at the addressed I/O or memory location.

XTAL. *Crystal (Input, active High)*. Crystal oscillator connection. This pin should be left open if an external clock is used instead of a crystal. The oscillator input is not a TTL level (reference DC characteristics).

### Multiplexed pin descriptions

A<sub>18</sub>/ $\overline{TOUT}$

During RESET, this pin is initialized as A<sub>18</sub> pin. If either TOC1 or TOC0 bit of the Timer Control Register (TCR) is set to 1, TOUT function is selected. If TOC1 and TOC0 bits are cleared to 0, A<sub>18</sub> function is selected.

CKA<sub>0</sub>/ $\overline{DREQ_0}$

During RESET, this pin is initialized as CKA<sub>0</sub> pin. If either DM1 or SM1 in DMA Mode Register (DMODE) is set to 1,  $\overline{DREQ_0}$  function is always selected.

CKA<sub>1</sub>/ $\overline{TEND_0}$

During RESET, this pin is initialized as CKA<sub>1</sub> pin. If CKA1D bit in ASCII control register ch 1 (CNTLA1) is set to 1,  $\overline{TEND_0}$  function is selected. If CKA1D bit is set to 0, CKA<sub>1</sub> function is selected.

RXS/ $\overline{CTS_1}$

During RESET, this pin is initialized as RXS pin. If CTS1E bit in ASCII status register ch1 (STAT1) is set to 1,  $\overline{CTS_1}$  function is selected. If CTS1E bit is set to 0, RXS function is selected.

## ARCHITECTURE:

The Z180 combines a high performance CPU core with a variety of system and I/O resources useful in a broad range of applications. The CPU core consists of five functional blocks: clock generator, bus state controller (including dynamic memory refresh), interrupt controller, memory management unit (MMU), and the central processing unit (CPU). The integrated I/O resources make up the remaining four functional blocks: direct memory access (DMA) control (2 channels), asynchronous serial communications interface (ASCI, 2 channels), programmable reload timers (PRT, 2 channels), and a clock serial I/O (CSIO) channel.

**Clock Generator.** This logic generates the system clock from either an external crystal or clock input. The external clock is divided by two and provided to both internal and external devices.

**Bus State Controller.** This logic performs all of the status and bus control activity associated with both the CPU and some on-chip peripherals. This includes wait state timing, reset cycles, DRAM refresh, and DMA bus exchanges.

**Interrupt Controller.** This block monitors and prioritizes the variety of internal and external interrupts and traps to provide the correct responses from the CPU. To remain compatible with the Z80 CPU, three different interrupt modes are supported.

**Memory Management Unit.** The MMU allows the user to "map" the memory used by the CPU (logically only 64K) into the 1M Byte addressing range supported by the Z180. The organization of the MMU object code compatibility with the Z80 CPU while offering access to an extended memory space. This is accomplished by using an effective "common area - banked area" scheme.

**Central Processing Unit.** The CPU is microcoded to provide a core that is object code compatible with the Z80 CPU. It also provides a superset of the Z80 instruction set, including 8-bit multiply and divide. This core has been enhanced to allow many of the instructions to execute in fewer clock cycles.

**DMA Controller.** The DMA controller provides high speed transfers between memory and I/O devices. Transfer operations supported are memory to memory, memory to/from I/O, and I/O to I/O. Transfer modes supported are request, burst, and cycle steal. DMA transfers can access the full 1 Mbyte addressing range with a block length up to 64K bytes, and can cross over 64K boundaries.

**Asynchronous Serial Communications Interface (ASCI).** The ASCI logic provides two individual full-duplex UARTs. Each channel includes a programmable baud rate generator and modem control signals. The ASCI channels can also support a multiprocessor communications format.

**Programmable Reload Timer (PRT).** This logic consists of two separate channels, each containing a 16-bit counter (timer) and count reload register. The time base for the counters is derived from the system clock (divided by 20) before reaching the counter. PRT channel 1 provides an optional output to allow for waveform generation.

**Clocked Serial I/O (CSIO).** The CSIO channel provides a half-duplex serial transmitter and receiver. This channel can be used for simple high-speed data connection to another microprocessor or microcomputer.

## OPERATION MODES:

The Z180 can be configured to operate like the 64180. This is accomplished by allowing the user to have control over the  $\overline{M1}$ ,  $\overline{IORQ}$ ,  $\overline{WR}$ , and  $\overline{RD}$  signals. The Operation Mode Control Register (OMCR) determines the  $\overline{M1}$  options; the timing of the  $\overline{IORQ}$ ,  $\overline{RD}$ , and  $\overline{WR}$  signals; and the RETI operation.

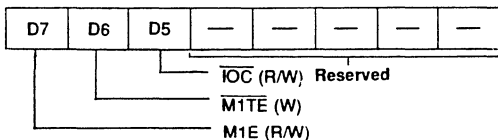


Figure 4. Operation Mode Control Register (I/O Address = 3 EH)

**M1E ( $\overline{M1}$  Enable):** This bit controls the  $\overline{M1}$  output and is set to a 1 during reset.

When  $M1E=1$ , the  $\overline{M1}$  output is asserted LOW during the opcode fetch cycle, the  $\overline{INT0}$  acknowledge cycle, and the first machine cycle of the  $\overline{NMI}$  acknowledge. This will also cause the  $\overline{M1}$  signal to be active during both fetches of the RETI instruction sequence, which may cause corruption of the external interrupt daisy chain. Hence, this bit should be set to 0 for the Z180. When  $M1E=0$ , the  $\overline{M1}$  output is normally inactive and asserted LOW only during the refetch of the RETI instruction sequence and during the  $\overline{INT0}$  acknowledge cycle.

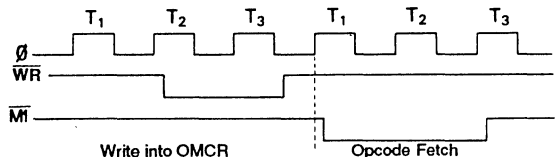


Figure 5.  $\overline{M1}$  Temporary Enable Timing

**$\overline{M1TE}$  ( $\overline{M1}$  Temporary Enable):** This bit controls the temporary assertion of the  $\overline{M1}$  signal. It is always read back as



a 1 and is set to 1 during reset. This function is used to "arm" the internal interrupt structure of the Z80PIO. When a control word is written to the Z80PIO to enable interrupts, no enable actually takes place until the PIO sees an active  $\overline{M1}$  signal. When  $\overline{M1TE}=1$ , there is no change in the operation of the  $\overline{M1}$  signal and M1E controls its function. When  $\overline{M1TE}=0$ , the  $\overline{M1}$  output will be asserted during the next opcode fetch cycle regardless of the state programmed into the M1E bit. This is only momentary (one time) and the user need not reprogram a 1 to disable the function (See Figure 5).

$\overline{IOC}$ : this bit controls the timing of the  $\overline{IORQ}$  and  $\overline{RD}$  signals. It is set to 1 by reset.

When  $\overline{IOC}=1$ , the  $\overline{IORQ}$  and  $\overline{RD}$  signals function the same as the Z64180.

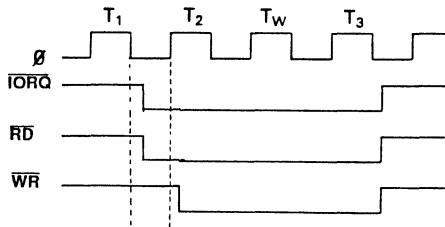


Figure 6. I/O Read and Write Cycles with  $\overline{IOC} = 1$

When  $\overline{IOC}=0$ , the timing of the  $\overline{IORQ}$  and  $\overline{RD}$  signals match the timing required by the Z80 family of peripherals. The  $\overline{IORQ}$  and  $\overline{RD}$  signals will go active as a result of the rising edge of T2. This allows the Z180 to satisfy the setup times required by the Z80 peripherals on those two signals.

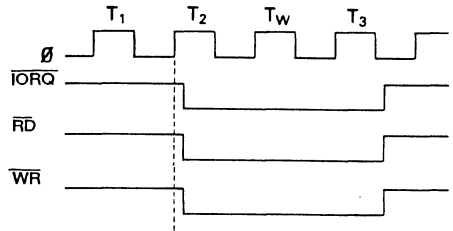


Figure 7. I/O Read and Write Cycles with  $\overline{IOC} = 0$

For the rest of this manual, it is assumed that  $\overline{M1E}=0$  and  $\overline{IOC}=0$ . The user must program the Operation Mode Control Register before the first I/O instruction is executed.

## TIMING:

This section explains the Z180 CPU timing for the following operations:

- Instruction (op-code) fetch timing.
- Operand and data read/write timing.
- I/O read/write timing.
- Basic instruction (fetch and execute) timing.
- RESET timing.
- BUSREQ/BUSACK bus exchange timing.

The basic CPU operation consists of one or more "Machine Cycles" (MC). A machine cycle consists of three system clocks, T1, T2, and T3 while accessing memory or I/O, or it consists of one system clock (T1) during CPU internal operations. The system clock is half the frequency of the Crystal oscillator (e.g., an 8 MHz crystal produces 4 MHz or 250 nsec). For interfacing to slow memory or peripherals, optional wait states (Tw) may be inserted between T2 and T3.

**Instruction (op-code) Fetch Timing.** Fig. 8 shows the instruction (op-code) fetch timing with no wait states. An op-code fetch cycle is externally indicated when the  $\overline{M1}$  output pin is LOW.

In the first half of T1, the address bus (A0-A19) is driven

from the contents of the Program Counter (PC). Note that this is the translated address output of the Z180 on-chip MMU.

In the second half of T1, the  $\overline{MREQ}$  (Memory Request) and  $\overline{RD}$  (Read) signals are asserted LOW, enabling the memory.

The op-code on the data bus is latched at the rising edge of T3 and the bus cycle terminates at the end of T3.

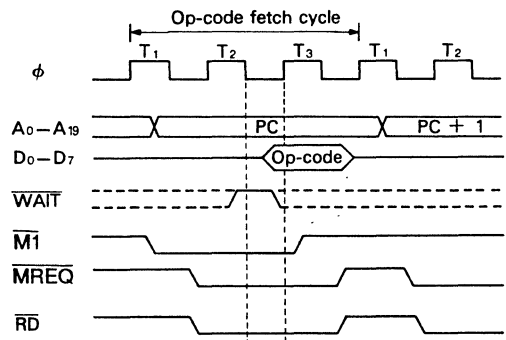


Figure 8. Opcode Fetch timing (Without Wait State)

Fig. 9 illustrates the insertion of wait states ( $T_w$ ) into the op-code fetch cycle. Wait states ( $T_w$ ) are controlled by the external  $\overline{\text{WAIT}}$  input combined with an on-chip programmable wait state generator.

At the falling edge of  $T_2$  the combined  $\overline{\text{WAIT}}$  input is sampled. If  $\overline{\text{WAIT}}$  input is asserted LOW, a wait state ( $T_w$ ) is inserted. The address bus,  $\overline{\text{MREQ}}$ ,  $\overline{\text{RD}}$  and  $\overline{\text{M}}$  are held stable during wait states. When the  $\overline{\text{WAIT}}$  is sampled inactive HIGH at the falling edge of  $T_w$ , the bus cycle enters  $T_3$  and completes at the end of  $T_3$ .

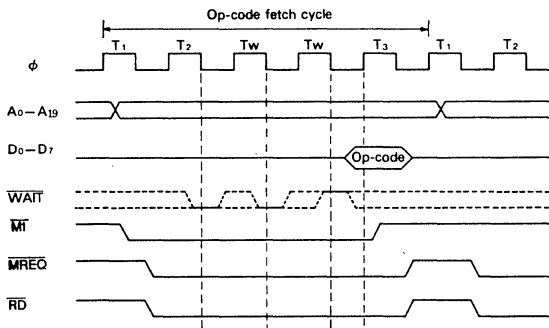


Figure 9. Opcode Fetch Timing (With Wait State)

**Operand and Data Read/Write Timing.** The instruction operand and data read/write timing differs from op-code fetch timing in two ways. First, the  $\overline{\text{M}}$  output is held inactive. Second, the read cycle timing is relaxed by one-half clock cycle since data is latched at the falling edge of  $T_3$ .

Instruction operands include immediate data, displacement, and extended addresses, and have the same timing as memory data reads.

During memory write cycles the  $\overline{\text{MREQ}}$  signal goes active in the second half of  $T_1$ . At the end of  $T_1$ , the data bus is driven with the write data.

At the start of  $T_2$ , the  $\overline{\text{WR}}$  signal is asserted LOW enabling the memory.  $\overline{\text{MREQ}}$  and  $\overline{\text{WR}}$  go inactive in the second half of  $T_3$  followed by disabling of the write data on the data bus.

Wait states ( $T_w$ ) are inserted as previously described for op-code fetch cycles. Fig. 10 illustrates the read/write timing without wait states ( $T_w$ ), while Fig. 11 illustrates read/write timing with wait states ( $T_w$ ).

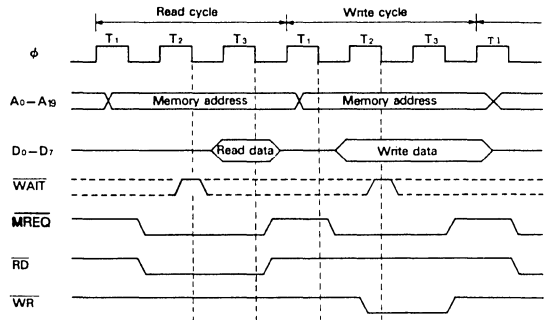


Figure 10. Memory Read/Write Timing (Without Wait State)

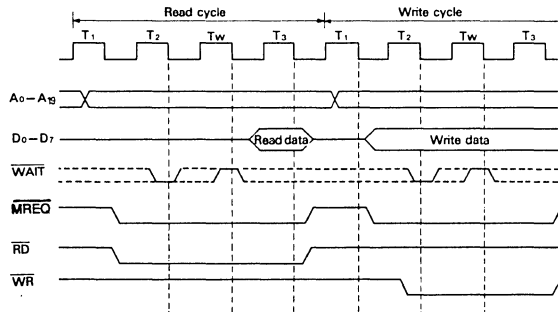


Figure 11. Memory Read/Write Timing (With Wait State)

**I/O Read/Write Timing.** I/O instructions cause data read/write transfers which differ from memory data transfers in the following three ways:

1. The  $\overline{\text{IORQ}}$  (I/O Request) signal is asserted LOW instead of the  $\overline{\text{MREQ}}$  signal.
2. The 16-bit I/O address is not translated by the MMU.
3.  $A_{16}-A_{19}$  are held LOW.

At least one wait state ( $T_w$ ) is always inserted for I/O read and write cycles (except internal I/O cycles).

Fig. 12 shows I/O read/write timing with the automatically inserted wait state ( $T_w$ ).

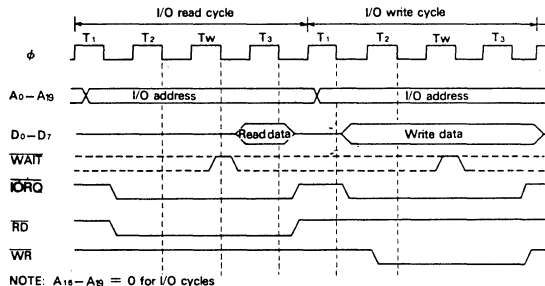
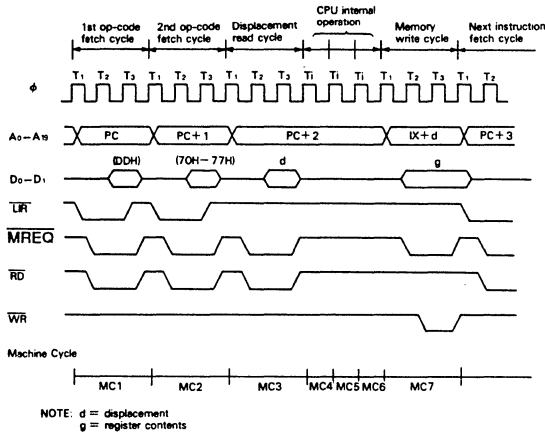


Figure 12. I/O Read/Write Timing

**Basic Instruction Timing.** An instruction may consist of a number of machine cycles including op-code fetch, operand fetch, and data read/write cycles. An instruction may also include cycles for internal processes which make the bus idle.



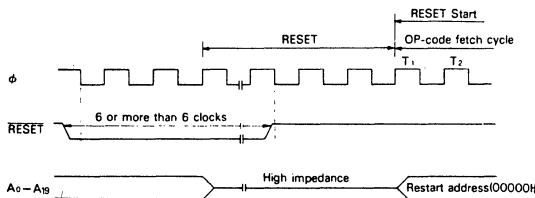
**Figure 13. Instruction Timing**

The example in Fig. 13 illustrates the bus timing for the data transfer instruction LD (IX+d),g. This instruction moves the contents of a CPU register (g) to the memory location with address computed by adding a signed 8-bit displacement (d) to the contents of an index register (IX).

The instruction cycle starts with the two machine cycles to read the two byte instruction op-code as indicated by  $\overline{MREQ}$  LOW. Next, the instruction operand (d) is fetched.

The external bus is idle while the CPU computes the effective address. Finally, the computed memory location is written with the contents of the CPU register (g).

**RESET Timing.** Fig. 14 shows the Z180 hardware RESET timing. If the RESET pin is LOW for six or more than six clock cycles, processing is terminated and the Z180 restarts execution from (logical and physical) address 00000H.



**Figure 14. Reset Timing**

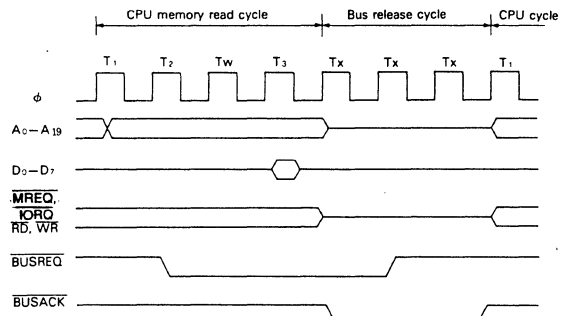
**BUSREQ/BUSACK Bus Exchange Timing.** The Z180 can coordinate the exchange of control, address and data bus ownership with another bus master. The alternate bus master can request the bus release by asserting the  $\overline{BUSREQ}$  (Bus Request) input LOW. After the Z180 releases the bus, it relinquishes control to the alternate bus master by asserting the  $\overline{BUSACK}$  (Bus Acknowledge) output LOW.

The bus may be released by the Z180 at the end of each machine cycle. In this context, a machine cycle consists of a minimum of 3 clock cycles (more if wait states are inserted) for op-code fetch, memory read/write, and I/O read/write cycles. Except for these cases, a machine cycle corresponds to one clock cycle.

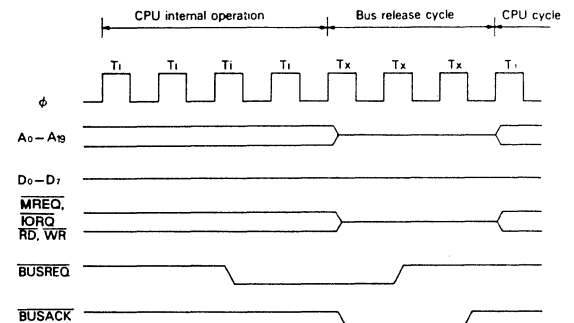
When the bus is released, the address ( $A_0-A_{19}$ ), data ( $D_0-D_7$ ), and control ( $\overline{MREQ}$ ,  $\overline{IORQ}$ ,  $\overline{RD}$ , and  $\overline{WR}$ ) signals are placed in the high impedance state.

Note that dynamic RAM refresh is not performed when the Z180 has released the bus. The alternate bus master must provide dynamic memory refreshing if the bus is released for long periods of time.

Fig. 15 illustrates  $\overline{BUSREQ}/\overline{BUSACK}$  bus exchange during a memory read cycle. Fig. 16 illustrates bus exchange when the bus release is requested during a Z180 CPU internal operation.  $\overline{BUSREQ}$  is sampled at the falling edge of the system clock prior to  $T_3$ ,  $T_i$  and  $T_x$  ( $\overline{BUS RELEASE}$  state). If  $\overline{BUSREQ}$  is asserted LOW at the falling edge of the clock state prior to  $T_x$ , another  $T_x$  is executed.



**Figure 15. Bus Exchange Timing**



**Figure 16. Bus Exchange Timing**

## WAIT State Generator

To ease interfacing with slow memory and I/O devices, the Z180 uses wait states ( $T_w$ ) to extend bus cycle timing. A wait state(s) is inserted based on the combined (logical OR) state of the external  $\overline{\text{WAIT}}$  input and an internal programmable wait state ( $T_w$ ) generator. Wait states ( $T_w$ ) can be inserted in both CPU execution and DMA transfer cycles.

When the external  $\overline{\text{WAIT}}$  input is asserted LOW, wait state(s) ( $T_w$ ) are inserted between  $T_2$  and  $T_3$  to extend the bus cycle duration. The  $\overline{\text{WAIT}}$  input is sampled at the falling edge of the system clock in  $T_2$  or  $T_w$ . If the  $\overline{\text{WAIT}}$  input is asserted LOW at the falling edge of the system clock in  $T_w$ , another  $T_w$  is inserted into the bus cycle. Note that  $\overline{\text{WAIT}}$  input transitions must meet specified set-up and hold times. This can easily be accomplished by externally synchronizing  $\overline{\text{WAIT}}$  input transitions with the rising edge of the system clock.

Dynamic RAM refresh is not performed during wait states ( $T_w$ ) and thus system designs which use the automatic

refresh function must consider the affects of the occurrence and duration of wait states ( $T_w$ ). Figure 17 shows WAIT timing.

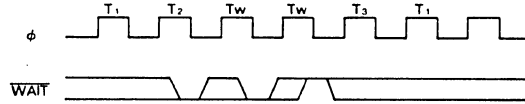


Figure 17.  $\overline{\text{WAIT}}$  Timing

**Programmable Wait State Insertion.** In addition to the  $\overline{\text{WAIT}}$  input, wait states ( $T_w$ ) can also be inserted by program using the Z180 on-chip wait state generator. Wait state ( $T_w$ ) timing applies for both CPU execution and on-chip DMAC cycles.

By programming the four significant bits of the DMA/WAIT Control Register (DCNTL) the number of wait states, ( $T_w$ ) automatically inserted in memory and I/O cycles, can be separately specified.

## HALT and Low Power Operation Modes

The Z180 can operate in 4 different modes. HALT mode, IOSTOP mode and 2 low power operation modes - SLEEP and SYSTEM STOP. Note that in all operating modes, the basic CPU clock (XTAL, EXTAL) must remain active.

**HALT mode.** HALT mode is entered by execution of the HALT instruction (op-code = 76H) and has the following characteristics.

- (1) The internal CPU clock remains active.
- (2) All internal and external interrupts can be received.
- (3) Bus exchange ( $\overline{\text{BUSREQ}}$  and  $\overline{\text{BUSACK}}$ ) can occur.
- (4) Dynamic RAM refresh cycle ( $\overline{\text{RFSH}}$ ) insertion continues at the programmed interval.
- (5) I/O operations (ASCI, CSI/O and PRT) continue.
- (6) The DMAC can operate.
- (7) The  $\overline{\text{HALT}}$  output pin is asserted LOW.
- (8) The external bus activity consists of repeated "dummy" fetches of the op-code following the HALT instruction.

Essentially, the Z180 operates normally in HALT mode, except that instruction execution is stopped.

HALT mode can be exited in the following two ways.

**RESET Exit from HALT mode.** If the  $\overline{\text{RESET}}$  input is as-

serted LOW for at least 6 clock cycles, HALT mode is exited and the normal RESET sequence (restart at address 00000H) is initiated.

**Interrupt Exit from HALT mode.** When an internal or external interrupt is generated, HALT mode is exited and the normal interrupt response sequence is initiated.

If the interrupt source is masked (individually by enable bit, or globally by IEF1 state), the Z180 remains in HALT mode. However,  $\overline{\text{NMI}}$  interrupt will initiate the normal  $\overline{\text{NMI}}$  interrupt response sequence independent of the state of IEF1.

HALT timing is shown in Fig 18.

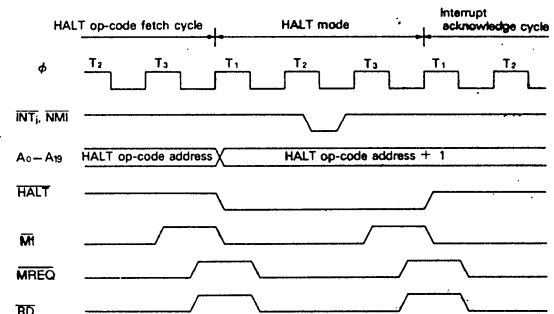


Figure 18. HALT Timing

**SLEEP mode.** SLEEP mode is entered by execution of the 2 byte SLP instruction. SLEEP mode has the following characteristics.

- (1) The internal CPU clock stops, reducing power consumption.
- (2) The internal crystal oscillator does not stop.
- (3) Internal and external interrupt inputs can be received.
- (4) DRAM refresh cycles stop.
- (5) I/O operations using on-chip peripherals continue.
- (6) The internal DMAC stop.
- (7)  $\overline{\text{BUSREQ}}$  can be received and acknowledged.
- (8) Address outputs go HIGH and all other control signal output become inactive HIGH.
- (9) Data Bus, 3-state.

SLEEP mode is exited in one of two ways as shown below.

**RESET Exit from SLEEP mode.** If the  $\overline{\text{RESET}}$  input is held LOW for at least 6 clock cycles, it will exit SLEEP mode and begin the normal RESET sequence with execution starting at address (logical and physical) 00000H.

**Interrupt Exit from SLEEP mode.** The SLEEP mode is exited by detection of an external (NMI, INT<sub>0</sub>-INT<sub>2</sub>) or internal (ASCI, CSI/O, PRT) interrupt.

In case of  $\overline{\text{NMI}}$ , SLEEP Mode is exited and the CPU begins the normal NMI interrupt response sequence.

In the case of all other interrupts, the interrupt response depends on the state of the global interrupt enable flag (IEF<sub>1</sub>) and the individual interrupt source enable bit.

If the individual interrupt condition is disabled by the corresponding enable bit, occurrence of that interrupt is ignored and the CPU remains in the SLEEP state.

Assuming the individual interrupt condition is enabled, the response to that interrupt depends on the global interrupt enable flag (IEF<sub>1</sub>). If interrupts are globally enabled

(IEF<sub>1</sub>=1) and an individually enabled interrupt occurs, SLEEP mode is exited and the appropriate normal interrupt response sequence is executed.

If interrupts are globally disabled (IEF<sub>1</sub>=0) and an individually enabled interrupt occurs, SLEEP mode is exited and instruction execution begins with the instruction following the SLP instruction. Note that this provides a technique for synchronization with high speed external events without incurring the latency imposed by an interrupt response sequence.

Figure 19 shows SLEEP timing.

**IOSTOP mode.** IOSTOP mode is entered by setting the IOSTOP bit of the I/O Control Register (ICR) to 1. In this case, on-chip I/O (ASCI, CSI/O, PRT) stops operating. However, the CPU continues to operate. Recovery from IOSTOP mode is by resetting the IOSTOP bit in ICR to 0.

**SYSTEM STOP mode.** SYSTEM STOP mode is the combination of SLEEP and IOSTOP modes. SYSTEM STOP mode is entered by setting the IOSTOP bit in ICR to 1 followed by execution of the SLP instruction. In this mode, on-chip I/O and CPU stop operating, reducing power consumption. Recovery from SYSTEM STOP mode is the same as recovery from SLEEP mode, noting that internal I/O sources (disabled by IOSTOP) cannot generate a recovery interrupt.

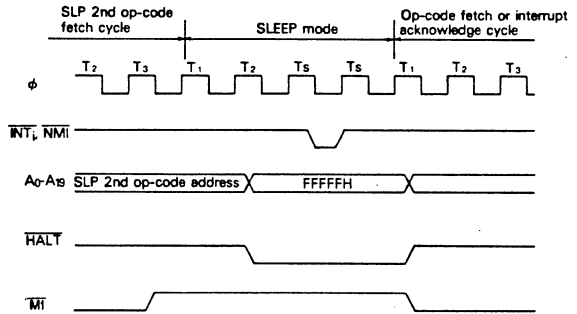


Figure 19. SLEEP Timing

## Trap and Interrupts

The Z180 CPU has twelve interrupt sources, 4 external and 8 internal, with fixed priority. (Reference Figure 20).

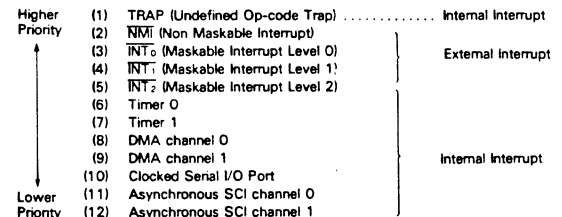


Figure 20. Interrupt Sources

**TRAP Interrupt.** The Z180 generates a non-maskable TRAP interrupt when an undefined op-code fetch occurs. This feature can be used to increase software reliability, implement an "extended" instruction set, or both. TRAP may occur during op-code fetch cycles and also if an undefined op-code is fetched during the interrupt acknowledge cycle for  $\overline{INT}_0$  when Mode 0 is used.

When a TRAP interrupt occurs the Z180 operates as follows.

- (1) The TRAP bit in the Interrupt TRAP/Control (ITC) register is set to 1.
- (2) The current PC (Program Counter) value, reflecting location of the undefined op-code, is saved on the stack.
- (3) The Z180 vectors to logical address 0. Note that if logical address 0000H is mapped to physical address 00000H, the vector is the same as for RESET. In this case, testing the TRAP bit in ITC will reveal whether the restart at physical address 00000H was caused by RESET or TRAP.

**External Interrupts.** The Z180 has four external hardware interrupt inputs.

- (1)  $\overline{NMI}$  - Non-maskable Interrupt
- (2)  $\overline{INT}_0$  - Maskable Interrupt Level 0
- (3)  $\overline{INT}_1$  - Maskable Interrupt Level 1
- (4)  $\overline{INT}_2$  - Maskable Interrupt Level 2

$\overline{NMI}$ ,  $\overline{INT}_1$  and  $\overline{INT}_2$  have fixed interrupt response modes.  $\overline{INT}_0$  has 3 different software programmable interrupt response modes - Mode 0, Mode 1 and Mode 2.

**$\overline{NMI}$  - Non-Maskable Interrupt.** The  $\overline{NMI}$  interrupt input is edge sensitive and cannot be masked by software. When  $\overline{NMI}$  is detected, the Z180 operates as follows.

- (1) DMAC operation is suspended by the clearing of the DME (DMA Main Enable) bit in DCNTL.
- (2) The PC is pushed onto the stack.
- (3) The contents of IEF<sub>1</sub> are copied to IEF<sub>2</sub>. This saves the interrupt reception state that existed prior to  $\overline{NMI}$ .
- (4) IEF<sub>1</sub> is cleared to 0. This disables all external and internal maskable interrupts (i.e. all interrupts except  $\overline{NMI}$  and TRAP).
- (5) Execution commences at logical address 0066H.

The last instruction of an  $\overline{NMI}$  service routine should be RETN (Return from Non-maskable Interrupt). This restores the stacked PC, allowing the interrupted program to continue.

#### $\overline{INT}_0$ - Maskable Interrupt Level 0

The next highest priority external interrupt after  $\overline{NMI}$  is  $\overline{INT}_0$ .  $\overline{INT}_0$  is sampled at the falling edge of the clock state prior to T<sub>3</sub> or T<sub>1</sub> in the last machine cycle. If  $\overline{INT}_0$  is asserted LOW at the falling edge of the clock state prior to T<sub>3</sub> or T<sub>1</sub> in the last machine cycle,  $\overline{INT}_0$  is accepted. The interrupt is masked if either the IEF<sub>1</sub> flag or the ITE<sub>0</sub> (Interrupt Enable 0) bit in ITC are reset to 0.

The  $\overline{INT}_0$  interrupt is unique in that 3 programmable interrupt response modes are available - Mode 0, Mode 1 and Mode 2. The specific mode is selected with the IM 0, IM 1 and IM 2 (Set Interrupt Mode) instructions. During RESET, the Z180 is initialized to use Mode 0 for  $\overline{INT}_0$ . The 3 interrupt response modes for  $\overline{INT}_0$  are:

- (1) Mode 0 - Instruction fetch from data bus.
- (2) Mode 1 - Restart at logical address 0038H.
- (3) Mode 2 - Low byte vector table address fetch from data bus.

#### $\overline{INT}_0$ Mode 0

During the interrupt acknowledge cycle, an instruction is fetched from the data bus (D0-D7) at the rising edge of T<sub>3</sub>. Often, this instruction is one of the eight single byte RST (RESTART) instructions which stack the PC and restart execution at a fixed logical address. However, multibyte instructions can be processed if the interrupt acknowledging device can provide a multibyte response. Unlike all other interrupts, the PC is not automatically stacked.

Note that TRAP interrupt will occur if an invalid instruction is fetched during Mode 0 interrupt acknowledge.

#### $\overline{INT}_0$ Mode 1

When  $\overline{INT}_0$  is received, the PC is stacked and instruction execution restarts at logical address 0038H. Both IEF<sub>1</sub> and IEF<sub>2</sub> flags are reset to 0, disabling all maskable interrupts. The interrupt service routine should normally terminate with the EI (Enable Interrupts) instruction followed by the RETI (Return from Interrupt) instruction, to reenable the interrupts.

#### $\overline{INT}_0$ Mode 2

This method determines the restart address by reading the contents of a table residing in memory. The vector table consists of up to 128 two-byte restart addresses stored in low byte, high byte order.

The vector table address is located on 256 byte boundaries in the 64K byte logical address space programmed in the 8-bit Interrupt Vector Register (I).

During the  $\overline{INT}_0$  Mode 2 acknowledge cycle, the low-order 8 bits of the vector is fetched from the data bus at the rising edge of T<sub>3</sub> and the CPU acquires the 16-bit vector.

Next, the PC is stacked. Finally, the 16-bit restart address is fetched from the vector table and execution begins at that address.

Note that external vector acquisition is indicated by both  $\overline{M1}$  and  $\overline{IORQ}$  LOW. Two wait states ( $T_w$ ) are automatically inserted for external vector fetch cycles.

### $\overline{INT}_1$ , $\overline{INT}_2$

The operation of external interrupts  $\overline{INT}_1$  and  $\overline{INT}_2$  is a vector mode similar to  $\overline{INT}_0$  Mode 2. The difference is that  $\overline{INT}_1$  and  $\overline{INT}_2$  generate the low-order byte of vector table address using the IL (Interrupt Vector Low) register rather than fetching it from the data bus. This is also the interrupt response sequence used for all internal interrupts (except TRAP).

**Internal Interrupts.** Internal interrupts (except TRAP) use the same vectored response mode as  $\overline{INT}_1$  and  $\overline{INT}_2$ . Internal interrupts are globally masked by  $IEF_1 = 0$ . Individual internal interrupts are enabled/disabled by programming each individual I/O (PRT, DMAC, CSI/O, ASCII) control register. The lower vector of  $\overline{INT}_1$ ,  $\overline{INT}_2$  and internal interrupt are summarized in Table 2.

Interrupt Source	Priority	IL		Fixed Code					
		b7	b6	b5	b4	b3	b2	b1	b0
$\overline{INT}_1$	Highest ↑ ↓ Lowest	*	*	*	0	0	0	0	0
$\overline{INT}_2$		*	*	*	0	0	0	1	0
PRT channel 0		*	*	*	0	0	1	0	0
PRT channel 1		*	*	*	0	0	1	1	0
DMA channel 0		*	*	*	0	1	0	0	0
DMA channel 1		*	*	*	0	1	0	1	0
CSI/O		*	*	*	0	1	1	0	0
ASCII channel 0		*	*	*	0	1	1	1	0
ASCII channel 1		*	*	*	1	0	0	0	0

\* Programmable

Table 2. Vector Table

### RETI Instruction Sequence:

When the EDH/4DH sequence is fetched by the Z180, it is recognized as the RETI instruction sequence. The Z180 will then refetch the RETI instruction with 4  $T$ -states in the EDH cycle to allow the Z80 peripherals time to decode that cycle (See Figure 21). This allows the internal interrupt structure of the peripheral to properly decode the instruction and behave accordingly.

The  $M1E$  bit of the Operation Mode Control Register (OMCR) should be set to 0 so that  $\overline{M1}$  signal is active only during the refetch of the RETI instruction sequence. This is the desired operation when Z80 peripherals are connected to the Z180.

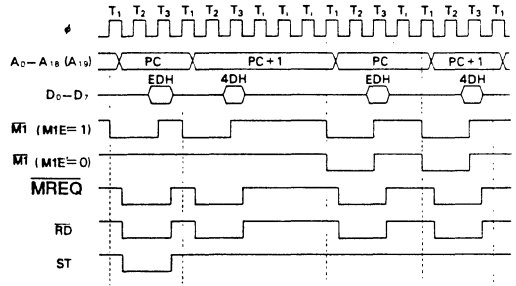


Figure 21. RETI Instruction Sequence

The RETI instruction takes 22  $T$ -states and 10 machine cycles.

**Interrupt Control Registers and Flags.** The Z180 has three registers and two flags which are associated with interrupt processing.

Function	Name	Access Method
(1) Interrupt Vector High	I	LD A, I and LD I, A instructions (addr=33H)
(2) Interrupt Vector Low	IL	I/O instruction (addr=33H)
(3) Interrupt/Trap Control	ITC	I/O instruction (addr=34H)
(4) Interrupt Enable Flag	$IEF_1, IEF_2$	EI and DI 1,2

### Interrupt Enable/Disable Operation

Two flags,  $IEF_1$  and  $IEF_2$ , are used to signal the Z180 CPU interrupt status.  $IEF_1$  controls the overall enabling and disabling of all internal and external maskable interrupts (i.e. all interrupts except NMI and TRAP).

If  $IEF_1 = 0$ , all maskable interrupts are disabled.  $IEF_1$  can be reset to 0 by the DI (Disable Interrupts) instruction and set to 1 by the EI (Enable Interrupts) instruction.

The purpose of  $IEF_2$  is to correctly manage the occurrence of NMI. During NMI, the prior interrupt reception state is saved and all maskable interrupts are automatically disabled ( $IEF_1$  copied to  $IEF_2$  and then  $IEF_1$  cleared to 0). At the end of the NMI interrupt service routine, execution of the RETN (Return from Non-maskable Interrupt) will automatically restore the interrupt receiving state (by copying  $IEF_2$  to  $IEF_1$ ) prior to the occurrence of NMI.

$IEF_2$  state can be reflected in the P/V bit of the CPU Status Register by executing LD A, I or LD A, R instructions.

CPU Operation	IEF <sub>1</sub>	IEF <sub>2</sub>	REMARKS
RESET	0	0	Inhibits the interrupt except NMI and TRAP.
NMI	0	IEF <sub>1</sub>	Copies the contents of IEF <sub>1</sub> to IEF <sub>2</sub> .
RETN	IEF <sub>2</sub>	not affected	Returns from the NMI service routine.
Interrupt except NMI and TRAP	0	0	Inhibits the interrupt except NMI and TRAP.
RETI	not affected	not affected	.
TRAP	not affected	not affected	.
EI	1	1	
DI	0	0	
LD A, I	not affected	not affected	Transfers the contents of IEF <sub>2</sub> to P-V flag.
LD A, R	not affected	not affected	Transfers the contents of IEF <sub>2</sub> to P-V flag.

**Table 3. State of IEF<sub>1</sub> and IEF<sub>2</sub>**

## Internal I/O Registers

The Z180 internal I/O Registers occupy 64 I/O addresses (including reserved addresses). These registers access the internal I/O modules (ASCI, CSI/O, PRT) and control functions (DMAC, DRAM refresh, interrupts, wait state generator, MMU and I/O relocation).

To avoid address conflicts with external I/O, the Z180 internal I/O addresses can be relocated on 64 byte boundaries within the bottom 256 bytes of the 64K byte I/O address space.



## ADDITIONAL FEATURES ON Z8S180™

### STANDBY Mode

The Z8S180 MPU has been designed to be fully static. A very low power programmable standby mode has been added. To enter STANDBY mode:

1. Set the standby enable bit (D6 of the CPU Control Register, I/O Address = 1FH)
2. Execute the SLP instruction

When the part is in STANDBY mode, it behaves similar to the SYSTEM STOP mode which currently exists on the Z180 MPU, except the STANDBY mode stops the external oscillator, internal clocks and reduce power consumption to less than 10  $\mu$ A.

Since the external oscillator has been stopped, a restart of the oscillator requires a period of time for stabilization. A 20-bit counter has been added in the Z8S180 to allow for

oscillator stabilization. When the part receives an external IRQ or BUSREQ during STANDBY mode, the oscillator is restarted and the timer counts down  $2^{19}$  counts before acknowledgement is sent to the interrupt source.

The following is a description of how the part exits STANDBY for different interrupts and modes of operation.

### STANDBY Mode Exit With RESET

The /RESET input is to be asserted for a duration long enough for the crystal oscillator to stabilize (10 ms MAX) to exit from the STANDBY mode. When /RESET is de-asserted, it goes through the normal reset timing to start instruction execution at address (logical and physical) 0000H.

The clocking is resumed within the Z8S180 and at the system clock output after /RESET is asserted, when the crystal oscillator is restarted but not yet stabilized.

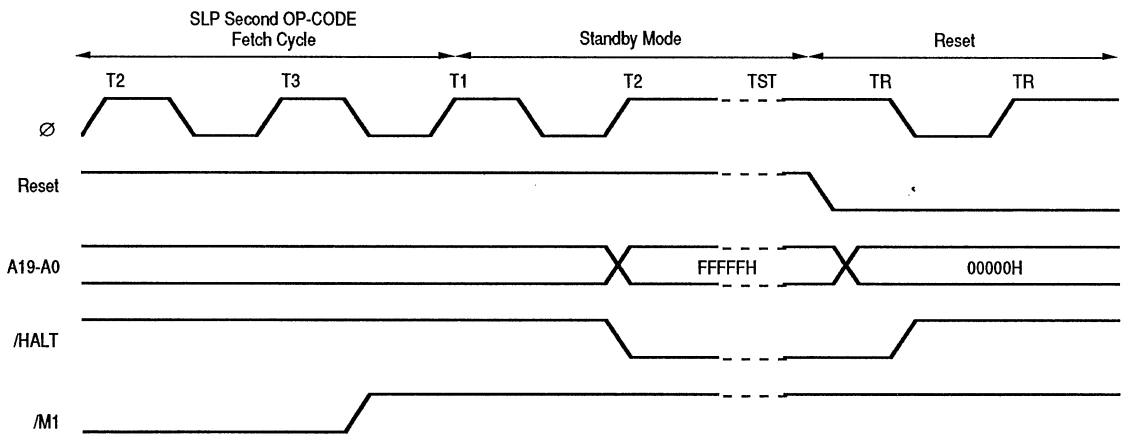


Figure A. Z8S180 Standby Mode Timing With Reset Exit

### STANDBY Mode Exit With BUS REQUEST

Optionally, if the BREXT bit (D5 of CPU Control Register) is set to 1, the Z8S180 exits STANDBY mode when the /BUSREQ input is asserted. The crystal oscillator is then restarted. An internal counter automatically provides time for the oscillator to stabilize, before the internal clocking and the system clock output of the Z8S180 are resumed.

The Z8S180 relinquishes the system bus after the clocking is resumed by:

- Tri-Stating the address outputs A19 through A0
- Tri-Stating the bus Control outputs /MREQ, /IORQ, /RD and /WR
- Asserting /BUSACK

The Z8S180 will regain the system bus when /BUSREQ is deactivated. The address outputs and the bus control outputs will then be driven High. The STANDBY mode is then resumed.

If the BREXT bit of the CPU Control Register (CCR) is cleared, asserting the /BUSREQ would not cause the Z8S180 to exit STANDBY mode.

If STANDBY mode is exited, due to a reset or an external interrupt, the Z8S180 will keep relinquished from the system bus as long as the /BUSREQ is active.

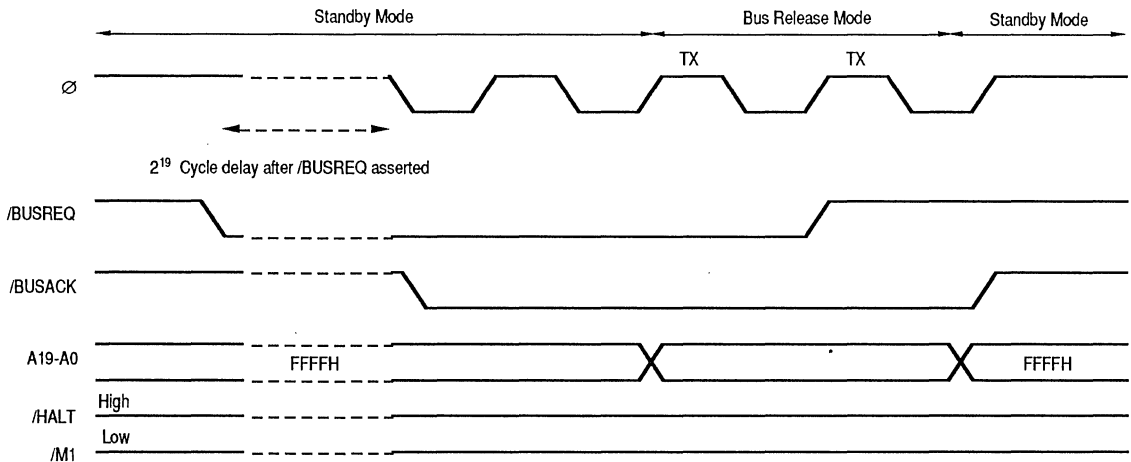


Figure B. Z8S180 Standby Mode Exit With Bus Request

### STANDBY Mode Exit With EXTERNAL INTERRUPTS

STANDBY mode can be exited by asserting input /NMI. The STANDBY mode may also exit by asserting /INT0, /INT1 or /INT2, depending on the conditions specified in the following paragraphs. If exit conditions are met, the internal counter provides time for the crystal oscillator to stabilize, before the internal clocking and the system clock output within the Z8S180 are resumed.

#### a. Exit with Non-Maskable Interrupts

If /NMI is asserted, the CPU begins a normal NMI interrupt acknowledge sequence after clocking resumes.

#### b. Exit with External Maskable Interrupts

If an external maskable interrupt input is asserted, the CPU responds according to the status of the Global Interrupt Enable Flag IEF1 (determined by the ITE1 bit) and the settings of the corresponding interrupt enable bit in the Interrupt/Trap Control Register (ITC: I/O Address = 34H):

1. If an interrupt source is disabled in the ITC, asserting the corresponding interrupt input would not cause the Z8S180 to exit STANDBY mode. This is true regardless of the state of the Global Interrupt Enable Flag IEF1.
2. If the Global Interrupt Flag IEF1 is set to 1, and if an interrupt source is enabled in the ITC, asserting the corresponding interrupt input causes the Z8S180 to exit STANDBY mode.

The CPU will perform an interrupt acknowledge sequence appropriate to the input being asserted when clocking is resumed if:

- the interrupt input follows the normal interrupt daisy-chain protocol
- the interrupt source is active until the acknowledge cycle is completed

3. If the Global Interrupt Flag IEF1 is disabled, i.e., reset to 0, and if an interrupt source is enabled in the ITC, asserting the corresponding interrupt input still causes the Z8S180 to exit STANDBY mode.

The CPU proceeds to fetch and execute instructions that follows the SLP instruction when clocking is resumed.

If the external maskable interrupt input is not active until clocking resumes, the Z8S180 will not exit STANDBY mode.

If the Non-Maskable Interrupt (/NMI) is not active until clocking resumes, the Z8S180 still exits the STANDBY mode even if the interrupt sources go away before the timer times out. It is because /NMI is edge-triggered. The condition is latched internally once /NMI is asserted Low.

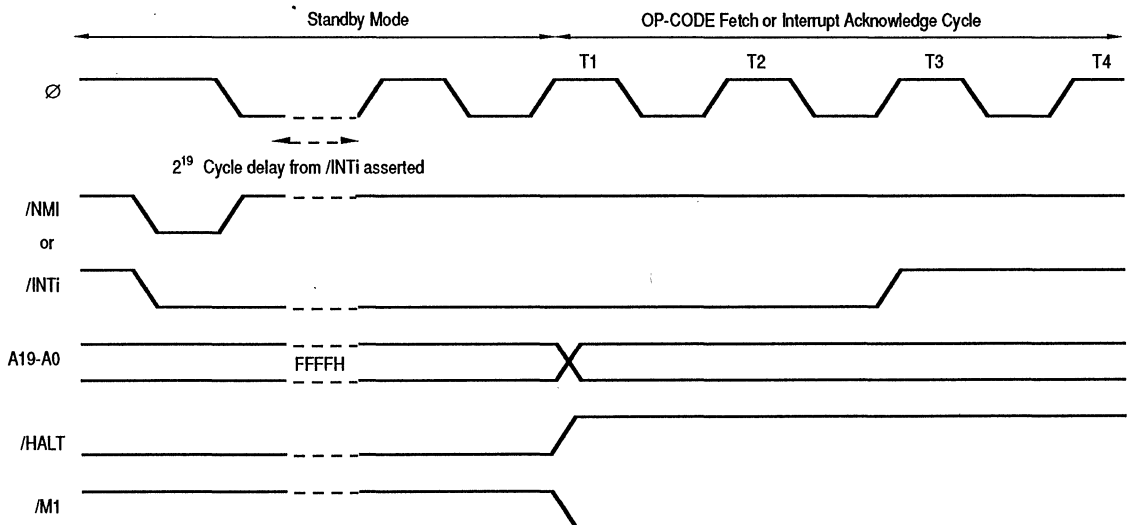


Figure C. Z8S180 Standby Mode Exit With External Interrupts

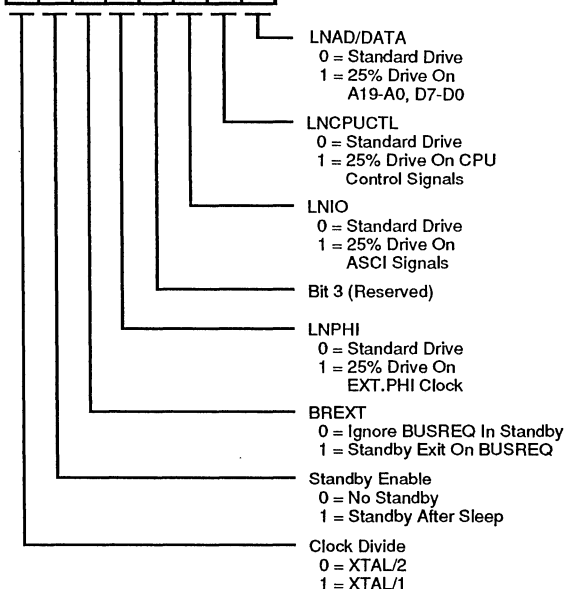
## CPU Control Register

The Z8S180 has an additional register that allows the programmer to select options. This directly affects the CPU performance as well as controlling the STANDBY operating mode of the chip. The CPU CONTROL REGISTER (CCR) allows the programmer to change the divide-by-two internal clocks to divide-by-one. In addition, applications where EMI noise is a problem, the Z8S180 can reduce the output drivers on selected groups of pins to 20% of normal pad driver capability which reduces the EMI noise generated by the part to minimal.

### CPU Control Register (CCR)

D7 D6 D5 D4 D3 D2 D1 D0

0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---



**Figure D. Z8S180 Only CPU Control Register (CCR)  
Address \$1F (Reset Values Shown)**

**Bit 7. Clock Divide Select.** Bit 7 of the CCR allows the programmer to set the internal clock to divide the external clock by 2 if the bit is 0, and divide by 1 if the bit is 1.

Upon reset this bit is set to 0 and the part is in divide-by-2 mode. Since the on-board oscillator is not guaranteed to operate above 20 MHz, an external source must be used to achieve the maximum 20 MHz operation of the part, i.e., an external clock at 40 MHz with 50% duty cycle.

If the external oscillator is used in divide-by-1 mode, the duty cycle of the external oscillator should be as close to 50% as possible. A maximum 60%/40% or 40%/60% duty cycle is permissible at 10 MHz.

**Bit 6. STANDBY Enable.** This bit is used for enabling/disabling the STANDBY mode. Setting this bit to 1 enables the STANDBY mode. The Z8S180 will enter STANDBY after fetching the second opcode of a SLP instruction if the IOSTOP bit is set.

**Bit 5. BREXT.** This bit controls the ability of the Z8S180 to honor a bus request during STANDBY mode. If this bit is set to 1 and the part is in STANDBY mode, a BUSREQ is honored after the clock stabilization timer is timed out.

**Bit 4. LNPHI.** This bit controls the drive capability on the PHI Clock output. If this bit is set to 1, the PHI Clock output is reduced to 25% of its drive capability.

**Bit 3. Reserved.**

### CPU Control Register (Continued)

**Bit 2. LNIO.** This bit controls the drive capability of the external I/O pins of the Z8S180. When this bit is set to 1, the output driving capability of the following pins is reduced to 25% of the original drive capability:

- CKS
- RXS/CTS1
- TXS
- CKA1/TEND0
- TXA1
- CKA0/DREQ0
- TXA0

**Bit 1. LNCPUCTL.** This bit controls the drive capability of the CPU Control pins. When this bit is set to 1, the output driving capability of the following pins is reduced to 25% of the original drive capability:

- /BUSACK
- /RD
- /WR
- /M1
- E
- /MREQ
- /IORQ
- /RFSH
- /HALT
- /TEND1

**Bit 0. LNAD/DATA.** This bit controls the drive capability of the Address/Data bus output drivers. If this bit is set to 1, the output driving capability of the Address and Data bus output is reduced to 25% of its original drive capability.

# Internal I/O Registers

By programming IOA7 and IOA6 in the I/O control register, internal I/O register addresses are relocatable within ranges from 0000H to 00FFH in the I/O address space.

REGISTER	MNEMONICS	ADDRESS	REMARKS																											
ASCI Control Register A Channel 0 : CNTLA0		0 0	<table border="1"> <thead> <tr> <th>bit</th> <th>MPE</th> <th>RE</th> <th>TE</th> <th>RTS0</th> <th>MPBR/EFR</th> <th>MOD2</th> <th>MOD1</th> <th>MOD0</th> </tr> </thead> <tbody> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>invalid</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </tbody> </table> <p>                     MODE Selection                      Multi Processor Bit Receive/ Error Flag Reset                      Request To Send                      Transmit Enable                      Receive Enable                      Multi Processor Enable                 </p>	bit	MPE	RE	TE	RTS0	MPBR/EFR	MOD2	MOD1	MOD0	during RESET	0	0	0	1	invalid	0	0	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			bit	MPE	RE	TE	RTS0	MPBR/EFR	MOD2	MOD1	MOD0																			
during RESET	0	0	0	1	invalid	0	0	0																						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																						
ASCI Control Register A Channel 1 : CNTLA1		0 1	<table border="1"> <thead> <tr> <th>bit</th> <th>MPE</th> <th>RE</th> <th>TE</th> <th>CKA1D</th> <th>MPBR/EFR</th> <th>MOD2</th> <th>MOD1</th> <th>MOD0</th> </tr> </thead> <tbody> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>invalid</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </tbody> </table> <p>                     MODE Selection                      Multi Processor Bit Receive/ Error Flag Reset                      CKA1 Disable                      Transmit Enable                      Receive Enable                      Multi Processor Enable                 </p>	bit	MPE	RE	TE	CKA1D	MPBR/EFR	MOD2	MOD1	MOD0	during RESET	0	0	0	1	invalid	0	0	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			bit	MPE	RE	TE	CKA1D	MPBR/EFR	MOD2	MOD1	MOD0																			
during RESET	0	0	0	1	invalid	0	0	0																						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																						
			<p>MOD2, 1, 0</p> <p>0 0 0 Start + 7 bit Data + 1 Stop</p> <p>0 0 1 Start + 7 bit Data + 2 Stop</p> <p>0 1 0 Start + 7 bit Data + Parity + 1 Stop</p> <p>0 1 1 Start + 7 bit Data + Parity + 2 Stop</p> <p>1 0 0 Start + 8 bit Data + 1 Stop</p> <p>1 0 1 Start + 8 bit Data + 2 Stop</p> <p>1 1 0 Start + 8 bit Data + Parity + 1 Stop</p> <p>1 1 1 Start + 8 bit Data + Parity + 2 Stop</p>																											
ASCI Control Register B Channel 0 : CNTLB0		0 2	<table border="1"> <thead> <tr> <th>bit</th> <th>MPBT</th> <th>MP</th> <th>CTS/PS</th> <th>PEO</th> <th>DR</th> <th>SS2</th> <th>SS1</th> <th>SS0</th> </tr> </thead> <tbody> <tr> <td>during RESET</td> <td>invalid</td> <td>0</td> <td>.</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </tbody> </table> <p>                     Clock Source and Speed Select                      Divide Ratio                      Parity Even or Odd                      Clear To Send/Prescale                      Multi Processor                      Multi Processor Bit Transmit                 </p>	bit	MPBT	MP	CTS/PS	PEO	DR	SS2	SS1	SS0	during RESET	invalid	0	.	0	0	1	1	1	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			bit	MPBT	MP	CTS/PS	PEO	DR	SS2	SS1	SS0																			
during RESET	invalid	0	.	0	0	1	1	1																						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																						
			<p>• CTS : Depending on the condition of CTS Pin.                      PS : Cleared to 0.</p>																											

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS																																																																																					
ASCI Control Register B Channel 1 : CNTLB1		0 3	<table border="1"> <thead> <tr> <th>bit</th> <th>MPBT</th> <th>MP</th> <th>CTS/ PS</th> <th>PEO</th> <th>DR</th> <th>SS2</th> <th>SS1</th> <th>SS0</th> </tr> </thead> <tbody> <tr> <td>during RESET</td> <td>invalid</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </tbody> </table> <p>           Clock Source and Speed Select            Divide Ratio            Parity Even or Odd            Clear To Send/Prescale            Multi Processor            Multi Processor Bit Transmit         </p> <table border="1"> <thead> <tr> <th rowspan="2">General divide ratio</th> <th colspan="2">PS=0 (divide ratio= 10)</th> <th colspan="2">PS=1 (divide ratio= 30)</th> </tr> <tr> <th>SS2,1,0</th> <th>DR=0 (× 16)</th> <th>DR=1 (× 64)</th> <th>DR=0 (× 16)</th> <th>DR=1 (× 64)</th> </tr> </thead> <tbody> <tr> <td>0 0 0</td> <td><math>\phi + 160</math></td> <td><math>\phi + 640</math></td> <td><math>\phi + 480</math></td> <td><math>\phi + 1920</math></td> <td></td> </tr> <tr> <td>0 0 1</td> <td><math>+ 320</math></td> <td><math>+ 1280</math></td> <td><math>+ 960</math></td> <td><math>+ 3840</math></td> <td></td> </tr> <tr> <td>0 1 0</td> <td><math>+ 640</math></td> <td><math>+ 2560</math></td> <td><math>+ 1920</math></td> <td><math>+ 7680</math></td> <td></td> </tr> <tr> <td>0 1 1</td> <td><math>+ 1280</math></td> <td><math>+ 5120</math></td> <td><math>+ 3840</math></td> <td><math>+ 15360</math></td> <td></td> </tr> <tr> <td>1 0 0</td> <td><math>+ 2560</math></td> <td><math>+ 10240</math></td> <td><math>+ 7680</math></td> <td><math>+ 30720</math></td> <td></td> </tr> <tr> <td>1 0 1</td> <td><math>+ 5120</math></td> <td><math>+ 20480</math></td> <td><math>+ 15360</math></td> <td><math>+ 61440</math></td> <td></td> </tr> <tr> <td>1 1 0</td> <td><math>+ 10240</math></td> <td><math>+ 40960</math></td> <td><math>+ 30720</math></td> <td><math>+ 122880</math></td> <td></td> </tr> <tr> <td>1 1 1</td> <td colspan="4">External clock (frequency &lt; <math>\phi + 40</math>)</td> <td></td> </tr> </tbody> </table>	bit	MPBT	MP	CTS/ PS	PEO	DR	SS2	SS1	SS0	during RESET	invalid	0	0	0	0	1	1	1	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	General divide ratio	PS=0 (divide ratio= 10)		PS=1 (divide ratio= 30)		SS2,1,0	DR=0 (× 16)	DR=1 (× 64)	DR=0 (× 16)	DR=1 (× 64)	0 0 0	$\phi + 160$	$\phi + 640$	$\phi + 480$	$\phi + 1920$		0 0 1	$+ 320$	$+ 1280$	$+ 960$	$+ 3840$		0 1 0	$+ 640$	$+ 2560$	$+ 1920$	$+ 7680$		0 1 1	$+ 1280$	$+ 5120$	$+ 3840$	$+ 15360$		1 0 0	$+ 2560$	$+ 10240$	$+ 7680$	$+ 30720$		1 0 1	$+ 5120$	$+ 20480$	$+ 15360$	$+ 61440$		1 1 0	$+ 10240$	$+ 40960$	$+ 30720$	$+ 122880$		1 1 1	External clock (frequency < $\phi + 40$ )				
			bit	MPBT	MP	CTS/ PS	PEO	DR	SS2	SS1	SS0																																																																													
during RESET	invalid	0	0	0	0	1	1	1																																																																																
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																																																																																
General divide ratio	PS=0 (divide ratio= 10)		PS=1 (divide ratio= 30)																																																																																					
	SS2,1,0	DR=0 (× 16)	DR=1 (× 64)	DR=0 (× 16)	DR=1 (× 64)																																																																																			
0 0 0	$\phi + 160$	$\phi + 640$	$\phi + 480$	$\phi + 1920$																																																																																				
0 0 1	$+ 320$	$+ 1280$	$+ 960$	$+ 3840$																																																																																				
0 1 0	$+ 640$	$+ 2560$	$+ 1920$	$+ 7680$																																																																																				
0 1 1	$+ 1280$	$+ 5120$	$+ 3840$	$+ 15360$																																																																																				
1 0 0	$+ 2560$	$+ 10240$	$+ 7680$	$+ 30720$																																																																																				
1 0 1	$+ 5120$	$+ 20480$	$+ 15360$	$+ 61440$																																																																																				
1 1 0	$+ 10240$	$+ 40960$	$+ 30720$	$+ 122880$																																																																																				
1 1 1	External clock (frequency < $\phi + 40$ )																																																																																							
ASCI Status Register Channel 0 : STAT0		0 4	<table border="1"> <thead> <tr> <th>bit</th> <th>RDRF</th> <th>OVRN</th> <th>PE</th> <th>FE</th> <th>RIE</th> <th>DCD<sub>0</sub></th> <th>TDRE</th> <th>TIE</th> </tr> </thead> <tbody> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>*</td> <td>**</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R</td> <td>R</td> <td>R</td> <td>R</td> <td>R/W</td> <td>R</td> <td>R</td> <td>R/W</td> </tr> </tbody> </table> <p>           Transmit Interrupt Enable            Transmit Data Register Empty            Data Carrier Detect            Receive Interrupt Enable            Framing Error            Parity Error            Over Run Error            Receive Data Register Full            * DCD<sub>0</sub> : Depending on the condition of DCD<sub>0</sub> Pin.            ** CTS<sub>0</sub> Pin            TDRE            L 1            H 0         </p>	bit	RDRF	OVRN	PE	FE	RIE	DCD <sub>0</sub>	TDRE	TIE	during RESET	0	0	0	0	0	*	**	0	R/W	R	R	R	R	R/W	R	R	R/W																																																										
			bit	RDRF	OVRN	PE	FE	RIE	DCD <sub>0</sub>	TDRE	TIE																																																																													
during RESET	0	0	0	0	0	*	**	0																																																																																
R/W	R	R	R	R	R/W	R	R	R/W																																																																																
ASCI Status Register Channel 1 : STAT1		0 5	<table border="1"> <thead> <tr> <th>bit</th> <th>RDRF</th> <th>OVRN</th> <th>PE</th> <th>FE</th> <th>RIE</th> <th>CTS1E</th> <th>TDRE</th> <th>TIE</th> </tr> </thead> <tbody> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R</td> <td>R</td> <td>R</td> <td>R</td> <td>R/W</td> <td>R/W</td> <td>R</td> <td>R/W</td> </tr> </tbody> </table> <p>           Transmit Interrupt Enable            Transmit Data Register Empty            CTS1 Enable            Receive Interrupt Enable            Framing Error            Parity Error            Over Run Error            Receive Data Register Full         </p>	bit	RDRF	OVRN	PE	FE	RIE	CTS1E	TDRE	TIE	during RESET	0	0	0	0	0	0	1	0	R/W	R	R	R	R	R/W	R/W	R	R/W																																																										
			bit	RDRF	OVRN	PE	FE	RIE	CTS1E	TDRE	TIE																																																																													
during RESET	0	0	0	0	0	0	1	0																																																																																
R/W	R	R	R	R	R/W	R/W	R	R/W																																																																																

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS																																																																														
ASCII Transmit Data Register Channel 0	: TDRO	0 6	<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">bit</div> <table border="1" style="border-collapse: collapse; text-align: center;"> <tr> <td>EF</td> <td>EIE</td> <td>RE</td> <td>TE</td> <td>—</td> <td>SS2</td> <td>SS1</td> <td>SS0</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td></td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> </div> <p style="margin-left: 40px;">during RESET</p> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="text-align: center;"> <p>End Flag</p> <p>End Interrupt Enable</p> <p>Receive Enable</p> <p>Transmit Enable</p> </div> <div style="text-align: center;"> <p>Speed Select</p> </div> </div> <table border="1" style="border-collapse: collapse; text-align: center; margin: 10px auto;"> <thead> <tr> <th>SS2,1,0</th> <th>Baud Rate</th> <th>SS2,1,0</th> <th>Baud Rate</th> </tr> </thead> <tbody> <tr> <td>000</td> <td><math>\phi + 20</math></td> <td>100</td> <td><math>\phi + 320</math></td> </tr> <tr> <td>001</td> <td>+ 40</td> <td>101</td> <td>+ 640</td> </tr> <tr> <td>010</td> <td>+ 80</td> <td>110</td> <td>+ 1280</td> </tr> <tr> <td>011</td> <td>+ 160</td> <td>111</td> <td>External (frequency &lt; + 20)</td> </tr> </tbody> </table> <div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">bit</div> <table border="1" style="border-collapse: collapse; text-align: center;"> <tr> <td>TIF1</td> <td>TIF0</td> <td>TIE1</td> <td>TIE0</td> <td>TOC1</td> <td>TOC0</td> <td>TDE1</td> <td>TDE0</td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R</td> <td>R</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> </div> <p style="margin-left: 40px;">during RESET</p> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="text-align: center;"> <p>Timer Interrupt Flag 1,0</p> <p>Timer Interrupt Enable 1,0</p> <p>Timer Output Control 1,0</p> <p>Timer Down Count Enable 1,0</p> </div> </div> <table border="1" style="border-collapse: collapse; text-align: center; margin: 10px auto;"> <thead> <tr> <th>TOC1,0</th> <th>A<sub>18</sub>/TOUT</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>Inhibited</td> </tr> <tr> <td>01</td> <td>Toggle</td> </tr> <tr> <td>10</td> <td>0</td> </tr> <tr> <td>11</td> <td>1</td> </tr> </tbody> </table>	EF	EIE	RE	TE	—	SS2	SS1	SS0	0	0	0	0	1	1	1	1	R/W	R/W	R/W	R/W		R/W	R/W	R/W	SS2,1,0	Baud Rate	SS2,1,0	Baud Rate	000	$\phi + 20$	100	$\phi + 320$	001	+ 40	101	+ 640	010	+ 80	110	+ 1280	011	+ 160	111	External (frequency < + 20)	TIF1	TIF0	TIE1	TIE0	TOC1	TOC0	TDE1	TDE0	0	0	0	0	0	0	0	0	R	R	R/W	R/W	R/W	R/W	R/W	R/W	TOC1,0	A <sub>18</sub> /TOUT	00	Inhibited	01	Toggle	10	0	11	1
EF	EIE	RE		TE	—	SS2	SS1	SS0																																																																									
0	0	0		0	1	1	1	1																																																																									
R/W	R/W	R/W		R/W		R/W	R/W	R/W																																																																									
SS2,1,0	Baud Rate	SS2,1,0		Baud Rate																																																																													
000	$\phi + 20$	100		$\phi + 320$																																																																													
001	+ 40	101	+ 640																																																																														
010	+ 80	110	+ 1280																																																																														
011	+ 160	111	External (frequency < + 20)																																																																														
TIF1	TIF0	TIE1	TIE0	TOC1	TOC0	TDE1	TDE0																																																																										
0	0	0	0	0	0	0	0																																																																										
R	R	R/W	R/W	R/W	R/W	R/W	R/W																																																																										
TOC1,0	A <sub>18</sub> /TOUT																																																																																
00	Inhibited																																																																																
01	Toggle																																																																																
10	0																																																																																
11	1																																																																																
ASCII Transmit Data Register Channel 1	: TDR1	0 7																																																																															
ASCII Receive Data Register Channel 0	: TSRO	0 8																																																																															
ASCII Receive Data Register Channel 1	: TSR1	0 9																																																																															
CS/V0 Control Register	: CNTR	0 A																																																																															
CS/V0 Transmit/Receive Data Register	: TRDR	0 B																																																																															
Timer Data Register Channel 0L	: TMDROL	0 C																																																																															
Timer Data Register Channel 0H	: TMDROH	0 D																																																																															
Timer Reload Register Channel 0L	: RLDROL	0 E																																																																															
Timer Reload Register Channel 0H	: RLDROH	0 F																																																																															
Timer Control Register	: TCR	1 0																																																																															

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS																									
Timer Data Register Channel 1L : TMDR1L		1 4																										
Timer Data Register Channel 1H : TMDR1H		1 5																										
Timer Reload Register Channel 1L : RLDR1L		1 6																										
Timer Reload Register Channel 1H : RLDR1H		1 7																										
Free Running Counter : FRC		1 8	read only																									
DMA Source Address Register Channel 0L : SAR0L		2 0																										
DMA Source Address Register Channel 0H : SAR0H		2 1																										
DMA Source Address Register Channel 0B : SAR0B		2 2	Bits 0-2 (3) are used for SAR0B. <table border="1"> <thead> <tr> <th>A<sub>19</sub>*</th> <th>A<sub>18</sub></th> <th>A<sub>17</sub></th> <th>A<sub>16</sub></th> <th>DMA Transfer Request</th> </tr> </thead> <tbody> <tr> <td>X</td> <td>X</td> <td>0</td> <td>0</td> <td>DREQ<sub>0</sub> (external)</td> </tr> <tr> <td>X</td> <td>X</td> <td>0</td> <td>1</td> <td>RDRO (ASCI0)</td> </tr> <tr> <td>X</td> <td>X</td> <td>1</td> <td>0</td> <td>RDR1 (ASCI1)</td> </tr> <tr> <td>X</td> <td>X</td> <td>1</td> <td>1</td> <td>Not Used</td> </tr> </tbody> </table>	A <sub>19</sub> *	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	DMA Transfer Request	X	X	0	0	DREQ <sub>0</sub> (external)	X	X	0	1	RDRO (ASCI0)	X	X	1	0	RDR1 (ASCI1)	X	X	1	1	Not Used
A <sub>19</sub> *	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	DMA Transfer Request																								
X	X	0	0	DREQ <sub>0</sub> (external)																								
X	X	0	1	RDRO (ASCI0)																								
X	X	1	0	RDR1 (ASCI1)																								
X	X	1	1	Not Used																								
DMA Destination Address Register Channel 0L : DAR0L		2 3																										
DMA Destination Address Register Channel 0H : DAR0H		2 4																										
DMA Destination Address Register Channel 0B : DAR0B		2 5	Bits 0-2 (3) are used for DAR0B. <table border="1"> <thead> <tr> <th>A<sub>19</sub>*</th> <th>A<sub>18</sub></th> <th>A<sub>17</sub></th> <th>A<sub>16</sub></th> <th>DMA Transfer Request</th> </tr> </thead> <tbody> <tr> <td>X</td> <td>X</td> <td>0</td> <td>0</td> <td>DREQ<sub>0</sub> (external)</td> </tr> <tr> <td>X</td> <td>X</td> <td>0</td> <td>1</td> <td>TDRO (ASCI0)</td> </tr> <tr> <td>X</td> <td>X</td> <td>1</td> <td>0</td> <td>TDR1 (ASCI1)</td> </tr> <tr> <td>X</td> <td>X</td> <td>1</td> <td>1</td> <td>Not Used</td> </tr> </tbody> </table>	A <sub>19</sub> *	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	DMA Transfer Request	X	X	0	0	DREQ <sub>0</sub> (external)	X	X	0	1	TDRO (ASCI0)	X	X	1	0	TDR1 (ASCI1)	X	X	1	1	Not Used
A <sub>19</sub> *	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	DMA Transfer Request																								
X	X	0	0	DREQ <sub>0</sub> (external)																								
X	X	0	1	TDRO (ASCI0)																								
X	X	1	0	TDR1 (ASCI1)																								
X	X	1	1	Not Used																								
DMA Byte Count Register Channel 0L : BC0L		2 6																										
DMA Byte Count Register Channel 0H : BC0H		2 7																										
DMA Memory Address Register Channel 1L : MAR1L		2 8																										
DMA Memory Address Register Channel 1H : MAR1H		2 9																										
DMA Memory Address Register Channel 1B : MAR1B		2 A	Bits 0-2 (3) are used for MAR1B.																									
DMA I/O Address Register Channel 1L : IAR1L		2 B																										
DMA I/O Address Register Channel 1H : IAR1H		2 C																										

(to be continued)

- In the R1 and Z Mask, these DMAC registers are expanded from 4 bits to 3 bits in the package version of CP-68 and FP-80.



REGISTER	MNEMONICS	ADDRESS	REMARKS																																																															
DMA Byte Count Register Channel 1L	: BCR1L	2 E																																																																
DMA Byte Count Register Channel 1H	: BCR1H	2 F																																																																
DMA Status Register	: DSTAT	3 0	<table border="1"> <tr> <td>bit</td> <td>DE1</td> <td>DE0</td> <td>DWE1</td> <td>DWE0</td> <td>DIE1</td> <td>DIE0</td> <td>—</td> <td>DME</td> </tr> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>W</td> <td>W</td> <td>R/W</td> <td>R/W</td> <td></td> <td>R</td> </tr> </table> <p>           DMA Master Enable            DMA Interrupt Enable 1,0            DMA Enable Bit Write Enable 1,0            DMA Enable ch 1,0         </p>	bit	DE1	DE0	DWE1	DWE0	DIE1	DIE0	—	DME	during RESET	0	0	1	1	0	0	1	0	R/W	R/W	R/W	W	W	R/W	R/W		R																																				
bit	DE1	DE0	DWE1	DWE0	DIE1	DIE0	—	DME																																																										
during RESET	0	0	1	1	0	0	1	0																																																										
R/W	R/W	R/W	W	W	R/W	R/W		R																																																										
DMA Mode Register	: DMODE	3 1	<table border="1"> <tr> <td>bit</td> <td>—</td> <td>—</td> <td>DM1</td> <td>DM0</td> <td>SM1</td> <td>SM0</td> <td>MMOD</td> <td>—</td> </tr> <tr> <td>during RESET</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>R/W</td> <td></td> <td></td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td></td> </tr> </table> <p>           Memory MODE Select            Ch 0 Source Mode 1,0            Ch 0 Destination Mode 1,0         </p> <table border="1"> <thead> <tr> <th>DM1, 0</th> <th>Destination</th> <th>Address</th> <th>SM1, 0</th> <th>Source</th> <th>Address</th> </tr> </thead> <tbody> <tr> <td>0 0</td> <td>M</td> <td>DAR0+1</td> <td>0 0</td> <td>M</td> <td>SAR0+1</td> </tr> <tr> <td>0 1</td> <td>M</td> <td>DAR0-1</td> <td>0 1</td> <td>M</td> <td>SAR0-1</td> </tr> <tr> <td>1 0</td> <td>M</td> <td>DAR0 fixed</td> <td>1 0</td> <td>M</td> <td>SAR0 fixed</td> </tr> <tr> <td>1 1</td> <td>I/O</td> <td>DAR0 fixed</td> <td>1 1</td> <td>I/O</td> <td>SAR0 fixed</td> </tr> </tbody> </table> <table border="1"> <thead> <tr> <th>MMOD</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Cycle Steal Mode</td> </tr> <tr> <td>1</td> <td>Burst Mode</td> </tr> </tbody> </table>	bit	—	—	DM1	DM0	SM1	SM0	MMOD	—	during RESET	1	1	0	0	0	0	0	1	R/W			R/W	R/W	R/W	R/W	R/W		DM1, 0	Destination	Address	SM1, 0	Source	Address	0 0	M	DAR0+1	0 0	M	SAR0+1	0 1	M	DAR0-1	0 1	M	SAR0-1	1 0	M	DAR0 fixed	1 0	M	SAR0 fixed	1 1	I/O	DAR0 fixed	1 1	I/O	SAR0 fixed	MMOD	Mode	0	Cycle Steal Mode	1	Burst Mode
bit	—	—	DM1	DM0	SM1	SM0	MMOD	—																																																										
during RESET	1	1	0	0	0	0	0	1																																																										
R/W			R/W	R/W	R/W	R/W	R/W																																																											
DM1, 0	Destination	Address	SM1, 0	Source	Address																																																													
0 0	M	DAR0+1	0 0	M	SAR0+1																																																													
0 1	M	DAR0-1	0 1	M	SAR0-1																																																													
1 0	M	DAR0 fixed	1 0	M	SAR0 fixed																																																													
1 1	I/O	DAR0 fixed	1 1	I/O	SAR0 fixed																																																													
MMOD	Mode																																																																	
0	Cycle Steal Mode																																																																	
1	Burst Mode																																																																	

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS																																																																				
DMA/WAIT Control Register	: DCNTL	3 2	<table border="1"> <tr> <td>bit</td> <td>MW11</td> <td>MW10</td> <td>IW11</td> <td>IW10</td> <td>DMS1</td> <td>DMS0</td> <td>DIM1</td> <td>DIM0</td> </tr> <tr> <td>during RESET</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> <p>           Memory Wait Insertion            I/O Wait Insertion            DMA Ch 1 I/O Memory Mode Select            DREQ<sub>i</sub> Select, i = 1,0         </p> <table border="1"> <tr> <td>MW11,0</td> <td>The number of wait states</td> <td>IW11,0</td> <td>The number of wait states</td> </tr> <tr> <td>00</td> <td>0</td> <td>00</td> <td>0</td> </tr> <tr> <td>01</td> <td>1</td> <td>01</td> <td>2</td> </tr> <tr> <td>10</td> <td>2</td> <td>10</td> <td>3</td> </tr> <tr> <td>11</td> <td>3</td> <td>11</td> <td>4</td> </tr> </table> <table border="1"> <tr> <td>DMS<sub>i</sub></td> <td>Sense</td> </tr> <tr> <td>1</td> <td>Edge sense</td> </tr> <tr> <td>0</td> <td>Level sense</td> </tr> </table> <table border="1"> <tr> <td>DIM1,0</td> <td>Transfer Mode</td> <td>Address Increment/Decrement</td> </tr> <tr> <td>00</td> <td>M→I/O</td> <td>MAR1+1 IAR1 fixed</td> </tr> <tr> <td>01</td> <td>M→I/O</td> <td>MAR1-1 IAR1 fixed</td> </tr> <tr> <td>10</td> <td>I/O→M</td> <td>IAR1 fixed MAR1+1</td> </tr> <tr> <td>11</td> <td>I/O→M</td> <td>IAR1 fixed MAR1-1</td> </tr> </table>	bit	MW11	MW10	IW11	IW10	DMS1	DMS0	DIM1	DIM0	during RESET	1	1	1	1	0	0	0	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	MW11,0	The number of wait states	IW11,0	The number of wait states	00	0	00	0	01	1	01	2	10	2	10	3	11	3	11	4	DMS <sub>i</sub>	Sense	1	Edge sense	0	Level sense	DIM1,0	Transfer Mode	Address Increment/Decrement	00	M→I/O	MAR1+1 IAR1 fixed	01	M→I/O	MAR1-1 IAR1 fixed	10	I/O→M	IAR1 fixed MAR1+1	11	I/O→M	IAR1 fixed MAR1-1
bit	MW11	MW10	IW11	IW10	DMS1	DMS0	DIM1	DIM0																																																															
during RESET	1	1	1	1	0	0	0	0																																																															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																																																															
MW11,0	The number of wait states	IW11,0	The number of wait states																																																																				
00	0	00	0																																																																				
01	1	01	2																																																																				
10	2	10	3																																																																				
11	3	11	4																																																																				
DMS <sub>i</sub>	Sense																																																																						
1	Edge sense																																																																						
0	Level sense																																																																						
DIM1,0	Transfer Mode	Address Increment/Decrement																																																																					
00	M→I/O	MAR1+1 IAR1 fixed																																																																					
01	M→I/O	MAR1-1 IAR1 fixed																																																																					
10	I/O→M	IAR1 fixed MAR1+1																																																																					
11	I/O→M	IAR1 fixed MAR1-1																																																																					
Interrupt Vector Low Register	: IL	3 3	<table border="1"> <tr> <td>bit</td> <td>IL7</td> <td>IL6</td> <td>IL5</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> </tr> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </table> <p>Interrupt Vector Low</p>	bit	IL7	IL6	IL5	-	-	-	-	-	during RESET	0	0	0	0	0	0	0	0	R/W	R/W	R/W	R/W																																														
bit	IL7	IL6	IL5	-	-	-	-	-																																																															
during RESET	0	0	0	0	0	0	0	0																																																															
R/W	R/W	R/W	R/W																																																																				
INT/TRAP Control Register	: ITC	3 4	<table border="1"> <tr> <td>bit</td> <td>TRAP</td> <td>UFO</td> <td>-</td> <td>-</td> <td>-</td> <td>ITE2</td> <td>ITE1</td> <td>ITE0</td> </tr> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R</td> <td></td> <td></td> <td></td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> <p>           TRAP            Undefined Fetch Object            INT Enable 2,1,0         </p>	bit	TRAP	UFO	-	-	-	ITE2	ITE1	ITE0	during RESET	0	0	1	1	1	0	0	1	R/W	R/W	R				R/W	R/W	R/W																																									
bit	TRAP	UFO	-	-	-	ITE2	ITE1	ITE0																																																															
during RESET	0	0	1	1	1	0	0	1																																																															
R/W	R/W	R				R/W	R/W	R/W																																																															
Refresh Control Register	: RCR	3 8	<table border="1"> <tr> <td>bit</td> <td>REFE</td> <td>REFW</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> <td>CYC1</td> <td>CYC0</td> </tr> <tr> <td>during RESET</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td></td> <td></td> <td></td> <td></td> <td>R/W</td> <td>R/W</td> </tr> </table> <p>           Refresh Wait State            Refresh Enable            Cycle Select         </p> <table border="1"> <tr> <td>CYC1,0</td> <td>Interval of Refresh Cycle</td> </tr> <tr> <td>00</td> <td>10 States</td> </tr> <tr> <td>01</td> <td>20</td> </tr> <tr> <td>10</td> <td>40</td> </tr> <tr> <td>11</td> <td>80</td> </tr> </table>	bit	REFE	REFW	-	-	-	-	CYC1	CYC0	during RESET	1	1	1	1	1	1	0	0	R/W	R/W	R/W					R/W	R/W	CYC1,0	Interval of Refresh Cycle	00	10 States	01	20	10	40	11	80																															
bit	REFE	REFW	-	-	-	-	CYC1	CYC0																																																															
during RESET	1	1	1	1	1	1	0	0																																																															
R/W	R/W	R/W					R/W	R/W																																																															
CYC1,0	Interval of Refresh Cycle																																																																						
00	10 States																																																																						
01	20																																																																						
10	40																																																																						
11	80																																																																						

(to be continued)

REGISTER	MNEMONICS	ADDRESS	REMARKS																											
MMU Common Base Register	: CBR	3 8	<table border="1"> <tr> <td>bit</td> <td>CB7*</td> <td>CB6</td> <td>CB5</td> <td>CB4</td> <td>CB3</td> <td>CB2</td> <td>CB1</td> <td>CB0</td> </tr> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> <p style="text-align: center;">└── MMU Common Base Register</p>	bit	CB7*	CB6	CB5	CB4	CB3	CB2	CB1	CB0	during RESET	0	0	0	0	0	0	0	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
bit	CB7*	CB6	CB5	CB4	CB3	CB2	CB1	CB0																						
during RESET	0	0	0	0	0	0	0	0																						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																						
MMU Bank Base Register	: BBR	3 9	<table border="1"> <tr> <td>bit</td> <td>BB7*</td> <td>BB6</td> <td>BB5</td> <td>BB4</td> <td>BB3</td> <td>BB2</td> <td>BB1</td> <td>BB0</td> </tr> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> <p style="text-align: center;">└── MMU Bank Base Register</p>	bit	BB7*	BB6	BB5	BB4	BB3	BB2	BB1	BB0	during RESET	0	0	0	0	0	0	0	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
bit	BB7*	BB6	BB5	BB4	BB3	BB2	BB1	BB0																						
during RESET	0	0	0	0	0	0	0	0																						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																						
MMU Common/Bank Area Register	: CBAR	3 A	<table border="1"> <tr> <td>bit</td> <td>CA3</td> <td>CA2</td> <td>CA1</td> <td>CA0</td> <td>BA3</td> <td>BA2</td> <td>BA1</td> <td>BA0</td> </tr> <tr> <td>during RESET</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> </tr> </table> <p style="text-align: center;">└── MMU Common Area Register      └── MMU Bank Area Register</p>	bit	CA3	CA2	CA1	CA0	BA3	BA2	BA1	BA0	during RESET	1	1	1	1	0	0	0	0	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
bit	CA3	CA2	CA1	CA0	BA3	BA2	BA1	BA0																						
during RESET	1	1	1	1	0	0	0	0																						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W																						
Operation Mode Control Register	: OMCR	3 E	<table border="1"> <tr> <td>bit</td> <td>M1E</td> <td>MTE</td> <td>IOC</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> </tr> <tr> <td>during RESET</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>W</td> <td>R/W</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </table> <p style="text-align: center;">└── M1 Enable      └── M1 Temporary Enable      └── I/O Compatibility</p>	bit	M1E	MTE	IOC	-	-	-	-	-	during RESET	1	1	1	1	1	1	1	1	R/W	R/W	W	R/W					
bit	M1E	MTE	IOC	-	-	-	-	-																						
during RESET	1	1	1	1	1	1	1	1																						
R/W	R/W	W	R/W																											
I/O Control Register	: ICR	3 F	<table border="1"> <tr> <td>bit</td> <td>IOA7</td> <td>IOA6</td> <td>IOSTP</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> <td>-</td> </tr> <tr> <td>during RESET</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>1</td> </tr> <tr> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td>R/W</td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </table> <p style="text-align: center;">└── I/O Address      └── I/O Stop</p>	bit	IOA7	IOA6	IOSTP	-	-	-	-	-	during RESET	0	0	0	1	1	1	1	1	R/W	R/W	R/W	R/W					
bit	IOA7	IOA6	IOSTP	-	-	-	-	-																						
during RESET	0	0	0	1	1	1	1	1																						
R/W	R/W	R/W	R/W																											

These MMU registers are expanded from 7 bits to 8 bits in the PLCC package

## Memory Management Unit (MMU)

The Z180 has an on-chip MMU which performs the translation of the CPU 64K byte (16-bit addresses 0000H to FFFFH) logical memory address space into a 1024K byte (20-bit addresses 00000H to FFFFFH) physical memory address space. Address translation occurs internally in parallel with other CPU operation.

**Logical Address Spaces.** The 64K byte CPU logical address space is interpreted by the MMU as consisting of up to three separate logical address areas, Common Area 0, Bank Area, and Common Area 1.

As shown in Fig.22, a variety of logical memory configurations are possible. The boundaries between the Common and Bank Areas can be programmed with 4K byte resolution.

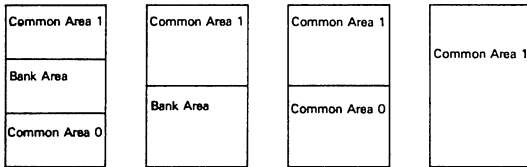


Figure 22. Logical Address Mapping Examples

Whether address translation takes place depends on the type of CPU cycle as follows.

### (1) Memory Cycles

Address Translation occurs for all memory access cycles including instruction and operand fetches, memory data reads and writes, hardware interrupt vector fetch, and software interrupt restarts.

### (2) I/O Cycles

The MMU is logically bypassed for I/O cycles. The 16-bit logical I/O address space corresponds directly with the 16-bit physical I/O address space. The four high-order bits (A16-A19) of the physical address are always 0 during I/O cycles.

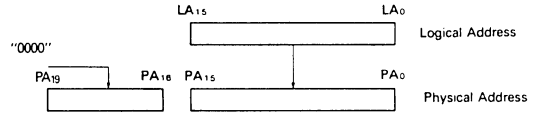
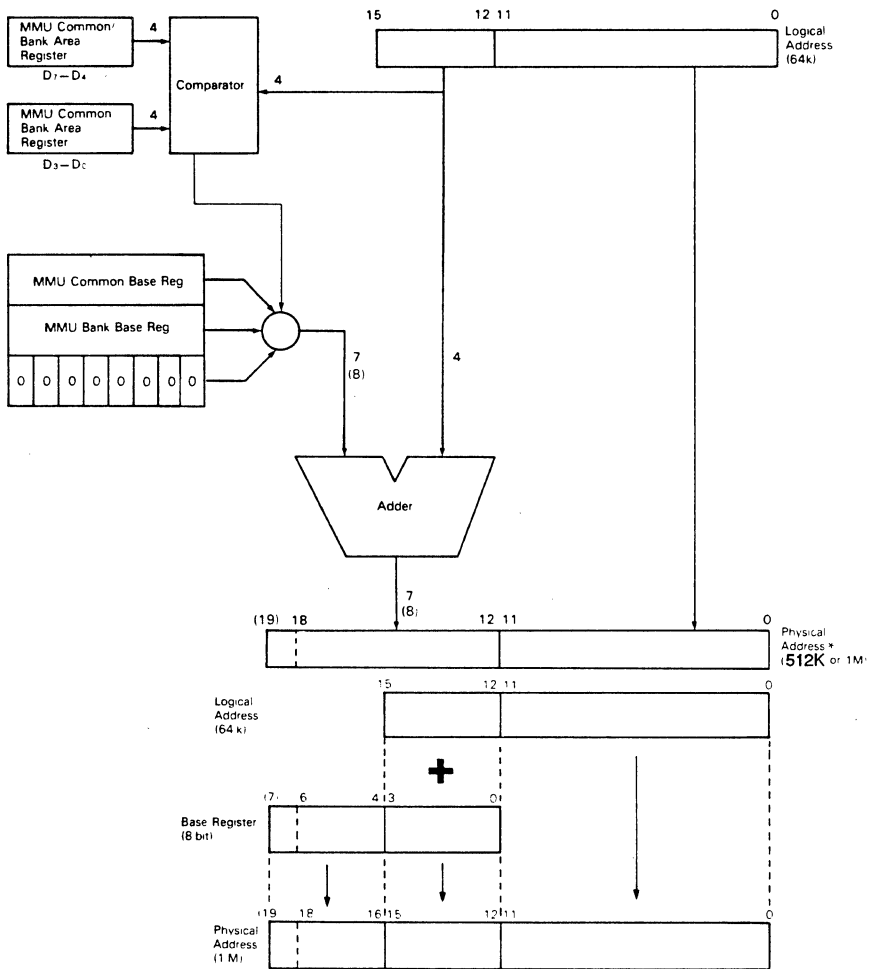


Figure 23. I/O Address Translation

### (3) DMA Cycles

When the Z180 on-chip DMAC is using the external bus, the MMU is physically bypassed. The 20-bit source and destination registers in the DMAC are directly output on the physical address bus (A0-A19).

**Physical address translation.** Fig. 24 shows the way in which physical addresses are generated based on the contents of CBAR, CBR and BBR. MMU comparators classify an access by logical area as defined by CBAR. Depending on which of the three potential logical areas (Common Area 1, Bank Area, or Common Area 0) is being accessed, the appropriate 8-bit base address is added to the high-order 4 bits of the logical address, yielding a 20-bit physical address. CBR is associated with Common Area 1 accesses. Common Area 0, if defined, is always based at physical address 00000H.



**Figure 24. Physical Address Generation**

(\*Z80180 in DIP package is addressable to 512K. All other Z180 can address up to 1M.)

## Dynamic RAM Refresh Control

The Z180 incorporates a dynamic RAM refresh control circuit including 8-bit refresh address generation and programmable refresh timing. This circuit generates asynchronous refresh cycles inserted at the programmable interval independent of CPU program execution. For systems which do not use dynamic RAM, the refresh function can be disabled.

When the internal refresh controller determines that a refresh cycle should occur, the current instruction is interrupted at the first breakpoint between machine cycles. The refresh cycle is inserted by placing the refresh address on A<sub>0</sub>-A<sub>7</sub> and the  $\overline{\text{RFSH}}$  output is driven LOW.

Refresh cycles may be programmed to be either 2 or 3 clock cycles in duration by programming the REFW (Refresh Wait) bit in the Refresh Control Register (RCR). Note that the external WAIT input and the internal wait state generator are not effective during refresh.

Fig. 25 shows the timing of a refresh cycle with a refresh wait ( $T_{RW}$ ) cycle.

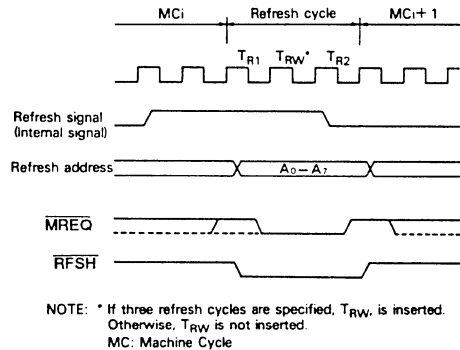


Figure 25. Refresh Cycle Timing

## DMA Controller (DMAC)

The Z180 contains a two-channel DMA (Direct Memory Access) controller which supports high speed data transfer. Both channels (channel 0 and channel 1) have the following capabilities.

**Memory Address Space.** Memory source and destination addresses can be directly specified anywhere within the 1024K byte physical address space using 20-bit source and destination memory addresses. In addition, memory transfers can arbitrarily cross 64K byte physical address boundaries without CPU intervention.

**I/O Address Space.** I/O source and destination addresses can be directly specified anywhere within the 64K byte I/O address space (16-bit source and destination I/O addresses).

**Transfer Length.** Up to 64K bytes are transferred based on a 16-bit byte count register.

**$\overline{\text{DREQ}}$  Input.** Level and edge sense DREQ input detection are selectable.

**$\overline{\text{TEND}}$  Output.** Used to indicate DMA completion to external devices.

**Transfer Rate.** Each byte transfer can occur every 6 clock cycles. Wait states can be inserted in DMA cycles for slow memory or I/O devices. At the system clock ( $\phi$ ) = 6 MHz, the DMA transfer rate is as high as 1.0 megabytes/second (no wait states).

There is an additional feature disc for DMA interrupt request by DMA END. Each channel has the following additional specific capabilities.

### Channel 0

Memory ↔ memory, memory ↔ I/O, memory ↔ memory mapped I/O transfers.

- Memory address increment, decrement, no-change.
- Burst or cycle steal memory to/from memory transfers.
- DMA to/from both ASCII channels.
- Higher priority than DMAC channel 1.

### Channel 1

Memory ↔ I/O transfer.

Memory address increment, decrement

### DMAC Registers

Each channel of the DMAC (channel 0, 1) has three registers specifically associated with that channel.

### Channel 0

SAR0 - Source Address Register  
DAR0 - Destination Address Register  
BCR0 - Byte Count Register

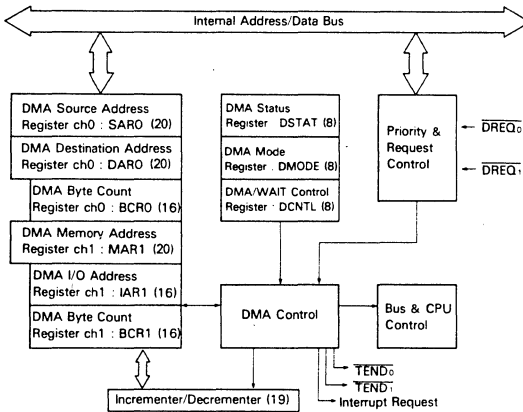
### Channel 1

MAR1 - Memory Address Register  
IAR1 - I/O Address Register  
BCR1 - Byte Count Register

The two channels share the following three additional registers in common.

DSTAT - DMA Status Register  
DMODE - DMA Mode Register  
DCNTL - DMA Control Register

**DMAC Block Diagram.** Fig.26 shows the Z180 DMAC Block Diagram.



**Figure 26. DMAC Block Diagram**

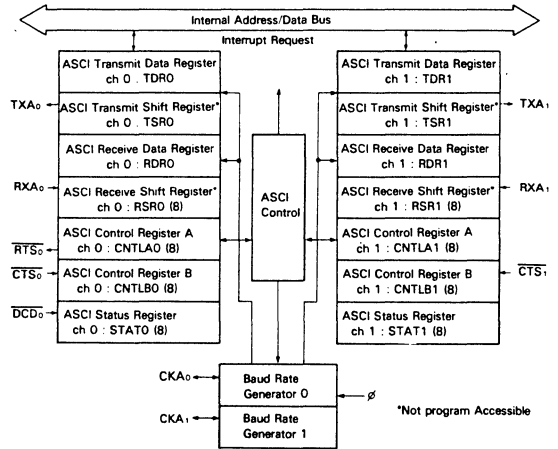
**Asynchronous Serial Communication Interface (ASCI)**

The Z180 on-chip ASCI has two independent full-duplex channels. Based on full programmability of the following functions, the ASCI directly communicates with a wide variety of standard UARTs (Universal Asynchronous Receiver/Transmitter) including the Z8440 SIO, the Z8530 SCC and the Z85230 ESCC.

The key functions for ASCI are shown below. Each channel is independently programmable.

- Full-duplex communication.
- 7- or 8-bit data length.
- Program controlled 9th data bit for multiprocessor communication.
- 1 or 2 stop bits.
- Odd, even, no parity.
- Parity, overrun, framing error detection.
- Programmable baud rate generator, /16 and /64 modes.
- Speed to 38.4K bits per second (CPU  $f_c = 6.144$  MHz).
- Modem control signals - Channel 0 has DCD0, CTS0 and RTS0 Channel 1 has CTS1.
- Programmable interrupt condition enable and disable.
- Operation with on-chip DMAC.

**ASCI Block Diagram.** Fig. 27 shows the ASCI Block Diagram.



**Figure 27. ASCI Block Diagram**

## Clocked Serial I/O Port (CSI/O)

The Z180 includes a simple, high speed clock, synchronous serial I/O port. The CSI/O includes transmit/receive (half-duplex), fixed 8-bit data, and internal or external data clock selection. High speed operation (baud rate 200K bits/second at  $f_C = 4$  MHz) is provided. The CSI/O is ideal for implementing a multiprocessor communication link between multiple Z180s. These secondary devices may typically perform a portion of the system I/O processing, i.e. keyboard scan/decode, LDC interface, etc.

**CSI/O Block Diagram.** The CSI/O block diagram is shown in Fig. 28. The CSI/O consists of two registers - the Transmit/Receive Data Register (TRDR) and Control Register (CNTR).

**CSI/O Transmit/Receive Data Register (TRDR: I/O Address = 0BH).** TRDR is used for both CSI/O transmission and reception. Thus, the system design must insure that the constraints of half-duplex operation are met (Transmit and receive operation cannot occur simultaneously). For example, if a CSI/O transmission is attempted while the CSI/O is receiving data, a CSI/O will not work. Also note that TRDR is not buffered. Therefore, attempting to perform a CSI/O transmit while the previous transmit data is still being shifted out causes the shift data to be immediately updated, thereby corrupting the transmit operation in

progress. Similarly, reading TRDR while a transmit or receive is in progress should be avoided.

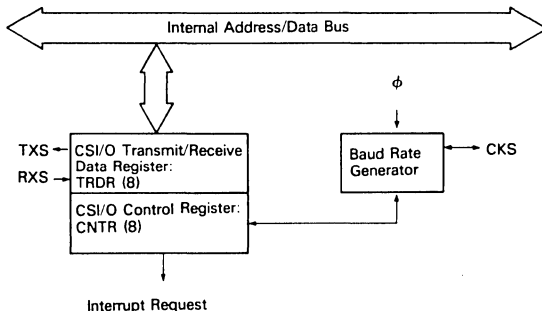


Figure 28. CSI/O Block Diagram

### CSI/O Register Description

**CSI/O Control/Status Register (CNTR: I/O Address = 0AH).** CNTR is used to monitor CSI/O status, enable and disable the CSI/O, enable and disable interrupt generation, and select the data clock speed and source.

## Programmable Reload Timer (PRT)

The Z180 contains a two channel 16-bit Programmable Reload Timer. Each PRT channel contains a 16-bit down counter and a 16-bit reload register. The down counter is directly read and written and a down counter overflow interrupt can be programmably enabled or disabled. Also, PRT channel 1 has a TOUT output pin (pin 31 - multiplexed with A18) which can be set HIGH, LOW, or toggled. Thus, PRT1 can perform programmable output waveform generation.

**PRT block diagram.** The PRT block diagram is shown in Fig. 29. The two channels have separate timer data and reload registers and a common status/control register. The PRT input clock for both channels is equal to the system clock divided by 20.

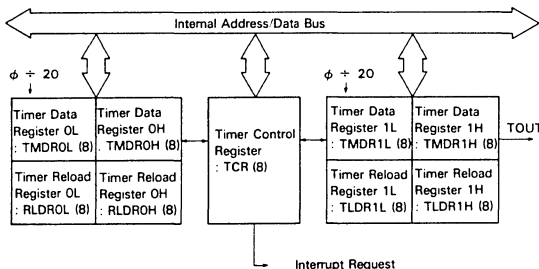


Figure 29. PRT Block Diagram

## Secondary Bus Interface

**E clock Output Timing.** The Z180 also has a secondary bus interface that allows it to easily interface with other peripheral families.

These devices require connection with the Z180

synchronous E clock output. The speed (access time) required for the peripheral devices are determined by the Z180 clock rate. Table 19, and Figures 80-82 define E clock output timing.



## On-Chip Clock Generator

The Z180 contains a crystal oscillator and system clock generator. A crystal can be directly connected or an external clock input can be provided. In either case, the system clock is equal to one-half the input clock. For example, a crystal or external clock input of 8 MHz corresponds with a system clock rate of 4 MHz.

The following table shows the AT cut crystal characteristics ( $C_o$ ,  $R_s$ ) and the load capacitance ( $CL_1$ ,  $CL_2$ ) required for various frequencies of Z80180 operation.

Clock Frequency	4MHz	$4\text{MHz} < f \leq 12\text{MHz}$	$12\text{MHz} < f \leq 16\text{MHz}$
Item			
$C_o$	< 7 pF	< 7 pF	< 7 pF
$R_s$	< 60Ω	< 60Ω	< 60Ω
$CL_1$ , $CL_2$	10 to 22 pF ± 10%	10 to 22 pF ± 10%	10 to 22 pF ± 10%

Table 4.

If an external clock input is used instead of a crystal, the waveform (twice the clock rate) should exhibit a  $50\% \pm 10\%$  duty cycle. Note that the minimum clock input HIGH voltage level is  $V_{cc}-0.6V$ . The external clock input is connected to the EXTAL pin, while the XTAL pin is left open. Fig. 30 shows external clock interface.

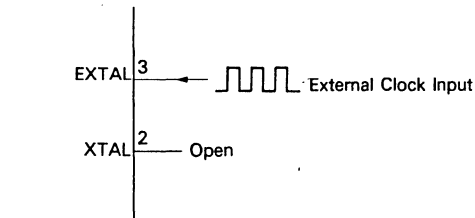


Figure 30. External Clock Interface

## Miscellaneous

### Free Running Counter (I/O Address = 18H)

Read only 8-bit free running counter without control registers and status registers. The contents of the 8-bit free running counter is counted down by one with an interval of 10 clock cycles. The free running counter continues counting down without being affected by the read operation.

If data is written into the free running counter, the interval of DRAM refresh cycle and baud rates for the ASCII and CSI/O are not guaranteed.

In IOSTOP mode, the free running counter continues counting down. It is initialized to FFH during RESET.

---

## SOFTWARE ARCHITECTURE

**Instruction Set.** The Z180 is object code compatible with the Z80 CPU, refer to the Z80 CPU Technical Manual or the Z80 Assembly Language Programming Manual for further details.

### New Instructions    Operation

SLP	Enter SLEEP mode
MLT	8-bit multiply with 16-bit result
INO g, (m)	Input contents of immediate I/O address
OUT0 (m), g	Output register contents to immediate I/O address
OTIM	Block output - increment
OTIMR	Block output - increment and repeat
OTDM	Block output - decrement
OTDMR	Block output - decrement and repeat
TSTIO m	Non-destructive AND, I/O port, and accumulator
TST g	Non-destructive AND, register, and accumulator
TST m	Non-destructive AND, immediate data and accumulator.
TST (HL)	Non-destructive AND, memory data, and accumulator.

**SLP - Sleep.** The SLP instruction causes the Z180 to enter the SLEEP low power consumption mode. See section 2.4 for a complete description of the SLEEP state.

**MLT - Multiply.** The MLT performs unsigned multiplication on two 8-bit numbers yielding a 16-bit result. MLT may specify BC, DE, HL or SP registers. In all cases, the 8-bit

operands are loaded into each half of the 16-bit register and the 16-bit result is returned in that register.

**OTIM, OTIMR, OTDM, OTDMR - Block I/O.** The contents of memory pointed to by HL is output to the I/O address in (C). The memory address (HL) and I/O address (C) are incremented in OTIM and OTIMR and decremented in OTDM and OTDMR, respectively. The B register is decremented. The OTIMR and OTDMR variants repeat the above sequence until register B is decremented to 0. Since the I/O address (C) is automatically incremented or decremented, these instructions are useful for block I/O (such as Z180 on-chip I/O) initialization. When I/O is accessed, 00H is output in high-order bits of address automatically.

**TSTIO m - Test I/O Port.** The contents of the I/O port addressed by C are ANDed with immediately specified 8-bit data and the status flags are updated. The I/O port contents are not written (non-destructive AND). When I/O is accessed, 00H is output in higher bits of address automatically.

**TST g - Test Register.** The contents of the specified register are ANDed with the accumulator (A) and the status flags are updated. The accumulator and specified register are not changed (non-destructive AND).

**TST m - Test Immediate.** The contents of the immediately specified 8-bit data are ANDed with the accumulator (A) and the status flags are updated. The accumulator is not changed (non-destructive AND).

**TST (HL) - Test Memory.** The contents of memory pointed to by HL are ANDed with the accumulator (A) and the status flags are updated. The memory contents and accumulator are not changed (non-destructive AND).

**INO g, (m) - Input, Immediate I/O address.** The contents of immediately specified 8-bit I/O address are input into the specified register. When I/O is accessed, 00H is output in high-order bits of the address automatically.

**OUT0 (m), g - Output, immediate I/O address.** The contents of the specified register are output to the immediately specified 8-bit I/O address. When I/O is accessed, 00H is output in high-order bits of the address automatically.

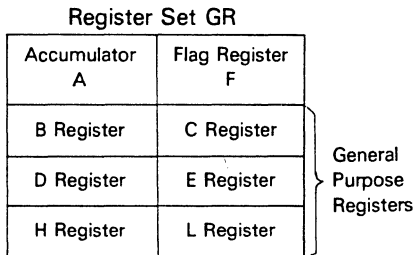
### CPU Registers

The Z180 CPU registers consist of Register Set GR, Register Set GR' and Special Registers.

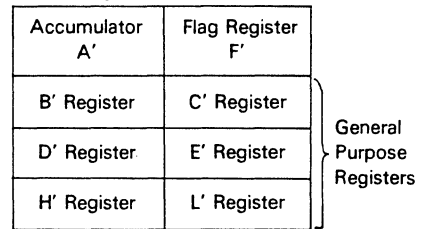
The Register Set GR consists of 8-bit Accumulator (A), 8-bit Flag Register (F), and three General Purpose Registers (BC, DE, and HL) which may be treated as 16-bit registers (BC, DE, and HL) or as individual 8-bit registers (B, C, D, E, H, and L) depending on the instruction to be executed. The Register Set GR' is alternate register set of Register Set GR and also contains Accumulator (A'), Flag Register (F') and three General Purpose Registers (BC', DE', and HL'). While the alternate Register Set GR' contents are not directly accessible, the contents can be programmably exchanged at high speed with those of Register Set GR.

The Special Registers consist of 8-bit Interrupt Vector Register (I), 8-bit R Counter (R), two 16-bit Index Registers (IX and IY), 16-bit Stack Pointer (SP), and 16-bit Program Counter (PC).

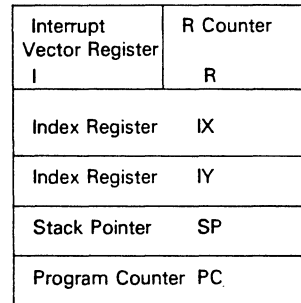
Fig. 31 shows CPU registers configuration.



### Register Set GR'



### Special Registers



**Figure 31. CPU Registers**

## Z80180 ELECTRICAL CHARACTERISTICS

### ABSOLUTE MAXIMUM RATINGS

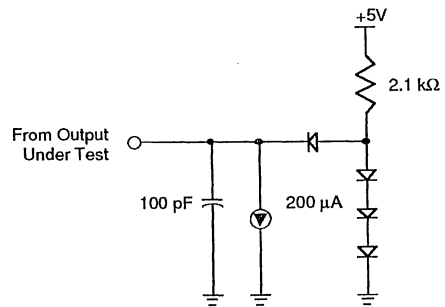
Item	Symbol	Value	Unit
Supply Voltage	$V_{CC}$	-0.3 ~ +7.0	V
Input Voltage	$V_{in}$	-0.3 ~ $V_{CC} + 0.3$	V
Operating Temperature	Standard	$T_{opr}$	0 - 70
	Extended	$T_{opr}$	-40 - 85
Storage Temperature	$T_{stg}$	-55 ~ +150	°C

[NOTE] Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

### STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance sections below apply to the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin (Test Load Configuration).

All AC parameters assume a load capacitance of 100 pF. Add 10 ns delay for each 50 pF increase in load up to a maximum of 200 pF for the data bus and 100 pF for the address and control lines. AC timing measurements are referenced to 1.5 volts (except for CLOCK, which is referenced to the 10% and 90% points).



Test Load Configuration

## Z80180 DC CHARACTERISTICS

$V_{CC}=5V \pm 10\%$ ,  $V_{SS}=0V$ , over specified temperature range unless otherwise noted.

Sym	Item	Condition	Min	Typ	Max	Unit
$V_{IH1}$	Input "H" Voltage /RESET, EXTAL /NMI		$V_{CC} - 0.6$	-	$V_{CC} + 0.3$	V
$V_{IH2}$	Input "H" Voltage except /RESET, EXTAL /NMI	2.0	-	$V_{CC} + 0.3$	V	
$V_{IL1}$	Input "L" Voltage /RESET, EXTAL /NMI		-0.3	-	0.6	V
$V_{IL2}$	Input "L" Voltage except /RESET, EXTAL /NMI	-0.3	-	0.8	V	
$V_{OH}$	Output "H" Voltage All outputs	$I_{OH} = -200 \mu A$	2.4	-	-	V
		$I_{OH} = -20 \mu A$	$V_{CC} - 1.2$	-	-	V
$V_{OL}$	Output "L" Voltage All outputs	$I_{OL} = 2.2 \text{ mA}$	-	-	0.45	V
$V_{IL}$	Input Leakage Current All inputs except XTAL, ETAL	$V_{IN} = 0.5, V_{CC} - 0.5$	-	-	1.0	$\mu A$
$I_{TL}$	Tri-state Leakage Current	$V_{IN} = 0.5, V_{CC} - 0.5$	-	-	1.0	$\mu A$
$I_{CC}^*$	Power Dissipation* (Normal Operation)	f = 6 MHz	-	15	40	mA
		f = 8 MHz	-	20	50	mA
		f = 10 MHz	-	25	60	mA
$I_{CC}^*$	Power Dissipation* (SYSTEM STOP mode)	f = 6 MHz	-	3.8	12.5	mA
		f = 8 MHz	-	5	15.0	mA
		f = 10 MHz	-	6.3	17.5	mA
$C_p$	Pin Capacitance	$V_{IN} = 0V, f = 1 \text{ MHz}$ $T_A = 25^\circ C$	-	-	12	pF

### Notes:

\* $V_{H,min} = V_{CC} - 1.0V$ ,  $V_{L,max} = 0.8V$  (All output terminals are at no load.)  
 $V_{CC} = 5.0V$

## Z80180 AC CHARACTERISTICS

( $V_{CC}=5V \pm 10\%$ ,  $V_{SS}=0V$ , over specified temperature range, unless otherwise noted.)

No	Sym	Parameter	Z8018006		Z8018008		Z8018010		Unit	Note
			Min	Max	Min	Max	Min	Max		
1	tcyc	Clock Cycle Time	162	2000	125	2000	100	2000	ns	[1]
2	tCHW	Clock Pulse Width (High)	65		55		40		ns	[1]
3	tCLW	Clock Pulse Width (Low)	65		55		40		ns	[1]
4	tcf	Clock Fall Time		15		15		10	ns	[1]
5	tcr	Clock Rise Time		15		15		10	ns	[1]
6	tAD	Address Valid From Clock Rise		90		80		70	ns	
7	tAS	Address Valid To /MREQ, /IORQ Fall	30		20		10		ns	
8	tMED1	Clock Fall To /MREQ Fall Delay		60		50		50	ns	
9	IRDD1	Clock Fall To /RD Fall (/IOC=1) Clock Rise To /RD Fall (/IOC=0)		60		50		50	ns	
				65		60		55	ns	
10	tM1D1	Clock Rise To /M1 Fall Delay		80		70		60	ns	
11	tAH	Address Hold Time (/MREQ, /IORQ, /RD, /WR)	35		20		10		ns	
12	tMED2	Clock Fall To /MREQ Rise Delay		60		50		50	ns	
13	IRDD2	Clock Fall To /RD Rise Delay		60		50		50	ns	
14	tM1D2	Clock Rise To /M1 Rise Delay		80		70		60	ns	
15	tDRS	Data Read Setup Time	40		30		25		ns	
16	tDRH	Data Read Hold time	0		0		0		ns	
17	tSTD1	Clock Edge To ST Fall		90		70		60	ns	
18	tSTD2	Clock Edge To ST Rise		90		70		60	ns	
19	tWS	/WAIT Setup Time To Clock Fall	40		40		30		ns	
20	tWH	/WAIT Hold Time From Clock Fall	40		40		30		ns	
21	tWDZ	Clock Rise To Data Float Delay		95		70		60	ns	
22	tWRD1	Clock Rise To /WR Fall Delay		65		60		50	ns	
23	tWDD	Clock Fall To Write Data Delay		90		80		60	ns	
24	tWDS	Write Data Setup Time To /WR	40		20		15		ns	
25	tWRD2	Clock Fall To /WR Rise		80		60		50	ns	
26	tWRP	/WR Pulse Width (Memory Write Cycles)	170		130		110		ns	
26a		/WR Pulse Width (I/O Write Cycles)	332		225		210		ns	
27	tWDH	Write Data Hold Time From /WR Rise	40		15		10		ns	
28	tIOD1	Clock Fall To /IORQ Fall Delay (/IOC=1) Clock Rise To /IORQ Fall Delay (/IOC=0)		60		50		50	ns	
				65		60		55	ns	
29	tIOD2	Clock Fall /IOQR Rise Delay		60		50		50	ns	
30	tIOD3	/M1 Fall To /IORQ Fall Delay	340		250		200		ns	
31	tINTS	/INT Setup Time To Clock Fall	40		40		30		ns	
32	tINTH	/INT Hold Time From Clock Fall	40		40		30		ns	
33	tNMIW	/NMI Pulse Width	120		100		80		ns	
34	tBRS	/BUSREQ Setup Time To Clock Fall	40		40		30		ns	
35	tBRH	/BUSREQ Hold Time From Clock Fall	40		40		30		ns	
36	tBAD1	Clock Rise To /BUSACK Fall Delay		95		70		60	ns	
37	tBAD2	Clock Fall To /BUSACK Rise Delay		95		70		60	ns	
38	tBZD	Clock Rise To Bus Floating Delay Time		125		90		80	ns	
39	tMEWH	/MREQ Pulse Width (High)	110		90		70		ns	
40	tMEWL	/MREQ Pulse Width (Low)	125		100		80		ns	

## Z80180 AC CHARACTERISTICS (Continued)

( $V_{CC}=5V \pm 10\%$ ,  $V_{SS}=0V$ , over specified temperature range, unless otherwise noted.)

No	Sym	Parameter	Z8018006		Z8018008		Z8018010		Unit	Note
			Min	Max	Min	Max	Min	Max		
41	tRFD1	Clock Rise To /RFSH Fall Delay		90		80		60	ns	
42	tRFD2	Clock Rise To /RFSH Rise Delay		90		80		60	ns	
43	tHAD1	Clock Rise To /HALT Fall Delay		90		80		50	ns	
44	tHAD2	Clock Rise To /HALT Rise Delay		90		80		50	ns	
45	tDRQS	/DREQi Setup Time To Clock Rise	40		40		30		ns	
46	tDRQH	/DREQi Hold Time From Clock Rise	40		40		30		ns	
47	tTED1	Clock Fall To /TENDi Fall Delay		70		60		50	ns	
48	tTED2	Clock Fall To /TENDi Rise Delay		70		60		50	ns	
49	tED1	Clock Rise To E Rise Delay		95		70		60	ns	
50	tED2	Clock Edge To E Fall Delay		95		70		60	ns	
51	PWEH	E Pulse Width (High)	75		65		55		ns	
52	PWEL	E Pulse Width (Low)	180		130		110		ns	
53	tEr	Enable Rise Time		20		20		20	ns	
54	tEf	Enable Fall Time		20		20		20	ns	
55	tTOD	Clock Fall To Timer Output Delay		300		200		150	ns	
56	tSTDI	CSI/O Tx Data Delay Time (Internal Clock Operation)		200		200		150	ns	
57	tSTDE	CSI/O Tx Data Delay Time (External Clock Operation)		7.5tcyc+300		7.5tcyc+200		7.5tcyc+150	ns	
58	tSRSI	CSI/O Rx Data Setup Time (Internal Clock Operation)	1		1		1		tcyc	
59	tSRHI	CSI/O Rx Data Hold Time (Internal Clock Operation)	1		1		1		tcyc	
60	tSRSE	CSI/O Rx Data Setup Time (External Clock Operation)	1		1		1		tcyc	
61	tSRHE	CSI/O Rx Data Hold Time (External Clock Operation)	1		1		1		tcyc	
62	tRES	/RESET Setup Time To Clock Fall	120		100		80		ns	
63	tREH	/RESET Hold Time From Clock Fall	80		70		50		ns	
64	tOSC	Oscillator Stabilization Time		20		20		20	ms	
65	tEXr	External Clock Rise Time (EXTAL)		25		25		15	ns	
66	tEXf	External Clock Fall Time (EXTAL)		25		25		15	ns	
67	tRr	/RESET Rise Time		50		50		50	ms	[2]
68	tRf	/RESET Fall Time		50		50		50	ms	[2]
69	tI <sub>r</sub>	Input Rise Time (Except EXTAL, /RESET)	100		100		100		ns	[2]
70	tI <sub>f</sub>	Input Fall Time (Except EXTAL, /RESET)	100		100		100		ns	[2]

### Notes:

[1]  $tcyc = tCHW + tCLW + tcf + tcr$

[2] This parameter has to be modified if other specification(s) cannot be met.

## Z8S180 ELECTRICAL CHARACTERISTICS ABSOLUTE MAXIMUM RATINGS

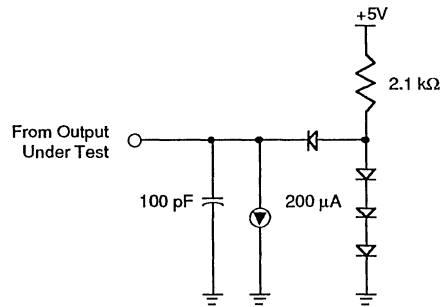
Symbol	Description	Min	Max	Unit
$V_{CC}$	Supply Voltage *	-0.3	+7.0	V
$V_{IN}$	Input Voltage	-0.3	$V_{CC}+0.3$	V
$T_{OPR}$	Operating Temp	-40	+100	C
$T_{STG}$	Storage Temp	-55	+150	C

Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating conditions. If these conditions are exceeded, it could affect reliability of LSI.

## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance sections below apply to the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin (Test Load Configuration).

All AC parameters assume a load capacitance of 100 pF. Add 10 ns delay for each 50 pF increase in load up to a maximum of 200 pF for the data bus and 100 pF for the address and control lines. AC timing measurements are referenced to 1.5 volts (except for CLOCK, which is referenced to the 10% and 90% points).



**Test Load Configuration**



## Z8S180 DC ELECTRICAL CHARACTERISTICS

( $V_{CC} = 5V \pm 10\%$ ,  $V_{SS} = 0V$ , over specified temperature range, unless otherwise noted.)

Symbol	Item	Condition	Min	Typ	Max	Unit
$V_{IH1}$	Input "H" Voltage Reset, EXTAL, NMI		$V_{CC}-0.6$		$V_{CC}+0.3$	V
$V_{IH2}$	Input "H" Voltage Except Reset, EXTAL, NMI		2.0		$V_{CC}+0.3$	V
$V_{IL1}$	Input "L" Voltage Reset, EXTAL, NMI		-0.3		0.6	V
$V_{IL2}$	Input "L" Voltage Except Reset, EXTAL, NMI		-0.3		0.8	V
$V_{OH1}$	Output "H" Voltage All Outputs	$I_{OH} = -200 \mu A$ $I_{OH} = -20 \mu A$	2.4			V
			$V_{CC}-1.2$			V
$V_{OH2}$	Output "H" PHI	$I_{OH} = -200 \mu A$	$V_{CC}-0.6$			V
$V_{OL1}$	Output "L" Voltage All Outputs	$I_{OL} = 2.2 \text{ mA}$			0.45	V
$V_{OL2}$	Output "L" PHI	$I_{OL} = 2.2 \text{ mA}$			0.45	V
$I_{IL}$	Input Leakage Current All Inputs Except XTAL, EXTAL	$V_{IN} = 0.5 \text{ to } V_{CC}-0.5$			1.0	$\mu A$
ITL	Tri-state Leakage Current	$V_{IN} = 0.5 \text{ to } V_{CC}-0.5$			1.0	$\mu A$
$I_{CC}^*$	Power Dissipation (Normal Operation)	$f = 16 \text{ MHz}$ $f = 20 \text{ MHz}$		45	100	mA
				50	120	mA
	Power Dissipation (System STOP mode)	$f = 16 \text{ MHz}$ $f = 20 \text{ MHz}$		10 15	30 40	mA
	Power Dissipation (STANDBY Mode)	External Oscillator, Internal Clock Stops			10	$\mu A$
$C_p$	Pin Capacitance	$V_{IN} = 0V$ , $f = 1 \text{ MHz}$ , $T_A = 25^\circ C$			12	pF

### Notes:

\*  $V_{IH} \text{ min} = V_{CC} - 1.0V$ ,  $V_{IL} \text{ max} = 0.8V$  (all output terminals are at no load).  
 $V_{CC} = 5.0V$

## Z8S180 AC CHARACTERISTICS

( $V_{CC}=5V \pm 10\%$ ,  $V_{SS}=0V$ , over specified temperature range, unless otherwise noted.)

No	Sym	Parameter	Z8S18016		Z8S18020		Unit	Note
			Min	Max	Min	Max		
1	t <sub>cyc</sub>	Clock Cycle time	62	2000	50	2000	ns	[1]
2	t <sub>CHW</sub>	Clock Pulse width (high)	25		20		ns	[1]
3	t <sub>CLW</sub>	Clock Pulse width (low)	25		20		ns	[1]
4	t <sub>cf</sub>	Clock Fall time		6		5	ns	[1]
5	t <sub>cr</sub>	Clock Rise time		6		5	ns	[1]
6	t <sub>AD</sub>	Address valid from Clock Rise		35		30	ns	
7	t <sub>AS</sub>	Address valid to /MREQ, /IORQ Fall	5		5		ns	
8	t <sub>MED1</sub>	Clock Fall to /MREQ Fall delay		25		25	ns	
9	t <sub>RDD1</sub>	Clock Fall to /RD Fall (/IOC=1) Clock Rise to /RD Fall (/IOC=0)		25		25	ns	
10	t <sub>M1D1</sub>	Clock Rise to /M1 Fall delay		45		35	ns	
11	t <sub>AH</sub>	Address Hold time (/MREQ, /IORQ, /RD, /WR)	5		5		ns	
12	t <sub>MED2</sub>	Clock Fall to /MREQ Rise Delay		30		25	ns	
13	t <sub>RDD2</sub>	Clock Fall to /RD Rise delay		30		25	ns	
14	t <sub>M1D2</sub>	Clock Rise to /M1 Rise delay		45		40	ns	
15	t <sub>D<sub>RS</sub></sub>	Data Read Setup Time	15		10		ns	
16	t <sub>D<sub>RH</sub></sub>	Data Read Hold time	0		0		ns	
17	t <sub>STD1</sub>	Clock Edge to ST Fall		35		30	ns	
18	t <sub>STD2</sub>	Clock Edge to ST Rise		35		30	ns	
19	t <sub>WS</sub>	/WAIT setup time to Clock Fall	15		15		ns	[2]
20	t <sub>WH</sub>	/WAIT Hold time from Clock Fall	10		10		ns	
21	t <sub>WDZ</sub>	Clock Rise to Data Float Delay		40		35	ns	
22	t <sub>WRD1</sub>	Clock Rise to /WR Fall delay		25		25	ns	
23	t <sub>WDD</sub>	Clock Fall to Write Data Delay		30		25	ns	
24	t <sub>WDS</sub>	Write Data Setup time to /WR	10		10		ns	
25	t <sub>WRD2</sub>	Clock Fall to /WR Rise		30		25	ns	
26	t <sub>WRP</sub>	/WR Pulse Width (Memory Write Cycles)	80		80		ns	
26a		/WR Pulse Width (I/O Write Cycles)	150		150		ns	
27	t <sub>WDH</sub>	Write Data Hold time from /WR Rise	10		10		ns	
28	t <sub>IOD1</sub>	Clock Fall to /IORQ Fall delay (/IOC=1) Clock Rise to /IORQ Fall delay (/IOC=0)		30		25	ns	
29	t <sub>IOD2</sub>	Clock Fall /IOQR Rise Delay		30		25	ns	
30	t <sub>IOD3</sub>	/M1 Fall to /IORQ Fall delay	120		100		ns	
31	t <sub>INTS</sub>	/INT Setup Time to Clock Fall	20		20		ns	
32	t <sub>INTH</sub>	/INT Hold Time from Clock Fall	10		10		ns	
33	t <sub>NMIW</sub>	/NMI Pulse width	40		35		ns	
34	t <sub>BRS</sub>	/BUSREQ Setup Time to Clock Fall	10		10		ns	
35	t <sub>BRH</sub>	/BUSREQ Hold Time from Clock Fall	10		10		ns	
36	t <sub>BAD1</sub>	Clock Rise to /BUSACK Fall delay		30		25	ns	
37	t <sub>BAD2</sub>	Clock Fall to /BUSACK Rise delay		30		25	ns	
38	t <sub>BZD</sub>	Clock Rise to Bus Floating Delay Time		45		40	ns	
39	t <sub>MEWH</sub>	/MREQ Pulse Width (High)	45		35		ns	
40	t <sub>MEWL</sub>	/MREQ Pulse Width (LOW)	45		35		ns	

## Z8S180 AC CHARACTERISTICS (Continued)

( $V_{CC}=5V \pm 10\%$ ,  $V_{SS}=0V$ , over specified temperature range, unless otherwise noted.)

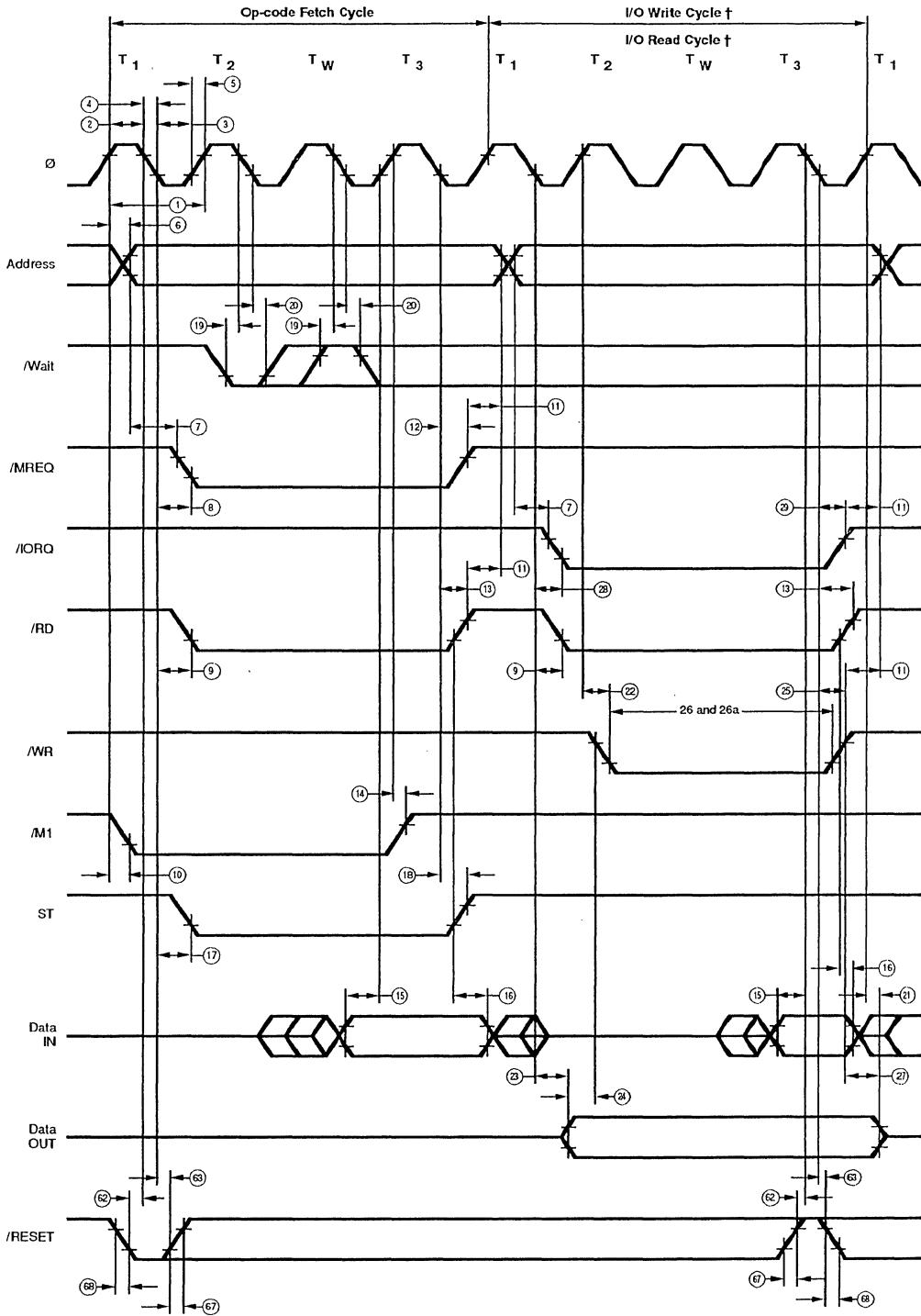
No	Sym	Parameter	Z8S18016		Z8S18020		Unit	Note
			Min	Max	Min	Max		
41	tRFD1	Clock Rise to /RFSH Fall Delay		25	20		ns	
42	tRFD2	Clock Rise to /RFSH Rise Delay		25	20		ns	
43	tHAD1	Clock Rise to /HALT Fall Delay		20	15		ns	
44	tHAD2	Clock Rise to /HALT Rise Delay		20	15		ns	
45	tDRQS	/DREQi Setup Time to Clock Rise	20		20		ns	
46	tDRQH	/DREQi Hold Time from Clock Rise	20		20		ns	
47	tTED1	Clock Fall to /TENDi Fall Delay		30	25		ns	
48	tTED2	Clock Fall to /TENDi Rise Delay		30	25		ns	
49	tED1	Clock Rise to E Rise Delay		35	30		ns	
50	tED2	Clock Edge to E Fall Delay		35	30		ns	
51	PWEH	E Pulse Width (High)	30		25		ns	
52	PWEL	E Pulse Width (Low)	60		50		ns	
53	tEr	Enable Rise Time		10	10		ns	
54	tEf	Enable Fall Time		10	10		ns	
55	tTOD	Clock Fall to Timer Output Delay		100	75		ns	
56	tSTDi	CSI/O Tx Data Delay Time (Internal Clock Operation)		100	75		ns	
57	tSTDE	CSI/O Tx Data Delay Time (External Clock Operation)		7.5 tcyc +100	7.5 tcyc+75		ns	
58	tSRSi	CSI/O Rx Data Setup Time (Internal Clock Operation)	1		1		tcyc	
59	tSRHi	CSI/O Rx Data Hold Time (Internal Clock Operation)	1		1		Tcyc	
60	tSRSE	CSI/O Rx Data Setup Time (External Clock Operation)	1		1		Tcyc	
61	tSRHE	CSI/O Rx Data Hold Time (External Clock Operation)	1		1		Tcyc	
62	tRES	/RESET Setup time to Clock Fall	45		40		ns	
63	tREH	/RESET Hold time from Clock Fall	30		25		ns	
64	tOSC	Oscillator Stabilization Time		20	20		ms	
65	tEXr	External Clock Rise Time (EXTAL)		6	5		ns	
66	tEXf	External Clock Fall Time (EXTAL)		6	5		ns	
67	tRr	/RESET Rise Time		50	50		ms	[2]
68	tRf	/RESET Fall Time		50	50		ms	[2]
69	tIrr	Input Rise Time (Except EXTAL, /RESET)		50	50		ns	[2]
70	tIf	Input Fall Time (Except EXTAL, /RESET)		50	50		ns	[2]

### Notes:

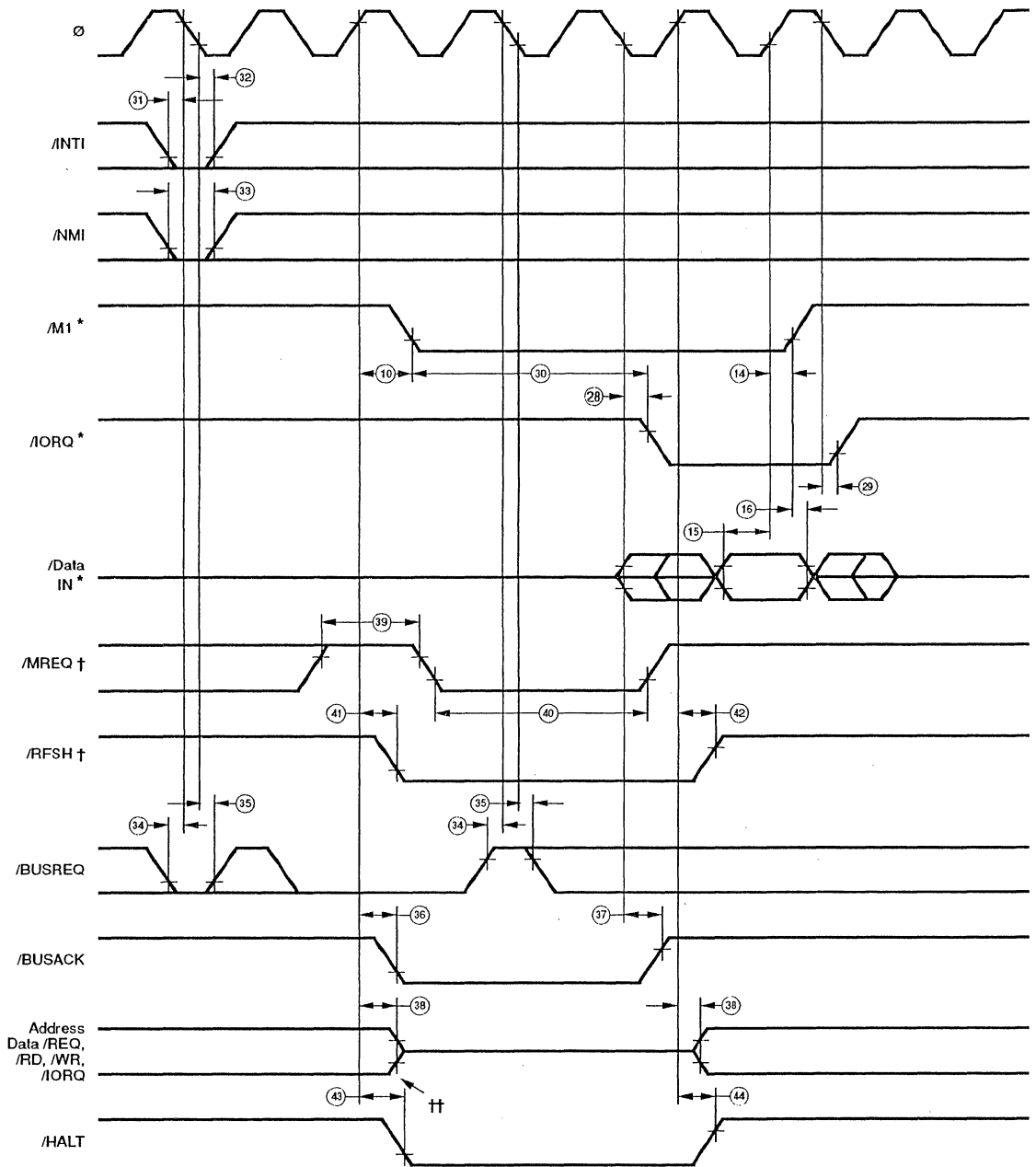
[1]  $tcyc=tCHW+tCLW+tcf+tcr$

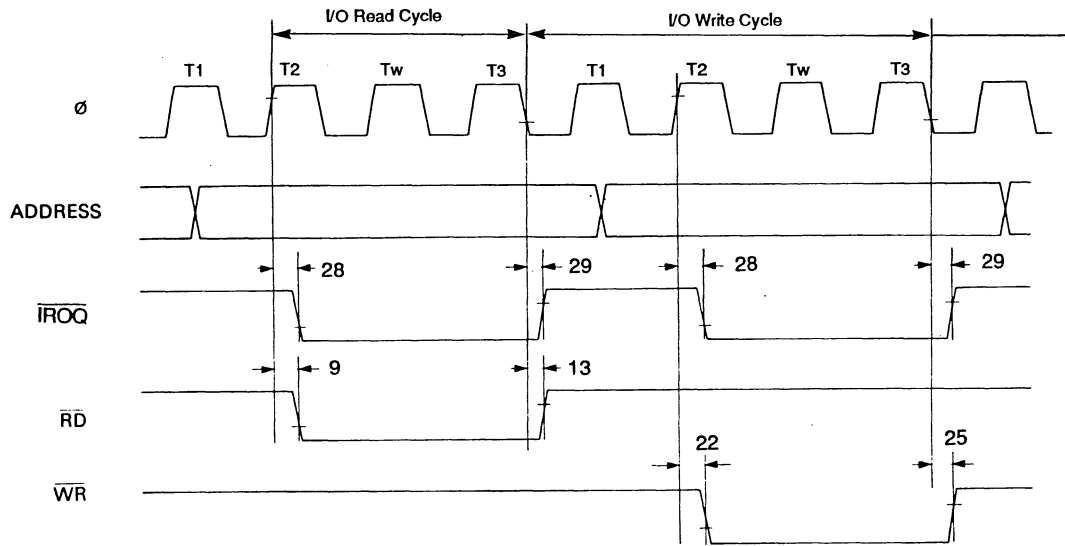
[2] This parameter has to be modified if other specification(s) can not be met.

# TIMING DIAGRAMS



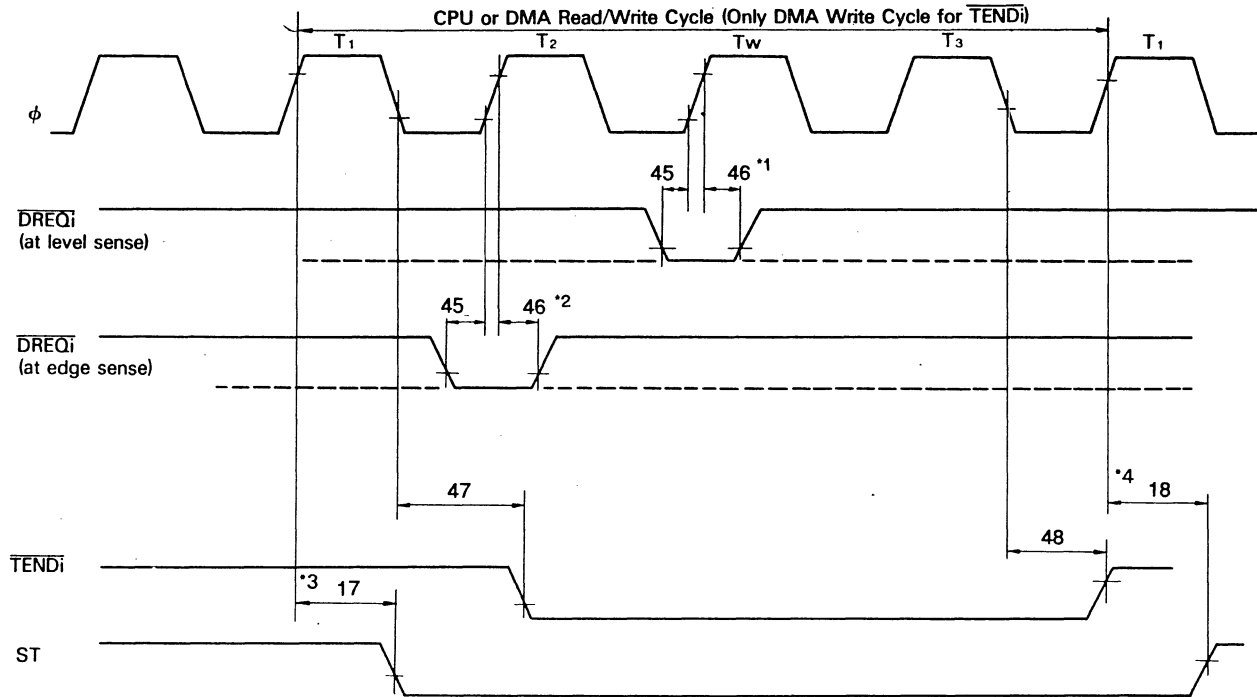
# TIMING DIAGRAMS (Continued)





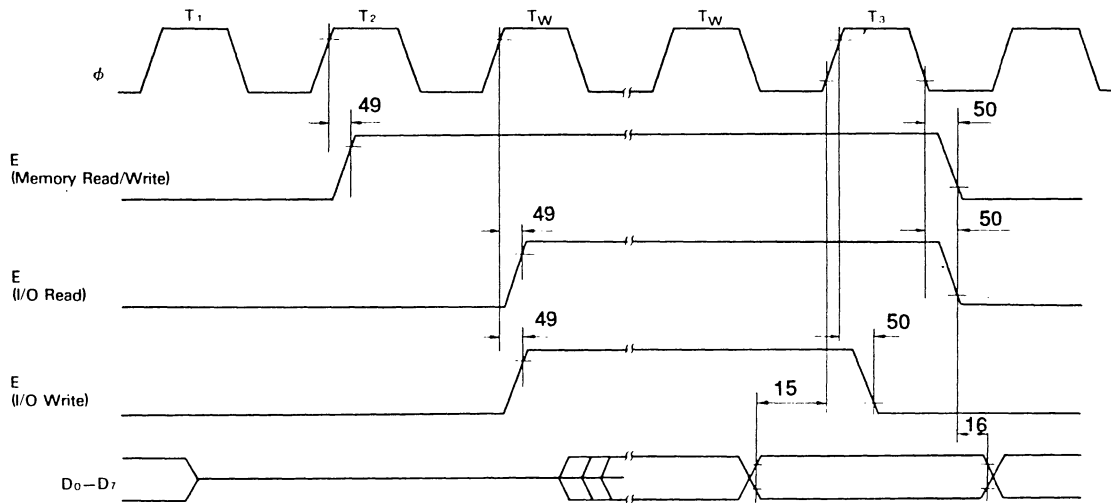
**CPU Timing ( $\overline{IO\bar{C}}=0$ )**

{  
 I/O Read Cycle  
 I/O Write Cycle  
 }

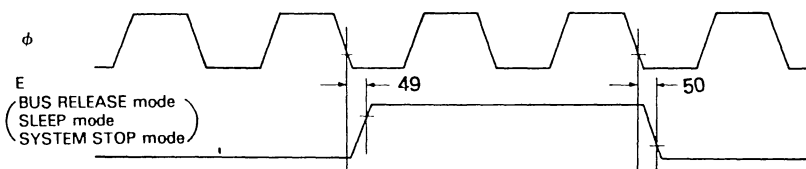


### DMA Control Signals

- \*1  $t_{DRQS}$  and  $t_{DRQH}$  are specified for the rising edge of clock followed by  $T_3$ .
- \*2  $t_{DRQS}$  and  $t_{DRQH}$  are specified for the rising edge of clock.
- \*3 DMA cycle starts.
- \*4 CPU cycle starts.

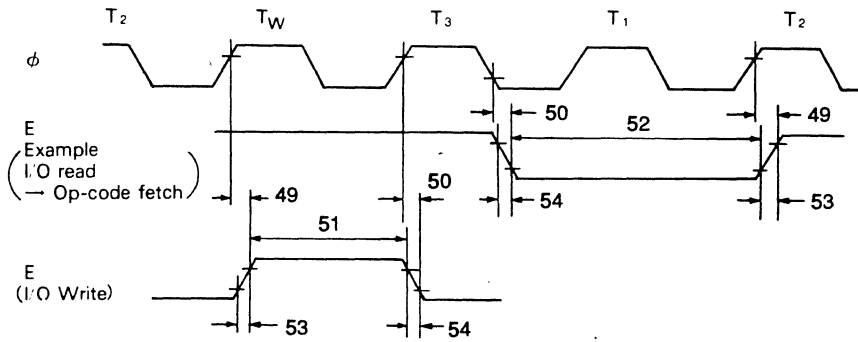


**E Clock Timing ( Memory Read/Write Cycle )**  
**( I/O Read/Write Cycle )**

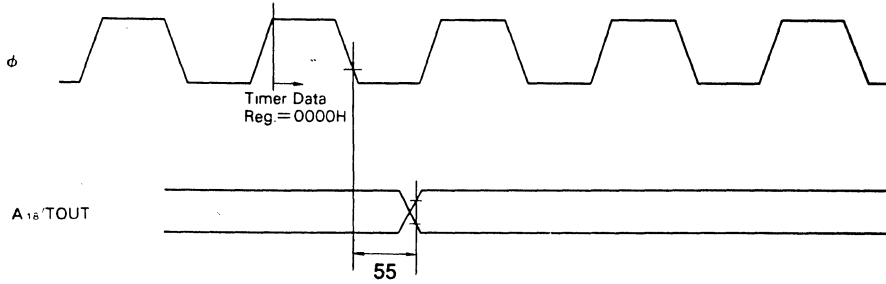


**E Clock Timing ( BUS RELEASE Mode )**  
**( SLEEP Mode )**  
**( SYSTEM STOP Mode )**

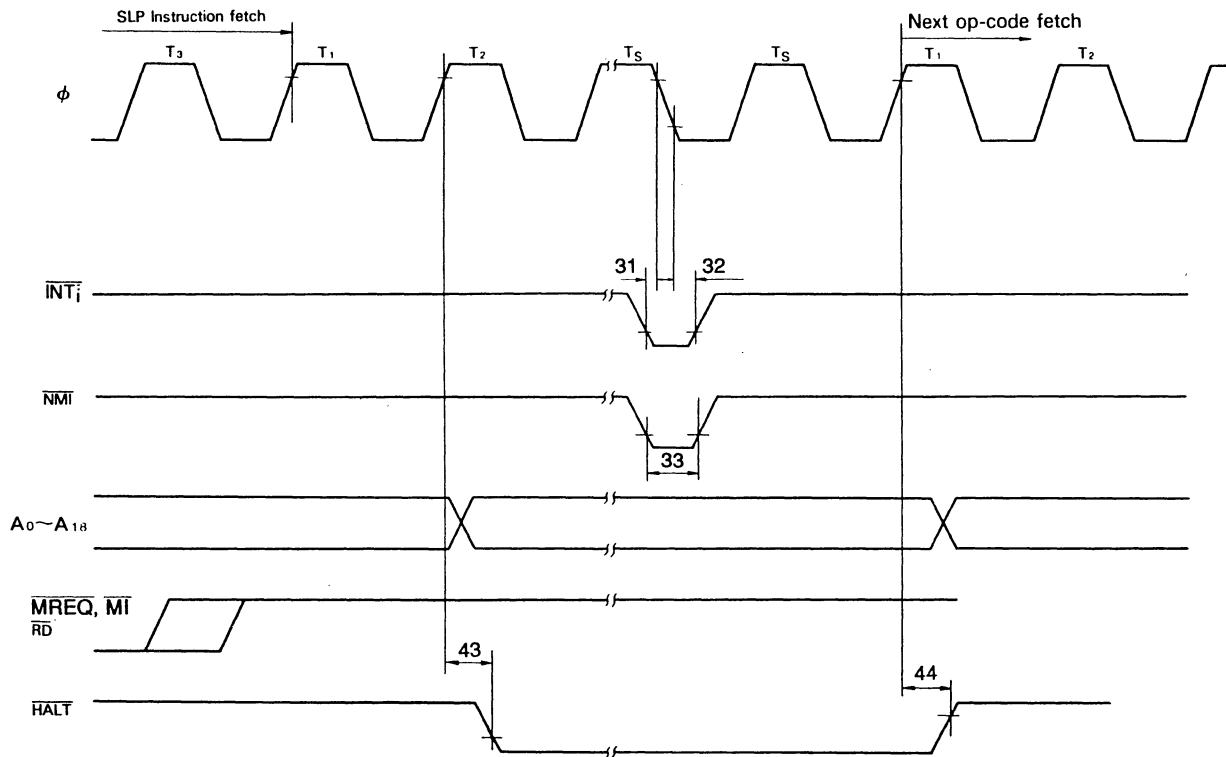




**E Clock Timing (Minimum timing example)  
of  $P_{WEL}$  and  $P_{WEH}$**



**Timer Output Timing**



SLP Execution Cycle

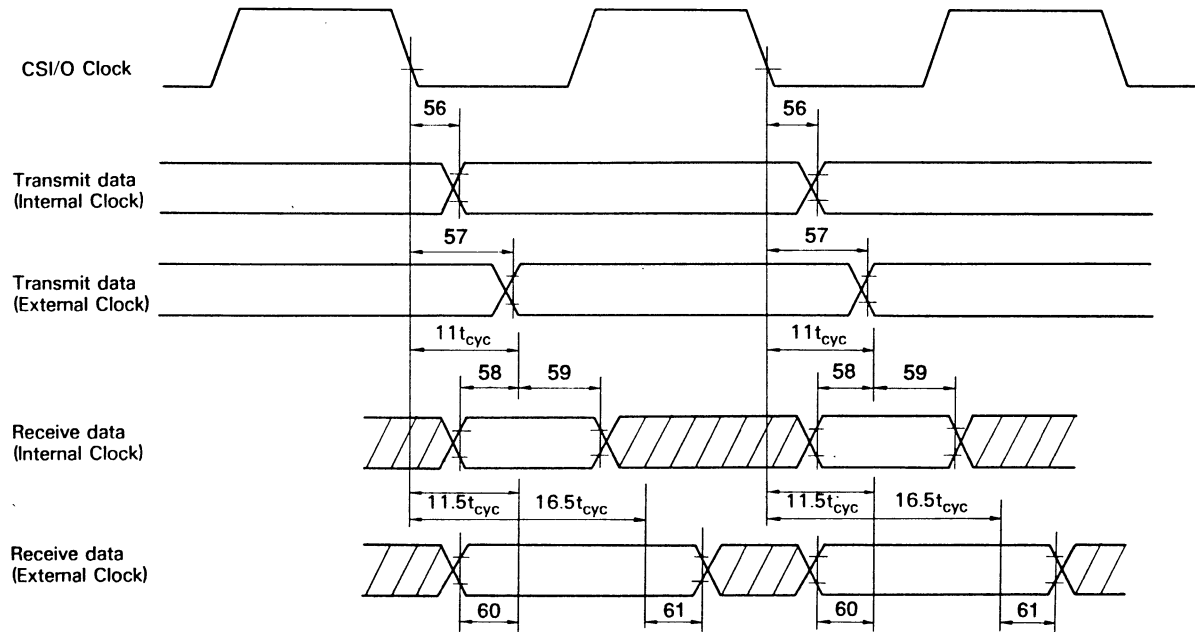
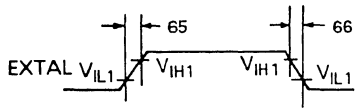


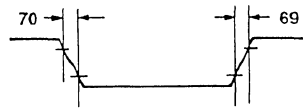
Figure 68. CSI/O Receive/Transmit Timing

---

**TIMING DIAGRAMS** (Continued)



**External Clock Rise Time and Fall Time**



**Input Rise Time and Fall Time  
(Except EXTAL, RESET)**

---



## Z80181

ZIO™ CONTROLLER  
ZILOG I/O CONTROLLER

### FEATURES

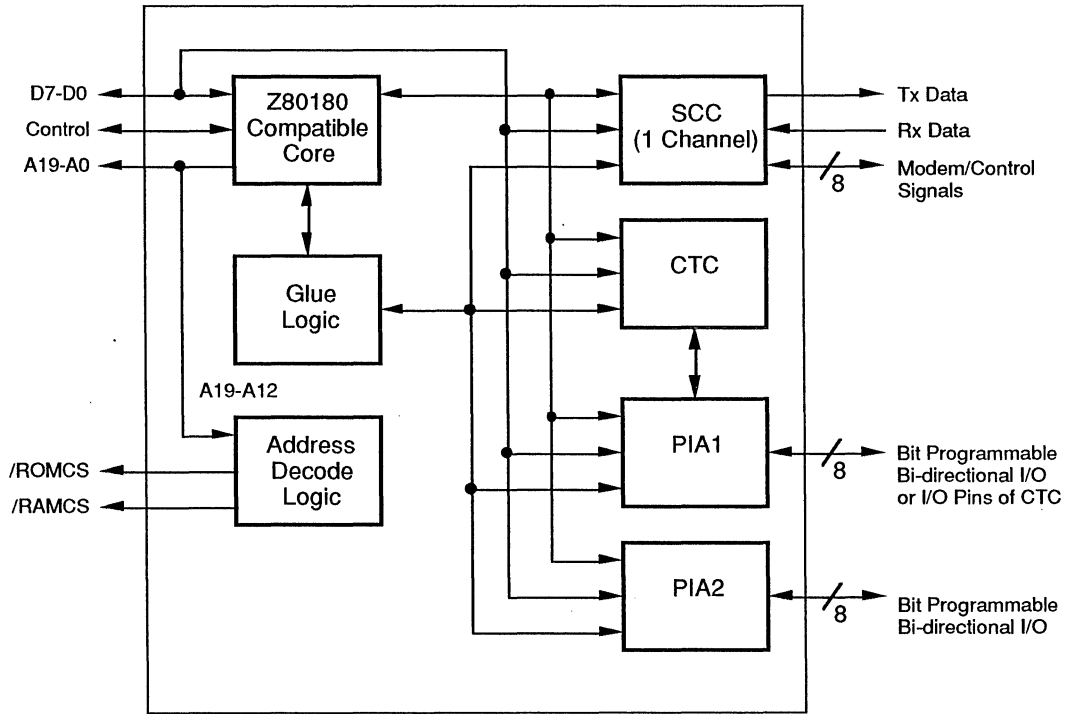
- Z80180 Compatible MPU Core with 1 channel of Z85C30 SCC, Z80 CTC, two 8-bit general purpose parallel ports, and two chip select signals.
- High speed operation (10/12.5 MHz)
- Low power consumption in two operating modes:
  - 55 mA Typ. (Run mode)
  - 30 mA Typ. (STOP mode)
- Wide operational voltage range (5V ± 10%)
- TTL/CMOS compatible
- Clock Generator
- One channel of Z85C30 Serial Communication Controller (SCC)
- Z180 Compatible MPU core, which has:
  - Enhanced Z80 CPU core
  - Memory Management Unit (MMU) enables access to 1Mbyte of memory
  - Two Asynchronous channels
  - Two DMA channels
  - Two 16-bit Timers
  - Clocked serial I/O Port
- Z84C30 CTC
- Two 8-bit general purpose parallel ports
- Memory configurable RAM and ROM chip select pins
- 100-pin QFP Package

### GENERAL DESCRIPTION

The Z80181 I/O Controller (hereinafter, referred to as Z181) is a CMOS 8-bit microprocessor. It is integrated with the Z180 compatible MPU (Z181 MPU), one channel of Z85C30 Serial Communication Controller (SCC), Z80 CTC, two 8-bit general purpose parallel ports, and two chip select signals, all into a single 100-pin QFP (Quad Flat Pack) package. This high-end superintegrated intelligent peripheral controller is targeted for a broad range of intelligent communication control applications, i.e., terminals, printers, modems, and slave communication proces-

sors for 8-, 16- and 32-bit MPU based systems. Also included are enhancement/cost reductions of existing hardware using Z80/Z180 with Z8530/Z85C30 applications. Figure 1 shows the block diagram of the Z80181.

**Note: All Signals with a preceding front slash, "/", are active Low e.g.; B/W (WORD is active Low); /B/W (BYTE is active Low, only); /N/S (NORMAL and SYSTEM are both active Low).**



$$\text{Z80181} = \text{Z180} + \text{SCC}/2 + \text{CTC} + \text{PIA}$$

Figure 1. Z80181 Block Diagram

## PIN DEFINITIONS

The pin assignment is shown on Figure 2. Following is the description on each pin.

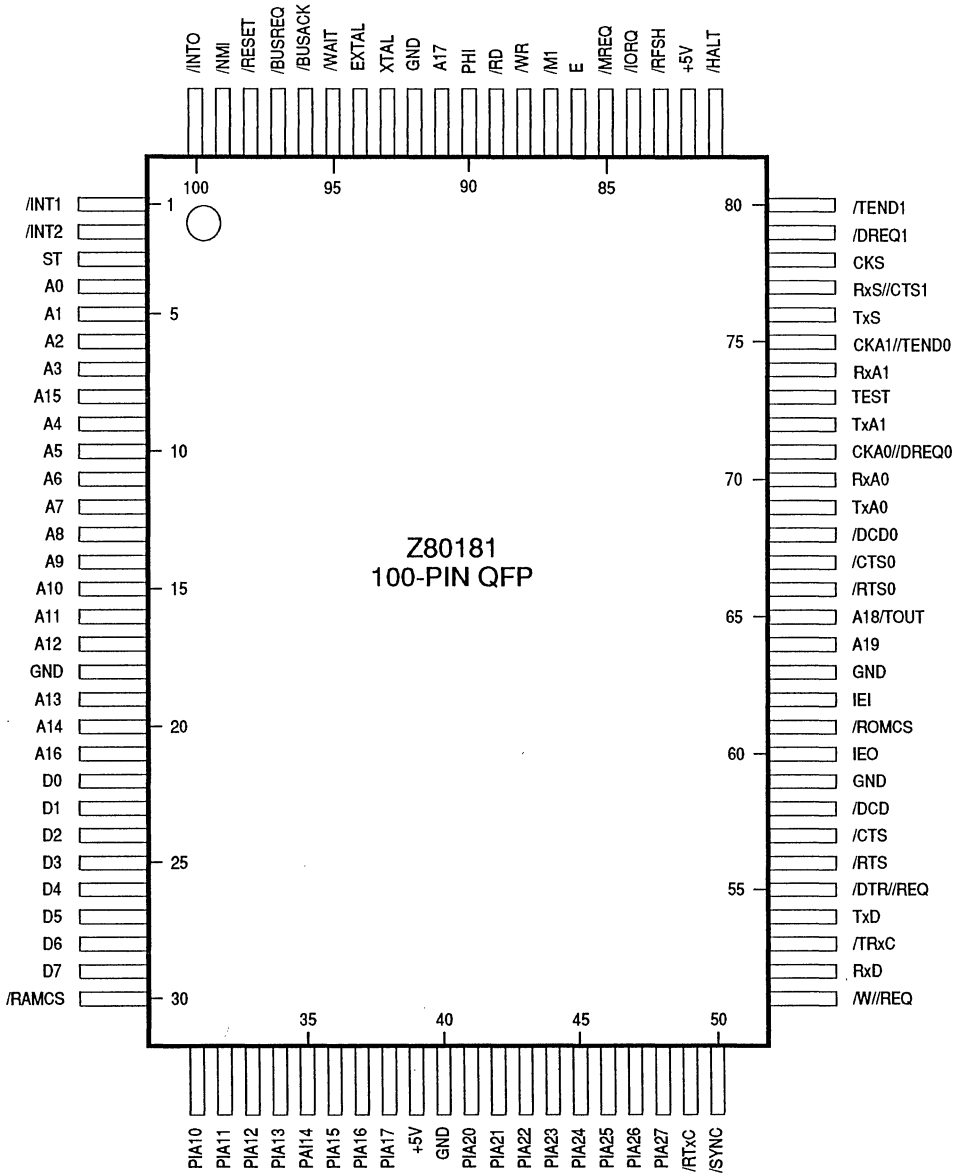


Figure 2. Z80181 Pin-out Assignment



## CPU SIGNALS

Pin Name	Pin Number	Input/Output, 3-State	Function
A19 - A0	4-17, 19-21, 64, 65, 91	I/O, Active 1	<b>Address Bus.</b> A19 - A0 form a 20-bit address bus which specifies I/O and memory addresses to be accessed. During the refresh period, addresses for refreshing are output. The address bus enters a high-impedance state during Reset and external bus acknowledge cycles. The bus is an input when the external bus master is accessing the on-chip peripherals. Address line A18 is multiplexed with the output of PRT Channel 1 (TOUT, selected as address output on Reset).
D7-D0	22-29	I/O, Active 1	<b>8-bit bidirectional data bus.</b> When the on-chip CPU is accessing on-chip peripherals, these lines are outputs and hold the data to/from the on-chip peripherals.
/RD	89	I/O, Active 0	<b>Read signal.</b> CPU read signal for accepting data from memory or I/O devices. When an external master is accessing the on-chip peripherals, it is an input signal.
/WR	88	I/O, Active 0	<b>Write Signal.</b> This signal is active when data to be stored in a specified memory or peripheral device is on the MPU data bus. When an external master is accessing the on-chip peripherals, it is an input signal.
/MREQ	85	I/O, 3-State, Active 0	<b>Memory request signal.</b> When an effective address for memory access is on the address bus, /MREQ is active. This signal is analogous to the /ME signal of the Z64180.
/IORQ	84	I/O, 3-State, Active 0	<b>I/O request signal.</b> When addresses for I/O are on the lower 8 bits (A7-A0) of the address bus in the I/O operation, "0" is output. In addition, the /IORQ signal is output with the /M1 signal during the interrupt acknowledge cycle to inform peripheral devices that the interrupt response vector is on the data bus. This signal is analogous to the /IOE signal of the Z64180.
/M1	87	Out, 3-State, Active 0	<b>Machine cycle "1".</b> /MREQ and /M1 are active together during the operation code fetch cycle. /M1 is output for every opcode fetch when a two byte opcode is executed. In the maskable interrupt acknowledge cycle, this signal is output together with /IORQ. It is also used with /HALT and ST signal to decode the status of the CPU Machine cycle. This signal is analogous to the /LIR signal of the Z64180.
/RFSH	83	Out, 3-state, Active 0	<b>The Refresh signal.</b> When the dynamic memory refresh address is on the low order 8-bits of the address bus (A7 - A0), /RFSH is active along with the /MREQ signal. This signal is analogous to the /REF signal of the Z64180.

---

## CPU SIGNALS (Continued)

---

Pin Name	Pin Number	Input/Output, 3-State	Function
/INT0	100	Wired-OR I/O, Active 0	<b>Maskable Interrupt Request 0.</b> Interrupt is generated by peripheral devices. This signal is accepted if the interrupt enable Flip-Flop (IFF) is set to "1". Internally, the SCC and CTC's interrupt signals are connected to this line, and require an external pull-up resistor.
/INT1, /INT2	1, 2,	In, Active 0	<b>Maskable Interrupt Request 1 and 2.</b> This signal is generated by external peripheral devices. The CPU honors these requests at the end of current instruction cycle as long as the /NMI, /BUSREQ and /INT0 signals are inactive. The CPU will acknowledge these interrupt requests with an interrupt acknowledge cycle. Unlike the acknowledgment for /INT0, during this cycle, neither /M1 or /IORQ will become active.
/NMI	99	In, Active 0	<b>Non-maskable interrupt request signal.</b> This interrupt request has a higher priority than the maskable interrupt request and does not rely upon the state of the interrupt enable Flip-Flop (IFF).
/HALT	81	Out, 3-State, Active 0	<b>Halt signal.</b> This signal is asserted after the CPU has executed either the HALT or SLP instruction, and is waiting for either non-maskable interrupt maskable interrupt before operation can resume. It is also used with the /M1 and ST signals to decode the status of the CPU machine cycle.
/BUSREQ	97	In, Active 0	<b>BUS request signal.</b> This signal is used by external devices (such as a DMA controller) to request access to the system bus. This request has higher priority than /NMI and is always recognized at the end of the current machine cycle. This signal will stop the CPU from executing further instructions and place the address bus, data bus, /MREQ, /IORQ, /RD and /WR signals into the high impedance state. /BUSREQ is normally wired-OR and a pull-up resistor is externally connected.
/BUSACK	96	Out, Active 0	<b>Bus Acknowledge signal.</b> In response to /BUSREQ signal, /BUSACK informs a peripheral device that the address bus, data bus, /MREQ, /IORQ, /RD and /WR signals have been placed in the high impedance state.
/WAIT	95	Wired-OR I/O, Active 0	<b>Wait signal.</b> /WAIT informs the CPU that the specified memory or peripheral is not ready for a data transfer. As long as /WAIT signal is active, the MPU is continuously kept in the wait state. Internally, the /WAIT signal from the SCC interface logic is connected to this line, and requires an external pull-up resistor.

---

## PERIPHERAL SIGNALS

Pin Name	Pin Number	Input/Output, 3-State	Function
RXA0, RXA1	70, 74	In, Active 1	<b>ASCI Receive data 0 and 1.</b> These signals are the receive data to the ASCI channels.
TXA0, TXA1	69, 72	Out, Active 1	<b>ASCI Transmit data 0 and 1.</b> These signals are the receive data to the ASCI channels. Transmit data changes are with respect to the falling edge of the transmit clock.
/RTS0	66	Out, Active 0	<b>Request to send 0.</b> This is a programmable modem control signal for ASCI channel 0.
/DCD0	68	In, Active 0	<b>Data Carrier Detect 0.</b> This is a programmable modem control signal for ASCI channel 0.
/CTS0	67	In, Active 0	<b>Clear To Send 0.</b> This is a programmable modem control signal for ASCI channel 0.
/CTS1/RXS	77	In, Active 0	<b>Clear To Send 0/Clocked Serial Receive Data.</b> This is a programmable modem control signal for ASCI channel 0. Also, this signal becomes receive data for the CSIO channel under program control. On power-on Reset, this pin is set as RxS.
CKA0//DREQ0	71	I/O, Active 1	<b>Asynchronous Clock0/DMAC0 request.</b> This pin is the transmit and receive clock for the Asynchronous channel 0. Also, under program control, this pin is used to request a DMA transfer from DMA channel 0. DMA0 monitors this input to determine when an external device is ready for a read or write operation. On power-on Reset, this pin is initialized as CKA0.
CKA1//TEND0	75	I/O, Active 1	<b>Asynchronous Clock1/DMAC0 Transfer end.</b> This pin is the transmit and receive clock for the Asynchronous channel 1. Also, under program control, this pin becomes /TEND0 and is asserted during the last write cycle of the DMA0 operation and is used to indicate the end of the block transfer. On power-on Reset, this pin initializes as CKA1.
/TEND1	80	Out, Active 0	<b>DMAC1 Transfer end.</b> This pin is asserted during the last write cycle of the DMA1 operation and is used to indicate the end of the block transfer.
CKS	78	I/O, Active 1	<b>CSIO clock</b> This line is the clock for the CSIO channel.
TXS	76	Out, Active 1	<b>CSIO Tx Data.</b> This line carries the transmit data from the CSIO channel.
/DREQ1	79	In, Active 0	<b>DMAC1 request.</b> This pin is used to request a DMA transfer from DMA channel 1. DMA1 monitors this input to determine when an external device is ready for a read or write operation.

---

## SCC SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
/W//REQ	51	Active 0	<b>Wait/Request.</b> Open-drain when programmed for a Wait function, driven "1" or "0" when programming for a Request function. Used as /WAIT or /REQUEST depending upon SCC programming. When programmed as /WAIT, this signal is asserted to alert the CPU that addressed memory or I/O devices are not ready and that the CPU should wait. When programmed as /REQUEST, this signal is asserted when a peripheral device associated with a DMA port is ready to read/write data. After reset, this pin becomes "/WAIT".
/SYNC	50	I/O, Active 0	<b>Synchronization.</b> This pin can act either as input, output, or part of the crystal oscillator circuit. In asynchronous receive mode (crystal oscillator option not selected), this pin is an input similar to /CTS and /DCD. In this mode, transitions on this line affect the state of the Sync/Hunt status bit in Read Register 0 but has no other function.  In external sync mode with crystal oscillator option not selected, this line also acts as an input. In this mode, /SYNC must be driven "0" two receive clock cycles after the last bit in the synchronous character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of /SYNC.  In internal sync mode (Monosync and Bisync) with the crystal oscillator option not selected, this line acts as output and is active only during the part of the receive clock cycle in which a synchronous character is recognized (regardless of character boundaries). In SDLC mode, this pin acts as an output and is valid on receipt of a flag.
RxD	52	In, Active 1	<b>Receive Data.</b> This input signal receives serial data at standard TTL levels.
/RTxC	49	In, Active 0	<b>Receive/Transmit clock.</b> This pin can be programmed in several different modes of operation. /RTxC may supply the receive clock, the transmit clock, the clock for the Baud Rate Generator, or the clock for the Digital Phase-Locked Loop. This pin can also be programmed for use with the /SYNC pin as a crystal oscillator. The receive clocks can be 1, 16, 32, or 64 times the data transfer rate in Asynchronous mode.

---

---

**SCC SIGNALS** (Continued)

---

Pin Name	Pin Number	Input/Output, 3-State	Function
/TRxC	53	I/O, Active 0	<b>Transmit/Receive Clock.</b> This pin can be programmed in several different modes of operation. /TRxC can supply the receive clock or the transmit clock in the input mode. Also, it can supply the output of the Digital Phase-Locked Loop, the crystal oscillator, the Baud Rate Generator, or the transmit clock in the output mode.
TxD	54	Out, Active 1	<b>Transmit Data.</b> This Output signal transmits serial data at standard TTL level.
/DTR//REQ	55	Out, Active 0	<b>Data Terminal Ready/Request.</b> This output follows the state programmed into the DTR bit. It can also be used as general purpose output or as Request line for a DMA controller.
/RTS	56	Out, Active 0	<b>Request To Send.</b> When the RTS bit in Write Register 5 is set, the /RTS signal goes low. When the RTS bit is reset in Asynchronous mode and auto enable is on, the signal goes high after the transmitter is empty. In synchronous mode or in Asynchronous mode, with Auto Enable off, the /RTS pin follows the state of the RTS bit. This pin can be used as a general purpose output.
/CTS	57	In, Active 0	<b>Clear To Send.</b> If this pin is programmed as auto enable, a "0" on the input enables the transmitter. If not programmed as Auto Enable, it may be used as a general purpose input. This input is Schmitt-trigger buffered to accommodate inputs with slow rise times. The SCC detects pulses on this input and can interrupt the CPU on both logic level transitions.
/DCD	58	In, Active 0	<b>Data Carrier Detect.</b> This pin functions as receiver enable if it is programmed for auto enable. Otherwise, it may be used as a general purpose input. This input is Schmitt-trigger buffered to accommodate slow rise-time inputs. The SCC detects pulses on this input and can interrupt the CPU on both logic level transitions.

---

---

## PIA/CTC SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
PIA17-PIA14	35-38	I/O	<b>Port 1 Data 7-Port 1 Data 4 or CTC ZC/TO3 - ZC/TO0.</b> These lines can be configured as inputs or outputs on a bit -by-bit basis. Also, under program control, these bits become Z80 CTC's ZC/TO3 - ZC/TO0, and in either timer or counter mode, pulses are output when the down counter has reached zero. On reset, these signals function as PIA17-14 and are inputs.
PIA13-PIA10	31-34	I/O	<b>Port 1 Data 3-Port 1 Data 0 or CTC CLK/TRG3-0.</b> These lines can be configured as inputs or outputs on a bit by bit basis. Also, under program control, these bits become Z80 CTC's CLK/TRG3-CLK/TRG0, and correspond to four Counter/Timer Channels. In the counter mode, each active edge causes the downcounter to decrement by one. In timer mode, an active edge starts the timer. It is program selectable whether the active edge is rising or falling. On reset, these signals are set to PIA13-10 as inputs.
PIA27-20	41-48	I/O	<b>Port 2 Data.</b> These lines are configured as inputs or outputs on a bit-by-bit basis. On reset, they are inputs.

---

## SYSTEM CONTROL SIGNALS

---

Pin Name	Pin Number	Input/Output, 3-State	Function
ST	3	Out, Active 1	<b>Status.</b> This signal is used with the /M1 and /HALT output to decode the status of the CPU machine cycle. Note that the /M1 output is affected by the status of the M1E bit in the OMCR register. The following table shows the status while M1E=1.

ST	/HALT	/M1	Operation
0	1	0	CPU Operation (1st Op-code fetch)
1	1	0	CPU Operation (2nd and 3rd Op-code fetch)
1	1	1	CPU Operation (MC other than Op-code fetch)
0	X	1	DMA operation
0	0	0	HALT mode
1	0	1	SLEEP mode (Incl. System STOP mode)

---

---

## SYSTEM CONTROL SIGNALS (Continued)

---

Pin Name	Pin Number	Input/Output, 3-State	Function
IEI	62	In, Active 1	<b>Interrupt enable input signal.</b> IEI is used with the IEO to form a priority daisy chain when there is more than one interrupt-driven peripheral.
IEO	60	Out, Active 1	<b>The interrupt enable output signal.</b> In the daisy-chain interrupt control, IEO controls the interrupt of external peripherals. IEO is active when IEI is "1" and the CPU is not servicing an interrupt from the on-chip peripherals.
/ROMCS	61	Out, Active 0	<b>ROM Chip select.</b> Used to access ROM. Refer to "Functional Description" on chip select signals for further explanation.
/RAMCS	30	Out, Active 0	<b>RAM Chip Select.</b> Used to access RAM. Refer to "Functional Description" on chip select signals for further explanation.
/RESET	98	In, Active 0	<b>Reset signal.</b> /RESET signal is used for initializing the MPU and other devices in the system. It must be kept in the active state for a period of at least 3 system clock cycles.
EXTAL	94	In, Active 1	<b>Crystal oscillator connecting terminal.</b> A parallel resonant crystal is recommended. If an external clock source is used as the input to the Z180 Clock Oscillator unit, supply the clock into this terminal.
XTAL	93	Out	<b>Crystal oscillator connecting terminal.</b>
PHI	90	Out, Active 1	<b>System Clock.</b> Single-phase clock output from Z181 MPU.
E	86	Out, Active 1	<b>Enable Clock.</b> Synchronous Machine cycle clock output during a bus transaction.
TEST	73	Out	<b>Test pin.</b> Used in the open state.
V <sub>cc</sub>	39, 82		<b>Power Supply.</b> +5 Volts
V <sub>ss</sub>	18, 40, 59, 63, 92		<b>Power Supply.</b> 0 Volts

---

## FUNCTIONAL DESCRIPTION

Functionally, the on-chip Z181 MPU, SCC, and CTC are the same as the discrete devices (Figure 1). Therefore, for a detailed description of each individual unit, refer to the

Product Specification/Technical Manual of each discrete product. The following subsections describe each individual functional unit of the Z181.

### Z181 MPU

This unit provides all the capabilities and pins of the Zilog Z180 MPU. Figure 3 shows the Z181 MPU block diagram. This allows 100% software compatibility with existing Z180 (and Z80) software. Note that the on-chip I/O address

should not be relocated to the I/O address (from 0C0h to 0FFh) to avoid address conflicts. The following is an overview of the major functional units of the Z181.

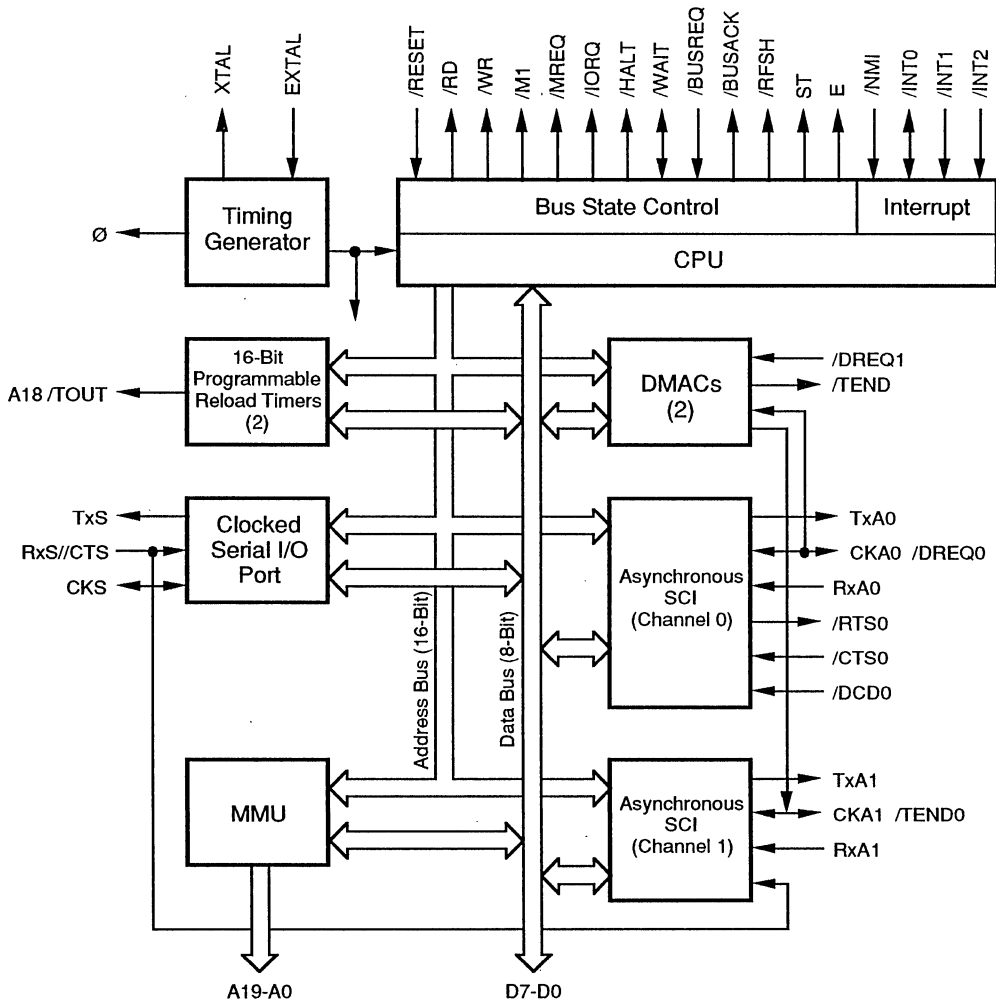


Figure 3. Z181 MPU Block Diagram



---

## FUNCTIONAL DESCRIPTION (Continued)

### Z181 CPU

The Z181 CPU has 100% software compatibility with the Z80 CPU. In addition, the Z181 CPU has the following features:

**Faster execution speed.** The Z181 CPU is "fine tuned" making execution speed, on average, 10% to 20% faster than the Z80 CPU.

**Enhanced DRAM Refresh Circuit.** Z181 CPU's DRAM refresh circuit does periodic refresh and generates an 8-bit refresh address. It can be disabled or the refresh period adjusted, via software control.

**Enhanced Instruction Set.** The Z181 CPU has seven additional instructions to those of the Z80 CPU which include the MLT (Multiply) instruction.

**HALT and Low Power Modes of Operation.** The Z181 CPU has HALT and low power modes of operation, which are ideal for the applications requiring low power consumption like battery operated portable terminals.

**System Stop Mode.** When the Z181 is in SYSTEM STOP mode, it is only the Z181 MPU which is in STOP mode. The on-chip CTC and SCC continue their normal operation.

**Instruction Set.** The instruction set of the Z181 CPU is identical to the Z180. For more details about each transaction, please refer to the Data Sheet/Technical Manual for the Z180/Z80 CPU.

### Z181 CPU Basic Operation

Z181 CPU's basic operation consists of the following events. These are identical to the Z180 MPU. For more details about each operation, please refer to the Data Sheet/Technical manual for the Z180.

- Operation code fetch cycle.
- Memory Read/Write operation.
- Input/Output operation.
- Bus request/acknowledge operation.
- Maskable interrupt request operation.
- Trap and Non-Maskable interrupt request operation.
- HALT and low power modes of operation.
- Reset Operation.

### Memory Management Unit (MMU)

The Memory Management Unit (MMU) allows the user to "map" the memory used by the CPU (64 Kbytes of logical addressing space) into 1 Mbyte of physical addressing space. The organization of the MMU allows object code compatibility with the Z80 CPU while offering access to an extended memory space. This is accomplished by using an effective "common area-banked area" scheme.

### DMA Controller

The Z181 MPU has two DMA controllers. Each DMA controller provides high-speed data transfers between memory and I/O devices. Transfer operations supported are memory to memory, memory to/from I/O, and I/O to I/O. Transfer modes supported are request, burst, and cycle steal. The DMA can access the full 1M bytes addressing range with a block length up to 64K bytes and can cross over 64K boundaries.

### Asynchronous Serial Communication Interface (ASCI)

This unit provides two individual full-duplex UARTs. Each channel includes a programmable baud rate generator and modem control signals. The ASCI channels also support a multiprocessor communication format.

### Programmable Reload Timer (PRT)

The Z181 MPU has two separate Programmable Reload Timers, each containing a 16-bit counter (timer) and count reload register. The time base for the counters is system clock divided by 20. PRT channel 1 provides an optional output to allow for waveform generation.

### Clocked Serial I/O (CSI/O)

The CSI/O channel provides a half-duplex serial transmitter and receiver. This channel can be used for simple high-speed data connection to another CPU or MPU.

### Programmable Wait State Generator

To ease interfacing with slow memory and I/O devices, the Z181 MPU unit has a programmable wait state generator. By programming the DMA/WAIT Control Register (DCNTL), up to three wait states are automatically inserted in memory and I/O cycles. This unit also inserts wait states during on-chip DMA transactions.

### Z85C30 Serial Communication Controller Logic Unit

This logic unit provides the user with a multi-protocol serial I/O channel that is completely compatible with the two channel Z85C30 SCC with the following exceptions:

Their basic functions as serial-to-parallel and parallel-to-serial converters can be programmed by the CPU for a

broad range of serial communications applications. This logic unit is capable of supporting all common asynchronous and synchronous protocols (Monosync, Bisync, and SDLC/HDLC, byte or bit oriented - Figure 4).

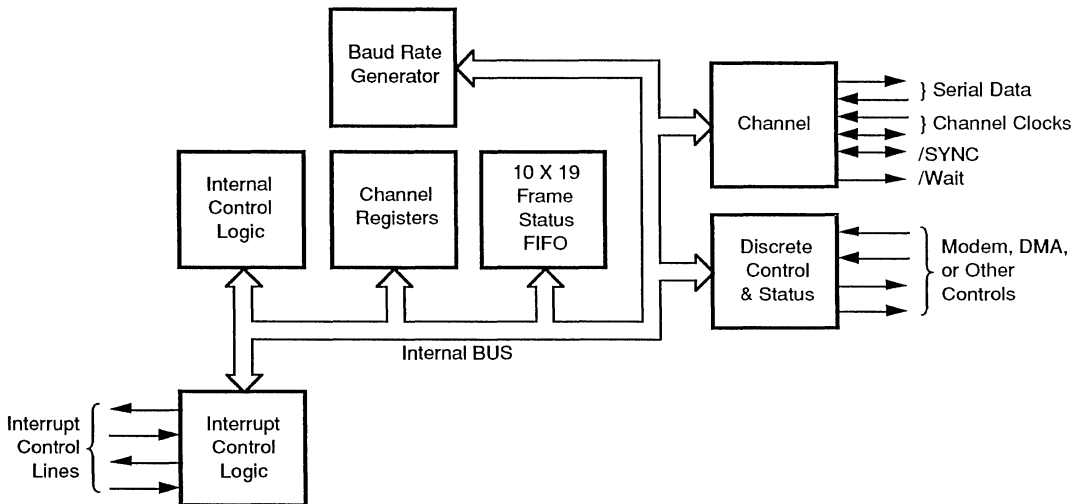


Figure 4. SCC Block Diagram

On the discrete version of the SCC (dual channel version), there are two registers shared between channels A and B, and two registers whose functions are different by channel. These are: WR2, WR9 (shared registers), and RR2 and RR3 (different functionality).

Following are the differences in functionality:

- **RR2** - Returns unmodified vector or modified vector which depends on the status of VIS (Vector Include Status) bit in WR9.
- **RR3** - Returns IP status (Ch.A side).
- **WR9** - Ch.B Software Reset command has no effect.

The PCLK for the SCC is connected to PHI (System clock), the /INT signal is connected to /INT0 signal internally (requires external pull-up resistor) and SCC is reset when /RESET input becomes active. Interrupt from the SCC is handled via Mode 2 interrupt. During the interrupt acknowledge cycle, the on-chip SCC interface circuit inserts two wait states automatically.

#### Z84C30 Counter/Timer Logic Unit

This logic unit provides the user with four individual 8-bit Counter/Timer Channels that are compatible with the Z84C30 CTC (Figure 5). The Counter/Timers are programmed by the CPU for a broad range of counting and timing applications. Typical applications include event counting, interrupt and interval counting, and serial baud rate clock generation.

Each of the Counter/Timer Channels, designated Channels 0-3, have an 8-bit prescaler (when used in timer mode) and its own 8-bit counter to provide a wide range of count resolution. Each of the channels have their own Clock/Trigger input to quantify the counting process and an output to indicate zero crossing/timeout conditions. These signals are multiplexed with the Parallel Interface Adapter 1 (PIA1). With only one interrupt vector programmed into the logic unit, each channel can generate a unique interrupt vector in response to the interrupt acknowledge cycle.

## FUNCTIONAL DESCRIPTION (Continued)

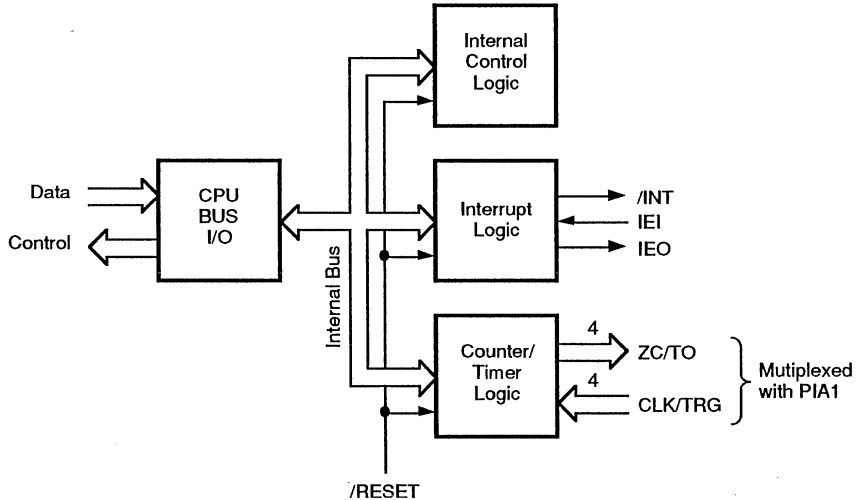


Figure 5. CTC Block Diagram

### Parallel Interface Adapter (PIA)

The Z181 has two 8-bit Parallel Interface Adapter (PIA) Ports. The ports are referred to as PIA1 and PIA2. Each port has two associated control registers; a Data Register and a register to determine each bit's direction (input or output). PIA1 is multiplexed with the CTC I/O pins. When the CTC I/O feature is selected, the CTC I/O functions override the PIA1 feature. Mode Selection is made through the System Configuration Register (Address: EDh; Bit D0). PIA1 has Schmitt-trigger inputs to have a better noise margin. These ports are inputs after reset.

### Clock Generator

The Z181 Controller uses the Z181 MPU's on-chip clock generator to supply system clock. The required clock is easily generated by connecting a crystal to the external terminals (XTAL, EXTAL). The clock output runs at half the crystal frequency. The system clock inputs of the SCC and the CTC are internally connected to the PHI output of the Z181 MPU.

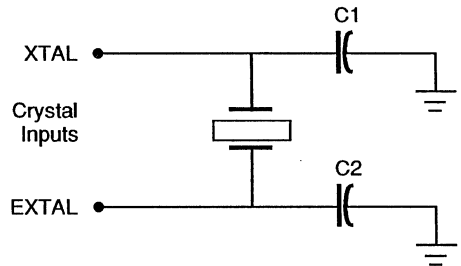


Figure 6. Circuit Configuration For Crystal

---

Recommended characteristics of the crystal and the values for the capacitor are as follows (the values will change with crystal frequency).

**Type of crystal:** Fundamental, parallel type crystal (AT cut is recommended).

**Frequency tolerance:** Application dependent.

**CL, Load capacitance:** Approximately 22 pF (acceptable range is 20-30 pF)

**Rs, equivalent-series resistance:**  $\leq 30$  Ohms

**Drive level:** 10mW (for  $\leq 10$  MHz crystal) 5 mW (for  $\geq 10$  MHz crystal)

$C_{IN} = C_{OUT} = 15 \sim 22$  pF.

### Chip Select Signals

The Z181 has two chip select (/RAMCS, /ROMCS) pins. /ROMCS is the chip select signal for ROM and /RAMCS is the chip select signal for RAM. The boundary value for each chip select signal is 8 bits wide allowing all memory accesses with addresses less than or equal to this boundary value. This causes assertion of the corresponding /CS pin. These features are controlled via the RAM upper boundary address register (I/O address EAH), RAM lower boundary address register (I/O address EBH) and ROM upper boundary address register (I/O address ECH).

These two signals are generated by decoding address lines A19-A12. Note that glitches may be observed on the /RAMCS and /ROMCS signals because the address decoding logic decodes only A19-A12, without any control signals.

Bit D5 of the System Configuration Register allows the option of disabling the /ROMCS signal. This feature is used in systems which, for example, have a shadow RAM. However, prior to disabling the /ROMCS signal, the ROMBR and RAMLBR registers must be re-initialized from their default values.

For more details, please refer to "Programming section".

### ROM Emulator Mode

To ease development, the Z181 has a mode to support "ROM emulator" development systems. In this mode, a read data from on-chip registers (except Z181 MPU on-chip registers) are available (data bus direction set to output) to make data visible from the outside, so that a ROM Emulator/Logic Analyzer can monitor internal transactions. Otherwise, a read from an internal transaction is not available to the outside (data bus direction set to Hi-Z status). Mode selection is made through the D1 bit in the System Configuration Register (I/O Address: EDH).

---

## PROGRAMMING

The following subsections explain and define the parameters for I/O Address assignments, I/O Control Register Addresses and all pertinent Timing parameters.

### I/O Address Assignment

The Z181 has 78 internal 8-bit registers to control on-chip peripherals and features. Sixty-four registers out of 78 registers are occupied by the Z181 MPU control registers;

two for SCC control registers, four for PIA control registers, four for the Counter/Timer, three for RAM/ROM configuration (memory address boundaries) and one for Z181's system control. The Z181 controller's I/O addresses are listed in Table 1. These registers are assigned in the Z181 controller's I/O addressing space and the I/O addresses are fully decoded from A7-A0 and have no image.

---

## PROGRAMMING (Continued)

**Table 1. I/O Control Register Address**

Address*	Register
00 to 3F	Z181 MPU Control Registers (Relocatable to 040H-07FH, or 080H-0BFH)
E0	PIA1 Data Direction Register (P1DDR)
E1	PIA1 Data Port (P1DP)
E2	PIA2 Data Direction Register (P2DDR)
E3	PIA2 Data Register (P2DP)
E4	CTC Channel 0 Control Register (CTC0)
E5	CTC Channel 1 Control Register (CTC1)
E6	CTC Channel 2 Control Register (CTC2)
E7	CTC Channel 3 Control Register (CTC3)
E8	SCC Control Register (SCCCR)
E9	SCC Data Register (SCCDR)
EA	RAM Upper Boundary Address Register (RAMUBR)
EB	RAM Lower Boundary Address Register (RAMLBR)
EC	ROM Address Boundary Register (ROMBR)
ED	System Configuration Register (SCR)
EE	Reserved
EF	Reserved

\* All addresses in Hexadecimal

### Z181 MPU Control Registers

The I/O address for these registers can be relocated in 64 byte boundaries by programming of the I/O Control Register (Address xx111111b).

Do not relocate these registers to address from 0C0H since this will cause an overlap of the Z180 registers and the 16 registers of the Z181 (address 0E0H to 0EFH).

Also, the OMCR register (Address: xx111101b) has to be programmed as 0x0xxxxxb (x: don't care) as a part of the initialization procedure. The M1E bit (Bit D7) of this register must be programmed as 0 or the interrupt daisy chain is corrupted. The /IOC bit (Bit D5) of this register is programmed as 0 so that the timing of the /RD and /IORQ signals are compatible with Z80 peripherals.

For detailed information, refer to the Z180 Technical Manual.

# ASCII CHANNELS CONTROL REGISTERS

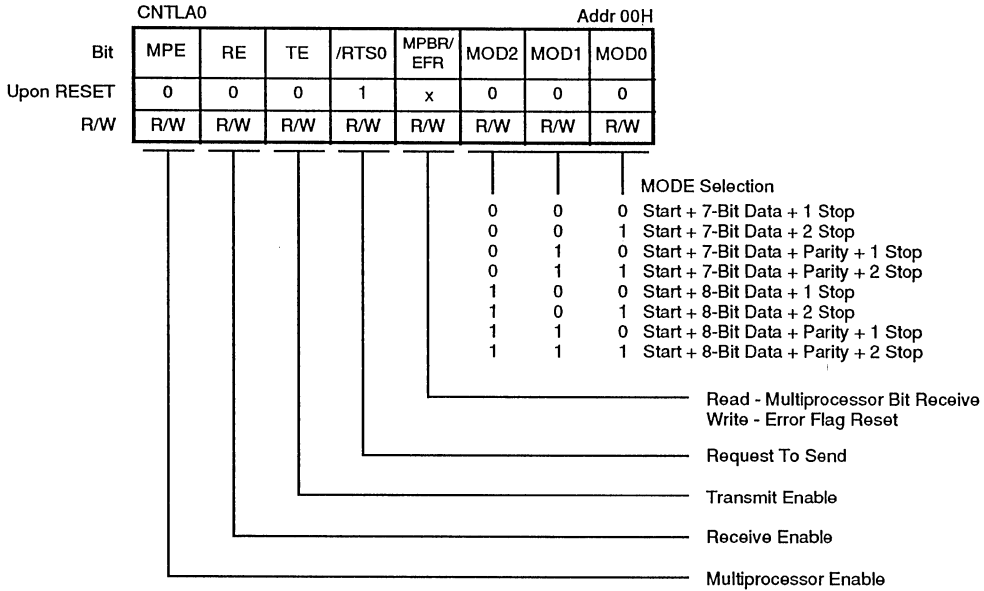


Figure 7. ASCII Control Register A (Ch. 0)

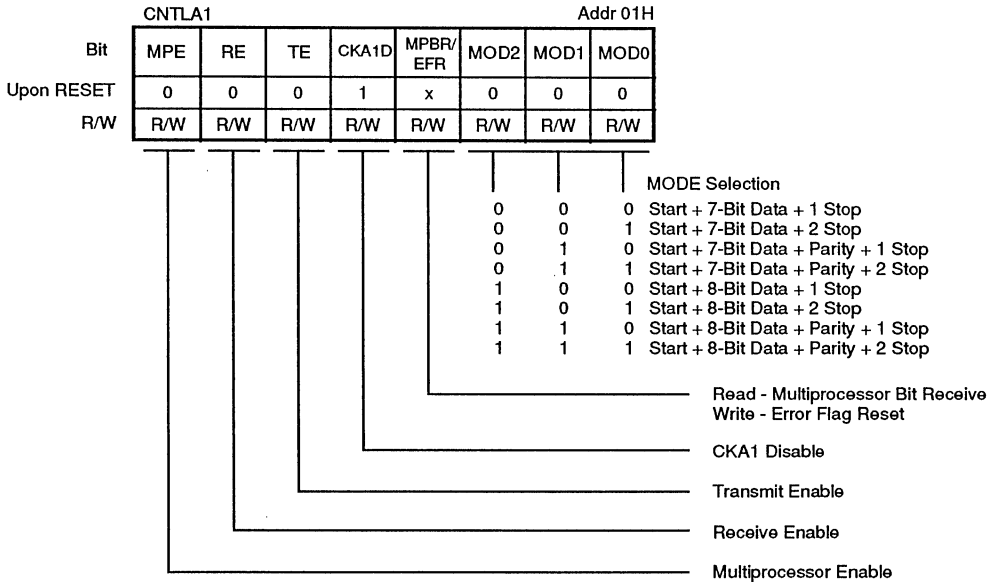
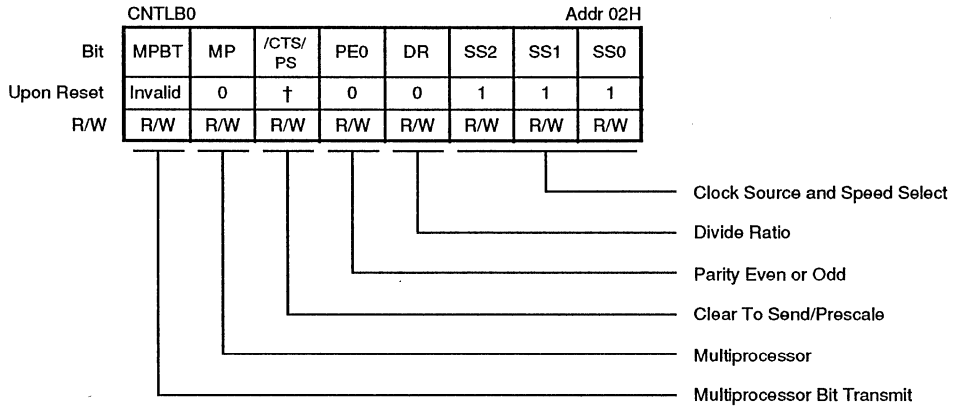


Figure 8. ASCII Control Register A (Ch. 1)

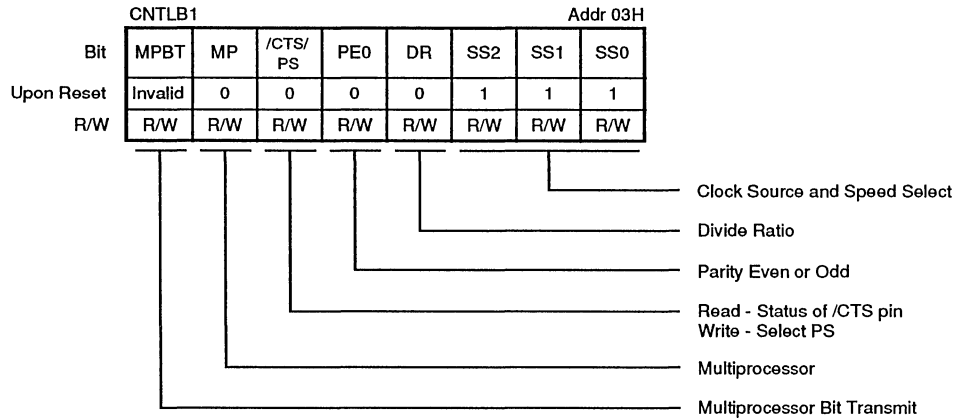
## ASCI CHANNELS CONTROL REGISTERS (Continued)



† /CTS - Depending on the condition of /CTS pin.  
PS - Cleared to 0.

General Divide Ratio	PS = 0 (Divide Ratio = 10)		PS = 1 (Divide Ratio = 30)	
SS, 2, 1, 0	DR = 0 (x16)	DR = 1 (x64)	DR = 0 (x16)	DR = 1 (x64)
000	$\emptyset + 160$	$\emptyset + 640$	$\emptyset + 480$	$\emptyset + 1920$
001	$\emptyset + 320$	$\emptyset + 1280$	$\emptyset + 960$	$\emptyset + 3840$
010	$\emptyset + 640$	$\emptyset + 2580$	$\emptyset + 1920$	$\emptyset + 7680$
011	$\emptyset + 1280$	$\emptyset + 5120$	$\emptyset + 3840$	$\emptyset + 15360$
100	$\emptyset + 2560$	$\emptyset + 10240$	$\emptyset + 7680$	$\emptyset + 30720$
101	$\emptyset + 5120$	$\emptyset + 20480$	$\emptyset + 15360$	$\emptyset + 61440$
110	$\emptyset + 10240$	$\emptyset + 40960$	$\emptyset + 30720$	$\emptyset + 122880$
111	External Clock (Frequency < $\emptyset + 40$ )			

**Figure 9. ASCI Control Register B (Ch. 0)**

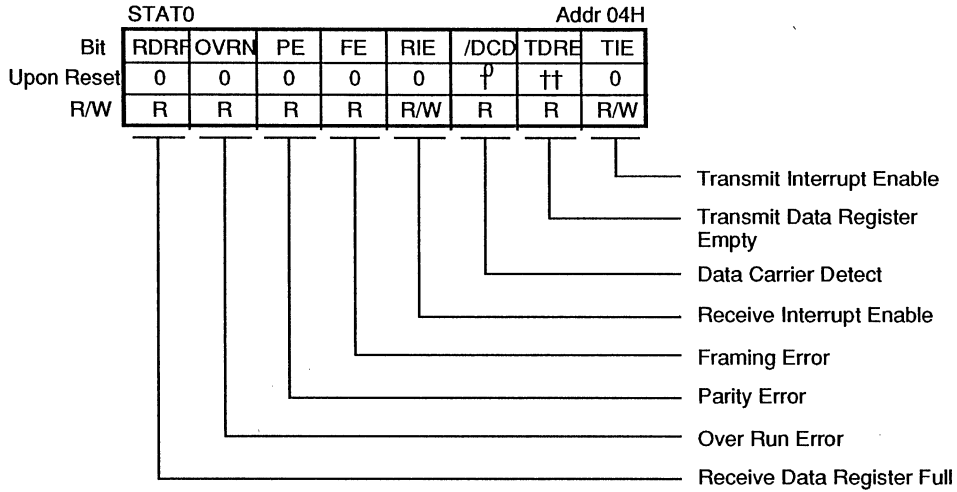


General Divide Ratio	PS = 0 (Divide Ratio = 10)		PS = 1 (Divide Ratio = 30)	
SS, 2, 1, 0	DR = 0 (x16)	DR = 1 (x64)	DR = 0 (x16)	DR = 1 (x64)
000	$\emptyset + 160$	$\emptyset + 640$	$\emptyset + 480$	$\emptyset + 1920$
001	$\emptyset + 320$	$\emptyset + 1280$	$\emptyset + 960$	$\emptyset + 3840$
010	$\emptyset + 640$	$\emptyset + 2580$	$\emptyset + 1920$	$\emptyset + 7680$
011	$\emptyset + 1280$	$\emptyset + 5120$	$\emptyset + 3840$	$\emptyset + 15360$
100	$\emptyset + 2560$	$\emptyset + 10240$	$\emptyset + 7680$	$\emptyset + 30720$
101	$\emptyset + 5120$	$\emptyset + 20480$	$\emptyset + 15360$	$\emptyset + 61440$
110	$\emptyset + 10240$	$\emptyset + 40960$	$\emptyset + 30720$	$\emptyset + 122880$
111	External Clock (Frequency < $\emptyset + 40$ )			

**Figure 10. ASCI Control Register B (Ch. 1)**



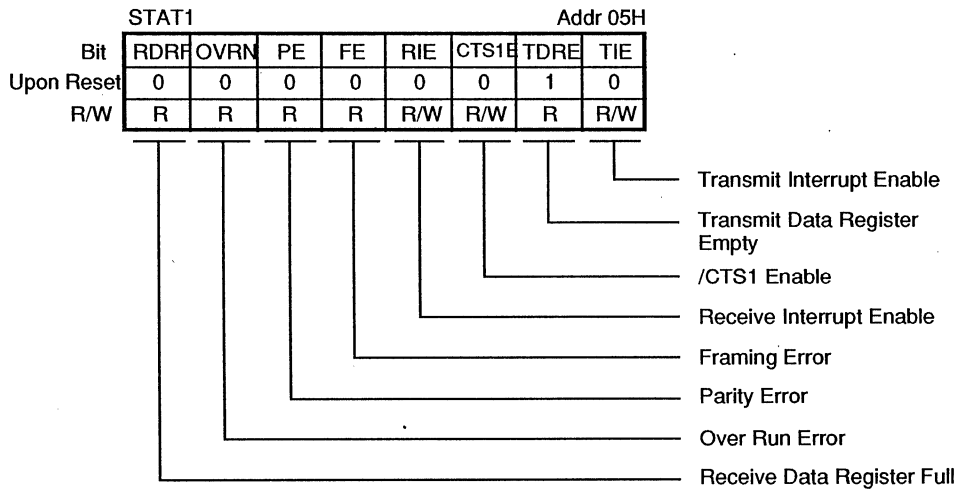
**ASCI CHANNELS CONTROL REGISTERS (Continued)**



† /DCD0 - Depending on the condition of /DCD0 Pin.

†† /CTS0 Pin	TDRE
L	1
H	0

**Figure 11. ASCI Status Register**



**Figure 12. ASCI Status Register (Ch. 1)**

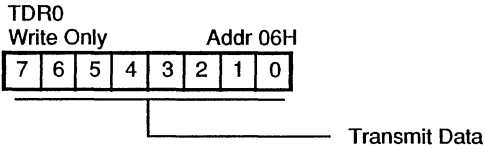


Figure 13. ASCII Transmit Data Register (Ch. 0)

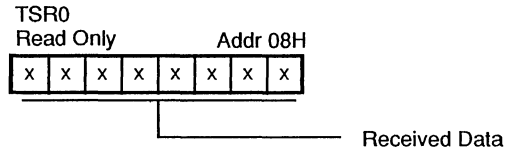


Figure 15. ASCII Receive Data Register (Ch. 0)

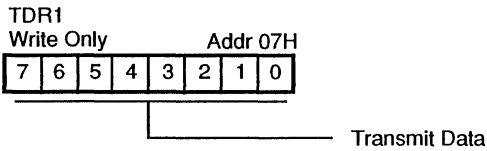


Figure 14. ASCII Transmit Data Register (Ch. 1)

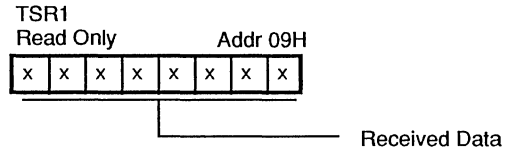
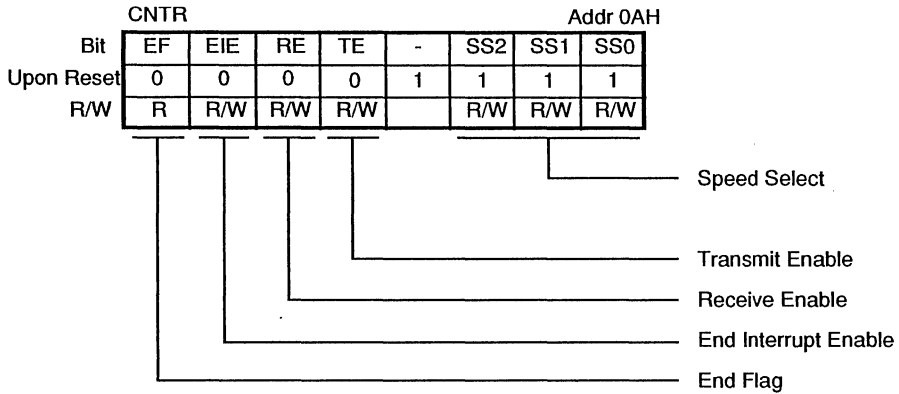


Figure 16. ASCII Receive Data Register (Ch. 1)

## CSI/O REGISTERS



SS2, 1, 0	Baud Rate	SS2, 1, 0	Baud Rate
000	$\emptyset \div 20$	100	$\emptyset \div 320$
001	$\emptyset \div 40$	101	$\emptyset \div 640$
010	$\emptyset \div 80$	110	$\emptyset \div 1280$
011	$\emptyset \div 100$	111	External Clock (Frequency < $\emptyset \div 20$ )

Figure 17. CSI/O Control Register

---

## CSI/O REGISTERS (Continued)

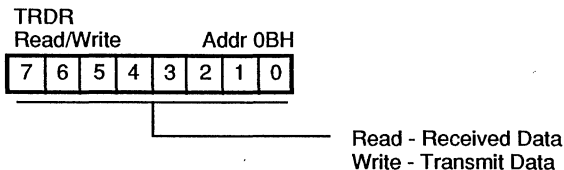


Figure 18. CSI/O Transmit/Receive Data Register

---

## TIMER REGISTERS

### Timer Data Registers

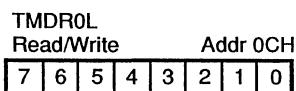


Figure 19. Timer 0 Data Register L

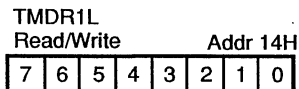
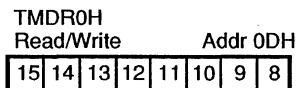
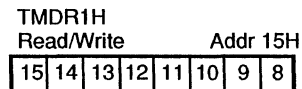


Figure 20. Timer 1 Data Register L



When Read, read Data Register L  
before reading Data Register H.

Figure 21. Timer 0 Data Register H



When Read, read Data Register L  
before reading Data Register H.

Figure 22. Timer 1 Data Register H

---

## TIMER RELOAD REGISTERS

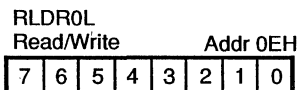


Figure 23. Timer 0 Reload Register L

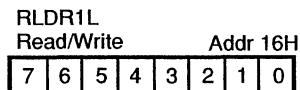


Figure 24. Timer 1 Reload Register L

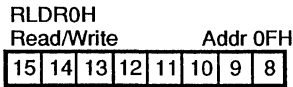


Figure 25. Timer 0 Reload Register H

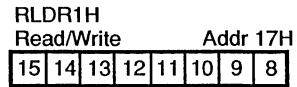


Figure 26. Timer 1 Reload Register H

## TIMER CONTROL REGISTER

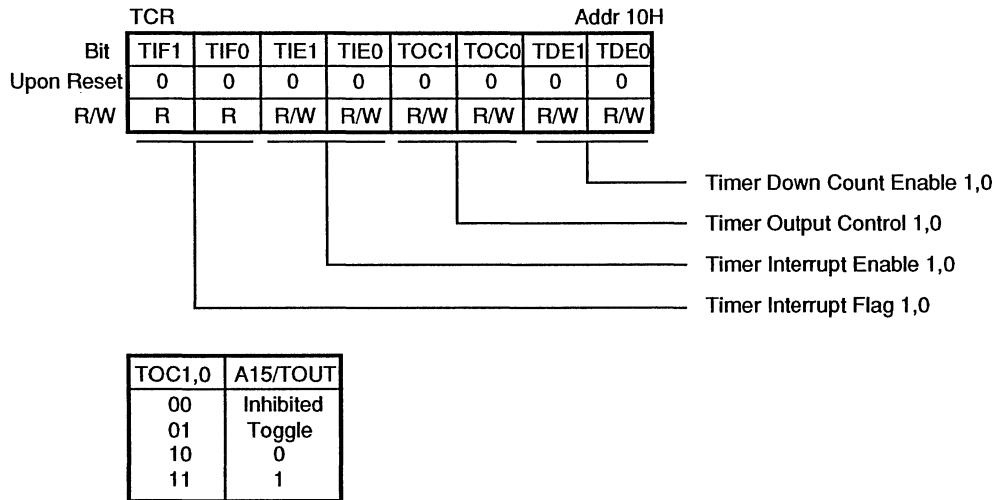


Figure 27. Timer Control Register

## FREE RUNNING COUNTER

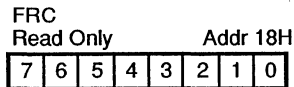
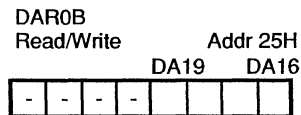
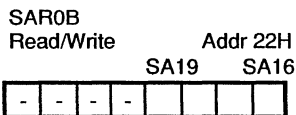
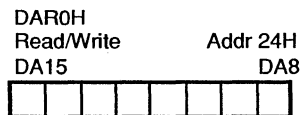
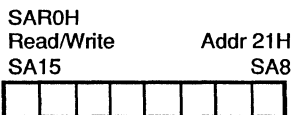
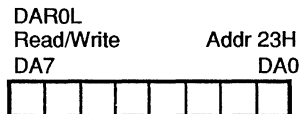
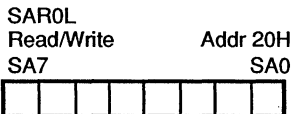


Figure 28. Free Running Counter

## DMA REGISTERS



Bits 0-2 (3) are used for SAR0B

A19, A18, A17, A16	DMA Transfer Request
x x 0 0	/DREQ0 (external)
x x 0 1	RDR0 (ASC10)
x x 1 0	TDR0 (ASC11)
x x 1 1	Not Used

Bits 0-2 (3) are used for DAR0B

A19, A18, A17, A16	DMA Transfer Request
x x 0 0	/DREQ0 (external)
x x 0 1	RDR0 (ASC10)
x x 1 0	TDR0 (ASC11)
x x 1 1	Not Used

Figure 29. DMA 0 Source Address Registers

Figure 30. DMA 0 Destination Address Registers

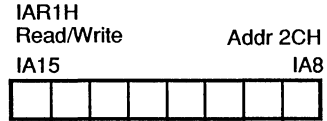
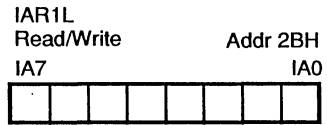
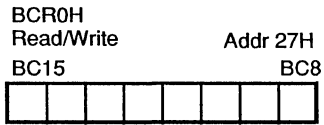
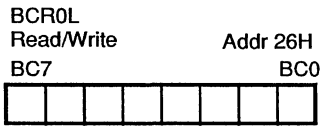


Figure 31. DMA 0 Byte Counter Registers

Figure 33. DMA 1 I/O Address Registers

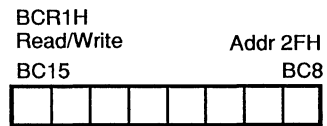
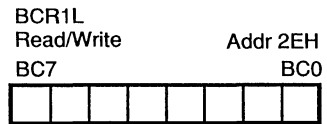
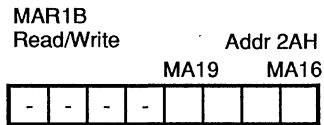
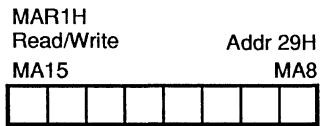
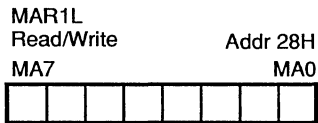


Figure 34. DMA 1 Byte Count Registers

Figure 32. DMA 1 Memory Address Registers

## DMA REGISTERS (Continued)

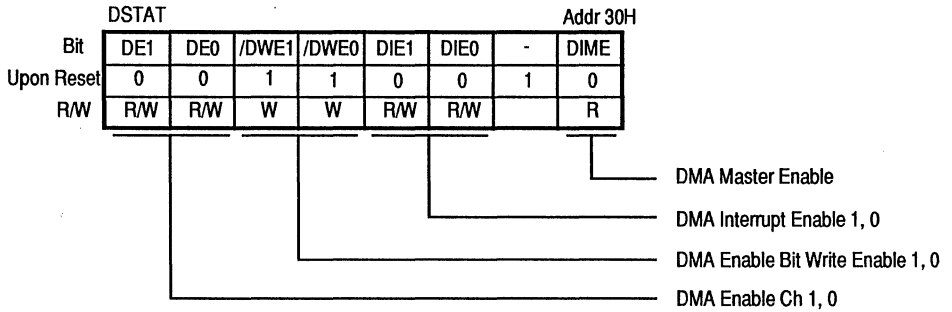
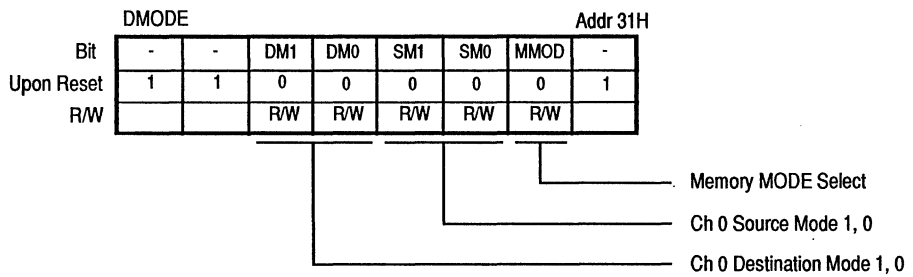


Figure 35. DMA Status Register

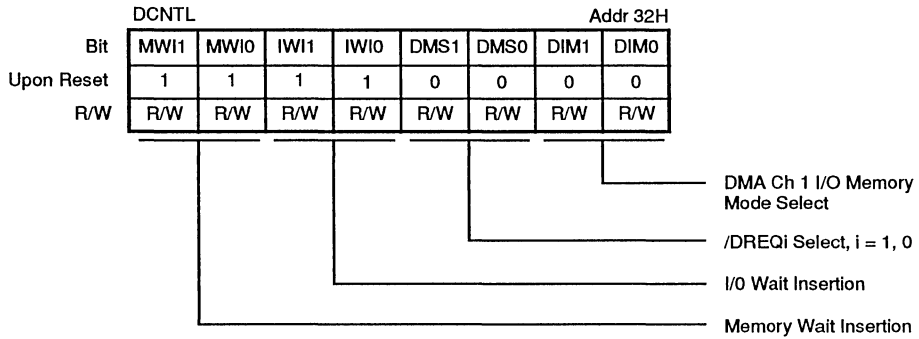


DM1, 0	Destination	Address
00	M	DAR0+1
01	M	DAR0-1
10	M	DAR0 Fixed
11	I/O	DAR0 Fixed

SM1, 0	Source	Address
00	M	SAR0+1
01	M	SAR0-1
10	M	SAR0 Fixed
11	I/O	SAR0 Fixed

MMOD	Mode
0	Cycle Steal Mode
1	Burst Mode

Figure 36. DMA Mode Registers



MW11, 0	No. of Wait States
00	0
01	1
10	2
11	3

IW11, 0	No. of Wait States
00	0
01	2
10	3
11	4

DMSi	Sense
1	Edge Sense
0	Level Sense

DM1, 0	Transfer Mode	Address Increment/Decrement	
00	M - I/O	MAR1+1	IAR1 Fixed
01	M - I/O	MAR1-1	IAR1 Fixed
10	I/O - M	IAR1 Fixed	MAR1+1
11	I/O - M	IAR1 Fixed	MAR1-1

**Figure 37. DMA/WAIT Control Register**



## MMU REGISTERS

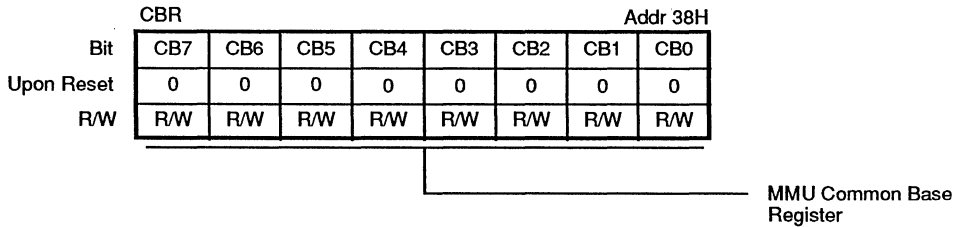


Figure 38. MMU Common Base Register

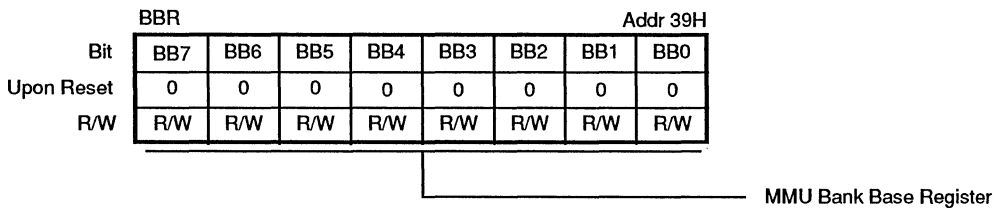


Figure 39. MMU Bank Base Register

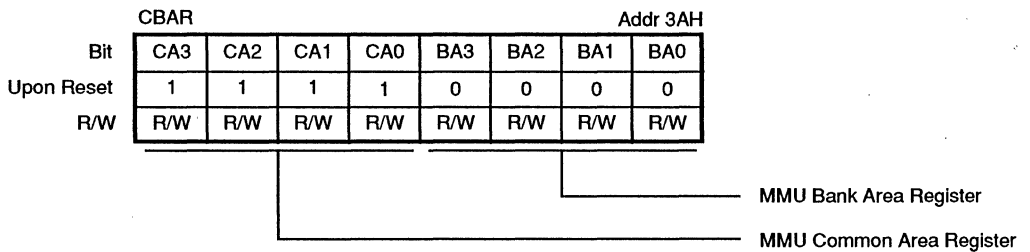


Figure 40. MMU Common/Bank Area Register

## SYSTEM CONTROL REGISTERS

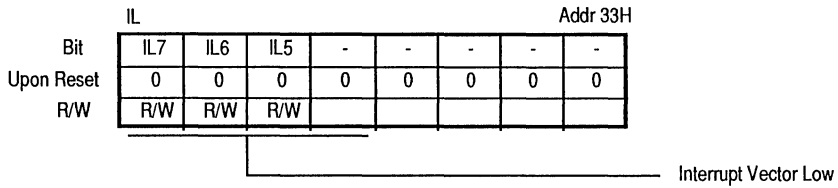


Figure 41. Interrupt Vector Low Register

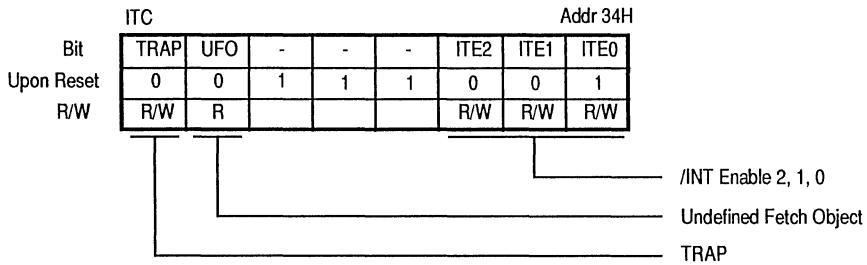
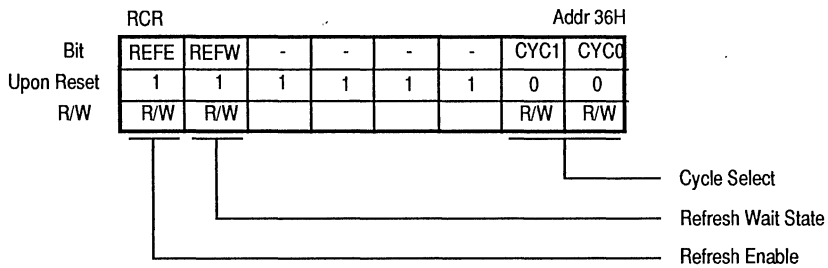


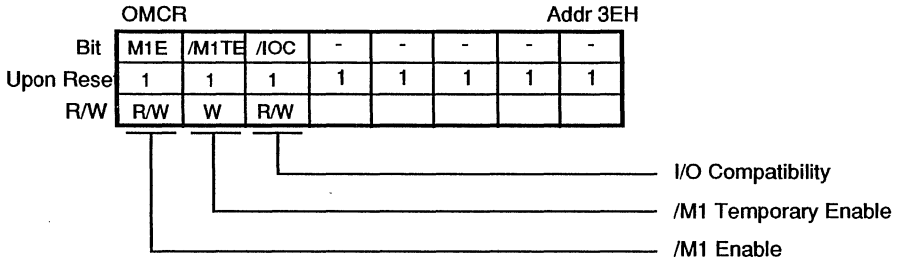
Figure 42. INT/TRAP Control Register



CYC1, 0	Interval of Refresh Cycle
00	10 states
01	20 states
10	40 states
11	80 states

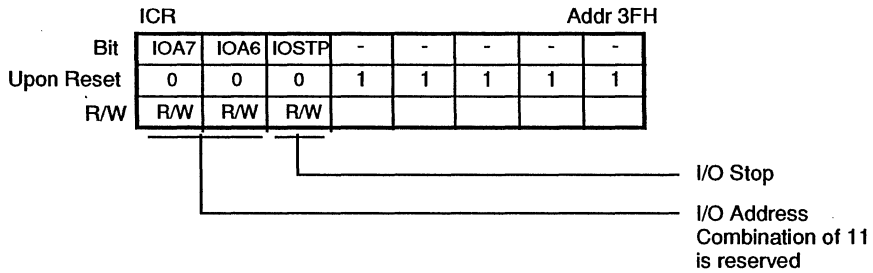
Figure 43. Refresh Control Register

**SYSTEM CONTROL REGISTERS** (Continued)



Note: This register has to be programmed as 0x0xxxxb(x:don't care) as a part of Initialization.

**Figure 44. Operation Mode Control Register**



**Figure 45. I/O Control Register**

## CTC CONTROL REGISTERS

### Channel Control Word

This word sets the operating modes and parameters as described below. Bit D0 must be a 1 to indicate that this is a Control Word (Figure 46).

For more detailed information, refer to the CTC Technical Manual.

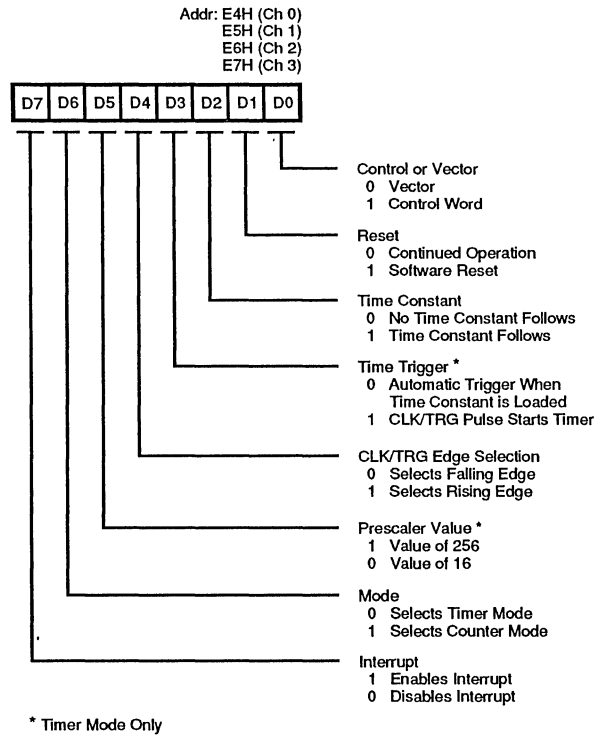


Figure 46. CTC Channel Control Word

This register has the following fields:

**Bit D7. Interrupt Enable.** This bit enables the interrupt logic so that an internal INT is generated at zero count. Interrupts are programmed in either mode and may be enabled or disabled at any time.

**Bit D6. Mode Bit.** This bit selects either Timer Mode or Counter Mode.

**Bit D5. Prescaler Factor.** This bit selects the prescaler factor for use in the timer mode. Either divide-by-16 or divide-by-256 is available.

**Bit D4. Clock/Trigger Edge Selector.** This bit selects the active edge of the CLK/TRG input pulses.

**Bit D3. Timer Trigger.** This bit selects the trigger mode for timer operation. Either automatic or external trigger may be selected.

**Bit D2. Time Constant.** This bit indicates that the next word programmed is time constant data for the downcounter.

**Bit D1. Software Reset.** Writing a 1 to this bit indicates a software reset operation, which stops counting activities until another time constant word is written.

## CTC CONTROL REGISTERS (Continued)

### Time Constant Word

Before a channel can start counting, it must receive a time constant word. The time constant value may be anywhere between 1 and 256, with 0 being accepted as a count of 256 (Figure 47).

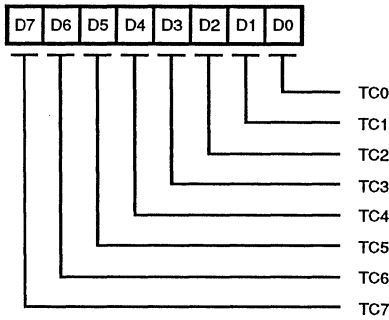


Figure 47. CTC Time Constant Word

### Interrupt Vector Word

If one or more of the CTC channels have interrupt enabled, then the Interrupt Vector Word is programmed. Only the five most significant bits of this word are programmed, and bit D0 must be 0. Bits D2-D1 are automatically modified by the CTC channels after responding with an interrupt vector (Figure 48).

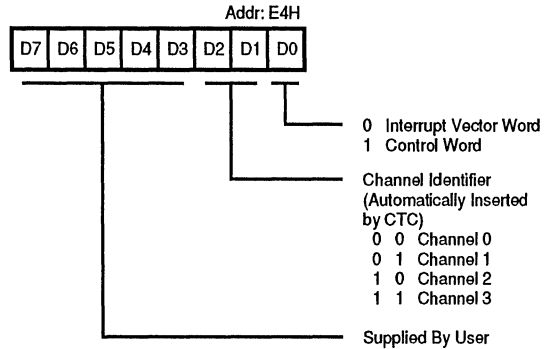


Figure 48. CTC Interrupt Vector Word

## SCC REGISTERS

For more detailed information, please refer to the Z8030/ Z8530 SCC Technical Manual.

Note:

The Address for the Control/Status Register is E8H. The Address for the Data Register is E9H.

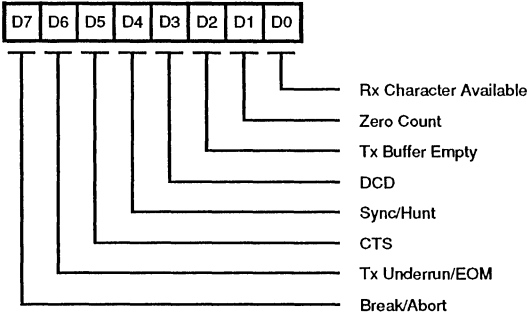
### Read Registers

The SCC contains eight read registers. To read the contents of a register (rather than RR0), the program must first initialize a pointer to WR0 in exactly the same manner as a write operation. The next I/O read cycle will place the contents of the selected read registers onto the data bus (Figure 49).

Table 2. SCC Read Registers

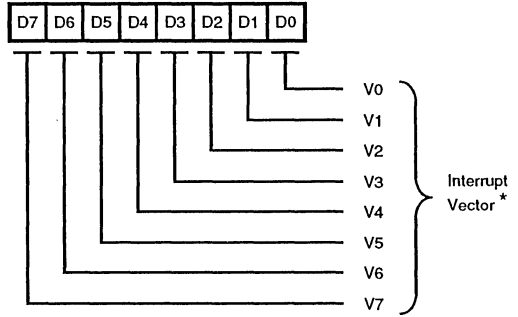
Bit	Description	Bit	Description
RR0	Transmit and Receive buffer status and external status.	RR7	SDLC FIFO byte count and status (only when enabled).
RR1	Special Receive Condition status.	RR8	Receive buffer.
RR2	Interrupt vector (modified if VIS Bit in WR9 is set).	RR10	Miscellaneous status bits.
RR3	Interrupt pending bits.	RR12	Lower byte of baud rate.
RR6	SDLC FIFO byte counter lower byte (only when enabled).	RR13	Upper byte of baud rate generator time constant.
		RR15	External Status interrupt information.

Read Register 0



a)

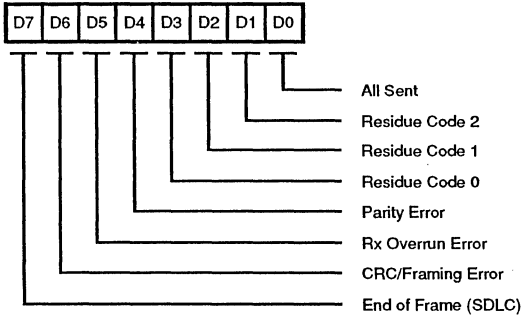
Read Register 2



\* Modified if VIS bit in Write register 9 is set.

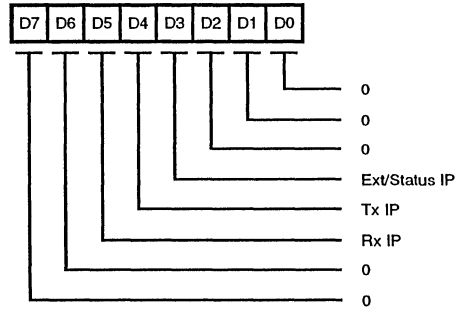
b)

Read Register 1



c)

Read Register 3

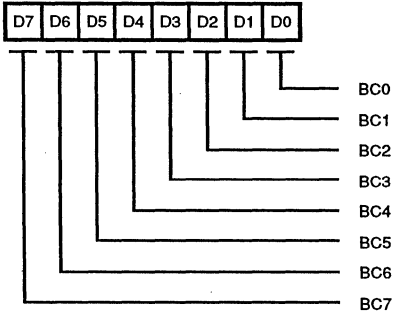


d)

Figure 49. SCC Read Register Bit Functions

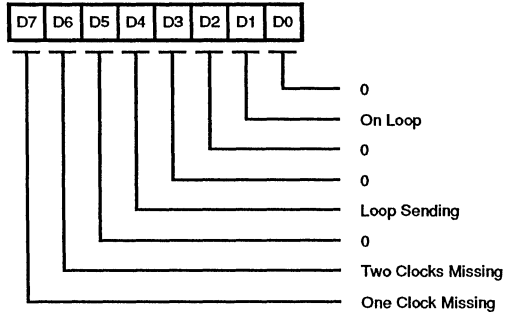
## SCC REGISTERS (Continued)

Read Register 6 \*



\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

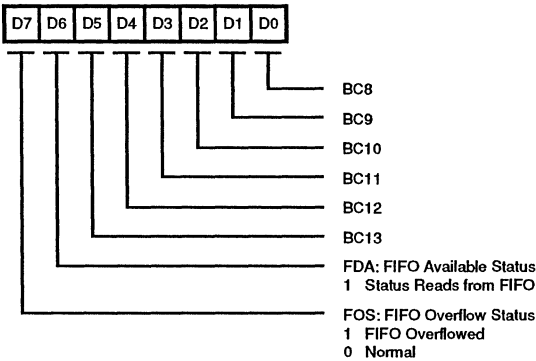
Read Register 10



g)

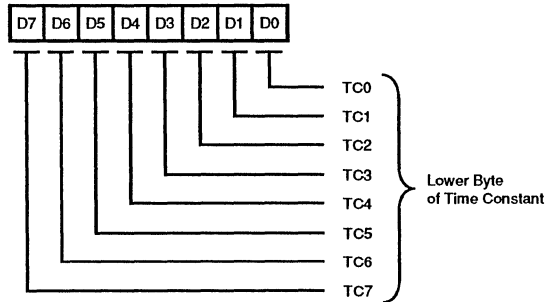
### e) SDLC FIFO Status and Byte Count (LSB)

Read Register 7 \*



\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

Read Register 12



h)

### f) SDLC FIFO Status and Byte Count (MSB)

Figure 49. SCC Read Register Bit Functions (Continued)

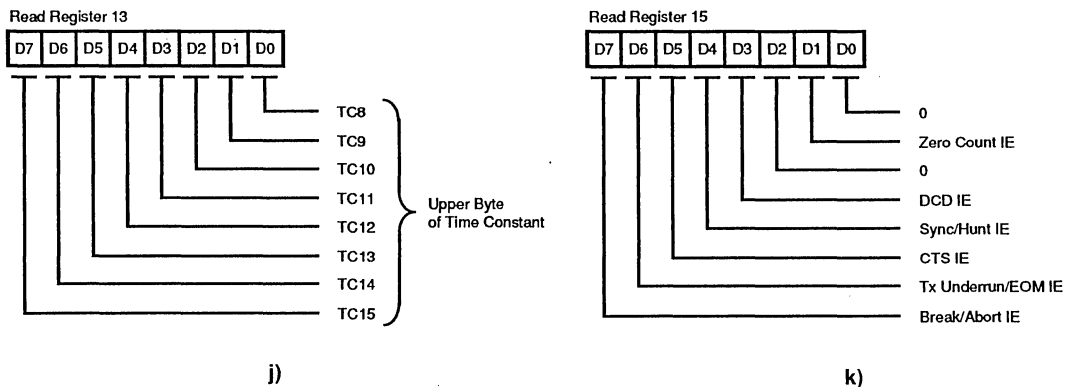


Figure 49. SCC Read Register Bit Functions (Continued)

**Write Registers**

The SCC contains fifteen write registers that are programmed to configure the operating modes of the channel. With the exception of WR0, programming the write registers is a two step operation. The first operation is a

pointer written to WR0 that points to the selected register. The second operation is the actual control word that is written into the register to configure the SCC channel (Figure 50).

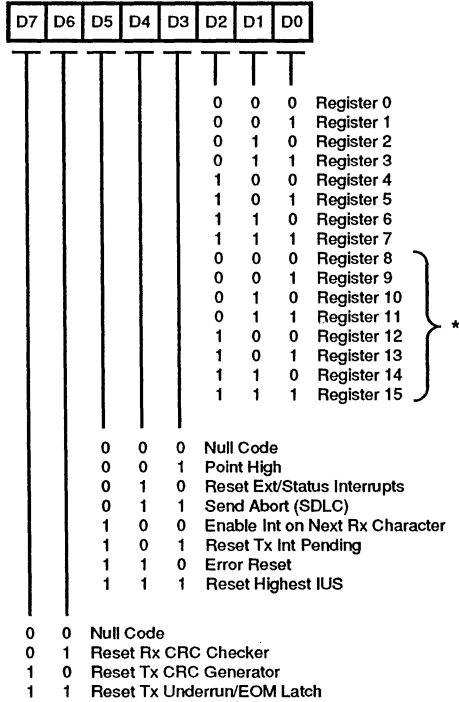
**Table 2. SCC Write Registers**

Bit	Description	Bit	Description
WR0	Register Pointers, various initialization commands	WR8	Transmit buffer
WR1	Transmit and Receive interrupt enables, WAIT/DMA commands	WR9	Master Interrupt control and reset commands
WR2	Interrupt Vector	WR10	Miscellaneous transmit and receive control bits
WR3	Receive parameters and control modes	WR11	Clock mode controls for receive and transmit
WR4	Transmit and Receive modes and parameters	WR12	Lower byte of baud rate generator
WR5	Transmit parameters and control modes	WR13	Upper byte of baud rate generator
WR6	Sync Character or SDLC address	WR14	Miscellaneous control bits
WR7	Sync Character or SDLC flag	WR15	External status interrupt enable control



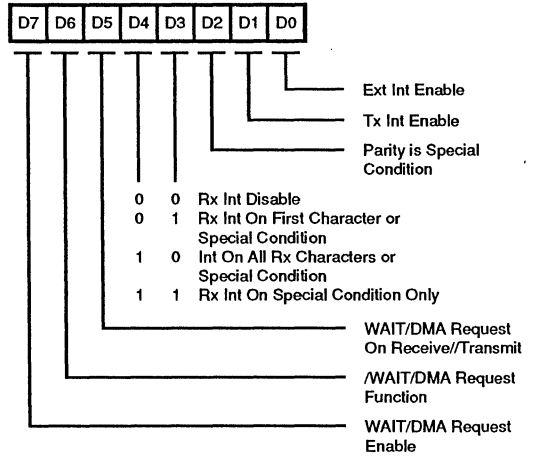
# SCC REGISTERS (Continued)

Write Register 0 (non-multiplexed bus mode)



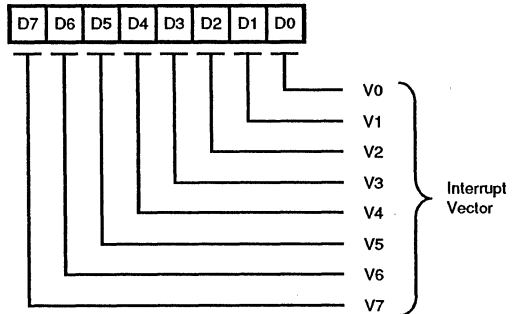
a)

Write Register 1



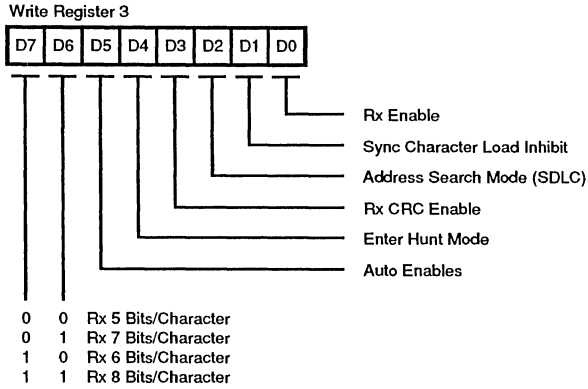
b)

Write Register 2

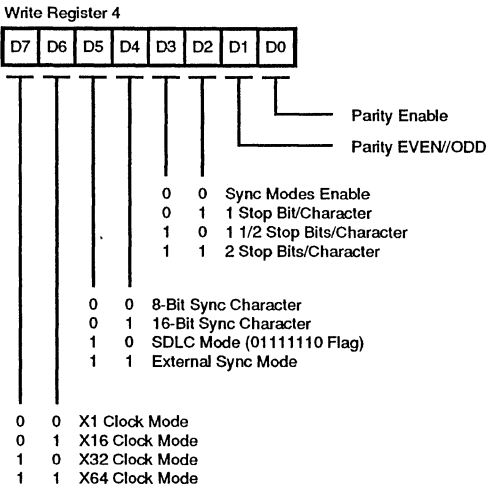


c)

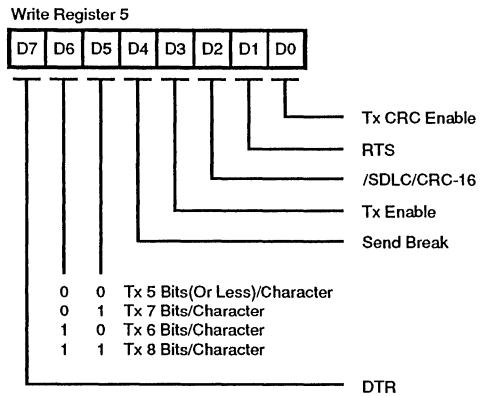
Figure 50. Write Register Bit Functions



d)



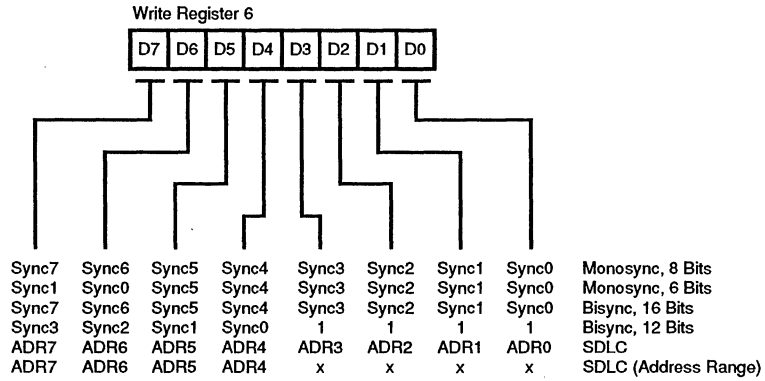
e)



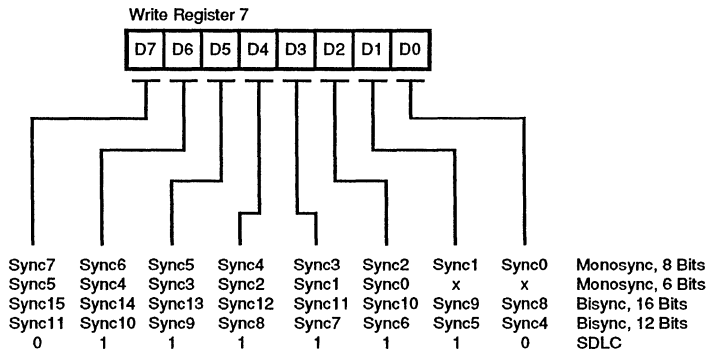
f)

Figure 50. Write Register Bit Functions (Continued)

## SCC REGISTERS (Continued)



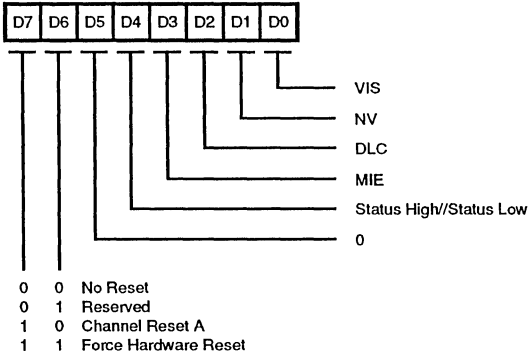
g)



h)

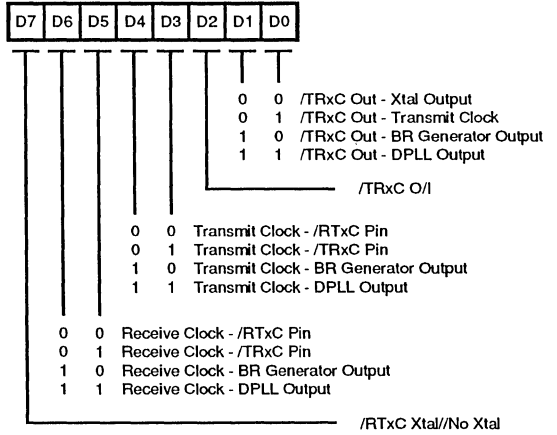
Figure 50. Write Register Bit Functions (Continued)

Write Register 9



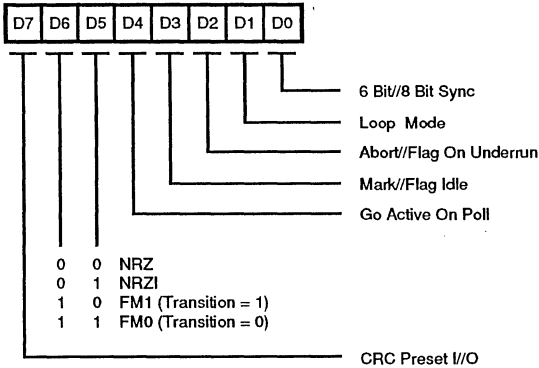
i)

Write Register 11



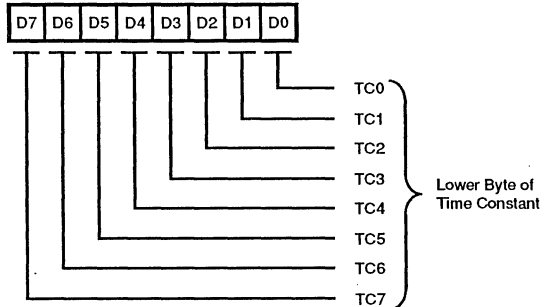
k)

Write Register 10



j)

Write Register 12

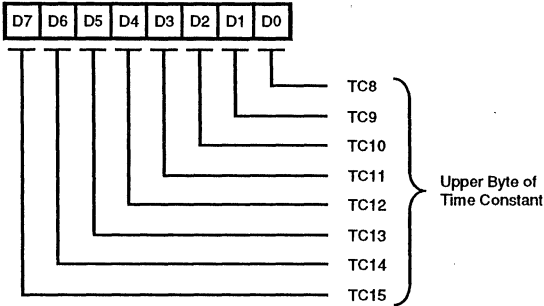


l)

Figure 50. Write Register Bit Functions (Continued)

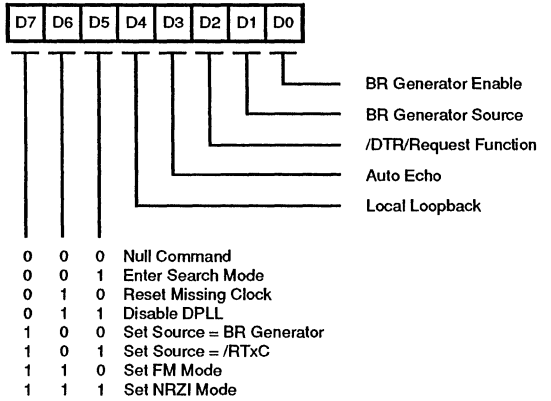
# SCC REGISTERS (Continued)

Write Register 13



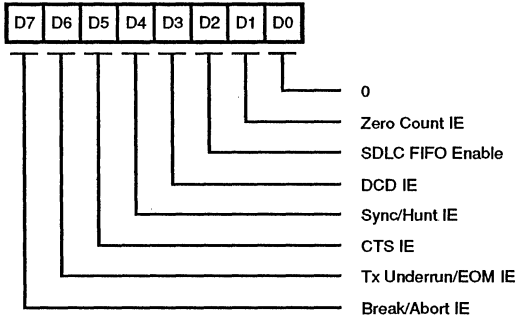
m)

Write Register 14



n)

Write Register 15



o)

Figure 50. Write Register Bit Functions (Continued)

## PIA CONTROL REGISTERS

PIA1 Data Direction Register (P1DDR, I/O Address E0H), PIA1 Data Port (P1DP, I/O address E1H), PIA2 Data Direction Register (P2DDR, I/O Address E2H) and PIA2 Data Register (P2DP, I/O Address E3H). These four regis-

ters are shown in Figures 51-54. Note that if the CTC/PIA bit in the System Configuration Register is set to one, the CTC I/O functions override the PIA1 function, and programming of P1DDR is ignored.

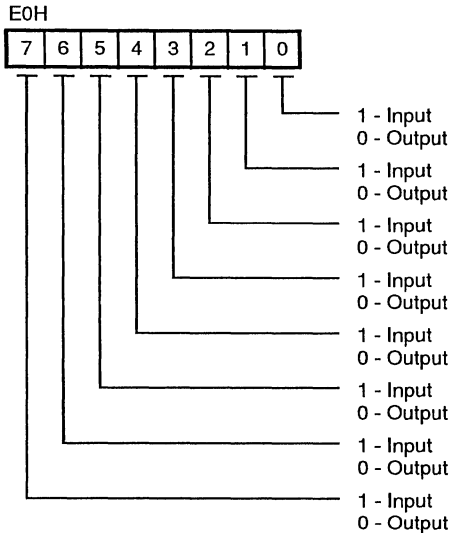


Figure 51. PIA 1 Data Direction Register

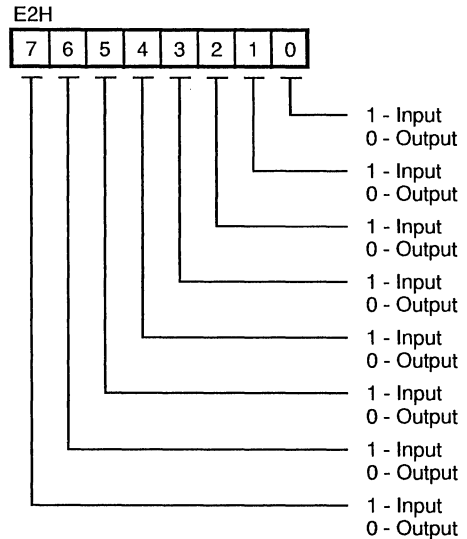


Figure 53. PIA 2 Data Direction Register

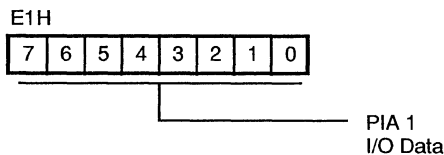


Figure 52. PIA 1 Data Register

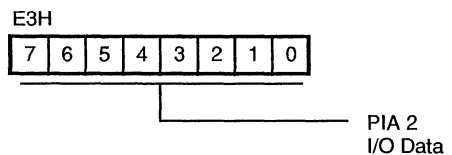


Figure 54. PIA 2 Data Register

The Data Port is the register to/from the 8-bit parallel port. At power on Reset, they are initialized to 1.

reset, these registers are initialized to 1, resulting in all lines being inputs.

The Data Direction Register has eight control bits. Individual bits specify each bit's direction. When the bit is set to a 1, the bit becomes an input, otherwise it is an output. On

## REGISTERS FOR SYSTEM CONFIGURATION

There are four registers to determine system configuration with the Z181. These registers are: RAM upper boundary address register (RAMUBR, I/O address EAH), RAM lower boundary address register (RAMLBR, I/O address EBH), ROM address boundary register (ROMBR, I/O address ECH) and System Configuration Register (SCR, I/O address EDH).

### ROM Address Boundary Register (ROMBR, I/O Address ECH)

This register specifies the address range for the /ROMCS signal. When accessed memory addresses are less than or equal to the value programmed in this register, the /ROMCS signal is asserted (Figure 55).

The A18 signal from the CPU is obtained before it is multiplexed with "TOUT". This signal can be forced to 1 (inactive state) by setting Bit D5 of the System Configuration Register, to allow the user to overlay the RAM area over the ROM area. At power-up reset, this register contains all 1s so that /ROMCS is asserted for all addresses.

### RAM Lower Boundary Address Register (RAMLBR, I/O Address EBH) and RAM Upper Boundary Address Register (RAMUBR, I/O Address EAH)

These two registers specify the address range for the /RAMCS signal. When accessed memory addresses are less than or equal to the value programmed in the RAMUBR and greater than or equal to the value programmed in the RAMLBR, /RAMCS is asserted (Figure 13). The A18 signal

from the CPU is taken before it is multiplexed with TOUT. In the case that these registers are programmed to overlap, /ROMCS takes priority over /RAMCS (/ROMCS is asserted and /RAMCS is inactive).

Chip Select signals are going active for the address range:

/ROMCS: (ROMBR)  $\geq$  A19-A12  $\geq$  0

/RAMCS: (RAMUBR)  $\geq$  A19-A12  $>$  (RAMLBR)

These registers are set to FFH at power-on Reset, and the boundary addresses of ROM and RAM are the following:

ROM lower boundary address (fixed) = 00000H

ROM upper boundary address (ROMBR register) = 0FFFFFFH

RAM lower boundary address (RAMLBR register) = 0FFFFFFH

RAM upper boundary address (RAMUBR register) = 0FFFFFFH

Since /ROMCS takes priority over /RAMCS, the latter will never be asserted until the value in the ROMBR and RAMLBR registers are re-initialized to lower values.

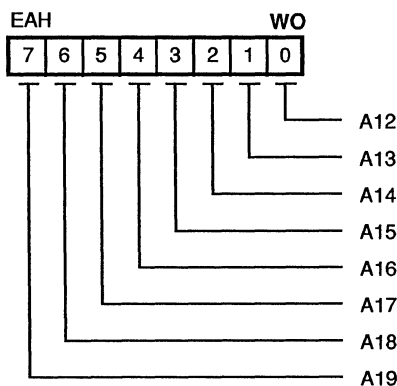


Figure 55. RAM Upper Boundary Register

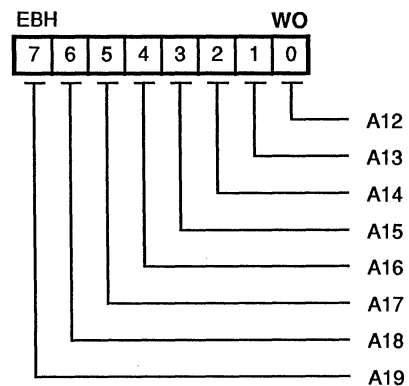


Figure 56. RAM Lower Boundary Register

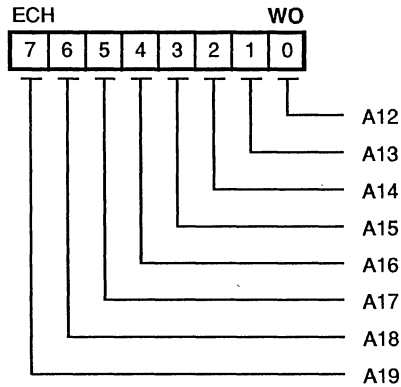


Figure 57. ROM Boundary Register

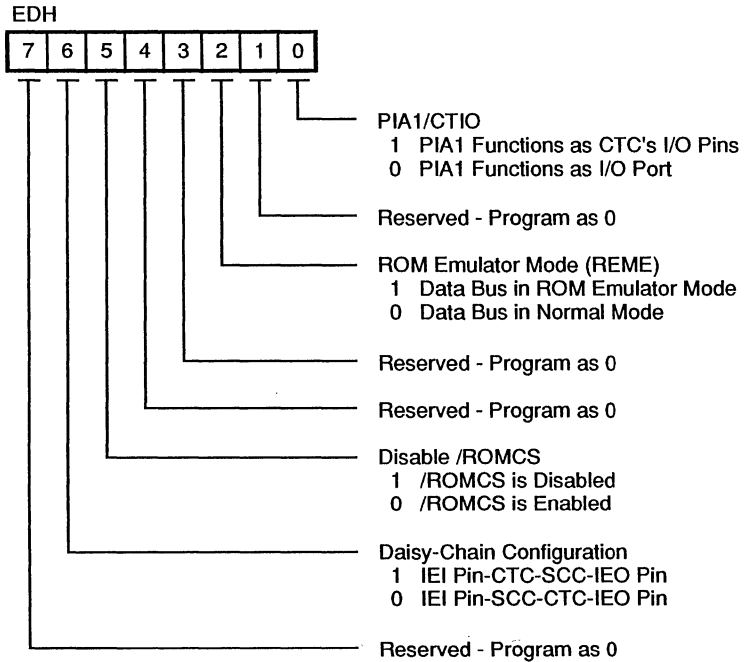


Figure 58. System Configuration Register



---

## REGISTERS FOR SYSTEM CONFIGURATION (Continued)

### System Configuration Register (I/O address EDH)

This register is to determine the functionality of PIA1 and the Interrupt Daisy-Chain Configuration (Figure 13). This register has the following control bits:

**Bit D7.** Reserved and should be programmed as 0.

**Bit D6.** *Daisy-Chain Configuration.* Determines the arrangement of the interrupt priority daisy chain.

When this bit is set to 1, priority is as follows:

IEI pin - CTC - SCC - IEO pin

When this bit is 0, priority is as follows:

IEI pin - SCC - CTC - IEO pin

This bit's default (after Reset) is 0.

**Bit D5.** *Disable /ROMCS.* When this bit is set to 1, /ROMCS is forced to a 1 regardless of the status of the address decode logic. This bit's default (after Reset) is 0 and /ROMCS function is enabled.

**Bit D4-D3.** Reserved and should be programmed as 00.

**Bit D2.** *ROM Emulator Mode Enable.* When this bit is set to a 1, the Z181 is in ROM Emulator Mode. In this mode, bus direction for certain transaction periods are set to the opposite direction to export internal bus transactions outside the Z181. This allows the use of ROM emulators/logic analyzers for applications development. This bit's default (after Reset) is 0.

**Bit D1.** Reserved and shall be programmed as 0.

**Bit D0.** *CTC/PIA1.* When this bit is set to 1, PIA1 functions as the CTC's I/O pins. This bit's default (after Reset) is 0.

## DATA BUS DIRECTION

Table 3 shows the state of the Z181 controller's data bus for the condition that the Z181 is bus master.

**Table 3. Data Bus Direction (Z181 Is Bus Master)**

### I/O And Memory Transactions

	I/O Write To On-Chip Peripherals (SCC/CTC/PIA1/PIA2)	I/O Read From On-Chip Peripherals (SCC/CTC/PIA1/PIA2)	I/O Write To Off-Chip Peripheral	I/O Read From Off-Chip Peripheral	Write To Memory	Read From Memory	Refresh	Z80181 Idle Mode
Z181 Data Bus (REME Bit = 0)	Out	Z	Out	In	Out	In	Z	Z
Z181 Data Bus (REME Bit = 1)	Out	Out	Out	In	Out	In	Z	Z

### Interrupt Acknowledge Transaction

	Intack For On-chip Peripheral (SCC/CTC)	Intack For Off-chip Peripheral	
Z181 Data Bus (REME Bit = 0)	Z	In	
Z181 Data Bus (REME Bit = 1)	Out	In	

## DATA BUS DIRECTION (Continued)

Table 4 shows the state of the Z181 controller's data bus for the condition that the Z181 is NOT bus master.

**Table 4. Data Bus Direction for External Bus Master (Z181 Is Not Bus Master)**

### I/O And Memory Transactions

	I/O Read From On-Chip Peripherals (SCC/CTC/PIA1/PIA2)	I/O Write To On-Chip Peripherals (SCC/CTC/PIA1/PIA2)	I/O Read From Off-Chip Peripheral	I/O To Off-Chip Peripheral	Write From Memory	Read Memory	Refresh	Ext. Bus-Master Is Idle
Z181 Data Bus (REME Bit = 0)	In	Out	Z	Z	Z	In	Z	Z
Z181 Data Bus (REME Bit = 1)	In	Out	Z	Z	Z	In	Z	Z

### Interrupt Acknowledge Transaction

	Intack For On-chip Peripheral (SCC/CTC)	Intack For Off-chip Peripheral	
Z181 Data Bus (REME Bit = 0)	Out	In	
Z181 Data Bus (REME Bit = 1)	Out	In	

The word OUT means that the Z181 data bus direction is in output mode, IN means input mode, and HI-Z means high impedance.

REME stands for ROM EmulatorMode and is the status of D2 bit in the System Configuration Register.

---

## ABSOLUTE MAXIMUM RATINGS

Voltage on Vcc with respect to Vss ..... -0.3V to +7.0V  
Voltages on all inputs  
with respect to Vss ..... -0.3V to Vcc+0.3V

### Operating Ambient

Temperature ..... See Ordering Information  
Storage Temperature ..... -65°C to + 150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

---

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

Available operating temperature range is:  
E = -40°C to 100°C

Voltage Supply Range:  
 $+4.50V \leq V_{cc} \leq +5.50V$

All AC parameters assume a load capacitance of 100 pF. Add 10 ns delay for each 50 pF increase in load up to a maximum of 150 pF for the data bus and 100 pF for address and control lines. AC timing measurements are referenced to 1.5 volts (except for clock, which is referenced to the 10% and 90% points). Maximum capacitive load for CLK is 125 pF.

The Ordering Information section lists temperature ranges and product numbers. Refer to the Literature List for additional documentation.

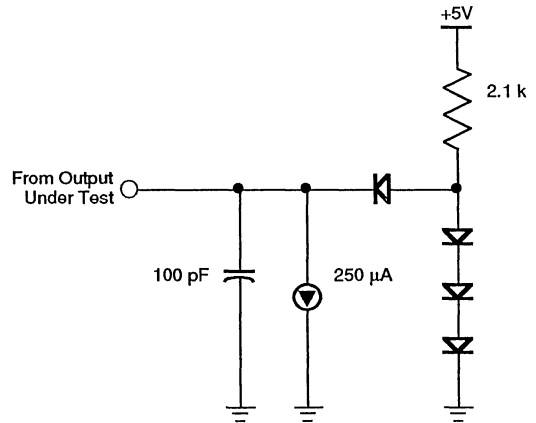


Figure 59. Standard Test Circuit

## DC CHARACTERISTICS

Z80181

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH1}$	Input H Voltage /RESET, EXTAL, /NMI	$V_{CC} - 0.6$		$V_{CC} + 0.3$	V	
$V_{IH2}$	Input H Voltage Except /RESET, EXTAL, /NMI	2.0		$V_{CC} + 0.3$	V	
$V_{IL1}$	Input L Voltage /RESET, EXTAL, /NMI	-0.3		0.6	V	
$V_{IL2}$	Input L Voltage Except /RESET, EXTAL, /NMI	-0.3		0.8	V	
$V_{OH}$	Output H Voltage All outputs.	2.4			V	$I_{OH} = -200 \mu A$
$V_{OL}$	Output L Voltage All outputs.	$V_{CC} - 1.2$		0.45	V	$I_{OH} = -20 \mu A$ $I_{OL} = 2.2 mA$
$I_{IL}$	Input Leakage Current All Inputs Except XTAL, EXTAL			10	$\mu A$	$V_{IN} = 0.5 - V_{CC} - 0.5$
$I_{TL}$	Tri-State Leakage Current			10	$\mu A$	$V_{IN} = 0.5 - V_{CC} - 0.5$
$I_{CC}^*$	Power Dissipation* (Normal Operation)		55	100	mA	$f = 12.5 MHz$
	Power Dissipation* (SYSTEM STOP mode)		30	80		$f = 10 MHz$
				50		$f = 12.5 MHz$
				40		$f = 10 MHz$
$C_p$	Pin Capacitance			12	pF	$V_{IN} = 0V, f = 1 MHz$ $T_A = 25^\circ C$

\*  $V_{IH}$  Min =  $V_{CC} - 1.0V$ ,  $V_{IL}$  Max =  $0.8V$  (all output terminals are at no load.)  
 $V_{CC} = 5.00V$

# AC CHARACTERISTICS

## Z180 MPU Timing

Figures 60-68 show the timing for the Z181 MPU and the referenced parameters appear in Table A.

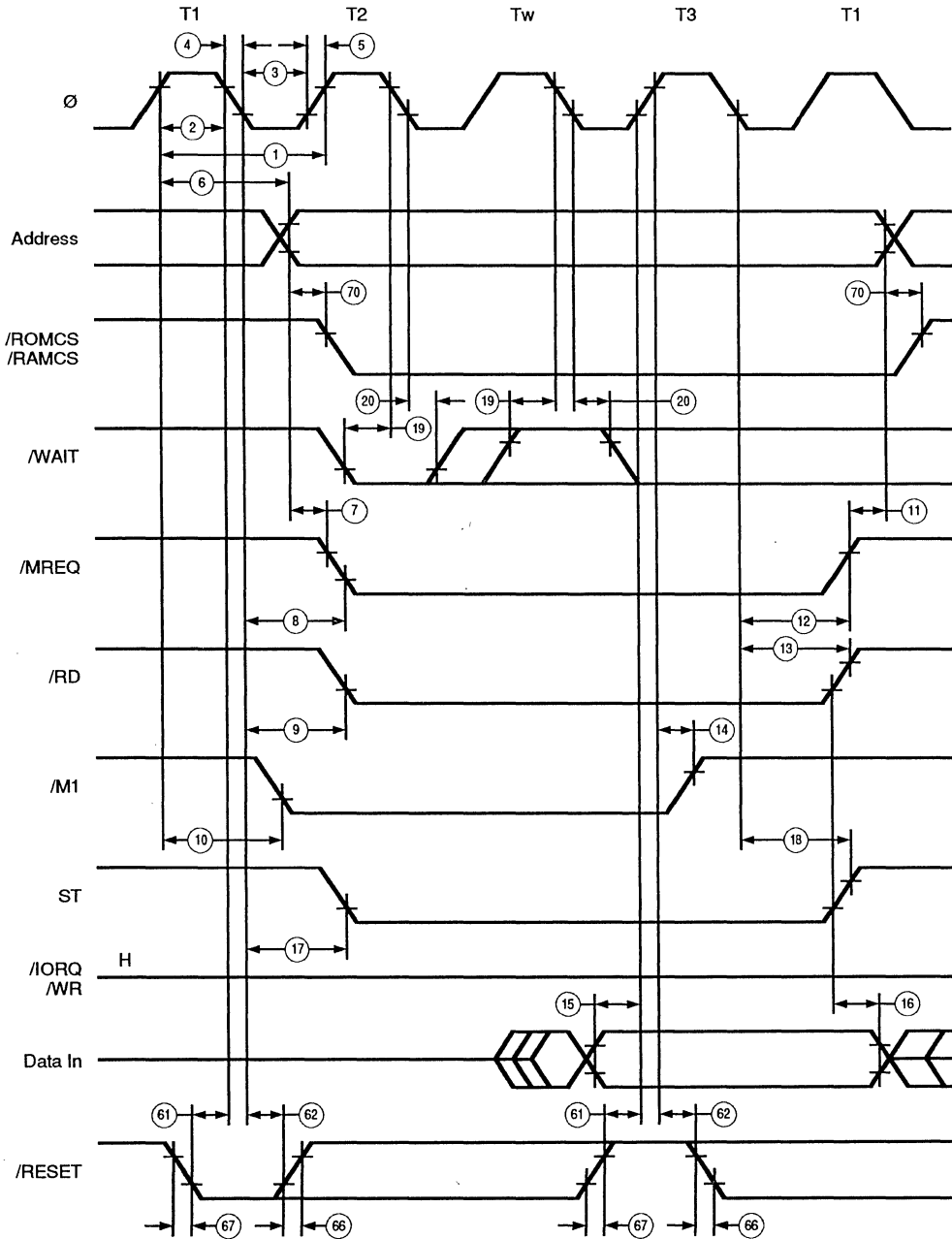
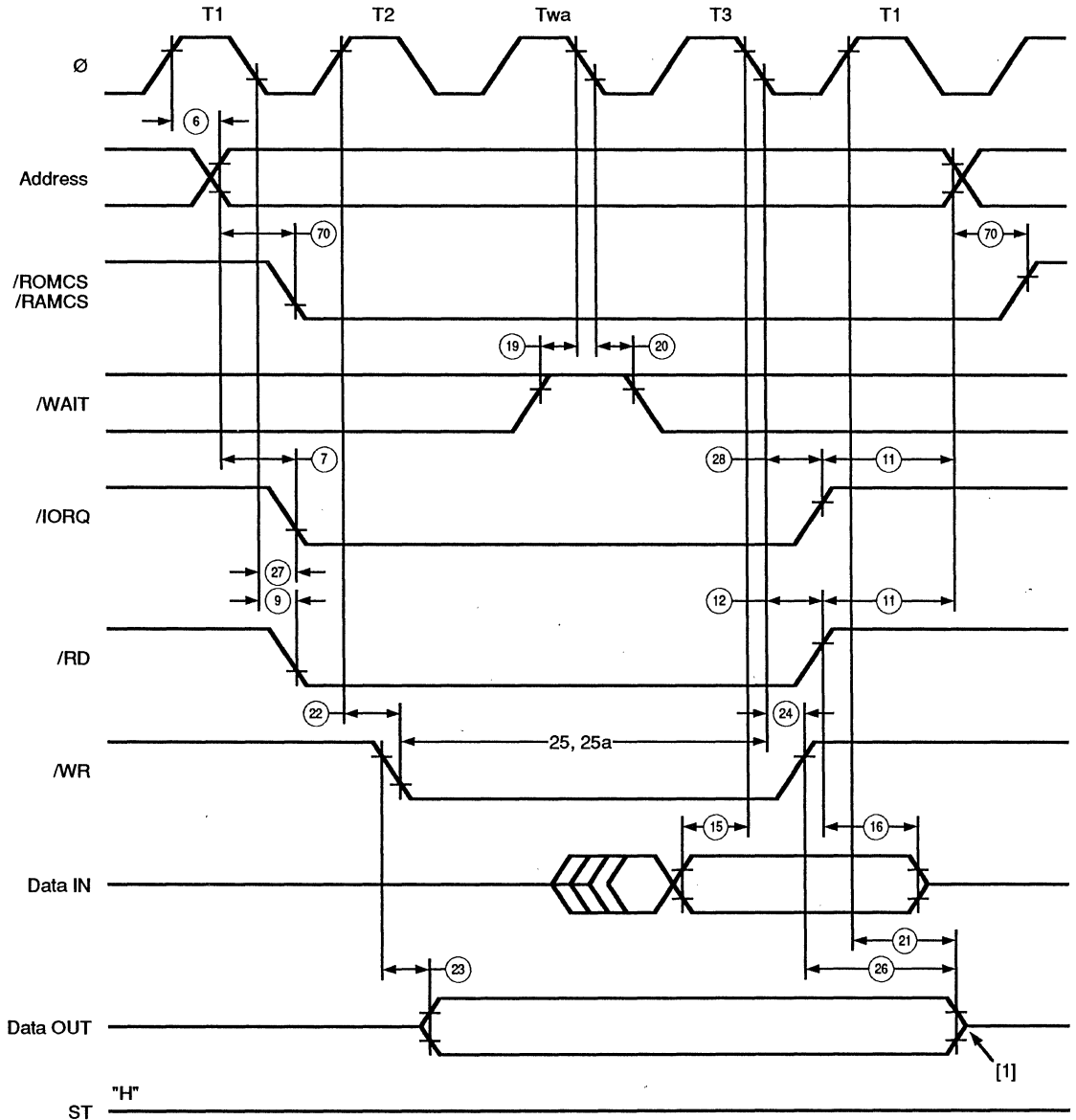


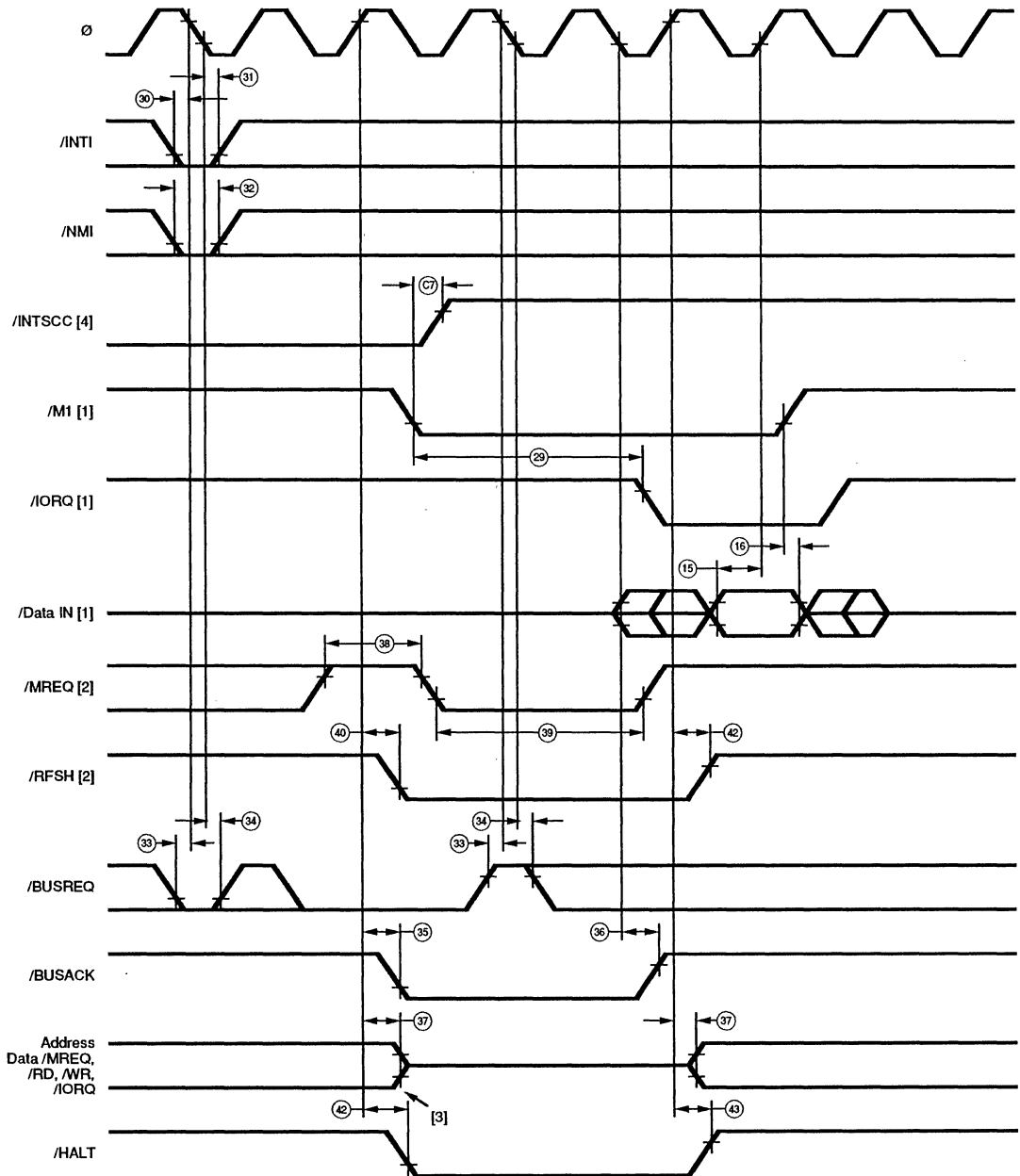
Figure 60a. Op-code Fetch Cycle

**AC CHARACTERISTICS** (Continued)  
Z180 MPU Timing



- [1] Output buffer is off at this point.
- [2] Memory Read/Write cycle timing is the same as this figure, except there is no automatic wait status (Twa), and /MREQ is active instead of /IORQ.

Figure 60b. I/O Read/Write, Memory Read/Write Timing



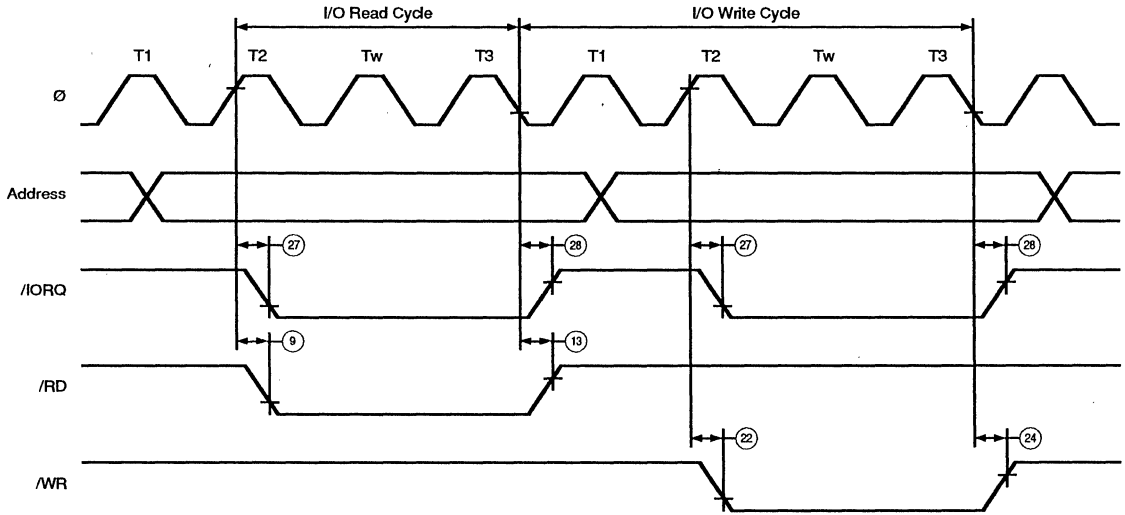
- Notes:
- [1] During /INTO acknowledge cycle
  - [2] During refresh cycle
  - [3] Output buffer is off at this point
  - [4] Refer to Table C, parameter 7

**Figure 61. CPU Timing**

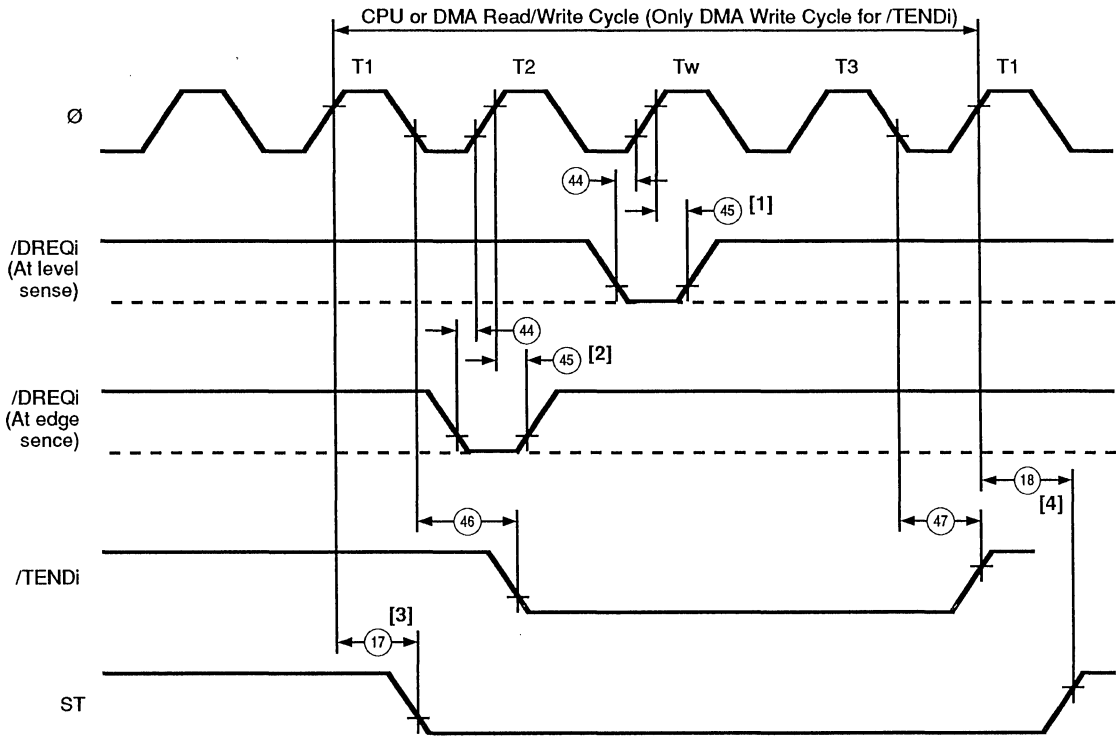
(/INTO Acknowledge Cycle, Refresh Cycle, BUS RELEASE Mode  
HALT Mode, SLEEP Mode, SYSTEM STOP Mode)



**AC CHARACTERISTICS** (Continued)  
 Z180 MPU Timing



**Figure 62. CPU Timing** (/IOC = 0)  
 (I/O Read Cycle, I/O Write Cycle)

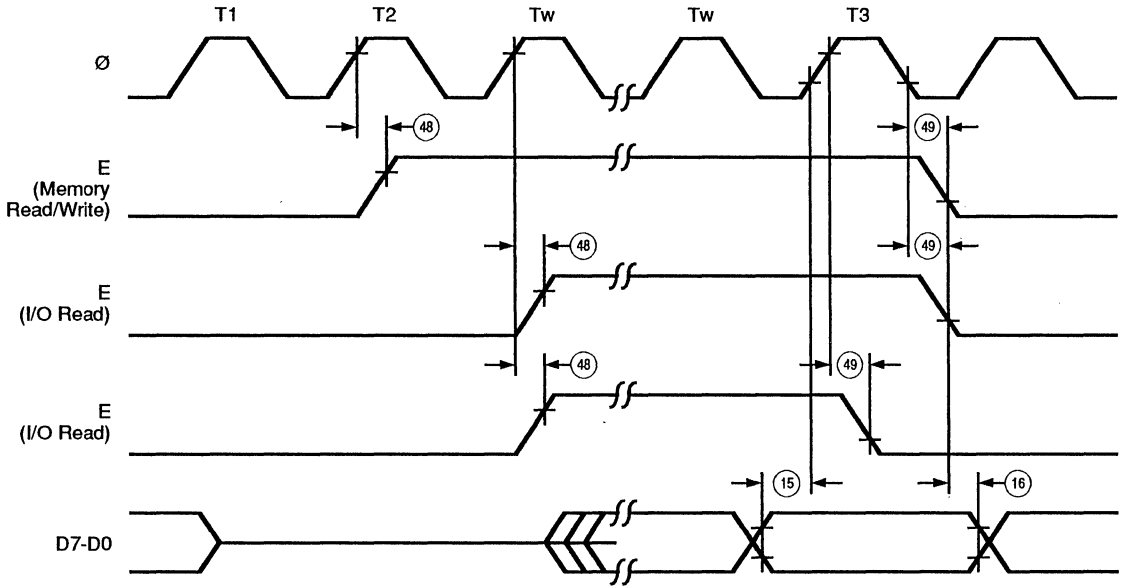


**DMA Control Signals**

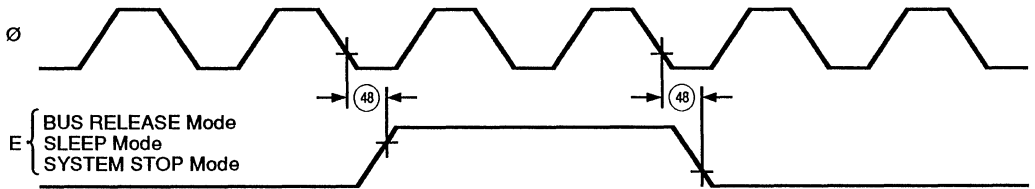
- [1] /DREQS and /DREQH are specified for the rising edge of clock followed by T3.
- [2] /DREQS and /DREQH are specified for the rising edge of clock.
- [3] DMA cycle starts.
- [4] CPU cycle starts.

**Figure 63. DMA Control Signals**

**AC CHARACTERISTICS** (Continued)  
Z180 MPU Timing

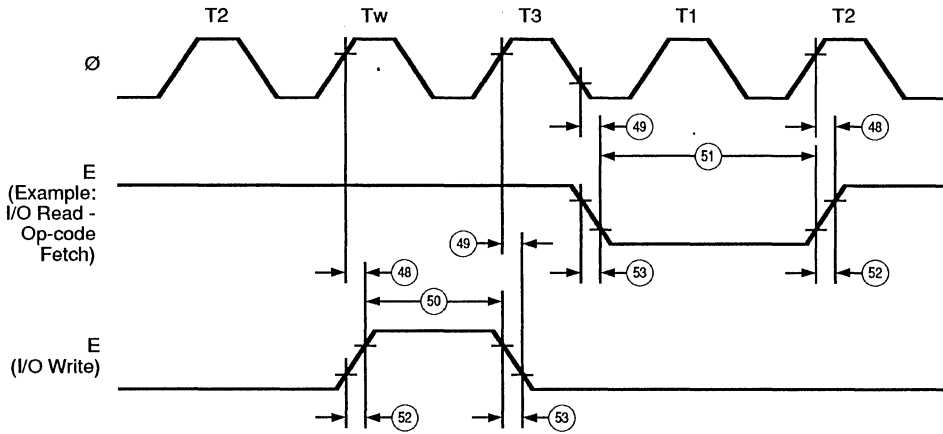


**a) E Clock Timing**  
(Memory Read/Write Cycle, I/O Read/Write Cycle)

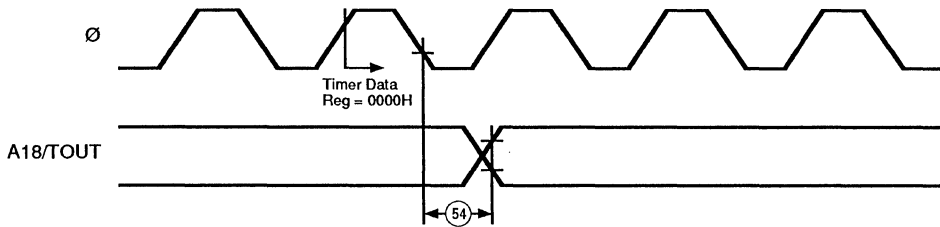


**b) E Clock Timing**  
(BUS RELEASE Mode, SLEEP Mode, SYSTEM STOP Mode)

**Figure 64. E Clock Timing**



**Figure 65. E Clock Timing**  
 (Minimum timing example of PWEL and PWEH)



**Figure 66. Timer Output Timing**

**AC CHARACTERISTICS** (Continued)  
 Z180 MPU Timing

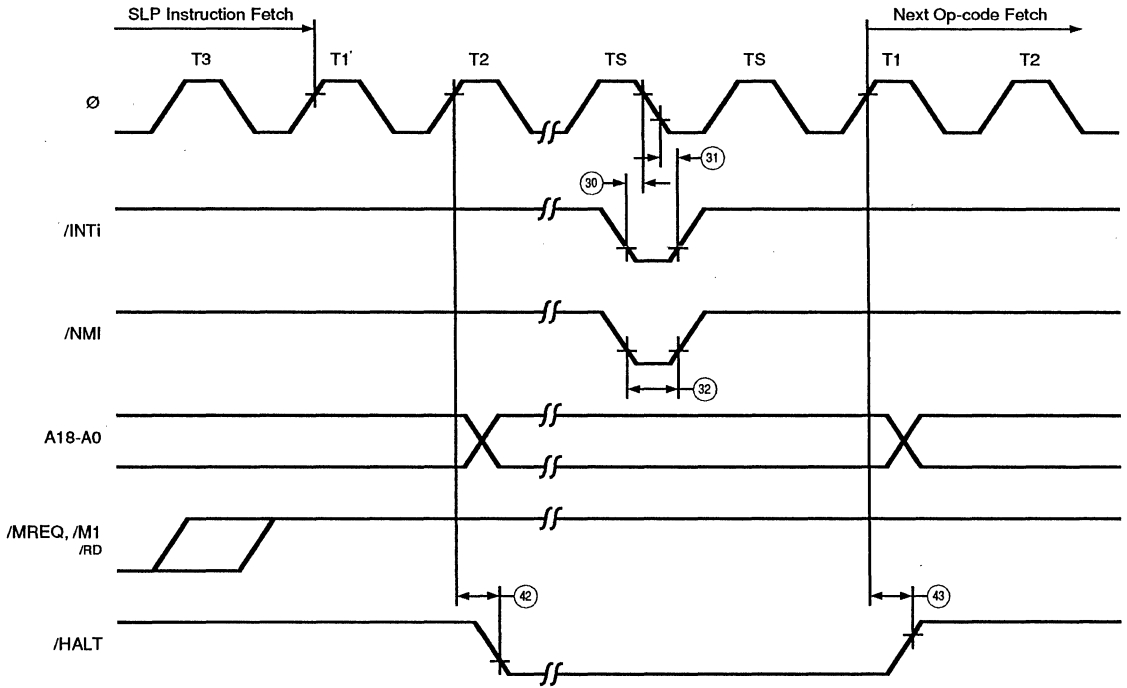


Figure 67. SLP Execution Cycle

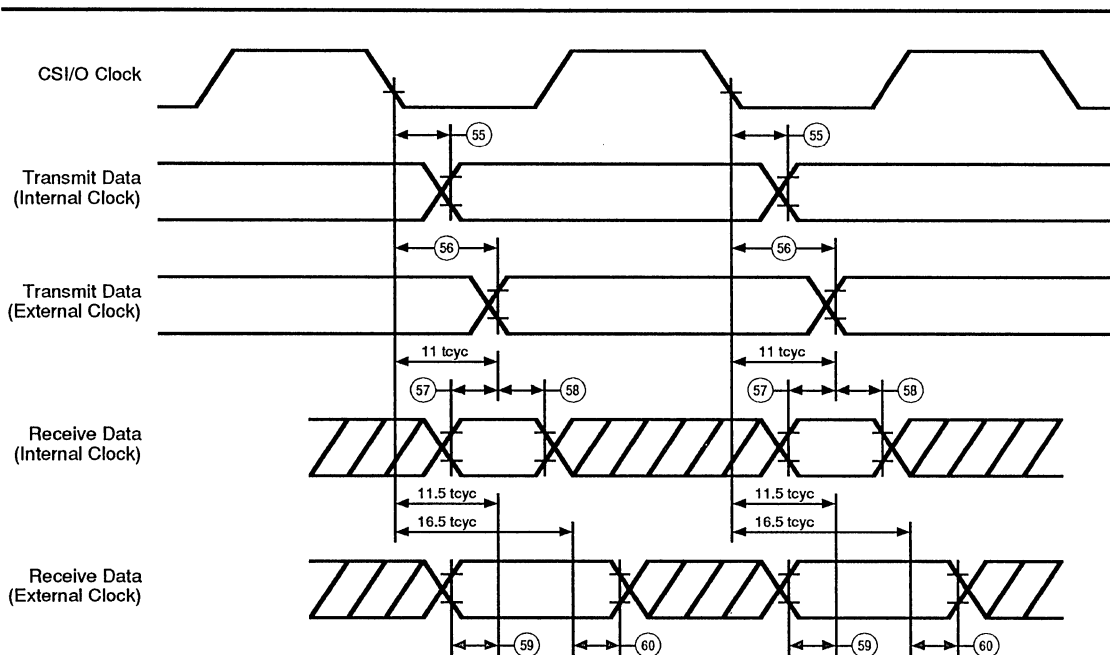


Figure 68. CSI/O Receive/Transmit Timing

Table A. Z180 CPU & 180 Peripherals Timing

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	t <sub>cy</sub>	Clock Cycle Time	100	2000	80	2000	ns	
2	t <sub>CHW</sub>	Clock Pulse Width (High)	40		30		ns	
3	t <sub>CLW</sub>	Clock Pulse Width (Low)	40		30		ns	
4	t <sub>cf</sub>	Clock Fall Time		10		10	ns	
5	t <sub>cr</sub>	Clock Rise Time		10		10	ns	
6	t <sub>AD</sub>	Address Valid from Clock Rise		70		60	ns	
7	t <sub>AS</sub>	Address Valid to /MREQ, /IORQ Fall	10		10		ns	
8	t <sub>MED1</sub>	Clock Fall to /MREQ Fall Delay		50		45	ns	
9	t <sub>RDD1</sub>	Clock Fall to /RD Fall (/IOC=1)		50		45	ns	
		Clock Rise to /RD Fall (/IOC=0)		55		50	ns	
10	t <sub>M1D1</sub>	Clock Rise to /M1 Fall Delay		60		50	ns	

**AC CHARACTERISTICS** (Continued)  
Z180 MPU Timing

**Table A. Z180 CPU & 180 Peripherals Timing** (Continued)

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
11	tAH	Address Hold Time (/MREQ, /IORQ, /RD, /WR)	10		10		ns	
12	tMED2	Clock Fall to /MREQ Rise Delay		50		45	ns	
13	tRDD2	Clock Fall to /RD Rise Delay		50		45	ns	
14	tM1D2	Clock Rise to /M1 Rise Delay		60		50	ns	
15	tDRS	Data Read Setup Time	45		45		ns	
16	tDRH	Data Read Hold Time	0		0		ns	
17	tSTD1	Clock Fall to ST Fall		60		50	ns	
18	tSTD2	Clock Fall to ST Rise		60		50	ns	
19	tWS	/WAIT Setup Time to Clock Fall	30		25		ns	
20	tWH	/WAIT Hold time from Clock Fall	30		20		ns	
21	tWDZ	Clock Rise to Data Float Delay		60		60	ns	
22	tWRD1	Clock Rise to /WR Fall Delay		50		30	ns	
23	tWDO	/WR fall to Data Out Delay		15		15	ns	
24	tWRD2	Clock Fall to /WR Rise		50		45	ns	
25	tWRP	/WR Pulse Width (Memory Write Cycles)	110		85		ns	
25a		/WR Pulse Width (I/O Write Cycles)	210		165		ns	
26	tWDH	Write Data Hold Time from /WR Rise	7.5		7.5		ns	
27	tIOD1	Clock Fall to /IORQ Fall Delay (/IOC=1)		50		45	ns	
		Clock Rise to /IORQ Fall Delay (/IOC=0)		55		50	ns	
28	tIOD2	Clock Fall to /IORQ Rise Delay		50		50	ns	
29	tIOD3	/M1 Fall to /IORQ Fall Delay	200		160		ns	
30	tINTS	/INT Setup Time to Clock Fall	30		25		ns	
31	tINTH	/INT Hold Time from Clock Fall	30		20		ns	
32	tNMIW	/NMI Pulse Width	80		60		ns	
33	tBRS	/BUSREQ Setup Time to Clock Fall	30		25		ns	
34	tBRH	/BUSREQ Hold Time from Clock Fall	30		20		ns	
35	tBAD1	Clock Rise to /BUSACK Fall Delay		60		50	ns	
36	tBAD2	Clock Fall to /BUSACK Rise Delay		60		50	ns	
37	tBZD	Clock Rise to Bus Floating Delay Time		80		60	ns	
38	tMEWH	/MREQ Pulse Width (High)	70		60		ns	
39	tMEWL	/MREQ Pulse Width (LOW)	80		60		ns	
40	tRFD1	Clock Rise to /RFSH Fall Delay		60		40	ns	
41	tRFD2	Clock Rise to /RFSH Rise Delay		60		40	ns	
42	tHAD1	Clock Rise to /HALT Fall Delay		50		30	ns	
43	tHAD2	Clock Rise to /HALT Rise Delay		50		30	ns	
44	tDRQS	/DREQi Setup Time to Clock Rise	30		20		ns	
45	tDRQH	/DREQi Hold Time from Clock Rise	30		20		ns	
46	tTED1	Clock Fall to /TENDi Fall Delay		50		50	ns	

**AC CHARACTERISTICS** (Continued)  
Z180 MPU Timing

**Table A. Z180 CPU & 180 Peripherals Timing** (Continued)

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
47	tTED2	Clock Fall to /TENDi Rise Delay		50		50	ns	
48	tED1	Clock Rise to E Rise Delay		60		40	ns	
49	tED2	Clock Edge to E Fall Delay		60		40	ns	
50	PWEH	E Pulse Width (High)	55		45		ns	
51	PWEL	E Pulse Width (Low)	110		90		ns	
52	tEr	Enable Rise Time		20		10	ns	
53	tEf	Enable Fall Time		20		10	ns	
54	tTOD	Clock Fall to Timer Output Delay		150		120	ns	
55	tSTDI	CSI/O Tx Data Delay Time (Internal Clock Operation)		150		120	ns	
56	tSTDE	CSI/O Tx Data Delay Time (External Clock Operation)		7.5tcyc+150		7.5tcyc+120	ns	
57	tSRSI	CSI/O Rx Data Setup Time (Internal Clock Operation)	1		1		tcyc	
58	tSRHI	CSI/O Rx Data Hold Time (Internal Clock Operation)	1		1		tcyc	
59	tSRSE	CSI/O Rx Data Setup Time (External Clock Operation)	1		1		tcyc	
60	tSRHE	CSI/O Rx Data Hold Time (External Clock Operation)	1		1		tcyc	
61	tRES	/RESET Setup Time to Clock Fall	80		60		ns	
62	tREH	/RESET Hold Time from Clock Fall	50		45		ns	
63	tOSC	Oscillator Stabilization Time		20		20	ms	
64	tEXr	External Clock Rise Time (EXTAL)		25		20	ns	
65	tEXf	External Clock Fall Time (EXTAL)		25		20	ns	
66	tRr	/RESET Rise Time		50		50	ns	
67	tRf	/RESET Fall Time		50		50	ns	
68	tIr	Input Rise Time (Except EXTAL, /RESET)		100		80	ns	
69	tIf	Input Fall Time (Except EXTAL, /RESET)		100		80	ns	
70	TdCS(A)	Address Valid to /ROMCS, /RAMCS Valid Delay		20		20	ns	



## AC CHARACTERISTICS (Continued)

### CTC Timing

Figure 69 shows the timing for the on-chip CTC. Parameters referred to in this figure appear in Table B.

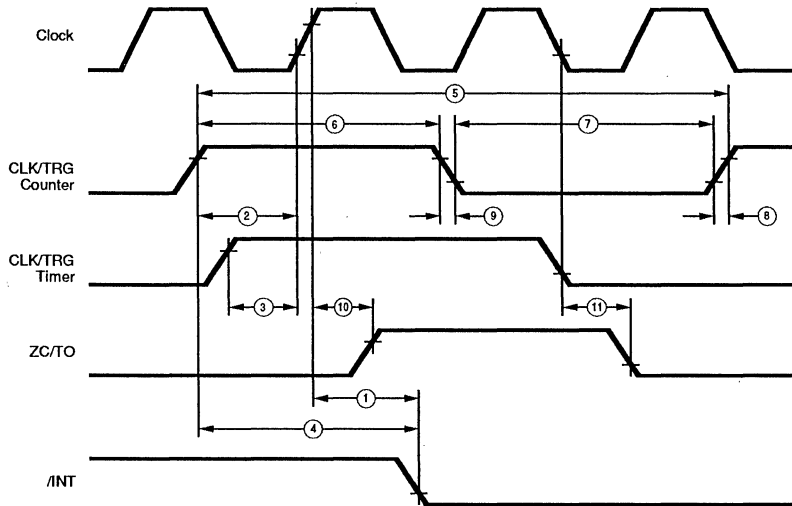


Figure 69. CTC Timing

Table B. CTC Timing Parameters

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	TdCr(/INTf)	Clock Rise to /INT Fall Delay		(TcC+100)		(TcC+80)	ns	[B1]
2	TsCTR(Cr)c	CLK/TRG to Clock Rise Setup Time for Immediate Count	90		60		ns	[B2]
3	TsCTR(Ct)	CLK/TRG Rise to Clock Rise Setup Time for Enabling of Prescaler On Following Clock Rise	90		60		ns	[B1]
4	TdCTR(INTf)	CLK/TRG to /INT Fall Delay TsCTR(C) Satisfied TsCTR(C) Not Satisfied		(1)+(3) TcC+(1)+(3)		(1)+(3) TcC+(1)+(3)	ns	[B2] [B2]
5	TcCTR	CLK/TRG Cycle Time	(2TcC)	DC	(2TcC)	DC	ns	[B3]
6	TwCTRh	CLK/TRG Width (Low)	90	DC	90	DC	ns	
7	TwCTRI	CLK/TRG Width (High)	90	DC	90	DC	ns	
8	TrCTR	CLK/TRG Rise Time		30		30	ns	
9	TfCTR	CLK/TRG Fall Time		30		30	ns	
10	TdCr(ZCr)	Clock Rise to ZC/TO Rise Delay		80		80	ns	
11	TdCl(ZCf)	Clock Fall to ZC/TO Fall Delay		80		80	ns	

#### Notes for Table B:

[B1] Timer Mode

[B2] Counter Mode

[B3] Counter Mode Only; When using a cycle time less than 3TcC, parameter #2 must be met.

## AC CHARACTERISTICS (Continued)

### SCC Timing

Figure 70 shows the AC characteristics for the on-chip SCC. Parameters referred to in this figure appear in Table C.

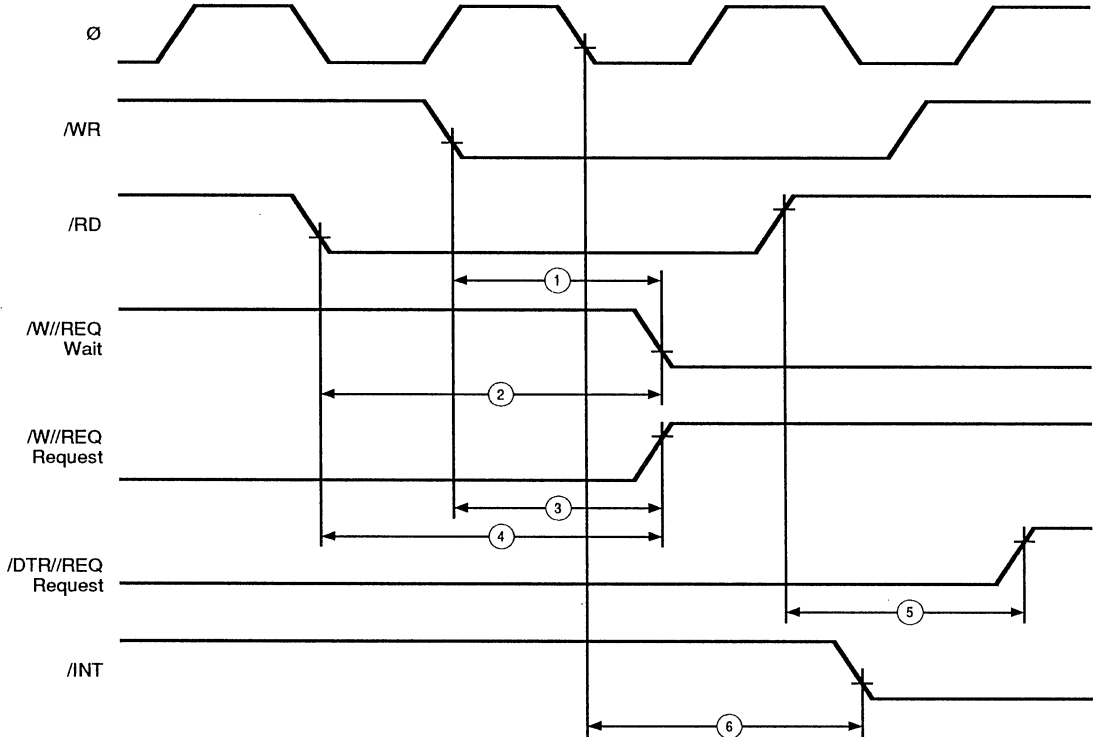


Figure 70. SCC AC Parameters

Table C. SCC Timing Parameters (85C30 AC Characteristics)

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	TdWR(W)	/WR Fall to Wait Valid Delay		180 + TcC		125 + TcC	ns	[C1]
2	TdWR(W)	/RD Fall to Wait Valid Delay		180		125	ns	[C1]
3	TdWRf(REQ)	/WR Fall to /W//REQ Not Valid Delay		180 + TcC		125 + TcC	ns	
4	TdRDf(REQ)	/RD Fall to /W//REQ Not Valid Delay		180		125	ns	
5	TdWRr(REQ)	/WR Rise to /DTR//REQ Not Valid Delay		5TcC		5TcC	ns	
6	TdPC(INT)	Clock to /INT Valid Delay		500		500	ns	[C1]
7	TdRDA(INT)	/M1 Fall to /INT Inactive Delay		TBS		TBS	ns	[C1]

**Note for Table C:**

[C1] Open-drain output, measured with Open-drain test load.

**AC CHARACTERISTICS** (Continued)  
 SCC General Timing

Figure 71 shows the general timing for the on-chip SCC. Parameters referred to in this figure appear in Table D.

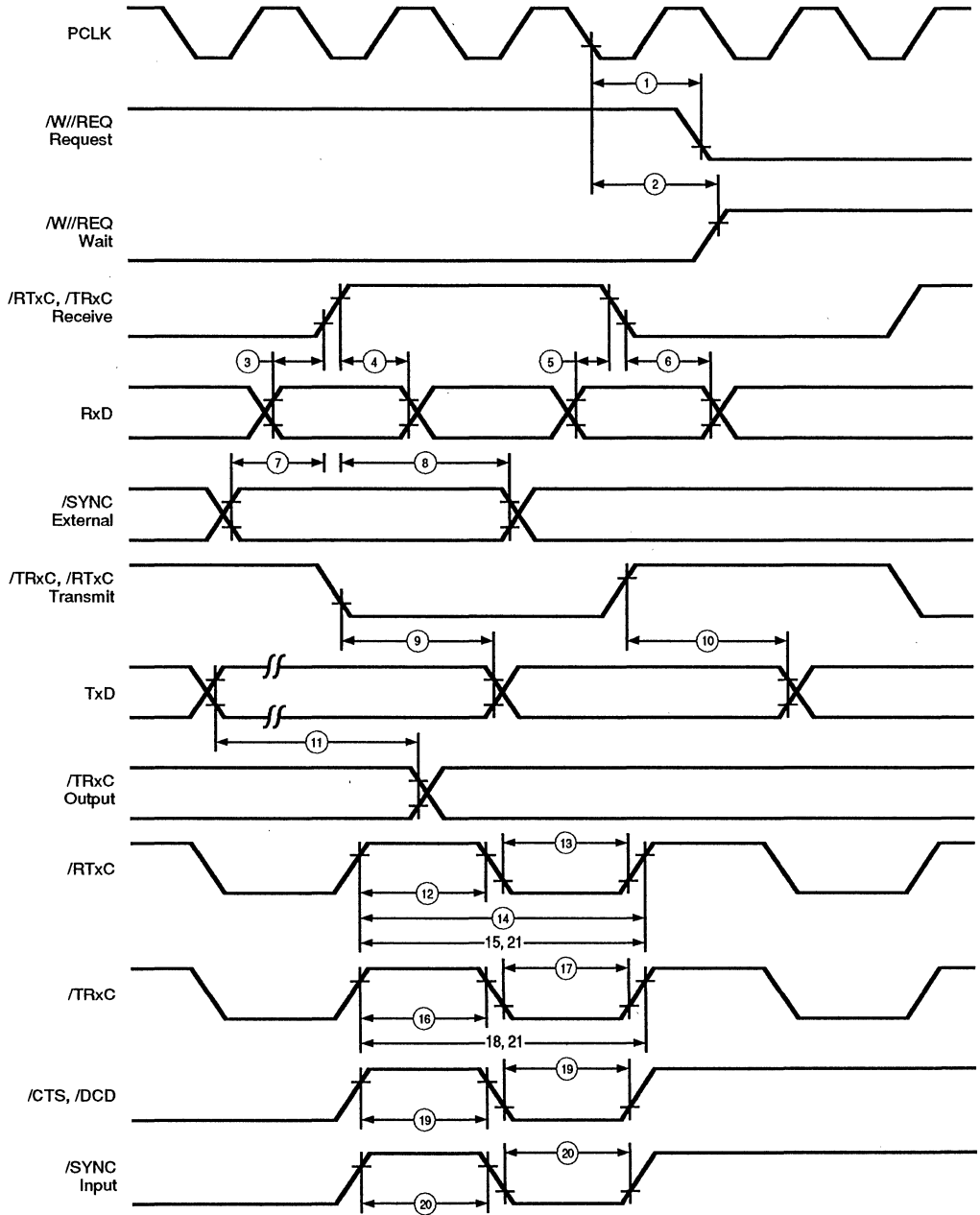


Figure 71. SCC General Timing

**AC CHARACTERISTICS** (Continued)  
SCC General Timing

**Table D. SCC General Timing Parameters**

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	TdPC(REQ)	Clock Fall to /W//REQ Valid		200		120	ns	
2	TdPC(W)	Clock Fall to Wait Inactive		300		220	ns	
3	TsRXD(RXCr)	RxD to /RxC Rise Setup Time			0		ns	[D1]
4	ThRXD(RXCr)	RxD to /RxC Rise Hold Time	125		100		ns	[D1]
5	TsRXD(RXCf)	RxD to /RxC Fall Setup Time	0		0		ns	[D1,4]
6	ThRXD(RXCf)	RxD to /RxC Fall Hold Time	125		100		ns	[D1,4]
7	TsSY(RXC)	/SYNC to /RxC Setup Time	-150		-125		ns	[D1]
8	ThSY(RXC)	/SYNC to /RxC Hold Time	5TcC		5TcC		ns	[D1]
9	TdTXCf(TXD)	/TxC Fall to TxD Delay		150		130	ns	[D2]
10	TdTXCr(TXD)	/TxC Rise to TxD Delay		150		130	ns	[D2,4]
11	TdTXD(TRX)	TxD to /TRxC Delay		140		120	ns	
12	TwRTXh	/RTxC High Width	120		100		ns	[D5]
13	TwRTXI	/RTxC Low Width	120		100		ns	[D5]
14	TcRTX	/RTxC Cycle Time (RxD, TxD)	400		320		ns	[D5,6]
15	TcRTXX	Xtal OSC Period	100	1000	80	1000	ns	[D3]
16	TwTRXh	/TRxC High Width	120		100		ns	[D5]
17	TwTRXI	/TRxC Low Width	120		100		ns	[D5]
18	TcTRX	/TRxC Cycle Time	400		320		ns	[D5,7]
19	TwEXT	/DCD or /CTS Pulse Width	120		100		ns	
20	TwSY	/SYNC Pulse Width	100		70		ns	
21	TxRx(DPLL)	DPLL Cycle Time	50		40		ns	[D6,7]

**Notes to Table D:**

[D1] /RxC is /RTxC or /TRxC, whichever is supplying the receiver clock.

[D2] /TxC is /TRxC or /RTxC, whichever is supplying the transmitter clock.

[D3] Both /RTxC and /SYNC pin has 30pF Capacitors (to ground).

[D4] Parameter applies only to FM encoding/decoding.

[D5] Parameter applies only to transmitter and receiver; baud rate generator timing requirements are different.

[D6] The maximum receive or transmit data rate is 1/4 TcC.

[D7] Applies to DPLL clock source only. Maximum data rate of 1/4 TcC still applies. DPLL.

**AC CHARACTERISTICS** (Continued)  
 SCC System Timing

Figure 72 shows the system timing for the on-chip SCC. Parameters referred to in this figure appear in Table E.

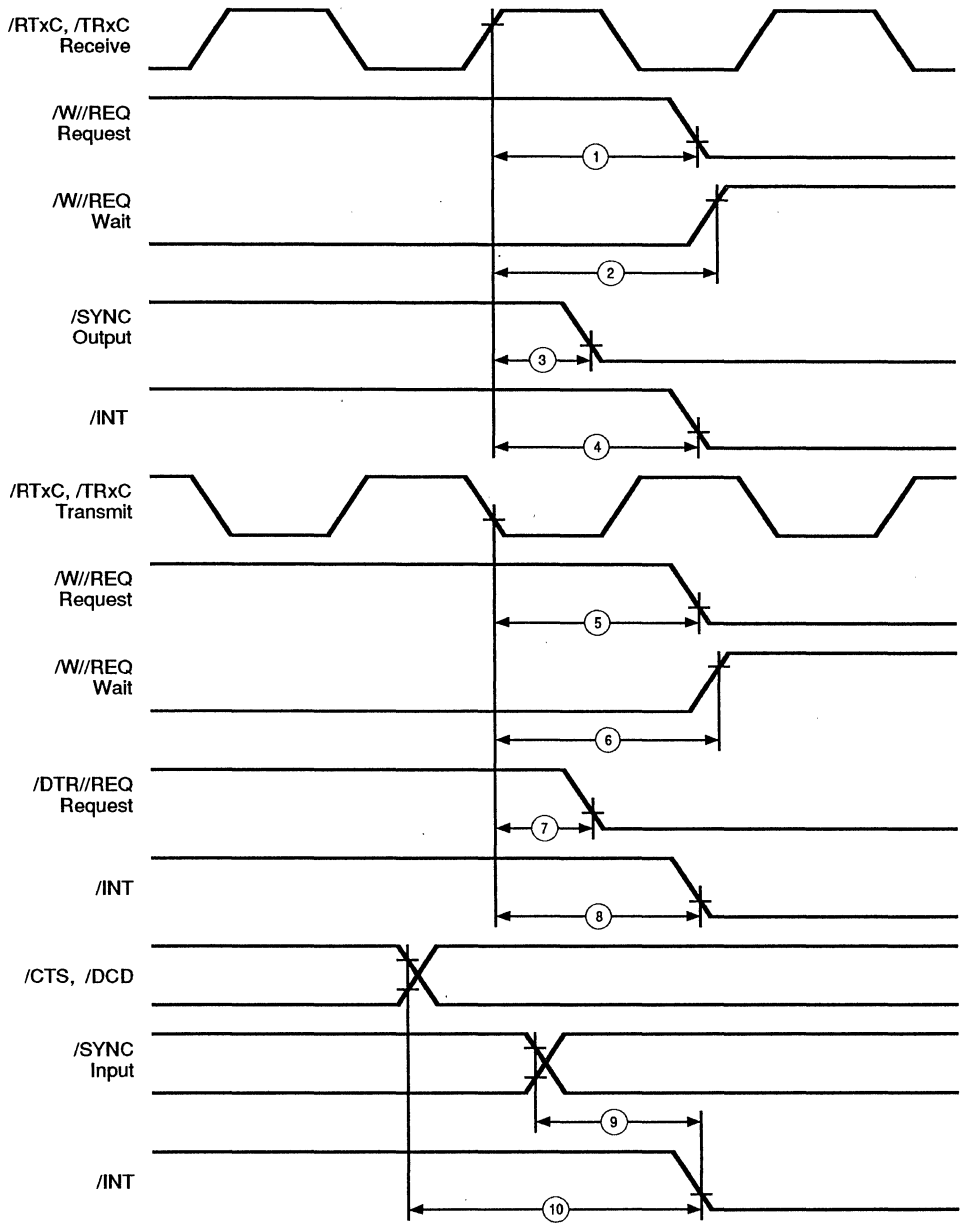


Figure 72. SCC System Timing

**AC CHARACTERISTICS** (Continued)  
 SCC System Timing

**Table E. SCC System Timing Parameters**

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	TdRxC(REQ)	/RxC to /W//REQ Valid	8	12	8	12	TcC	[E2]
2	TdRxC(W)	/RxC to Wait inactive	8	14	8	14	TcC	[E1,2]
3	TdRxC(SY)	/RxC to /SYNC Valid	4	7	4	7	TcC	[E2]
4	TdRxC(INT)	/RxC to /INT Valid	10	16	10	16	TcC	[E1,2]
5	TdTxC(REQ)	/TxC to /W//REQ Valid	5	8	5	8	TcC	[E3]
6	TdTxC(W)	/TxC to Wait inactive	5	11	5	11	TcC	[E1,3]
7	TdRxC(DRQ)	/TxC to /DTR//REQ Valid	4	7	4	7	TcC	[E3]
8	TdTxC(INT)	/TxC to /INT Valid	6	10	6	10	TcC	[E1,3]
9	TdSY(INT)	/SYNC to /INT Valid	2	6	2	6	TcC	[E1]
10	TdEXT(INT)	/DCD or /CTS to /INT Valid	2	6	2	6	TcC	[E1]

**Notes for Table E:**

[E1] Open-drain output, measured with Open-drain test load.

[E2] /RxC is /RTxC or /TRxC, whichever is supplying the receiver clock.

[E3] /TxC is /TRxC or /RTxC, whichever is supplying the transmitter clock.

**AC CHARACTERISTICS** (Continued)  
 PIA General Purpose I/O Port Timing

Figure 73 shows the timing for the PIA ports. Parameters referred to in this figure appear in Table F.

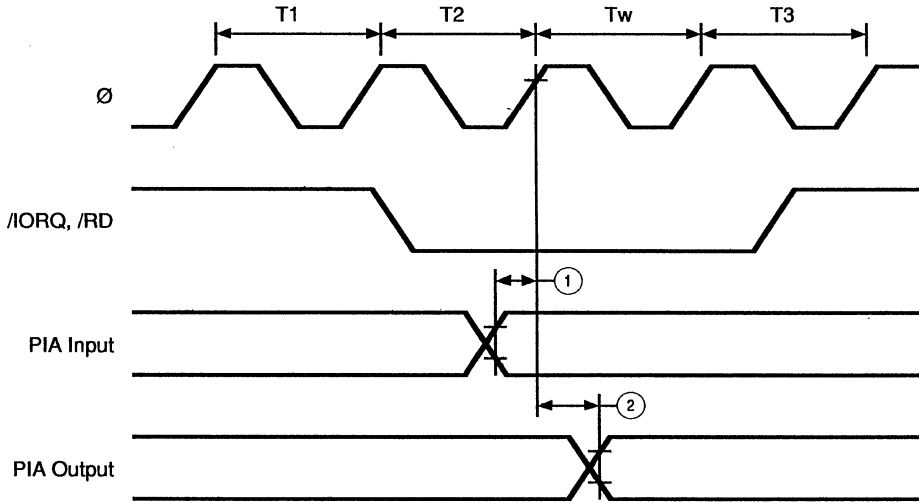


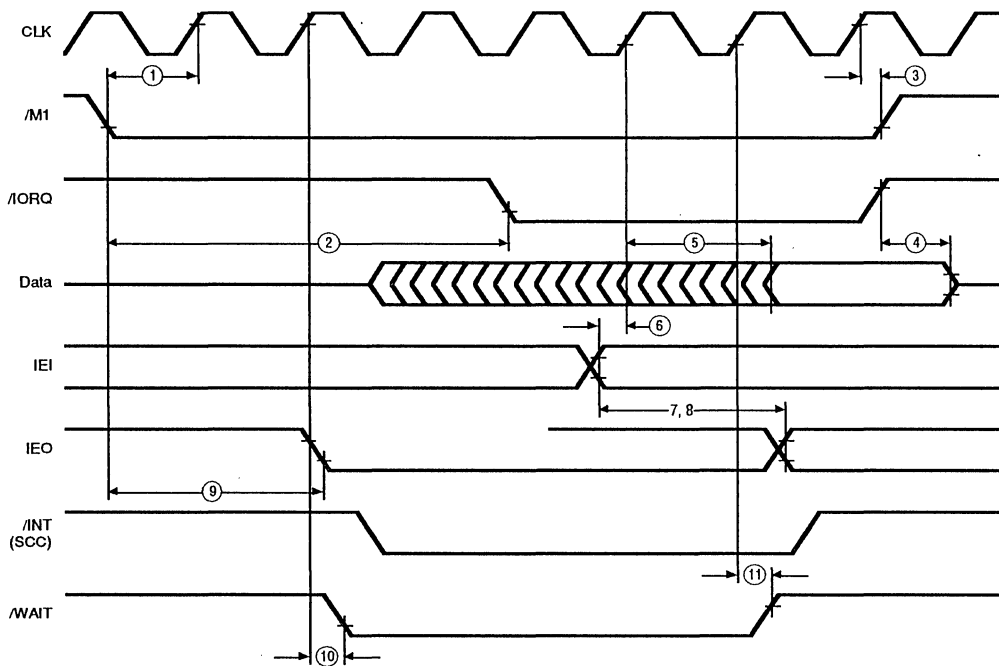
Figure 73. PIA Timing

Table F. PIA General Purpose I/O Timing Parameters

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	TsPIA(C)	PIA Data Setup time to Clock Rise	20	60	20	60	ns	
2	TdCr(PIA)	Clock Rise to PIA Data Valid Delay		60		60	ns	

## Interrupt Daisy-Chain Timing

Figure 74 shows the interrupt daisy-chain timing. Parameters referred to in this figure appear in Table G.



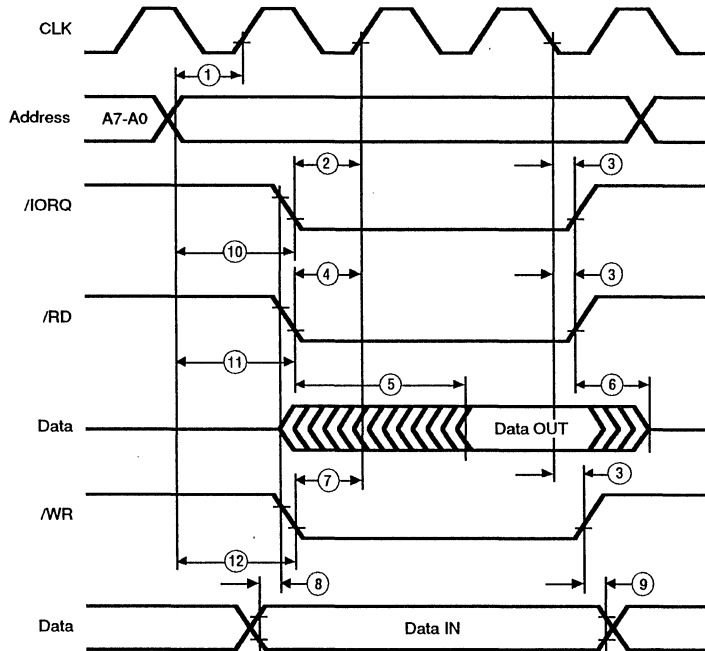
**Figure 74. Interrupt Daisy-Chain Timing**

**Table G. Interrupt Daisy-Chain Timing Parameters**

No	Symbol	Parameter	Z8018110		Z8018112	
			Min	Max	Min	Max
1	TsM1(Cr)	/M1 Fall to Clock Rise Setup Time	20		20	
2	TsM1(IO)INTA	/M1 Fall to /IORQ Fall Setup Time (During INTACK Cycle)	2TcC		2TcC	
3	Th	Hold Time	0		0	
4	TdM1r(DOz)	/M1 Rise to Data Out Float Delay	0		0	
5	TdCr(DO)	Clock Rise to Data Out Delay		120		100
6	TsIEI(TW4)	IEI to T <sub>wait</sub> Rise Setup Time	95		80	
7	TdIEIf(IEOf)	IEI Fall to IEO Fall Delay		60		60
8	TdIEIr(IEOr)	IEO Rise to IEO Rise Delay		60		60
9	TdM1f(IEOf)	/M1 Fall to IEO Fall Delay		140		120
10	TdCWA(f)INTA	Clock Rise to /WAIT Fall Delay		30		25
11	TdCWA(r)INTA	Clock Rise to /WAIT Rise Delay		30		25



**AC CHARACTERISTICS** (Continued)  
Read Write External BUS Master Timing



**Figure 75. Read/Write External BUS Master Timing**

**Table H. External Bus Master Interface Timing (Read/Write Cycles)**

No	Symbol	Parameter	Z8018110		Z8018112	
			Min	Max	Min	Max
1	TsA(Cr)	Address to CLK Rise Setup Time	35		35	
2	TsIO(Cr)	/IORQ Fall to CLK Rise Setup Time	35		35	
3	Th	Hold Time	0		0	
4	TsRD(Cr)	/RD Fall to CLK Rise Setup Time	35		35	
5	TdRD(DO)	/RD Fall to Data Out Delay		120		100
6	TdRlr(DOz)	/RD, /IORQ Rise to Read Data Float	0		0	
7	TsWR(Cr)	/WR Fall to CLK Rise Setup Time	35		35	
8	TsDi(WRf)	Data in to /WR Fall Setup Time	0		0	
9	ThWlr(Di)	/IORQ, /WR Rise to Data In Hold Time	0		0	
10	TsA(IORQf)	Address to /IORQ Fall Setup Time	50		40	
11	TsA(RDf)	Address to /RD Fall Setup Time	50		40	
12	TsA(WRf)	Address to /WR Fall Setup Time	50		40	

## SCC External BUS Master Timing

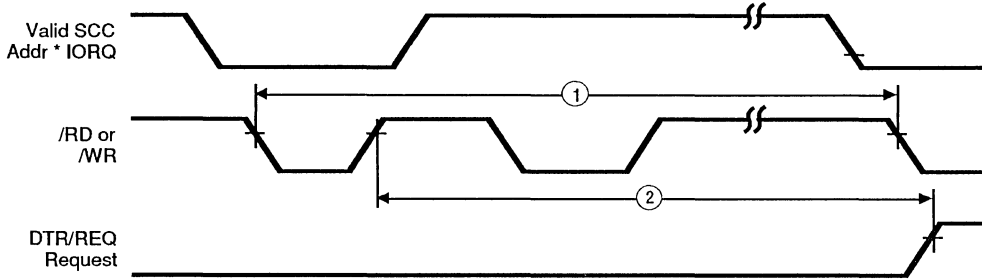


Figure 76. SCC External BUS Master Timing

Table I. External Bus Master Interface Timing (SCC Related Timing)

No	Symbol	Parameter	Z8018110		Z8018112		Unit	Note
			Min	Max	Min	Max		
1	TrC	Valid Access Recovery Time	4TcC		4TcC		nS	[1]
2	TdRD <sub>r</sub> (REQ)	/RD Rise to /DTR//REQ Not Valid Delay	4TcC		4TcC		nS	

[1] Applies only between transactions involving the SCC.

## AC CHARACTERISTICS (Continued)

### Note for Interrupt Acknowledge Cycle and Daisy Chain

When using the interrupt daisy chained device(s) for other than the Z181, these are the following restrictions/notes (without external logic):

The device(s) has to be connected to the higher priority location (Figure 77).

The device(s) IEI-IEO delay has to be less than two clock cycles.

The Z181 on-chip interface logic inserts another three wait states into the interrupt acknowledge cycle to meet the on-chip SCC and the Z80 CTC timing requirements. (Total of five wait states; includes the automatic inserted two wait states).

To meet the timing requirements, the Z181's on-chip circuit generates interface signals for the SCC and CTC. Figure 78 has the timing during the interrupt acknowledge cycle, including the internally generated signals.

The following are three separate cases for the daisy-chain settle times:

**Case 1 - SCC:** The SCC /INTACK signal goes active on the T1 clock fall time. The settle time is from SCC /INTACK active until the SCC /RD signal goes active on the fourth rising wait state clock.

**Case 2 - CTC:** The settle time for the on-chip /IORQ is between the fall of /M1 until the internal CTC /IORQ goes active on the rise of the fourth wait state (the same time as SCC /RD goes active).

**Case 3 - OFF-chip Z80 Peripheral:** The settle time for the off-chip Z80 peripheral is from the fall of /M1 until CTC /IORQ goes active. Since the Z181's external /IORQ signal goes active on the clock fall of the first automatically inserted wait state ( $T_{WA}$ ), the external daisy-chain device has to be connected to the upper chain location. Also, it must settle within two clock cycles.

If any peripheral is connected externally with a lower daisy chain priority than Z181 peripherals, /IORQ has to be delayed by external logic as shown in Figure 79.

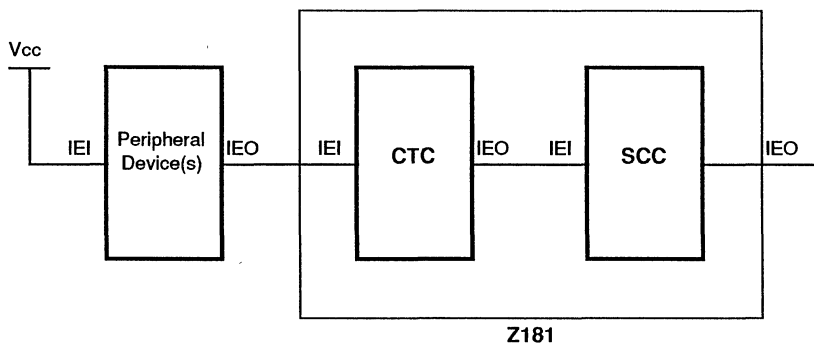


Figure 77. Peripheral Device as Part of the Daisy Chain

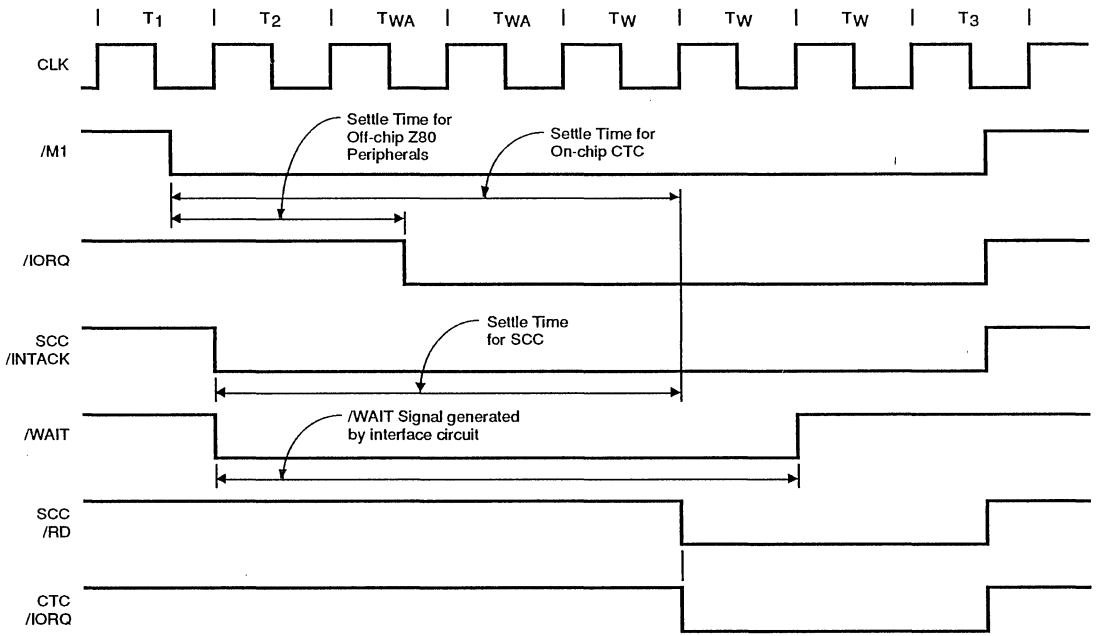


Figure 78. Interrupt Acknowledge Cycle Timing

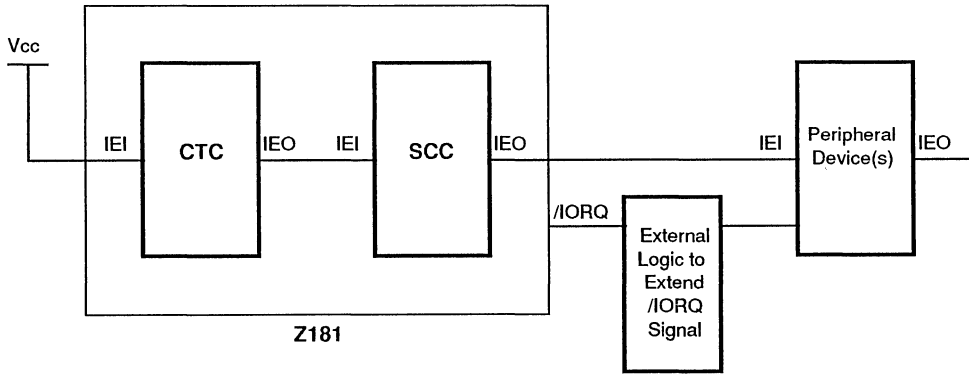


Figure 79. Peripheral Device as Part of the Daisy Chain



## Z280™ MPU Microprocessor Unit

### FEATURES

- Designed in CMOS for low power operations.
- Enhanced Z80® CPU instruction set that maintains object-code compatibility with Z80 microprocessor.
- Three-stage pipelined, 16-bit CPU architecture with user and system modes.
- Direct coprocessor and multiprocessor interface support.
- On-chip paged Memory Management Unit (MMU) addresses up to 16 Mbytes.
- On-chip 256-byte instruction and data associative cache memory with burst load.
- High performance 16-bit Z-BUS® bus interface or 8-bit Z80 CPU compatible bus interface.
- Three on-chip 16-bit counter/timers.
- Four on-chip DMA channels.
- On-chip full duplex UART.
- Refresh controller for dynamic RAMs.
- **On-chip oscillator or direct input clock options.**
- **25 MHz oscillator clock frequency.**

### GENERAL DESCRIPTION

The Z280 microprocessor features a high-performance microprocessor designed to give the end-user a powerful and cost effective solution to application requirements. The Z280 microprocessor unit (MPU) incorporates advanced architectural features that allow fast and efficient throughput and increased memory addressing while maintaining Z80 object-code compatibility. The Z280 microprocessor offers a continuing growth path for present Z80-based designs and serves as a high-performance microprocessor for new, advanced designs.

Central to the Z280 microprocessor is an enhanced version of the Z80 Central Processing Unit (CPU). To assure system integrity, the Z280 microprocessor can operate in either user or system mode, allowing protection of system resources from user tasks and programs. System mode operation is supported by the addition of the system Stack Pointer to the working register set. The IX and IY registers have been modified so that in addition to their regular function as index registers, each register can be accessed as a 16-bit general purpose register or as two byte registers. The R register, used for refresh by the Z80 CPU, is now available to the programmer as a data register in the Z280 microprocessor.

The Z80 CPU instruction set has been retained, meaning that the Z280 microprocessor is completely binary-code compatible with present Z80 code. The basic addressing modes of the Z80 microprocessor have been augmented with the addition of Indexed mode with full 16-bit displacement, Program Counter Relative with 16-bit displacement, Stack Pointer Relative with 16-bit displacement, and Base Index modes. The new addressing modes are incorporated into many of the old Z80 CPU instructions, resulting in greater flexibility and power. Some additions to the instruction set include 8-and 16-bit signed and unsigned multiply and divide, 8-and 16-bit sign extension, and a test and set instruction to support multiprocessing. The 16-bit instructions have been expanded to include 16-bit compare, memory increment, memory decrement, negate, add, and subtract, in addition to the previously mentioned multiply and divide.

A requirement of many of today's microprocessor-based system designs is to increase the memory address space beyond the 64K byte range of typical 8-bit microprocessors. The Z280 microprocessor has an on-chip Memory Management Unit (MMU) that enables addressing of up to 16M bytes of memory. In addition to enabling the address

---

space to be expanded, the MMU performs other memory management functions previously handled by dedicated off-chip memory management devices.

I/O address space has been expanded by the addition of an I/O Page register used to select pages of I/O addresses. The 8-bit I/O Page register can select one of 256 possible pages of I/O addresses to be active at one time, allowing a total of 64K I/O addresses to be accessed.

There are 256 bytes of on-chip memory present on the Z280 MPU. This memory can be configured as a high-speed cache or as a fixed address local memory. When configured as a cache, the memory can be programmed to be instruction only, data only, or both data and instruction. The cache memory allows programs to run significantly faster by reducing the number of external bus accesses. Operation and update of the cache is performed automatically and is completely transparent to the user. When used as a local memory, the addresses are programmable, which permits selected storage of time-critical loops in local memory.

Many features that have traditionally been handled by external peripheral devices have been incorporated in the design of the Z280 microprocessor. The "on-chip peripherals" reduce system chip count and reduce interconnection on the external bus. The Z280 MPU contains an on-chip clock oscillator and a refresh controller that provides 10-bit refresh addresses for dynamic memories. Also present are additional on-chip peripherals to provide system design flexibility. To support high-bandwidth data transmission, four Direct Memory Access (DMA) channels are incorporated on-chip. Each DMA channel operates using full 24-bit source and destination addresses with a 16-bit count. The channels can be programmed to operate in single transaction, burst, or continuous mode. System event counting and timing requirements are met with the help of the three 16-bit counter/timers. The counter/timer functions can be externally controlled with gate and trigger inputs, and can be programmed as retriggeable or nonretriggeable. A full duplex UART, capable of handling a variety of data and character formats, is present to facilitate asynchronous serial communication.

---

## Z280 CPU

### User and System Modes of Operation

The Z280 CPU can operate in either user or system mode. In user mode, some instructions cannot be executed and some registers of the CPU are inaccessible. In general, this mode of operation is intended for use by application programs. In system mode, all of the instructions can be executed and all of the CPU registers can be accessed. This mode is intended for use with programs that perform operating system functions. This separation of CPU resources promotes the integrity of the system, since programs operating in user mode cannot access those aspects of the CPU that deal with system interface events.

The Z280 MPU also features programmable bus timing, allowing the user to tailor timing to the individual system. Upon reset the Z280 microprocessor can be programmed for system timing that is one-fourth, one-half, or equal to the speed of the MPU's internal Central Processing Unit (CPU), with one-half being the default. In addition to clock scaling, programmable wait states can be inserted during various bus transactions. Without the use of external hardware, one to three wait states can be inserted into memory, I/O, and interrupt acknowledge transactions. Furthermore, separate memory wait states can be specified for upper and lower memory areas, facilitating the use of different speeds of ROMs and RAMs in the same system.

An additional feature of the 16-bit bus interface is the ability to support "nibble-mode" dynamic RAMs. Using this feature (known as burst mode), the bus bandwidth of memory read transactions is essentially doubled. Burst mode transactions have the further benefit of allowing the cache to operate more efficiently by guaranteeing a high probability that the contents of the accessed memory will be present in the cache.

The Z280 MPU supports Zilog's Extended Processor Architecture (EPA) in a number of ways. It is capable of trapping Extended Processor Unit (EPU) instructions in order to perform software emulation of the EPU. With its 16-bit external bus interface, the Z280 MPU directly interfaces with an EPU and operates in a manner that is completely transparent to the user and the program.

Multiprocessor system architectures are also supported by the Z280 MPU. When operating in multiprocessor mode, the Z280 MPU's Local Address register is used to distinguish between local and global memory access. Global accesses are controlled through a global request and global acknowledge protocol.

The pin functions and the pin assignments of the Z280 MPU are illustrated in Figures 1 and 2. Figure 3 shows the block diagram.

To further support the dual user/system mode, there are two Stack Pointers—one for the user stack and another for the system stack. These two stacks facilitate the task switching involved when interrupts or traps occur. To ensure that the user stack is free of system information, the information saved on the occurrence of interrupts or traps is always pushed onto the system stack before the new program status is loaded.

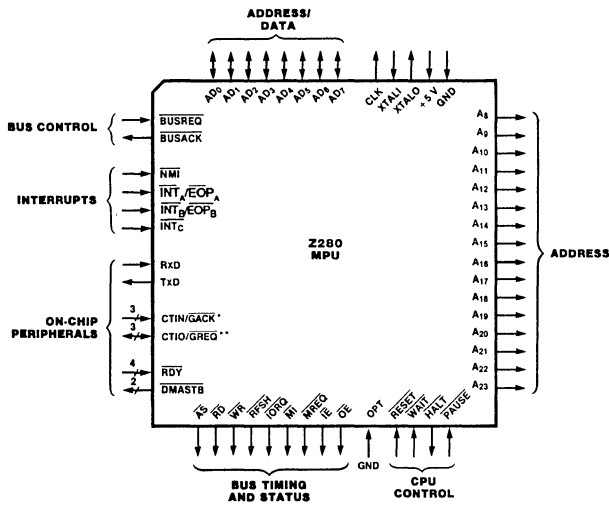


Figure 1a. Z280 Pin Functions, Z80 Bus Configuration (Input OPT tied to GND)

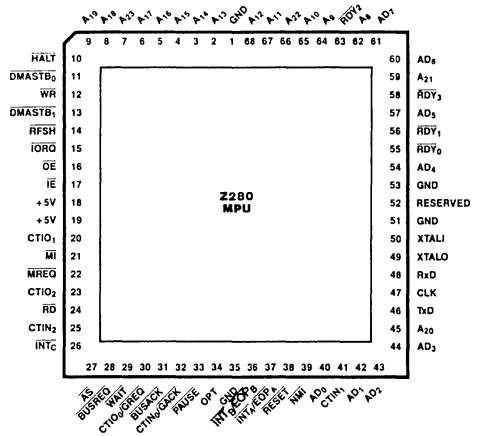


Figure 1b. Z280 Pin Assignments, Z80 Bus (Input OPT tied to GND)

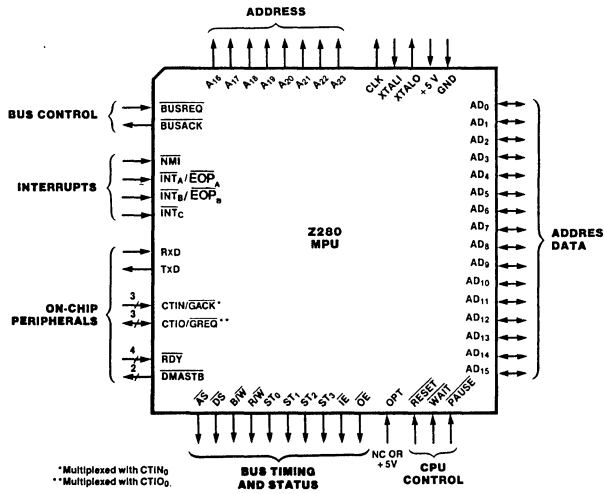


Figure 2a. Z280 Pin Functions, Z-BUS Configuration (Input OPT tied to +5V or not connected)

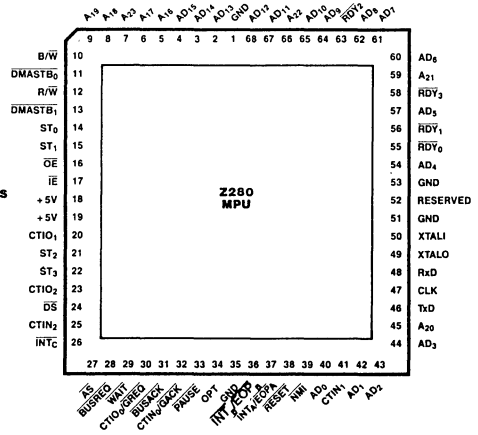


Figure 2b. Z280 Pin Assignments, Z-BUS (Input OPT tied to +5V or not connected)



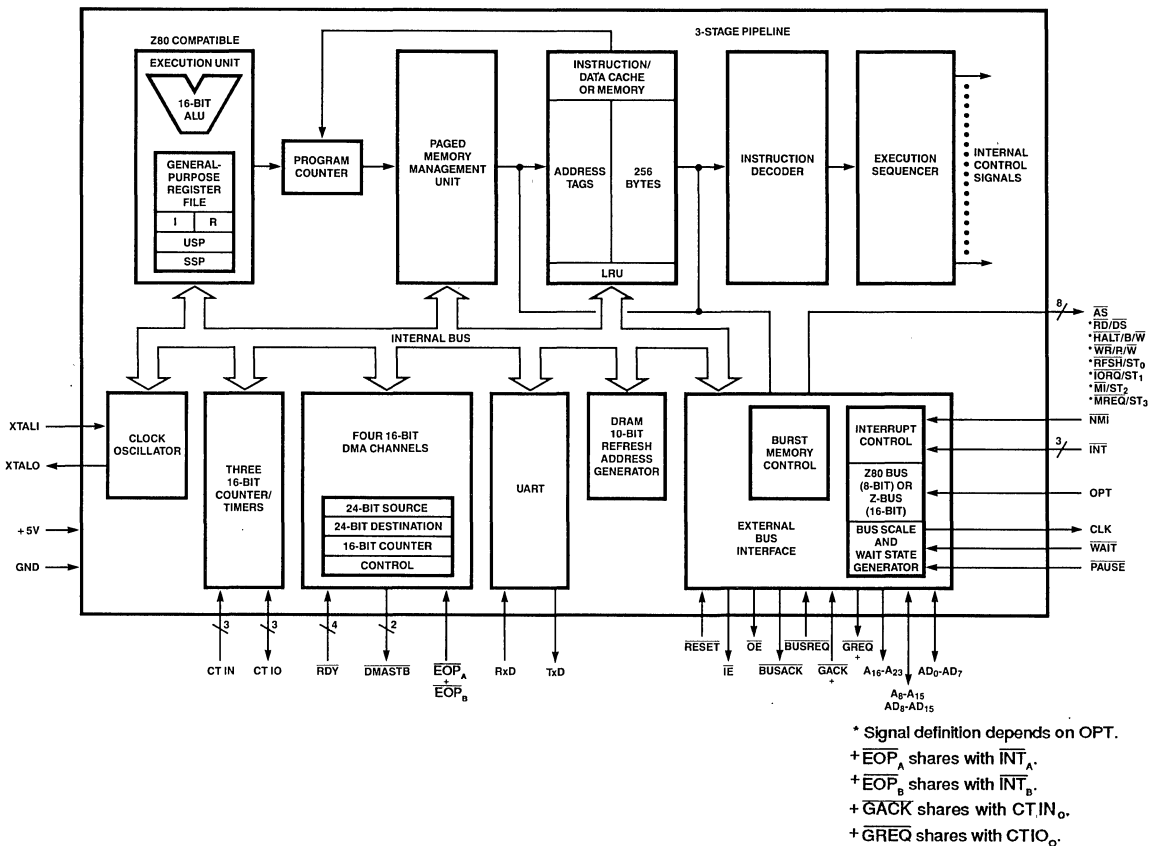


Figure 3. Z280 MPU Block Diagram.

### Address Spaces

The Z280 CPU architecture supports four distinct address spaces corresponding to the different types of locations that can be accessed by the CPU. These four address spaces are:

- CPU register space
- CPU control and status register space
- Memory address space
- I/O address space

**CPU Register Space.** The CPU register space (Figure 4) consists of all of the registers in the CPU register file. The CPU registers are used for data and address manipulation. Access to these registers is specified in the instruction. The CPU registers are labeled A, F, B, C, D, E, H, L, A', F', B', C', D', E', H', L', IX, IY, SSP, USP, PC, I, and R.

### CPU Control and Status Register Space.

The CPU control register space consists of all of the control and status registers found in the CPU control register file (Figure 5). These registers govern the operation of the CPU and are accessible only by the privileged Load Control instruction. The registers in the CPU control file consist of the Bus Timing and Control register, Bus Timing and Initialization register, Local Address register, Cache Control register, Master Status register, Interrupt Status register, Interrupt/Trap Vector Table Pointer, I/O Page register, Trap Control register, and System Stack Limit register.

**Memory Address Space.** Two memory address spaces are supported by the Z280 CPU; one for user and one for system mode of operation. They are selected by the User/System Mode (U/S) bit in the Master Status register, which governs the selection of Page Descriptor registers during address translation.

Each address space can be viewed as a string of 64K bytes numbered consecutively in ascending order. The 8-bit byte is the basic addressable element in the memory address spaces. However, there are other addressable data elements: bits, 2-byte words, byte strings and multiple-byte EPU operands.

The address of a multiple-byte entity is the address of the byte with the lowest address. Multiple-byte entities can be stored beginning at either even or odd memory addresses.

**I/O Address Space.** I/O addresses are generated only by the I/O instructions (IN, OUT, and the I/O block move instructions). Logical I/O addresses are eight bits in length, augmented by the A register on lines A<sub>8</sub>-A<sub>15</sub> in Direct Address addressing mode and by the B register on lines A<sub>8</sub>-A<sub>15</sub> in Indirect Register addressing mode and for block I/O instructions. The 16-bit logical I/O address is always extended by appending the contents of the 8-bit page register to the augmented I/O address. Thus the complete address generated to address an I/O port consists of an I/O page number on A<sub>23</sub>-A<sub>16</sub>, the contents of the A or B register on A<sub>8</sub>-A<sub>15</sub>, and the 8-bit I/O address on A<sub>7</sub>-A<sub>0</sub>.

Unlike memory references, in which a 16-bit word store or fetch can generate two memory references, an I/O word store or fetch is always one I/O bus transaction, regardless of bus size or I/O port address. Note, however, that on-chip peripherals with word registers are accessed via word I/O instructions for those 16-bit registers, regardless of the external bus size (Table 1).

## Data Types

The CPU can operate on bits, binary-coded decimal (BCD) digits (4 bits), bytes (8 bits), words (16 bits), byte strings, and word strings. Bits in registers or memory can be set, cleared, and tested. BCD digits, packed two to the byte, can be manipulated with the Decimal Adjust Accumulator instruction (in conjunction with binary addition and subtraction) and the Rotate Digit instructions. Bytes are operated on by 8-bit load, arithmetic, logical, and shift and rotate instructions. Words are operated on in a similar manner by the 16-bit load and 16-bit arithmetic instructions. Block move and search operations can manipulate byte strings up to 64K bytes long. Block I/O word instructions can manipulate word strings up to 32K words long. To support EPU operations, byte strings up to 16 bytes in length can be transferred by the CPU.

## CPU Registers

The Z280 MPU contains 23 programmable registers (Figure 4) in the CPU register address space.

**Primary and Working Register Set.** The working register set is divided into the two 8-bit register files—the primary file and alternate (designated by ' ') file. Each file contains an 8-bit accumulator (A), a Flag register (F), and six general-purpose registers (B, C, D, E, H, and L). Only one file can be active at any given time. Upon reset, the primary register file is active. Exchange instructions allow the programmer to exchange the active file with the inactive file.

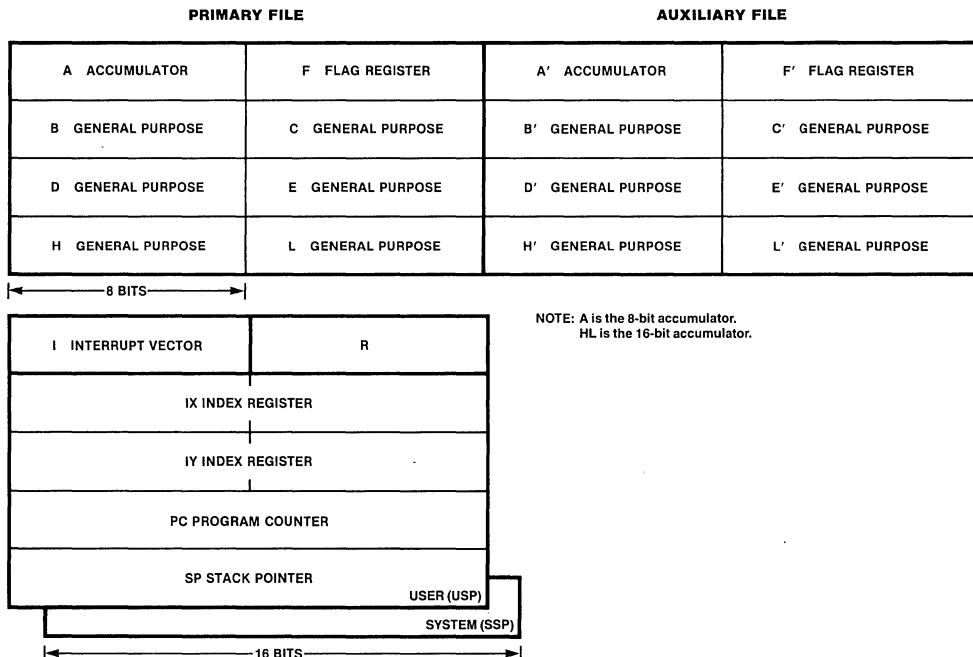
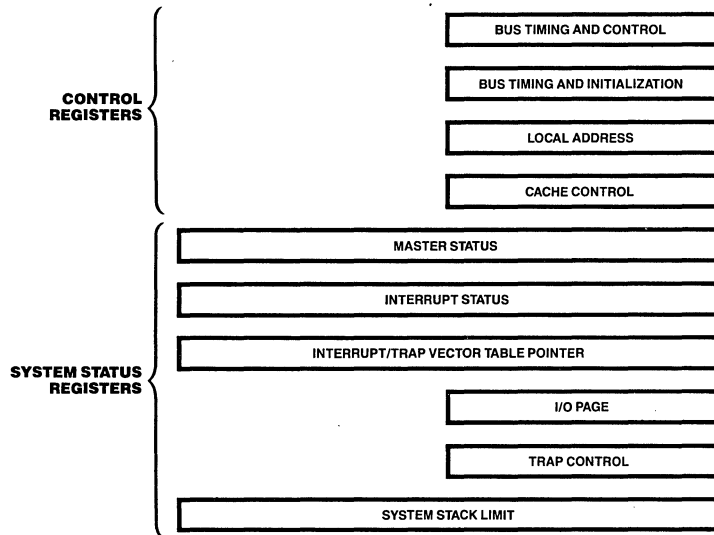


Figure 4. CPU Register Configuration



The accumulator is the destination register for 8-bit arithmetic and logical operations. The six general-purpose registers can be paired (BC, DE, and HL) to form three 16-bit general-purpose registers. The HL register pair serves as a 16-bit accumulator for 16-bit arithmetic operations.

**CPU Flag Register.** The Flag register contains six flags that are set or reset by various CPU operations. This register is illustrated in Figure 6.

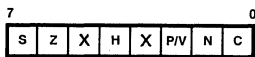


Figure 6. CPU Flag Register

The flags in this register are:

**Carry (C).** This flag is set when an add instruction generates a carry or a subtract instruction generates a borrow. Certain logical and rotate and shift instructions affect the Carry flag.

**Add/Subtract (N).** This flag is used by the Decimal Adjust Accumulator instruction to distinguish between add and subtract operations. The flag is set for subtract operations and cleared for add operations.

**Parity/Overflow (P/V).** During arithmetic operations this flag is set to indicate a two's complement overflow. During logical and rotate operations, this flag is set to indicate even parity of the result or cleared to indicate odd parity.

**Half Carry (H).** This flag is set if an 8-bit arithmetic operation generates a carry or borrow between bits 3 and 4, or if a 16-bit operation generates a carry or borrow between bits 11 and 12. This bit is used to correct the result of a packed BCD addition or subtract operation.

**Zero (Z).** This flag is set if the result of an arithmetic or logical operation is a zero.

**Sign (S).** This flag stores the state of the most significant bit of the accumulator. The Sign flag is also used to indicate the results of a test and set instruction.

### Dedicated MPU Registers

**Index Registers.** The two Index registers, IX and IY, each hold a 16-bit base address that is used in the Indexed addressing mode. The Index registers can also function as general-purpose registers with the upper and lower bytes capable of being accessed individually. The high and low bytes of the IX register are called IXH and IXL. The high and low bytes of the IY register are called IYH and IYL.

**Interrupt Register.** The Interrupt register (I) is used in interrupt mode 2 to generate a 16-bit indirect logical address to an interrupt service routine. The Interrupt register supplies the upper eight bits of the indirect address and the interrupting peripheral supplies the lower eight bits.

**Program Counter.** The Program Counter (PC) is used to sequence through instructions in the currently executing program and to generate relative addresses. The Program Counter contains the 16-bit logical address of the current instruction being fetched from memory.

**R Register.** The R register can be used as a general-purpose 8-bit read/write register. The R register is not associated with the refresh address and its contents are changed only by the user.

**NOTE:** To be compatible with possible future enhancements, a user should write 0's into reserved register bits. A user should not rely on values read from reserved register bits. In figures and tables, unless otherwise noted, reserved bits are labeled with "X".

**Table 1. On-Chip Peripheral I/O Port Addresses**

<b>Peripheral</b>	<b>Address (Hexadecimal)</b>			
<b>Refresh Rate Register</b>	FFxxE8			
<b>UART</b>				
Configuration	FExx10			
Transmitter Control/Status	FExx12			
Receiver Control/Status	FExx14			
Receiver Data	FExx16			
Transmitter Data	FExx18			
<b>MMU</b>				
Master Control	FFxxF0			
Page Descriptor Register Pointer	FFxxF1			
Descriptor Select Port	FFxxF5			
Block Move Port	FFxxF4			
Invalidation I/O Port	FFxxF2			
Page Descriptor Registers *				
User PDR 0	00			
User PDR 1	01			
.	.			
User PDR 14	0E			
User PDR 15	0F			
System PDR 0	10			
System PDR 1	11			
.	.			
System PDR 14	1E			
System PDR 15	1F			
<b>DMA</b>				
Master Control	FFxx1F			
	<b>DMA0</b>	<b>DMA1</b>	<b>DMA2</b>	<b>DMA3</b>
Destination Address (bits 0-11)	FFxx00	FFxx08	FFxx10	FFxx18
Destination Address (bits 12-23)	FFxx01	FFxx09	FFxx11	FFxx19
Source Address (bits 0-11)	FFxx02	FFxx0A	FFxx12	FFxx1A
Source Address (bits 12-23)	FFxx03	FFxx0B	FFxx13	FFxx1B
Count	FFxx04	FFxx0C	FFxx14	FFxx1C
Transaction Descriptor	FFxx05	FFxx0D	FFxx15	FFxx1D
<b>Counter/Timer</b>				
	<b>C/T0</b>	<b>C/T1</b>	<b>C/T2</b>	
Configuration	FExxE0	FExxE8	FExxF8	
Command/Status	FExxE1	FExxE9	FExxF9	
Time Constant	FExxE2	FExxEA	FExxFA	
Count-Time	FExxE3	FExxEB	FExxFB	

\* The Page Descriptor register address must be loaded into the Page Descriptor Register Pointer in order to access that Page Descriptor register.

**Stack Pointers.** Two hardware Stack Pointers, the User Stack Pointer (USP) and the System Stack Pointer (SSP), support the dual mode of operation of the microprocessor. The SSP is used for saving information when an interrupt or trap occurs and for supporting subroutine calls and returns in system mode. The USP is used for supporting subroutine calls and returns in user mode.

**Status and Control Registers.** There are ten status and control registers available to the programmer in the Z280 MPU. Table 2 shows the addresses occupied by the registers in the status and control register addressing space.

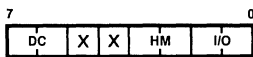
**Table 2. Status and Control Register I/O Port Addresses**

Control Register Name	Address (Hexadecimal)
Bus Timing and Control	Control 02
Bus Timing and Initialization	Control FF
Cache Control <sup>1</sup>	Control 12
Interrupt Status	Control 16
Interrupt/Trap Vector Table	Control 06
I/O Page Register	Control 08
Local Address Register <sup>2</sup>	Control 14
Master Status (MSR)	Control 00
Stack Limit	Control 04
Trap Control	Control 10

**NOTES:**

1. See section on on-chip memory for register description.
2. See section on multiprocessing mode of operation for register description.

**Bus Timing and Control Register.** This 8-bit register (Figure 7) governs the timing of transactions to high memory addresses and the daisy-chain timing for interrupt requests, as well as the functionality of requests on the various Z280 MPU interrupt request lines.



**Figure 7. Bus Timing and Control Register**

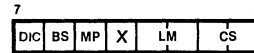
The fields in this register are:

**I/O Wait Insertion (I/O).** This 2-bit field specifies the number of additional wait states (in addition to the one automatically inserted for I/O) to be inserted by the CPU in both I/O transactions and vector response timing (00 = none, 01 = one, 10 = two, 11 = three).

**High Memory Wait Insertion (HM).** This 2-bit field specifies the number of automatic wait states (00 = none, 01 = one, 10 = two, 11 = three) for the CPU to insert in memory transactions when the MMU is enabled and there is a 1 in bit 15 of the selected Page Descriptor register.

**Daisy Chain Timing (DC).** This 2-bit field determines the number of additional automatic wait states the CPU inserts while the interrupt acknowledge daisy chain is settling (00 = none, 01 = one, 10 = two, 11 = three). A value of 01 in the DC field indicates that one additional cycle will be added to the four cycles that normally elapse between interrupt acknowledge, AS and DS (or IORQ) assertions.

**Bus Timing and Initialization Register.** This 8-bit register (Figure 8) is used to specify the duration of control signals for the external interface bus when the MMU is disabled or when the MMU is enabled and there is a 0 in bit 15 of the selected Page Descriptor register. It also controls the relationship between internal processor clock rates and bus timing. It can be programmed by external hardware upon reset.



**Figure 8. Bus Timing and Initialization Register**

During reset this register is initialized to one of two settings, depending on the state of the WAIT input line on the rising edge of Reset: if the WAIT line is not asserted, the register is set to 00<sub>H</sub>. If the WAIT line is asserted during reset, then this register is set to the contents of the AD lines.

The fields in this register are:

**Clock Scaling (CS).** This 2-bit field specifies the scaling of the CPU clock for all bus transactions (00 = one bus clock cycle is equal to two internal processor clock cycles, 01 = bus clock cycle is equal to the internal processor clock cycle, 10 = one bus clock cycle is equal to four internal processor clock cycles, 11 = reserved). This field cannot be modified by software.

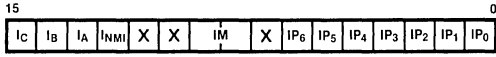
**Low Memory Wait Insertion (LM).** This 2-bit field specifies the number of automatic wait states (00 = none, 01 = one, 10 = two, 11 = three) for the CPU to insert in memory transactions when the MMU is disabled or when the MMU is enabled and there is a 0 in bit 15 of the selected Page Descriptor register.

**Multiprocessor Configuration Enable (MP).** This 1-bit field enables the multiprocessor mode of operation (0 = disabled, 1 = enabled). (See the Multiprocessor Mode section.)

**Bootstrap Mode Enable (BS).** This 1-bit field enables the bootstrap mode of operation (0 = disabled, 1 = enabled). (See the UART section for details about bootstrap mode.)

**Direct Input Clock Option (DIC).** This bit when set (0=disabled, 1=enabled) selects the direct clock source option for the XTALI input. In this mode, the crystal oscillator and divide by 2 circuits are bypassed and XTALI input is used to directly generate the MPU internal clocks. The XTALI input must have TTL levels, 50% duty cycle, and 10MHz maximum frequency. When disabled, the input frequency is divided by 2 to generate the internal processor clock. A maximum crystal or input clock frequency of 20MHz is supported in this case.

**Interrupt Status Register.** This 16-bit register (Figure 9) indicates which interrupt mode is in effect and which interrupt sources have interrupt requests pending. It also contains the bits that specify whether the interrupt inputs are to be vectored. Only the interrupt vector enable bits are writeable; all other bits are read-only.



**Figure 9. Interrupt Status Register**

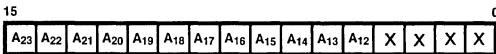
The fields in this register are:

**Interrupt Request Pending (IP).** When bit  $IP_n$  is set to 1, an interrupt request from sources at level  $n$  is pending. (See the Interrupt and Trap Structure section.)

**Interrupt Mode (IM).** A value of  $n$  in this 2-bit field indicates that interrupt mode  $n$  is in effect. This field can be changed by executing the IM instruction.

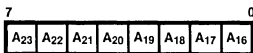
**Interrupt Vector Enable (I).** These four bits indicate whether each of the four interrupt inputs are to be vectored. When  $I_n$  is set to 1, interrupts on the Interrupt  $n$  line are vectored when the CPU is in interrupt mode 3; when cleared to 0, all interrupts on this line use the same entry in the Interrupt/Trap Vector Table. These bits are ignored except in interrupt mode 3.

**Interrupt/Trap Vector Table Pointer.** This 16-bit register (Figure 10) contains the most significant 12 bits of the physical address at the beginning of the Interrupt/Trap Vector Table: the lower 12 bits of the physical address are assumed to be 0.



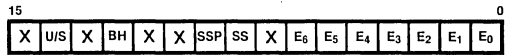
**Figure 10. Interrupt/Trap Vector Table Pointer**

**I/O Page Register.** This 8-bit register (Figure 11) indicates the bits to be appended to the 16 bits that are output during the I/O address phase of I/O transactions.



**Figure 11. I/O Page Register**

**Master Status Register.** The Master Status register (Figure 12) is a 16-bit register containing status information about the currently executing program. This register is cleared to 0 during reset.



**Figure 12. Master Status Register**

The fields in this register are:

**Interrupt Request Enable ( $E_n$ ).** There are seven Interrupt Enable bits, one for each type of maskable interrupt source (both external and internal). When bit  $E_n$  is set to 1, interrupt requests from sources at level  $n$  are accepted by the CPU; when this bit is cleared to 0, interrupt requests at level  $n$  are not accepted.

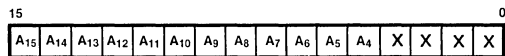
**Single-Step (SS).** While this bit is set to 1, the CPU is in single-stepping mode; while this bit is cleared to 0, automatic single-stepping is disabled. This bit is automatically cleared when a trap or interrupt is taken.

**Single-Step Pending (SSP).** While this bit is set to 1, the CPU generates a trap prior to executing an instruction. The SS bit is automatically copied into this field at the completion of each instruction. This bit is automatically cleared to 0 when a Single-Step, Page Fault, Privileged Instruction, Breakpoint-on-Halt or Division trap is taken so that the SSP bit in the saved Master Status register is cleared to 0.

**Breakpoint-on-Halt Enable (BH).** While this bit is set to 1, the CPU generates a Breakpoint trap whenever a HALT instruction is encountered; while this bit is cleared to 0, the HALT instruction is executed normally.

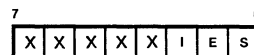
**User/System Mode (U/S).** While this bit is cleared to 0, the CPU is in the system mode of operation; while it is set to 1 the CPU is in the user mode of operation.

**System Stack Limit Register.** This 16-bit register (Figure 13) indicates when a System Stack Overflow Warning trap is to be generated. If enabled, by setting a control bit in the Trap Control register, pushes onto the system stack cause the 12 most significant bits in this register to be compared to the upper 12 bits of the system Stack Pointer and a trap is generated if they match.



**Figure 13. System Stack Limit Register**

**Trap Control Register.** This 8-bit register (Figure 14) enables the maskable traps. Upon reset this register is initialized to all 0s.



**Figure 14. Trap Control Register**

---

The bits in this register are:

**System Stack Overflow Warning (S).** While this bit is set to 1, the CPU generates a Stack Overflow Warning trap when the system stack enters the specified region of memory.

**EPU Enable (E).** While this bit is cleared to 0, the CPU generates a trap whenever an EPA instruction is encountered.

**Inhibit User I/O (I).** While this bit is set to 1, the CPU generates a Privileged Instruction trap when an I/O instruction is encountered in user mode.

**Cache Control and Local Address Registers.** See the On-Chip Memory section for information about the Cache Control register and the Multiprocessor Mode section for information about the Local Address register.

### Interrupt and Trap Structure

The Z280 MPU provides a very flexible and powerful interrupt and trap structure. Interrupts are external asynchronous events requiring CPU attention and are generally triggered by peripherals needing service. Traps are synchronous events resulting from the execution of certain instructions.

**Interrupts.** Two types of interrupt, nonmaskable and maskable, are supported by the Z280 MPU. The nonmaskable interrupt ( $\overline{\text{NMI}}$ ) cannot be disabled (masked) by software and is generally reserved for highest priority external events that require immediate attention. Maskable interrupts, however, can be selectively disabled by software. Both nonmaskable and maskable interrupts can be programmed to be vectored or nonvectored. Interrupts are always accepted between instructions and acknowledged after execution of the prior instruction is complete. The block move, search, and I/O instructions can be safely interrupted after any iteration and restarted after the interrupt is serviced.

**Interrupt Sources.** The Z280 MPU accepts nonmaskable interrupts on the  $\overline{\text{NMI}}$  pin only. The Z280 MPU accepts maskable interrupts on the  $\overline{\text{INT}}$  pins and from the on-chip counter/timers, DMA channels, and the UART receiver and transmitter.

Interrupt Lines A, B, and C can be selectively programmed to support vectored interrupts by setting the appropriate bits in the Interrupt Status register. The external interrupts can be programmed to be vectored or nonvectored in interrupt mode 3.

**Interrupt Modes of Operation.** The CPU has four modes of interrupt handling. The first three modes extend the Z80 interrupt modes to accommodate additional interrupt input lines in a compatible fashion. The fourth mode provides more flexibility in handling the interrupts. On-chip peripherals use the fourth mode regardless of which mode is selected for externally generated interrupt requests. The interrupt mode is selected by using the privileged instructions IM 0, IM 1, IM 2, or IM 3. On reset, the Z280 MPU is automatically set to interrupt mode 0. The current interrupt mode in effect can be read from the Interrupt Status register.

**Mode 0.** This mode is identical to the 8080 interrupt response mode. With this mode, the interrupting device on any of the maskable interrupt lines can place a call or restart instruction on the data bus and the CPU will execute it. As a result, the interrupting device, instead of the memory, provides the next instruction to be executed.

**Mode 1.** When this mode is selected, the CPU responds to a maskable external interrupt by executing a restart to the logical address 0038<sub>H</sub> in the system program address space.

**Mode 2.** This mode is a vectored interrupt response mode. With a single 8-bit byte from the interrupting device, an indirect call can be made to any memory location. With this mode the system maintains a table of 16-bit starting addresses for every interrupt service routine. This table can be located anywhere in the system mode logical data address space on a 256-byte boundary. When an interrupt is accepted, a 16-bit pointer is formed to obtain the desired interrupt service routine starting address from the table. The upper eight bits of this pointer are formed from the contents of the I register. The lower eight bits of the pointer must be supplied by the interrupting device. The 16-bit pointer so formed is treated as a logical address in the system data address space, which can be translated by the MMU to a physical address.

**Mode 3.** This is the intended mode of operation for systems that take advantage of the enhancements of the Z280 microprocessor (such as single-step and user/system mode) since the Master Status register is automatically saved and another loaded for the interrupts. Also, vector tables can be used for the external interrupt sources to provide more interrupt vectors for the Z8000<sup>®</sup> family, Z80 family, and Z8500 Universal Peripherals. When an interrupt request (either maskable or nonmaskable) is accepted, the Master Status register, the address of the next instruction to be executed, and a 16-bit "reason code" are pushed onto the system stack. A new Master Status register and Program Counter are then fetched from the Interrupt/Trap Vector Table. The "reason code" for externally generated interrupts is the contents of the bus during the interrupt acknowledge sequence; for 8-bit data buses, the least significant byte of the reason code is all 1's. For interrupts generated by on-chip peripherals, the reason code identifies which peripheral generated the interrupt and is identical to the vector address in the Interrupt/Trap Vector Table. The Interrupt/Trap Table Pointer is used to reference the table.

**Traps.** The Z280 CPU supports eight traps that are generated internally. The following traps can be disabled: the EPA trap, which allows software to emulate an EPU; the Stack Warning trap, which is taken at the end of an instruction causing the trap; the Breakpoint-on-Halt trap, which is taken when a HALT instruction is encountered; and the Single-Step trap, which is taken for each instruction. In addition, I/O instructions can be specified as privileged instructions. Traps cause the instruction to be terminated without altering CPU registers (except for the System Stack

---

Pointer, which is modified when the program status is pushed onto the system stack).

The saving of the program status on the system stack and the fetching of a new program status from the Interrupt/Trap Vector Table is the same in any interrupt mode of operation.

Traps can only occur if the trap generating features of the Z280 CPU (such as System Stack Overflow warning) have been explicitly enabled. Traps cannot occur on instructions of the Z80 instruction set unless explicitly enabled by the operating system using Z280 CPU extensions.

*Extended Instruction.* This trap occurs when the CPU encounters an extended instruction while the Extended Processing Architecture (EPA) bit in the Trap Control register is 0. Four trap vectors are used by the EPA trap—one for each type of EPA instruction. This greatly simplifies trap handlers that use I/O instructions to access an EPU or software to emulate an EPU.

*Privileged Instruction.* This trap occurs whenever an attempt is made to execute a privileged instruction while the CPU is in user mode (User/System Mode control bit in the Master Status register is 1).

*System Call.* This trap occurs whenever a System Call (SC) instruction is executed.

*Access Violation.* This trap occurs whenever the MMU's translation mode is enabled and an address to be translated is invalid or (for writes) is write-protected.

*System Stack Overflow Warning.* This trap occurs only while the Stack Overflow Warning bit in the Trap Control register is set to 1. For each system stack push operation, the most significant bits in the Stack Pointer register are compared with the contents of the Stack Limit register and a trap is signaled if they match. The Stack Overflow Warning bit is then automatically cleared in order to prevent repeated traps.

*Division Exception.* This trap occurs whenever the divisor is zero (divide-by-zero case) or the true quotient cannot be represented in the destination precision (overflow); the CPU flags are set to distinguish these two cases.

*Single-Step.* This trap occurs before executing an instruction if the Single-Step Pending control bit in the Master Status register is set to 1. Two control bits in the Master Status register are used for the Single-Step trap. The Single-Step bit (bit 8), on being set when previously clear, causes a trap to occur after the execution of the next instruction. While this bit is set to 1, if an instruction execution causes a trap, the Single-Step trap occurs after the execution of the trap-handling routine. The Single-Step

Pending bit (bit 9), is used by the processor to ensure that only one Single-Step trap occurs for each instruction executed while the Single-Step bit is set to 1.

*Breakpoint-on-Halt.* This trap occurs whenever the Breakpoint-on-Halt control bit in the Master Status register is 1 and a HALT instruction is encountered.

**Interrupt and Trap Disabling.** Maskable interrupts can be enabled or disabled independently via software by setting or clearing the appropriate control bits in the Master Status register.

A 7-bit mask field in the Master Status register indicates which of the requested interrupts will be accepted. Interrupt requests are grouped as follows, with each group controlled by a separate Interrupt Enable control bit. The list is presented in order of decreasing priority, with sources within a group listed in order of descending priority.

- Maskable Interrupt A line (bit 0)
- Counter/Timer 0, DMA0 (bit 1)
- Maskable Interrupt B line (bit 2)
- Counter/Timer 1, UART receiver, DMA1 (bit 3)
- Maskable Interrupt C line (bit 4)
- UART Transmitter, DMA2 (bit 5)
- Counter/Timer 2, DMA3 (bit 6)

When a source of interrupts has been disabled, the CPU ignores any interrupt request from that source.

The System Stack Overflow Warning trap, Privileged Instruction trap (I/O instructions in user mode), or Extended Instruction trap can be enabled by setting control bits in the Trap Control register, and the Single-Step and Breakpoint-on-Halt trap can be enabled by setting control bits in the Master Status register; these are the only traps that can be disabled.

**Interrupt/Trap Vector Table.** The format of the Interrupt/Trap Vector Table consists of pairs of Master Status register and Program Counter words, one pair for each separate on-chip interrupt or trap source. For each external interrupt, there is a separate Master Status register word and Program Counter word (for use if the input is not vectored). If the external interrupt is vectored, a vector table consisting of one Program Counter word for each of the 128 possible vectors that can be returned for each input line is used instead of the dedicated Program Counter word; thus for vectored interrupts, there is only one Master Status register for each interrupt type.



The format of the Interrupt/Trap Vector Table is shown in Table 3.

**Table 3. Interrupt/Trap Vector Table**

Address (Hexadecimal)	Contents
00	Reserved
04	NMI Vector
08	Interrupt Line A Vector
0C	Interrupt Line B Vector
10	Interrupt Line C Vector
14	<b>Counter/Timer 0 Vector</b>
18	<b>Counter/Timer 1 Vector</b>
1C	<b>Reserved</b>
20	<b>Counter/Timer 2 Vector</b>
24	DMA0 Vector
28	DMA1 Vector
2C	DMA2 Vector
30	DMA3 Vector
34	UART Receiver Vector
38	UART Transmitter Vector
3C	Single-Step Trap Vector
40	Breakpoint-on-Halt Trap Vector
44	Division Exception Trap Vector
48	Stack Overflow Warning Trap Vector
4C	Page Fault Trap Vector
50	System Call Trap Vector
54	Privileged Instruction Trap Vector
58	EPU ← Memory Trap Vector
5C	Memory ← EPU Trap Vector
60	A ← EPU Trap Vector
64	EPU Internal Operation Trap Vector
68-6C	Reserved
70-16E	<b>128 Program Counter Values for NMI and Interrupt Line A Vectors (MSR from 04 and 08, respectively)</b>
170-26E	<b>128 Program Counter Values for Interrupt Line B Vectors(MSR from 0C)</b>
270-36E	<b>128 Program Counter Values for Interrupt Line C Vectors(MSR from 10)</b>

## Addressing Modes

Addressing modes (Figure 15) are used by the CPU to calculate the effective address of an operand needed for execution of an instruction. Nine addressing modes are supported by the Z280 CPU. Of these nine, four are additions to the Z80 addressing modes (Indexed with 16-bit displacement, Stack Pointer Relative, Program Counter Relative, and Base Index) and the remaining five modes are either existing or extensions to the existing Z80 addressing modes.

**Register.** The operand is one of the 8-bit registers (A, B, C, D, E, H, L, IXH, IXL, IYH or IYL); or one of the 16-bit registers (BC, DE, HL, IX, IY, or SP), or one of the special byte registers (I or R).

**Immediate.** The operand is in the instruction itself and has no effective address.

**Indirect Register.** The contents of a register specify the effective address of an operand. The HL register is the register used for memory accesses. (For the Load To or Load From Accumulator instruction, BC and DE can also be used for indirection; for the JP instruction, IX and IY can also be used for indirection.) The C register is used for I/O and control register space accesses.

**Direct Address.** The effective address of the operand is the location whose address is contained in the instruction. Depending on the instruction, the specified operand is either in the I/O or data memory address space.

**Indexed.** The effective address of the operand is the location specified by adding the 16-bit address contained in the instruction to a two's complement "index" contained in the HL, IX, or IY register.

**Short Index.** The effective address of the operand is the location computed by adding the 8-bit two's complement signed displacement contained in the instruction to the contents of the IX or IY register. This addressing mode is equivalent to the Z80 CPU indexed mode.

**Program Counter Relative.** An 8- or 16-bit displacement contained in the instruction is added to the Program Counter to generate the effective address of the operand.

**Stack Pointer Relative.** The effective address of the operand is the location computed by adding a 16-bit two's complement displacement contained in the instruction to the contents of the Stack Pointer.

**Base Index.** The effective address of the operand is the location whose address is computed by adding the contents of HL, IX, or IY to the contents of another of these three registers.

---

## EXTENDED PROCESSING ARCHITECTURE

### Features

The Zilog Extended Processing Architecture (EPA) provides an extremely flexible and modular approach to expanding both the hardware and software capabilities of the Z280 CPU. Features of the EPA include:

- Allows Z280 CPU instruction set to be extended by external devices.
- Increases throughput of the system by using up to four specialized external processors in parallel with the CPU.
- Permits modular design of Z280 CPU-based systems.
- Provides easy management of multiple microprocessor configurations via "single instruction stream" communication.
- Direct interconnection between EPUs and Z280 MPU requires no additional external supporting logic.
- EPUs can be added as the system grows and as EPUs with specialized functions are developed.

### General Description

The processing power of the Zilog Z-BUS Z280 microprocessor can be boosted beyond its intrinsic capability by the Extended Processing Architecture (EPA). The EPA allows the Z280 CPU to accommodate up to four Extended Processing Units (EPUs), which perform specialized functions in parallel with the CPU's main instruction execution stream.

The EPUs connect directly to the Z-BUS and continuously monitor the CPU instruction stream for an instruction intended for the EPU (template). When a template is detected, the appropriate EPU responds, obtaining or placing data or status information on the Z-BUS by using the Z280 CPU-generated control signals and performing its function as directed.

The CPU is responsible for instructing the EPU and delivering operands and data to it. The EPU recognizes templates intended for it and executes them, using data supplied with the template and/or data within its internal registers. There are three classes of EPU instructions:

- Data transfers between main memory and EPU registers
- Data transfers between CPU registers and EPU status registers
- EPU internal operations

Six addressing modes can be utilized with transfers between EPU registers and the MPU and main memory:

- Indirect Register
- Direct Address
- Indexed
- Program Counter Relative
- Stack Pointer Relative
- Base Index

In addition to the hardware-implemented capabilities of the EPA, there is an extended instruction trap mechanism to permit software simulation of EPU functions. An EPU present bit in the Z280 MPU Trap Control register indicates **whether actual EPUs are present or not. If not, the CPU generates a trap when an extended instruction is detected, and a software "trap handler" can emulate the desired EPU function. Thus, the EPA software trap routine supports systems not containing an EPU.**

EPA and CPU instruction execution are shown in Figure 16. If an instruction has been fetched and decoded, the CPU determines whether or not it is an EPU instruction. If the instruction is an EPU instruction, the state of the EPU Enable bit in the Trap Control register is examined. If the EPU Enable bit is reset ( $E = 0$ ), the CPU generates a trap and the EPU instruction can be simulated by an EPU instruction trap software routine. However, if the EPU Enable bit is set ( $E = 1$ ), indicating that an EPU is present in the system, then the 4-byte EPU template is fetched from memory. The fetching of the EPU template is indicated by the status lines  $ST_0-ST_3$ . Each EPU continuously monitors the Z-BUS and the status lines for its own templates. After fetching the EPU template, the CPU, if necessary, transfers appropriate data between the EPU and memory or between the CPU and the EPU. These transactions are indicated by unique encodings of the status lines. If the EPU is free when the template and the data appear, the EPU template is executed. If the EPU is still processing a previous instruction, the  $\overline{PAUSE}$  line can be activated to halt further execution of CPU instructions until EPU execution is complete. After the execution of the template is complete, the EPU deactivates the  $\overline{PAUSE}$  line and CPU instruction execution continues.

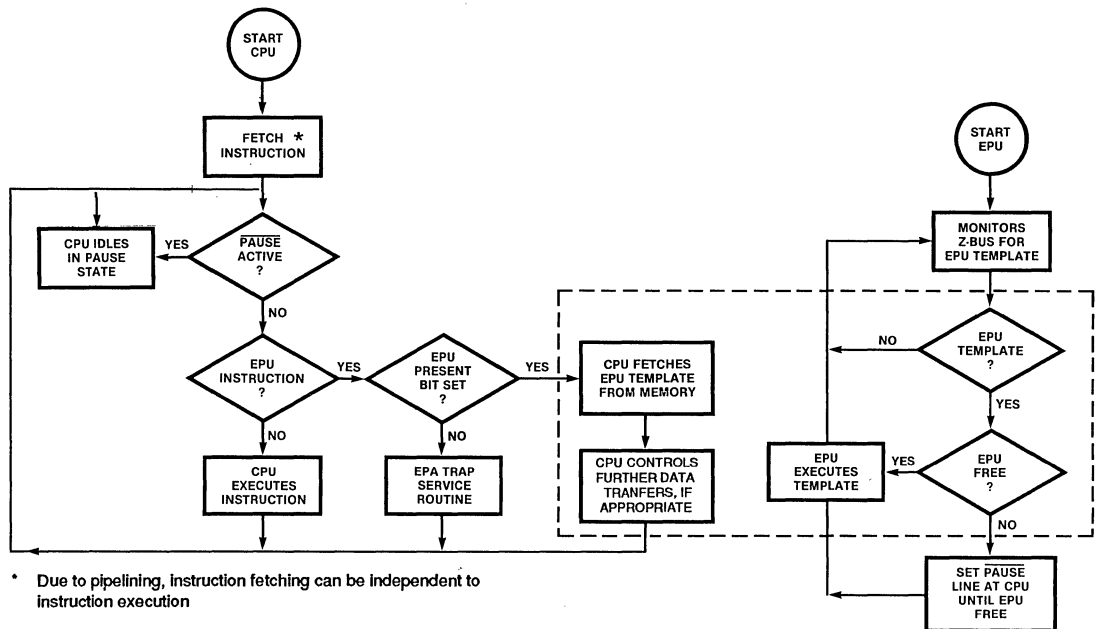


Figure 16. EPA and Z280 MPU Instruction Execution.

## MEMORY MANAGEMENT

### Features

- On-chip dynamic address translation
- Permits addressing of 16M bytes of physical memory
- Separate translation facilities for user and system modes
- Permits instructions and data to reside in separate memory areas.
- Write protection for individual pages of memory
- Aborts CPU on access violation to support virtual memory

### General Description

The Z280 microprocessor contains an on-chip Memory Management Unit (MMU), which translates logical addresses into physical addresses. This allows access to more than 64K bytes of physical memory and provides memory protection features typical of those found on large systems. With the MMU, the CPU can access up to 16M bytes of physical memory. The MMU features a sophisticated trapping mechanism that generates page faults on error conditions. Instructions that are aborted by a

page fault can be restarted in a manner compatible with virtual memory system requirements. On reset, the MMU features are not enabled, thus permitting logical addresses to pass to the physical memory untranslated.

The physical address space is expanded by dividing the 64K byte logical address space (the space manipulated by the program) into pages. The pages are then mapped (translated) into the larger physical address space of the Z280 microprocessor. The mapping process makes the user software addresses independent of the physical memory, so the user is freed from specifying where information is actually stored in physical memory. The actual size of the page depends on whether the program/data separation mode is enabled—if it is enabled, each page is 8K bytes in length, and if it is not enabled, the page length is 4K bytes. With the page mapping technique, 16-bit logical addresses can be translated into 24-bit physical addresses. Address translation can occur both in system and in user mode, with separate translation facilities available to each mode. The MMU further allows instruction references to be separated from data references, which enables programs of up to 64K bytes in length to manipulate up to 64K bytes of data without operating system intervention.

## MMU Architecture

The Z280 MMU consists of two sets of sixteen Page Descriptor registers (Figure 17) that are used to translate addresses, a 16-bit control register that governs the translation facilities, a Page Descriptor Register Pointer, an I/O write-only port that can be used to invalidate sets of page descriptors, and two I/O ports for accesses to the Page Descriptor registers. One set of Page Descriptor registers is dedicated to the system mode of operation and the other set is dedicated to the user mode of operation.

While an address is being translated, attributes associated with the logical page containing that location are checked. The correct logical page is determined by the CPU mode (user or system), address space (program/data), and the four most significant bits of the logical address. Pages can be write-protected to prevent them from being modified by the executing task and can also be marked as non-cacheable to prevent information from being copied into the cache for later reference. The latter capability is useful in multiprocessor systems, to ensure that the processor always accesses the most current version of information being shared among multiple devices. The MMU also maintains a bit for each page that indicates if the page has been modified.

Each Page Descriptor register contains a Valid bit, which indicates that the descriptor contains valid information. Any attempt by the MMU to translate an address using an invalid descriptor generates a page fault. Valid bits for groups of Page Descriptor registers can be reset by writing to an MMU control port.

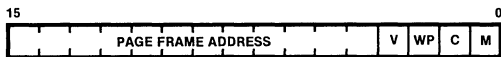


Figure 17. Page Descriptor Register

For each mode of CPU operation, the MMU can be configured to separate instruction fetches from data fetches, and thus separate the program address space from the data address space. When the program/data separation mode is in effect, the sixteen Page Descriptor registers for the current CPU mode of operation (user or system) are partitioned into two sets, one for instruction fetches and one for data fetches. An instruction fetch or data access using the Program Counter Relative addressing mode is translated by the MMU registers associated with the program address space; data accesses using other addressing modes and accesses to the Interrupt Vector Table in interrupt mode 2 use the MMU registers associated with the data address space. In this mode of MMU operation, the page size is 8192 bytes. There are two control bits in the MMU Master Control register that independently specify whether the user and system modes of MPU operation have separate program and data address spaces.

Each 16-bit Page Descriptor register consists of a 4-bit attribute field and a 12-bit page frame address field. The attribute field consists of the least significant bits of the descriptor and contains four control and status bits, listed below.

*Modified (M)*. This bit is automatically set whenever a write is successfully performed to a logical address in this page; it can be cleared to 0 only by a software routine that loads the Page Descriptor register. If the Valid bit is 0, the contents of this bit are undefined.

*Cacheable (C)*. While this bit is set to 1, information fetched from this page can be placed in the cache. While this bit is cleared to 0, the cache control mechanism is inhibited from retaining a copy of the information.

*Write-Protect (WP)*. While this bit is set to 1, CPU writes to logical addresses in this page cause a page fault to be generated and prevent a write operation from occurring. While this bit is cleared to 0, all valid accesses are permitted.

*Valid (V)*. While this bit is set to 1, the descriptor contains valid information. While this bit is cleared to 0, all CPU accesses to logical addresses in this page cause a page fault to be generated.

## MMU Control Registers and I/O Ports

MMU operation is controlled by one control register and four dedicated I/O ports. The MMU Master Control register (Figure 18) determines the program/data address space separation in effect in both user and system modes and whether logical addresses generated in user and system mode will be translated by the MMU. Page Descriptor registers are accessed indirectly through the register address contained in the Page Descriptor Register Pointer. The Descriptor Select Port is used to access the Page Descriptor register that is pointed to by the Page Descriptor Register Pointer. After this access the Page Descriptor Register Pointer is left unchanged. The Block Move I/O Port is used to move blocks of words between the Page Descriptor registers and memory; reads or writes to this I/O port access data pointed to by the Page Descriptor Register Pointer, then increment the pointer by one. The Invalidation I/O Port is used to invalidate blocks of Page Descriptor registers; writes to this port cause the Valid bits in selected blocks of Page Descriptor registers to be cleared to 0, which indicates that the descriptors no longer contain valid information.

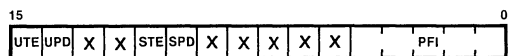


Figure 18. MMU Master Control Register

**MMU Master Control Register.** The MMU Master Control register (I/O address location FFxxF0) controls the operation of the MMU. This register contains four control bits; all other bits in this register must be cleared to 0. The four control bits of the MMU Master Control register are described below.

**Page Fault Identifier (PF).** This 5-bit field latches information that indicates which Page Descriptor register was being accessed when the access violation was detected.

**System Mode Program/Data Separation Enable (SPD).** While this bit is set to 1, instruction fetches and data accesses via the PC Relative addressing mode use the system mode Page Descriptor registers 8-15, and data references that do not use the PC Relative addressing mode use the system mode Page Descriptor registers 0-7. While this bit is cleared to 0, system mode Page Descriptor registers 0-15 are used to translate instruction and data references.

**System Mode Translate Enable (STE).** While this bit is set to 1, logical addresses generated in the system mode of operation are translated. While this bit is cleared to 0, addresses are passed through the MMU extended with zeros in the most significant bits and no attribute checking or modified bit setting is performed.

**User Mode Program/Data Space Separation Enable (UPD).** While this bit is set to 1, instruction fetches and data accesses via the PC Relative addressing mode use the user mode Page Descriptor registers 8-15, and data references that do not use the PC Relative addressing mode use the user mode Page Descriptor registers 0-7. While this bit is cleared to 0, user mode Page Descriptor registers 0-15 are used to translate instruction and data references.

**User Mode Translated Enable (UTE).** While this bit is set to 1, logical addresses generated in the user mode of operation are translated. While this bit is cleared to 0, addresses are passed through the MMU extended with zeros in the most significant bits and no attribute checking or modified bit setting is performed.

**Page Descriptor Register Pointer.** Moves of data into and out of the MMU Page Descriptor registers use the Page Descriptor Register Pointer, which is at I/O address location FFxxF1. This 8-bit register contains the address of one of the Page Descriptor registers. When a word I/O instruction accesses I/O address FFxxF5 (Descriptor Select Port), this register is used to access a Page Descriptor register. When a word I/O instruction accesses I/O address FFxxF4 (Block Move I/O Port), this register is also used to access a Page Descriptor register, but after the access, this register is automatically incremented by one.

**Descriptor Select Port.** Moves of one word of data into and out of a Page Descriptor register are accomplished by writing and reading words to or from this dedicated I/O port at location FFxxF5. Any word I/O instruction can be used to access a Page Descriptor register via this port, provided that the Page Descriptor Register Pointer is properly initialized.

**Block Move I/O Port.** Block moves of data into and out of the Page Descriptor registers are accomplished by writing and reading words to or from this dedicated I/O port at location FFxxF4. Any word I/O instruction can be used to access Page Descriptor registers via this port, provided that the Page Descriptor Register Pointer is properly initialized.

**Invalidation I/O Port.** Valid bits can be cleared (i.e., the Page Descriptor registers invalidated) by writing to this dedicated 8-bit port (Table 4), which is at I/O address location FFxxF2. Individual Valid bits can subsequently be set by software writing to the Page Descriptor registers. Reading this I/O port returns unpredictable data.

**Table 4. Invalidation Port Table**

Encoding	Registers Invalid
01 <sub>H</sub>	System Page Descriptor Registers 0-7
02 <sub>H</sub>	System Page Descriptor Registers 8-15
03 <sub>H</sub>	System Page Descriptor Registers 0-15
04 <sub>H</sub>	User Page Descriptor Registers 0-7
08 <sub>H</sub>	User Page Descriptor Registers 8-15
0C <sub>H</sub>	User Page Descriptor Registers 0-15

### Translation Mechanism

**Address Translation.** Address translation is illustrated in Figure 19. While the Program/Data Space Separation bit is cleared to 0, the 16-bit logical address is divided into two fields, a 4-bit index field used to select one of 16 Page Descriptor registers and a 12-bit offset field that forms the lower 12 bits of the physical address. The physical address is composed of the 12-bit page frame address (bits 4-15) supplied by the selected Page Descriptor register and the 12-bit offset supplied by the logical address.

While the Program/Data Space Separation bit is set to 1, the logical address is divided into a 3-bit index field and a 13-bit offset field. The Page Descriptor register consists of an 11-bit Page Frame Address field (bits 5-15, with bit 4 = 0). The physical address is a result of concatenating the page frame address and the logical offset. The Page Descriptor register is chosen by a 4-bit index field, which consists of a Program/Data Address bit from the CPU and the three Index bits from the logical address.

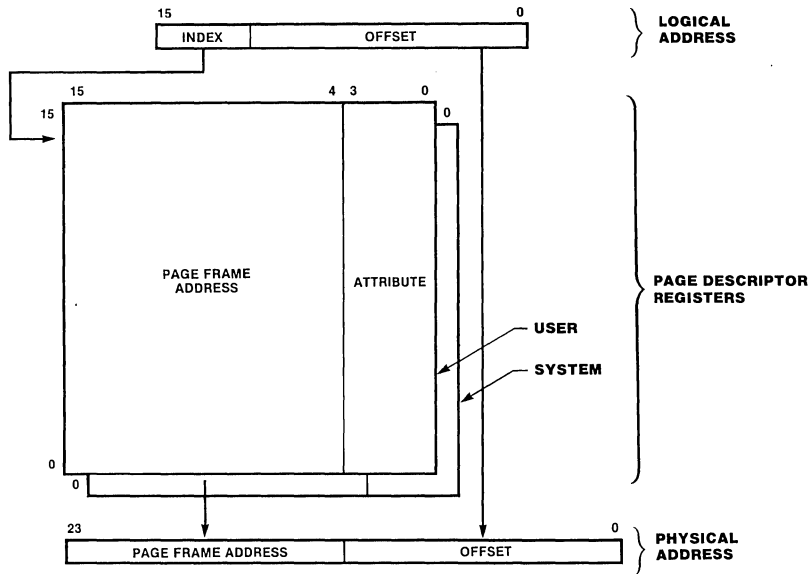


Figure 19. Address Translation

## ON-CHIP MEMORY

### Features

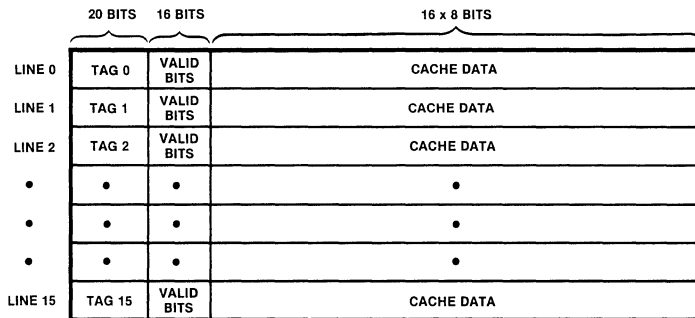
- 256-byte local memory
- Configurable as high-speed associative cache
- Programmable to cache instructions, data, or both
- Permits faster execution by minimizing external bus accesses
- Operation is transparent to user
- Configurable as local RAM with user-definable addresses

The Z280 MPU has 256 bytes of on-chip memory, which can be dedicated to memory locations programmed by the

system or used as a cache for instructions or data. Its mode of use (dedicated memory or cache) is programmable; on reset it is automatically enabled for use as a cache for instructions only.

### On-Chip Memory Architecture

The on-chip memory is organized as 16 lines of 16 bytes each. Each line can hold a copy of 16 consecutive bytes in physical memory locations whose 20 most significant bits of physical address are identical. Each byte in the cache has an associated Valid bit that indicates whether the cache holds a valid copy of the memory contents at the associated physical memory location. Figure 20 illustrates the cache organization.



Tag n = the 20 Address bits associated with line n  
 Valid bits = 16 bits that indicate which bytes in the cache line contain valid data  
 Cache data = 16 bytes

Figure 20. Cache Organization

The on-chip memory has two modes of operation. If the Memory/Cache bit in the Cache Control register is set to 1, then the 256 bytes of on-chip memory are treated as physical memory locations. Memory accesses to addresses covered by the on-chip memory do not generate bus transactions on the external bus and hence the accesses are faster. In this mode, the Valid bits are ignored.

If the Memory/Cache bit is cleared to 0, then the 256 bytes of on-chip memory are treated as a cache memory. The lines are allocated using a least-recently used (LRU) algorithm. When a cache "miss" on a read occurs (and the MMU does not assert cache inhibit), the line in the cache that has been least recently accessed is selected to hold the newly read data. All bytes in the selected line are marked invalid except for the bytes containing the newly accessed data. On a cache miss, one or two bytes, depending on the bus size, are fetched from main memory. Except for burst mode instruction fetches, the cache does not pre-fetch beyond the currently-requested address. A cache miss on a data write does not cause a line to be allocated to the memory location accessed.

The cache can hold both instructions and data. Two control bits in the Cache Control register can be separately set to enable the cache to hold instructions and to hold data. If the cache contains data, writes to data at locations contained in the cache also cause external bus transactions to update the appropriate memory location.

Both the CPU and the on-chip DMAs access the cache. For the CPU, if the MMU is enabled, the access can be either cacheable or non-cacheable, depending on the value of the Cacheable bit in the Page Descriptor register used to translate the logical address. If the MMU is not enabled, all memory transactions are considered to be cacheable. Two bits in the Cache Control register, the Cache Instructions Disable bit and the Cache Data Disable bit, further determine the operation of the cache for various situations. These bits enable the cache for instructions and for data.

When the on-chip memory is used as fixed memory locations, neither the Cache Instruction Disable or Cache Data Disable bits are used, and no distinction is made as to whether the CPU is accessing data or instructions.

In general, when devices such as on-chip DMAs transfer data to the memory, the cache data is modified if the write is to a valid location in the cache but the LRU mechanism is

unaffected. Also, for the EPU to memory transfer, if the cache contains valid locations that are updated by an EPU transaction, the on-chip cache is also updated.

**Cache Control Register.** The operation of the on-chip memory is controlled by an 8-bit Cache Control register (Figure 21) that is accessed using a load control instruction. This register contains five control bits.



**Figure 21. Cache Control Register**

The bits in this register are:

**High Memory Burst Capability (HMB).** This 1-bit field specifies whether a memory burst transaction occurs when the MMU is enabled and there is a 1 in bit 15 of the selected Page Descriptor register (0 = burst mode not supported, 1 = burst mode supported).

**Low Memory Burst Capability (LMB).** This 1-bit field specifies whether a memory burst transaction occurs when the MMU is disabled or when the MMU is enabled and there is a 0 in bit 15 of the selected Page Descriptor register (0 = burst mode not supported, 1 = burst mode supported).

**Cache Data Disable (D).** While this bit is cleared to 0, data fetches are copied into the cache if the M/C bit = 0 (cache mode). If M/C = 1, the state of this bit is ignored.

**Cache Instructions Disable (I).** While this bit is cleared to 0, instruction fetches are copied into the cache when the M/C bit = 0 (cache mode). When M/C = 1, the state of this bit is ignored.

**Memory/Cache (M/C).** While this bit is set to 1, the on-chip memory is to be accessed as physical memory; while it is cleared to 0, the memory is accessed associatively as a cache.

If the on-chip memory is to be used as fixed memory locations, the user can programmably select the ranges of memory addresses for which the on-chip memory responds.

## CLOCK OSCILLATOR

The Z280 MPU has an on-chip clock oscillator/generator that can be connected to a fundamental, parallel-resonant crystal or any suitable clock source. The bus timing clock generated from the on-chip oscillator is output for use by the rest of the system.

---

## REFRESH

The Z280 MPU has an internal mechanism for refreshing dynamic memory. This mechanism can be activated by setting the Refresh Enable bit in the Refresh Rate register to 1. Memory refresh is performed periodically at a rate specified by the Refresh Rate register. Refresh transactions are identical to memory transactions except that different status signals are used and no data is transferred. They can be inserted immediately after the last clock cycle of any bus transaction, including an internal operation.

The refresh transaction is generated as soon as possible after the refresh period has elapsed (generally after the last clock cycle of the current bus transaction). If the MPU receives an interrupt request, the refresh operation is performed first. When the Z280 MPU does not have control of the bus or is in the Wait state, internal circuitry records the number of refresh periods that have elapsed and refresh cycles cannot be generated. When the MPU regains control of the bus or the WAIT input signal is deactivated and the bus transaction completes, the refresh mechanism immediately issues the skipped refresh cycles. The internal circuitry can record up to 256 such skipped refresh operations.

A 10-bit refresh address is generated for each refresh operation with the refresh address being incremented by two between refreshes for 16-bit data bus and by one for 8-bit data bus.

On reset, the Refresh Rate register contains 88<sub>H</sub>, refresh is enabled, the rate is 32 processor clock cycles, and the refresh address is not affected.

The Refresh mechanism is controlled by an 8-bit control register, described below.

### Refresh Rate Register

This 8-bit register (Figure 22) enables the refresh mechanism and specifies the frequency of refresh transactions.

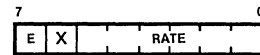


Figure 22. Refresh Rate Register

The fields in this register are:

*Refresh (Rate).* This field indicates in processor clock cycles the rate at which refresh transactions are to be generated; a value of  $n$  in this field indicates a refresh period of  $4n$ , with  $\text{Rate} = 0$  indicating 256 clock cycles.

*Refresh Enable (E).* When this 1-bit field is set to 1, the refresh mechanism is enabled.

---

## UART

The Z280 UART transmits and receives serial data using any common asynchronous data-communication protocol.

Transmission and reception can be performed independently with five, six, seven, or eight bits per character, plus optional even or odd parity. The transmitter can supply one or two stop bits and can provide a break output at any time. Reception is protected from spikes by a "transient spike-rejection" mechanism that checks the signal one-half a bit time after a Low level is detected on the receiver data input; if the Low does not persist—as in the case of a transient—the character assembly process is not started. Framing errors and overruns are detected and buffered with the partial character on which they occur. Furthermore, a built-in checking process avoids interpreting a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit is begun.

The UART uses the same clock frequency for both the transmitter and the receiver. The input for the UART clocking circuitry is derived from counter/timer 1, either from its external input line for an external clock or from the counter/timer output for a bit rate generated from the internal processor clock. The UART input clock is further scaled by 1, 16, 32, or 64 for clocking the transmitter and receiver.

Two of the DMA channels can be used independently to move characters between memory and the transmitter or receiver without CPU intervention. Both the transmitter and receiver can interrupt the CPU for processor assistance.

The UART uses two external pins, Transmit and Receive. Data that is to be transmitted is placed serially on the Transmit pin and data that is to be received is read in from the Receive pin.

### Asynchronous Transmission

The Transmitter Data Output line is held High (marking) when the transmitter has no data to send. Under program control, the Send Break command can be issued to hold the Data Output line Low (spacing) until the command is cleared.

The UART automatically adds the start bit, the programmed parity bit (odd, even, or no parity), and the programmed number of stop bits to the data character to be transmitted. When the character is five, six, or seven bits, the unused most significant bits in the Transmitter Data register are automatically ignored by the UART.

Serial data is shifted from the transmitter at a rate equal to 1, 1/16th, 1/32nd or 1/64th of the clock rate supplied to the transmitter clock input. Serial data is shifted out on the falling edge of the clock input.



## Asynchronous Reception

An asynchronous receive operation begins when the Receive Enable bit in the Receiver Control/Status register is set to 1. A Low (spacing) condition on the Receive input line indicates a start bit. If this Low persists for at least one-half of a bit time, the start bit is assumed to be valid and the data input is then sampled at mid-bit time until the entire character is assembled. This method of detecting a start bit improves error rejection when noise spikes exist on an otherwise marking line. If the  $\times 1$  clock mode is selected, bit synchronization must be accomplished externally; received data is sampled on the rising edge of the clock.

Received characters are read from the Receive Data register. If parity is enabled, the parity bit is assembled as part of the character and is not removed from the assembled character for character lengths other than 8 bits. If the resulting character is still less than 8 bits, 1s are appended in the unused high-order bit positions.

Since the receiver is buffered by one 8-bit register in addition to the receiver shift register, the CPU has adequate time to service an interrupt and to accept the data character assembled by the UART. The receiver also has a buffer that stores error flags for each data character in the receive buffer. These error flags are loaded at the same time as the data character.

After a character is received, it is checked for the following error conditions:

- Parity Error: when the parity bit of the character does not match the programmed parity.
- Framing Error: if the character is assembled without any stop bits (i.e., a Low level is detected for a stop bit).
- Receiver Overrun Error: if the CPU fails to read a data character when more than one character has been received.

Since the Parity Error and Receiver Overrun Error flags are latched, the error status that is read reflects an error in the current character in the Receiver Data register plus any Parity or Overrun Errors detected since the last write to the Receiver Control/Status register. To keep correspondence between the state of the error buffers and the contents of the receiver data buffers, the Receiver Control/Status register must be read before the data.

## Polled Operation

In a polled environment, the Receive Character Available bit in the Receiver Control/Status register must be monitored so the CPU can know when to read a character. This bit is automatically cleared when the Receiver Data register is read. To prevent overwriting data in polled operations, the transmitter buffer status must be checked before writing into the transmitter. The Transmit Buffer Empty bit in the Transmitter Control/Status register is set to 1 whenever the transmit buffer is empty.

## UART Control and Status Registers

The UART operation is controlled by three control and status registers. The UART configuration register specifies the character size, parity, clock source, scaling, and loop-back enable. Both the transmitter and the receiver have their own control/status register.

**UART Configuration Register.** This 8-bit register (Figure 23) contains control information for both the transmitter and receiver.

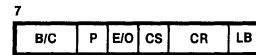


Figure 23. UART Configuration Register

The control fields for this register are:

**Loopback Enable (LB).** The UART is capable of local loopback. In this mode the internal transmit data line is tied to the internal receiver line and the external receiver input is ignored. If this bit is set to 1, loop back mode is enabled.

**Clock Rate (CR).** These two bits specify the multiplier between the clock and data rates (00 = data rate  $\times 1$ , 01 = data rate  $\times 16$ , 10 = data rate  $\times 32$ , 11 = data rate  $\times 64$ ). The same rate is used for both the receiver and transmitter. If the  $\times 1$  clock rate is selected, bit synchronization must be accomplished externally.

**Clock Select (CS).** This bit specifies the clock input for the UART. If the bit is set to 1, the counter/timer 1 output pulse is used for bit-rate generation; if the bit is cleared to 0, the input line to counter/timer 1 provides the clock from an external source.

**Parity Even/Odd (E/O).** If parity is specified, this bit determines whether it is sent and checked as even or odd (1 = even).

**Parity (P).** If this bit is set to 1, an additional bit position (in addition to those specified in the bits/character control field) is added to transmitted data and is expected in received data. In the Receiver, the parity bit received is transferred to the CPU as a part of the character, unless eight bits/character is selected.

**Bits/Character (B/C).** Together, these two bits determine the number of bits to form a character. If these bits are changed during the time that a character is being assembled, the results are unpredictable (00 = 5 bits/character, 01 = 6 bits/character, 10 = 7 bits/character, 11 = 8 bits/character).

**Transmitter Control/Status Register.** This 8-bit register (Figure 24) specifies the operation of the transmitter.

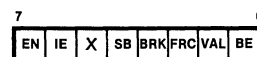


Figure 24. Transmitter Control/Status Register

The control bits for this register are:

**Transmitter Buffer Empty (BE).** This bit is automatically set to 1 whenever the transmitter buffer becomes empty and cleared to 0 when a character is loaded into the transmit buffer. This bit is in the set condition after a reset. This bit is controlled by the UART control circuitry; it can be read by an I/O read but cannot be set to 1 or cleared to 0 by an I/O write.

**Value (VAL).** This bit determines the value of the bits transmitted while the FRC bit is 1 and dummy characters are loaded into the transmitter buffer. When this bit is 1, a mark character (all 1s) is sent; when this bit is 0, a break character (all 0s) is sent.

**Force Character (FRC).** When this bit is set to 1, any character loaded into the transmitter buffer causes the transmitter output to be held High or Low (as indicated by the VAL bit) for the length of time required to transmit a character. This allows a program to generate a marking signal or a break of multiple-character duration simply by setting this bit to 1, setting the VAL bit to 1 or 0, and loading the appropriate number of dummy characters into the transmitter buffer.

**Send Break (BRK).** When set to 1, this bit immediately forces the transmitter output to the spacing condition, regardless of any data being transmitted. When this bit is cleared to 0, the transmitter returns to marking.

**Stop Bits (SB).** This bit determines the number of stop bits added to each asynchronous character sent. The receiver always checks for one stop bit. If this bit is set to 1, two stop bits are automatically appended to the character sent; if this bit is cleared to 0, only one stop bit is appended.

**Transmitter Interrupt Enable (IE).** When this bit is set to 1, interrupt requests are generated whenever the transmitter buffer becomes empty; when this bit is cleared to 0, no requests are made.

**Transmitter Enable (EN).** While this bit is cleared to 0, data is not transmitted and the transmitter output is held marking. Data characters in the process of being transmitted are completely sent if this bit is cleared to 0 after transmission has started.

**Receiver Control/Status Register.** This 8-bit register (Figure 25) specifies the operation of the receiver. The control bits are described below.

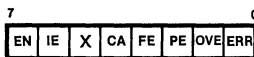


Figure 25. Receiver Control/Status Register

**Receiver Error (ERR).** This bit is the logical OR of the PE, OVE, and FE bits.

**Framing Error (FE).** This bit is automatically set to 1 for the received character in which the framing error occurred. Detection of a framing error adds an additional one-half of a bit time to the character time so the framing error is not interpreted as a new start bit.

**Parity Error (PE).** When parity is enabled, this bit is automatically set to 1 for those characters whose parity does not match the programmed sense (even/odd). This bit is latched, so once an error occurs, it remains set until it is cleared by software.

**Receiver Overrun Error (OVE).** This bit is automatically set to 1 to indicate that more than two characters have been received without a read from the CPU (or DMA). Only the most recently received character is flagged with this error, but when this character is read, the error condition is latched until cleared by software.

**Receiver Character Available (CA).** This bit is automatically set to 1 when at least one character is available in the receive buffer; it is automatically cleared to 0 when the Receiver Data register is read. This bit is controlled by the UART control circuitry; it can be read by an I/O read but cannot be set or cleared by an I/O write.

**Receiver Interrupt Enable (IE).** While this bit is set to 1, interrupt requests are generated whenever the receiver detects an error or the receiver has a character available.

**Receiver Enable (EN).** When this bit is set to 1, receiver operations begin. This bit should be set only after the parameters in the UART Configuration register are set.

### UART Bootstrapping Option

The Z280 CPU supports an automatic initialization of memory via the UART after a reset operation. This system bootstrapping capability permits ROMless system configurations: the memory can be initialized by a serial link before the Z280 CPU fetches information from memory after the reset.

On the rising edge of Reset, the AD lines are sensed if  $\overline{\text{WAIT}}$  is asserted; if AD<sub>6</sub> is being driven High, the Z280 CPU automatically enters a Halt state. The UART is also automatically initialized to receive 8-bit character data with odd parity at a  $\times 16$  clock rate. An external clock source is assumed. A minimum of 15 processor clock cycles must elapse before the transmission can begin.

During the bootstrapping operation, DMA Channel 0 is used to transfer received characters into the memory. This channel is initialized as follows:

**Transaction Descriptor register**—IE, EPS, and TC cleared, ST-byte transfer, BRP-continuous, TYPE-flowthrough, DAD-Auto-increment memory address

**DMA Master Control register**—DOR and EOP set

**Count register**—0100 (256 bytes to be transferred)

**Destination Address register**—000000 (starting memory address = 0)

**Source Address register**—undefined (not used when DMA0 is linked to UART)

---

Characters received are placed in memory starting at physical memory location zero. If an error occurs, the Z280 CPU drives the Transmitter Output line Low. External circuitry monitoring this line can use this signal to cause the transmitting device to begin the initialization procedure again, starting with a reset and AD<sub>6</sub> asserted on the rising edge of Reset.

---

After 256 bytes of data have been transferred, the Z280 CPU automatically begins execution by fetching the first instruction from memory location 0.

---

## DMA CHANNELS

The Z280 MPU has four on-chip Direct Memory Access (DMA) channels to provide high bandwidth data transmission capabilities. There are two types of DMA channels; two support flyby transactions and the other two do not. The two types of DMA channels otherwise have identical capabilities, although they have different priorities with respect to interrupt requests and bus requests.

Each DMA channel is a powerful and versatile device for controlling and processing transfers of data. Its basic function of managing CPU-independent transfers between two ports is augmented by an array of features requiring little or no external logic in systems using an 8- or 16-bit data bus.

Transfers can be performed between any two ports (source and destination), including memory-to-I/O, I/O-to-memory, memory-to-memory, and I/O-to-I/O. Except for flyby, two port addresses are automatically generated for each transaction and can be either fixed or incrementing/decrementing.

During a transfer, a DMA channel assumes control of the system address and data bus. Data is read from one addressable port and written to the other addressable port, byte-by-byte or word-by-word. The ports can be programmed to be either system main memory or peripheral I/O devices.

For both flyby and flowthrough DMA transactions, if the destination is a memory location that corresponds to an entry in the on-chip memory (either cache or fixed memory location), the on-chip memory is updated to reflect the new contents of the memory location.

Except in flyby mode, two 24-bit addresses are generated by the DMA for every transfer operation, one address for the source port and another for the destination port. Two readable address counters (three bytes each) keep the current address of each port.

The DMA devices use the same memory and I/O timing as the CPU for bus transactions, as indicated by the appropriate bus timing register.

### Modes of Transfer Operation

Each DMA can be programmed to operate in one of three transfer modes:

- *Single Transaction.* Data operations are performed one byte or word at a time.
- *Burst.* Data operations continue until a port's Ready line to the DMA goes inactive.

- *Continuous.* Data operations continue until either the end of the programmed block of data is reached or an end of process has been signaled before the system bus is released.

In all modes, once a byte or word of data is read by the DMA channel, the operation is completed in an orderly fashion, regardless of the state of other signals (including a port's Ready line).

### Pin Descriptions

Each DMA channel has a Ready input line. In addition, two DMA channels have a flyby output line to support high speed data transfers between I/O devices and memory.

The flyby output is asserted by the DMA channel to signal a peripheral device associated with the DMA channel that it should participate in the data transmission during the current flyby bus transaction.

**If Ready is active, the DMA channel requests control of the external system bus to perform the DMA transaction. When the external system bus is available for DMA transfers, the DMA channel with a request pending and the highest priority assumes bus mastership. The priority of DMA channels from highest to lowest is: DMA0, DMA1, DMA2, and DMA3. A DMA channel in burst mode relinquishes bus mastership to a higher priority DMA channel only when its Ready line is deasserted (or EOP is signaled or terminal count is reached). A DMA channel in continuous mode relinquishes bus mastership only when EOP is signaled or terminal count is reached.**

### Priority of On-Chip DMA Channels and External Bus Requesters

The on-chip DMA channels are arranged in a daisy chain with the external Bus Request input line being the "next lower bus requester" on this chain. The on-chip DMAs behave as if they were external bus requestors with respect to acquiring the bus, relinquishing the bus, and priority access to the bus.

### End-of-Process

**If the end-of-process (EOP) capability is enabled, transfers by DMA channels can be prematurely terminated by a Low on Interrupt A line or Interrupt B line during the transfer. This capability is programmed by control bits in the DMA Master Control register. EOP occurs regardless of the**

setting of the Interrupt A Enable bit in the Master Status register. When an EOP is signaled, the EOP Signaled (EPS) bit in the Transaction Descriptor register of the active DMA channel is set to 1 and the Enable bit is cleared to 0. If interrupt requests are enabled (IE = 1 in the Transaction Descriptor register), an interrupt request is generated by the channel that was active when the EOP was signaled. After an EOP has been signaled, the DMA relinquishes the bus within 16 cycles of the last DMA bus transaction.

If the End-Of-Process signal on Interrupt A or B line is still asserted when the CPU is bus master, the signal is interpreted as an interrupt request; thus, both the DMA channel and the external EOP generating device can request interrupts simultaneously. Separate mask bits in the Master Status register enable the CPU to accept interrupts from these two sources.

On a flowthrough transaction, if the EOP signal is received while the information is being read into the Z280 MPU, the transfer is aborted and the data is not written out from the Z280 MPU.

### DMA Linking

The DMA devices can be linked together to achieve DMA transfers to non-contiguous memory locations (linked operation). Bits in the DMA Master Control register allow DMA3 to be linked to DMA1 and DMA2 to be linked to DMA0. If the appropriate bit is set to 1 in the DMA Master Control register, the master DMA (0 or 1) signals its linked DMA each time its transfer is complete (count = 0). This acts as an internal ready input to the linked DMA that reloads the master DMA control registers.

Words are loaded into the master DMA control registers in the following order: Destination Address register (two words), Source Address register (two words), Count (one word), Transfer Descriptor register (one word). After six words have been transferred, the master DMA deasserts its internal ready line and begins the transfer of the next block of data. The master DMA can be programmed to interrupt the CPU on "count equals 0" when the last block transfer is completed by the master DMA (to notify software that the entire sequence of transfers is completed).

When programming linked DMAs, the last word to be programmed must be the master DMA's Transaction Descriptor register. Also, the linked DMA must be programmed before the master DMA's status register is programmed.

**DMA Master Control Register.** This 16-bit register (Figure 26) specifies the general configuration of the four on-chip DMA channels: the linking of the DMA channels, the software ready enables, and EOP enable.

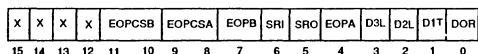


Figure 26. DMA Master Control Register

The fields in this register are:

**DMA0 to Receiver Link (DOR).** When this bit is set to 1, DMA channel 0 is linked to the UART receiver.

**DMA1 to Transmitter Link (D1T).** When this bit is set to 1, DMA channel 1 is linked to the UART transmitter.

**DMA2 Link (D2L).** When this bit is set to 1, DMA channel 2 is linked to DMA channel 0.

**DMA3 Link (D3L).** When this bit is set to 1, DMA channel 3 is linked to DMA channel 1.

**End-of-Process (EOP<sub>A</sub>).** When this bit is set to 1, the INT<sub>A</sub> line is used as an end-of-process signal for the DMA channel defined by the EOPCSA field.

**End-of-Process (EOP<sub>B</sub>).** When this bit is set to 1, the INT<sub>B</sub> input acts as an EOP input for the DMA channel defined by the EOPCSB field.

**Software Ready for DMA0 (SRO).** When this bit is set to 1, DMA channel 0 requests service if enabled.

**Software Ready for DMA1 (SR1).** When this bit is set to 1, DMA channel 1 requests service if enabled.

**End-of-Process Channel Select A (EOPCSA).** This field defines the DMA channel that has INT<sub>A</sub> as its EOP input. This field has no effect if EOP<sub>A</sub> bit (bit 4) is cleared to zero

00	DMA Channel 0
01	DMA Channel 1
02	DMA Channel 2
03	DMA Channel 3

**End-of-Process Channel Select B (EOPCSB).** This field defines the DMA channel that has INT<sub>B</sub> as its EOP input. This field has no effect if EOP<sub>B</sub> bit (bit 7) is cleared to zero.

00	DMA Channel 0
01	DMA Channel 1
02	DMA Channel 2
03	DMA Channel 3

Note that while the EOP<sub>A</sub> and EOP<sub>B</sub> bits are active, INT<sub>A</sub> and INT<sub>B</sub> can still serve as interrupt inputs.

### DMA Channel Control Registers

**Transaction Descriptor Registers.** These four 16-bit registers, one for each channel (Figure 27), describe the type of DMA transfer to be performed and contain control and status information.

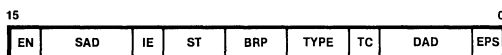


Figure 27. Transaction Descriptor Register

The fields in this register are:

**End-of-Process Signaled (EPS).** This bit is set to 1 automatically when the channel is active and an end-of-process is signaled for this channel as programmed on the Interrupt A or Interrupt B input lines, thus prematurely terminating the transfer.

*Destination Address Descriptor (DAD).* The setting of this 3-bit field indicates the type of location (memory or I/O) and how the address is to be manipulated (incremented, decremented or left unchanged), as shown in Table 5.

**Table 5. SAD and DAD Encodings**

Encoding	Address Modification Operation
000	Auto-increment memory location
001	Auto-decrement memory location
010	Memory address unmodified by transaction
011	Reserved
100	Auto-increment (by 1) I/O location
101	Auto-decrement (by 1) I/O location
110	I/O address unmodified by transaction
111	Reserved

*Transfer Complete (TC).* This bit is set to 1 automatically when the count register has reached zero.

*Transaction Type (Type).* This 2-bit field specifies flyby or flowthrough type of operation (00 = flowthrough, 01 = reserved, 10 = flyby write, 11 = flyby read). In flowthrough mode of operation, two bus transactions occur for each DMA operation—a read from the source followed by a write to the destination. In a flyby operation, only one bus transaction occurs for each DMA operation. In flyby write to memory, the flyby output pin is pulsed instead of an I/O transaction being performed and the contents of the Destination Address register are output to specify the memory location. In flyby read from memory, the flyby output pin is pulsed instead of an I/O transaction being performed and the contents of the Source Address register are output to specify the memory location. Only two DMAs have flyby capability.

*Bus Request Protocol (BRP).* The setting of these two bits indicates the mode of DMA operation (Table 6).

**Table 6. Bus Request Protocol (BRP)**

Encoding	DMA
0 0	Single Transaction
0 1	Burst
1 0	Continuous
1 1	Reserved

*Size of Transfer (ST).* This 2-bit field specifies the size of the entity to be transferred by the DMA channel (Table 7). For word transfers to or from memory locations, the memory address must be even (least significant bit is 0). Long word (32-bit) transfers are supported only in flyby mode, with the cache disabled.

**Table 7. Size of Transaction (ST)**

Encoding	Size of Transfer	Number to Increment/Decrement By
0 0	Byte	1
0 1	16-bit word	2
1 0	32-bit longword	4
1 1	Reserved	

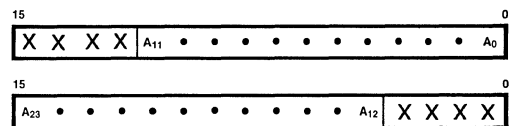
*Interrupt Enable (IE).* When this bit is set to 1, the DMA generates an interrupt request at end of count or end of process. When this bit is 0, no interrupt request is generated.

*Source Address Descriptor (SAD).* The setting of this 3-bit field indicates the type of location (memory or I/O) and how the address is to be manipulated (incremented, decremented or left unchanged), as shown in Table 5.

*DMA Enable (EN).* While this bit is 1, the DMA transfer is enabled.

**Count Register.** This 16-bit register is programmed to contain the number of DMA transfers to be performed. When the contents of the count register reach zero, further requests on the RDY input line are ignored. The DMA channel can be programmed to generate an interrupt when the count register reaches zero.

**Source Address Register and Destination Address Register.** These 24-bit registers contain the 24-bit physical addresses to be used during the DMA transaction. They are not translated by the MMU. In flyby mode, only one of these registers is used to supply the address for the bus transaction as indicated in the Mode field in the Transfer Descriptor register. The format for these registers is shown in Figure 28.



**Figure 28. Source and Destination Address Registers Format**

### Flyby Transaction Timing

The Transaction Type field in the Transaction Descriptor register indicates whether the transaction is a read or a write. For flyby read transactions, the Source Address Descriptor indicates the transaction is a read from memory; for write flyby transactions the Destination Address Descriptor indicates the transaction is a write to memory. Additional wait states can be automatically inserted if programmed in the appropriate timing register. See Figures 29 and 30 for timing diagrams.

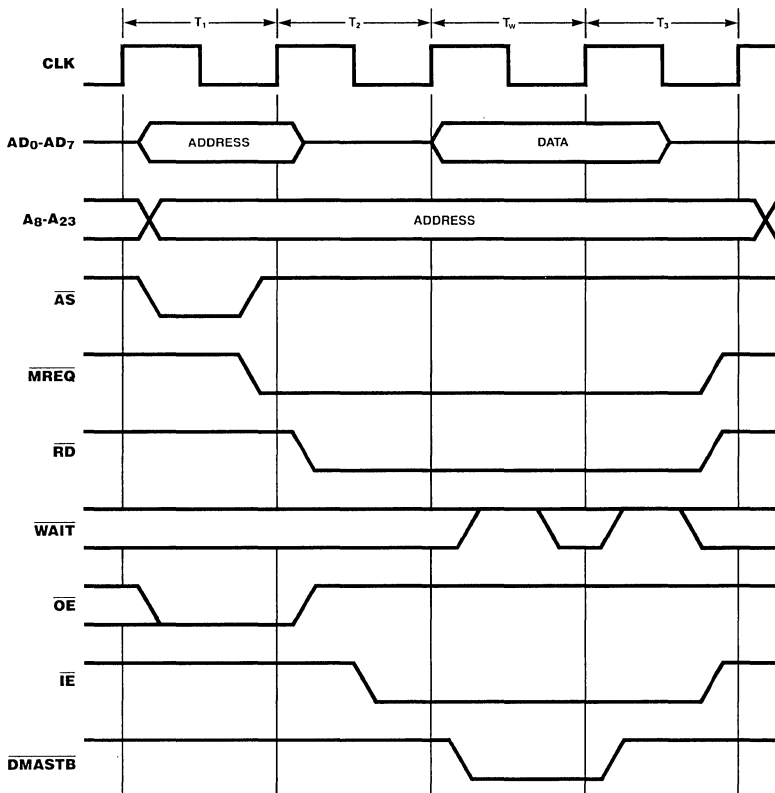


Figure 29a. On-Chip DMA Channel Flyby Memory Read Transaction, Z80 Bus

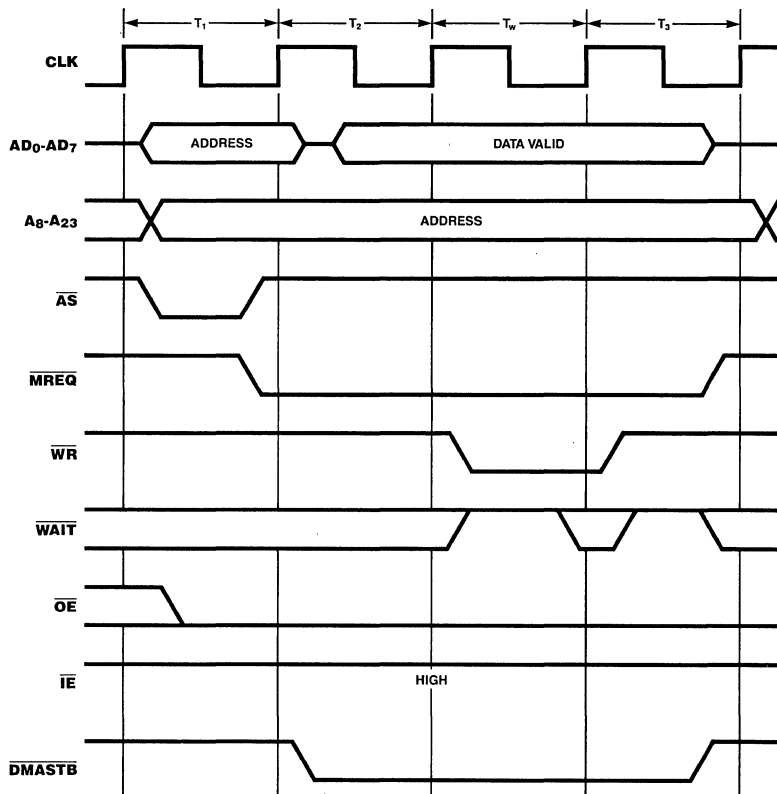


Figure 29b. On-Chip DMA Channel Flyby Memory Write Transaction, Z80 Bus

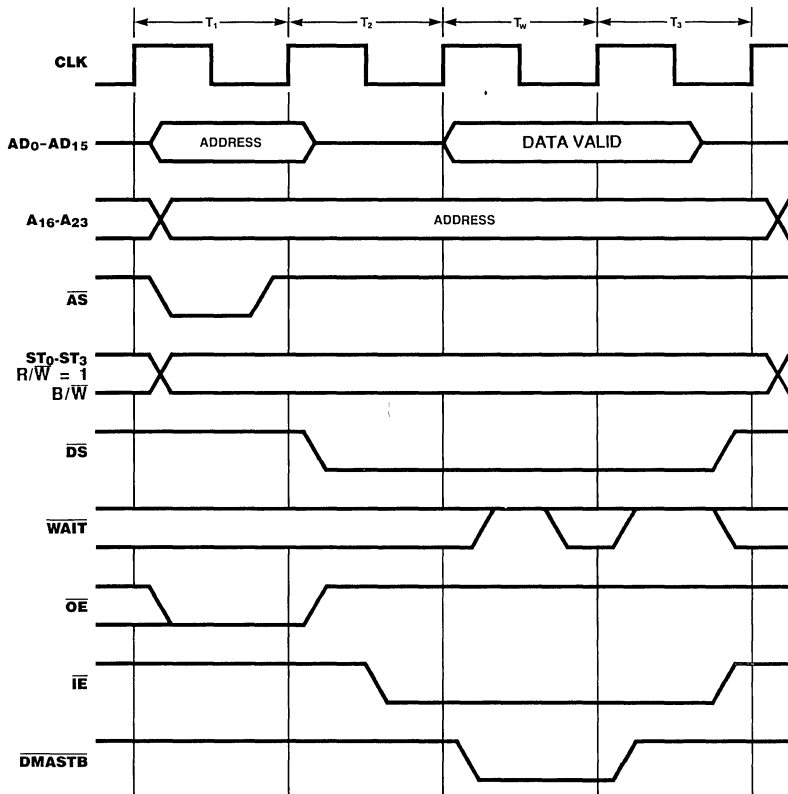


Figure 30a. On-Chip DMA Channel Flyby Memory Read Transaction, Z-BUS



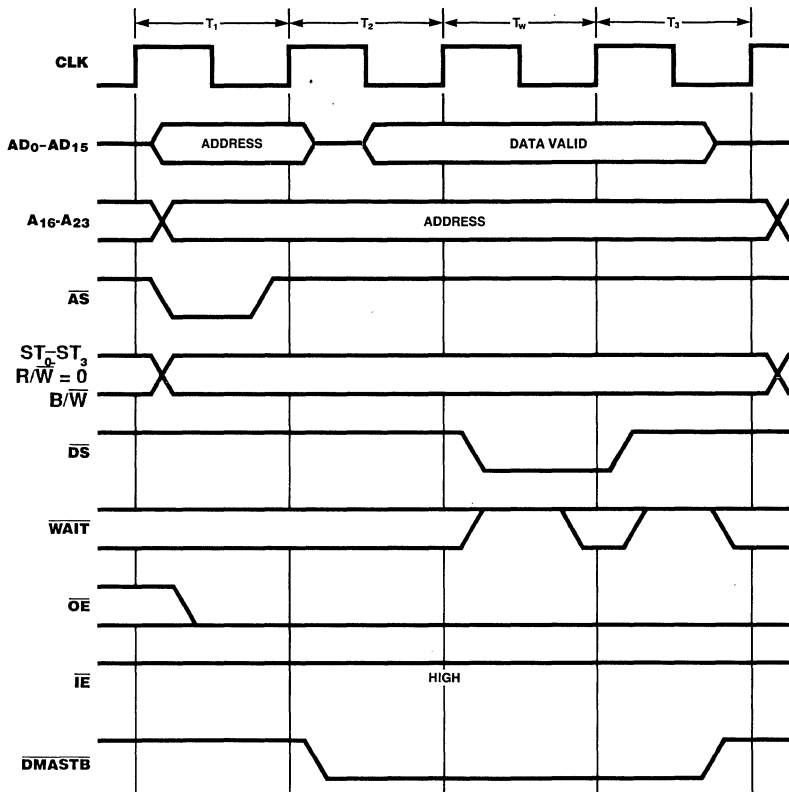


Figure 30b. On-Chip DMA Channel Flyby Memory Write Transaction, Z-BUS

## COUNTER/TIMERS

The Z280 MPU's three counter/timers can be programmed by system software for a broad range of counting and timing applications. The three independently programmable channels satisfy common microcomputer system requirements for event counting, interrupt and interval timing, and general clock generation.

Programming the counter/timers is straightforward: each channel is programmed with four bytes. Once started, the channel counts down, and optionally reloads its time constant automatically and resumes counting. Software timing loops are completely eliminated. Interrupt processing is simplified because each channel uses a unique vector from the Interrupt/Trap Vector Table.

Each channel is individually programmed with three registers: a configuration byte, a control byte, and a

time-constant word. The configuration byte selects the operating mode (counter or timer), enables or disables the channel interrupt, and selects certain other operating parameters. In the timing mode, the CPU processor clock is divided by four for input to the counter/timers. The time-constant word contains a value from 0 to 65,535.

During operation, the individual counter channel counts down from the present time-constant value. In counter mode operation, the counter decrements on each of the input pulses until the count/time output condition is met. Each decrement is synchronized by the scaled internal processor clock. For counts greater than 65,536, two of the counters can be programmably cascaded. When the count/time output condition is reached, the downcounter is automatically reset with the time constant value, if so programmed.

The timer mode determines time intervals without additional logic or software timing loops. Time intervals are generated by dividing the internal processor clock by four and decrementing a presetable downcounter. Thus, the time interval is an integral multiple of the processor clock period, the prescaler value four, and the time constant that is preset in the downcounter. A timer is triggered by setting the software trigger control bit in the Control/Status register or by an external input.

All three channels can generate an external output when the count/time output condition is met. The output is high when the internal presetable downcounter contains all zeros.

Each channel can be programmed to generate an Interrupt Request, which occurs only if the channel has its Interrupt Enable control bit set to 1 by software programming. When the Z280 CPU accepts the interrupt request it automatically vectors through the Interrupt Vector Table.

The three channels of the Z280 MPU are fully prioritized and fit into three different slots in the Z280 internal peripheral daisy-chain interrupt structure. Channel 0 has the highest priority and Channel 2 has the lowest. The channels have separate interrupt enables and the CPU's Master Status register has individual control bits that selectively inhibit interrupts from each channel.

### Modes of Operation

The counter/timer channels have two basic modes of operation: as counters or as timers. As counters they monitor external input lines and record Low to High transitions on these lines. In the timer mode, the processor clock, scaled by four, is used instead of the external input line. The duration of this counting or timing can be either continuous from initial enabling (trigger operation) or only during intervals specified by signals on an input line (gate and gate/trigger operation). The count can be automatically

restarted by programming the Retrigger Enable control bit in the counter/timer's Configuration register.

Each of the three counter/timers has a software gate and trigger facility that extends the hardware capabilities of the counter/timers.

**Counting Operation.** While the appropriate enabling conditions are met, the counter/timer monitors its input line for Low-to-High transitions. When such a transition occurs, the Count/Time register is decremented by 1.

**Timing Operation.** While the appropriate enabling conditions are met, the counter/timer monitors the internal processor clock scaled by four for Low-to-High transitions. When such a transition occurs the Count/Time register is decremented by 1.

**Gate Operation.** A counter/timer can be programmed to count or time only when a gating condition is met. While the counter/timer is enabled and the external gate capability is selected, an external input line is monitored; only while this line is High are the counting or timing operations performed. The software gate facility filters the state of the input line; while the software gate bit in the Command and Status register is cleared to 0, the gating condition is not met regardless of the signals on the gating line. The gate facility is illustrated in Figure 31.

**Trigger Operation.** A counter/timer can be programmed to count or time only after a triggering condition occurs. While the counter/timer is enabled and the external trigger capability is programmed, an external input line is monitored; only after this line makes a Low-to-High transition is a counting or timing operation performed. The software trigger facility causes the triggering condition to be met regardless of the activity of this line. The trigger operation is illustrated in Figure 32.

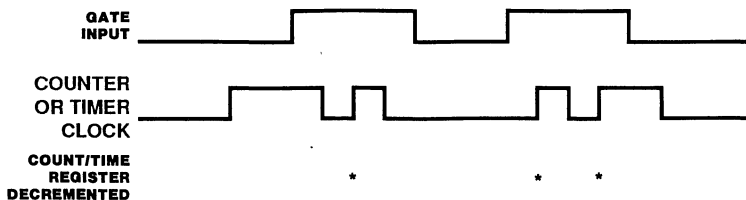


Figure 31. Gate Facility

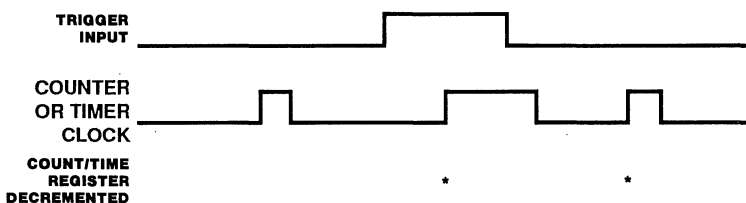


Figure 32. Trigger Operation

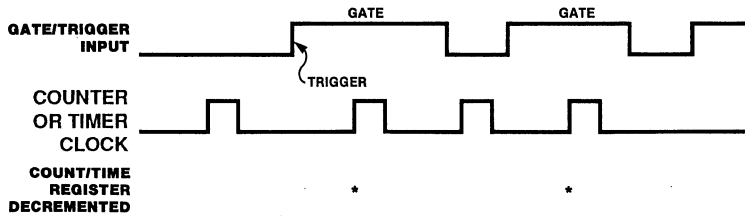


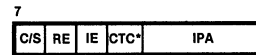
Figure 33. Gate/Trigger Operation

**Gate/Trigger Operation.** One input line can be used for both the gating and the triggering functions. A Low-to-High transition on this line acts as a trigger and subsequent High signals on this line function as gate signals. If non-retriggerable mode is programmed, subsequent Low-to-High transactions do not cause a trigger. Gate/Trigger Operation is shown in Figure 33.

The software gate and trigger mechanism can also be used in this mode of operation. A software gate before a trigger (hardware or software) has no effect on the counter/timer. After a hardware or software trigger, the software gate must be set to 1 for the Count/Time register to be decremented. A software trigger after a hardware or software trigger has no effect unless the Retrigger Enable control bit is set to 1.

### Counter/Timer Control and Status Registers

Each counter/timer has two 8-bit control registers and two 16-bit count registers. The Configuration register and Command/Status register determine the counter/timers' operation, the Counter/Timer Command/Status register provides information about the current operation, the Time Constant register contains the initialization value for the counter/timer, and the Count/Time register contains the current value of the count in progress.



\* Only the CTC bit in Counter/Timer 0 is used.

Figure 34. Counter/Timer Configuration Register

**Counter/Timer Configuration Register.** This 8-bit register (Figure 34) specifies the counter/timer's mode of operation: the pin configuration, whether an interrupt request is generated, and whether the countdown sequence is automatically restarted when the count reaches zero or when a trigger occurs.

The fields in this register are:

**Input Pin Assignments (IPA).** This 4-bit field specifies the functionality of the input lines associated with the counter/timer and whether the counter/timer monitors an external input (counting operation) or uses the scaled internal processor clock (timing operation). The four bits in this field can be associated with enabling output generation (EO), selecting the external signal or internal clock (C/T), enabling the gating facility (G), and enabling the triggering facility (T). The selected options determine the functions associated with each input line associated with the counter/timer, as illustrated in Table 8.

Table 8. Input Pin Functionality

E0	IPA Field			Pin Functionality		Notes
	C/T	G	T	Counter/Timer I/O	Counter/Timer Input	
0	0	0	0	Unused	Unused	Timer
0	0	0	1	Unused	Trigger	Timer
0	0	1	0	Gate	Unused	Timer
0	0	1	1	Gate	Trigger	Timer
0	1	0	0	Unused	Input	Counter
0	1	0	1	Trigger	Input	Counter
0	1	1	0	Gate	Input	Counter
0	1	1	1	Gate/Trigger	Input	Counter
1	0	0	0	Output	Unused	Timer
1	0	0	1	Output	Trigger	Timer
1	0	1	0	Output	Gate	Timer
1	0	1	1	Output	Gate/Trigger	Timer
1	1	0	0	Output	Input	Counter
1	1	0	1	Unused	Unused	Reserved
1	1	1	0	Unused	Unused	Reserved
1	1	1	1	Unused	Unused	Reserved

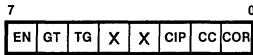
**Counter/Timer Cascade (CTC).** When this bit is set to 1, counter/timers 0 and 1 form a 32-bit counter. When used as a 32-bit counter/timer, the fields in the Configuration register and Command/Status register for Counter/Timer 0 are ignored with the exception of the IE, CTC, EO, CIP, CC, and COR fields. The CTC bits in the Counter/Timer Configuration registers of counter/timers 1 and 2 are never used.

**Interrupt Enable (IE).** While this bit is set to 1, the counter/timer generates an interrupt request when the count/time output condition is met. While this bit is 0, no interrupt request is generated.

**Retrigger Enable (RE).** While this bit is set to 1, the time constant value is automatically loaded into the Count/Time register when a trigger input is received while the counter/timer is counting down. While this bit is 0, no reloading occurs.

**Continuous/Single Cycle (C/S).** While this bit is set to 1, the countdown sequence is automatically restarted when the count reaches zero by loading the time constant value into the Count/Time register. While this bit is 0, no reloading occurs.

**Counter/Timer Command/Status Register.** This 8-bit register (Figure 35) provides software control over the operation of the counter/timer and reflects the current status of the counter/timer's operation. Control bits in this register enable the counter/timer's operation and provide software gate and trigger capabilities. Status bits indicate whether a count is in progress, the count/time output condition has been reached, or the condition has been reached a second time.



**Figure 35. Counter/Timer Command/Status Register**

The fields of this register are:

**Count Overrun (COR).** When this bit is set to 1, the count/time output condition has been reached and the CC bit is set to 1, thus indicating a count overrun condition. While this bit is cleared to 0, the count/time output condition has not been reached with the CC bit set since the time the CC bit was cleared by software. This bit can be read or written (set or cleared) by software I/O instructions.

**Count/Time Output Condition has been Met (CC).** When this bit is set to 1, the Count/Time register has been decremented to zero by the counter/timer control circuitry in single cycle mode or the Count/Time register has been reloaded in continuous mode. When this bit is cleared to 0, the count has not reached the count/time output condition since the bit was cleared by software. This bit can be read or written (set or cleared) by software I/O instructions.

**Count In Progress (CIP).** While this bit is set to 1, the counter/timer is operating and the Count/Time register is non-zero; while this bit is cleared to 0, the counter/timer is

not operating. This bit is controlled by the counter/timer control circuitry; it can be read by an I/O read but cannot be set or cleared by an I/O write instruction.

**Software Trigger (TG).** When this bit is set to 1 (and the trigger operation of the counter/timer is enabled), if the Enable bit is also set to 1, the trigger operation is enabled on the rising edge of the first processor clock period following the setting of this bit from a previously cleared value. That is, if a hardware trigger has not already occurred, the contents of the Time Constant register are loaded into the Count/Time register and the countdown sequence begins. If a hardware trigger has already occurred, then if Retrigger Enable is set to 1, the counter/timer is retriggered; otherwise, setting this bit has no effect. Writing a 1 in this field when the previous value was 1 has no effect on the operation of the counter/timer. When this bit is cleared to 0, this bit has no effect on the operation of the counter/timer.

**Software Gate (GT).** When this bit is set to 1 (and the gate operation of the counter/timer is enabled), if the Enable bit is also set to 1, operation begins on the rising edge of the first processor clock period following the setting of this bit from a previously cleared value. Writing a 1 in this field when the previous value was 1 has no effect on the operation of the counter/timer. When this bit is cleared to 0, the countdown sequence is halted.

**Enable (EN).** While this bit is set to 1, the counter/timer is enabled; operation begins on the rising edge of the first processor clock period following the setting of this bit from a previously cleared value. Reset clears this bit. While this bit is cleared to 0, the value in the Time Constant register is constantly transferred to the Count/Time register. If the Time Constant register is all zeros, the output of the counter/timer is one. Thus, when the counter/timer is not enabled, the counter/timer output in conjunction with the Time Constant register can be used as an I/O port. Writing a 1 in this field when the previous value was 1 has no effect on the operation of the counter/timer. While this bit is 0, the counter/timer performs no operation during the next (and subsequent) processor clock periods.

**Time Constant Register.** This 16-bit register holds the value that is automatically loaded into the Count/Time register when the counter/timer is enabled or in the continuous or retrigger mode when the count reaches zero or the trigger is asserted, respectively. This register can be read or written by I/O instructions.

**Count/Time Register.** This 16-bit register holds the current value of the count or timing in progress. It is automatically loaded from the Time Constant register, and can be read by software using the I/O read instructions.

### Pin Descriptions

The counter/timers have two external input lines associated with them. The I/O lines transfer signals between the counter/timers and external devices. The input lines receive signals from external devices for the counter/timers. The interpretations of the signals on these lines is determined by the Input Pin Assignment field in the Configuration register.

## MULTIPROCESSOR MODE OF OPERATION

### Features

- Allows global memory areas for shared resources
- Global memory addresses are user-specified
- Separate requests for local and global buses
- Requesting mechanism is transparent to user
- Easily interfaces to external arbiters

The Z280 supports various multiprocessor configurations, wherein it is the default bus master of the local bus, and it goes through a defined protocol to access the global bus. To invoke the multiprocessor mode, the Local Address Register contents should be defined, and the MP bit of the Bus Timing and Initialization Register set.

**Pin Functionality** When the Z280 is in the multiprocessor mode, Counter/Timer 0's IO pin is used as the Global Request ( $\overline{\text{GREQ}}$ ) output, and Counter/Timer 0's Input pin is used as the Global Acknowledge ( $\overline{\text{GACK}}$ ) input.

**Local Address Register.** Before an external memory bus transaction is to proceed, the Z280 distinguishes whether a bus transaction uses the local or global bus by comparing the four most significant bit of the physical address (address bits 20 through 23) with a 4-bit Base field in the Local Address register (Figure 36). A mask field in this register specifies which bits are to be compared. If all corresponding address bits match the Base field bits (for those bit positions specified by the mask field), then bus transaction can proceed on the local bus without requesting the global bus; if there is a mismatch in at least one specifies bit position, then the global bus is requested and the bus transaction does not proceed until the global bus acknowledge signal is asserted.

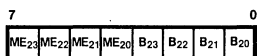


Figure 36. Local Address Register

The bits in the Local Address register are:

**Base ( $B_n$ ).** When  $B_n$  is 1, address bit  $A_n$  must be 1 for a local bus transaction to be performed (unless Match Enable bit  $ME_n$  is 0); when bit  $B_n$  is 0, address bit  $A_n$  must be 0 for a local bus transaction to be performed.

**Match Enable ( $ME_n$ ).** When  $ME_n$  is 1, address bit  $A_n$  is compared to base bit  $B_n$  to determine if the address requires the use of the global bus. When  $ME_n$  is 0, then any values for  $A_n$  and  $B_n$  will produce a match. If each  $ME_n$  is 0, then all bus transactions are performed on the local bus.

### CPU Accesses on the Global Bus

The Z280 is the default local bus master, whether it is in the multi-processor mode or not. It relinquishes the local bus by following a protocol controlled by the  $\overline{\text{BUSREQ}}$  input and  $\overline{\text{BUSACK}}$  output pins. When  $\overline{\text{BUSREQ}}$  is asserted, it is synchronized internally by the CPU. When the CPU is ready to relinquish the local bus, it places all its bus control outputs, including  $\overline{\text{GREQ}}$ , in 3-state, and then drives  $\overline{\text{BUSACK}}$  active. After reset, the CPU acknowledges a request for the local bus before performing any transactions.

In multi-processor mode, the CPU determines if the next external memory transaction should access the global bus. If such is the case, and if the CPU currently is the local bus master, it puts the global address on the address outputs, and the status signals are also made valid, at the beginning of a bus clock cycle.  $\overline{\text{GREQ}}$  is asserted in the second half of the same bus clock cycle. The CPU then samples  $\overline{\text{BUSREQ}}$  and  $\overline{\text{GACK}}$  continuously. Both inputs are synchronized internally by the CPU. The CPU will proceed with the global transaction after it samples that  $\overline{\text{GACK}}$  is asserted, with the absence of  $\overline{\text{BUSREQ}}$ . Once the CPU controls the global bus, it can perform multiple global transactions. It relinquishes the global bus when the next transaction should not be global, when  $\overline{\text{BUSREQ}}$  becomes active, or when  $\overline{\text{GACK}}$  is de-asserted. A global test and set instruction is atomic (global read is followed by global write), and a global memory burst transaction completes its entire sequence of data transfers.

### DMA Accesses on the Global Bus

Each on-chip DMA channel can access the global bus to perform data transfers. The address generated during each DMA-initiated memory transfer is compared with the contents of the Local Address register to determine whether the global bus should be requested. The protocol is identical to the global memory transactions initiated by the CPU.

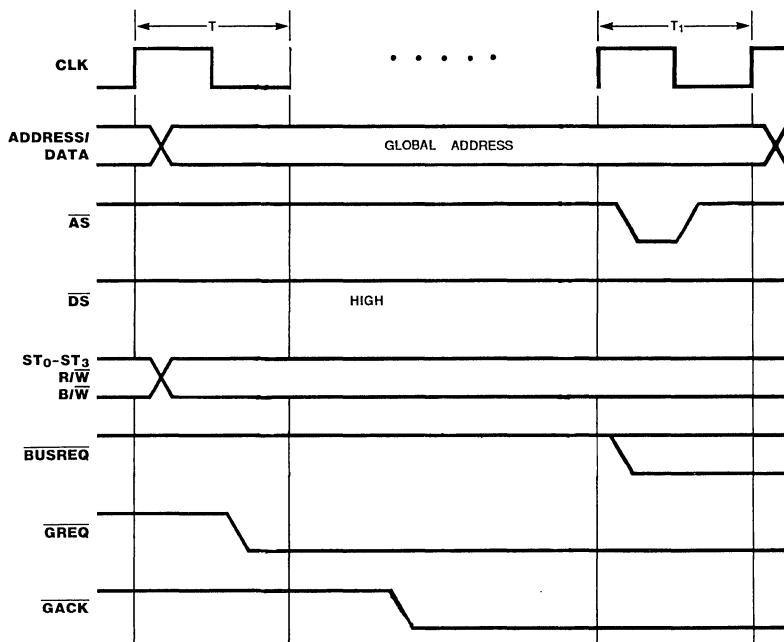


Figure 37. Multiprocessor Mode Timing, Z-Bus Example

## EXTERNAL INTERFACE

The two different external interfaces for the Z280 MPU are the 8-bit Z80 Bus and the 16-bit Z-BUS.

### Z80 Bus External Interface

#### Features

- 8-bit data bus
- Multiplexed address/data lines
- Supports Z80 Family peripherals

#### Pin Descriptions

**A<sub>8</sub>-A<sub>23</sub>.** *Address* (output, active High, 3-state). These address lines carry I/O addresses and memory addresses during bus transactions.

**AD<sub>0</sub>-AD<sub>7</sub>.** *Address/Data* (bidirectional, active High, 3-state). These eight multiplexed Data and Address lines carry I/O addresses, memory addresses, and data during bus transactions.

**AS.** *Address Strobe* (output, active Low, 3-state). The rising edge of AS indicates the beginning of a transaction and shows that the address is valid.

**BUSACK.** *Bus Acknowledge* (output, active Low). A Low on this line indicates that the CPU has relinquished control of the bus in response to a bus request.

**BUSREQ.** *Bus Request* (input, active Low). A Low on this line indicates that an external bus requester has obtained or is trying to obtain control of the bus.

**CLK.** *Clock Output* (output). The frequency of the processor timing clock is derived from the oscillator input (external oscillator) or crystal frequency (internal oscillator). The processor clock is further divided by one, two, or four (as programmed) and then output on this line.

**CTIN.** *Counter/Timer Input* (input, active High). These lines receive signals from external devices for the counter/timers.

**CTIO.** *Counter/Timer I/O* (bidirectional, active High, 3-state). These I/O lines transfer signals between the counter/timers and external devices.

**DMASTB.** *DMA Flyby Strobe* (output, active Low). These lines select peripheral devices for flyby transfers.

**EOP<sub>A</sub>, EOP<sub>B</sub>.** *End of Process* (input, active Low). An external source can terminate a DMA operation in progress by driving EOP<sub>A</sub> or EOP<sub>B</sub> Low. EOP always applies to the corresponding programmed channel; if no channel is active, EOP is ignored.

**GACK.** *Global Acknowledge* (input, active Low). A Low on this line indicates the CPU has been granted control of a global bus.

**GREQ.** *Global Request* (output, active Low, 3-state). A Low on this line indicates the CPU has obtained or is trying to obtain control of a global bus.

**GND.** *Ground*. Ground reference.

**HALT.** *Halt* (output, active Low, 3-state). This signal indicates that the CPU is in the Halt state and is awaiting an interrupt before operation can resume.

**IE.** *Input Enable* (output, active Low, 3-state). A Low on this line indicates that the direction of transfer on the Address/Data lines is toward the MPU.

**INT.** *Maskable Interrupts* (input, active Low). A Low on these lines requests an interrupt.

**IORQ.** *Input/Output Request* (output, active Low, 3-state). This signal indicates that AD<sub>0</sub>-AD<sub>7</sub> and A<sub>16</sub>-A<sub>23</sub> of the address bus hold a valid I/O address for an I/O read or write operation. An  $\overline{\text{IORQ}}$  signal is also generated with an  $\overline{\text{M1}}$  signal when an interrupt is being acknowledged, to indicate that an interrupt response vector can be placed on the data bus.

**M1.** *Machine Cycle One* (output, active Low, 3-state). This signal indicates that the current transaction is the opcode fetch cycle of a RETI instruction execution. M1 also occurs with  $\overline{\text{IORQ}}$  to indicate an interrupt acknowledge cycle.

**MREQ.** *Memory Request* (output, active Low, 3-state). This signal indicates that the address bus holds a valid address for a memory read or write operation.

**NMI.** *Nonmaskable Interrupt* (input, falling-edge activated). A High-to-Low transition on this line requests a nonmaskable interrupt.

**OE.** *Output Enable* (output, active Low, 3-state). A Low on this line indicates that the direction of transfer on the Address/Data lines is away from the MPU.

**OPT.** *Bus Option* (input). This signal establishes the bus option during reset.

<u>OPT</u>	<u>Bus Interface</u>
0	Z80 Bus, 8-bit
1	Z-BUS, 16-bit

**PAUSE.** *MPU Pause* (input, active Low). While this line is Low the MPU refrains from transferring data to or from an Extended Processing Unit in the system or from beginning the execution of an instruction.

**RD.** *Read* (output, active Low, 3-state). This signal indicates that the CPU or DMA peripheral is reading data from memory or an I/O device.

**RDY.** *DMA Ready* (input, active Low). These lines are monitored by the DMAs to determine when a peripheral device associated with a DMA port is ready for a read or write operation. When a DMA port is enabled to operate, its Ready line indirectly controls DMA activity; the manner in which DMA activity is controlled by the line varies with the operating mode (single-transaction, burst, or continuous).

**RESET.** *Reset* (input, active Low). A Low on this line resets the CPU and on-chip peripherals.

**RFSH.** *Refresh* (output, active Low, 3-state). This signal indicates that the lower ten bits of the Address bus contain a refresh address for dynamic memories and the current  $\overline{\text{MREQ}}$  signal should be used to perform a refresh to all dynamic memories.

**RxD.** *UART Receive* (input, active High). This line receives serial data at standard TTL levels.

**TxD.** *UART Transmit* (output, active High). This line transmits serial data at standard TTL levels.

**WAIT.** *Wait* (input, active Low). A Low on this line indicates that the responding device needs more time to complete a transaction.

**WR.** *Write* (output, active Low, 3-state). This signal indicates that the bus holds valid data to be stored at the addressed memory or I/O location.

**XTALI.** *Clock/Crystal Input* (time-base input). Connects a parallel-resonant crystal or an external single-phase clock to the on-chip oscillator.

**XTALO.** *Crystal Output* (time-base output). Connects a parallel-resonant crystal to the on-chip oscillator.

**+5V.** *Power Supply Voltage*. (+5 nominal).

## Bus Operations

Two kinds of operations can occur on the system bus: transactions and requests. At any given time, one device (either the CPU or a bus requester) has control of the bus and is known as the bus master. A transaction is initiated by the bus master and is responded to by some other device on the bus. Only one transaction can proceed at a time; seven kinds of transactions can occur:

*DMA Flyby.* This transaction is used by the DMA peripheral to transfer data between an external peripheral and memory.

*Halt.* This transaction is used to indicate that the CPU is entering the Halt state.

*Interrupt Acknowledge.* This transaction is used by the CPU to acknowledge an interrupt and to transfer additional information from the interrupting device.

*I/O.* This transaction is used by the CPU or DMA peripheral to transfer data to or from an external peripheral.

*Memory.* This transaction is used by the CPU or DMA peripheral to transfer data to or from a memory location.

*Refresh.* This type of transaction performed by the refresh peripheral does not transfer data; it refreshes dynamic memory.

*RETI.* This transaction is generated only by the CPU and is used in conjunction with the Z8400 peripheral's interrupt logic.

---

Only the bus master can initiate transactions. A request, however, can be initiated by a component that does not have control of the bus. Two types of these requests can occur:

**Bus.** This request is used by external devices to request control of the system bus to initiate transactions.

**Interrupt.** This request is used to request the attention of the CPU.

When an interrupt or bus request is made, it is answered by the CPU according to its type. For an interrupt request, the CPU initiates an interrupt acknowledge transaction and for bus requests, the CPU enters bus disconnect state, relinquishes the bus, and activates an Acknowledge signal.

Finally, the Z280 MPU itself may not be the system bus master. See the Multiprocessor Mode section for a discussion of this capability.

## Transactions

Information transfers (both instructions and data) to and from the Z280 MPU are accomplished through the use of transactions. All transactions start when  $\overline{AS}$  is driven Low and then raised High. This signal can be used to latch Z280 MPU addresses to de-multiplex the Z280 Address/Data lines required by Z80 Family peripherals. Coincident with  $\overline{AS}$  assertion, the Output Enable line is also asserted.

If the transaction requires an address, it is valid on the rising edge of  $\overline{AS}$ . No address is required for Interrupt Acknowledge transactions.

The Read and Write lines are used to time the actual data transfer. (Refresh transactions do not transfer any data and thus do not activate  $\overline{RD}$ .) For write operations, a Low on  $\overline{WR}$  indicates that valid data from the bus master is on the AD lines. The Output Enable line is also activated with  $\overline{WR}$ . For read operations, the bus master makes the AD lines 3-state before driving  $\overline{RD}$  Low so that the addressed device can put its data on the bus. The bus master samples this data on the falling clock edge just before raising  $\overline{RD}$  High. The Input Enable line is also activated with  $\overline{RD}$ .

**Wait Cycle.** The  $\overline{WAIT}$  line is sampled on the falling clock edge when data is to be sampled (i.e., when  $\overline{RD}$  or  $\overline{WR}$  rises).

If the  $\overline{WAIT}$  line is Low, another cycle is added to the transaction before data is sampled ( $\overline{RD}$  or  $\overline{WR}$  rises). In this added cycle and all subsequent cycles added due to  $\overline{WAIT}$  being Low, the  $\overline{WAIT}$  line is sampled on the falling edge and, if it is Low, another cycle is added to the transaction. In this way, the transaction can be extended by external devices to an arbitrary length to accommodate (for example) slow memories or I/O devices that are not yet ready for data transfer.

The  $\overline{WAIT}$  input is synchronous and thus must meet the specified setup and hold times in order for the Z280 MPU to function correctly. This requires asynchronously generated  $\overline{WAIT}$  signals to be synchronized to the CLK output before they are input into the Z280 MPU. Automatic wait states can also be generated by programming the Bus Timing and Control register and the Bus Timing and Initialization register; these are inserted in the transaction before the external  $\overline{WAIT}$  signal is sampled.

**Memory Transactions.** Memory transactions move instructions or data to or from memory when the Z280 MPU makes a memory access. Thus, they are generated during program execution to fetch instructions from memory and to fetch and store memory data. They are also generated to store old program status and fetch new program status during interrupt and trap handling, and are used by DMA peripherals to transfer information. A memory transaction is three bus cycles long unless extended with wait states (Figures 38 and 39).

**RETI Transactions.** These transactions (Figure 40) are similar to two memory read transactions except that  $\overline{M1}$  is asserted throughout each read transaction, falling early in the first bus cycle, and that  $\overline{MREQ}$ ,  $\overline{M1}$ ,  $\overline{RD}$  and  $\overline{IE}$  are deasserted on the rising edge of the clock following the third cycle. Each of the read transactions is followed by a minimum of three bus cycles of inactivity. These transactions are invoked when an RETI instruction is encountered in the instruction stream; they are used during the re-fetching of the instruction from memory so that interrupt logic within Z80 peripherals that monitor the bus for this instruction will function correctly.

**Note:** Refresh cycles and DMA transfers may occur between RETI bus cycles.



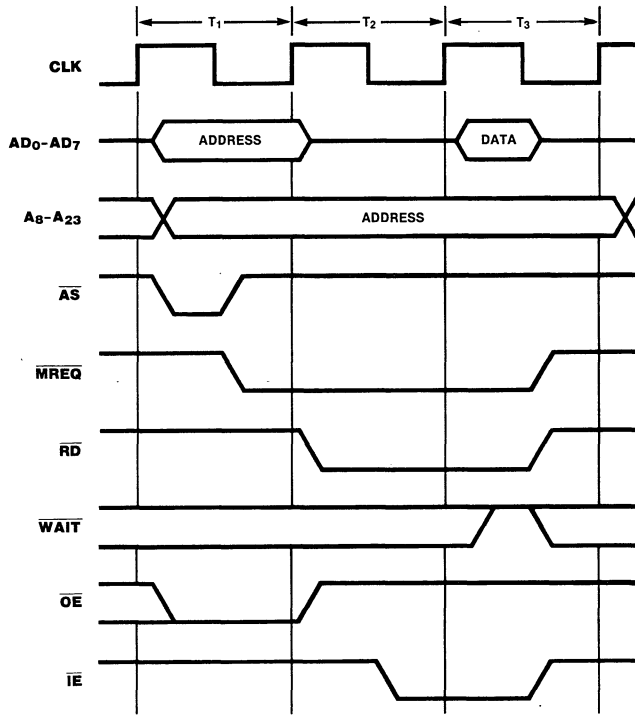


Figure 38. Memory Read Timing

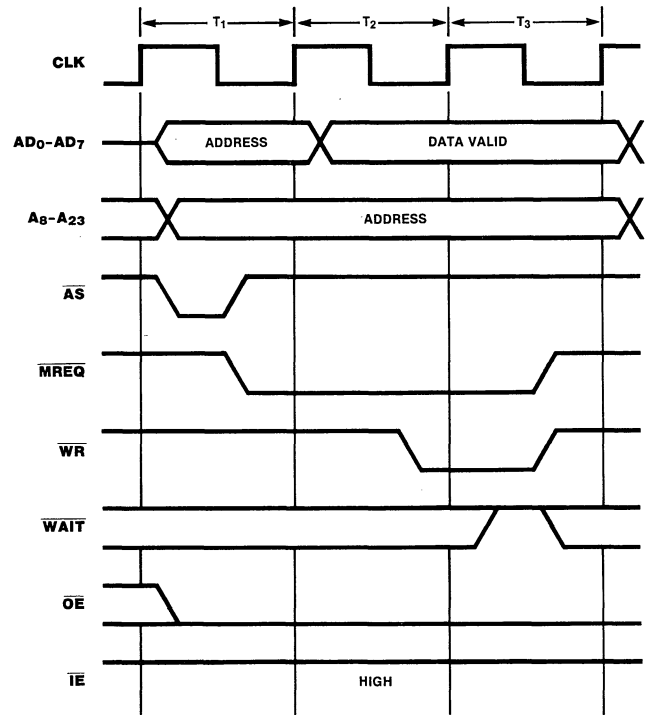


Figure 39. Memory Write Timing

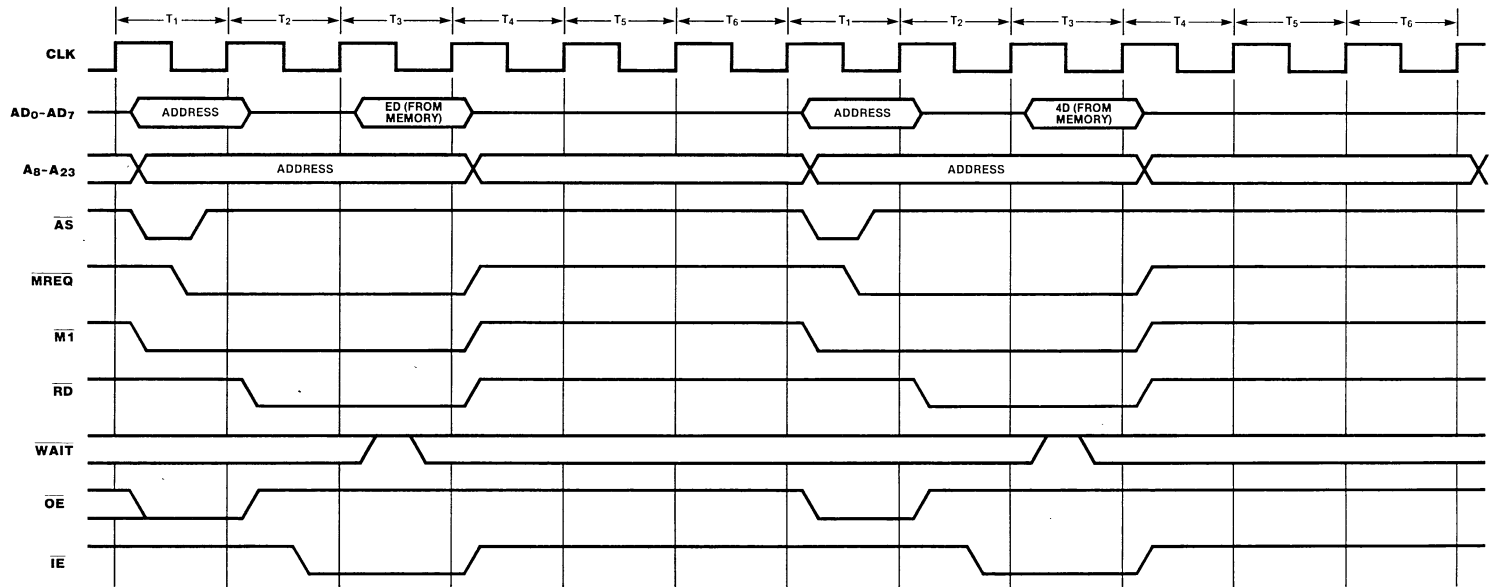


Figure 40. RETI Read Timing

**Halt Transactions.** The Halt bus transaction does not transfer data (Figure 41). It looks like a memory transaction, except that  $\overline{RD}$  and  $\overline{WR}$  remain High and no data is transferred. The  $\overline{WAIT}$  line is not sampled during the Halt transaction.

Halt transactions are identical to memory read transactions except that  $\overline{HALT}$  is asserted throughout the transaction, falling during the second half of the first bus cycle, and remains asserted until an interrupt is acknowledged. This transaction is invoked when a Halt instruction is encountered in the instruction stream or a fatal sequence of traps occurs. Although the Halt transaction is three cycles, the  $\overline{HALT}$  line remains asserted until an Interrupt request is acknowledged or a Reset is received. Refresh (to maintain a

minimum frequency of bus transactions) or DMA transfers may occur while  $\overline{HALT}$  is asserted; also, the bus can be granted. The address put out during the address phase of this cycle is the address of the Halt instruction.

**I/O Transactions.** I/O transactions move data to (Figure 42) or from (Figure 43) peripherals and are generated during the execution of I/O instructions.

I/O transactions are four clock cycles long at a minimum, and may be lengthened by the addition of wait cycles. The extra clock cycle allows for slower peripheral operation.

The  $\overline{IORQ}$  line indicates that an I/O transaction is taking place. The I/O address is found on  $AD_0$ - $AD_7$  and  $A_8$ - $A_{23}$  when  $\overline{AS}$  rises.

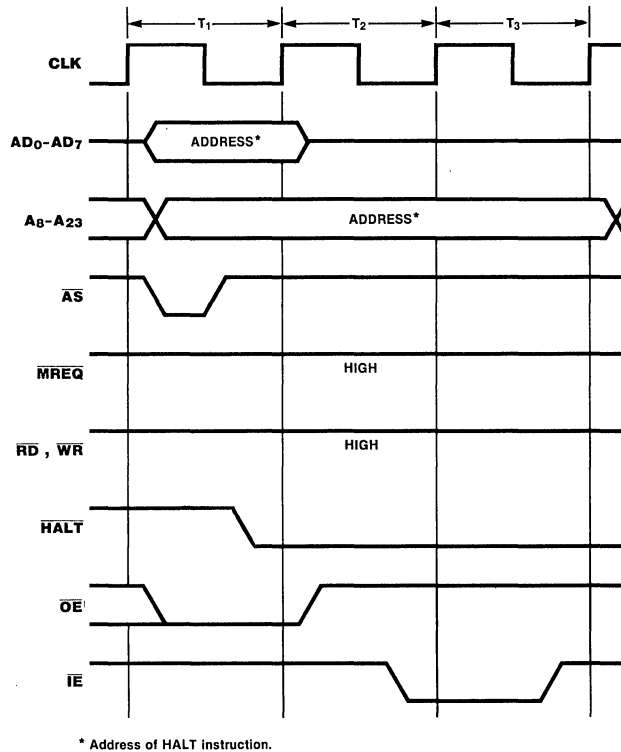


Figure 41. Halt Timing

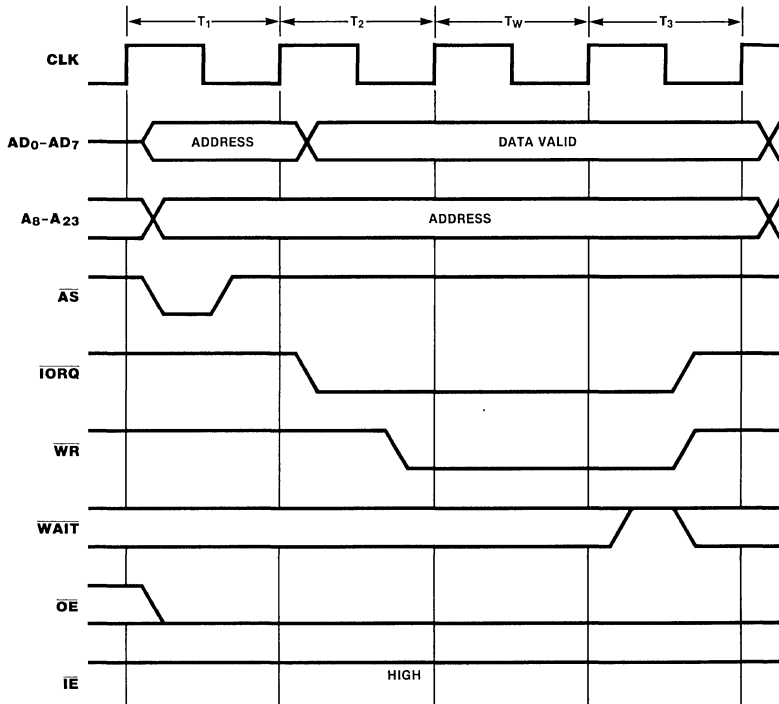


Figure 42. I/O Write Timing

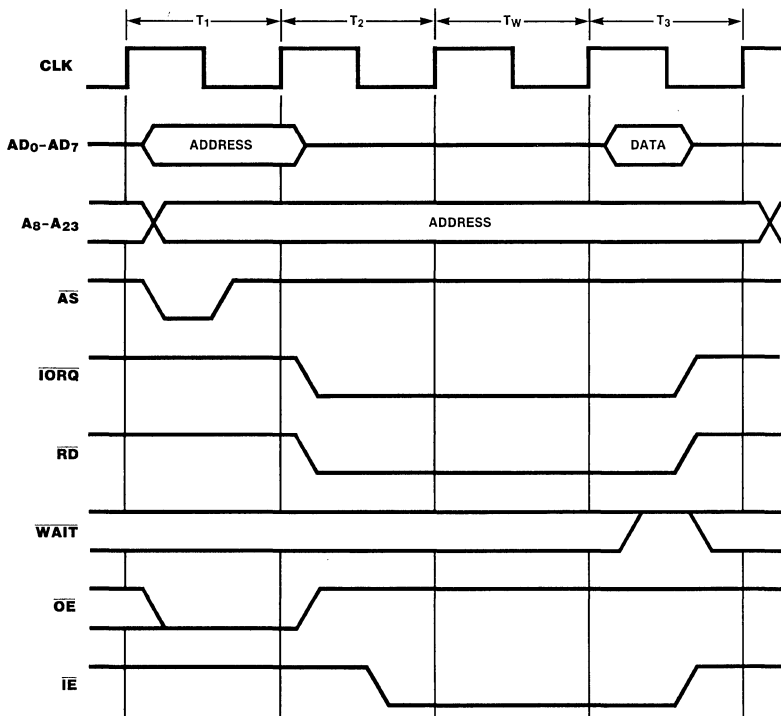


Figure 43. I/O Read Timing

**Interrupt Acknowledge Transactions.** These transactions (Figure 44) acknowledge an interrupt and read information from the device that generated the interrupt. The transactions are generated automatically by the hardware when an external interrupt request is detected.

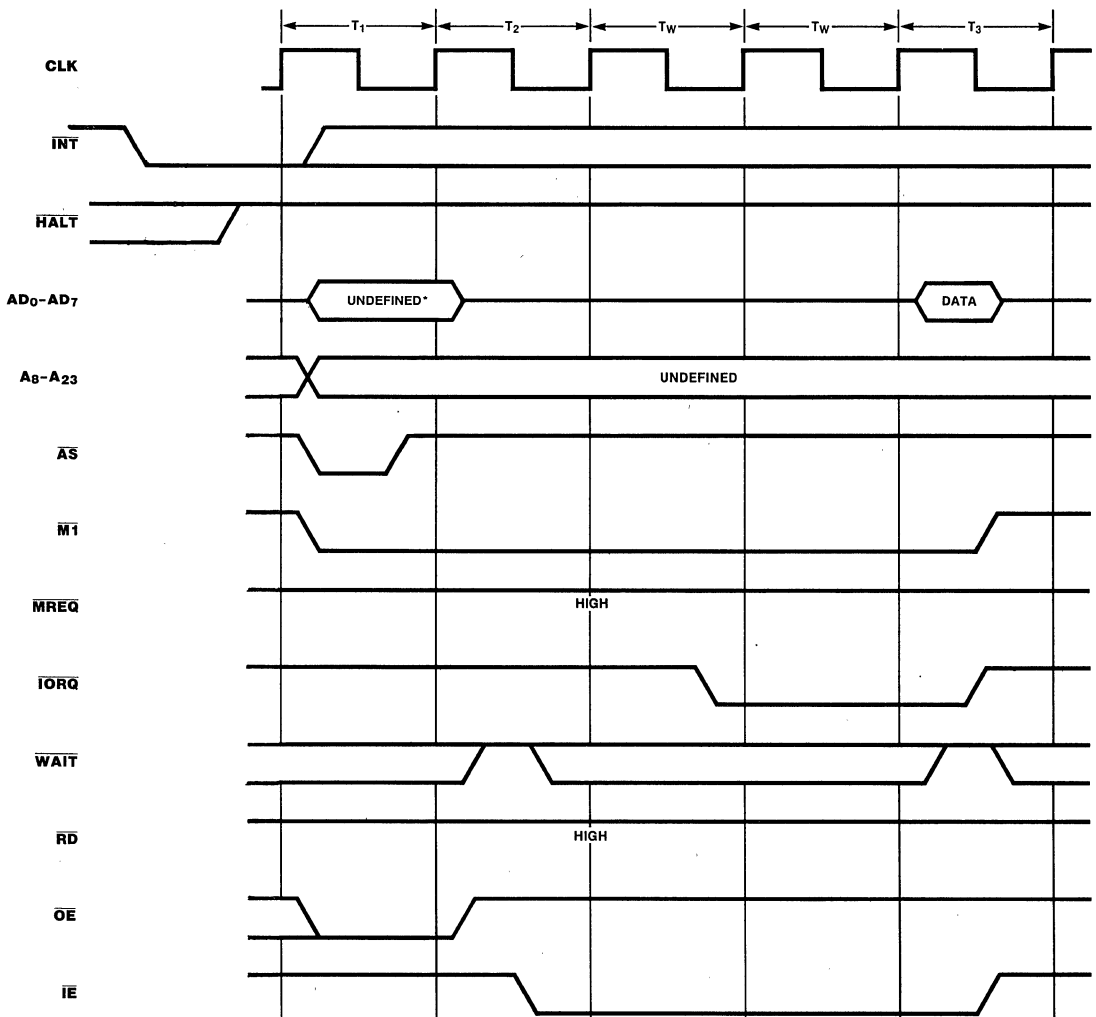
The Interrupt Acknowledge transactions are five cycles long at a minimum and have two automatic Wait cycles. The Wait cycles are used to give the interrupt priority daisy chain (or other priority resolution device) time to settle before the identifier is read. Additional automatic Wait states can be generated by programming the Bus Timing and Control register.

The Interrupt Acknowledge transaction is indicated by an  $\overline{M1}$  assertion without  $\overline{MREQ}$  during the first cycle. During this transaction the  $\overline{IORQ}$  signal becomes active during the third cycle to indicate that the interrupting device can place

an 8-bit vector on the bus. It is captured from the AD lines on the falling clock edge just before  $\overline{IORQ}$  is raised High.

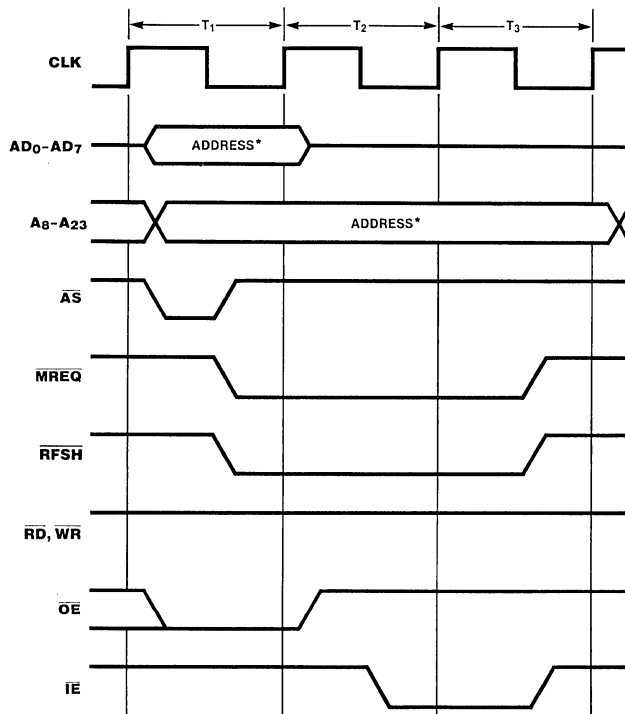
There are two places where the  $\overline{WAIT}$  line is sampled and, thus, where a Wait cycle can be inserted by external circuitry. The first serves to delay the falling edge of  $\overline{IORQ}$  to allow the daisy chain a longer time to settle, and the second serves to delay the point at which the vector is read.

**Refresh Transactions.** A memory refresh transaction (Figure 45) is generated by the Z280 refresh mechanism and can occur immediately after the final clock cycle of any other transaction. The memory refresh counter's 10-bit address is output on  $AD_0-AD_7$  and  $A_8-A_9$  during the normal time for addresses. The  $\overline{RFSH}$  line is activated with  $\overline{MREQ}$ . This transaction can be used to generate refreshes for dynamic RAMs.



\* AD<sub>1</sub> and AD<sub>2</sub> indicates type of interrupt being acknowledged, if interrupt mode 3 is in effect.

Figure 44. Maskable Interrupt Acknowledge Sequence



\*10 least significant bits are Refresh address, the rest are undefined.

Figure 45. Refresh Timing

## Requests

There are three kinds of request signals that the Z280 MPU supports. These are:

- Interrupt requests, which another device initiates and the CPU accepts and acknowledges.
- Bus requests, which an external potential bus master initiates and the Z280 MPU accepts and acknowledges.
- Global bus requests, which the CPU or on-chip DMA initiates to acquire a global System bus.

When a request is made, it is answered according to its type: for interrupt requests, an Interrupt Acknowledge transaction is initiated; for bus requests, an Acknowledge signal is sent; for global bus requests, an Acknowledge signal is received.

**Interrupt Requests.** The Z280 CPU supports two types of interrupt, maskable and nonmaskable (NMI). The Interrupt Request line of a device that is capable of generating an interrupt can be tied to the  $\overline{\text{NMI}}$  or maskable interrupt request lines. Several devices can be connected to one pin with the devices arranged in a priority daisy chain. However, all Z80 family peripherals should be on the same line (or no nesting of interrupts among different lines). The CPU uses different protocols for handling requests on the  $\overline{\text{NMI}}$  pin

than the protocol used for maskable interrupt pins. The sequence of events shown below should be followed:

Any High-to-Low transition on the  $\overline{\text{NMI}}$  input is asynchronously edge-detected, and the internal  $\overline{\text{NMI}}$  latch is set. At the beginning of the last clock cycle in the last internal machine cycle of any instruction, the interrupt inputs are sampled along with the state of the internal  $\overline{\text{NMI}}$  latch.

If a maskable interrupt is requested and the Master Status register indicates that requests on that line are to be accepted, the next possible bus transaction is the Interrupt Acknowledge transaction, which results in information from the highest-priority interrupting device being read off the AD lines. This data is used to initiate the interrupt service routine. For a nonmaskable interrupt request, the hexadecimal constant 0066 is used to initiate the interrupt service routine, except in mode 3.

**Bus Requests.** To generate transactions on the bus, a potential bus master (such as the DMA Controller) must gain control of the bus by making a bus request. A bus request is initiated by pulling  $\overline{\text{BUSREQ}}$  Low. Several bus requesters may be wired-OR to the  $\overline{\text{BUSREQ}}$  pin; priorities are resolved externally to the CPU, usually by a priority daisy chain.

The asynchronous  $\overline{\text{BUSREQ}}$  signal generates an internal  $\overline{\text{BUSREQ}}$ , which is synchronous. If the external  $\overline{\text{BUSREQ}}$  is Low at the beginning of any machine cycle, the internal  $\overline{\text{BUSREQ}}$  causes the Bus Acknowledge line ( $\overline{\text{BUSACK}}$ ) to be asserted after the current machine cycle is completed. (Exceptions are the TSET instruction where the read-modify-write cycle is atomic and DMA transfer in burst or continuous mode.) The CPU then enters Bus Disconnect state and gives up control of the bus. All MPU Output pins, except  $\overline{\text{BUSACK}}$ , are 3-stated.

The CPU regains control of the bus after  $\overline{\text{BUSREQ}}$  rises. Any device desiring control of the bus must wait at least two bus cycles after  $\overline{\text{BUSREQ}}$  has risen before pulling it down again.

The on-chip DMA channels have higher priority than external devices requesting the bus via  $\overline{\text{BUSREQ}}$ .

## Z-BUS External Interface

### Features

- 16-bit data bus
- Multiplexed address/data lines
- Supports high-speed burst mode transfers
- Provides EPA interface

### Pin Descriptions

**A<sub>16</sub>-A<sub>23</sub>.** *Address* (output, active High, 3-state). These address lines carry I/O addresses and memory addresses during bus transactions.

**AD<sub>0</sub>-AD<sub>15</sub>.** *Address/Data* (bidirectional, active High, 3-state). These 16 multiplexed address and data lines carry I/O addresses, memory addresses, and data during bus transactions.

**AS.** *Address Strobe* (output, active Low, 3-state). The rising edge of Address Strobe indicates the beginning of a transaction and shows that the address, status, R/W, and B/W signals are valid.

**$\overline{\text{BUSACK}}$ .** *Bus Acknowledge* (output, active Low). A Low on this line indicates that the CPU has relinquished control of the bus in response to a bus request.

**$\overline{\text{BUSREQ}}$ .** *Bus Request* (input, active Low). A Low on this line indicates that an external bus requester has obtained or is trying to obtain control of the bus.

**B/W.** *Byte/Word* (output, Low = Word, 3-state). This signal indicates whether a byte or a word of data is to be transmitted during a transaction.

**CLK.** *Clock Output* (output). The frequency of the processor timing clock is derived from the oscillator input (external oscillator) or crystal frequency (internal oscillator) by dividing the crystal or external oscillator input by two. The processor clock is further divided by one, two, or four (as programmed), and then output on this line.

**CTIN.** *Counter/Timer Input* (input, active High). These lines receive signals from external devices for the counter/timers.

**CTIO.** *Counter/Timer I/O* (bidirectional, active High, 3-state). These I/O lines transfer signals between the counter/timers and external devices.

**$\overline{\text{DMASTB}}$ .** *DMA Flyby Strobe* (output, active Low). These lines select peripheral devices for DMA flyby transfers.

**$\overline{\text{DS}}$ .** *Data Strobe* (output, active Low, 3-state). This signal provides timing for data movement to or from the bus master.

**$\overline{\text{EOP}}$ .** *End of Process* (input, active Low). An external source can terminate a DMA operation in progress by driving  $\overline{\text{EOP}}$  Low.  $\overline{\text{EOP}}$  always applies to the active channel; if no channel is active,  $\overline{\text{EOP}}$  is ignored.

**$\overline{\text{GACK}}$ .** *Global Acknowledge* (input, active Low). A Low on this line indicates the CPU has been granted control of a global bus.

**$\overline{\text{GREQ}}$ .** *Global Request* (output, active Low, 3-state). A Low on this line indicates the CPU has obtained or is trying to obtain control of a global bus.

**$\overline{\text{IE}}$ .** *Input Enable* (output, active Low, 3-state). A Low on this line indicates that the direction of transfer on the Address/Data lines is toward the CPU.

**$\overline{\text{INT}}$ .** *Maskable Interrupts* (input, active Low). A Low on these lines requests an interrupt.

**$\overline{\text{NMI}}$ .** *Nonmaskable Interrupt* (input, falling-edge activated). A High-to-Low transition on this line requests a nonmaskable interrupt.

**$\overline{\text{OE}}$ .** *Output Enable* (output, active Low, 3-state). A Low on this line indicates that the direction of transfer on the Address/Data lines is away from the MPU.

**OPT.** *Bus Option* (input). This signal establishes the bus option during reset as follows:

OPT	Bus Interface
0	Z80-Bus, 8-bit
1	Z-BUS, 16-bit

**$\overline{\text{PAUSE}}$ .** *CPU Pause* (input, active Low). While this line is Low the CPU refrains from transferring data to or from an Extended Processing Unit in the system or from beginning the execution of an instruction.

**$\overline{\text{RDY}}$ .** *DMA Ready* (input, active Low). These lines are monitored by the DMA channels to determine when a peripheral device associated with a DMA channel is ready for a read or write operation. When a DMA channel is enabled to operate, its Ready line indirectly controls DMA activity; the manner in which DMA activity is controlled by the line varies with the operating mode (single-transaction, burst, or continuous).

**$\overline{\text{RESET}}$ .** *Reset* (input, active Low). A Low on this line resets the CPU and on-chip peripherals.

**R/W.** *Read/Write* (output, Low = Write, 3-state). This signal determines the direction of data transfer for memory, I/O, or EPU transfer transactions.

**RxD.** *UART Receive* (input, active High). This line receives serial data at standard TTL levels.

**ST<sub>0</sub>-ST<sub>3</sub>.** *Status* (output, active High, 3-state). These four lines indicate the type of transaction occurring on the bus and give additional information about the transaction.

**TxD.** *UART Transmit* (output, active High). This line transmits serial data at standard TTL levels.

**WAIT.** *Wait* (input, active Low). A Low on this line indicates that the responding device needs more time to complete a transaction.

**XTALI.** *Clock/Crystal Input* (time-base input). Connects a parallel-resonant crystal or an external single-phase clock to the on-chip clock oscillator.

**XTALO.** *Crystal Output* (time-base output). Connects a parallel-resonant crystal to the on-chip clock oscillator.

**+ 5V.** *Power Supply Voltage.* (+ 5 nominal).

**GND.** *Ground.* Ground reference.

## Bus Operations

Two kinds of operations can occur on the system bus: transactions and requests. At any given time, one device (either the CPU or a bus requester) has control of the bus and is known as the bus master. A transaction is initiated by the bus master and is responded to by some other device on the bus. Only one transaction can proceed at a time; eight kinds of transactions can occur:

**Burst Memory.** These transactions are used to transfer four words of instructions from the memory to the CPU.

**DMA Flyby.** This transaction is used by the DMA peripheral to transfer data between an external peripheral and memory.

**EPU Transfer.** This transaction is used to transfer data between the CPU and an EPU.

**Halt.** This transaction is used to indicate that the CPU is entering the Halt state.

**Interrupt Acknowledge.** This transaction is used by the CPU to acknowledge an external interrupt request and to transfer additional information from the interrupting device.

**I/O.** This transaction is used by the bus master to transfer data to or from an external peripheral.

**Memory.** This transaction is used by the bus master to transfer data to or from a memory location.

**Refresh.** These transactions by the refresh mechanism do not transfer data; they refresh dynamic memory.

Only the bus master can initiate transactions. A request, however, can be initiated by a device that does not have control of the bus. Two types of requests can occur:

**Bus.** This request is used to request control of the bus to initiate transactions.

**Interrupt.** This request is used to request servicing by the CPU.

When an interrupt or bus request is made, it is answered according to its type: for an externally generated interrupt request, an Interrupt Acknowledge transaction is initiated by the CPU; for bus requests, the MPU enters Bus Disconnect state, relinquishes the bus, and activates an acknowledge signal.

## Transactions

Data transfers to and from the Z280 MPU are accomplished through the use of transactions.

All transactions start with Address Strobe ( $\overline{AS}$ ) being driven Low and then raised High by the Z280 MPU. On the rising edge of  $\overline{AS}$ , the Status lines ST<sub>0</sub>-ST<sub>3</sub> are valid; these lines indicate the type of transaction being initiated (Table 9); seven types of transactions are discussed in the sections that follow. Associated with the status lines are two other lines that become valid at this time:  $R/\overline{W}$ , and  $B/\overline{W}$ .

Table 9. Status Code Table

Status Lines	
3••0	Type of Transaction
0000	Reserved
0001	Refresh
0010	I/O transaction
0011	Halt
0100	Interrupt acknowledge line A
0101	$\overline{NMI}$ acknowledge
0110	Interrupt acknowledge line B
0111	Interrupt acknowledge line C
1000	Transfer between CPU and memory, cacheable
1001	Transfer between CPU and memory, non-cacheable
1010	Data transfer between EPU and memory
1011	Reserved
1100	EPU Instruction fetch, template, subsequent words
1101	EPU Instruction fetch, template, first word
1110	Data transfer between EPU and CPU
1111	Test and Set (data transfers)

If the transaction requires an address, it is valid on the rising edge of  $\overline{AS}$ . No address is required for EPU-CPU transfer transactions; the contents of the A and AD lines while  $\overline{AS}$  is asserted are undefined. If an address is generated, the  $\overline{OE}$  signal is also activated.



The Z-BUS MPUs use Data Strobe ( $\overline{DS}$ ) to time the actual data transfer. (Note that Refresh and Halt transactions do not transfer any data and thus do not activate  $\overline{DS}$ .) For write operations ( $R/\overline{W} = \text{Low}$ ), a Low on  $\overline{DS}$  indicates that valid data from the bus master is on the  $AD$  lines. The Output Enable continues to be asserted until  $\overline{DS}$  is deasserted. For read operations ( $R/\overline{W} = \text{High}$ ), the bus master makes  $AD$  lines 3-state, deasserts  $\overline{OE}$ , and asserts  $\overline{IE}$  after driving  $\overline{DS}$  Low so that the addressed device can put its data on the bus. The bus master samples this data on the falling clock edge just before raising  $\overline{DS}$  and  $\overline{IE}$  High.

**Wait Cycle.** The  $\overline{WAIT}$  line is sampled on the falling clock edge when data is sampled by the Z280 MPU (Read) or the falling clock edge before  $\overline{DS}$  rises (Read or Write). If  $\overline{WAIT}$  is Low, another cycle is added to the transaction before data is sampled or  $\overline{DS}$  rises. In this added cycle, and all subsequent cycles added when  $\overline{WAIT}$  is Low,  $\overline{WAIT}$  is again sampled on the falling clock edge and, if it is Low, another cycle is added to the transaction. In this way, the transaction can be extended to an arbitrary length by external circuitry to accommodate (for example) slow memories or I/O devices that are not yet ready for data transfer. Automatic insertions of wait states by the CPU or on-chip DMA channels can be programmed by setting fields in the Bus Timing and Control register and Bus Timing and Initialization register to indicate the number to be inserted.

**Memory Transactions.** Memory transactions move data to or from memory when a bus master makes a memory

access. Thus, they are generated during program execution to fetch instructions from memory and to fetch and store memory data. They are also generated to store old program status and fetch new program status during interrupt and trap handling and after reset.

A memory transaction is three bus cycles long unless extended when  $\overline{WAIT}$  is asserted.

Bytes transferred to or from odd memory locations (address bit 0 = 1) are always transmitted on lines  $AD_0$ - $AD_7$  (bit 0 on  $AD_0$ ). Bytes transferred to or from even memory locations (address bit 0 = 0) are always transmitted on lines  $AD_8$ - $AD_{15}$  (bit 0 on  $AD_8$ ). For byte reads ( $B/\overline{W}$  High,  $R/\overline{W}$  High), the CPU or on-chip DMA channel uses only the byte whose address it put out on the bus. For byte writes ( $B/\overline{W}$  High,  $R/\overline{W}$  Low), the memory should store only the byte whose address was output. During byte memory writes, the CPU (or on-chip DMA channel in non-Flyby transactions) places the same byte on both halves of the bus, and the proper byte must be selected by testing  $A_0$ . For word transfers ( $B/\overline{W} = \text{Low}$ ), all 16 bits are captured by the CPU or DMA channel (Read:  $R/\overline{W} = \text{High}$ ) or stored by the memory (Write:  $R/\overline{W} = \text{Low}$ ). For these transactions (either memory or I/O) the bytes of data appear swapped on the bus with the most significant byte on  $AD_7$ - $AD_0$  and the least significant byte on  $AD_{15}$ - $AD_8$ . A word is aligned if the address is even; otherwise it is unaligned.

Memory transaction timings are shown in Figures 46-50.

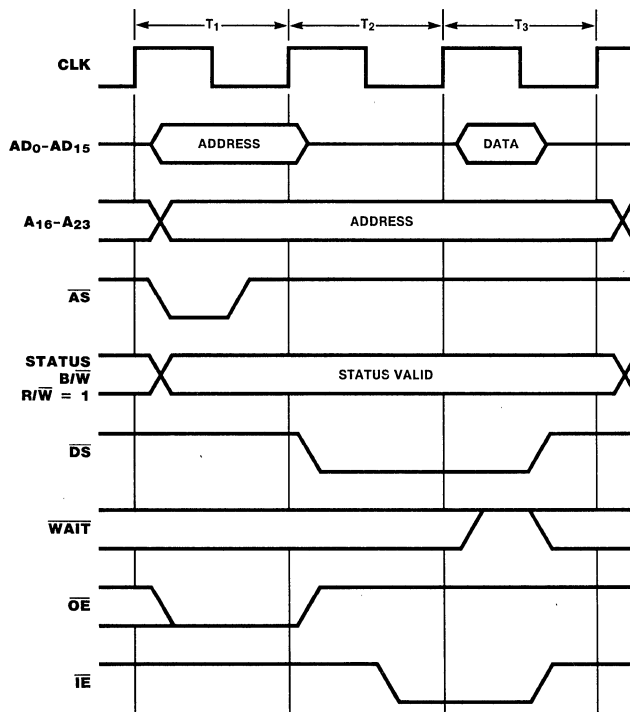


Figure 46. Memory Read Timing

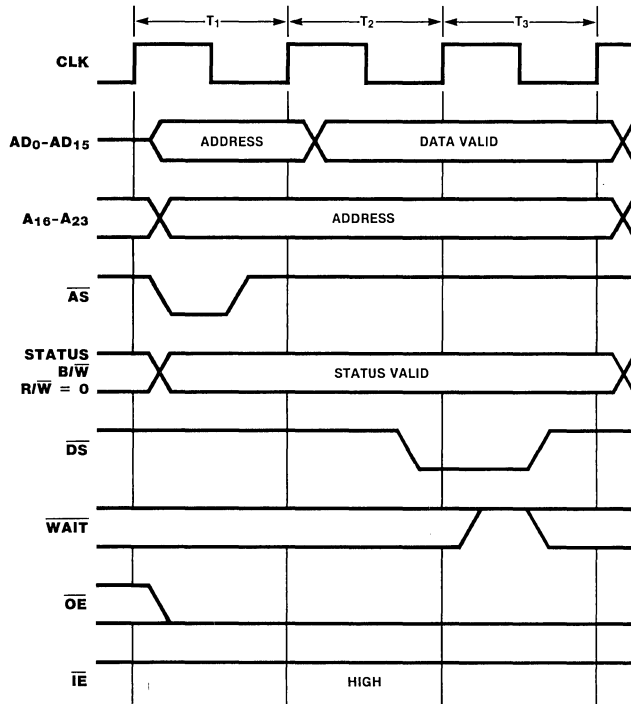


Figure 47. Memory Write Timing

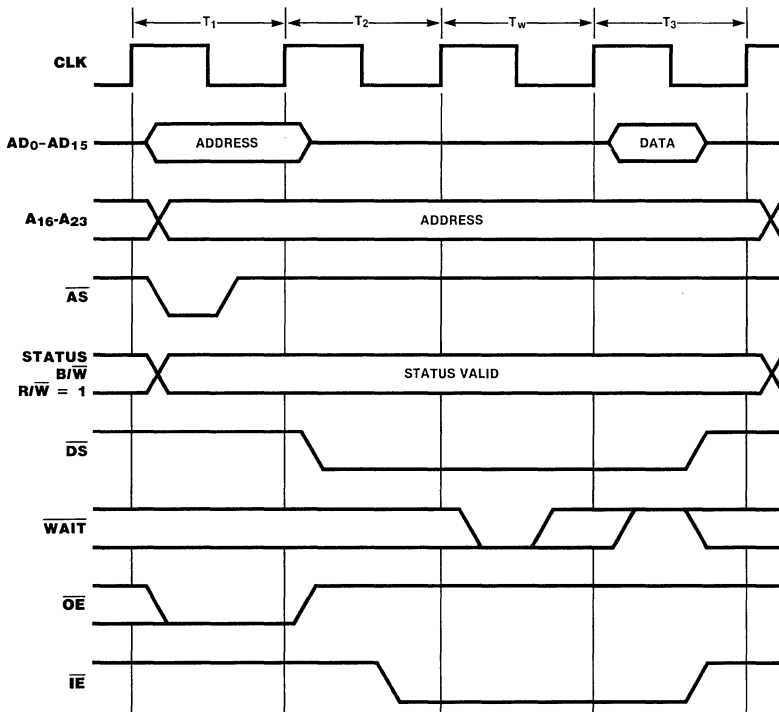


Figure 48. Memory Read Timing with External Wait Cycle

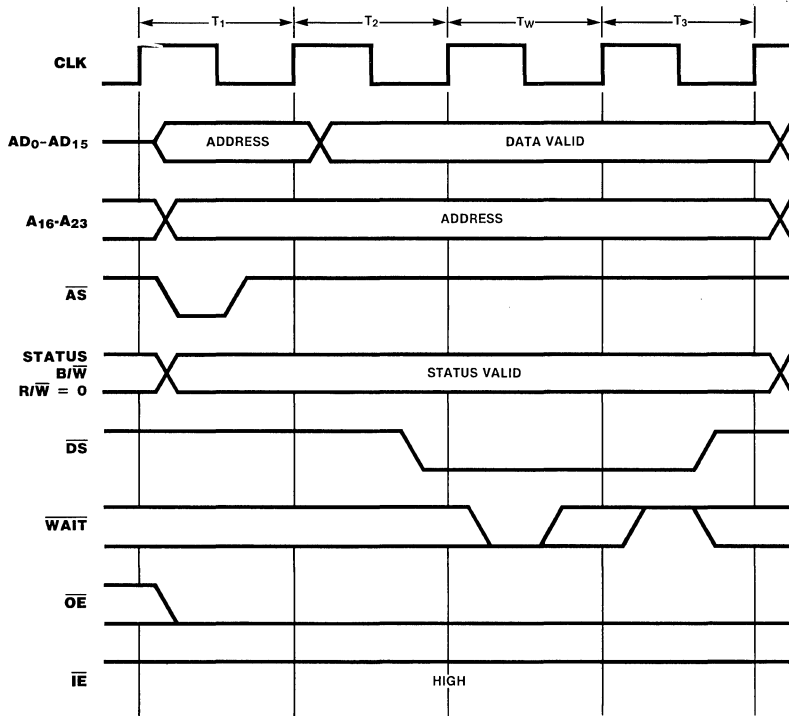


Figure 49. Memory Write Timing with External Wait Cycle

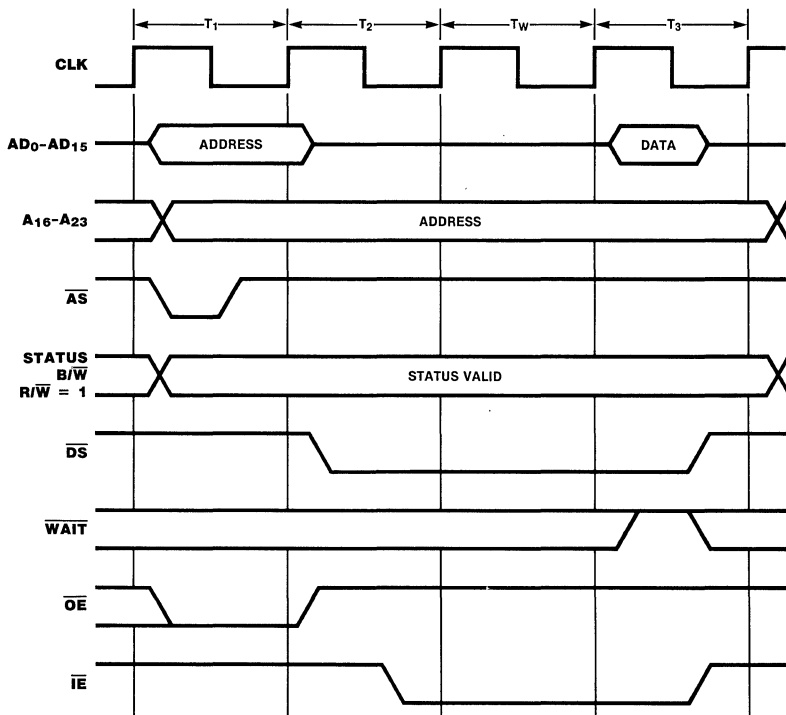


Figure 50. Memory Read Timing with Internal Wait Cycle

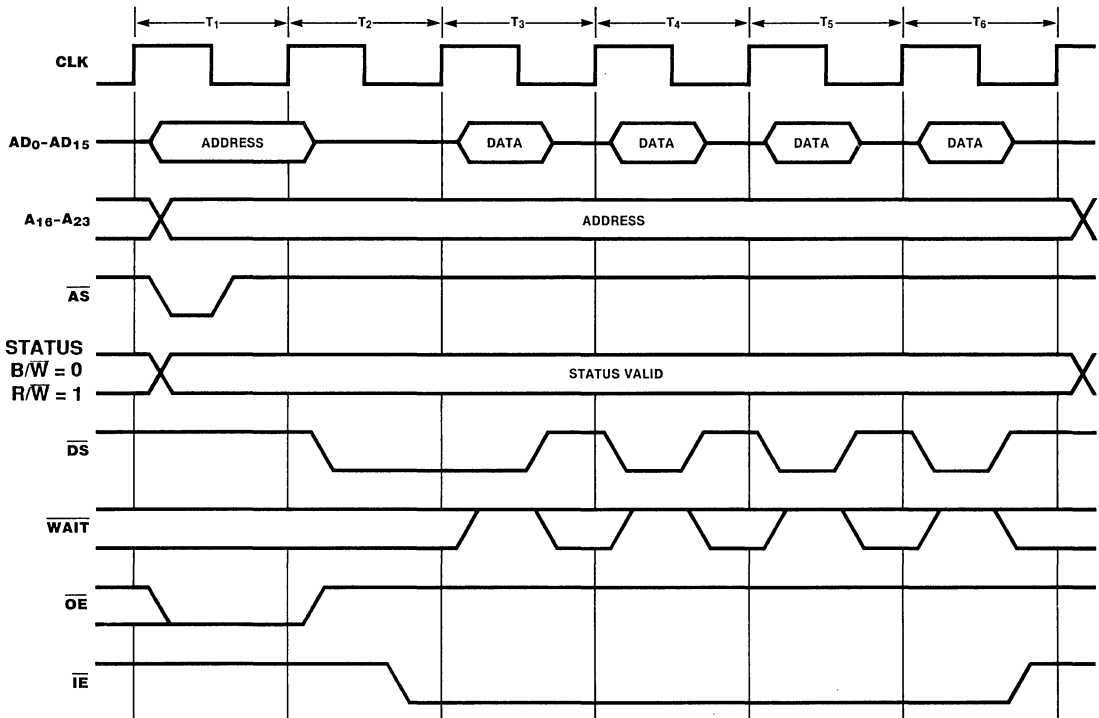


Figure 51. Burst Memory Read Timing

**Burst Memory Transactions.** Burst memory transactions use multiple Data Strokes associated with a single Address Strobe. The CPU uses burst transactions to read four consecutive words in four data transactions. The address of the first word read during a burst transaction has zeros in the three least significant bits. Control bits in the Cache Control register indicate whether or not portions of the memory system can support burst transactions.

The CPU uses burst mode reads only for fetching instructions. If an instruction is to be fetched from a location within a half of physical memory that supports burst transactions, the CPU reads the eight bytes that contain the first byte of the instruction. (EPA template fetches do not use the burst transaction.)

Timing for the first data transfer during a burst transaction is identical to that for a single memory read, including the automatic insertion of wait states, except there are four  $T_3$  states. Subsequent data transfers do not include automatic wait states. On the first data transfer, if  $\overline{WAIT}$  is sampled active then it is sampled again every bus clock cycle until it is inactive, at which time the data is read from the bus. Burst memory read timing is shown in Figure 51.

**Note:** Burst Transactions can occur only in Z-BUS mode.

**Halt Transactions.** Halt transactions do not transfer data. They look like a memory transaction, except that  $\overline{DS}$  remains High and no data is transferred.

A Halt transaction (Figure 52) is generated when the CPU executes a HALT instruction or when a fatal sequence of traps and bus errors occurs. The address placed on the AD lines is the location of the Halt instruction or the instruction that initiated the fatal sequence of traps and errors. The Status lines indicate a Halt transaction (0011).

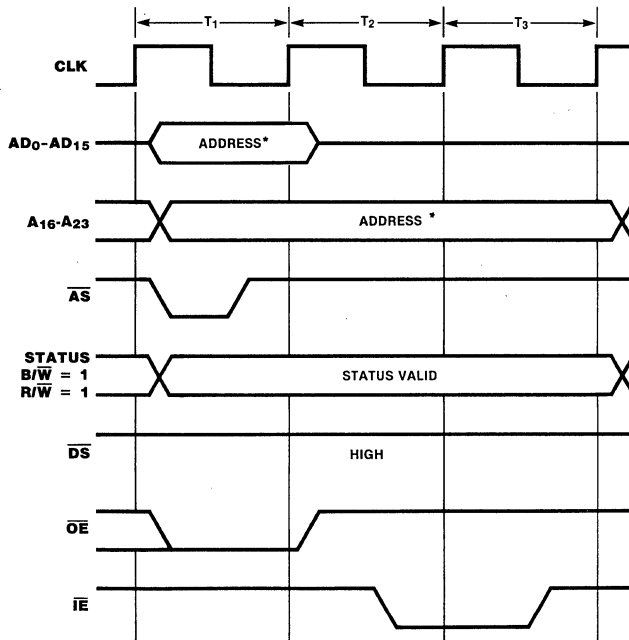
$\overline{WAIT}$  is not sampled during the Halt transaction.

**I/O Transactions.** I/O transactions (Figures 53 and 54) move data to or from peripherals and are generated during the execution of I/O instructions. I/O transactions to on-chip peripheral devices (I/O pages FE<sub>H</sub> and FF<sub>H</sub>) do not generate external bus transactions.

I/O transactions are four bus cycles long at a minimum, and they can be lengthened by the addition of wait cycles either automatically generated as indicated in the Bus Timing and Control register or generated by an external device. The extra clock cycles allow for slower peripheral operation.

The status lines indicate that the access is an I/O transaction (0010). The I/O address is found on AD<sub>0</sub>-AD<sub>15</sub> and A<sub>16</sub>-A<sub>23</sub>.

Byte data ( $B/\overline{W}$  = High) is transmitted on AD<sub>0</sub>-AD<sub>7</sub>. This allows peripheral devices to attach to only eight of the AD lines. Word data ( $B/\overline{W}$  = Low) is transmitted with the most significant byte on AD<sub>0</sub>-AD<sub>7</sub> and the least significant byte on AD<sub>8</sub>-AD<sub>15</sub>.



\*Address of Halt Instruction.

Figure 52. Halt Timing

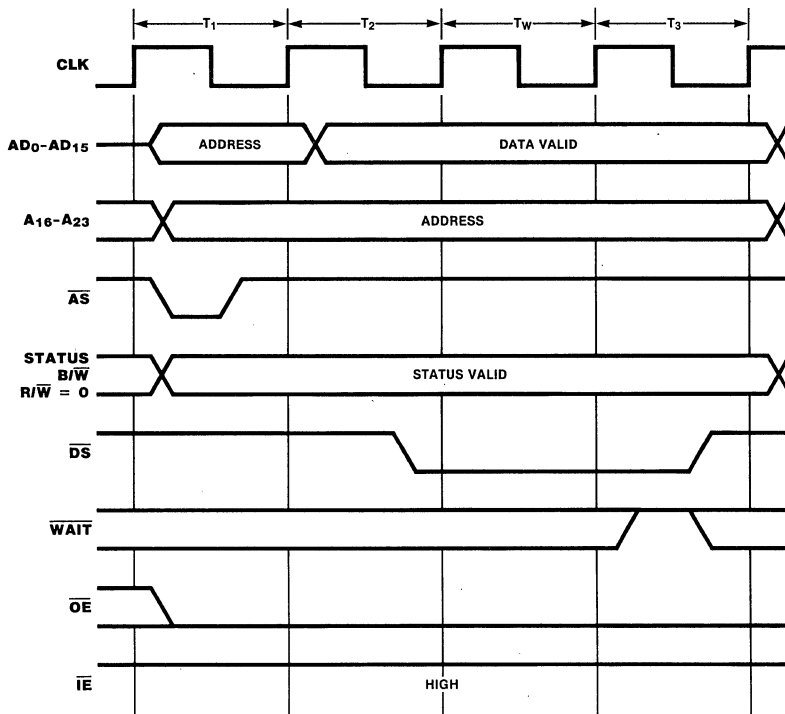


Figure 53. I/O Write Timing

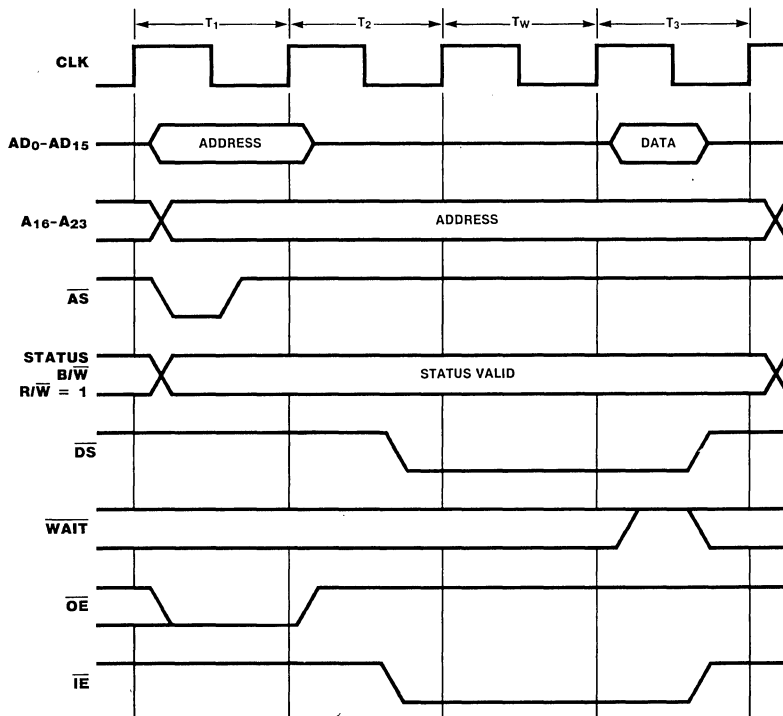


Figure 54. I/O Read Timing

**Interrupt Acknowledge Transactions.** These transactions (Figure 55) acknowledge an interrupt and read an identifier from the device that generated the interrupt. Interrupt Acknowledge transactions are generated automatically by the hardware when an external interrupt is detected.

These transactions are five cycles long at a minimum, with at least two automatic Wait cycles, although others can be added by programming the Bus Timing and Control register. The Wait cycles are used to give the interrupt priority daisy chain (or other priority resolution device) time to settle before the identifier is read.

The only item of data transferred is the identifier that is captured from the AD lines on the falling clock edge just before  $\overline{DS}$  is raised High. The length of time that  $\overline{DS}$  is asserted is identical with I/O timing programmed in the Bus Timing and Control register.

There are two places where  $\overline{WAIT}$  is sampled and thus a Wait cycle can be inserted by external devices. The first place serves to delay the falling edge of  $\overline{DS}$  to allow the daisy chain a longer time to settle, and the second place serves to delay the point at which data is read.

**Refresh Transactions.** A memory Refresh transaction (Figure 56) is generated by the refresh mechanism and can come immediately after the final clock cycle of any other transaction. The memory refresh counter's 10-bit address is output on the low order 10 bits of the bus during the first cycle of the transaction. The contents of the rest of the bus are undefined. The Status lines indicate Refresh (0001). This transaction can be used to generate refreshes for dynamic RAMs. Refreshes may occur while the CPU is in the Halt or Fatal state.

### CPU-Extended Processing Unit Interaction

The Z280 CPU with a Z-BUS interface and  $\overline{PAUSE}$  input line and one or more Extended Processing Units (EPUs) work together like a single CPU component, with the CPU providing address, status, and timing signals and the EPU supplying and capturing data. The EPU monitors the status and timing signals output by the CPU so that it knows when to participate in a memory transaction; for EPU to memory transfers, the CPU puts its AD lines in 3-state while  $\overline{DS}$  is Low, so that the EPU can use them.

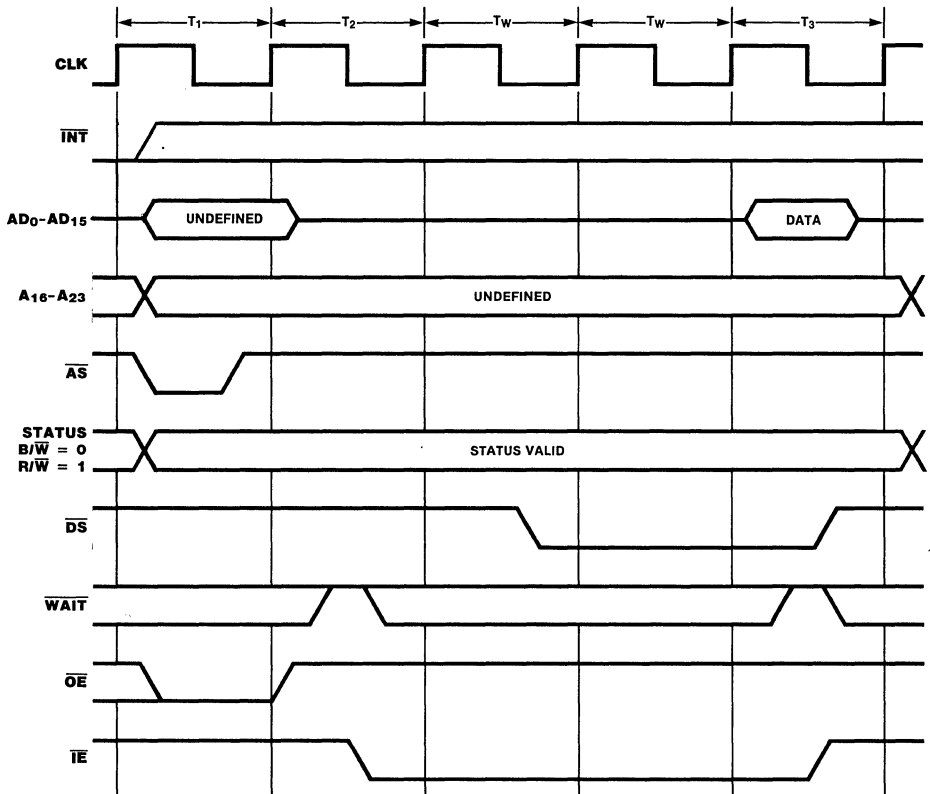
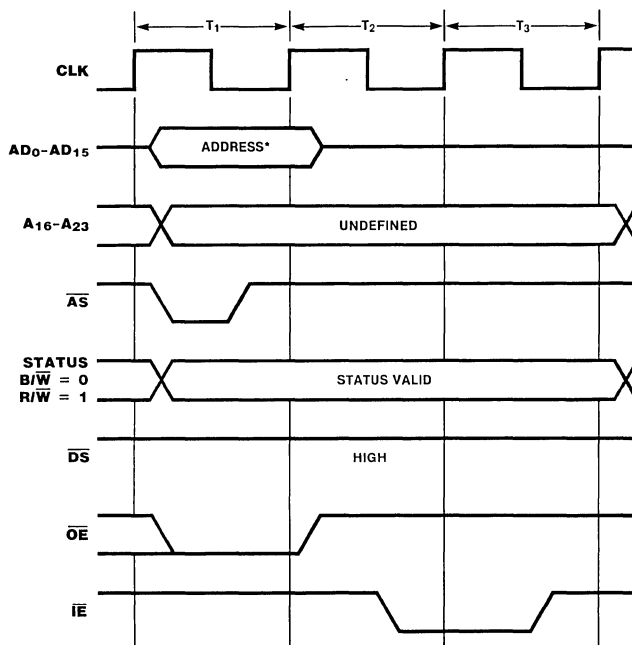


Figure 55. Interrupt Acknowledge Timing



\*10 least-significant bits are Refresh address.

Figure 56. Memory Refresh Timing

In order to know which transaction it is to participate in, the EPU must track the following sequence of events:

- When the CPU fetches the first word of an EPA instruction template from memory (ST<sub>3</sub>-ST<sub>0</sub> = 1101), the EPU must also capture the instruction returned by the memory. Within the template is an ID field that indicates whether or not the EPU is to execute the instruction.
- The next non-refresh transaction by the CPU is the fetch of the second word of the instruction (ST<sub>3</sub>-ST<sub>0</sub> = 1100). The EPU must also capture this word. If the template is not aligned, a third fetch is made (ST<sub>3</sub>-ST<sub>0</sub> = 1100).
- If the instruction involves a read or write to memory, then transfers of data between memory and the EPU (ST<sub>3</sub>-ST<sub>0</sub> = 1010) are the next non-refresh transactions performed by the CPU. The EPU must supply the data (Write: R/W = Low) or capture the data (Read: R/W = High) for each transaction, just as if it were part of the CPU. In both cases, the CPU 3-states its AD lines while data is being transferred (DS Low).

- If the instruction involves a transfer from the EPU to the Z280 MPU, the next non-refresh transaction is the CPU transferring data between the EPU and CPU (ST<sub>3</sub>-ST<sub>0</sub> = 1110).

In order to follow this sequence, an EPU has to monitor the status lines to verify that the transaction it is monitoring on the bus was generated by the CPU. In a multiple EPU system, there is no indication on the bus as to which EPU is cooperating with the CPU at any given time. This must be determined by the EPUs from the templates they capture.

When an EPU begins to execute an extended instruction, the CPU can continue fetching and executing instructions. If the EPU wishes to halt the CPU from executing another instruction or bus transaction, the EPU must activate the PAUSE line to stop the CPU until the EPU is ready for subsequent MPU activity. This mechanism is used to synchronize MPU-EPU activity.



**EPU Transfer Transactions.** These transactions (Figures 57-59) allow the CPU to transfer data to or from an EPU or to read or write an EPU's status registers. They are generated during the execution of the EPU instructions.

EPU-to-Memory transfers are five cycles unless extended by  $\overline{\text{WAIT}}$ . Memory-to-EPU transfers are three cycles unless extended by  $\overline{\text{WAIT}}$ .

EPU-CPU transfer transactions have the same form as I/O transactions and thus are four clock cycles long, unless extended by  $\overline{\text{WAIT}}$ . Although  $\overline{\text{AS}}$  is asserted, no address is generated and the contents of the bus are undefined; only one status code is used (1110).

In a multiple EPU system, the EPU that is to participate in a transaction is selected implicitly by the ID code in the EPU template, rather than by an address. The Read/Write line ( $\text{R}/\overline{\text{W}}$  = High) indicates the direction of the data transfer into the CPU.

### Requests

The Z280 MPU supports three types of request signals. These are:

- Interrupt requests, which another device initiates and the CPU accepts and acknowledges.

- Bus requests, which an external potential bus master initiates and the CPU accepts and acknowledges.
- Global bus requests, which the CPU or on-chip DMA initiates to acquire a global system bus.

When a request is made, it is answered according to its type: for interrupt requests, an Interrupt Acknowledge transaction is initiated by the CPU; for bus requests, an acknowledge signal is sent; for global bus request, an acknowledge signal is received.

**Interrupt Requests.** The Z280 MPU supports two types of external interrupts, maskable and nonmaskable ( $\overline{\text{NMI}}$ ). The Interrupt Request line of a device that is capable of generating an interrupt may be tied to any of the interrupt pins. Several devices can be connected to one pin, with the devices arranged in a priority daisy chain. The CPU uses the same protocol for handling requests on these pins. The sequence of events is given below:

Any High-to-Low transition on the  $\overline{\text{NMI}}$  input is asynchronously edge-detected, and the internal  $\overline{\text{NMI}}$  latch is set. At the beginning of the last processor clock cycle of any instruction, the interrupt inputs are sampled along with the state of the internal  $\overline{\text{NMI}}$  latch.

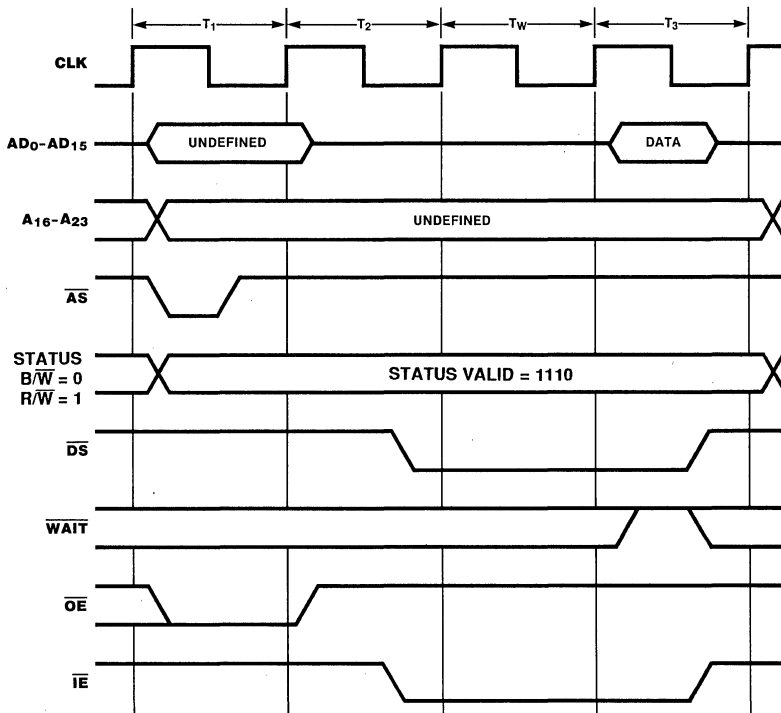


Figure 57. EPU to CPU Timing

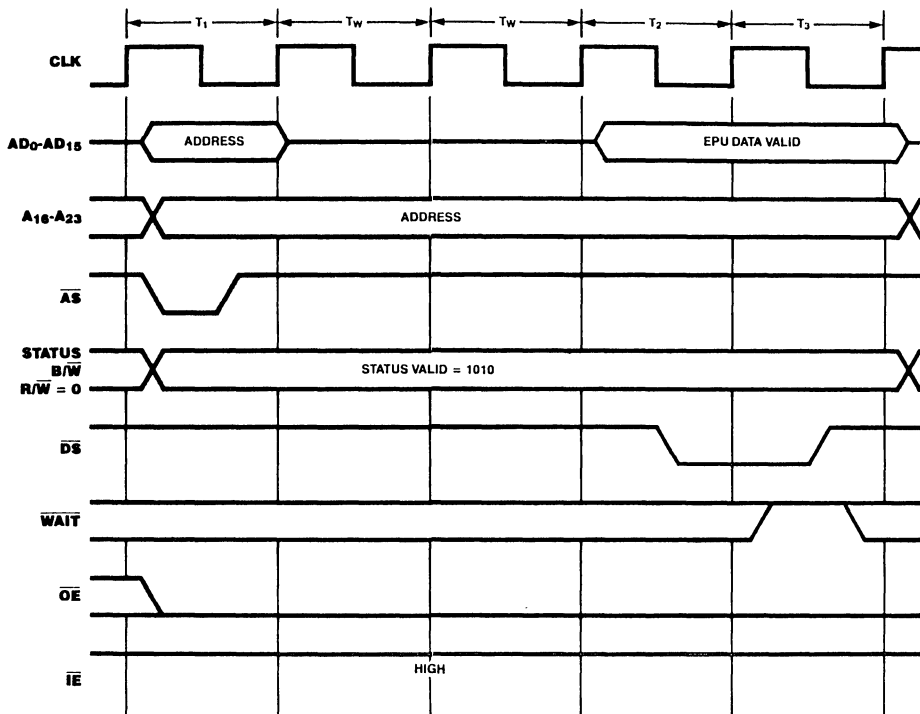


Figure 58. EPU Write to Memory

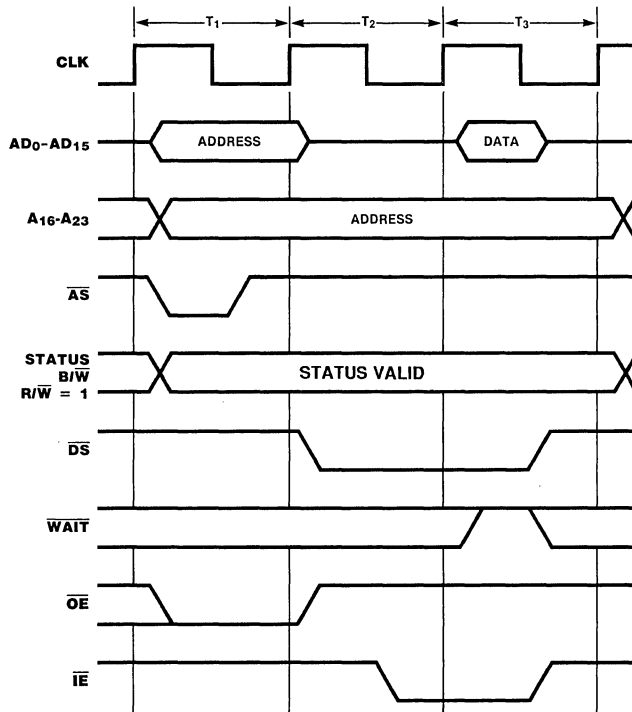


Figure 59. Memory to EPU Timing

---

If a maskable interrupt is requested and the Master Status register indicates that requests on that line are to be accepted, or if the  $\overline{\text{NMI}}$  latch is set, the next possible bus transaction is an interrupt acknowledge transaction that results in an identifier from the highest-priority interrupting device being read off the AD lines. This data is used as specified by the current interrupt mode.

**Bus Requests.** To generate transactions on the bus, a potential external bus master (such as a DMA Controller) must gain control of the bus by making a bus request. A bus request is initiated by pulling  $\overline{\text{BUSREQ}}$  Low. Several bus requesters can be wired-OR to the  $\overline{\text{BUSREQ}}$  pin; priorities are resolved externally to the CPU, usually by a priority daisy chain.

The asynchronous  $\overline{\text{BUSREQ}}$  signal generates an internal  $\overline{\text{BUSREQ}}$ , which is synchronous. If the external  $\overline{\text{BUSREQ}}$  is Low at the beginning of any processor clock cycle, the internal  $\overline{\text{BUSREQ}}$  will cause the bus acknowledge line ( $\overline{\text{BUSACK}}$ ) to be asserted after the current bus transaction is completed or after the write transaction of a TSET instruction. The CPU then enters Bus Disconnect state and gives up control of the bus. All Z280 Output pins except  $\overline{\text{BUSACK}}$  are 3-stated.

The on-chip DMA channels have higher priority than the off-chip devices requesting the external bus via  $\overline{\text{BUSREQ}}$ .

---

## RESET

A hardware reset puts the Z280 MPU into a known state and optionally initializes the Bus Timing and Initialization control register of the Z280 MPU to a system specifiable value. A reset begins at the end of any processor clock cycle if the  $\overline{\text{RESET}}$  line is Low. However, if a bus transaction is in progress it is allowed to be completed. A system reset overrides all other operations of the chip, including interrupts, traps and bus requests. A reset should be used to initialize a system, as part of the power-up sequence.

The  $\overline{\text{RESET}}$  input must be asserted for a minimum of 128 processor clock cycles. Within this time the Z280 lines assume their reset values. For either bus, the AD lines are 3-stated, and all control outputs are forced High. While  $\overline{\text{RESET}}$  is asserted, the CLK output is the processor clock frequency scaled by four.

**The  $\overline{\text{RESET}}$  line is sampled on the rising edge of an internal clock (derivative of XTALi). When the  $\overline{\text{RESET}}$  line is sampled High (de-asserted), the state of the  $\overline{\text{WAIT}}$  line is also noted: if  $\overline{\text{WAIT}}$  is asserted, then the contents of the AD lines are used to program the Bus Timing and Initialization register, otherwise the**

**constant 00 hexadecimal is used. If the hardware programming initialization option is used, AD4 must be 0 when the bus is sampled and the AD6 line determines whether the UART bootstrap option is selected.**

After reset, the Z280 MPU is initialized as shown in Tables 10 and 11.

The following registers are unaffected:

- CPU register file, including user Stack Pointer
- Page Descriptor registers
- Interrupt/Trap Vector Table Pointer register

On the rising edge of  $\overline{\text{RESET}}$ , if Bus Request is asserted the Z280 MPU will grant the bus before fetching the first instruction from location 0.

After  $\overline{\text{RESET}}$  has returned to High, the CPU begins to operate unless the Bootstrap UART feature is utilized.

**Table 10. Effect of a Reset on Z280 CPU and MMU Registers**

Register	Value Loaded on Reset (Hexadecimal)	Comments
Program Counter	0000	
System Stack Pointer	0000	
I	00	
R	00	
Master Status	0000	System mode, Single-Step disabled, Breakpoint-on-Halt disabled
Bus Timing and Control	00**	All maskable interrupts disabled No automatic wait states for I/O, upper 8M bytes of memory, or interrupt acknowledges
Bus Timing and Initialization	00	CLK output 2x processor clock period, no automatic wait states for lower 8 Mbytes of memory, bootstrap <b>mode disabled, direct clock option disabled, multiprocessor configuration disabled</b>
I/O Page	00	I/O Page 0 in use
Cache Control	20	Cache enabled for instructions All valid bits cleared to 0 Burst mode disabled
Trap Control	00	<b>EPA trap enabled, I/O not privileged, System Stack Overflow Warning trap disabled</b>
System Stack Limit	0000**	
Local Address	00	All memory transactions are made to local bus
Interrupt Status	00xx	Interrupt mode 0, nonvectored interrupts, current state of interrupt requests (indicated by xx)
Interrupt/Trap Vector Table Pointer		Unaffected
CPU Registers AF, BC, DE, HL, IX, IY, AF', BC', DE', DE', HL		Unaffected
User Stack Pointer		Unaffected
MMU Master Control	0000**	MMU disabled
MMU Page Descriptor Register, Page Descriptor Register Pointer		Unaffected

**Table 11. Effect of a Reset on Z280 On-Chip Peripheral Registers**

Register	Value Loaded on Reset (Hexadecimal)	Comments
Refresh	88	Refresh enabled, rate = 32
Counter/Timers:		
Configuration	00	<b>Timer mode, single-cycle, non-retrigger</b>
Command/Status	00	Timer disabled
DMA Channels:		
Master Control	0000***	No DMA linking, EOP disabled, Software Ready disabled
DMA0 Transaction Descriptor	0100*	DMA0 disabled, continuous mode
DMA1/2/3 Transaction Descriptor	—	EN, IE, TC, and EPS fields cleared, other fields unaffected
DMA0 Destination Address	000000	
DMA0 Count	0100	
UART:		
Configuration	00*	5 bits/character, parity disabled, external clock, x1 clock rate, loop back disabled
Transmitter Control/Status	01	Transmitter disabled, transmit buffer empty
Receiver Control/Status	00*	Receiver disabled

\*Unless bootstrap mode is selected.

\*\*Reserved bits are undefined on reads.

## ABSOLUTE MAXIMUM RATINGS

Voltage on Vcc with respect to Vss ..... -0.3V to + 7V  
 Voltages on all pins with respect to Vss ..... -0.3V to (Vcc + 0.3V)

Operating Ambient

Temperature ..... See Ordering Information

Storage Temperature ..... -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

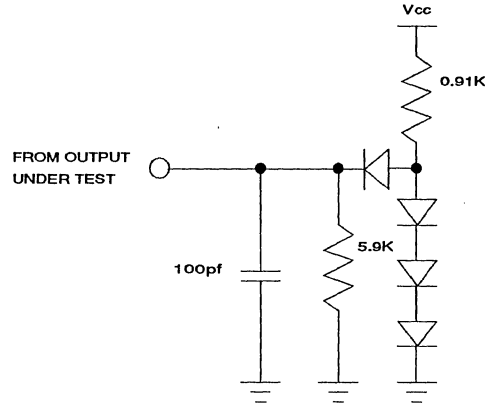
## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND (0V). Positive current flows into the referenced pin.

Available operating temperature ranges are:

- S = 0°C to +70°C

All ac parameters assume a load capacitance of 100pf. Add 10 ns delay for each 50 pf increase in load up to a maximum of 200 pf for the data bus.



## DC CHARACTERISTICS

Symbol	Parameter	Min	Max	Unit	Test Condition
$V_{IL}$	Input Low Voltage	-0.3	0.8	V	
$V_{IH}$	Input High Voltage	2.0	$V_{CC} + 0.3$	V	
$V_{OL}$	Output Low Voltage		0.4	V	$I_{OL} = 4.0 \text{ ma}$
$V_{OH}$	Output High Voltage	2.4		V	$I_{OH} = -400 \mu\text{a}$
$V_{CC}$	Operating Power Supply Voltage	4.5	5.5	V	
$I_{CC}$	Power Supply Current		200	ma	$V_{CC} = 5.5 \text{ V}$ $XTALI = 20 \text{ MHz}$ $V_{IH} = 2.0 \text{ V}$ $V_{IL} = 0.8 \text{ V}$ Outputs Unloaded

## Z280 AC CHARACTERISTICS

Z-Bus Timing (Refer to Figures 60 and 61)

No	Symbol	Parameter	10 MHz		12.5 MHz		Unit	Notes
			Min	Max	Min	Max		
1	TdCr(ST)	Clock rise to Status Delay		20		15	nS	
2	TdCr(A)	Clock rise to Address Valid Delay		20		15	nS	
3	TdCr(ASf)	Clock rise to /AS fall Delay		20		15	nS	
4	TdCf(ASr)	Clock fall to /AS rise Delay		20		15	nS	
5	TwAS	/AS Low Width	nTcXT-20		nTcXT-20		nS	1
6	TdCr(AZ)	Clock rise to Address Float Delay		25		25	nS	
7	TdCr(DSf)	Clock rise to /DS fall Delay		20		15	nS	
8	TdCf(DSr)	Clock Fall to /DS rise Delay		35		25	nS	
9	TsD(Cf)	Data to Clock fall Setup	30		30		nS	
10	ThD(Cf)	Data from Clock fall Hold	10		10		nS	
11	TdCf(DSf)	Clock fall to /DS fall Delay		20		15	nS	
12	TdCr(D)	Clock rise to Data Valid Delay		20		15	nS	
13	TdDSr(Dx)	/DS rise Data not Valid Delay	nTcXT-40		nTcXT-30		nS	1
14	TsW(Cf)	/Wait to Clock fall Setup	50		40		nS	
15	ThW(Cf)	/Wait from Clock fall Hold	0		0		nS	
16	TdCr(OEf)	Clock rise to /OE fall Delay		20		15	nS	
17	TdCr(OEr)	Clock rise to /OE rise Delay		20		15	nS	
18	TdCf(IEf)	Clock fall /IE fall Delay		20		15	nS	
19	TdCf(IEr)	Clock fall to /IE rise Delay		35		25	nS	
20	TdA(ASr)	Address Valid to /AS rise Delay	nTcXT-25		nTcXT-20		nS	1
21	TdDSr(ASf)	/DS rise to /AS fall Delay	nTcXT-40		nTcXT-25		nS	1
22	TdASr(Ax)	/AS rise to Address not Valid Delay	nTcXT-30		nTcXT-25		nS	1
24	TdDSr(A)	/DS rise to Address Active Delay	nTcXT-40		nTcXT-30		nS	1
25	TdAz(DSf)	Address Float to /DS fall Delay	0		0		nS	
26	TdD(DSf)	Data Valid to /DS fall Delay	nTcXT-20		nTcXT-20		nS	1
27	TwDSBh	/DS High Width(Burst Mode)	nTcXT-40		nTcXT-30		nS	1
28	TwDSbl	/DS Low Width(Burst Mode)	ntcXT-30		nTcXT-20		nS	1

### Notes:

- TcXT = XTALi Cycle Time  
 Clk = 1x(1x bus clock): n=1  
 2x(2x bus clock): n=2  
 4x(4x bus clock): n=4

† Units in nanoseconds unless otherwise specified.  
 $V_{M} = 2.0V$ ,  $V_{L} = 0.8V$ ,  $V_{OH} = 2.0V$ ,  $V_{OL} = 0.8V$

## Z280 AC CHARACTERISTICS

Z80-Bus Timing (Refer to Figures 62 and 63)

No	Symbol	Parameter	10 MHz		12.5 MHz		Unit	Notes
			Min	Max	Min	Max		
1	TdCr(OE)	Clock rise to /OE fall delay		20		15	nS	
2	TdCr(A)	Clock rise to Address Valid Delay		20		15	nS	
3	TdCr(AS)	Clock rise to /AS fall Delay		20		15	nS	
4	TdCr(ASr)	Clock fall to /AS rise Delay		20		15	nS	
5	TwAS	/AS Low Width	nTcXT-20		nTcXT-20		nS	1
6	TdCr(AZ)	Clock rise to Address Float Delay		25		25	nS	
7	TsW(CI)	/Wait to Clock fall setup	50		40		nS	
8	ThW(CI)	/Wait from Clock fall hold	0		0		nS	
9	TdA(ASr)	Address Valid to /AS rise delay	nTcXT-25		nTcXT-20		nS	1
10	TdASr(Ax)	/AS rise to Address not Valid Delay	nTcXT-30		nTcXT-25		nS	1
11	TdCr(RD)	Clock rise to /RD fall delay		20		15	nS	
12	TdCr(RDr)	Clock fall to /RD rise Delay		35		25	nS	
14	TsD(CI)	Data to Clock fall Setup	30		30		nS	
15	ThD(CI)	Data from Clock fall Hold	10		10		nS	
16	TdAz(RD)	Address Float to /RD fall Delay	0		0		nS	
19	TdCr(OEr)	Clock rise to /OE rise Delay		20		15	nS	
20	TdCr(IEf)	Clock fall to /IE fall Delay		20		15	nS	
21	TdCr(IEr)	Clock fall to /IE rise Delay		35		25	nS	
22	TdCr(IEr)	Clock rise to /IE rise Delay		20		15	nS	2
23	TdCr(RDr)	Clock rise to /RD rise Delay		20		15	nS	2
24	TdCr(WR)	Clock fall to /WR fall Delay		20		15	nS	
25	TdCr(WRr)	Clock fall to /WR rise Delay		35		25	nS	
26	TdWRr(AS)	/WR rise to /AS fall Delay	nTcXT-40		nTcXT-30		nS	1
27	TdWRr(A)	/WR rise to Address active Delay	nTcXT-40		nTcXT-30		nS	1
28	TdCr(D)	Clock rise to Data Valid Delay		20		15	nS	
29	TdWRr(Dx)	/WR rise to Data not Valid Delay	nTcXT-40		nTcXT-30		nS	1
30	Td(WRf)	Data Valid to /WR fall Delay	nTcXT-20		nTcXT-20		nS	1
31	TdCr(MREQf)	Clock fall to /MREQ fall Delay		20		15	nS	
32	TdCr(MREQr)	Clock fall to /MREQ rise Delay		35		25	nS	
33	TdCr(MREQr)	Clock rise to /MREQ rise Delay		20		15	nS	2
34	TdCr(IORQf)	Clock rise to /IORQ fall Delay		20		15	nS	
35	TdCr(IORQr)	Clock fall to /IORQ rise Delay		35		25	nS	
36	TdCr(IORQf)	Clock fall to /IORQ fall Delay		20		15	nS	3
37	TdCr(M1r)	Clock fall to /M1 rise Delay		35		25	nS	3
38	TdCr(M1r)	Clock rise to /M1 rise Delay		20		15	nS	
39	TdCr(M1f)	Clock rise to /M1 fall Delay		20		15	nS	2
40	TdCr(RFSHr)	Clock fall to /RFSH rise Delay		35		25	nS	2,3
41	TdCr(RFSHf)	Clock fall to /RFSH fall Delay		20		15	nS	
42	TdCr(HALTf)	Clock fall to /HALT fall Delay		20		15	nS	

### Notes:

1. TcXT = XTALi Cycle Time

CIk = 1x(1x bus clock) : n=1

2x(2x bus clock) : n=2

4x(4x bus clock) : n=4

2. This parameter is used for RETI (Return From Interrupt).

3. This parameter is used for Interrupt Acknowledge.

† Units in nanoseconds unless otherwise specified.

$V_{BI}=2.0V$ ,  $V_L=0.8V$ ,  $V_{OI}=2.0V$ ,  $V_{\alpha}=0.8V$

## Z280 AC CHARACTERISTICS

Z-Bus, Z80 Bus Common Signals and Peripherals Timing  
(Refer to Figures 64 through 71)

No	Symbol	Parameter	10 MHz		12.5 MHz		Unit	Notes
			Min	Max	Min	Max		
1	TcXT	XTALi Cycle time	50	lbd	40	lbd	nS	
2	TwXTh	XTALi High Width	15		15		nS	
3	TwXTl	XTALi Low Width	15		15		nS	
4	TrXT	XTALi Rise Time		10		10	nS	
5	TfXT	XTALi Fall Time		10		10	nS	
6	TdXT(C)	XTAL fall to Clock Delay		40		40	nS	
7	TrC	Clock rise time		12		10	nS	
8	TfC	Clock fall time		12		10	nS	
9	TdCr(CSt)	Clock rise to /DS, /RD, or /WR fall Delay		20		15	nS	
10	TdCr(CSr)	Clock rise to /DS, or /WR rise Delay		20		15	nS	
11	TdCr(STBf)	Clock rise to /DMASTB fall delay		20		15	nS	
12	TdCr(STBr)	Clock fall to /DMASTB rise Delay		35		25	nS	
13	TdCr(STBr)	Clock rise to /DMASTB rise Delay		20		15	nS	
14	TdCr(CSr)	Clock fall to /DS or /RD Rise Delay		35		25	nS	
15	TdCr(GREQf)	Clock fall to /GREQ Fall Delay		35		25	nS	
16	TdCr(GREQr)	Clock fall to /GREQ rise Delay		35		25	nS	
17	TdCr(BUSACKf)	Clock rise to /BUSACK fall Delay		20		15	nS	
18	TdCr(BUSACKr)	Clock rise to /BUSACK rise Delay		20		15	nS	
19	TcCTIN	CTIN Cycle Time	10TcXT		10TcXT		nS	
20	TwCTINh	CTIN High Width	4TcXT		4TcXT		nS	
21	TwCTINl	CTIN Low Width	4TcXT		4TcXT		nS	
22	TwCTIOh	CTIO High Width	4TcXT		4TcXT		nS	1
23	TwCTIOl	CTIO Low Width	4TcXT		4TcXT		nS	1
24	TdCTIN(CTIO)	CTIN to CTIO Delay	20TcXT	28TcXT	20TcXT	28TcXT	nS	2
25	TdCr(TD)	Baud Clock fall to Transmit Data Delay		70		70	nS	3
26	TsRD(Cr)	Receive Data to Baud Clock rise Setup	10		10		nS	3
27	ThRD(Cr)	Receive Data from Baud Clock rise Hold	50		50		nS	3
28	TrRESET	/Reset Rise Time		10		10	nS	
29	TfRESET	/Reset Fall Time		10		10	nS	
30	TsWAITr(RESETr)	/WAIT fall to /RESET rise Setup	4TcXT		4TcXT		nS	4
31	ThWAITr(RESETr)	/WAIT rise to /RESET rise Hold	6TcXT		6TcXT		nS	4
32	TsD(RESETr)	Data to /RESET rise Setup	0		0		nS	4
33	ThD(RESETr)	Data from /RESET rise Hold	6TcXT		6TcXT		nS	4
34	TrIN	Input Rise Time		20		20	nS	5
35	TfIN	Input Fall Time		20		20	nS	5
36	TwNMI	/NMI Low Width	4TcXT		4TcXT		nS	

### Notes:

1. CTIO as Gate or Trigger Input.
2. CTIO as Output, when CTIN causes terminal count.
3. CTIN1 as X1 Baud Clock Input. Refer to specs 20 and 21 for pulse widths.
4. To program Bus Timing and Initialization Register at reset.
5. Inputs AD, /BUSREQ, CTIN, CTIO, /INT, /NMI, /RDY, RxD, /PAUSE and /WAIT.

† Units in nanoseconds unless otherwise specified.  
V<sub>DD</sub>=2.0V, V<sub>z</sub>=0.8V, V<sub>OH</sub>=2.0V, V<sub>OL</sub>=0.8V



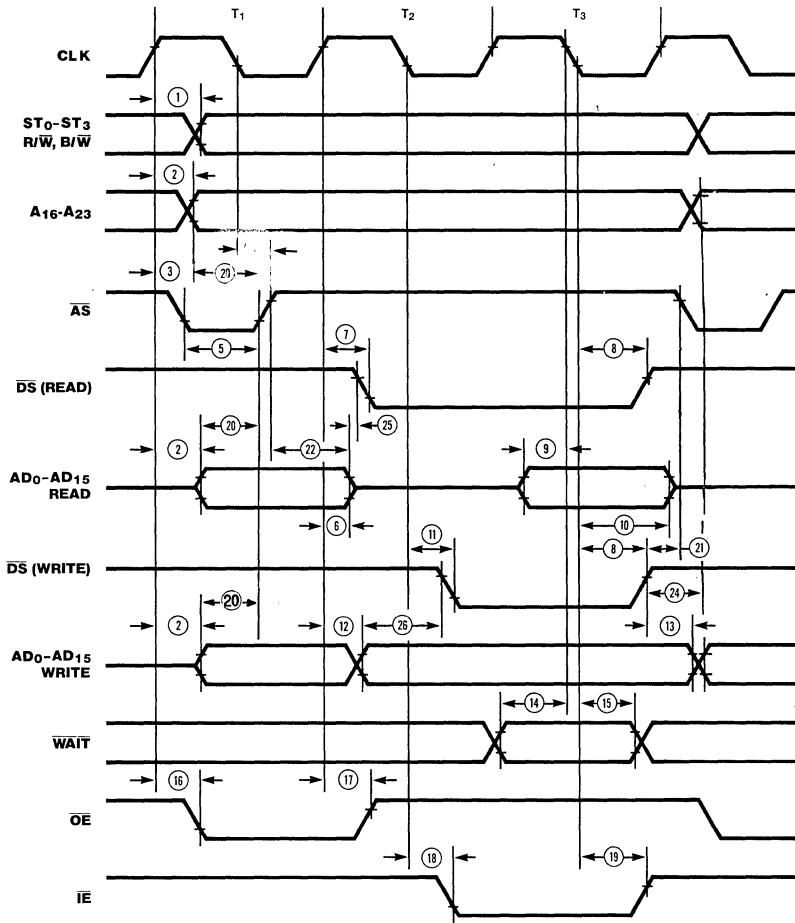


Figure 60. Z-Bus All Transactions

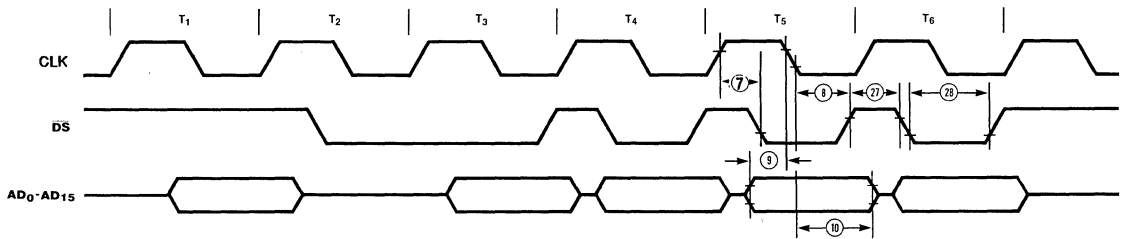


Figure 61. Z-Bus Burst Mode Timing

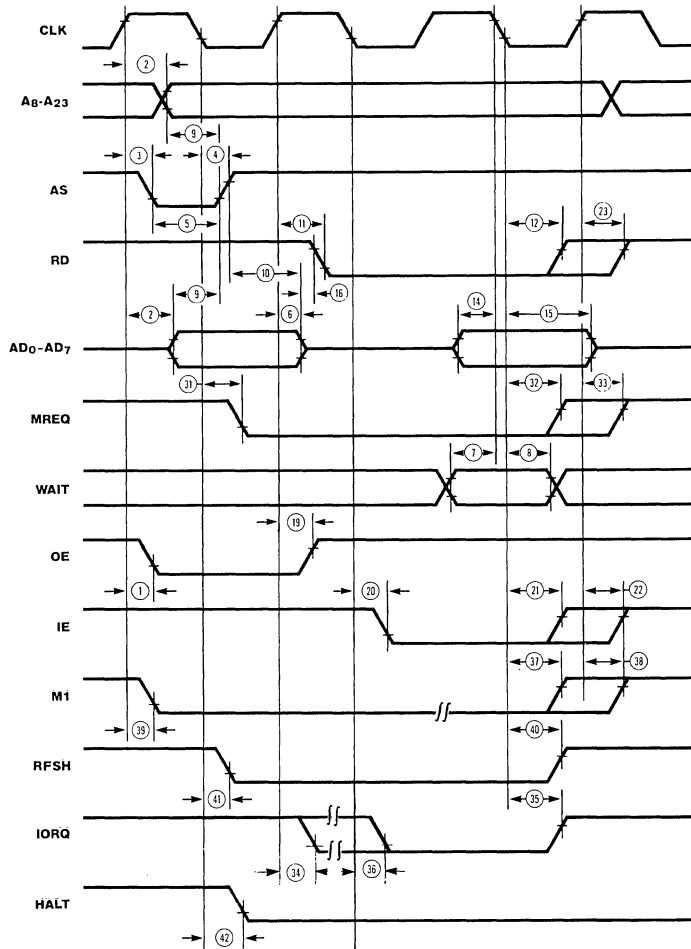


Figure 62. Z80 Bus Read Type Transactions

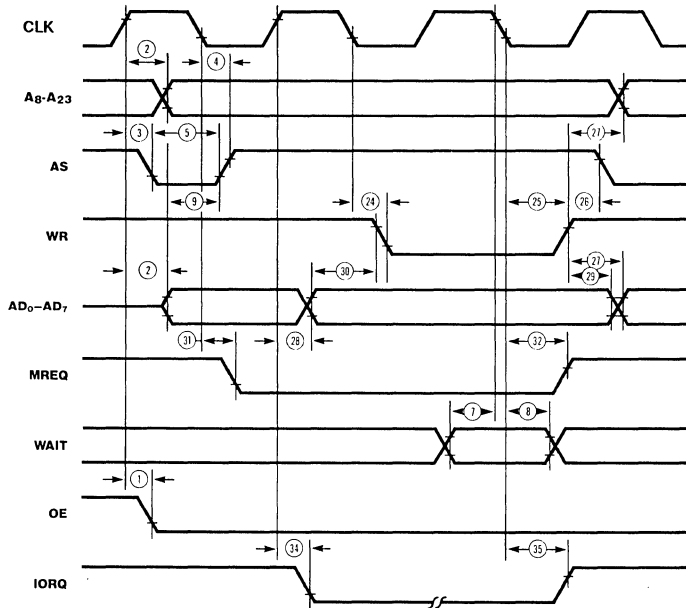


Figure 63. Z80 Bus Write Transactions

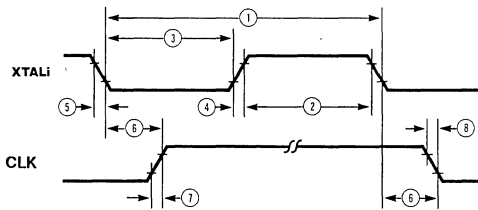


Figure 64. Z800 Clock Circuit

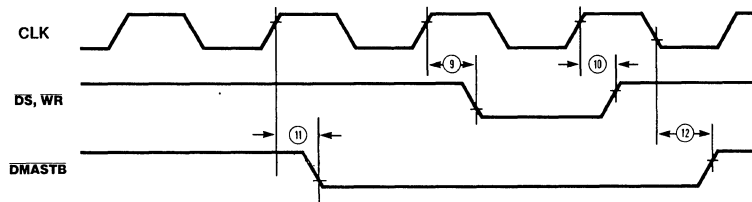


Figure 65. Flyby DMA Write to Memory  
(Z-Bus:  $\overline{DS}$ ; Z80 Bus:  $\overline{WR}$ )

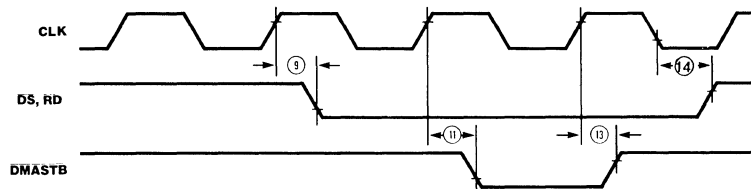


Figure 66. Flyby DMA Read from Memory  
(Z-Bus:  $\overline{DS}$ ; Z80 Bus:  $\overline{RD}$ )

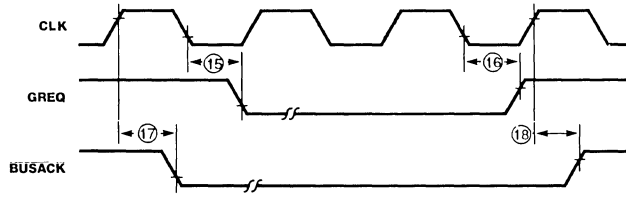


Figure 67.  $\overline{\text{GREQ}}$  and  $\overline{\text{BUSACK}}$  Timing

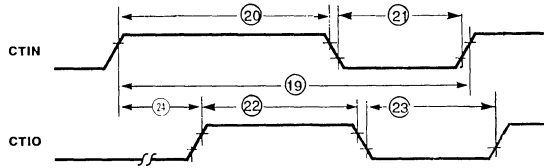


Figure 68. Counter/Timer Timing

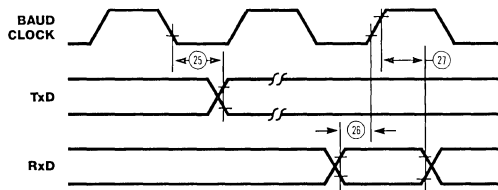


Figure 69. UART Timing

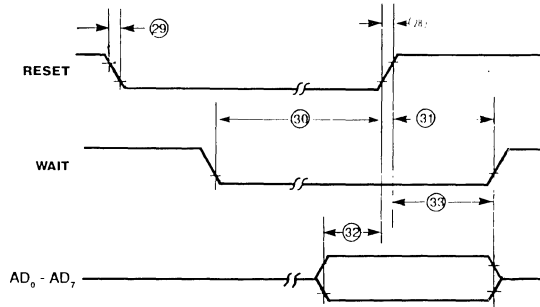


Figure 70. Reset Timing

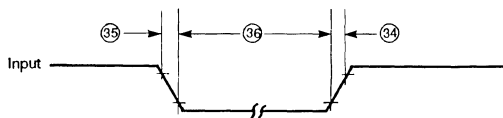


Figure 71. Inputs Timing





# **SERIAL COMMUNICATIONS CONTROLLERS**

**SERIAL COMMUNICATIONS  
CONTROLLERS**

---



# Z8030/Z8530

## Z-BUS SCC SERIAL COMMUNICATION CONTROLLER

**Features**

- Z8530 optimized for non-multiplexed Bus microprocessors.
- Z8030 optimized for multiplexed Bus microprocessors.
- Two independent, 0 to 2M bit/second, full-duplex channels, each with a separate crystal oscillator, baud rate generator, and Digital Phase-Locked Loop for clock recovery.
- Multi-protocol operation under program control; programmable for NRZ, NRZI, or FM data encoding.
- Asynchronous mode with five to eight bits and one, one and one-half, or two stop bits per character; programmable clock factor; break detection and generation; parity, overrun, and framing error detection.
- Synchronous mode with internal or external character synchronization on one or two synchronous characters and CRC generation and checking with CRC-16 or CRC-CCITT preset to either 1s or 0s.
- SDLC/HDLC mode with comprehensive frame-level control, automatic zero insertion and deletion, I-field residue handling, abort generation and detection, CRC generation and checking, and SDLC Loop mode operation.
- Local Loopback and Auto Echo modes.
- Supports T1 digital trunk.
- Speeds 4, 6, 8 MHz.

**General Description**

The SCC Serial Communications Controller is a dual-channel, multi-protocol data communications peripheral designed for use with conventional non-multiplexed buses and the Zilog Z-BUS.® The SCC functions as a serial-to-parallel, parallel-to-serial converter/controller. The SCC can be software-configured to satisfy a wide variety of serial communications applications. The

device contains a variety of new, sophisticated internal functions including on-chip baud rate generators, Digital PhaseLocked Loops, and crystal oscillators that dramatically reduce the need for external logic.

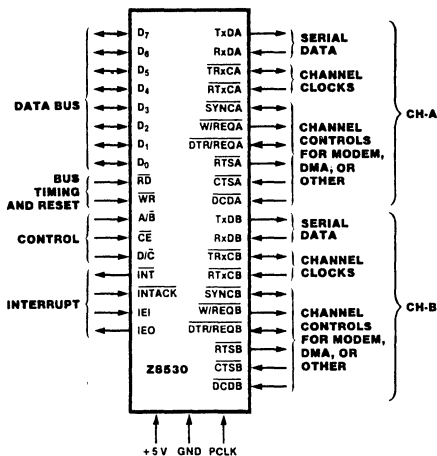


Figure 1a. Pin Functions, Z8530

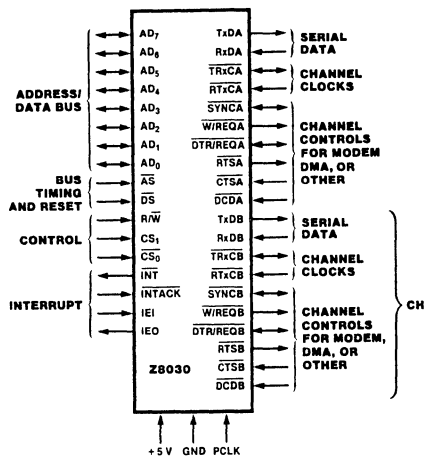


Figure 1b. Pin Functions, Z8030



**General Description**  
(Continued)

The SCC handles asynchronous formats, Synchronous byte-oriented protocols such as IBM Bisync, and Synchronous bit-oriented protocols such as HDLC and IBM SDLC. This versatile device supports virtually any serial data transfer application (cassette, diskette, tape drives, etc.).

The device can generate and check CRC codes in any Synchronous mode and can be programmed to check data integrity in various

modes. The SCC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

The daisy-chain interrupt hierarchy is also supported—as is standard for Zilog peripheral components.

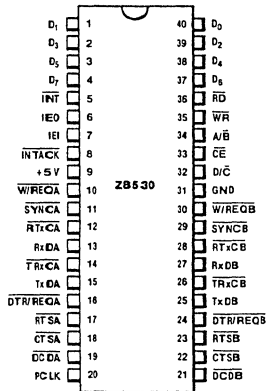


Figure 2a. DIP Pin Assignments, Z8530

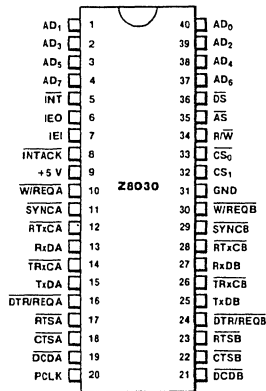


Figure 2b. DIP Pin Assignments, Z8030

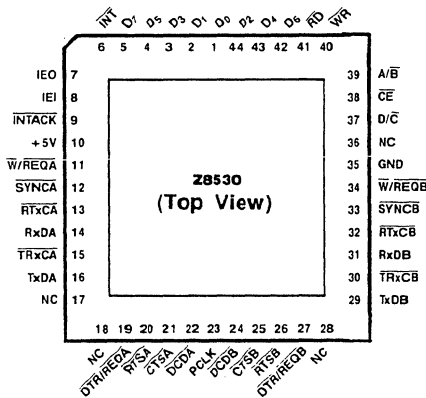


Figure 2c. Chip Carrier Pin Assignments, Z8530

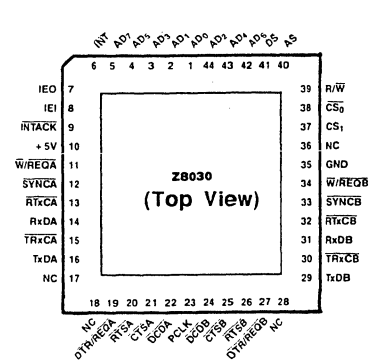


Figure 2d. Chip Carrier Pin Assignments, Z8030

Note: Power connections follow conventional descriptions below:

Connection	Circuit	Device
Power	V <sub>CC</sub>	V <sub>DD</sub>
Ground	GND	V <sub>SS</sub>

**Pin  
Description**

The following section describes the pin functions common to the Z8530 and the Z8030. Figures 1 and 2 detail the respective pin functions and pin assignments:

**CTSA, CTSB.** *Clear To Send* (inputs, active Low). If these pins are programmed as Auto Enables, a Low on the inputs enables the respective transmitters. If not programmed as Auto Enables, they may be used as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The SCC detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

**DCDA, DCDB.** *Data Carrier Detect* (inputs/outputs, active Low). These pins function as receiver enables if they are programmed for Auto Enables; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow rise-time signals. The SCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

**DTR/REQA, DTR/REQB.** *Data Terminal Ready/Request* (outputs, active Low). These outputs follow the state programmed into the DTR bit. They can also be used as general-purpose outputs or as Request lines for a DMA controller.

**IEI.** *Interrupt Enable In* (input, active High). IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt-driven device. A High IEI indicates that no other higher priority device has an interrupt under service or is requesting an interrupt.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing an SCC interrupt or the SCC is not requesting an interrupt (Interrupt Acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

**INT.** *Interrupt Request* (output, open-drain, active Low). This signal is activated when the SCC requests an interrupt.

**INTACK.** *Interrupt Acknowledge* (input, active Low). This signal indicates an active Interrupt Acknowledge cycle. During this cycle, the SCC interrupt daisy chain settles. When  $\overline{RD}$  or  $\overline{DS}$  becomes active, the SCC places an interrupt vector on the data bus (if IEI is High).  $\overline{INTACK}$  is latched by the rising edge of PCLK.

**PCLK.** *Clock* (input). This is the master SCC clock used to synchronize internal signals. PCLK is a TTL level signal. PCLK is not required to have any phase relationship with the master system clock.

**RxDA, RxDB.** *Receive Data* (inputs, active High). These input signals receive serial data at standard TTL levels.

**RTxCA, RTxCB.** *Receive/Transmit Clocks* (inputs, active Low). These pins can be programmed in several different modes of operation. In each channel,  $\overline{RTxC}$  may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the Digital Phase-Locked Loop. These pins can also be programmed for use with the respective  $\overline{SYNC}$  pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in Asynchronous modes.

**RTSA, RTSB.** *Request To Send* (outputs, active Low). When the Request To Send (RTS) bit in Write Register 5 (Figure 11) is set, the  $\overline{RTS}$  signal goes Low. When the RTS bit is reset in the Asynchronous mode and Auto Enable is on, the signal goes High after the transmitter is empty. In Synchronous mode or in Asynchronous mode with Auto Enable off, the RTS pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

**SYNCA, SYNCB.** *Synchronization* (inputs or outputs, active Low). These pins can act either as inputs, outputs, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to CTS and DCD. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 (Figure 10) but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode,  $\overline{SYNC}$  must be driven Low two receive clock cycles after the last bit in the synchronous character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of  $\overline{SYNC}$ .

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which synchronous characters are recognized. The synchronous condition is not latched, so these outputs are active each time a synchronization pattern is recognized (regardless of character boundaries). In SDL mode, these pins act as outputs and are valid on receipt of a flag.

**TxDA, TxDB.** *Transmit Data* (outputs, active High). These output signals transmit serial data at standard TTL levels.

**TRxCA, TRxCB.** *Transmit/Receive Clocks* (inputs or outputs, active Low). These pins can be programmed in several different modes of operation. TRxC may supply the receive clock or the transmit clock in the input mode or supply the output of the Digital Phase-Locked Loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

**W/REQA, W/REQB.** *Wait/Request* (outputs, open-drain when programmed for a Wait function, driven High or Low when programmed for a Request function). These dual-purpose outputs may be programmed as Request lines for a DMA controller or as Wait lines to synchronize the CPU to the SCC data rate. The reset state is Wait.

### Z8530

**A/B.** *Channel A/Channel B Select* (input). This signal selects the channel in which the read or write operation occurs.

**CE.** *Chip Enable* (input, active Low). This signal selects the SCC for a read or write operation.

**D<sub>0</sub>-D<sub>7</sub>** *Data Bus* (bidirectional, 3-state). These lines carry data and commands to and from the SCC.

**D/C.** *Data/Control Select* (input). This signal defines the type of information transferred to or from the SCC. A High means data is transferred; a Low indicates a command.

**RD.** *Read* (input, active Low). This signal indicates a read operation and when the SCC is selected, enables the SCC's bus drivers. During

the Interrupt Acknowledge cycle, this signal gates the interrupt vector onto the bus if the SCC is the highest priority device requesting an interrupt.

**WR.** *Write* (input, active Low). When the SCC is selected, this signal indicates a write operation. The coincidence of RD and WR is interpreted as a reset.

### Z8030

**AD<sub>0</sub>-AD<sub>7</sub>.** *Address/Data Bus* (bidirectional, active High, 3-state). These multiplexed lines carry register addresses to the SCC as well as data or control information.

**AS.** *Address Strobe* (input, active Low). Addresses on AD<sub>0</sub>-AD<sub>7</sub> are latched by the rising edge of this signal.

**CS<sub>0</sub>.** *Chip Select 0* (input, active Low). This signal is latched concurrently with the addresses on AD<sub>0</sub>-AD<sub>7</sub> and must be active for the intended bus transaction to occur.

**CS<sub>1</sub>.** *Chip Select 1* (input, active High). This second select signal must be active before the intended bus transaction can occur. CS<sub>1</sub> must remain active throughout the transaction.

**DS.** *Data Strobe* (input, active Low). This signal provides timing for the transfer of data into and out of the SCC. If AS and DS coincide, this is interpreted as a reset.

**R/W.** *Read/Write* (input). This signal specifies whether the operation to be performed is a read or a write.

## Functional Description

The functional capabilities of the SCC can be described from two different points of view: as a data communications device, it transmits and receives data in a wide variety of data communications protocols; as a microprocessor peripheral, the SCC offers valuable features such as vectored interrupts, polling, and simple handshake capability.

**Data Communications Capabilities.** The SCC provides two independent full-duplex channels programmable for use in any common Asynchronous or Synchronous data-communication protocol. Figure 3 and the following description briefly detail these protocols.

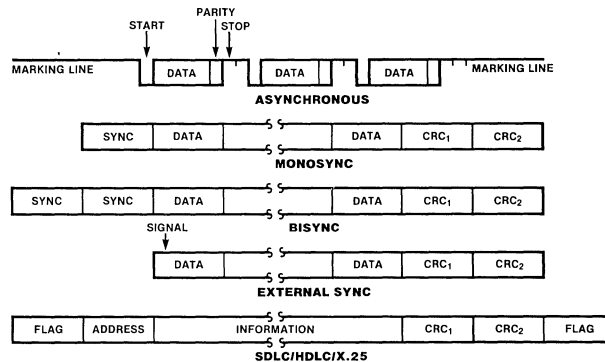


Figure 3. Some SCC Protocols

**Functional Description**  
(Continued)

*Asynchronous Modes.* Transmission and reception can be accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half, or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB in Figure 1). If the Low does not persist (as in the case of a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur. Vectored interrupts allow fast servicing or error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

The SCC does not require symmetric transmit and receive clock signals—a feature allowing use of the wide variety of clock sources. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32, or 1/64 of the clock rate supplied to the receive and transmit clock inputs. In Asynchronous modes, the SYNC pin may be programmed as an input used for functions such as monitoring a ring indicator.

*Synchronous Modes.* The SCC supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols can be handled in several modes, allowing character synchronization with a 6-bit or 8-bit synchronous character (Monosync), any 12-bit synchronization pattern (Bisync), or with an external synchronous signal. Leading sync characters can be removed without interrupting the CPU.

Five- or 7-bit synchronous characters are detected with 8- or 16-bit patterns in the SCC by overlapping the larger pattern across multiple incoming synchronous characters as shown in Figure 4.

CRC checking for Synchronous byte-oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols such as IBM Bisync.

Both CRC-16 ( $X^{16} + X^{15} + X^2 + 1$ ) and CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) error checking polynomials are supported. Either polynomial may be selected in all Synchronous modes. Users may preset the CRC generator and checker to all 1s or all 0s. The SCC also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows for high speed transmissions under DMA control, with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 6-, 8-, or 16-bit synchronous characters, regardless of the programmed character length.

The SCC supports Synchronous bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero insertion, and CRC generation. A special command can be used to abort a frame in transmission. At the end of a message, the SCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line consisting of continuous flag characters or a steady marking condition.

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. The SCC may also be programmed to send an abort itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent, allowing reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the SYNC pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all 0s inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of

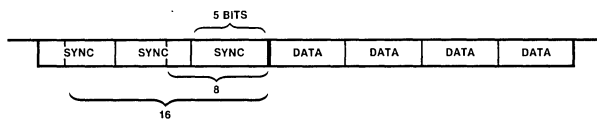


Figure 4. Detecting 5- or 7-Bit Synchronous Characters

**Functional Description**  
(Continued)

transmission, the status of a received frame is available in the status registers. In SDLC mode, the SCC must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all 1s or all 0s. The CRC is inverted before transmission and the receiver checks against the bit pattern 0001110100001111.

NRZ, NRZI or FM coding may be used in any 1x mode. The parity options available in Asynchronous modes are available in Synchronous modes.

The SCC can be conveniently used under DMA control to provide high speed reception or transmission. In reception, for example, the SCC can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The SCC then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received. The CPU may also enable the DMA first and have the SCC interrupt only on end-of-frame. This procedure allows all data to be transferred via the DMA.

**SDLC Loop Mode.** The SCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow on the loop and any number of secondary stations. In SDLC Loop mode, the SCC performs the functions of a secondary station while an SCC operating in regular SDLC mode can act as a controller (Figure 5).

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop, and in fact must pass these messages to the rest of the loop by retransmitting them with a one-bit-time delay. The secondary station can place its own message on the loop only at specific times. The controller signals that secondary stations may transmit messages by sending a special character, called an EOP (End Of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages, this bit pattern is unique and easily recognized.

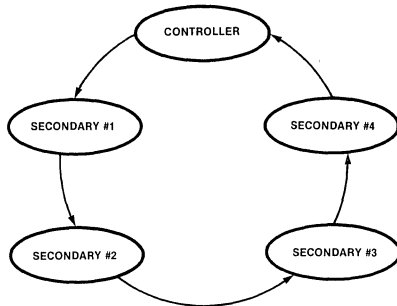


Figure 5. An SDLC Loop

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit can then append their messages to the message of the first secondary station by the same process. Any secondary stations without messages to send merely echo the incoming messages and are prohibited from placing messages on the loop (except upon recognizing an EOP).

SDLC Loop mode is a programmable option in the SCC. NRZ, NRZI, and FM coding may all be used in SDLC Loop mode.

**Baud Rate Generator.** Each channel in the SCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On startup, the flip-flop on the output is set in a High state, the value in the time constant register is loaded into the counter, and the counter starts counting down. The output of the baud rate generator toggles upon reaching 0, the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the Digital Phase-Locked Loop (see next section).

If the receive clock or transmit clock is not programmed to come from the TRxC pin, the output of the baud rate generator may be echoed out via the TRxC pin.

The following formula relates the time constant to the baud rate where PCLK or RTxC is the baud rate generator input frequency in Hz. The clock mode is 1, 16, 32, or 64 as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select 1 and Asynchronous should select 16, 32, or 64.

$$\text{Time Constant} = \frac{\text{PCLK or RTxC Frequency}}{2 (\text{Baud Rate}) (\text{Clock Mode})} - 2$$

**Digital Phase-Locked Loop.** The SCC contains a Digital Phase-Locked-Loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock may then be used as the SCC receive clock, the transmit clock, or both.

For NRZI encoding, the DPLL counts the 32x clock to create nominal bit times. As the 32x clock is counted, the DPLL is searching the

**Functional Description**  
(Continued)

incoming data stream for edges (either 1 to 0 or 0 to 1). Whenever an edge is detected, the DPLL makes a count adjustment (during the next counting cycle), producing a terminal count closer to the center of the bit cell.

For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16, and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15 to 16 counting transition.

The 32x clock for the DPLL can be programmed to come from either the  $RTxC$  input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the SCC via the  $TRxC$  pin (if this pin is not being used as an input).

**Data Encoding.** The SCC may be programmed to encode and decode the serial data in four different ways (Figure 6). In NRZ encoding, a 1 is represented by a High level and a 0 is represented by a Low level. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. In FM1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A 1 is represented by an additional transition at the center of the bit cell and a 0 is represented by no additional transition at the center of the bit cell. In FM0 (bi-phase space), a transition occurs at the beginning of every bit cell. A 0 is represented by an additional transition at the center of the bit cell, and a 1 is represented by no additional transition at the center of the bit cell. In addition to these four methods, the SCC can be used to decode Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0 to 1, the bit is a 0. If the transition is 1 to 0, the bit is a 1.

**Auto Echo and Local Loopback.** The SCC is capable of automatically echoing everything it receives. This feature is useful mainly in Asynchronous modes, but works in Synchronous and SDLC modes as well. In Auto Echo mode, TxD is RxD. Auto Echo mode can be used with NRZI or FM encoding with no additional delay, because the data stream is not decoded before retransmission. In Auto Echo mode, the CTS input is ignored as a transmitter enable (although transitions on this input can still cause interrupts if programmed to do so). In this mode, the transmitter is actually bypassed and the programmer is responsible for disabling transmitter interrupts and WAIT/REQUEST on transmit.

The SCC is also capable of local loopback. In this mode TxD is RxD, just as in Auto Echo mode. However, in Local Loopback mode, the internal transmit data is tied to the internal receive data and RxD is ignored (except to be echoed out via TxD). The CTS and DCD inputs are also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in Asynchronous, Synchronous and SDLC modes with NRZ, NRZI or FM coding of the data stream.

**I/O Interface Capabilities.** The SCC offers the choice of Polling, Interrupt (vectored or nonvectored), and Block Transfer modes to transfer data, status, and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

**Polling.** All interrupts are disabled. Three status registers in the SCC are automatically updated whenever any function is performed. For example, end-of-frame in SDLC mode sets a bit in one of these status registers. The idea behind polling is for the CPU to periodically read a status register until the register contents indicate the need for data to be transferred. Only one register needs to be

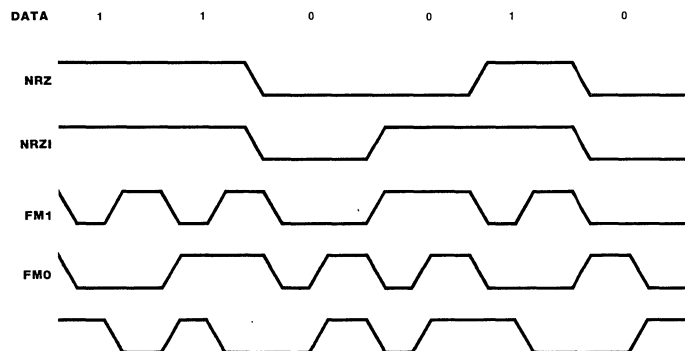


Figure 6. Data Encoding Methods

**Functional Description**  
(Continued)

read; depending on its contents, the CPU either writes data, reads data, or continues. Two bits in the register indicate the need for data transfer. An alternative is a poll of the Interrupt Pending register to determine the source of an interrupt. The status for both channels resides in one register.

**Interrupts.** When an SCC responds to an Interrupt Acknowledge signal ( $\overline{INTACK}$ ) from the CPU, an interrupt vector may be placed on the data bus. This vector is written in WR2 and may be read in RR2A or RR2B (Figures 10 and 11).

To speed interrupt response time, the SCC can modify three bits in this vector to indicate status. If the vector is read in Channel A, status is never included; if it is read in Channel B, status is always included.

Each of the six sources of interrupts in the SCC (Transmit, Receive, and External/Status interrupts in both channels) has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt Under Service (IUS), and Interrupt Enable (IE). Operation of the IE bit is straightforward. If the IE bit is set for a given interrupt source, then that source can request interrupts. The exception is when the MIE (Master Interrupt Enable) bit in WR9 is reset and no interrupts may be requested. The IE bits are write only.

The other two bits are related to the interrupt priority chain (Figure 7). As a microprocessor peripheral, the SCC may request an interrupt only when no higher priority device is requesting one, e.g., when IEI is High. If the device in question requests an interrupt, it pulls down  $\overline{INT}$ . The CPU then responds with  $\overline{INTACK}$ , and the interrupting device places the vector on the data bus.

In the SCC, the IP bit signals a need for interrupt servicing. When an IP bit is 1 and the IEI input is High, the  $\overline{INT}$  output is pulled Low, requesting an interrupt. In the SCC, if the IE bit is not set by enabling interrupts, then the IP for that source can never be set. The IP bits are readable in RR3A.

The IUS bits signal that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority in the SCC and

external to the SCC are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the SCC being pulled Low and propagated to subsequent peripherals. An IUS bit is set during an Interrupt Acknowledge cycle if there are no higher priority devices requesting interrupts.

There are three types of interrupts: Transmit, Receive, and External/Status. Each interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receiver, Transmit, and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted when the transmit buffer becomes empty. (This implies that the transmitter must have had a data character written into it so that it can become empty.) When enabled, the receiver can interrupt the CPU in one of three ways:

- Interrupt on First Receive Character or Special Receive Condition.
- Interrupt on All Receive Characters or Special Receive Condition.
- Interrupt on Special Receive Condition Only.

Interrupt on First Character or Special Condition and Interrupt on Special Condition Only are typically used with the Block Transfer mode. A Special Receive Condition is one of the following: receiver overrun, framing error in Asynchronous mode, end-of-frame in SDLC mode and, optionally, a parity error. The Special Receive Condition interrupt is different from an ordinary receive character available interrupt only in the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt can occur from Special Receive Conditions any time after the first receive character interrupt.

The main function of the External/Status interrupt is to monitor the signal transitions of the CTS, DCD, and SYNC pins; however, an

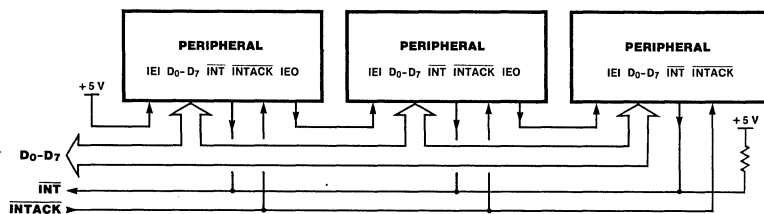


Figure 7. Interrupt Schedule

**Functional Description**  
(Continued)

External/Status interrupt is also caused by a Transmit Underrun condition, or a zero count in the baud rate generator, or by the detection of a Break (Asynchronous mode), Abort (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the Abort or EOP has a special feature allowing the SCC to interrupt when the Abort or EOP sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Abort condition in external logic in SDLC mode. In SDLC Loop mode, this feature allows secondary stations to recognize the wishes of the primary station to regain control of the loop during a poll sequence.

**CPU/DMA Block Transfer.** The SCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the  $\overline{\text{WAIT/REQUEST}}$  output in conjunction with the Wait/Request bits in WR1. The  $\overline{\text{WAIT/REQUEST}}$  output can be defined under software control as a WAIT line in the CPU Block Transfer mode or as a  $\overline{\text{REQUEST}}$  line in the DMA Block Transfer mode.

To a DMA controller, the  $\overline{\text{SCC REQUEST}}$  output indicates that the SCC is ready to transfer data to or from memory. To the CPU, the WAIT line indicates that the SCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The DTR/ $\overline{\text{REQUEST}}$  line allows full-duplex operation under DMA control.

**Architecture**

The SCC internal structure includes two full-duplex channels, two baud rate generators, internal control and interrupt logic, and a bus interface to a nonmultiplexed bus. Associated with each channel are a number of read and write registers for mode control and status information, as well as logic necessary to interface to modems or other external devices (Figure 8).

The logic for both channels provides formats, synchronization, and validation for data transferred to and from the channel interface. The modem control inputs are monitored

by the control logic under program control. All of the modem control signals are general-purpose in nature and can optionally be used for functions other than modem control.

The register set for each channel includes ten control (write) registers, two sync-character (write) registers, and four status (read) registers. In addition, each baud rate generator has two (read/write) registers for holding the time constant that determines the baud rate. Finally, associated with the interrupt logic is a write register for the interrupt vector accessible through either channel, a

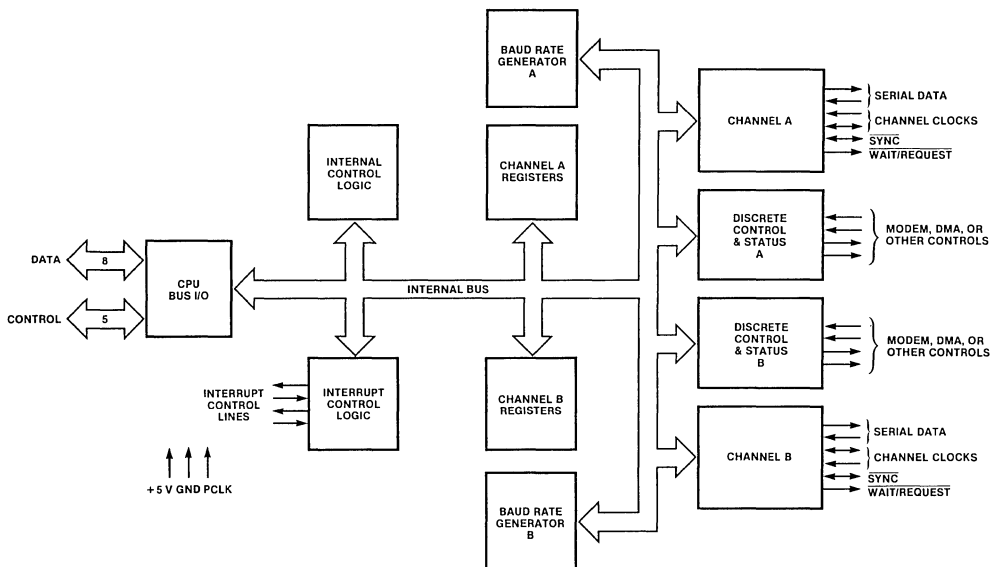


Figure 8. Block Diagram of SCC Architecture



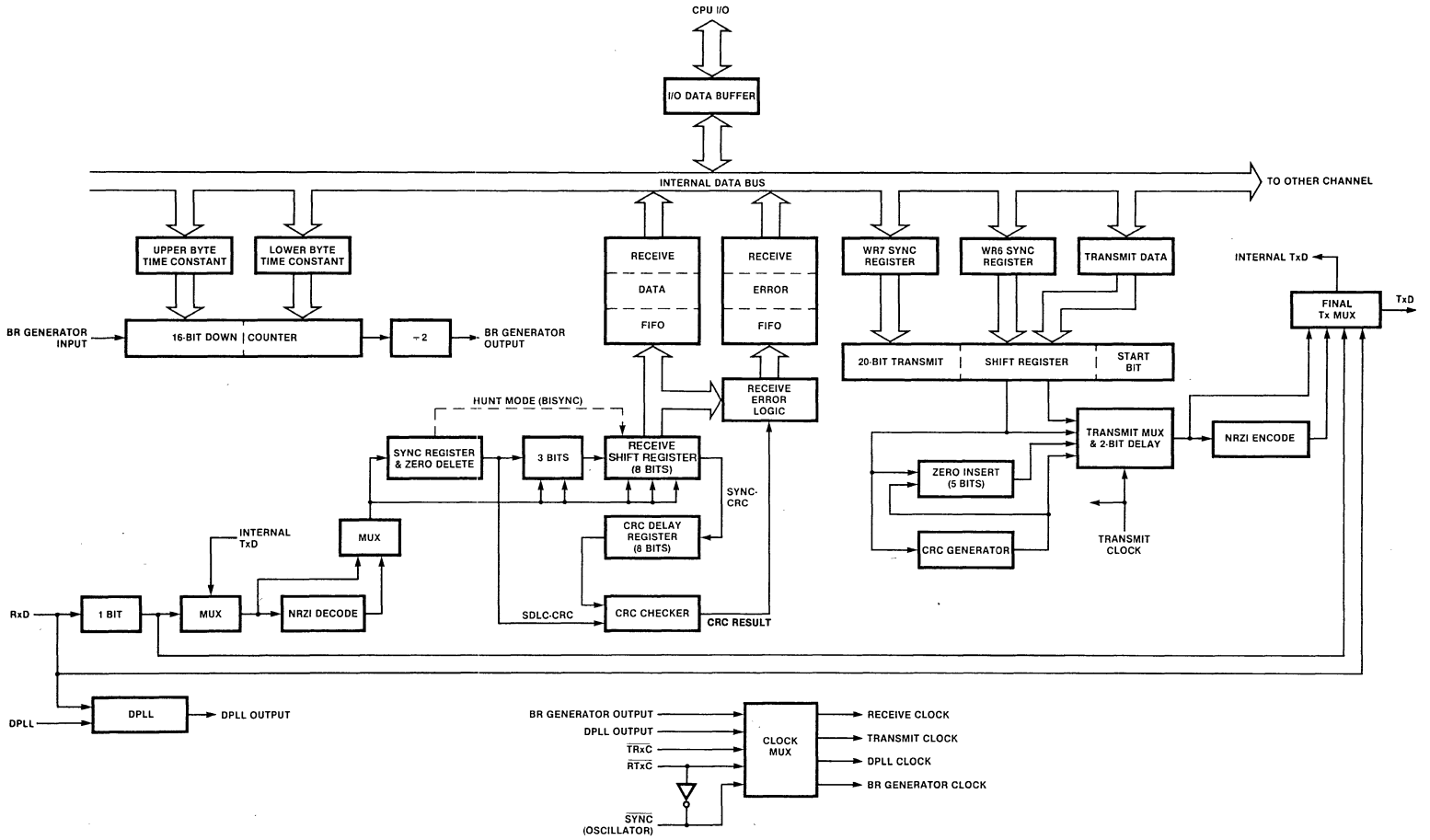


Figure 9. Data Path

## Architecture (Continued)

write only Master Interrupt Control register and three read registers: one containing the vector with status information (Channel B only), one containing the vector without status (Channel A only), and one containing the Interrupt Pending bits (Channel A only).

The registers for each channel are designated as follows:

WRO-WR15 — Write Registers 0 through 15.

RR0-RR3, RR10, RR12, RR13, RR15 — Read Registers 0 through 3, 10, 12, 13, 15.

Table 1 lists the functions assigned to each read or write register. The SCC contains only one WR2 and WR9, but they can be accessed by either channel. All other registers are paired (one for each channel).

**Data Path.** The transmit and receive data path illustrated in Figure 9 is identical for both channels. The receiver has three 8-bit buffer registers in an FIFO arrangement, in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high speed data. Incoming data is routed through one of several paths (data or CRC) depending on the selected mode (the character length in Asynchronous modes also determines the data path).

The transmitter has an 8-bit Transmit Data buffer register loaded from the internal data bus and a 20-bit Transmit Shift register that can be loaded either from the synchronous character registers or from the Transmit Data register. Depending on the operational mode, outgoing data is routed through one of four main paths before it is transmitted from the Transmit Data output (TxD)

## Programming

The SCC contains write registers in each channel that are programmed by the system separately to configure the functional personality of the channels.

### Z8530

In the SCC, register addressing is direct for the data registers only, which are selected by a High on the D/C pin. In all other cases (with the exception of WRO and RR0), programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to WRO and contains three bits that point to the selected register. The second write is the actual control word for the selected register, and if the second operation is read, the selected read register is accessed. All of the registers in the SCC, including the data registers, may be accessed in this fashion. The pointer bits are automatically cleared after the read or write operation so that WRO (or RR0) is addressed again.

### Read Register Functions

RR0	Transmit/Receive buffer status and External status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only) Unmodified interrupt vector (Channel A only)
RR3	Interrupt Pending bits (Channel A only)
RR8	Receive buffer
RR10	Miscellaneous status
RR12	Lower byte of baud rate generator time constant
RR13	Upper byte of baud rate generator time constant
RR15	External/Status interrupt information

### Write Register Functions

WRO	CRC initialize, initialization commands for the various modes, Register Pointers
WR1	Transmit/Receive interrupt and data transfer mode definition
WR2	Interrupt vector (accessed through either channel)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls
WR6	Sync characters or SDLC address field
WR7	Sync character or SDLC flag
WR8	Transmit buffer
WR9	Master interrupt control and reset (accessed through either channel)
WR10	Miscellaneous transmitter/receiver control bits
WR11	Clock mode control
WR12	Lower byte of baud rate generator time constant
WR13	Upper byte of baud rate generator time constant
WR14	Miscellaneous control bits
WR15	External/Status interrupt control

Table 1. Read and Write Register Functions

### Z8030

All SCC registers are directly addressable. How the SCC decodes the address placed on the address/data bus at the beginning of a Read or Write cycle is controlled by a command issued in WROB. In the Shift Right mode the channel select A/B is taken from AD<sub>0</sub> and the state of AD<sub>5</sub> is ignored. In the Shift Left mode the channel select A/B is taken from AD<sub>5</sub> and the state of AD<sub>0</sub> is ignored. AD<sub>7</sub> and AD<sub>6</sub> are always ignored as address bits and the register address itself occupies AD<sub>4</sub>-AD<sub>1</sub>.

### Z8530/Z8030

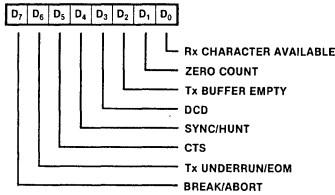
The system program first issues a series of commands to initialize the basic mode of operation. This is followed by other commands to qualify conditions within the selected mode. For example, the Asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first. Then the interrupt mode would be set, and finally, receiver or transmitter enable.

**Programming Read Registers.** The SCC contains eight read registers (actually nine, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers (RR12 and RR13) may be read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B).

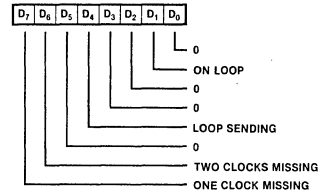
B). RR3 contains the Interrupt Pending (IP) bits (Channel A). Figure 10 shows the formats for each read register.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring; e.g., when the interrupt vector indicates a Special Receive Condition interrupt, all the appropriate error bits can be read from a single register (RR1).

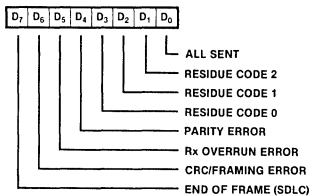
**Read Register 0**



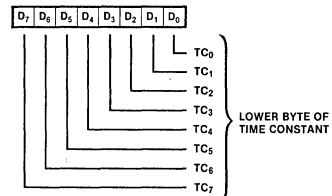
**Read Register 10**



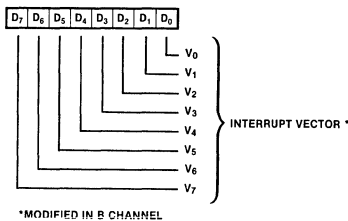
**Read Register 1**



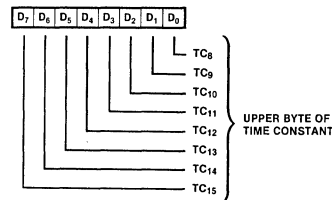
**Read Register 12**



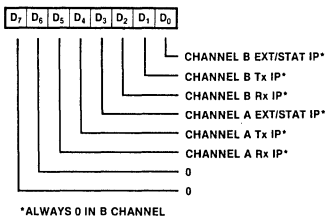
**Read Register 2**



**Read Register 13**



**Read Register 3**



**Read Register 15**

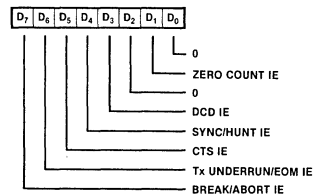


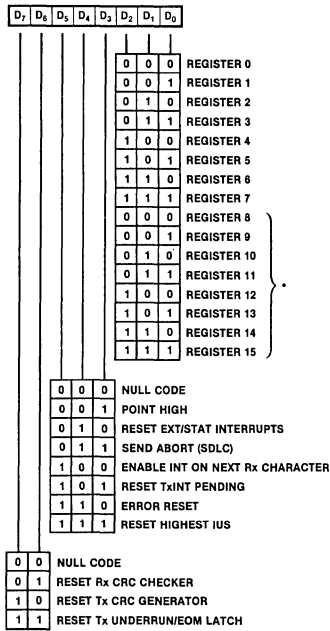
Figure 10. Read Register Bit Functions

**Write Registers.** The SCC contains 13 write registers (14 counting WR8, the transmit buffer) in each channel. These write registers are programmed separately to configure the functional "personality" of the channels. In addition, there are two registers (WR2 and

WR9) shared by the two channels that may be accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits. Figure 11 shows the format of each write register.

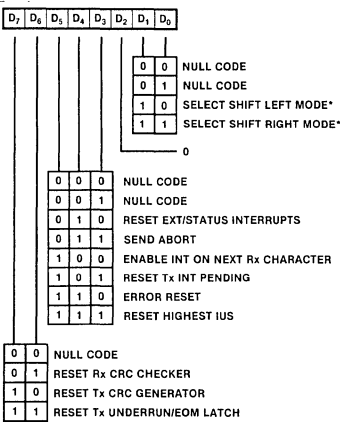
# Programming (Continued)

## Write Register 0 (Z8530)



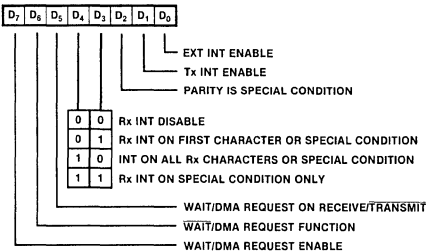
\*WITH POINT HIGH COMMAND

## Write Register 0 (Z8030)

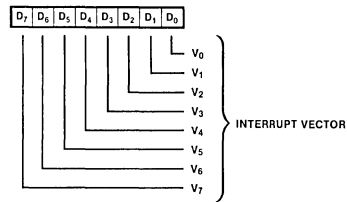


\* B CHANNEL ONLY

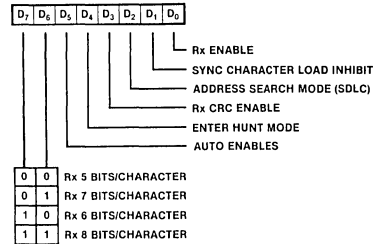
## Write Register 1



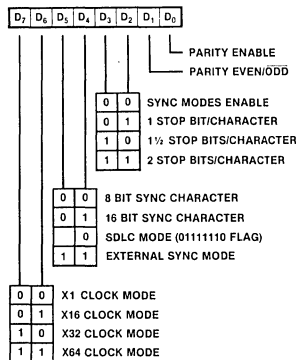
## Write Register 2



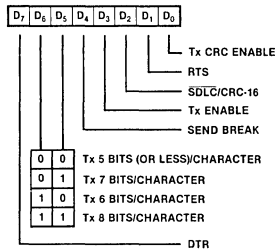
## Write Register 3



## Write Register 4



## Write Register 5



## Write Register 6

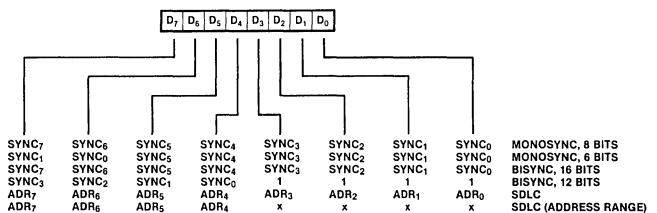
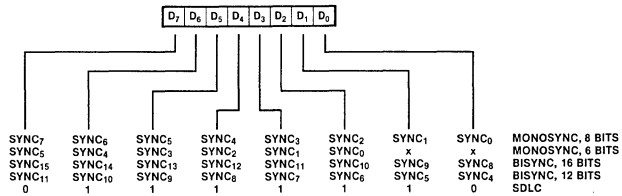
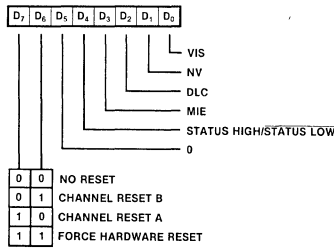


Figure 11. Write Register Bit Functions

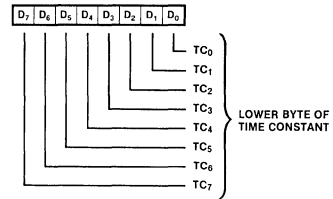
Write Register 7



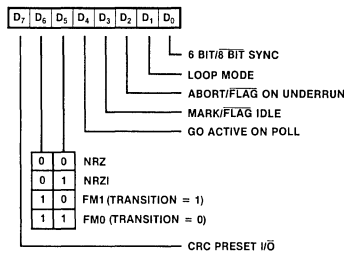
Write Register 9



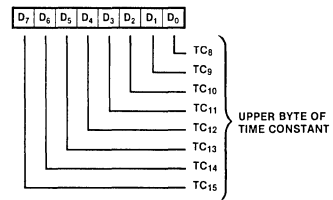
Write Register 12



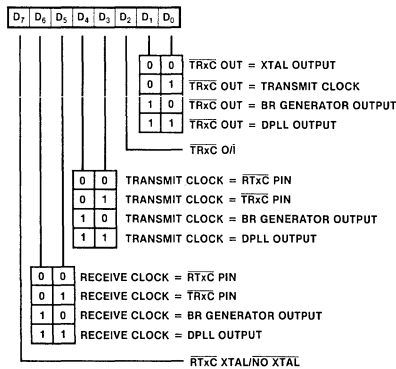
Write Register 10



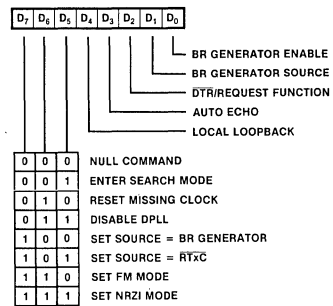
Write Register 13



Write Register 11



Write Register 14



Write Register 15

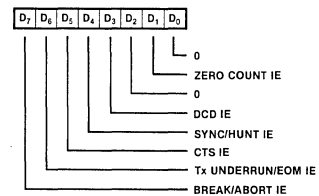


Figure 11. Write Register Bit Functions (Continued)

## 28530 Timing

The SCC generates internal control signals from  $\overline{WR}$  and  $\overline{RD}$  that are related to PCLK. Since PCLK has no phase relationship with  $\overline{WR}$  and  $\overline{RD}$ , the circuitry generating these internal control signals must provide time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK. The recovery time applies only between bus transactions involving the SCC. The recovery time required for proper operation is specified from the falling edge of  $\overline{WR}$  or  $\overline{RD}$  in the first transaction involving the SCC to the falling

edge of  $\overline{WR}$  or  $\overline{RD}$  in the second transaction involving the SCC. This time must be at least 4 PCLK regardless of which register or channel is being accessed.

**Read Cycle Timing.** Figure 12 illustrates Read cycle timing. Addresses on  $A/\overline{B}$  and  $D/\overline{C}$  and the status on  $\overline{INTACK}$  must remain stable throughout the cycle. If  $\overline{CE}$  falls after  $\overline{RD}$  falls or if it rises before  $\overline{RD}$  rises, the effective  $\overline{RD}$  is shortened.

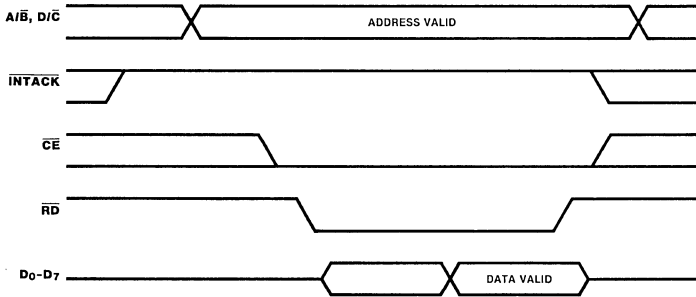


Figure 12. Read Cycle Timing

**Write Cycle Timing.** Figure 13 illustrates Write cycle timing. Addresses on  $A/\overline{B}$  and  $D/\overline{C}$  and the status on  $\overline{INTACK}$  must remain stable throughout the cycle. If  $\overline{CE}$  falls after  $\overline{WR}$  falls

or if it rises before  $\overline{WR}$  rises, the effective  $\overline{WR}$  is shortened. Data must be valid before the falling edge of  $\overline{WR}$ .

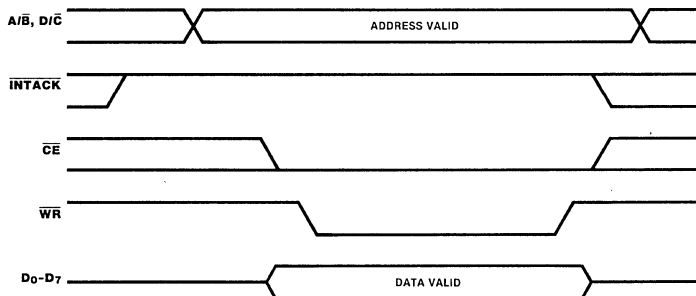


Figure 13. Write Cycle Timing

**Interrupt Acknowledge Cycle Timing.** Figure 14 illustrates Interrupt Acknowledge cycle timing. Between the time  $\overline{INTACK}$  goes Low and the falling edge of  $\overline{RD}$ , the internal and external IEI/IEO daisy chains settle. If there is an interrupt pending in the SCC and IEI is High

when  $\overline{RD}$  falls, the Acknowledge cycle is intended for the SCC. In this case, the SCC may be programmed to respond to  $\overline{RD}$  Low by placing its interrupt vector on  $D_0-D_7$  and it then sets the appropriate Interrupt-UnderService latch internally.

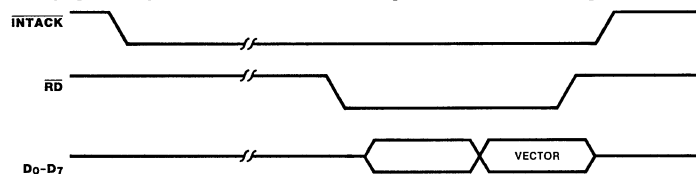


Figure 14. Interrupt Acknowledge Cycle Timing

**Z8030 Timing** The SCC generates internal control signals from  $\overline{AS}$  and  $\overline{DS}$  that are related to PCLK. Since PCLK has no phase relationship with  $\overline{AS}$  and  $\overline{DS}$ , the circuitry generating these internal control signals must provide time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK. The recovery time applies only between bus transactions involving the SCC. The recovery time required for proper operation is specified from the falling edge of  $\overline{DS}$  in the first transaction

involving the SCC to the falling edge of  $\overline{DS}$  in the second transaction involving the SCC.

**Read Cycle Timing.** Figure 15 illustrates Read cycle timing. The address on  $AD_0-AD_7$  and the state of  $\overline{CS}_0$  and  $\overline{INTACK}$  are latched by the rising edge of  $\overline{AS}$ .  $R/\overline{W}$  must be High to indicate a Read cycle.  $\overline{CS}_1$  must also be High for the Read cycle to occur. The data bus drivers in the SCC are then enabled while  $\overline{DS}$  is Low.

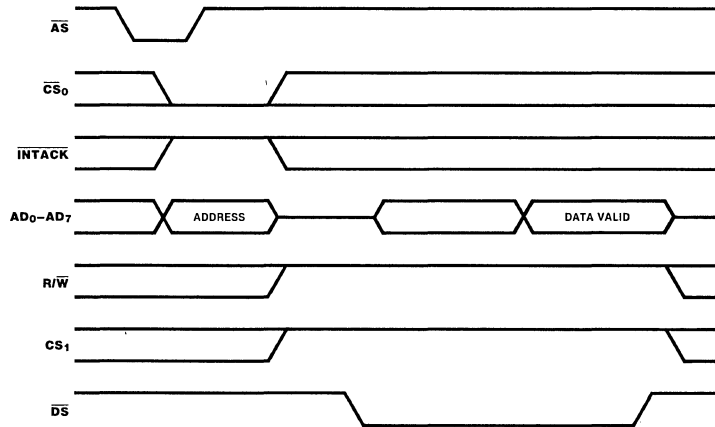


Figure 15. Read Cycle Timing

**Write Cycle Timing.** Figure 16 illustrates Write cycle timing. The address on  $AD_0-AD_7$  and the state of  $\overline{CS}_0$  and  $\overline{INTACK}$  are latched by the rising edge of  $\overline{AS}$ .  $R/\overline{W}$  must be Low to

indicate a Write cycle.  $\overline{CS}_1$  must be High for the Write cycle to occur.  $\overline{DS}$  Low strobes the data into the SCC.

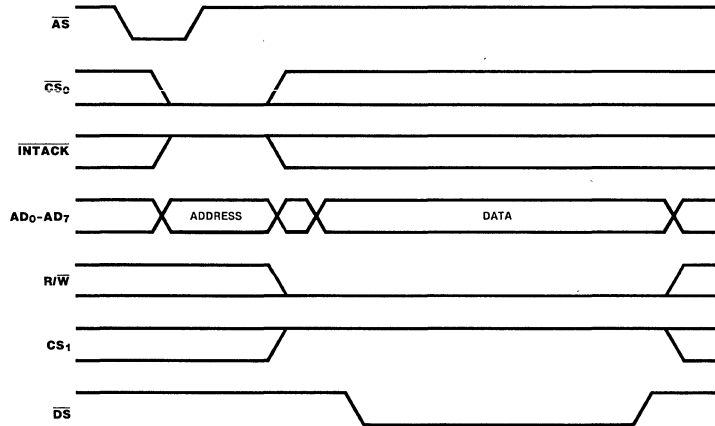


Figure 16. Write Cycle Timing

**Interrupt Acknowledge Cycle Timing.** Figure 17 illustrates Interrupt Acknowledge cycle timing. The address on  $AD_0$ - $AD_7$  and the state of  $\overline{CS}_0$  and  $\overline{INTACK}$  are latched by the rising edge of  $\overline{AS}$ . However, if  $\overline{INTACK}$  is Low, the address and  $\overline{CS}_0$  are ignored. The state of the  $\overline{R/W}$  and  $CS_1$  are also ignored for the duration of the Interrupt Acknowledge cycle. Between the rising edge of  $\overline{AS}$  and the falling edge of

$\overline{DS}$ , the internal and external IEI/IEO daisy chains settle. If there is an interrupt pending in the SCC and IEI is High when  $\overline{DS}$  falls, the Acknowledge cycle was intended for the SCC. In this case, the SCC may be programmed to respond to RD Low by placing its interrupt vector on  $D_0$ - $D_7$  and it then internally sets the appropriate Interrupt-Under-Service latch.

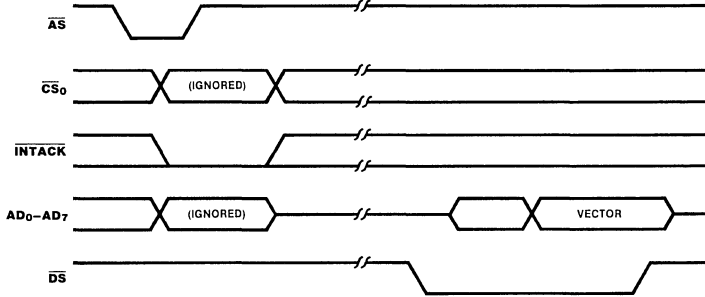


Figure 17. Interrupt Acknowledge Cycle Timing

**Absolute Maximum Ratings**

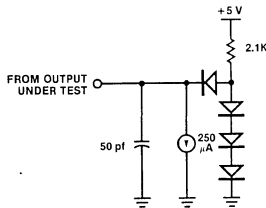
Voltages on all pins with respect to GND . . . . . -0.3V to +7.0V  
 Operating Ambient Temperature . . . . . See Ordering Information  
 Storage Temperature . . . . . -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

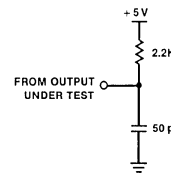
**Standard Test Conditions**

The DC characteristics and capacitance section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin.  
 Standard conditions are as follows:

- $+4.75\text{ V} \leq V_{CC} \leq +5.25\text{ V}$
  - $GND = 0\text{ V}$
  - $T_A$  as specified in Ordering Information
- All ac parameters assume a load capacitance of 50 pF max.



Standard Test Load



Open-Drain Test Load



**DC CHARACTERISTICS**

Z8530/8030

<b>DC Characteristics</b>	<b>Symbol</b>	<b>Parameter</b>	<b>Min</b>	<b>Max</b>	<b>Unit</b>	<b>Condition</b>
	V <sub>IH</sub>	Input High Voltage	2.0	V <sub>CC</sub> + 0.3	V	
	V <sub>IL</sub>	Input-Low Voltage	-0.3	0.8	V	
	V <sub>OH</sub>	Output High Voltage	2.4		V	I <sub>OH</sub> = -250 μA
	V <sub>OL</sub>	Output Low Voltage		0.4	V	I <sub>OL</sub> = +2.0 mA
	I <sub>IL</sub>	Input Leakage		±10.0	μA	0.4 ≤ V <sub>IN</sub> ≤ +2.4V
	I <sub>OL</sub>	Output Leakage		±10.0	μA	0.4 ≤ V <sub>OUT</sub> ≤ +2.4V
	I <sub>CC</sub>	V <sub>CC</sub> Supply Current		250	mA	

V<sub>CC</sub> = 5 V ± 5% unless otherwise specified, over specified temperature range.

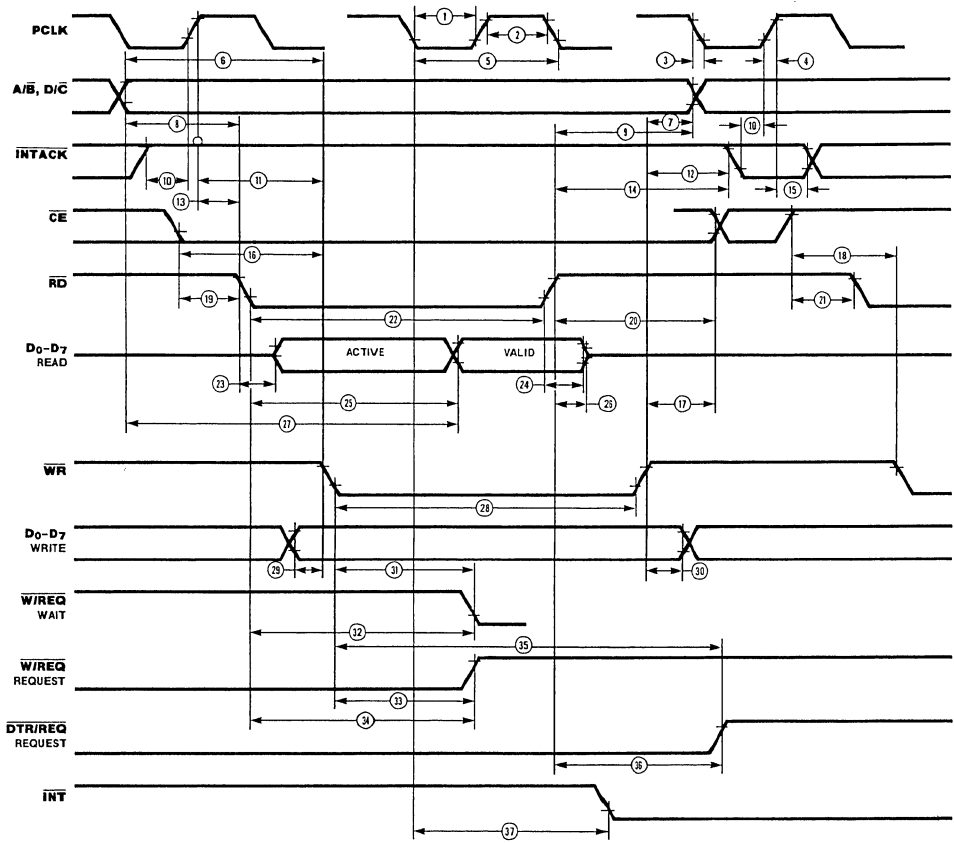
<b>Capacitance</b>	<b>Symbol</b>	<b>Parameter</b>	<b>Min</b>	<b>Max</b>	<b>Unit</b>	<b>Test Condition</b>
	C <sub>IN</sub>	Input Capacitance		10	pF	Unmeasured Pins Returned to Ground
	C <sub>OUT</sub>	Output Capacitance		15	pF	
	C <sub>I/O</sub>	Bidirectional Capacitance		20	pF	

f = 1 MHz, over specified temperature range.  
Unmeasured pins returned to ground.

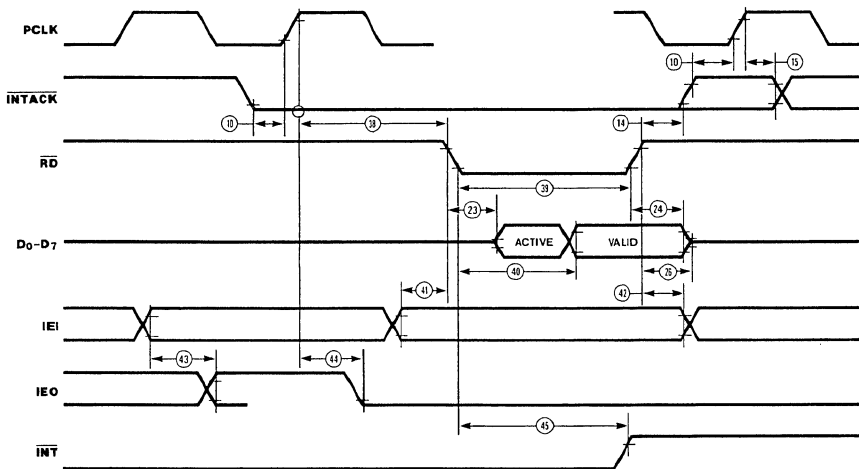
<b>Miscellaneous</b>	Gate Count	6000
----------------------	------------	------

# Z8530 AC CHARACTERISTICS

## Read and Write Timing Z8530



## Interrupt Acknowledge Timing Z8530



## Z8530 AC CHARACTERISTICS

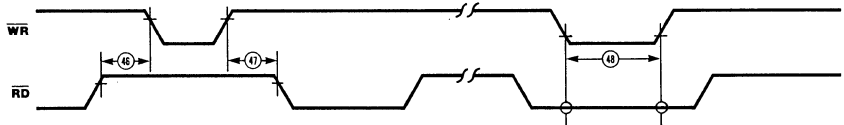
Number	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes [6]
			Min	Max	Min	Max	Min	Max	
1	TwPCI	PCLK Low Width	105	2000	70	1000	50	1000	
2	TwPCh	PCLK High Width	105	2000	70	1000	50	1000	
3	TfPC	PCLK Fall Time		20		10		10	
4	TrPC	PCLK Rise Time		20		10		10	
5	TcPC	PCLK Cycle Time	250	4000	165	2000	125	2000	
6	TsA(WR)	Address to $\overline{WR}$ ↓ Setup Time	80		80		70		
7	ThA(WR)	Address to $\overline{WR}$ ↑ Hold Time	0		0		0		
8	TsA(RD)	Address to $\overline{RD}$ ↓ Setup Time	80		80		70		
9	ThA(RD)	Address to $\overline{RD}$ ↑ Hold Time	0		0		0		
10	TsIA(PC)	$\overline{INTACK}$ to PCLK ↑ Setup Time	10		10		10		
11	TsIAi(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↓ Setup Time	200		160		145		1
12	ThIA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↑ Hold Time	0		0		0		
13	TsIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ Setup Time	200		160		145		1
14	ThIA(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↑ Hold Time	0		0		0		
15	ThIA(PC)	$\overline{INTACK}$ to PCLK ↑ Hold Time	100		100		85		
16	TsCEI(WR)	$\overline{CE}$ Low to $\overline{WR}$ ↓ Setup Time	0		0		0		
17	ThCE(WR)	$\overline{CE}$ to $\overline{WR}$ ↑ Hold Time	0		0		0		
18	TsCEh(WR)	$\overline{CE}$ High to $\overline{WR}$ ↓ Setup Time	100		70		60		
19	TsCEI(RD)	$\overline{CE}$ Low to $\overline{RD}$ ↓ Setup Time	0		0		0		1
20	ThCE(RD)	$\overline{CE}$ to $\overline{RD}$ ↑ Hold Time	0		0		0		1
21	TsCEh(RD)	$\overline{CE}$ High to $\overline{RD}$ ↓ Setup Time	100		70		60		1
22	TwRDI	$\overline{RD}$ Low Width	240		200		150		1
23	TdRD(DRA)	$\overline{RD}$ ↓ to Read Data Active Delay	0		0		0		
24	TdRDr(DR)	$\overline{RD}$ ↑ to Read Data Not Valid Delay	0		0		0		
25	TdRDf(DR)	$\overline{RD}$ ↓ to Read Data Valid Delay		250		180		140	
26	TdRD(DRz)	$\overline{RD}$ ↑ to Read Data Float Delay		70		45		40	2

### NOTES:

- Parameter does not apply to Interrupt Acknowledge transactions.
- Float delay is defined as the time required for a  $\pm 0.5V$  change at the output with a maximum dc load and minimum ac load.

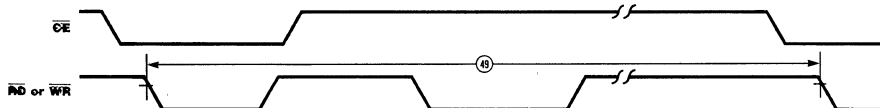
### Reset Timing

Z8530



### Cycle Timing

Z8530



## Z8530 AC CHARACTERISTICS (Continued)

Number	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes [6]
			Min	Max	Min	Max	Min	Max	
27	TdA(DR)	Address Required Valid to Read Data Valid Delay		300		280		220	
28	TwWRI	$\overline{WR}$ Low Width	240		200		150		
29	TsDW(WR)	Write Data to $\overline{WR}$ ↓ Setup Time	10		10		10		
30	ThDW(WR)	Write Data to $\overline{WR}$ ↑ Hold Time	0		0		0		
31	TdWR(W)	$\overline{WR}$ ↓ to Wait Valid Delay		240		200		170	4
32	TdRD(W)	$\overline{RD}$ ↓ Wait Valid Delay		240		200		170	4
33	TdWRI(REQ)	$\overline{WR}$ ↓ to $\overline{W}/\overline{REQ}$ Not Valid Delay		240		200		170	
34	TdRD(REQ)	$\overline{RD}$ ↓ to $\overline{W}/\overline{REQ}$ Not Valid Delay		240		200		170	
35	TdWRr(REQ)	$\overline{WR}$ ↓ $\overline{DTR}/\overline{REQ}$ Not Valid Delay		4TcPC		4TcPC		4TcPC	
36	TdRDr(REQ)	$\overline{RD}$ ↑ to $\overline{DTR}/\overline{REQ}$ Not Valid Delay		4TcPC		4TcPC		4TcPC	
37	TdPC(INT)	PCLK ↓ to $\overline{INT}$ Valid Delay		500		500		500	4
38	TdIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ (Acknowledge) Delay	250		200		150		5
39	TwRDA	$\overline{RD}$ (Acknowledge) Width	250		200		150		
40	TdRDA(DR)	$\overline{RD}$ ↓ (Acknowledge) to Read Data Valid Delay		250		180		140	
41	TsIEI(RDA)	IEI to $\overline{RD}$ ↓ (Acknowledge) Setup Time	120		100		95		
42	ThIEI(RDA)	IEI to $\overline{RD}$ ↑ (Acknowledge) Hold Time	0		0		0		
43	TdIEI(IEO)	IEI to IEO Delay Time		120		100		95	
44	TdPC(IEO)	PCLK ↑ to IEO Delay		250		250		200	
45	TdRDA(INT)	$\overline{RD}$ ↓ to $\overline{INT}$ Inactive Delay		500		500		450	4
46	TdRD(WRQ)	$\overline{RD}$ ↑ to $\overline{WR}$ ↓ Delay for No Reset	30		15		15		
47	TdWRQ(RD)	$\overline{WR}$ ↑ to $\overline{RD}$ ↓ Delay for No Reset	30		30		20		
48	TwRES	$\overline{WR}$ and $\overline{RD}$ Coincident Low for Reset	250		200		150		
49	Trc	Valid Access Recovery Time		4TcPC		4TcPC		4TcPC	3

### NOTES:

3. Parameter applies only between transactions involving the SCC.

4. Open-drain output, measured with open-drain test load.

5. Parameter is system dependent. For any SCC in the daisy chain, TdIAi(RD) must be greater than the sum of TdPC(IEO) for the highest priority device in the daisy chain, TsIEI(RDA) for the SCC, and TdIEI(IEO) for each device separating them in the daisy chain.

6. Units in nanoseconds (ns), otherwise noted.

# AC CHARACTERISTICS

## Z8530 General Timing Diagram

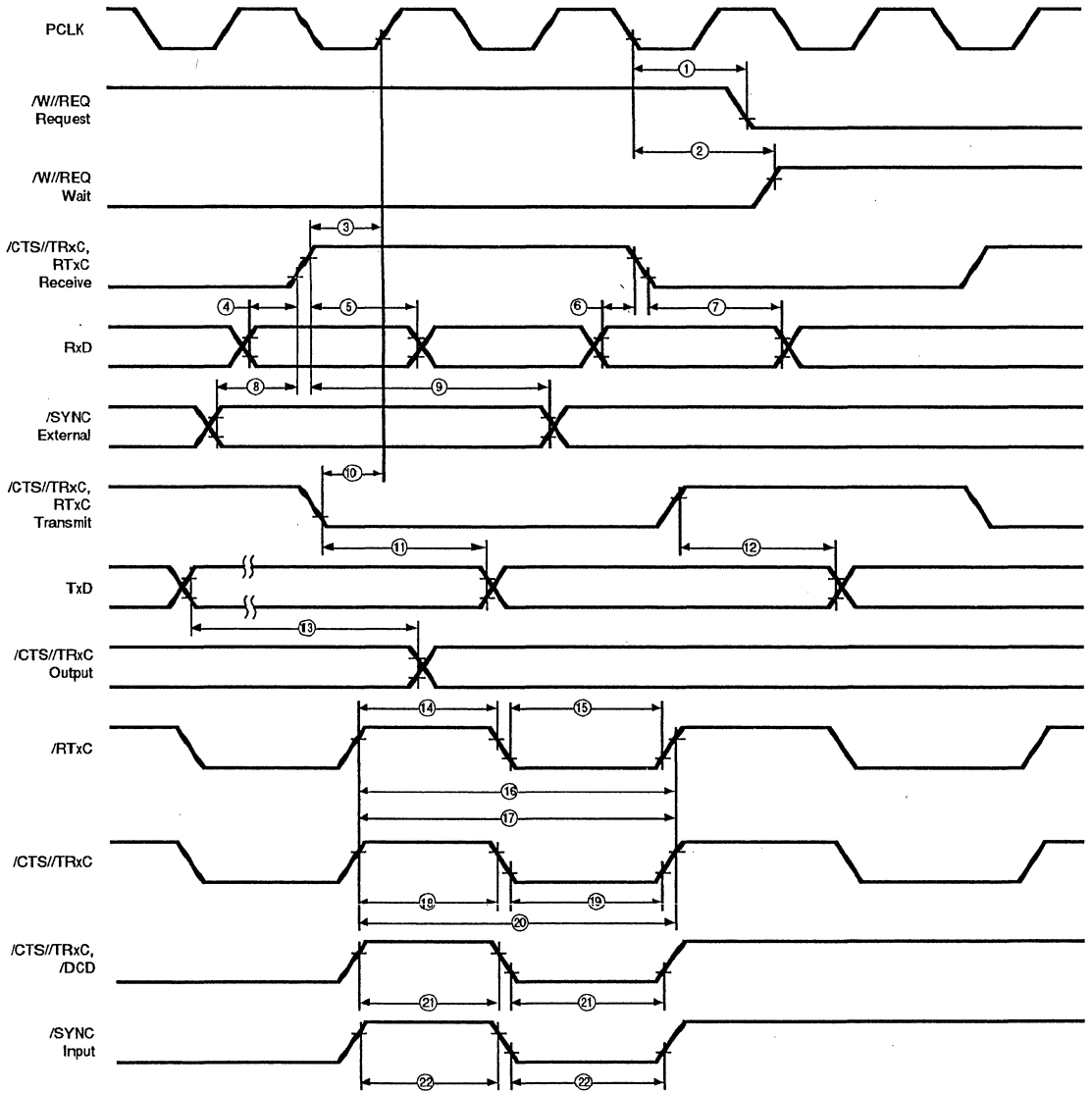


Figure 34. Z8530 General Timing Diagram

## AC CHARACTERISTICS

### Z8530 General Timing Table

No	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes [8]
			Min	Max	Min	Max	Min	Max	
1	TdPC(REQ)	/PCLK Low to /W//REQ Valid		250		250		250	
2	TsPC(W)	/PCLK Low to Wait Inactive		350		350		350	
3	TsRXC(PC)	/RxC High to /PCLK High Setup Time (PCLK ÷ 4 case only)	80	TwPCL	70	TwPCL	60	TwPCL	[1,4]
4	TsRXD(RXCr)	RxD to /RxC High Setup Time (X1 Mode)	0		0		0		[1]
5	ThRXD(RxCr)	RxD to /RxC High Hold Time (X1 Mode)	150		150		150		[1]
6	TsRXD(RXCf)	RxD to /RxC Low Setup Time (X1 Mode)	0		0		0		[1,5]
7	ThRXD(RXCf)	RxD to /RxC Low Hold Time (X1 Mode)	150		150		150		[1,5]
8	TsSY(RXC)	SYNC to /RxC High Setup Time	-200		-200		-200		[1]
9	ThSY(RXC)	SYNC to /RxC High Hold Time	3TcPc+400		3TcPc+400		3TcPc+400		[1]
10	TsTXC(PC)	/TxC Low to /PCLK High Setup Time	0		0		0		[2,4]
11	TdTXCf(TXD)	/TxC Low to TxD Delay (X1 Mode)		300		230		200	[2]
12	TdTxCr(TXD)	/TxC High to TxD Delay (X1 Mode)		300		230		200	[2,5]
13	TdTXD(TRX)	TxD to TRxC Delay (Send Clock Echo)		200		200		200	
14	TwRTXh	/RTxC High Width	180		180		150		[6]
15	TwRTXI	/RTxC Low Width	180		180		150		[6]
16	TcRTX	/RTxC Cycle Time (RxD, TxD)	1000		640		500		[6,7]
17	TcRTXX	Crystal Osc. Period	250	1000	165	1000	125	1000	[3]
18	TwTRXh	/TRxC High Width	180		180		150		[6]
19	TwTRXI	/TRxC Low Width	180		180		150		[6]
20	TcTRX	/TRxC Cycle Time	1000		640		500		[6,7]
21	TwEXT	/DCD or /CTS Pulse Width	200		200		200		
22	TwSY	/SYNC Pulse Width	200		200		200		

#### Notes:

[1] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[2] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[3] Both /RTxC and /SYNC have 30 pf capacitors to ground connected to them.

[4] Parameter applies only if the data rate is one-fourth the PCLK rate. In all other cases, no phase relationship between /RxC and PCLK or /TxC and PCLK is required.

[5] Parameter applies only to FM encoding/decoding.

[6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator timing requirements are identical to case PCLK requirements.

[7] The maximum receive or transmit data rate is 1/4 PCLK.

[8] Units in nanoseconds (ns), otherwise noted.

**AC CHARACTERISTICS**  
Z8530 System Timing Diagram

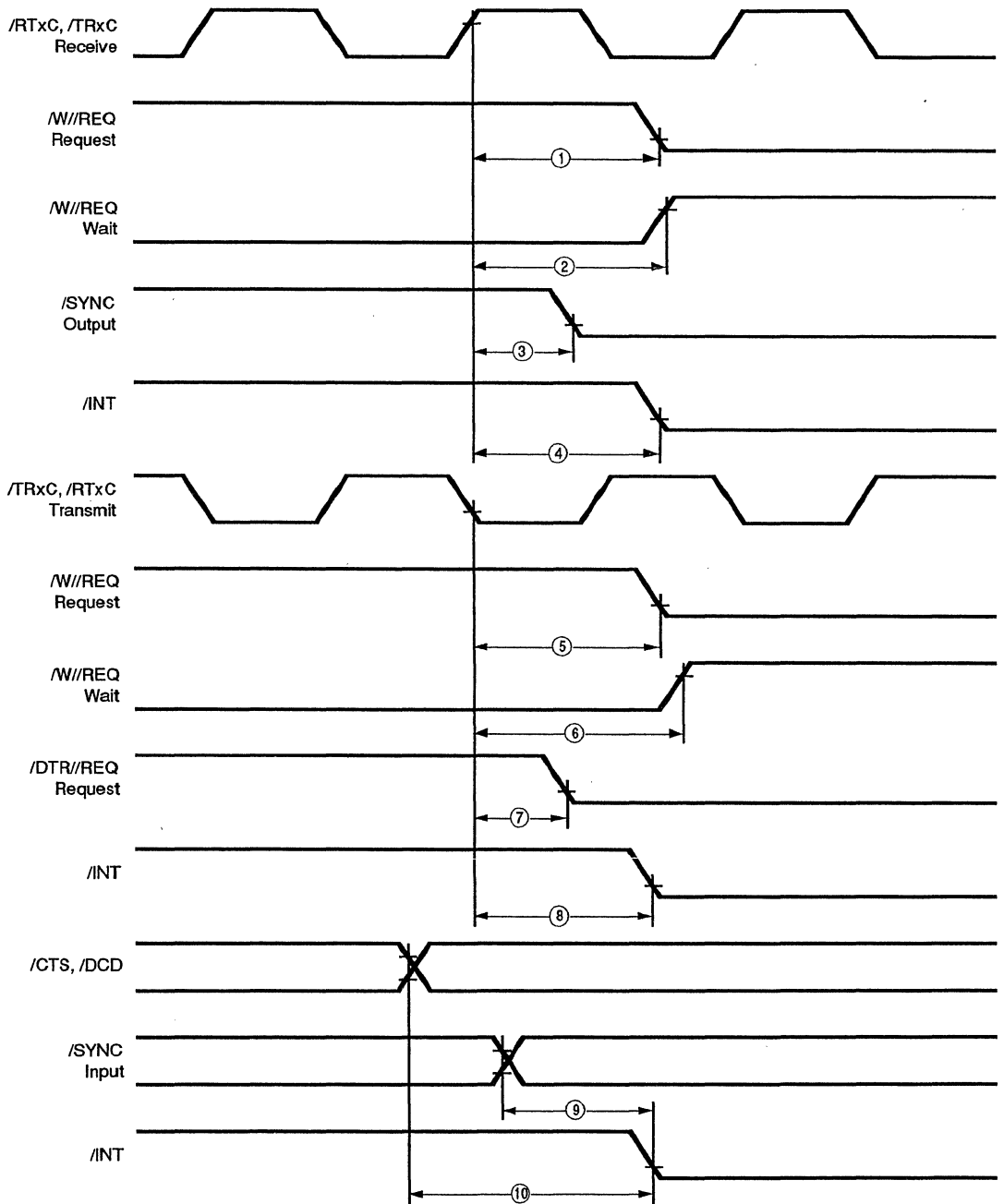


Figure 35. Z8530 System Timing Diagram

## AC CHARACTERISTICS

### Z8530 System Timing Table

No	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes <sup>[5]</sup>
			Min	Max	Min	Max	Min	Max	
1	TdRXC(REQ)	/RxC High to /W//REQ Valid	8	12	8	12	8	12	[2]
2	TdRXC(W)	/RxC High to Wait Inactive	8	14	8	14	8	14	[1,2]
3	TdRdXC(SY)	/RxC High to /SYNC Valid	4	7	4	7	4	7	[2]
4	TsRXC(INT)	/RxC High to INT Valid	10	16	10	16	10	16	[1,2]
			2	3	2	3	2	3	[4]
5	TdTXC(REQ)	/TxC Low to /W//REQ Valid	5	8	5	8	5	8	[3]
6	TdTXC(W)	/TxC Low to Wait Inactive	5	11	5	11	5	11	[1,3]
7	TdTXC(DRQ)	/TxC Low to /DTR//REQ Valid	4	7	4	7	4	7	[3]
8	TdTXC(INT)	/TxC Low to /INT Valid	6	10	6	10	6	10	[1,3]
			2	3	2	3	2	3	[4]
9	TdSY(INT)	/SYNC to INT Valid	2	6	2	6	2	6	[1]
10	TdEXT(INT)	/DCD or /CTS to /INT Valid	2	6	2	6	2	6	[1]

#### Notes:

[1] Open drain-output, measured with open-drain test load.

[2] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[3] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[4] Units equal to /AS.

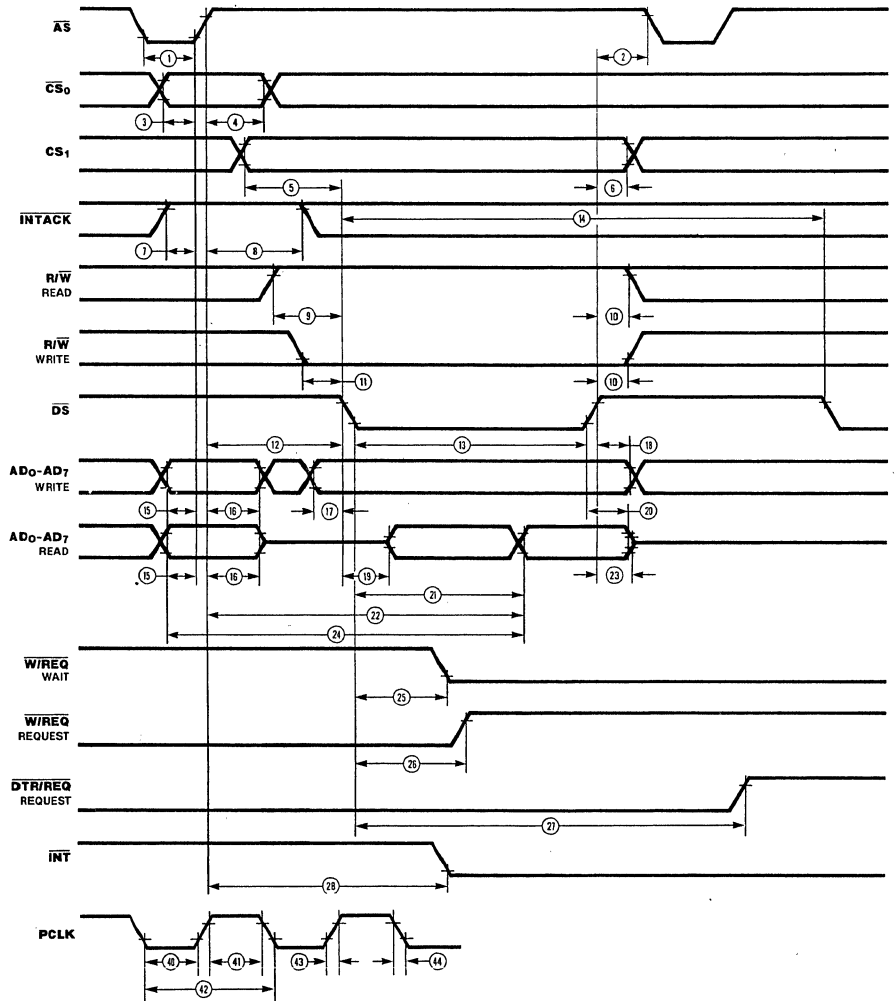
[5] Units equal to TcPc.



# Z8030 AC CHARACTERISTICS

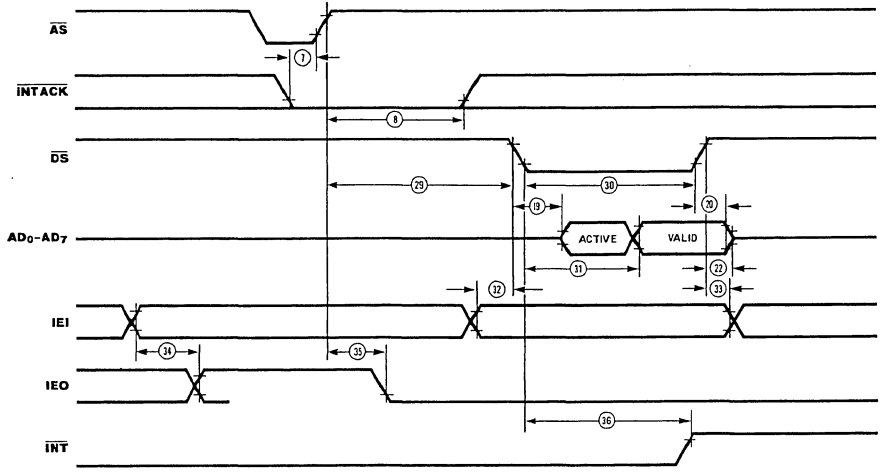
## Read and Write Timing

Z8030

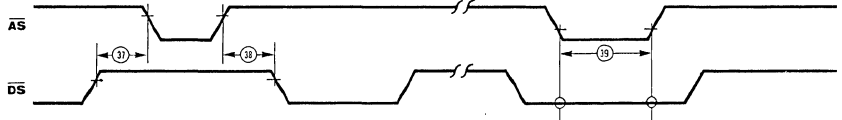


## Z8030 AC CHARACTERISTICS

### Interrupt Acknowledge Timing Z8030



### Reset Timing Z8030



## Z8030 AC CHARACTERISTICS

Number	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes <sup>[3]</sup>
			Min	Max	Min	Max	Min	Max	
1	TwAS	$\overline{AS}$ Low Width	70		50		35		
2	TdDS(AS)	$\overline{DS} \uparrow$ to $\overline{AS} \downarrow$ Delay	50		25		15		
3	TsCS0(AS)	$\overline{CS}_0$ to $\overline{AS} \uparrow$ Setup Time	0		0		0		1
4	ThCS0(AS)	$\overline{CS}_0$ to $\overline{AS} \uparrow$ Hold Time	60		40		30		1
5	TsCS1(DS)	$CS_1$ to $\overline{DS} \downarrow$ Setup Time	100		80		65		1
6	ThCS1(DS)	$CS_1$ to $\overline{DS} \uparrow$ Hold Time	55		40		30		1
7	TsIA(AS)	$\overline{INTACK}$ to $\overline{AS} \uparrow$ Setup Time	10		10		10		
8	ThIA(AS)	$\overline{INTACK}$ to $\overline{AS} \uparrow$ Hold Time	250		200		150		
9	TsRWR(DS)	$R/\overline{W}$ (Read) to $\overline{DS} \downarrow$ Setup Time	100		80		65		
10	ThRW(DS)	$R/\overline{W}$ to $\overline{DS} \uparrow$ Hold Time	55		40		35		
11	TsRWW(DS)	$R/\overline{W}$ (Write) to $\overline{DS} \downarrow$ Setup Time	0		0		0		
12	TdAS(DS)	$\overline{AS} \uparrow$ to $\overline{DS} \downarrow$ Delay	60		40		30		
13	TwDSI	$\overline{DS}$ Low Width	240		200		150		
14	TrC	Valid Access Recovery Time	4TcPC		4TcPC		4TcPC		2
15	TsA(AS)	Address to $\overline{AS} \uparrow$ Setup Time	30		10		10		1
16	ThA(AS)	Address to $\overline{AS} \uparrow$ Hold Time	50		30		25		1
17	TsDW(DS)	Write Data to $\overline{DS} \downarrow$ Setup Time	30		20		15		
18	ThDW(DS)	Write Data to $\overline{DS} \uparrow$ Hold Time	30		20		20		
19	TdDS(DA)	$\overline{DS} \downarrow$ to Data Active Delay	0		0		0		
20	TdDSr(DR)	$\overline{DS} \uparrow$ to Read Data Not Valid Delay	0		0		0		
21	TdDSf(DR)	$\overline{DS} \downarrow$ to Read Data Valid Delay		250		180		140	
22	TdAS(DR)	$\overline{AS} \uparrow$ to Read Data Valid Delay		520		300		250	

### NOTES:

1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Parameter applies only between transactions involving the SCC.
3. Units in nanoseconds (ns), otherwise noted.

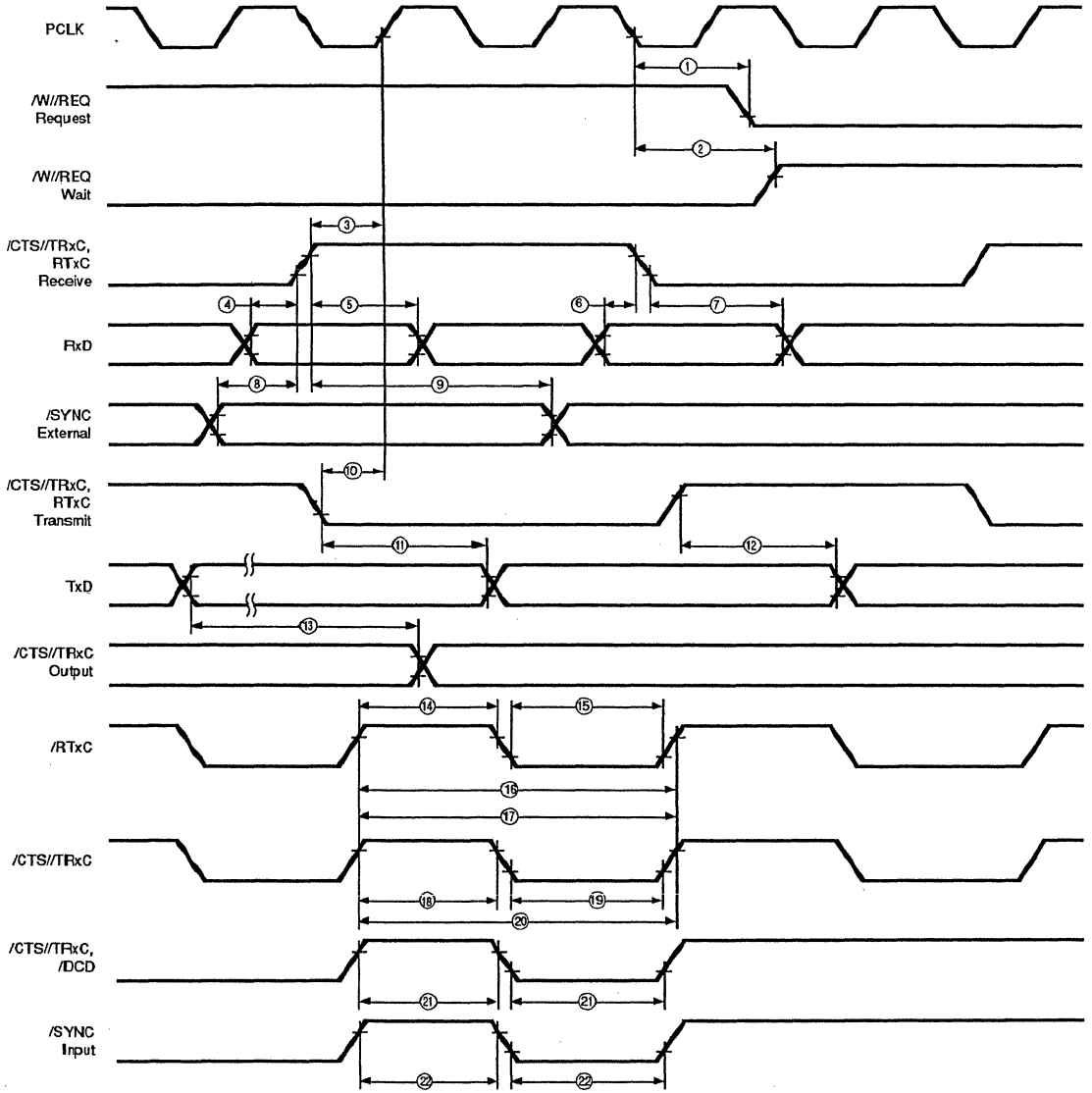
## Z8030 AC CHARACTERISTICS (Continued)

Number	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes [8]
			Min	Max	Min	Max	Min	Max	
23	TdDS(DRz)	$\overline{DS}$ ↑ to Read Data Float Delay		70		45		40	3
24	TdA(DR)	Address Required Valid to Read Data Valid Delay		570		310		260	
25	TdDS(W)	$\overline{DS}$ ↓ to Wait Valid Delay		240		200		170	4
26	TdDS(REQ)	$\overline{DS}$ ↓ to $\overline{WR}/\overline{REQ}$ Not Valid Delay		240		200		170	
27	TdDSr(REQ)	$\overline{DS}$ ↓ to $\overline{DFR}/\overline{REQ}$ Not Valid Delay		4TcPC		4TcPC		4TcPC	
28	TdAS(INT)	$\overline{AS}$ ↑ to $\overline{INT}$ Valid Delay		500		500		500	4
29	TdAS(DSA)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ (Acknowledge) Delay	250		250		250		5
30	TwDSA	$\overline{DS}$ (Acknowledge) Low Width	390		200		150		
31	TdDSA(DR)	$\overline{DS}$ ↓ (Acknowledge) to Read Data Valid Delay		250		180		140	
32	TsIEI(DSA)	IEI to $\overline{DS}$ ↓ (Acknowledge) Setup Time	120		100		80		
33	ThIEI(DSA)	IEI to $\overline{DS}$ ↑ (Acknowledge) Hold Time	0		0		0		
34	TdIEI(IEO)	IEI to IEO Delay		120		100		90	
35	TdAS(IEO)	$\overline{AS}$ ↑ to IEO Delay		250		250		200	6
36	TdDSA(INT)	$\overline{DS}$ ↓ (Acknowledge) to $\overline{INT}$ Inactive Delay		500		500		450	4
37	TdDS(ASQ)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay for No Reset	30			15		15	
38	TdASQ(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay for No Reset	30			30		20	
39	TwRES	$\overline{AS}$ and $\overline{DS}$ Coincident Low for Reset	250		200		150		7
40	TwPCI	PCLK Low Width	105	2000	70	1000	50		
41	TwPCh	PCLK High Width	105	2000	70	1000	50		
42	TcPC	PCLK Cycle Time	250	4000	165	2000	125		
43	TrPC	PCLK Rise Time		20		10		10	
44	TfPC	PCLK Fall Time		20		10		10	

### NOTES:

- Float delay is defined as the time required for a  $\pm 0.5V$  change in the output with a maximum dc load and a minimum ac load.
- Open-drain output, measured with open-drain test load.
- Parameter is system dependent. For any Z-SCC in the daisy chain, TdAS(DSA) must be greater than the sum of TdAS(IEO) for the highest priority device in the daisy chain, TsIEI(DSA) for the Z-SCC, and TdIEI(IEO) for each device separating them in the daisy chain.
- Parameter applies only to a Z-SCC pulling INT Low at the beginning of the Interrupt Acknowledge transaction.
- Internal circuitry allows for the reset provided by the Z8 to be recognized as a reset by the Z-SCC.
- All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".
- Units in nanoseconds (ns), otherwise noted.

**AC CHARACTERISTICS**  
**Z8030 General Timing Diagram**



**Figure 36. Z8030 General Timing Diagram**

## AC CHARACTERISTICS

### Z8030 General Timing Diagram

No	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes <sup>[8]</sup>
			Min	Max	Min	Max	Min	Max	
1	TdPC(REQ)	/PCLK Low to /W//REQ Valid		250		250		250	
2	TsPC(W)	/PCLK Low to Wait Inactive		350		350		350	
3	TsRXC(PC)	/RxC High to /PCLK High Setup Time (PCLK + 4 case only)	80	TwPCL	70	TwPCL	60	TwPCL	[1,4]
4	TsRXD(RxCr)	RxD to /RxC High Setup Time (X1 Mode)	0		0		0		[1]
5	ThRXD(RxCr)	RxD to /RxC High Hold Time (X1 Mode)	150		150		150		[1]
6	TsRXD(RxCf)	RxD to /RxC Low Setup Time (X1 Mode)	0		0		0		[1,5]
7	ThRXD(RxCf)	RxD to /RxC Low Hold Time (X1 Mode)	150		150		150		[1,5]
8	TsSY(RXC)	SYNC to /RxC High Setup Time	-200		-200		-200		[1]
9	ThSY(RXC)	/SYNC to /RxC High Hold Time	3TcPc+400		3TcPc+400		3TcPc+400		[1]
10	TsTXC(PC)	/TxC Low to /PCLK High Setup Time	0		0		0		[2,4]
11	TdTXCf(TXD)	/TxC Low to TxD Delay (X1 Mode)		300		230		200	[2]
12	TdTxCr(TXD)	/TxC High to TxD Delay (X1 Mode)		300		230		200	[2,5]
13	TdTXD(TRX)	TxD to /TRxC Delay (Send Clock Echo)		200		200		200	
14	TwRTXh	/RTxC High Width	180		180		150		[6]
15	TwRTXl	/RTxC Low Width	180		180		150		[6]
16	TcRTX	/RTxC Cycle Time (RxD,TxD)	1000		640		500		[6,7]
17	TcRTXX	Crystal Osc. Period	250	1000	165	1000	125	1000	[3]
18	TwTRXh	/TRxC High Width	180		180		150		[6]
19	TwTRXl	/TRxC Low Width	180		180		150		[6]
20	TcTRX	/TRxC Cycle Time	1000		640		500		[6,7]
21	TwEXT	/DCD or /CTS Pulse Width	200		200		200		
22	TwSY	/SYNC Pulse Width	200		200		200		

#### Notes:

[1] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[2] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[3] Both /RTxC and /SYNC have 30 pF capacitors to ground connected to them.

[4] Parameter applies only if the data rate is one-fourth the PCLK rate. In all other cases, no phase relationship between /RxC and PCLK or /TxC and PCLK is required.

[5] Parameter applies only to FM encoding/decoding.

[6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator timing requirements are identical to case PCLK requirements.

[7] The maximum receive or transmit data rate is 1/4 PCLK.

[8] Units in nanoseconds (ns), otherwise noted.

**AC CHARACTERISTICS**  
Z8030 System Timing Diagram

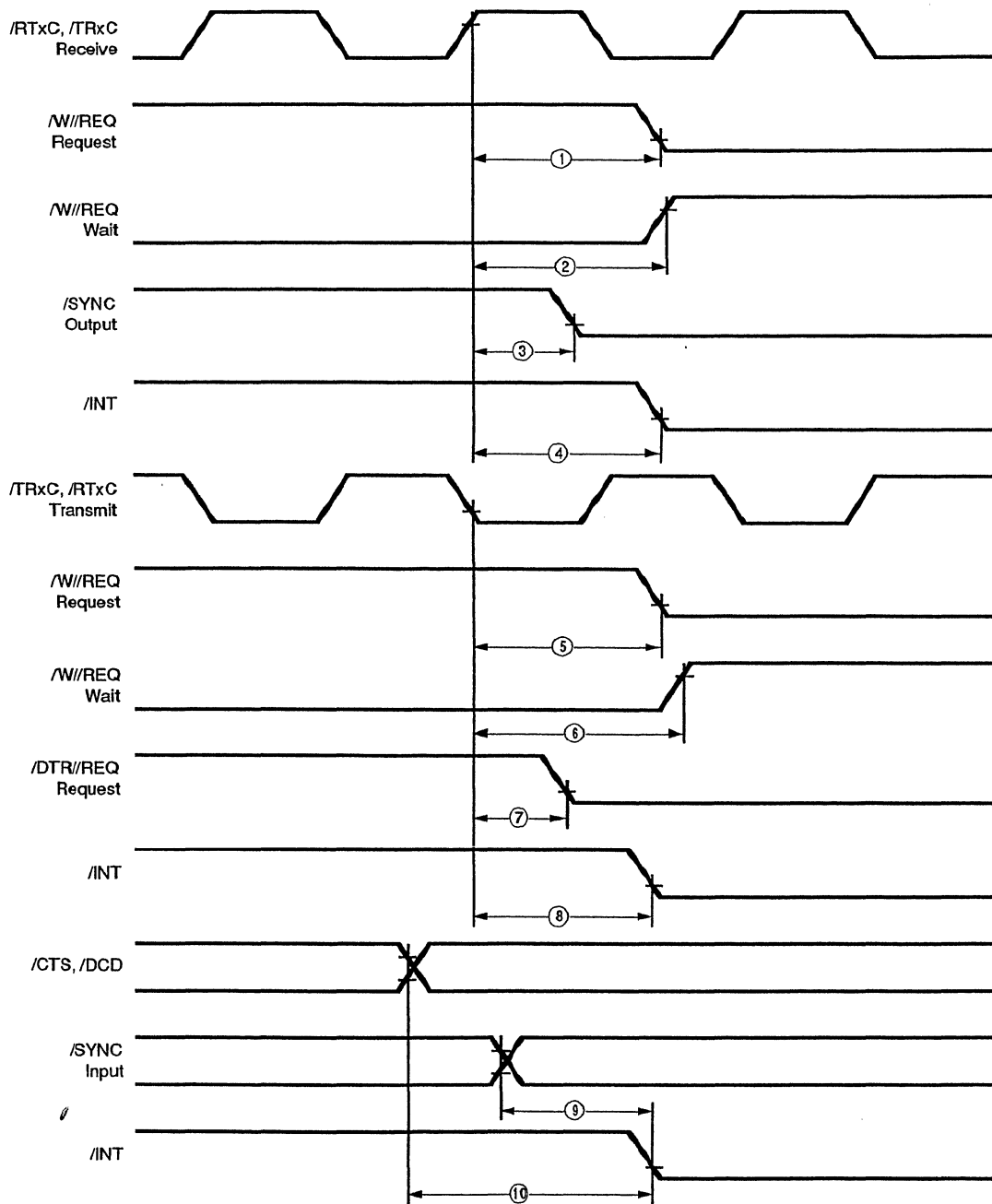


Figure 37. Z8530 System Timing Diagram

## AC CHARACTERISTICS

### Z8030 System Timing Diagram

No	Symbol	Parameter	4 MHz		6 MHz		8 MHz		Notes[5]
			Min	Max	Min	Max	Min	Max	
1	TdRXC(REQ)	/RxC High to /W//REQ Valid	8	12	8	12	8	12	[2]
2	TdRXC(W)	/RxC High to Wait Inactive	8	14	8	14	8	14	[1,2]
3	TdRdXC(SY)	/RxC High to /SYNC Valid	4	7	4	7	4	7	[2]
4	TdRXC(INT)	/RxC High to INT Valid	8	12	8	12	8	12	[1,2]
			2	3	2	3	2	3	[4]
5	TdTXC(REQ)	/TxC Low to /W//REQ Valid	5	8	5	8	5	8	[3]
6	TdTXC(W)	/TxC Low to Wait Inactive	5	11	5	11	5	11	[1,3]
7	TdTXC(DRQ)	/TxC Low to /DTR//REQ Valid	4	7	4	7	4	7	[3]
8	TdTXC(INT)	/TxC Low to /INT Valid	4	6	4	6	4	6	[1,3]
			2	3	2	3	2	3	[4]
9	TdSY(INT)	SYNC to INT Valid	2	3	2	3	2	3	[1,4]
10	TdEXT(INT)		2	3	2	3	2	3	[1,4]

#### Notes:

[1] Open drain-output, measured with open-drain test load.

[2] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[3] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[4] Units equal to /AS.

[5] Units in nanoseconds (ns), otherwise noted.







## Z80C30/Z85C30 CMOS Z-BUS® SCC SERIAL COMMUNICATION CONTROLLER

### FEATURES

- Z85C30 optimized for non-multiplexed bus microprocessors. Z80C30 optimized for multiplexed bus microprocessors.
- Pin compatible to NMOS versions
- Two independent, 0 to 4.1 Mbit/second, full-duplex channels, each with a separate crystal oscillator, baud rate generator, and Digital Phase-Locked Loop (DPLL) for clock recovery.
- Multi-protocol operation under program control; programmable for NRZ, NRZI, or FM data encoding.
- Asynchronous mode with five to eight bits and one, one and one-half, or two stop bits per character, programmable clock factor, break detection and generation; parity, overrun, and framing error detection.
- Synchronous mode with internal or external character synchronization on one or two synchronous characters and CRC generation and checking with CRC-16 or CRC-CCITT preset to either 1s or 0s.
- SDLC/HDLC mode with comprehensive frame-level control, automatic zero insertion and deletion, I-field residue handling, abort generation and detection, CRC generation and checking, and SDLC Loop.
- Software Interrupt Acknowledge feature (not with NMOS)
- Local Loopback and Auto Echo modes
- Supports T1 digital trunk
- Enhanced DMA support (not with NMOS)
  - 10 x 19-bit status FIFO
  - 14-bit byte counter
- Speeds:
  - Z85C30 - 8.5, 10, 16.384 MHz
  - Z80C30 - 8, 10 MHz

### GENERAL DESCRIPTION

The Zilog Serial Communications Controller, Z80C30/Z85C30 SCC, is a pin and software compatible CMOS member of the SCC family introduced by Zilog in 1981. It is a dual channel, multi-protocol data communications peripheral that easily interfaces to CPU's with either multiplexed or non-multiplexed address/data buses. The advanced CMOS process offers lower power consumption, higher performance, and superior noise immunity. The programming flexibility of the internal registers allows the SCC to be configured to satisfy a wide variety of serial communications applications. The many on-chip features such as baud rate generators, digital phase locked loops, and crystal oscillators dramatically reduce the need for external logic. Additional features including a 10x19-bit status FIFO and 14-bit byte counter were added to support high speed SDLC transfers using DMA controllers.

The SCC handles asynchronous formats, synchronous byte-oriented protocols such as IBM Bisync, and synchronous bit-oriented protocols such as HDLC and IBM SDLC. This versatile device supports virtually any serial data transfer application (cassette, diskette, tape drives, etc.)

The device can generate and check CRC codes in any synchronous mode and can be programmed to check data integrity in various modes. The SCC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

The daisy-chain interrupt hierarchy is also supported as is standard for Zilog peripheral components.

## GENERAL DESCRIPTION (Continued)

Note: All Signals with a preceding front slash, "/", are active Low, e.g.: B/W (WORD is active Low); /B/W (BYTE is active Low, only); /N/S (NORMAL and SYSTEM are both active Low).

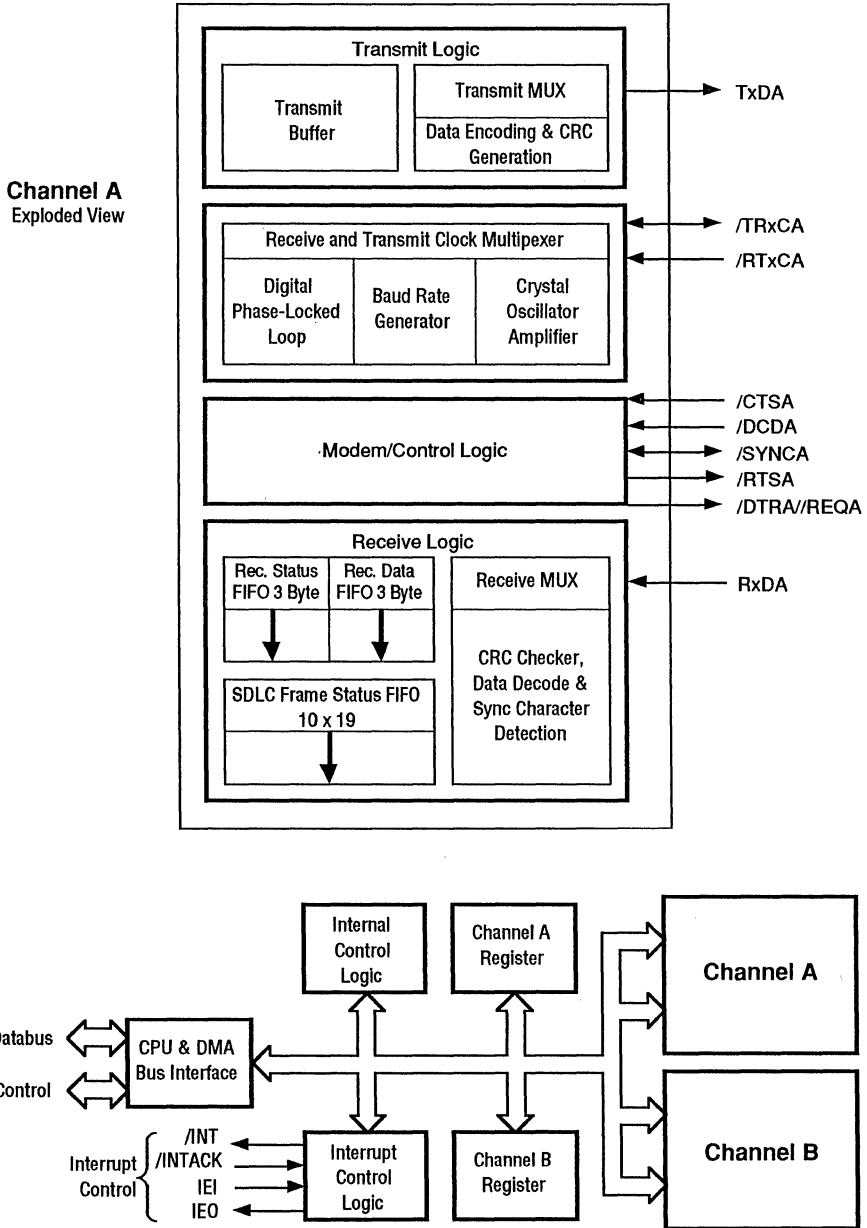


Figure 1. SCC Block Diagram

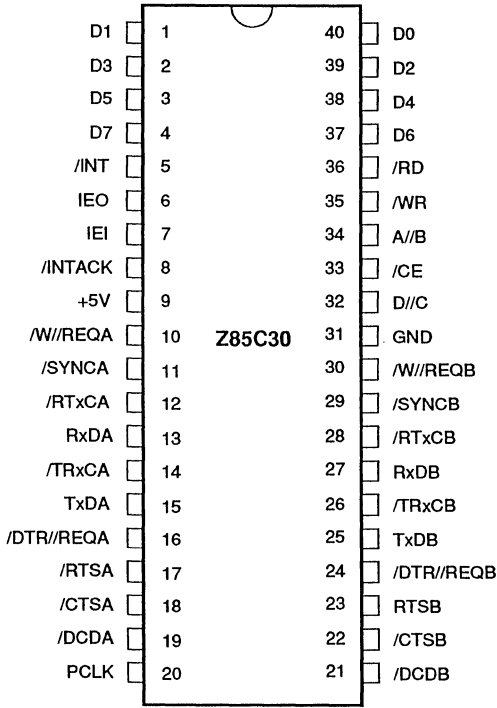


Figure 2. Z85C30 DIP Pin Assignments

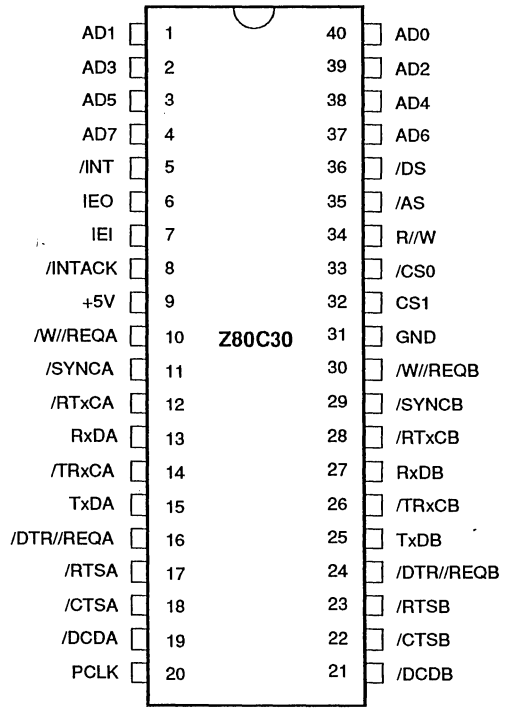


Figure 3. Z80C30 DIP Pin Assignments

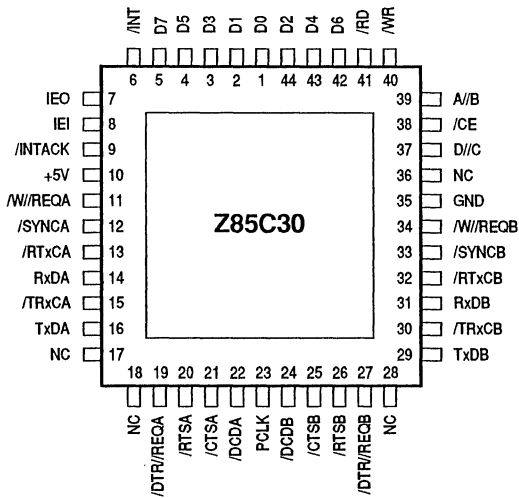


Figure 4. Z85C30 PLCC Pin Assignments

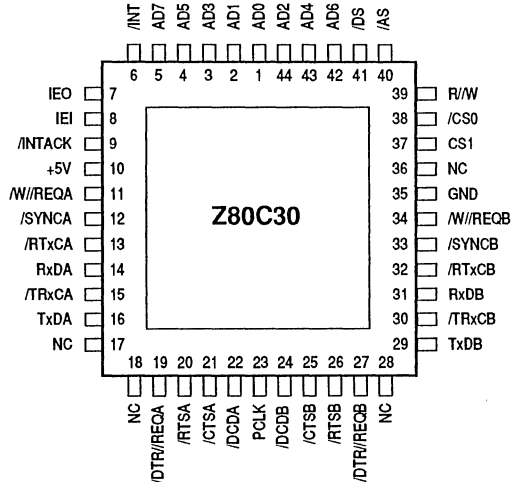


Figure 5. Z80C30 PLCC Pin Assignments

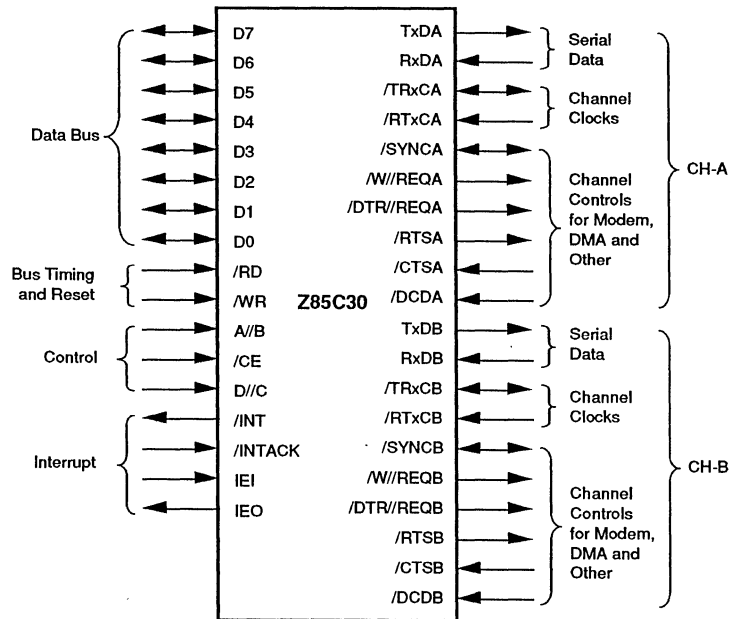


Figure 6. Z85C30 Pin Functions

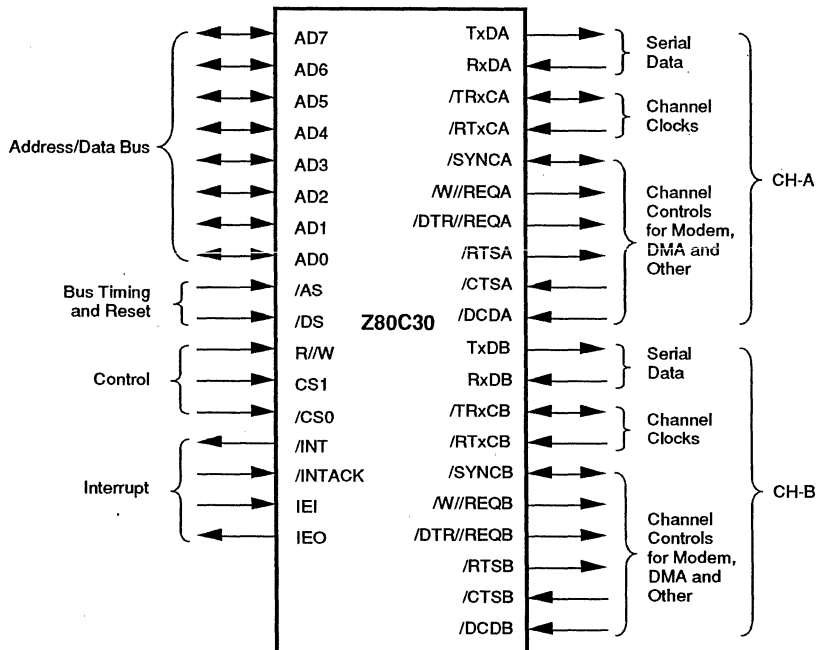


Figure 7. Z80C30 Pin Functions

---

## PIN DESCRIPTION

The following section describes the pin functions common to the Z85C30 and the Z80C30. Figures 2 and 3 detail the respective pin functions and pin assignments.

**/CTSA, /CTSB.** *Clear To Send* (inputs, active Low). If these pins are programmed for Auto Enables, a Low on the inputs enables the respective transmitters. If not programmed as Auto Enables, they may be used as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The SCC detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

**/DCDA, /DCDB.** *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if they are programmed for Auto Enables; otherwise, they are used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow rise-time signals. The SCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

**/DTR//REQA, /DTR//REQB.** *Data Terminal Ready/Request* (outputs, active Low). These outputs follow the state programmed into the DTR bit. They can also be used as general-purpose outputs or as Request lines for a DMA controller.

**IEI.** *Interrupt Enable In* (input, active High). IEI is used with IEO to form an interrupt daisy-chain when there is more than one interrupt driven device. A high IEI indicates that no other higher priority device has an interrupt under service or is requesting an interrupt.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing the SCC interrupt or the SCC is not requesting an interrupt (Interrupt Acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

**/INT.** *Interrupt Request* (output, open-drain, active Low). This signal is activated when the SCC requests an interrupt.

**/INTACK.** *Interrupt Acknowledge* (input, active Low). This signal indicates an active Interrupt Acknowledge cycle. During this cycle, the SCC interrupt daisy chain settles. When /RD or /DS becomes active, the SCC places an interrupt vector on the data bus (if IEI is High). /INTACK is latched by the rising edge of PCLK.

**PCLK.** *Clock* (input). This is the master SCC clock used to synchronize internal signals. PCLK is a TTL level signal. PCLK is not required to have any phase relationship with the master system clock.

**RxDA, RxDB.** *Receive Data* (inputs, active High). These signals receive serial data at standard TTL levels.

**/RTxCA, /RTxCB.** *Receive/Transmit Clocks* (inputs, active Low). These pins can be programmed in several different modes of operation. In each channel, /RTxC may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the Digital Phase-Locked Loop. These pins can also be programmed for use with the respective /SYNC pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in Asynchronous modes.

**/RTSA, /RTSB.** *Request To Send* (outputs, active Low). When the Request To Send (RTS) bit in Write Register 5 (Figure 11) is set, the /RTS signal goes Low. When the RTS bit is reset in the Asynchronous mode and Auto Enable is on, the signal goes High after the transmitter is empty. In Synchronous mode it strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

**/SYNCA, /SYNCB.** *Synchronization* (inputs or outputs, active Low). These pins can act either as inputs, outputs, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to /CTS and /DCD. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 (Figure 10) but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode, /SYNC must be driven Low for two receive clock cycles after the last bit in the synchronous character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of /SYNC.

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which synchronous characters are recognized. This synchronous condition is not latched, so these outputs are active each time a synchronization pattern is recognized (regardless of character boundaries). In SDLC mode, these pins act as outputs and are valid on receipt of a flag.

**TxDA, TxDB.** *Transmit Data* (outputs, active High). These output signals transmit serial data at standard TTL levels.

---

## PIN DESCRIPTION (Continued)

**/TRxCA, /TRxCB.** *Transmit/Receive Clocks* (inputs or outputs, active Low). These pins can be programmed in several different modes of operation. TRxC may supply the receive clock or the transmit clock in the input mode or supply the output of the Digital Phase-Locked Loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

**/W//REQA, /W//REQB.** *Wait/Request* (outputs, open-drain when programmed for a Wait function, driven High or Low when programmed for a Request function). These dual-purpose outputs may be programmed as Request lines for a DMA controller or as Wait lines to synchronize the CPU to the SCC data rate. The reset state is Wait.

### Z85C30

**A//B.** *Channel A/Channel B* (input). This signal selects the channel in which the read or write operation occurs.

**/CE.** *Chip Enable* (input, active Low). This signal selects the SCC for a read or write operation.

**D7-D0.** *Data Bus* (bidirectional, 3-state) These lines carry data and command to and from the SCC.

**D//C.** *Data/Control Select* (input). This signal defines the type of information transferred to or from the SCC. A High means data is transferred; a Low indicates a command.

**/RD.** *Read* (input, active Low). This signal indicates a read operation and when the SCC is selected, enables the SCC's bus drivers. During the Interrupt Acknowledge

cycle, this signal gates the interrupt vector onto the bus if the SCC is the highest priority device requesting an interrupt.

**/WR.** *Write* (input, active Low). When the SCC is selected, this signal indicates a write operation. The coincidence of /RD and /WR is interpreted as a reset.

### Z80C30

**AD7-AD0.** *Address/Data Bus* (bidirectional, active High, 3-state) These multiplexed lines carry register addresses to the SCC as well as data or control information.

**/AS.** *Address Strobe* (input, active Low). Addresses on AD7-AD0 are latched by the rising edge of this signal.

**/CS0.** *Chip Select 0* (input, active Low). This signal is latched concurrently with the addresses on AD7-AD0 and must be active for the intended bus transaction to occur.

**CS1.** *Chip Select 1* (input, active High). This second select signal must also be active before the intended bus transaction can occur. CS1 must remain active throughout the transaction.

**/DS.** *Data strobe* (input, active Low). This signal provides timing for the transfer of data into and out of the SCC. If /AS and /DS coincide, this is interpreted as a reset.

**R//W.** *Read/Write* (input). This signal specifies whether the operation to be performed is a read or a write.

---

## FUNCTIONAL DESCRIPTION

The architecture of the SCC is described from two points of view: as a datacommunications device which transmits and receives data in a wide variety of protocols; as a microprocessor peripheral in which the SCC offers valuable features such as vectored interrupts and DMA support.

The SCC's peripheral and datacommunication are described in the following sections. A block diagram is shown in Figure 1. The details of the communications between the receive and transmit logic to the system bus is shown in Figures 8 and 9. The features and data path for each of the SCC's A and B channels is identical. See the SCC Technical Manual for full details on using the SCC.

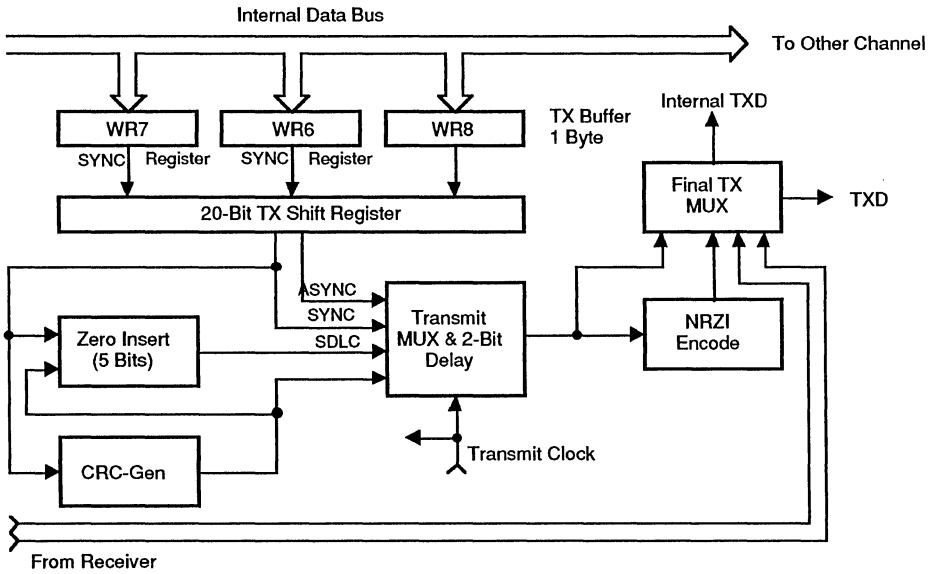


Figure 8. SCC Transmit Data Path

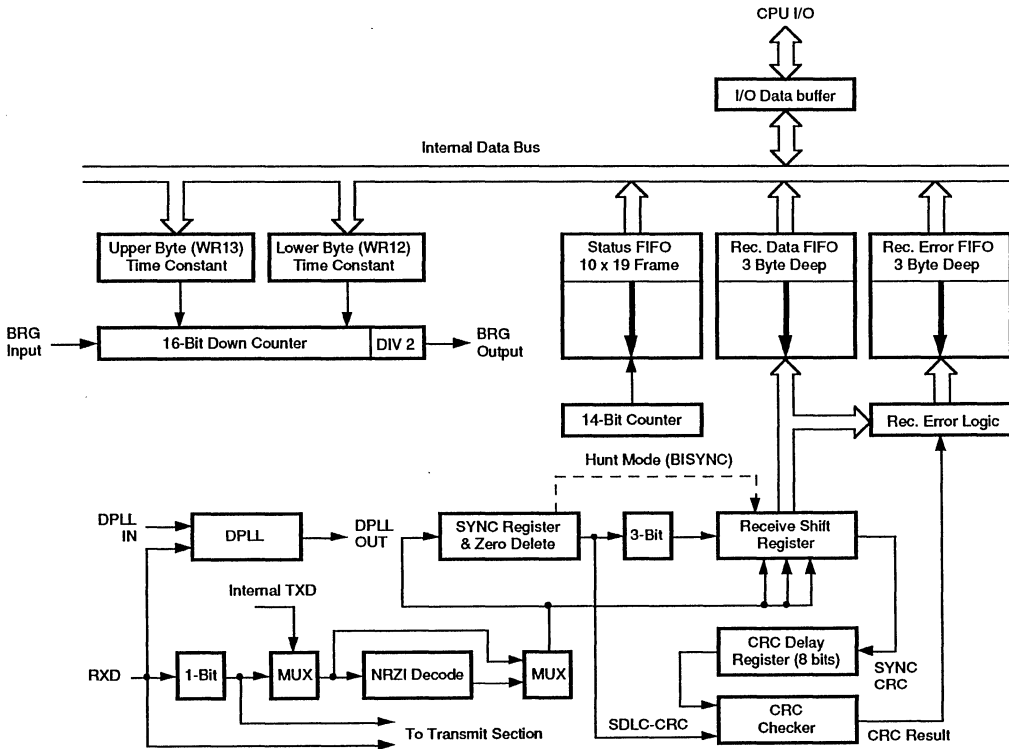


Figure 9. SCC Receive Data Path



## FUNCTIONAL DESCRIPTION (Continued)

### I/O Interface Capabilities

System communication to and from the SCC is done through the SCC's register set. There are sixteen write registers and eight read registers. Table 1 lists all of the SCC's registers and a brief description of their functions. Throughout this document, the write and read registers are referenced with the following notation: "WR" for Write Register and "RR" for Read Register. For example:

WR4A Write Register 4 for channel A  
RR3 Read Register 3 for either/both channels

**Table 1. SCC Read and Write Registers**

Read Register Functions	
RR0	Transmit/Receive buffer status and External status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only) Unmodified interrupt vector (Channel A only)
RR3	Interrupt Pending bits (Channel A only)
RR8	Receive Buffer
RR10	Miscellaneous status
RR12	Lower byte of baud rate generator time constant
RR13	Upper byte of baud rate generator time constant
RR15	External/Status interrupt information
Write Register Functions	
WR0	CRC initialize, initialization commands for the various modes, Register Pointers.
WR1	Transmit/Receive interrupt and data transfer mode definition
WR2	Interrupt vector (accessed through either channel)
WR3	Receive parameters and control
WR4	Transmit/Receive miscellaneous parameters and modes
WR5	Transmit parameters and controls
WR6	Sync characters or SDLC address field
WR7	Sync character or SDLC flag
WR8	Transmit buffer
WR9	Master interrupt control and reset (accessed through either channel)
WR10	Miscellaneous transmitter/receiver control bits
WR11	Clock mode control
WR12	Lower byte of baud rate generator time constant
WR13	Upper byte of baud rate generator time constant
WR14	Miscellaneous control bits
WR15	External/Status interrupt control

There are three choices to move data into and out of the SCC: Polling, interrupt (vectored and non-vectored), and Block Transfer. The Block Transfer mode can be implemented under CPU or DMA control.

#### Polling

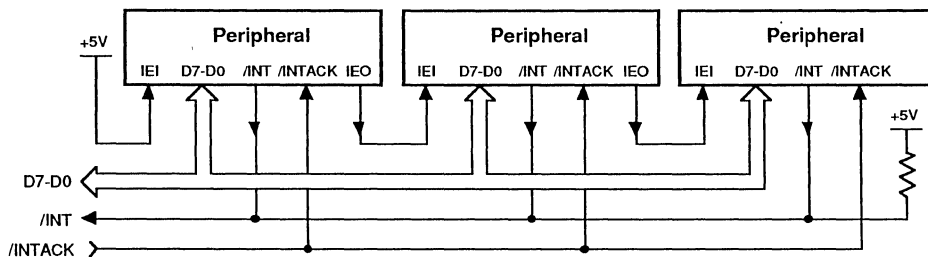
When polling, all interrupts are disabled. Three status registers in the SCC are automatically updated whenever any function is performed. For example, End-Of-Frame in SDLC mode sets a bit in one of these status registers. The purpose of polling is for the CPU to periodically read a status register until the register contents indicate the need for data to be transferred. Only one register needs to be read; depending on its contents, the CPU either writes data, reads data, or continues. Two bits in the register indicate the need for data transfer. An alternative is a poll of the Interrupt Pending register to determine the source of an interrupt. The status for both channels resides in one register.

#### Interrupts

The SCC's interrupt structure supports vectored and nested interrupts. Nested interrupts are supported with the interrupt acknowledge feature (/INTACK pin) of the SCC. This allows the CPU to recognize the occurrence of an interrupt, and re-enable higher priority interrupts. Because an INTACK cycle will release the /INT pin from the active state, a higher priority SCC interrupt or another higher priority device can interrupt the CPU. When an SCC responds to an Interrupt Acknowledge signal (INTACK) from the CPU, an interrupt vector can be placed on the data bus. This vector is written in WR2 and may be read in RR2. To speed interrupt response time, the SCC can modify three bits in this vector to indicate status. If the vector is read in Channel A, status is never included; if it is read in Channel B, status is always included.

Each of the six sources of interrupts in the SCC (Transmit, Receive, and External/Status interrupts in both channels) has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt Under Service (IUS), and Interrupt Enable (IE). Operation of the IE bit is straightforward. If the IE bit is set for a given interrupt source, then that source can request interrupts. The exception is when the MIE (Master Interrupt Enable) bit in WR9 is reset and no interrupts can be requested. The IE bits are write only.

The other two bits are related to the interrupt priority chain (Figure 10). As a microprocessor peripheral, the SCC may request an interrupt only when no higher priority device is requesting one, e.g., when IE1 is High. If the device in question requests an interrupt, it pulls down /INT. The CPU then responds with /INTACK, and the interrupting device places the vector on the data bus.



**Figure 10. SCC Interrupt Priority Schedule**

The SCC can also execute an interrupt acknowledge cycle through software. In some CPU environments it is difficult to create the /INTACK signal with the necessary timing to acknowledge interrupts and allow the nesting of interrupts. In these cases, the /INTACK signal can be created with a software command to the SCC.

In the SCC, the Interrupt Pending (IP) bit signals a need for interrupt servicing. When an IP bit is 1 and the IEI input is High, the /INT output is pulled Low, requesting an interrupt. In the SCC, if the IE bit isn't set by enabling interrupts, then the IP for that source is never set. The IP bits are readable in RR3A.

The IUS bits signal that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority in the SCC and external to the SCC are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the SCC being pulled Low and propagated to subsequent peripherals. An IUS bit is set during an Interrupt Acknowledge cycle if there are no higher priority devices requesting interrupts.

There are three types of interrupts: Transmit, Receive, and External/Status. Each interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receiver, Transmit, and External/Status interrupts prioritized in that order within each channel.

When enabled, the receiver can interrupt the CPU in one of three ways:

1. Interrupt on First Receive Character or Special Receive Condition.
2. Interrupt on All Receive Characters or Special Receive Conditions.
3. Interrupt on Special Receive Conditions Only.

Interrupt on First Character or Special Condition and Interrupt on Special Condition Only are typically used with the Block Transfer mode. A special Receive Condition is one of the following: receiver overrun, framing error in Asynchronous mode, end-of-frame in SDLC mode and, optionally, a parity error. The Special Receive Condition interrupt is different from an ordinary receive character available interrupt only by the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt occurs from Special Receive Conditions any time after the first receive character interrupt.

The main function of the External/Status interrupt is to monitor the signal transitions of the /CTS, /DCD, and /SYNC pins, however, an External/Status interrupt is also caused by a Transmit Underrun condition; a zero count in the baud rate generator; by the detection of a Break (Asynchronous mode), Abort (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the Abort or EOP has a special feature allowing the SCC to interrupt when the Abort or EOP sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Abort condition in external logic in SDLC mode. In SDLC Loop mode, this feature allows secondary stations to recognize the primary station wishes to regain control of the loop during a poll sequence.

### Software Interrupt Acknowledge

On the CMOS version of the SCC, the SCC interrupt acknowledge cycle can be initiated through software. If Write Register 9 (WR9) bit D5 is set, Read Register 2 (RR2) results in an interrupt acknowledge cycle to be executed internally. Like a hardware INTACK cycle, a software acknowledge causes the /INT pin to return high, the IEO pin to go low and set the IUS latch for the highest priority interrupt pending.

## FUNCTIONAL DESCRIPTION (Continued)

Similar to when the hardware INTACK signal can be used, a software acknowledge cycle requires that a Reset Highest IUS command be issued in the interrupt service routine. Whenever an interrupt acknowledge cycle is used, hardware or software, a reset highest IUS command is required. If RR2 is read from channel A, the unmodified vector is returned. If RR2 is read from channel B, then the vector is modified to indicate the source of the interrupt. The Vector Includes Status (VIS) and No Vector (NV) bits in WR9 are ignored when bit D5 is set to 1.

When the INTACK and IEI pins are not being used, they should be pulled up to  $V_{cc}$  through a resistor (10 kohm typical).

**CPU/DMA Block Transfer.** The SCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the /WAIT//REQUEST output in conjunction with the Wait/

Request bits in WR1. The /WAIT//REQUEST output can be defined under software control as a WAIT line in the CPU Block Transfer mode or as a REQUEST line in the DMA Block Transfer mode.

To a DMA controller, the SCC REQUEST output indicates that the SCC is ready to transfer data to or from memory. To the CPU, the WAIT line indicates that the ESCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The /DTR//REQUEST line allows full-duplex operation under DMA control.

## SCC Data Communications Capabilities

The SCC provides two independent full-duplex programmable channels for use in any common asynchronous or synchronous data communication protocols (Figure 11). Each of the datacommunication channels has identical features and capabilities.

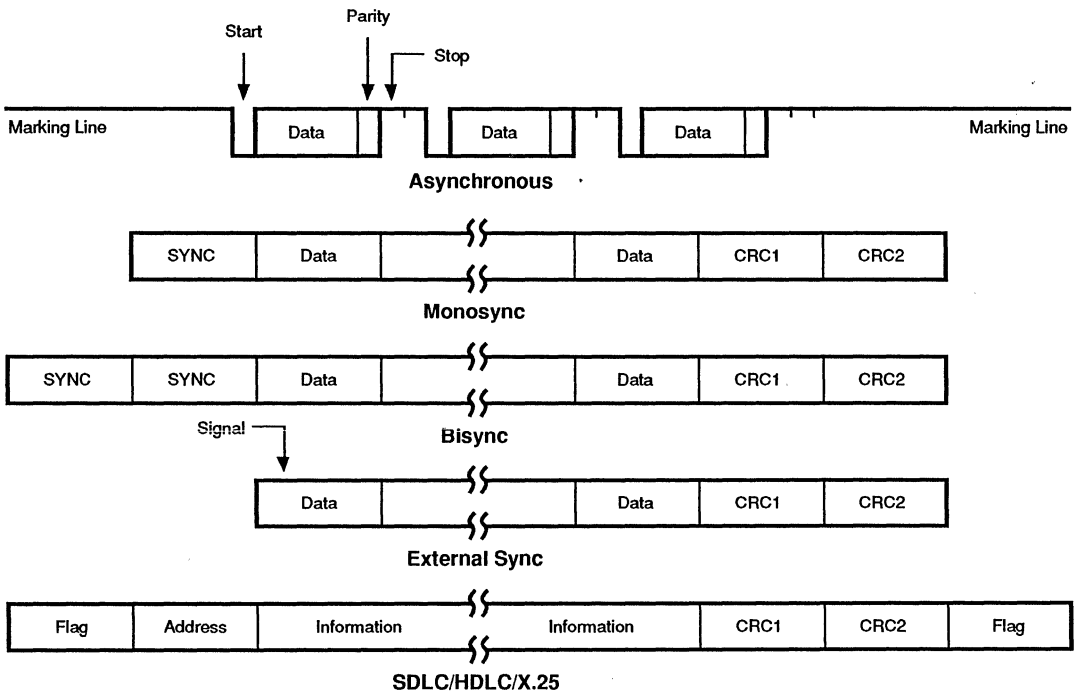


Figure 11. Some SCC Protocols

### Asynchronous Modes

Send and Receive is accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half, or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB pins). If the Low does not persist (e.g., a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur. Vectored interrupts allow fast servicing or error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

The SCC does not require symmetric transmit and receive clock signals - a feature allowing use of the wide variety of clock sources. The transmitter and receiver handle data at a rate supplied to the receive and transmit clock inputs. In Asynchronous modes, the SYNC pin may be programmed as an input used for functions such as monitoring a ring indicator.

### Synchronous Modes

The SCC supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols are handled in several modes. They allow character synchronization with a 6-bit or 8-bit sync character (Monosync), and a 12-bit or 16-bit synchronization pattern (Bisync), or with an external sync signal. Leading sync characters are removed without interrupting the CPU.

Five or 7-bit synchronous characters are detected with 8- or 16-bit patterns in the SCC by overlapping the larger pattern across multiple incoming synchronous characters as shown in Figure 12.

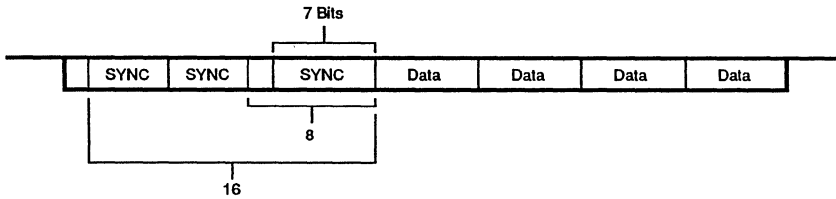


Figure 12. Detecting 5- or 7-Bit Synchronous Characters

CRC checking for Synchronous byte oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols such as IBM Bisync.

Both CRC-16 ( $X^{16} + X^{15} + X^2 + 1$ ) and CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) error checking polynomials are supported. Either polynomial may be selected in all Synchronous modes. Users may preset the CRC generator and checker to all 1's or all 0's. The SCC also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows for high speed transmissions under DMA control, with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 6-, 8-, or 16-bit sync characters, regardless of the programmed character length.

### SDLC Mode

The SCC supports Synchronous bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero insertion, and CRC generation. A special command is used to abort a frame in transmission. At the end of a message, the SCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line consisting of continuous flag characters or a steady marking condition.

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort can be issued. The SCC may also be programmed to send an abort itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent, allowing reception of a message with no prior information about the character structure in the information field of a frame.

## FUNCTIONAL DESCRIPTION (Continued)

The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the /SYNC pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored.

The number of address bytes are extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all 0's inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. In SDLC mode, the SCC must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all 1's or all 0's. The CRC is inverted before transmission and the receiver checks against the bit pattern 0001110100001111.

NRZ, NRZI or FM coding may be used in any 1x mode. The parity options available in Asynchronous modes are available in Synchronous modes.

**SDLC Loop Mode.** The SCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow on the loop and any number of secondary stations. In SDLC Loop mode, the SCC performs the functions of a secondary station while an SCC operating in regular SDLC mode acts as a controller (Figure 13). SDLC loop mode can be selected by setting WR10 bit D1.

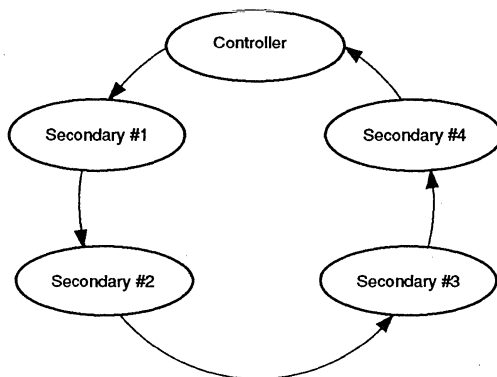


Figure 13. An SDLC Loop

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop and, in fact, passes these messages to the rest of the loop by retransmitting them with a one-bit-time delay. The secondary station places its own message on the loop only at specific times. The controller signals that secondary stations can transmit messages by sending a special character, called an EOP (End Of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages, this bit pattern is unique and easily recognized.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit appends their messages to the message of the first secondary station by the same process. Any secondary stations without messages to send merely echo the incoming message and are prohibited from placing messages on the loop (except upon recognizing an EOP). In SDLC Loop mode, NRZ, NRZI, and FM coding may all be used.

The SCC's ability to receive high speed back-to-back SDLC frames is maximized by a 10- deep by 19-bit wide status FIFO. When enabled (through WR15, bit D2), it provides the DMA the ability to continue to transfer data into memory so that the CPU can examine the message later. For each SDLC frame, a 14-bit byte count and 5 status/error bits are stored. The byte count and status bits are accessed through Read Registers 6 and 7. Read Registers 6 and 7 are only accessible when the SDLC FIFO is enabled. The 10x19 status FIFO is separate from the 3-byte receive data FIFO.

### Baud Rate Generator

Each channel in the SCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On startup, the flip-flop on the output is set in a High state, the value in the time constant register is loaded into the counter, and the counter starts counting down. The output of the baud rate generator toggles upon reaching 0, the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the Digital Phase-Locked Loop (see next section).

If the receive clock or transmit clock is not programmed to come from the TRxC pin, the output of the baud rate generator may be echoed out via the TRxC pin.

The following formula relates the time constant to the baud rate where PCLK or RTxC is the baud rate generator input frequency in Hertz. The clock mode is 1, 16, 32, or 64, as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select 1 and Asynchronous should select 16, 32 or 64.

$$\text{Time Constant} = \frac{\text{PCLK or RTxC Frequency}}{2(\text{Baud Rate})(\text{Clock Rate})} - 2$$

### Digital Phase-Locked Loop

The SCC contains a Digital Phase-Locked Loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock is then used as the SCC receive clock, the transmit clock, or both. When the DPLL is selected as the transmit clock source, it will provide a jitter free clock output that is the DPLL input frequency divided by the appropriate divisor for the selected encoding technique.

For NRZI encoding, the DPLL counts the 32x clock to create nominal bit times. As the 32x clock is counted, the DPLL is searching the incoming data stream for edges (either 1 to 0, or 0 to 1). Whenever an edge is detected, the DPLL makes a count adjustment (during the next counting

cycle), producing a terminal count closer to the center of the bit cell.

For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16 and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15 to 16 counting transition.

The 32x clock for the DPLL can be programmed to come from either the RTxC input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the SCC via the TRxC pin (if this pin is not being used as an input).

### Data Encoding

The SCC may be programmed to encode and decode the serial data in four different ways (Figure 14). In NRZ encoding, a 1 is represented by a High level and a 0 is represented by a Low level. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. In FM1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A 1 is represented by an additional transition at the center of the bit cell and a 0 is represented by no additional transition at the center of the bit cell. In FM0 (bi-phase space), a transition occurs at the beginning of every bit cell. A 0 is represented by an additional transition at the center of the bit cell, and a 1 is represented by no additional transition at the center of the bit cell. In addition to these four methods, the SCC can be used to decode Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0 to 1, the bit is a 0. If the transition is 1 to 0, the bit is a 1.

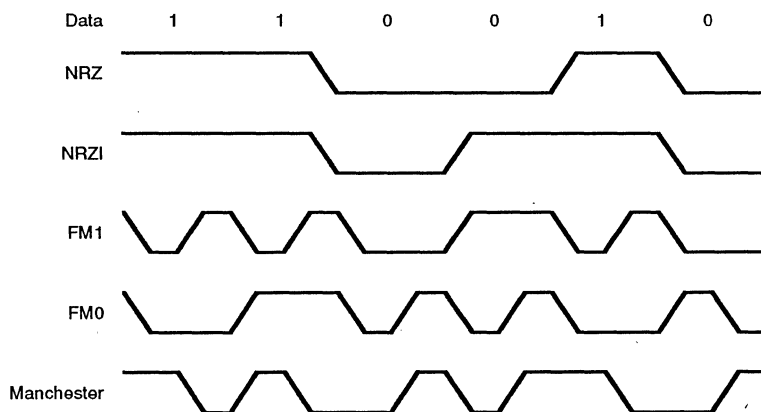


Figure 14. Data Encoding Methods

---

## FUNCTIONAL DESCRIPTION (Continued)

### Auto Echo and Local Loopback

The SCC is capable of automatically echoing everything it receives. This feature is useful mainly in Asynchronous modes, but works in Synchronous and SDLC modes as well. Auto Echo mode (TxD is RxD) is used with NRZI or FM encoding with no additional delay because the data stream is not decoded before retransmission. In Auto Echo mode, the /CTS input is ignored as a transmitter enable (although transitions on this input can still cause interrupts if programmed to do so). In this mode, the transmitter is actually bypassed and the programmer is responsible for disabling transmitter interrupts and /WAIT//REQUEST on transmit.

The SCC is also capable of local loopback. In this mode, TxD or RxD is just like Auto Echo mode. However, in Local Loopback mode the internal transmit data is tied to the internal receive data and RxD is ignored (except to be echoed out via TxD). The /CTS and /DCD inputs are also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in Asynchronous, Synchronous and SDLC modes with NRZ, NRZI or FM coding of the data stream.

### SDLC FIFO Frame Status FIFO Enhancement

The SCC's ability to receive high speed back-to-back SDLC frames is maximized by a 10- deep by 19-bit wide status FIFO. When enabled (through WR15, bit D2), it provides the DMA the ability to continue to transfer data into memory so that the CPU can examine the message later. For each SDLC frame, a 14-bit byte count and 5 status/error bits are stored. The byte count and status bits are accessed through Read Registers 6 and 7. Read Registers 6 and 7 are only accessible when the SDLC FIFO is enabled. The 10x19 status FIFO is separate from the 3-byte receive data FIFO.

When the enhancement is enabled, the status in read register 1 (RR1) and byte count for the SDLC frame are stored in the 10 x 19 bit status FIFO. This allows the DMA controller to transfer the next frame into memory while the CPU verifies that the message was properly received.

Summarizing the operation; data is received, assembled, and loaded into the eight byte FIFO before being transferred to memory by the DMA controller. When a flag is received at the end of an SDLC frame, the frame byte count from the 14-bit counter and five status bits are loaded into the status FIFO for verification by the CPU. The CRC

checker is automatically reset in preparation for the next frame which can begin immediately. Since the byte count and status are saved for each frame, the message integrity is verified at a later time. Status information for up to 10 frames is stored before a status FIFO overrun can occur.

If a frame is terminated with an ABORT, the byte count is loaded to the status FIFO and the counter reset for the next frame.

### FIFO Detail

For a better understanding of details of the FIFO operation, refer to the block diagram in Figure 15.

### Enable/Disable

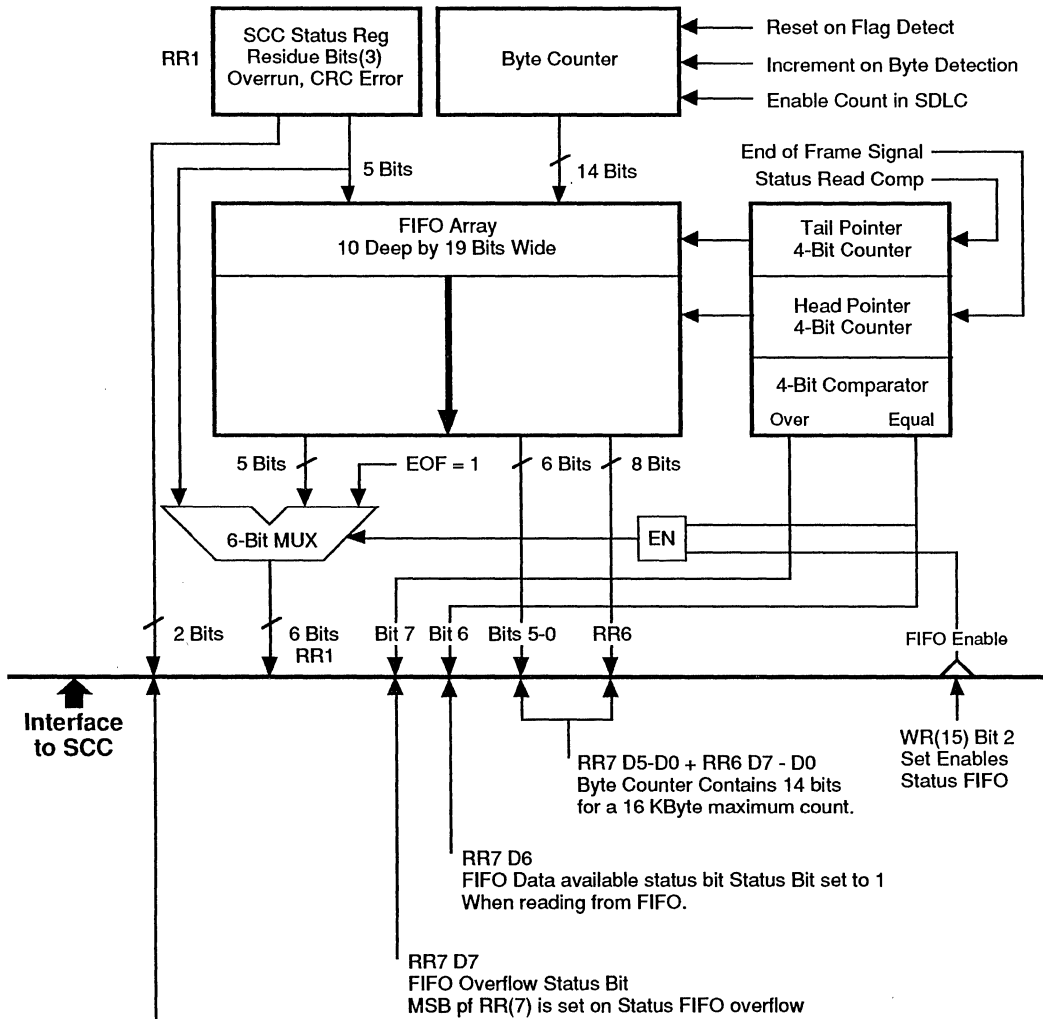
This FIFO is implemented so that it is enabled when WR15, bit D2, is set and the SCC is in the SDLC/HDLC mode. Otherwise, the status register contents bypass the FIFO and go directly to the bus interface (the FIFO pointer logic is reset either when disabled or via a channel or power-on reset). When the FIFO mode is disabled, the SCC is completely downward compatible with the NMOS Z8530. The FIFO mode is disabled on power-up (WR15 D2 is set to 0 on reset). The effects of backward compatibility on the register set are that RR4 is an image of RR0, RR5 is an image of RR1, RR6 is an image of RR2 and RR7 is an image of RR3. For the details of the added registers, refer to Figure 18. The status of the FIFO Enable signal is obtained by reading RR15, bit D2. If the FIFO is enabled, the bit will be set to 1; otherwise, it will be reset.

### Read Operation

When WR15 bit D2 is set and the FIFO is not empty, the next read to status register RR1 or the additional registers RR7 and RR6, are from the FIFO. Reading status register RR1 causes one location of the FIFO to be emptied, so status is read after reading the byte count, otherwise the count is incorrect. Before the FIFO underflows, it is disabled. In this case, the multiplexer is switched to allow status to read directly from the status register and reads from RR7 and RR6 contain bits that are undefined. Bit D6 of RR7 (FIFO Data Available) is used to determine if status data is coming from the FIFO or directly from the status register, since it is set to 1 whenever the FIFO is not empty.

Since not all status bits are stored in the FIFO, the All Sent, Parity, and EOF bits bypass the FIFO. The status bits sent through the FIFO are Residue Bits (3), Overrun, and CRC Error.

### Frame Status FIFO Circuitry



In SDLC Mode the following definitions apply.

- All Sent bypasses MUX and equals contents of SCC Status Register.
- Parity Bits bypasses MUX and does the same.
- EOF is set to 1 whenever reading from the FIFO.

Figure 15. SDLC Frame Status FIFO



## FUNCTIONAL DESCRIPTION (Continued)

The sequence for proper operation of the byte count and FIFO logic is to read the registers in the following order: RR7, RR6, and RR1 (reading RR6 is optional). Additional logic prevents the FIFO from being emptied by multiple reads from RR1. The read from RR7 latches the FIFO empty/full status bit (D6) and steers the status multiplexer to read from the SCC megacell instead of the status FIFO (since the status FIFO is empty). The read from RR1 allows an entry to be read from the FIFO (if the FIFO was empty, logic was added to prevent a FIFO underflow condition).

### Write Operation

When the end of an SDLC frame (EOF) has been received and the FIFO is enabled, the contents of the status and byte-count registers are loaded into the FIFO. The EOF signal is used to increment the FIFO. If the FIFO overflows, the RR7 bit D7 (FIFO Overflow) is set to indicate the overflow. This bit and the FIFO control logic is reset by disabling and re-enabling the FIFO control bit (WR15 bit D2). For details of FIFO control timing during an SDLC frame, refer to Figure 16.

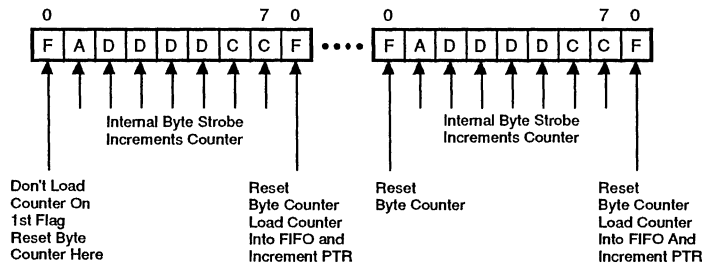


Figure 16. SDLC Byte Counting Detail

## PROGRAMMING

The SCC contains write registers in each channel that are programmed by the system separately to configure the functional personality of the channels.

### Z85C30

In the SCC, the data registers are directly addressed by selecting a High on the D/JC pin. With all other registers (except WR0 and RRO), programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to WR0 and contains three bits that point to the selected register. The second write is the actual control word for the selected register, and if the second operation is read, the selected read register is accessed. All of the SCC registers, including the data registers, may be accessed in this fashion. The pointer bits are automatically cleared after the read or write operation so that WR0 (or RRO) is addressed again.

### Z80C30

All SCC registers are directly addressable. How the SCC decodes the address placed on the address/data bus at the beginning of a Read or Write cycle is controlled by a command issued in WR0B. In the Shift Right mode the

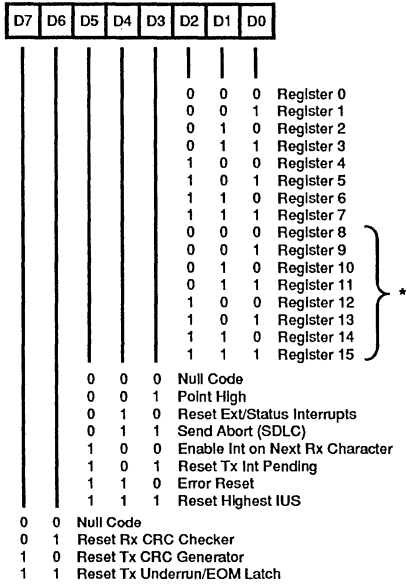
channel select A/B is taken from AD0 and the state of AD5 is ignored. In the Shift Left mode the channel select A/B is taken from AD5 and the state of AD0 is ignored. AD7 and AD6 are always ignored as address bits and the register address itself occupies AD4-AD1.

### Z85C30/Z80C30 Setup

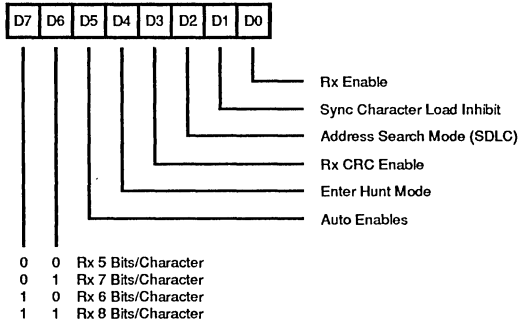
**Initialization.** The system program first issues a series of commands to initialize the basic mode of operation. This is followed by other commands to qualify conditions within the selected mode. For example, in the Asynchronous mode, character length, clock rate, number of stop bits, and even or odd parity should be set first. Then the interrupt mode is set, and finally, the receiver and transmitter are enabled.

**Write Registers.** The SCC contains 15 write registers (16 counting the transmit buffer) in each channel. These write registers are programmed separately to configure the functional "personality" of the channels. There are two registers (WR2 and WR9) shared by the two channels that are accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits and reset commands. Figure 17 shows the format of each write register.

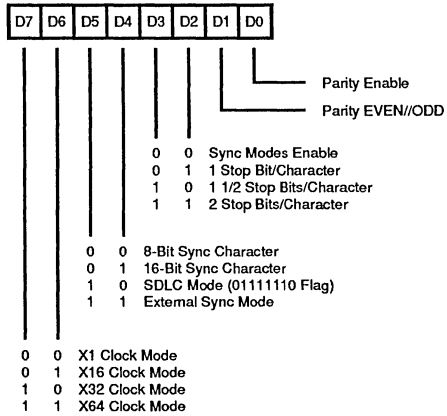
Write Register 0 (non-multiplexed bus mode)



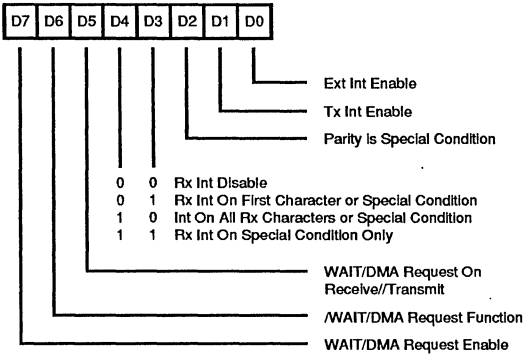
Write Register 3



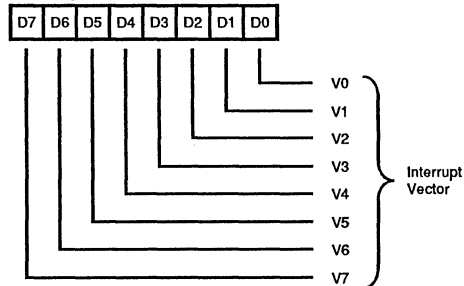
Write Register 4



Write Register 1



Write Register 2



Write Register 5

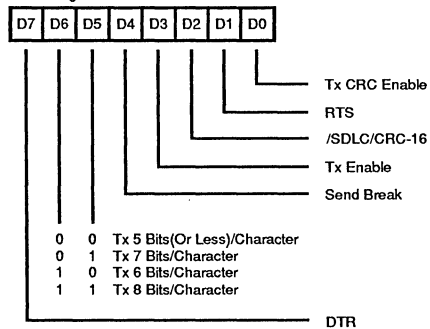
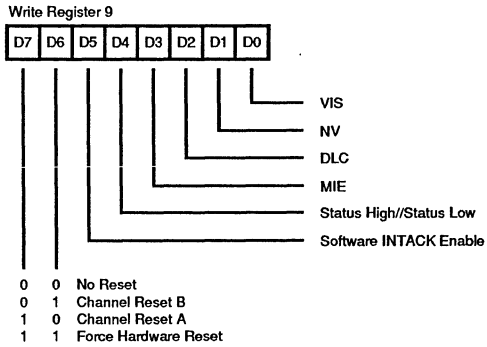
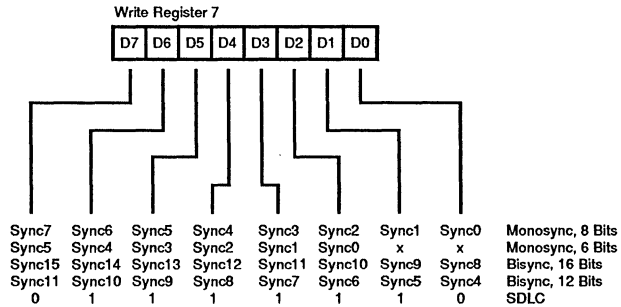
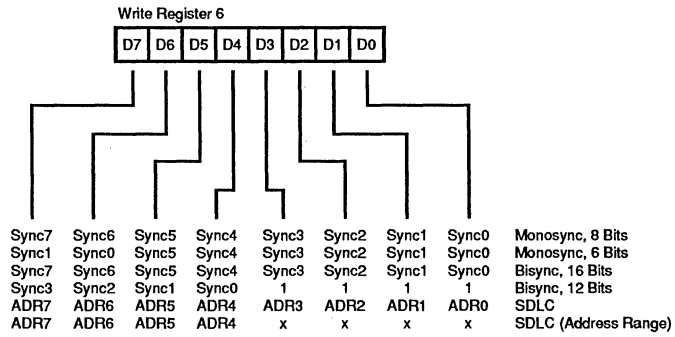


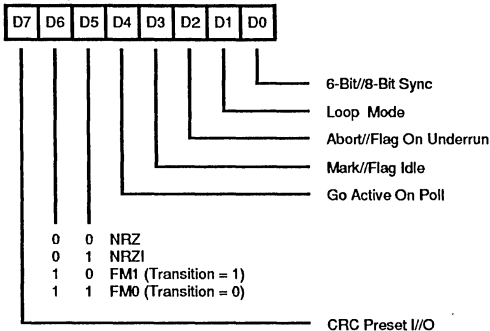
Figure 17. Write Register Bit Functions

**PROGRAMMING (Continued)**

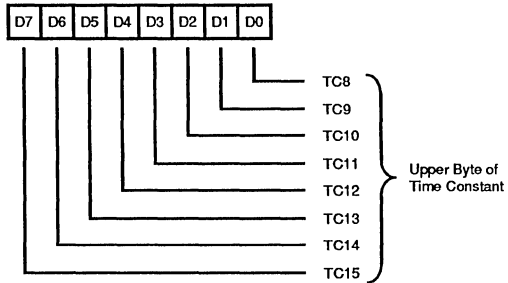


**Figure 17. Write Register Bit Functions (Continued)**

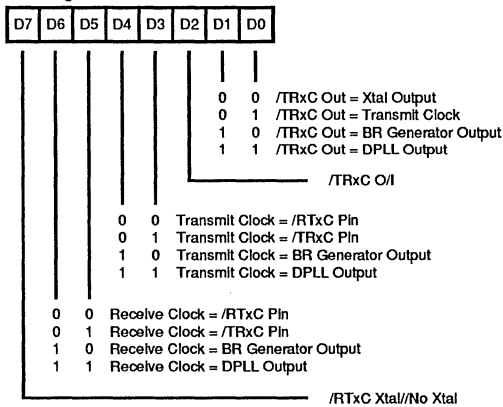
Write Register 10



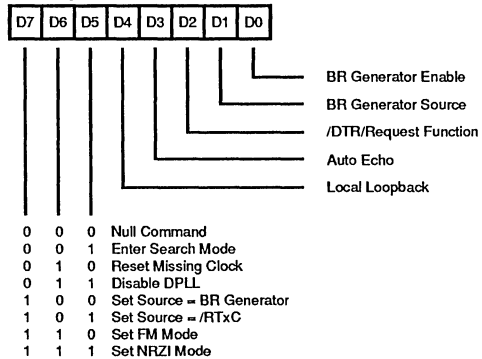
Write Register 13



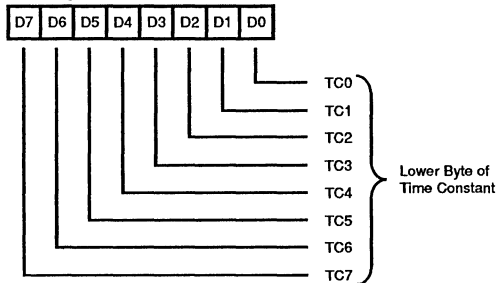
Write Register 11



Write Register 14



Write Register 12



Write Register 15

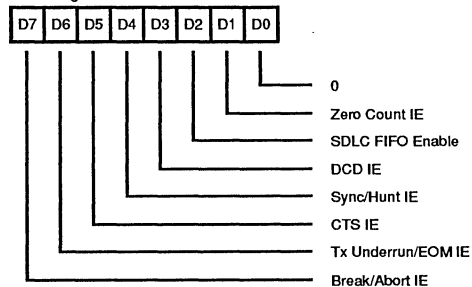


Figure 17. Write Register Bit Functions (Continued)

## PROGRAMMING (Continued)

**Read Registers.** The SCC contains ten read registers (eleven, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers (RR12 and RR13) are read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt

vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A only - , Figure 18). RR6 and RR7 contain the information in the SDLC Frame Status FIFO, but is only read when WR15 D2 is set (Figure 15).

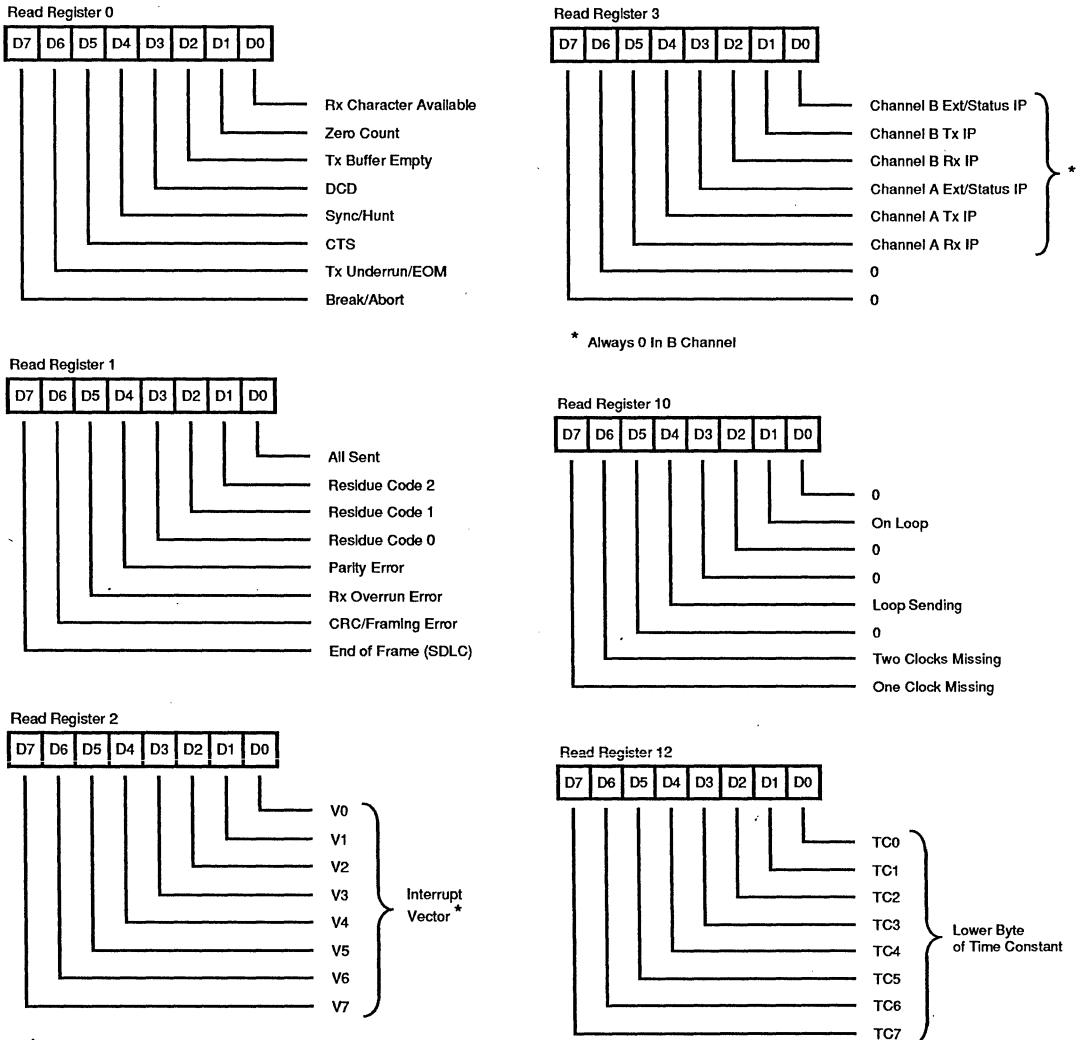


Figure 18. Read Register Bit Functions

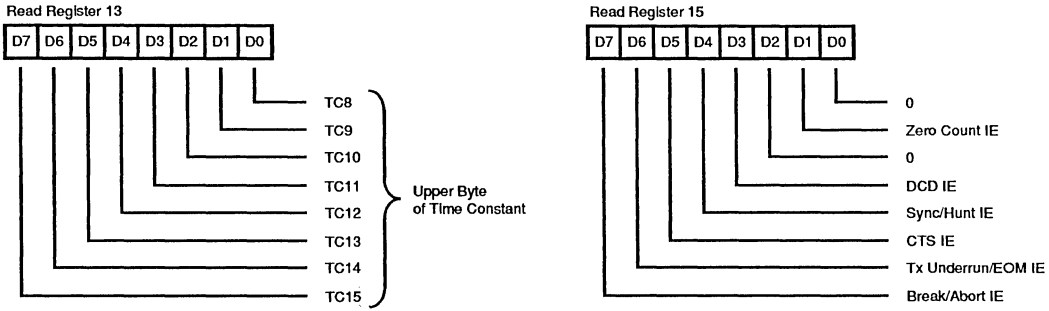


Figure 18. Read Register Bit Functions (Continued)

### Z85C30 Timing

The SCC generates internal control signals from the  $/WR$  and  $/RD$  that are related to PCLK. Since PCLK has no phase relationship with  $/WR$  and  $/RD$ , the circuitry generating the internal control signals provides time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK. The recovery time applies only between bus transactions involving the SCC. The recovery time required for proper operation is specified from the falling edge of  $/WR$  or  $/RD$  in the first transaction involving the SCC to the falling edge of  $/WR$  or  $/RD$  in the second

transaction involving the SCC. This time must be at least 4 PCLKs regardless of which register or channel is being accessed.

#### Read Cycle Timing

Figure 19 illustrates Read cycle timing. Addresses on  $A//B$  and  $D//C$  and the status on  $/INTACK$  must remain stable throughout the cycle. If  $/CE$  falls after  $/RD$  falls, or if it rises before  $/RD$  rises, the effective  $/RD$  is shortened.

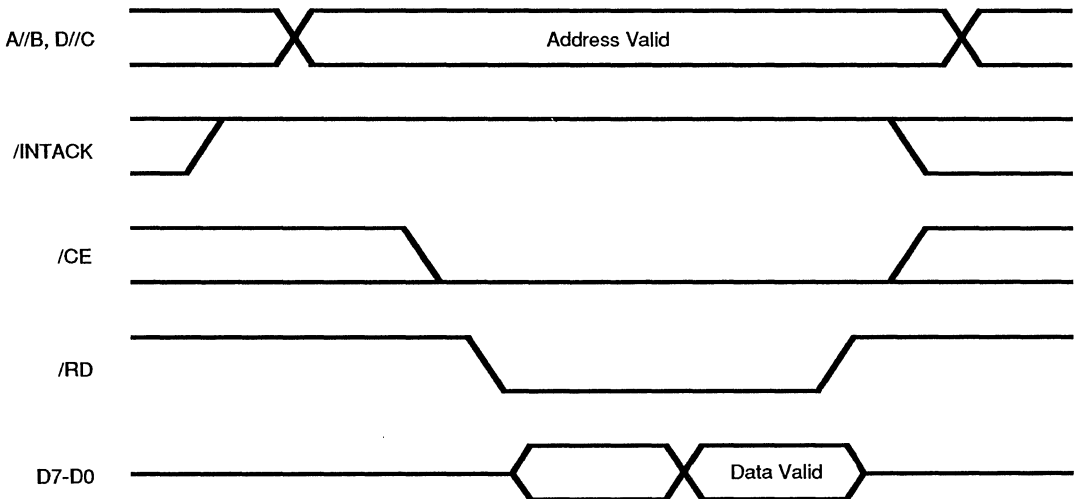


Figure 19. Read Cycle Timing

---

## PROGRAMMING (Continued)

### Write Cycle Timing

Figure 20 illustrates Write cycle timing. Addresses on A//B and D//C and the status on /INTACK must remain stable throughout the cycle. If /CE falls after /WR falls, or if it rises

before /WR rises, the effective /WR is shortened. Data must be valid before the falling edge of /WR.

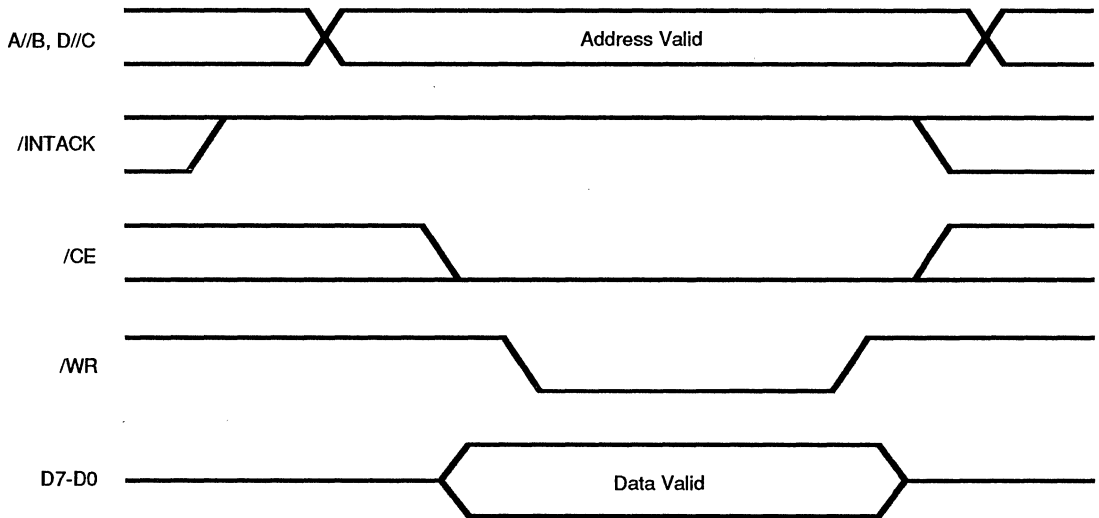


Figure 20. Write Cycle Timing

---

### Interrupt Acknowledge Cycle Timing

Figure 21 illustrates Interrupt Acknowledge cycle timing. Between the time /INTACK goes Low and the falling edge of /RD, the internal and external IEI/IEO daisy chains settle. If there is an interrupt pending in the SCC and IEI is High when /RD falls, the Acknowledge cycle is intended for the SCC. In this case, the SCC may be programmed to respond to /RD Low by placing its interrupt vector

on D7-D0. It then sets the appropriate Interrupt-Under-Service latch internally. If the external daisy chain is not used, then AC parameter #38 is required to settle the interrupt priority daisy chain internal to the SCC. If the external daisy chain is used, the user should follow the equation in AC Characteristics, Note 5 of the Read/Write Timing Table for calculating the required daisy-chain settle time.

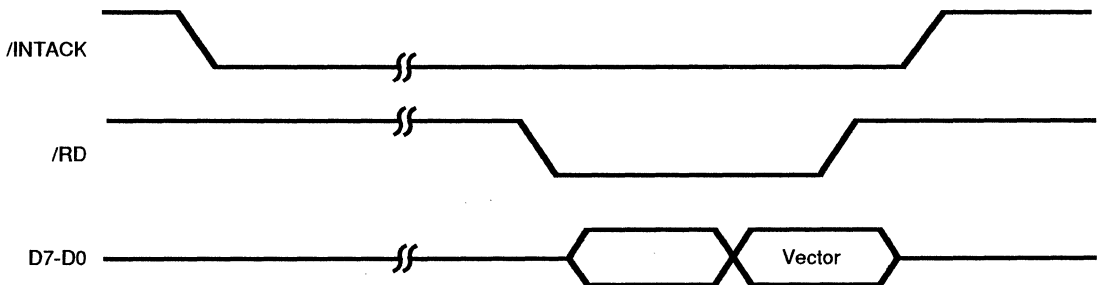


Figure 21. Interrupt Acknowledge Cycle Timing

## Z80C30 Timing

The SCC generates internal control signals from /AS and /DS that are related to PCLK. Since PCLK has no phase relationship with /AS and /DS, the circuitry generating these internal control signals must provide time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK. The recovery time applies only between bus transactions involving the SCC. The recovery time required for proper operation is specified from the falling edge of /DS in the first transaction involving the SCC to the falling edge of /DS in the second transaction involving the SCC.

### Read Cycle Timing

Figure 22 illustrates Read cycle timing. The address on AD7-AD0 and the state of /CS0 and /INTACK are latched by the rising edge of /AS. R/W must be High to indicate a Read cycle. CS1 must also be High for the Read cycle to occur. The data bus drivers in the SCC are then enabled while /DS is Low.

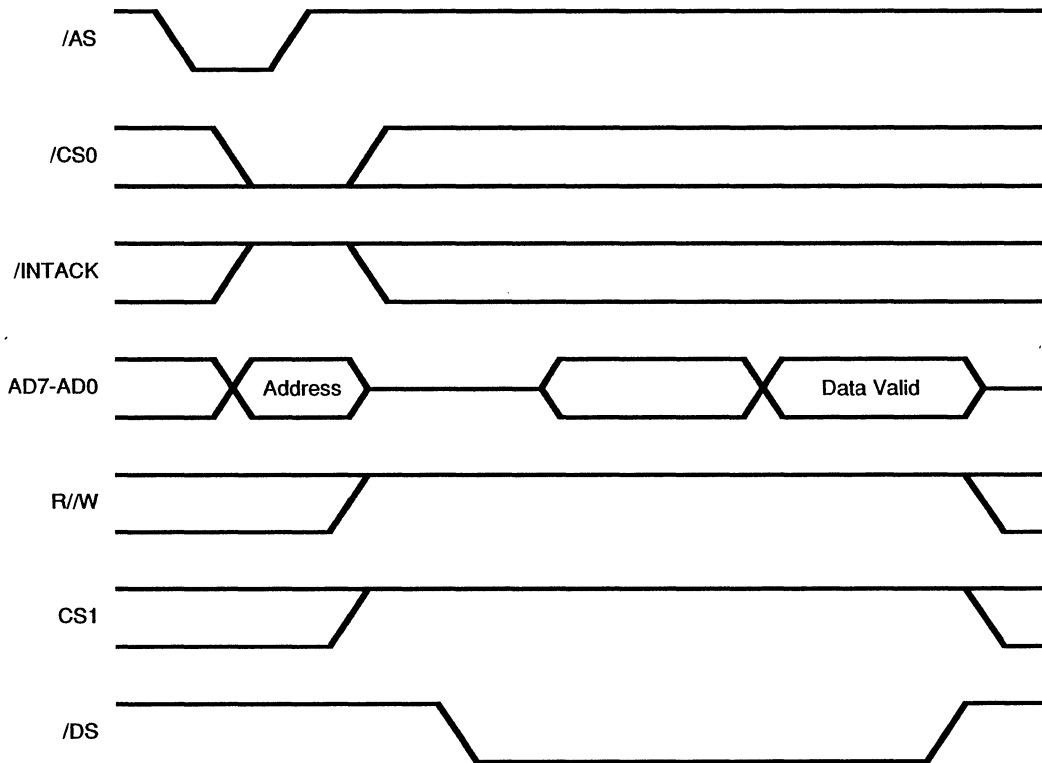


Figure 22. Read Cycle Timing



---

## PROGRAMMING (Continued)

### Write Cycle Timing

Figure 23 illustrates Write cycle timing. The address on AD7-AD0 and the state of /CS0 and /INTACK are latched by the rising edge of /AS. R/W must be Low to indicate a

Write cycle. CS1 must be High for the Write cycle to occur. /DS Low strobes the data into the SCC.

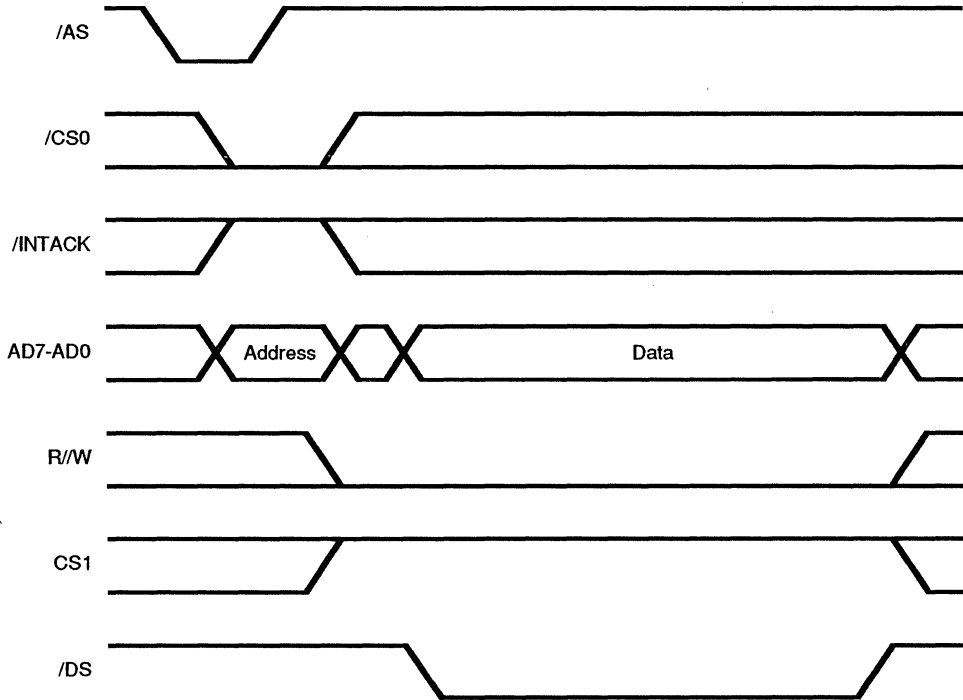


Figure 23. Write Cycle Timing

---

### Interrupt Acknowledge Cycle Timing

Figure 24 illustrates Interrupt Acknowledge cycle timing. The address on AD7-AD0 and the state of /CS0 and /INTACK are latched by the rising edge of /AS. However, if /INTACK is Low, the address and /CS0 are ignored. The state of the R/W and CS1 are also ignored for the duration of the Interrupt Acknowledge cycle. Between the rising edge of /AS and the falling edge of /DS, the internal and

external IEI/IEO daisy chains settle. If there is an interrupt pending in the SCC, and IEI is High when /DS falls, the Acknowledge cycle was intended for the SCC. In this case, the SCC is programmed to respond to RD Low by placing its interrupt vector on D7-D0 and then internally set the appropriate Interrupt-Under-Service latch.

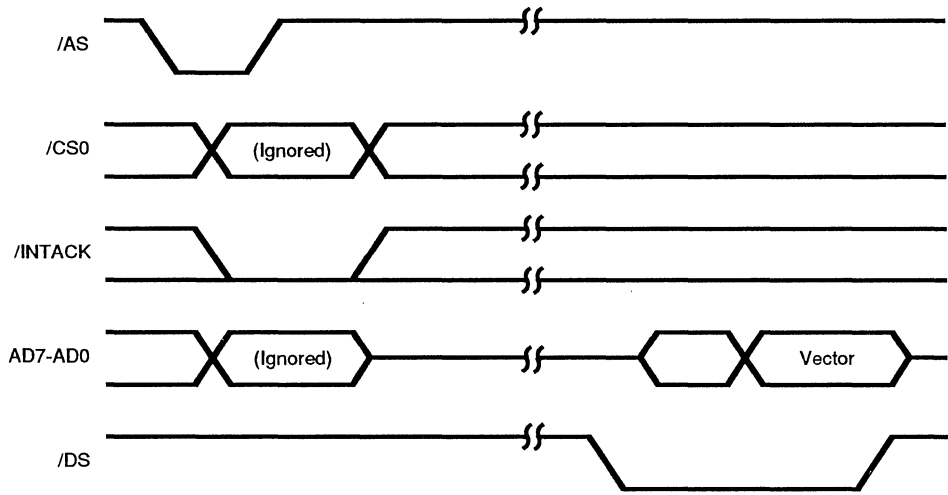


Figure 24. Interrupt Acknowledge Cycle Timing

## ABSOLUTE MAXIMUM RATINGS

$V_{CC}$ Supply Voltage range	-0.3V to +7.0V
Voltages on all pins with respect to GND	-3V to $V_{CC}+0.3V$
$T_A$ Operating Ambient Temperature	See Ordering Information
Storage Temperature	-65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin.

- $+4.50\text{ V} \leq V_{CC} \leq +5.50\text{ V}$
- $\text{GND} = 0\text{ V}$
- $T_A$  as specified in Ordering Information

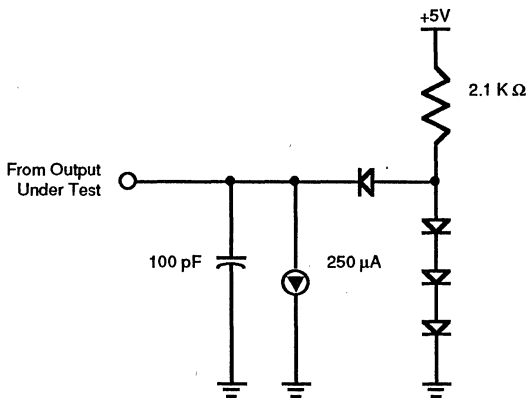


Figure 25. Standard Test Load

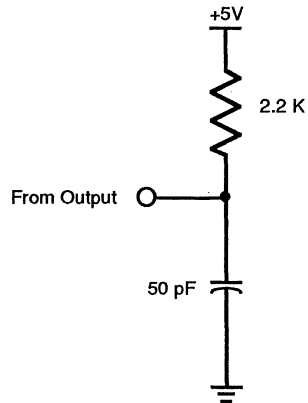


Figure 26. Open-Drain Test Load

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit	Test Condition
$C_{IN}$	Input Capacitance		10	pF	Unmeasured Pins
$C_{OUT}$	Output Capacitance		15	pF	Returned to Ground
$C_{IO}$	Bidirectional Capacitance		20	pF	

### Notes:

f = 1 MHz, over specified temperature range.

Unmeasured pins returned to Ground.

## MISCELLANEOUS

Gate Count 6800

## DC CHARACTERISTICS

Z80C30/Z85C30

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC} + 0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6$ mA
$V_{OH2}$	Output High Voltage	$V_{CC} - 0.8$			V	$I_{OH} = -250$ $\mu$ A
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = +2.0$ mA
$I_{IL}$	Input Leakage			$\pm 10.0$	$\mu$ A	$0.4 V_{IN} + 2.4V$
$I_{OL}$	Output Leakage			$\pm 10.0$	$\mu$ A	$0.4 V_{OUT} + 2.4V$
$I_{CC1}$	$V_{CC}$ Supply Current [2]		7	12(10 MHz)	mA	$V_{CC} = 5V$ $V_{IH} = 4.8$ $V_{IL} = 0$
			9	15(16.384 MHz)	mA	Crystal Oscillator off
$I_{CCOSC}$	Crystal OSC Current [3]		4		mA	Current for each OSC in addition to $I_{CC1}$

### Notes:

[1]  $V_{CC} = 5V \pm 10\%$  unless otherwise specified, over specified temperature range.

[2] Typical  $I_{CC}$  was measured with oscillator off.

[3] No  $I_{CC}$  (OSC) max is specified due to dependency on external circuit and frequency of oscillation.

**AC CHARACTERISTICS**  
**Z85C30 Read/Write Timing Diagrams**

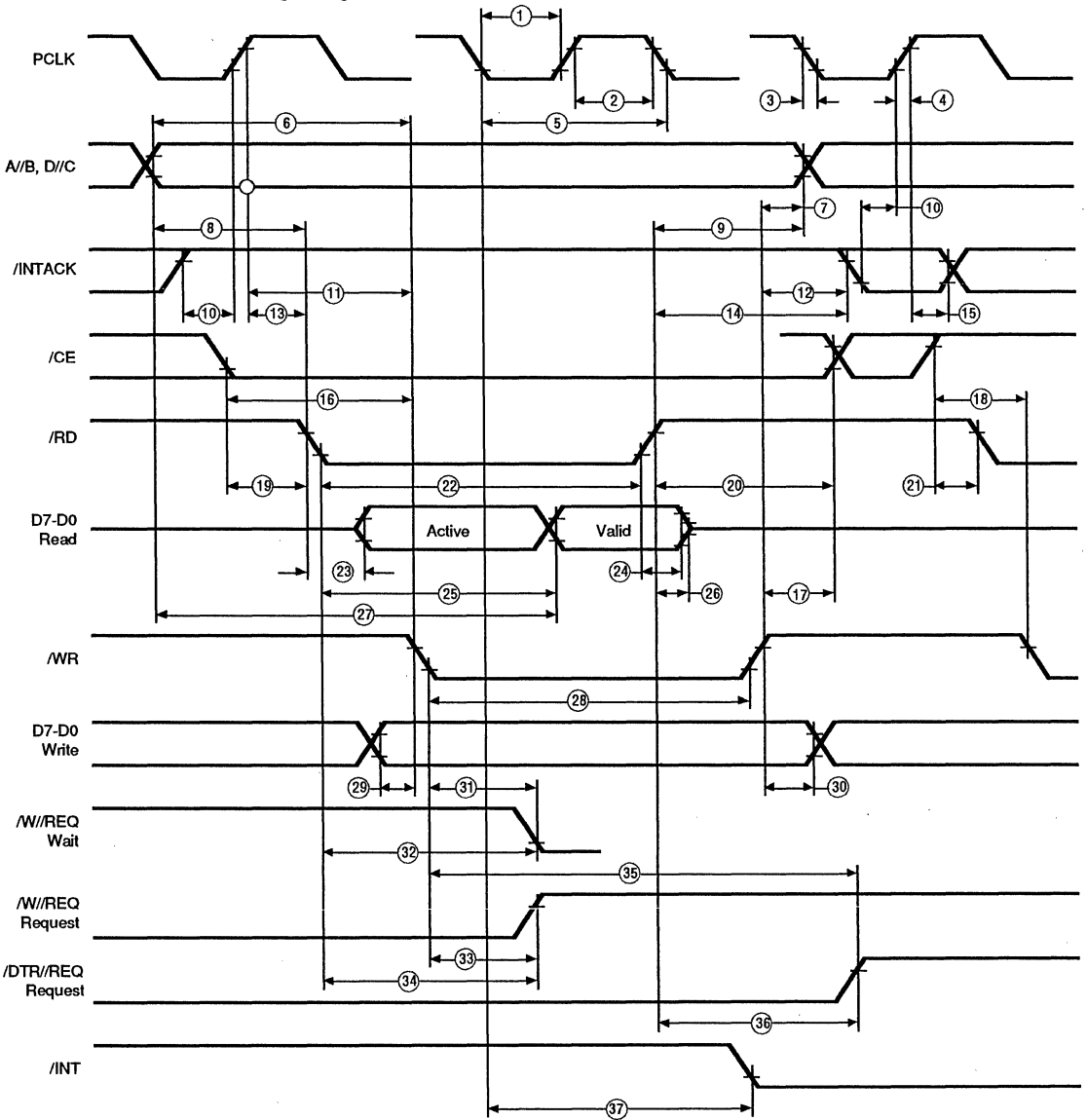


Figure 27. Z85C30 Read/Write Timing Diagram

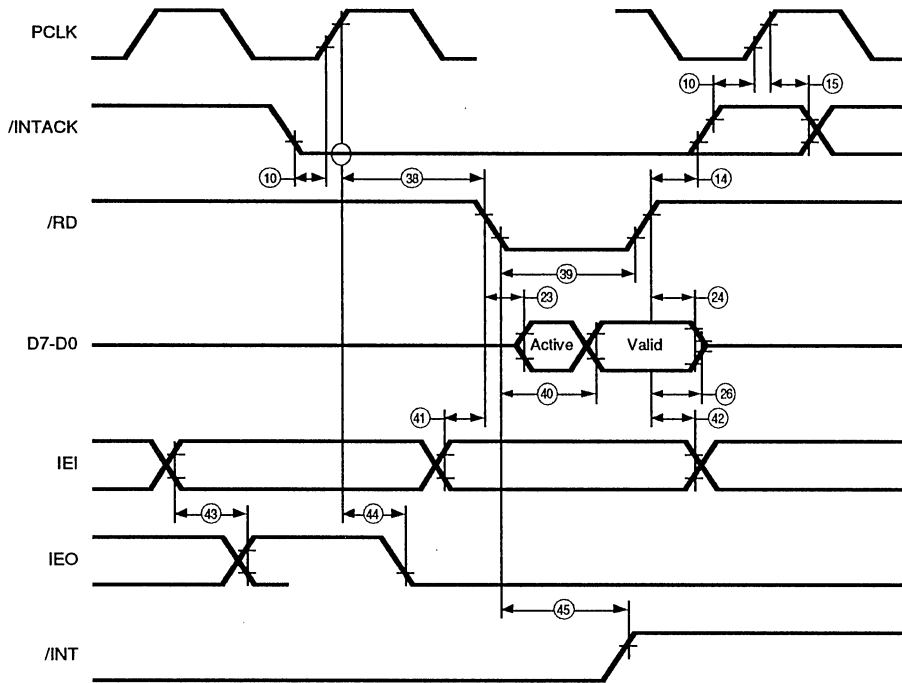


Figure 28. Z85C30 Interrupt Acknowledge Timing Diagram

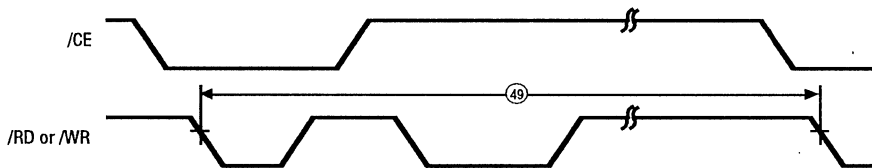


Figure 29. Z85C30 Cycle Timing Diagram

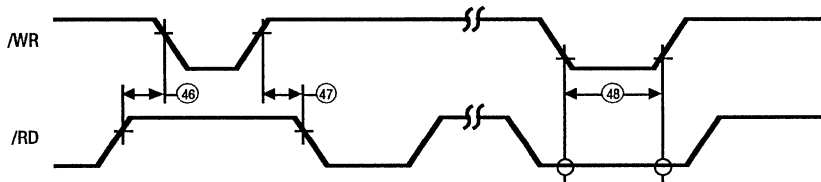


Figure 30. Z85C30 Reset Timing Diagram

## AC CHARACTERISTICS

Z85C30 Read/Write Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16.384 MHz		Notes*
			Min	Max	Min	Max	Min	Max	
1	TwPCI	PCLK Low Width	45	1000	40	1000	26	1000	
2	TwPCh	PCLK High Width	45	1000	40	1000	26	1000	
3	TfPC	PCLK Fall Time		10		10		5	
4	TrPC	PCLK Rise Time		10		10		5	
5	TcPC	PCLK Cycle Time	118	2000	100	2000	61	2000	
6	TsA(WR)	Address to /WR Fall Setup Time	66		50		35		
7	ThA(WR)	Address to /WR Rise Hold Time	0		0		0		
8	TsA(RD)	Address to /RD Fall Setup Time	66		50		35		
9	ThA(RD)	Address to /RD Rise Hold Time	0		0		0		
10	TsIA(PC)	/INTACK to /PCLK Rise Setup Time	20		20		15		
11	TsIAi(WR)	/INTACK to /WR Fall Setup Time	140		130		75		[1]
12	ThIA(WR)	/INTACK to /WR Rise Hold Time	0		0		0		
13	TsIAi(RD)	/INTACK to /RD Fall Setup Time	140		130		75		[1]
14	ThIA(RD)	/INTACK to /RD Rise Hold Time	0		0		0		
15	ThIA(PC)	/INTACK to /PCLK Rise Hold Time	38		30		15		
16	TsCEI(WR)	/CE Low to /WR Fall Setup Time	0		0		0		
17	ThCE(WR)	/CE to /WR Rise Hold Time	0		0		0		
18	TsCEh(WR)	/CE High to /WR Fall Setup Time	58		50		30		
19	TsCEI(RD)	/CE Low to /RD Fall Setup Time	0		0		0		[1]
20	ThCE(RD)	/CE to /RD Rise Hold Time	0		0		0		[1]
21	TsCEh(RD)	/CE High to /RD Fall Setup Time	58		50		30		[1]
22	TwRDI	/RD Low Width	145		125		70		[1]
23	TdRD(DRA)	/RD Fall to Read Data Active Delay	0		0		0		
24	TdRDr(DR)	/RD Rise to Read Data Not Valid Delay	0		0		0		
25	TdRDI(DR)	/RD Fall to Read Data Valid Delay		135		120		65	
26	TdRD(DRz)	/RD Rise to Read Data Float Delay		38		35		20	
27	TdA(DR)	Address to Read Data Valid Delay		210		180		100	
28	TwWRI	/WR Low Width	145		125		70		
29	TsDW(WR)	Write Data to /WR Fall Setup Time	10		10		10		
30	ThDW(WR)	/Write Data to /WR Rise Hold Time	0		0		0		
31	TdWR(W)	/WR Fall to Wait Valid Delay		168		160		80	[4]
32	TdRD(W)	/RD Low to Wait Valid Delay		168		160		80	[4]
33	TdWRf(REQ)	/WR Fall to /W//REQ Not Valid Delay		168		160		80	
34	TdRDf(REQ)	/RD Fall to /W//REQ Not Valid Delay		168		160		80	
35	TdWRr(REQ)	/WR Fall to /DTR//REQ Not Valid		4TcPc		4TcPc		4TcPc	
36	TdRDr(REQ)	/RD Rise to /DTR//REQ Not Valid		NA		NA		NA	
37	TdPC(INT)	/PCLK Fall to /INT Valid Delay		500		450		175	
38	TdIAi(RD)	/INTACK to /RD Fall (Ack) Delay	145		125		75		[5]
39	TwRDA	/RD (Acknowledge) Width	145		125		70		
40	TdRDA(DR)	/RD Fall (Ack) to Read Data Valid		135		120		70	

## AC CHARACTERISTICS

### Z85C30 Read/Write Timing Table (Continued)

No	Symbol	Parameter	8.5 MHz		10 MHz		16.384 MHz		Notes*
			Min	Max	Min	Max	Min	Max	
41	TsIEI(RDA)	IEI to /RD Fall (Ack) Setup Time	95		95		50		
42	ThIEI(RDA)	IEI to /RD Rise (Ack) Hold Time	0		0		0		
43	TdIEI(IEO)	IEI to IEO Delay Time		95		90		50	
44	TdPC(IEO)	/PCLK Rise to IEO Delay		195		175		80	
45	TdRDA(INT)	/RD Fall to /INT Inactive Delay		480		320		200	[4]
46	TdRD(WRQ)	/RD Rise to /WR Fall for No Reset	15		15		10		
47	TdWRQ(RD)	/WR Rise to /RD Fall for No Reset	15		15		10		
48	TwRES	/WR & /RD Low for Reset	145		100		75		
49	Trc	Valid Access Recovery Time	4TcPc		4TcPc		4TcPc		[3]

#### Notes:

[1] Parameter does not apply to Interrupt Acknowledge transactions.

[3] Parameter applies only between transactions involving the SCC.

[4] Open-drain output, measured with open-drain test load.

[5] Parameter is system dependent. For any SCC in the daisy chain, TdIAi(RD) must be greater than the sum of TdPC(IEO) for the highest priority device in the daisy chain, TsIEI(RDA) for the SCC, and TdIEI(IEO) for each device separating them in the daisy chain.

\* Units in Nanoseconds (ns), otherwise noted.



# AC CHARACTERISTICS

## Z85C30 General Timing Diagram

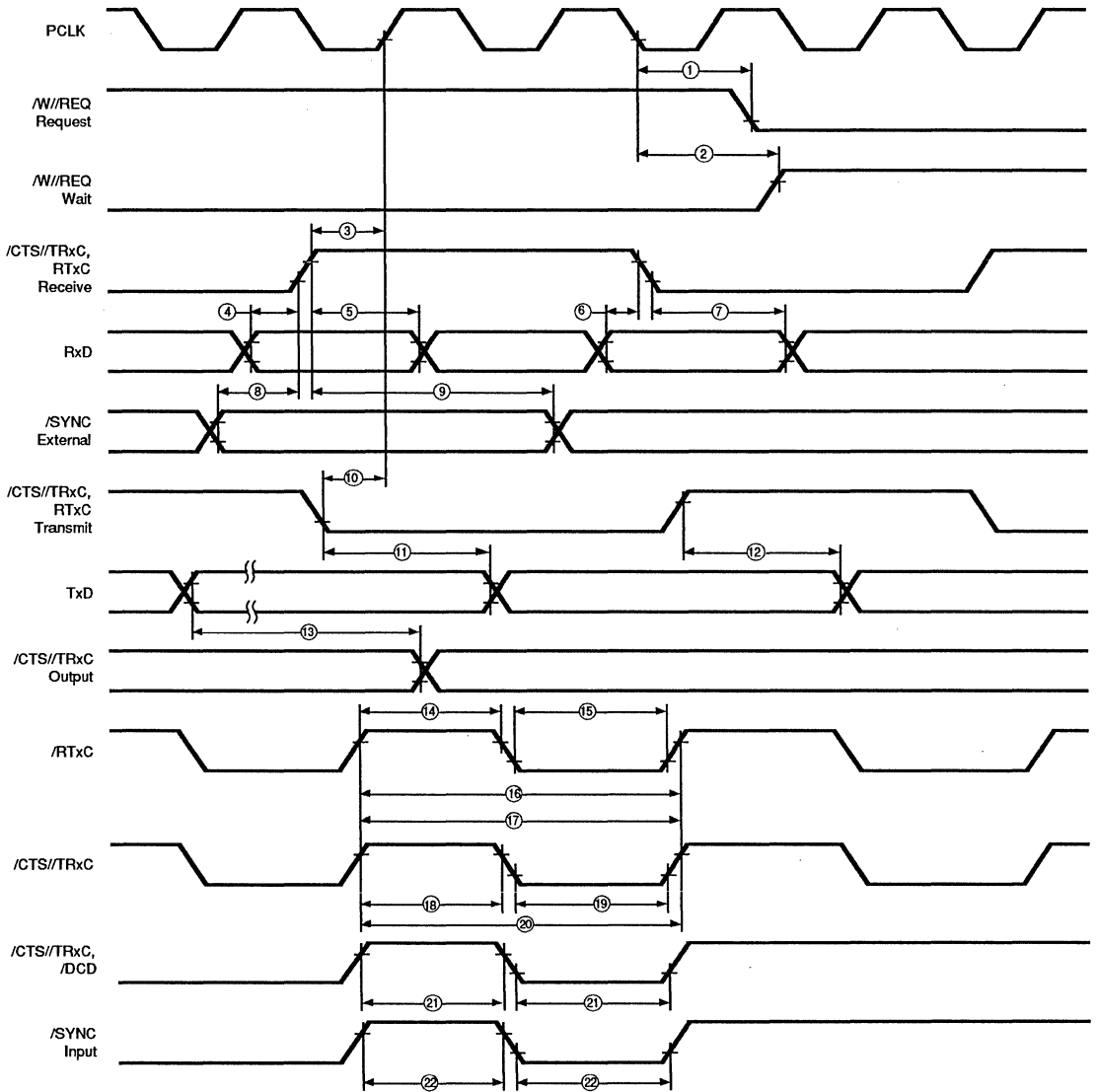


Figure 34. Z85C30 General Timing Diagram

## AC CHARACTERISTICS

### Z85C30 General Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16.384 MHz		Notes*
			Min	Max	Min	Max	Min	Max	
1	TdPC(REQ)	/PCLK Low to W/REQ Valid		250		200		110	
2	TsPC(W)	/PCLK Low to Wait Inactive		350		300		180	
3	TsRXC(PC)	/RxC High to /PCLK High Setup Time	NA	NA	NA	NA	NA	NA	[1,4]
4	TsRXD(RXCr)	RxD to /RxC High Setup Time	0		0		0		[1]
5	ThRXD(RxCr)	RxD to /RxC High Hold Time	150		125		60		[1]
6	TsRXD(RXCf)	RxD to /RxC Low Setup Time	0		0		0		[1,5]
7	ThRXD(RXCf)	RxD to /RxC Low Hold Time	150		125		60		[1,5]
8	TsSY(RXC)	SYNC to /RxC High Setup Time	-200		-150		-100		[1]
9	ThSY(RXC)	SYNC to /RxC High Hold Time	5TcPc		5TcPc		5TcPc		[1]
10	TsTXC(PC)	/TxC Low to /PCLK High Setup Time	NA		NA		NA		[2,4]
11	TdTXCf(TXD)	/TxC Low to TxD Delay		190		150		85	[2]
12	TdTxCr(TXD)	/TxC High to TxD Delay		190		150		85	[2,5]
13	TdTXD(TRX)	TxD to TRxC Delay		200		140		80	
14	TwRTXh	RTxC High Width	130		120		80		[6]
15	TwRTXI	TRxC Low Width	130		120		80		[6]
16a	TcRTX	RTxC Cycle Time	472		400		244		[6,7]
16b	TxRX (DPLL)	DPLL Cycle Time Min	59		50		31		[7,8]
17	TcRTXX	Crystal Osc. Period	118	1000	100	1000	100	1000	[3]
18	TwTRXh	TRxC High Width	130		120		80		[6]
19	TwTRXI	TRxC Low Width	130		120		80		[6]
20	TcTRX	TRxC Cycle Time	472		400		244		[6,7]
21	TwEXT	DCD or CTS Pulse Width	200		120		70		
22	TwSY	SYNC Pulse Width	200		120		70		

#### Notes:

- [1] RxC is /RTxC or /TRxC, whichever is supplying the receive clock.
- [2] TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.
- [3] Both /RTxC and /SYNC have 30 pf capacitors to ground connected to them.
- [4] Synchronization of RxC to PCLK is eliminated in divide by four operation.
- [5] Parameter applies only to FM encoding/decoding.
- [6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator timing requirements are identical to case PCLK requirements.
- [7] The maximum receive or transmit data rate is 1/4 PCLK.
- [8] Applies to DPLL clock source only. Maximum data rate of 1/4 PCLK still applies. DPLL clock should have a 50% duty cycle.

\* Units in nanoseconds (ns) otherwise noted.

# AC CHARACTERISTICS

## Z85C30 System Timing Diagram

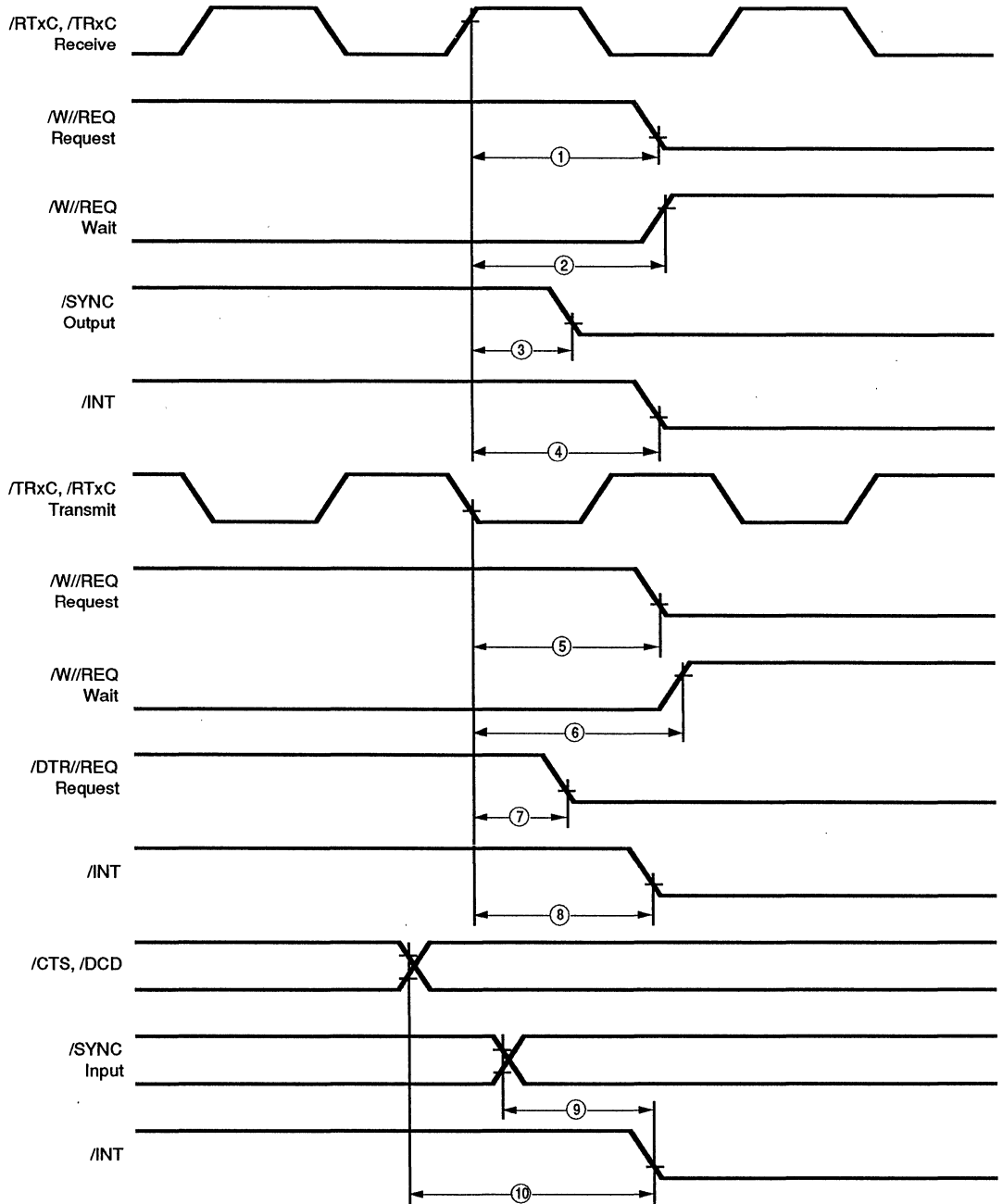


Figure 35. Z85C30 System Timing Diagram

## AC CHARACTERISTICS

### Z85C30 System Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16.384 MHz		Notes*
			Min	Max	Min	Max	Min	Max	
1	TdRXC(REQ)	/RxC High to /W//REQ Valid	8	12	8	12	8	12	[2,5]
2	TdRXC(W)	/RxC High to Wait Inactive	8	14	8	14	8	14	[1,2,5]
3	TdRdXC(SY)	/RxC High to /SYNC Valid	4	7	4	7	4	7	[2,5]
4	TsRXC(INT)	/RxC High to INT Valid	10	16	10	16	10	16	[1,2,5]
5	TdTXC(REQ)	/TxC Low to /W//REQ Valid	5	8	5	8	5	8	[3,5]
6	TdTXC(W)	/TxC Low to Wait Inactive	5	11	5	11	5	11	[1,3,5]
7	TdTXC(DRQ)	/TxC Low to /DTR//REQ Valid	4	7	4	7	4	7	[3,5]
8	TdTXC(INT)	/TxC Low to /INT Valid	6	10	6	10	6	10	[1,3,5]
9a	TdSY(INT)	SYNC to INT Valid	2	6	2	6	2	6	[1,5]
9b	TdSY(INT)	SYNC to INT Valid	2	3	2	3	2	3	[1,4,5]
10	TdEXT(INT)	/DCD or /CTS to /INT Valid	2	6	2	6	2	6	[1,5]

#### Notes:

[1] Open drain-output, measured with open-drain test load.

[2] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[3] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[4] Units equal to /AS.

[5] Units equal to TcPc.

# AC CHARACTERISTICS

## Z80C30 Read and Write Timing Diagrams

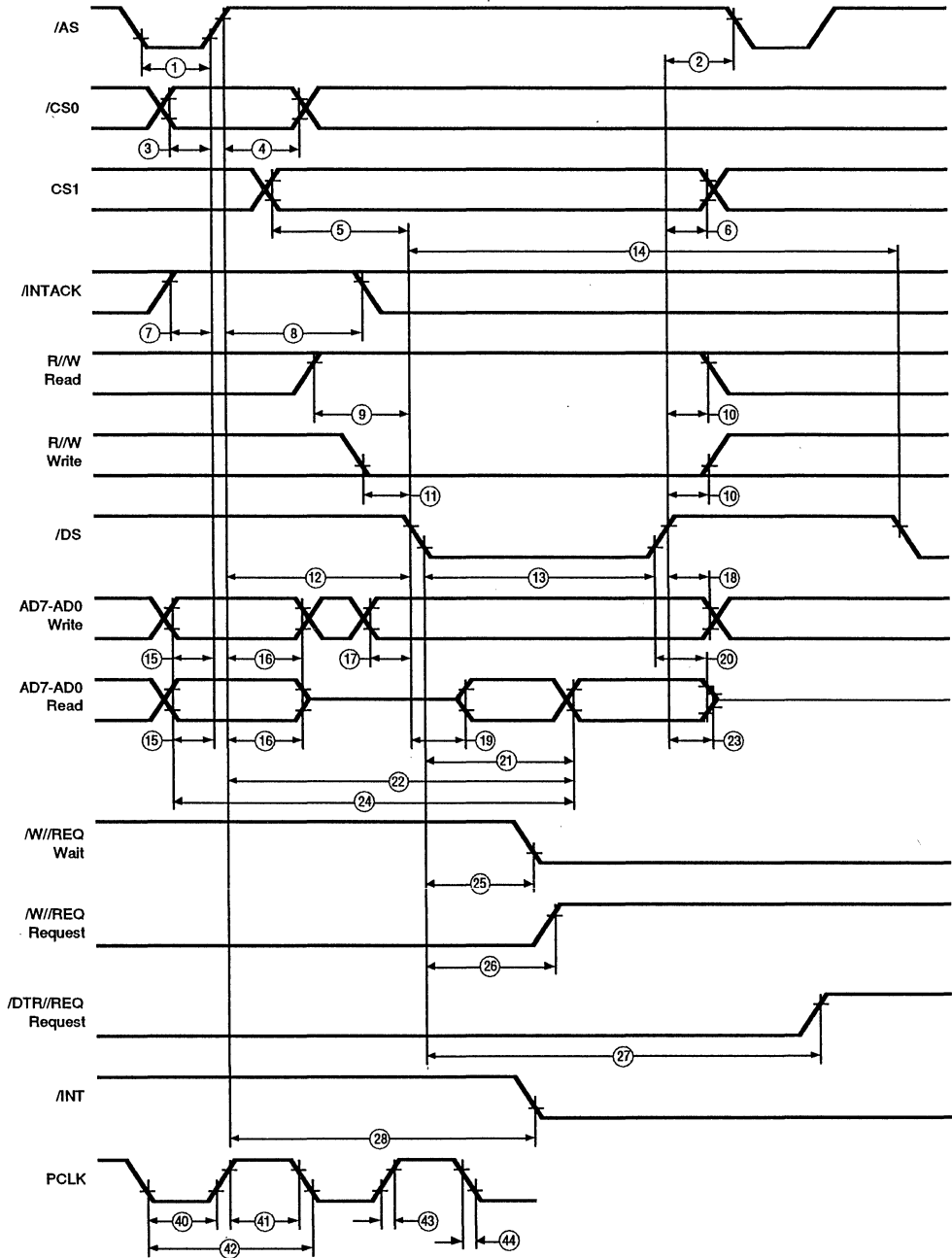


Figure 36. Z80C30 Read/Write Timing Diagram

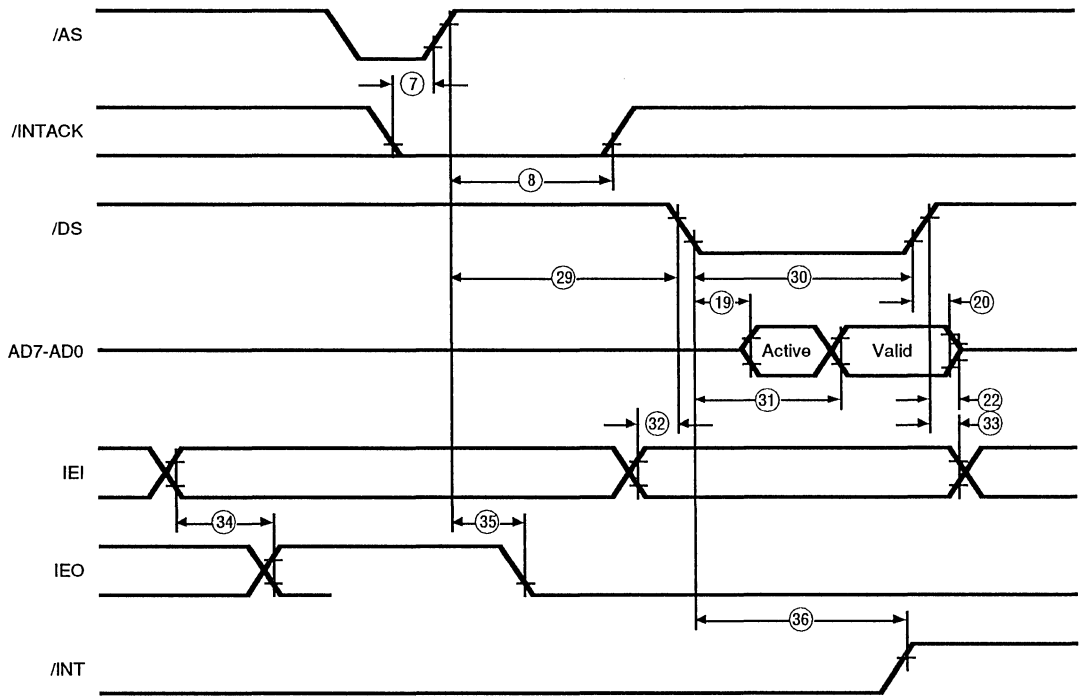


Figure 37. Z80C30 Interrupt Acknowledge Timing Diagram

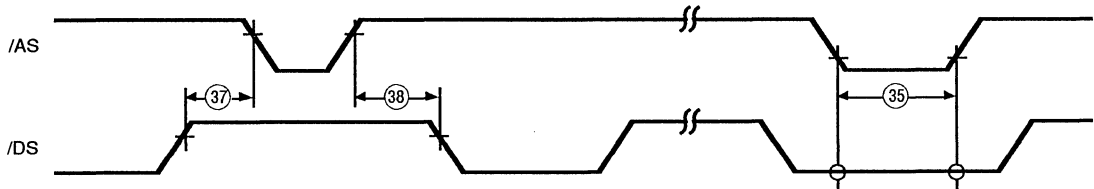


Figure 38. Z80C30 Reset Timing Diagram

## AC CHARACTERISTICS

Z80C30 Read/Write Timing Table

No	Symbol	Parameter	8 MHz		10 MHz		Notes *
			Min	Max	Min	Max	
1	TwAS	/AS Low Width	35		30		
2	TdDS(AS)	/DS Rise to /AS Fall Delay	15		10		[1]
3	TsCSO(AS)	/CS0 to /AS Rise Setup Time	0		0		[1]
4	ThCSO(AS)	/CS0 to /AS Rise Hold Time	30		20		[1]
5	TsCS1(DS)	CS1 to /DS Fall Setup Time	65		50		[1]
6	ThCS1(DS)	CS1 to /DS Rise Hold Time	30		20		[1]
7	TsIA(AS)	/INTACK to /AS Rise Setup Time	10		10		
8	ThIA(AS)	/INTACK to /AS Rise Hold Time	150		125		
9	TsRWR(DS)	R//W (Read) to /DS Fall Setup Time	65		50		
10	ThRW(DS)	R//W to /DS Rise Hold Time	0		0		
11	TsRWW(DS)	R//W (Write) to /DS Fall Setup Time	0		0		
12	TdAS(DS)	/AS Rise to /DS Fall Delay	30		20		
13	TwDSI	/DS Low Width	150		125		
14	TrC	Valid Access Recovery Time	4TcPC		4TcPC		[2]
15	TsA(AS)	Address to /AS Rise Setup Time	10		10		[1]
16	ThA(AS)	Address to /AS Rise Hold Time	25		20		[1]
17	TsDW(DS)	Write Data to /DS Fall Setup Time	15		10		
18	ThDW(DS)	Write Data to /DS Rise Hold Time	0		0		
19	TdDS(DA)	/DS Fall to Data Active Delay	0		0		
20	TdDSr(DR)	/DS Rise to Read Data Not Valid Delay	0		0		
21	TdDSf(DR)	/DS Fall to Read Data Valid Delay		140		120	
22	TdAS(DR)	/AS Rise to Read Data Valid Delay		250		190	
23	TdDS(DRz)	/DS Rise to Read Data Float Delay		40		35	[3]
24	TdA(DR)	Address Required Valid to Read Data Valid Delay		260		210	
25	TdDS(W)	/DS Fall to Wait Valid Delay		170		160	[4]
26	TdDSf(REQ)	/DS Fall to /W//REQ Not Valid Delay		170		160	
27	TdDSr(REQ)	/DS Fall to /DTR//REQ Not Valid Delay		4TcPC		4TcPC	
28	TdAS(INT)	/AS Rise to /INT Valid Delay		500		500	[4]
29	TdAS(DSA)	/AS Rise to /DS Fall (Acknowledge) Delay	250		225		[5]
30	TwDSA	/DS (Acknowledge) Low Width	150		125		
31	TdDSA(DR)	/DS Fall (Acknowledge) to Read Data Valid Delay		140		120	
32	TsIEI(DSA)	IEI to /DS Fall (Acknowledge) Setup Time	80		80		
33	ThIEI(DSA)	IEI to /DS Rise (Acknowledge) Hold Time	0		0		
34	TdIEI(IEO)	IEI to IEO Delay		90		90	
35	TdAS(IEO)	/AS Rise to IEO Delay		200		175	[6]
36	TdDSA(INT)	/DS Fall (Acknowledge) to /INT Inactive Delay		450		450	[4]
37	TdDS(ASQ)	/DS Rise to /AS Fall Delay for No Reset	15		15		
38	TdASQ(DS)	/AS Rise to /DS Fall Delay for No Reset	20		15		
39	TwRES	/AS and /DS Coincident Low for Reset	150		100		[7]
40	TwPCI	PCLK Low Width	50	1000	40	1000	

## AC CHARACTERISTICS

### Z80C30 Read/Write Timing Table (Continued)

No	Symbol	Parameter	8 MHz		10 MHz		Notes *
			Min	Max	Min	Max	
41	TwPCh	PCLK High Width	50	1000	40	1000	
42	TcPC	PCLK Cycle Time	125	2000	100	2000	
43	TrPC	PCLK Rise Time		10		10	
44	TfPC	PCLK Fall Time		10		10	

#### Notes:

[1] Parameter does not apply to interrupt Acknowledge transactions.

[2] Parameter applies only between transactions involving the SCC.

[3] Float delay is defined as the time required for a  $\pm 0.5V$  change in the output with a maximum DC load and a minimum AC load.

[4] Open-drain output, measured with open-drain test load.

[5] Parameter is system dependent. For any Z-SCC in the daisy chain, TdAS(DSA) must be greater than the sum of TdAS(IEO) for the highest priority device in the daisy chain, TsIEI(DSA) for the Z-SCC, and TdIEI(IEO) for each device separating them in the daisy chain.

[6] Parameter applies only to a Z-SCC pulling INT Low at the beginning of the Interrupt Acknowledge transaction.

[7] Internal circuitry allows for the reset provided by the Z8 to be recognized as a reset by the Z-SCC. All timing references assume 2.0V for a logic \*1\* and 0.8V for a logic \*0\*.

\* Units in nanoseconds(ns) otherwise noted.



**AC CHARACTERISTICS**  
**Z80C30 General Timing Diagram**

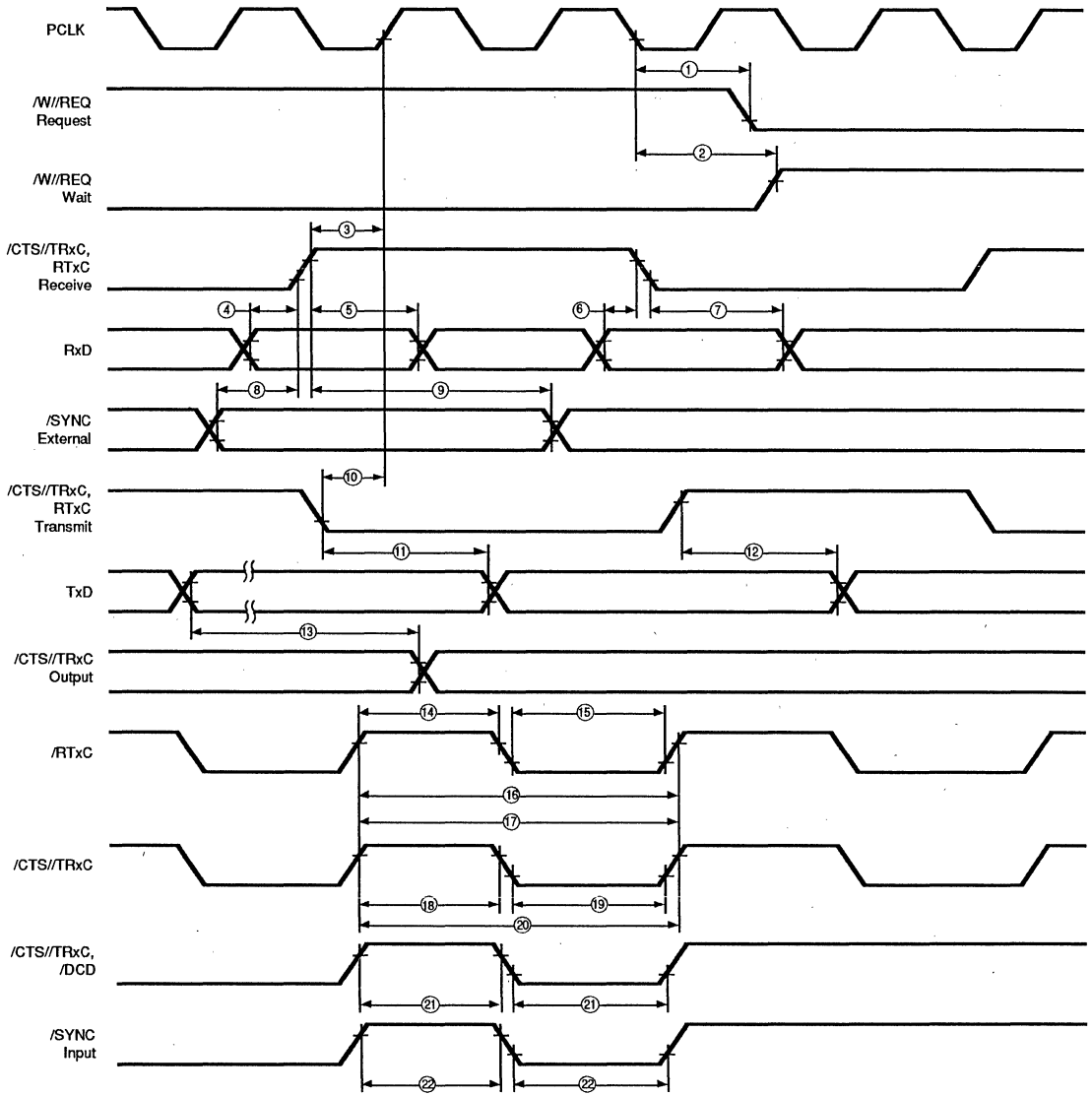


Figure 39. Z80C30 General Timing Diagram

## AC CHARACTERISTICS

### Z80C30 General Timing Table

No	Symbol	Parameter	8 MHz		10 MHz		Notes*
			Min	Max	Min	Max	
1	TdPC(REQ)	/PCLK Low to W/REQ Valid		250		200	
2	TsPC(W)	/PCLK Low to Wait Inactive		350		300	
3	TsRXC(PC)	/RxC High to /PCLK High Setup Time	NA	NA	NA	NA	[1,4]
4	TsRXD(RXCr)	RxD to /RxC High Setup Time	0		0		
5	ThRXD(RxCr)	RxD to /RxC High Hold Time	150		125		[1]
6	TsRXD(RXCf)	RxD to /RxC Low Setup Time	0		0		[1,5]
7	ThRXD(RXCf)	RxD to /RxC Low Hold Time	150		125		[1,5]
8	TsSY(RXC)	SYNC to /RxC High Setup Time	-200		-150		[1]
9	ThSY(RXC)	SYNC to /RxC High Hold Time	5TcPc		5TcPc		[1]
10	TsTXC(PC)	/TxC Low to /PCLK High Setup Time	NA		NA		[2,4]
11	TdTXCf(TXD)	/TxC Low to TxD Delay		190		150	[2]
12	TdTxCr(TXD)	/TxC High to TxD Delay		190		150	[2,5]
13	TdTXD(TRX)	TxD to TRxC Delay		200		140	
14	TwRTXh	TRxC High Width	130		120		[6]
15	TwRTXI	TRxC Low Width	130		120		[6]
16a	TcRTX	TRxC Cycle Time	472		400		[6,7]
16b	TxRX (DPLL)	DPLL Cycle Time Min	59		50		[7,8]
17	TcRTXX	Crystal Osc. Period	118	1000	100	1000	[3]
18	TwTRXh	TRxC High Width	130		120		[6]
19	TwTRXI	TRxC Low Width	130		120		[6]
20	TcTRX	TRxC Cycle Time	472		400	[6,7]	
21	TwEXT	DCD or CTS Pulse Width	200		120		
22	TwSY	SYNC Pulse Width	200		120		

#### Notes:

- [1] RxC is /RTxC or /TRxC, whichever is supplying the receive clock.
- [2] TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.
- [3] Both /RTxC and /SYNC have 30 pf capacitors to ground connected to them.
- [4] Synchronization of RxC to PCLK is eliminated in divide by four operation.
- [5] Parameter applies only to FM encoding/decoding.
- [6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator timing requirements are identical to case PCLK requirements.
- [7] The maximum receive or transmit data rate is 1/4 PCLK.
- [8] Applies to DPLL clock source only. Maximum data rate of 1/4 PCLK still applies. DPLL clock should have a 50% duty cycle.

\* Units in nanoseconds (ns) otherwise noted.

**AC CHARACTERISTICS**  
**Z80C30 System Timing Diagram**

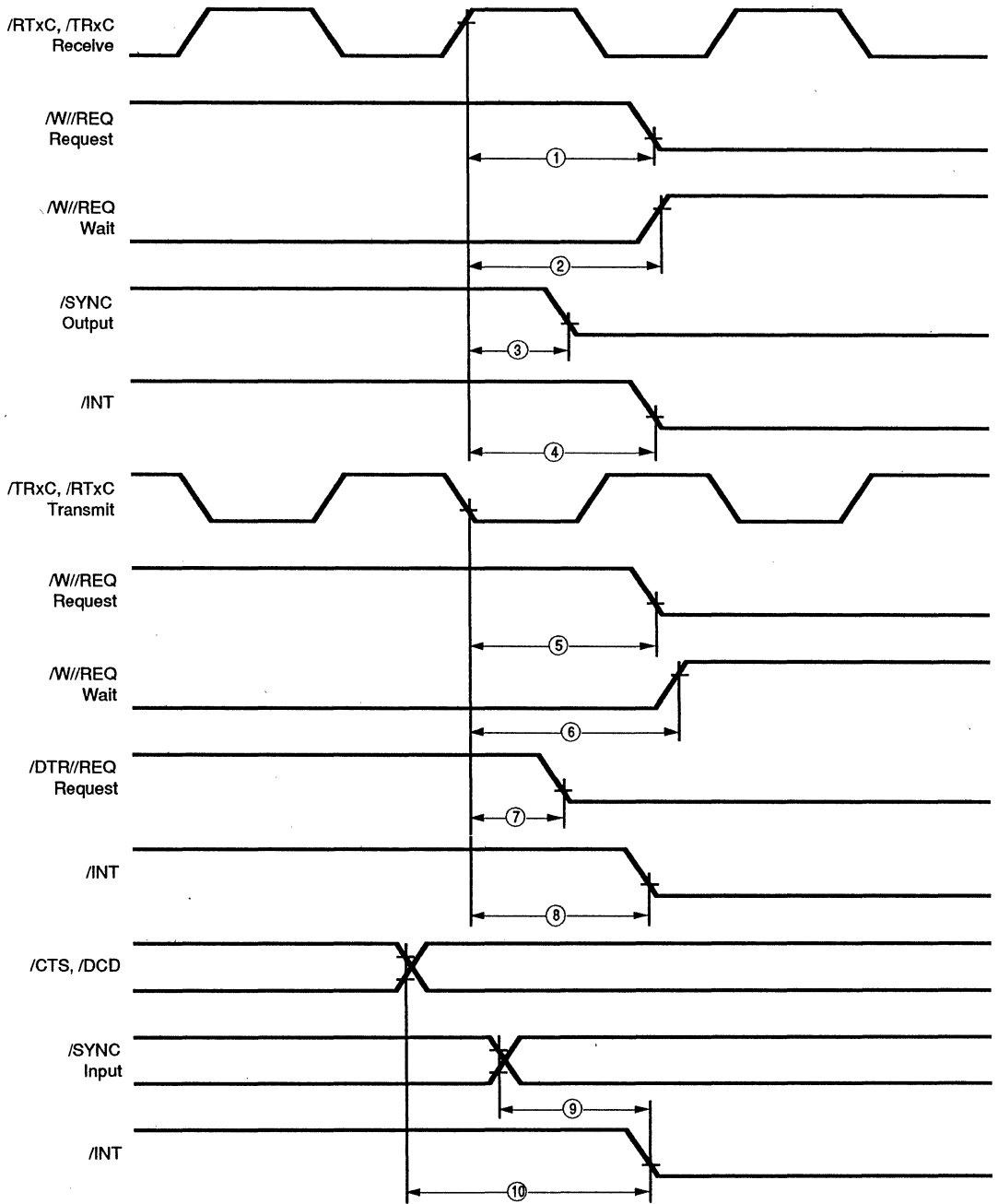


Figure 40. Z80C30 System Timing Diagram

## AC CHARACTERISTICS

### Z80C30 System Timing Table

No	Symbol	Parameter	8 MHz		10 MHz		Notes
			Min	Max	Min	Max	
1	TdRXC(REQ)	/RxC High to /W//REQ Valid	8	12	8	12	[2,5]
2	TdRXC(W)	/RxC High to Wait Inactive	8	14	8	14	[1,2,5]
3	TdRdXC(SY)	/RxC High to /SYNC Valid	4	7	4	7	[2,5]
4	TdRXC(INT)	/RxC High to INT Valid	8	12	8	12	[1,2,5]
			2	3	2	3	[4,5]
5	TdTXC(REQ)	/TxC Low to /W//REQ Valid	5	8	5	8	[3,5]
6	TdTXC(W)	/TxC Low to Wait Inactive	5	11	5	11	[1,3,5]
7	TdTXC(DRQ)	/TxC Low to /DTR//REQ Valid	4	7	4	7	[3,5]
8	TdTXC(INT)	/TxC Low to /INT Valid	4	6	4	6	[1,3,5]
			2	3	2	3	[4,5]
9a	TdSY(INT)	SYNC to INT Valid	2	6	2	6	[1,5]
9b	TdSY(INT)	SYNC to INT Valid	2	3	2	3	[1,4,5]
10	TdEXT(INT)		2	3	2	3	[1,4,5]

#### Notes:

[1] Open drain-output, measured with open-drain test load.

[2] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[3] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[4] Units equal to /AS.

[5] Units equal to TcPc.





## Z85230

### ESCC™ ENHANCED SERIAL COMMUNICATION CONTROLLER

#### FEATURES

- Deeper Data FIFOs
  - 4-byte transmit FIFO
  - 8-byte receive FIFO
- Programmable FIFO interrupt levels provide flexible interrupt response
- Pin and Function compatible to CMOS and NMOS Z85C30 SCC
- Many improvements to support SDLC/HDLC transfers:
  - Deactivation of /RTS pin after closing flag
  - Automatic transmission of the opening flag
  - Automatic reset of Tx Underrun/EOM latch
  - Complete CRC reception
  - TxD pin automatically forced high with NRZI encoding when using mark idle.
  - Receive FIFO automatically unlocked for special receive interrupts when using the SDLC status FIFO.
  - Back-to-back frame transmission simplified
- Easier interface to popular CPUs
- Fast speeds:
  - 8.0 MHz for data rates up to 2.0 Mbit/sec.
  - 10.0 MHz for data rates up to 2.5 Mbit/sec.
  - 16.384 MHz for data rates up to 4.096 Mbit/sec.
  - 20.0 MHz for data rates up to 5.0 Mbit/sec.
- Improved SDLC frame status FIFO
- Low Power CMOS
- New programmable features added with Write Register 7'
- Write registers: WR3, WR4, WR5, and WR10 are now readable
- Read Register 0 latched during access
- Software Interrupt Acknowledge Mode
- DPLL counter output available as jitter-free clock source
- /DTR//REQ pin deactivation time reduced
- Two independent full-duplex channels, each with a crystal oscillator, baud rate generator, and digital phase-locked loop.
- Multi-protocol operation under program control
- Asynchronous mode with five to eight bits, and one, one and one-half, or two stop bits per character; programmable clock factor; break detection and generation; parity, overrun, and framing error detection.
- Synchronous mode with internal or external character synchronization on one or two synchronous characters and CRC generation and checking with programmable CRC preset values.

#### GENERAL DESCRIPTION

The Zilog Enhanced Serial Communications Controller, Z85230 ESCC, is a pin and software compatible CMOS member of the SCC™ Family (The SCC was introduced by Zilog in 1981.). The ESCC is a dual-channel, full-duplex data communications controller capable of supporting a wide range of popular protocols. The ESCC is built from Zilog's industry standard SCC core and is compatible with

designs using Zilog's SCC to receive and transmit data. It has many improvements that significantly reduce CPU overhead. The addition of a 4-byte transmit FIFO and an 8-byte receive FIFO significantly reduces the overhead required to provide data to, and get data from, the transmitters and receivers.

---

## GENERAL DESCRIPTION (Continued)

The ESCC also has many features that improve packet handling in SDLC mode. The ESCC will automatically: transmit a flag before the data, reset the Tx Underrun/EOM latch, force the TxD pin high at the appropriate time when using NRZI encoding, deassert the /RTS pin after the closing flag, and better handle ABORTed frames when using the 10x19 status FIFO. The combination of these features along with the deeper data FIFOs significantly simplifies SDLC driver software.

The CPU hardware interface has been simplified by relieving the databus setup time requirement and supporting the software generation of the interrupt acknowledge signal (/INTACK). These changes allow an interface with less external logic to many microprocessor families while maintaining compatibility with existing designs. I/O handling of the ESCC is improved over the SCC with faster response of the /INT and /DTR//REQ pins.

The many enhancements added to the ESCC permits a system design that increases overall system performance with better data handling and less interface logic (Figure 1).

### Notes:

All Signals with a preceding front slash, "/", are active Low, e.g.: B//W (WORD is active Low); /B//W (BYTE is active Low, only).

Power connections follow conventional descriptions below:

Connection	Circuit	Device
Power Ground	V <sub>CC</sub> GND	V <sub>DD</sub> V <sub>SS</sub>

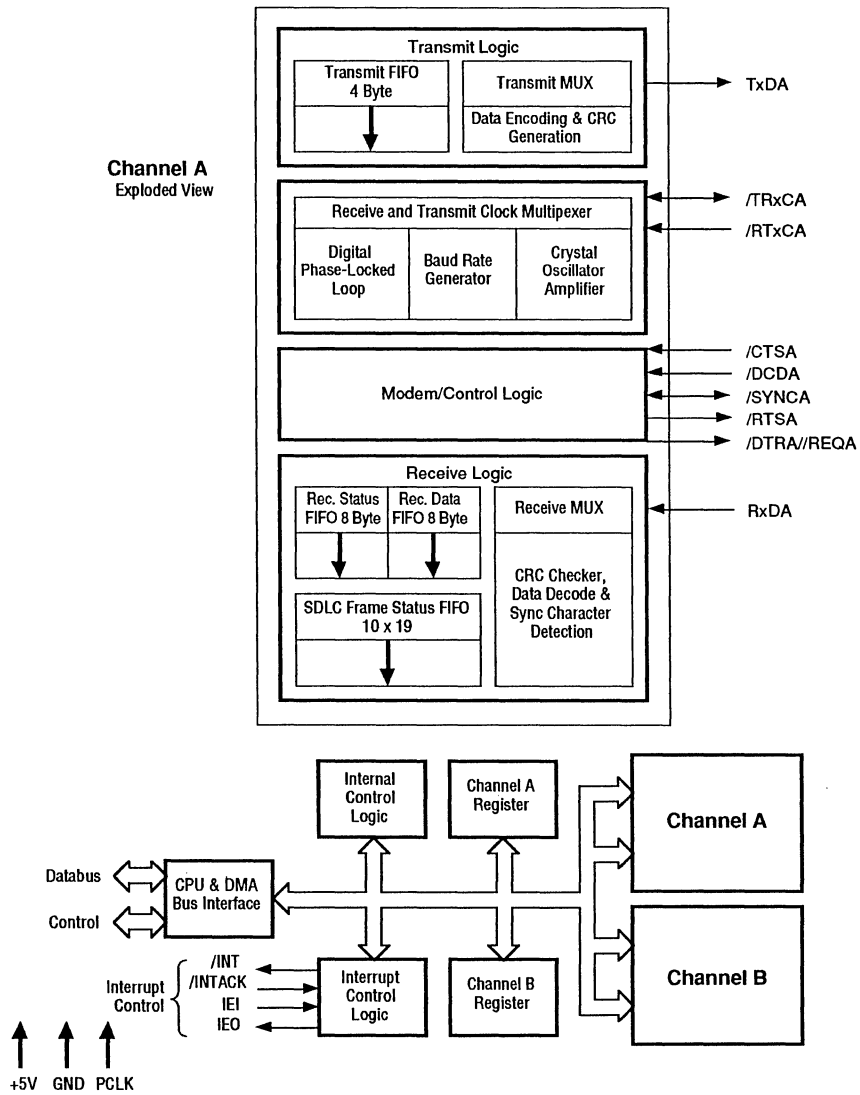


Figure 1. ESCC Block Diagram



## PIN DESCRIPTIONS

The following section describes the Z85230 pin functions. Figures 2 and 3 detail the pin assignments for the 40-pin DIP and 44-pin PLCC packages. The Z85230 ESCC is socket compatible with the Zilog Z8530 and Z85C30 as the

pin electrical characteristics and pin assignments are the same. Any unused input pins should be pulled up to the +5V supply.

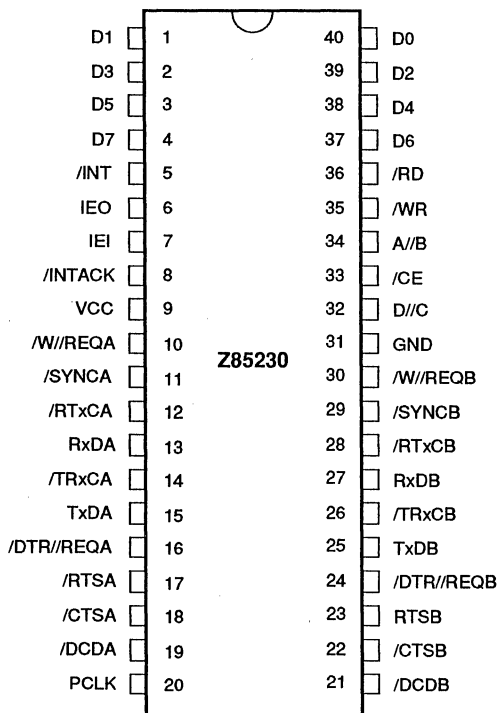


Figure 2. Z85230 DIP Pin Assignments

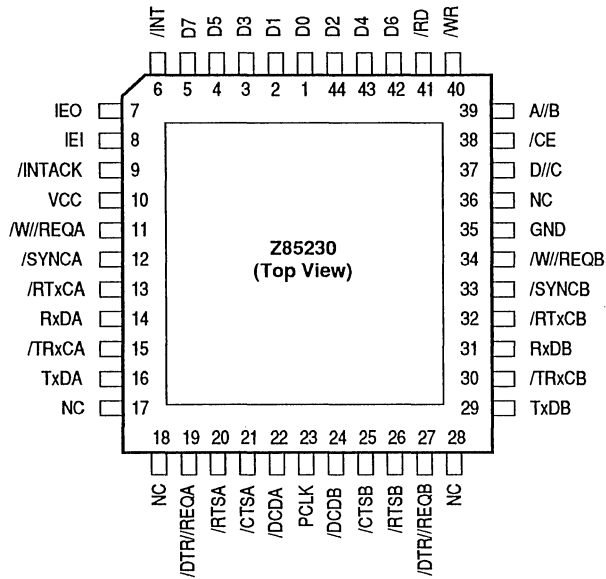


Figure 3. Z85230 PLCC Pin Assignments

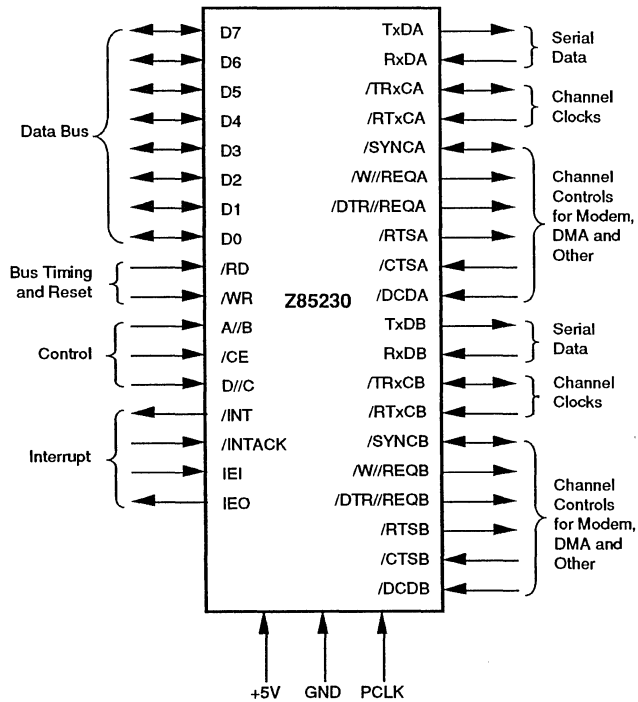


Figure 4. Z85230 Pin Functions

---

## PIN DESCRIPTIONS

**/CTSA, /CTSB.** *Clear To Send* (inputs, active Low). These pins function as transmitter enables if they are programmed for Auto Enables (WR3, D5=1). A Low on the inputs enables the respective transmitters. If not programmed as Auto Enables, they may be used as a general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The ESCC detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

**/DCDA, /DCDB.** *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if they are programmed for Auto Enables (WR3, D5=1); otherwise they are used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow rise time signals. The ESCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

**/RTSA, /RTSB.** *Request To Send* (outputs, active Low). The /RTS pins can be used as general purpose outputs or with the Auto Enables feature. When used with Auto Enables ON (WR3, D5=1) in asynchronous mode, the /RTS pin goes High after the transmitter is empty. When Auto Enable is OFF, the /RTS pins can be used as general purpose outputs and they strictly follow the inverse state of the RTS bit (WR5 bit D1).

In SDLC mode, the /RTS pins can be programmed to be deasserted when the closing flag of the message clears the TxD pin if WR7' D2 is set.

**/SYNCA, /SYNCB.** *Synchronization* (inputs or outputs, active Low). These pins can act either as inputs, outputs, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to CTS and DCD. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode, /SYNC must be driven Low for two receive clock cycles after the last bit in the synchronous character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of /SYNC.

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which synchronous condition is latched. These outputs are active each time a synchronization pattern is recognized (regardless of character boundaries). In SDLC mode, the pins act as outputs and are valid on receipt of a flag. The /SYNC pins switch from input to output when monosync, bisync, or SDLC is programmed in WR4 and sync modes are enabled.

**/DTR//REQA, /DTR//REQB.** *Data Terminal Ready/Request* (outputs, active Low). These pins are programmed (WR14, D2) to serve either as general purpose outputs or as DMA Request lines. When programmed for the DTR function (WR14, D2=0), these outputs follow the state programmed into the DTR bit of Write Register 5 (WR5, D7). When programmed for Request mode (WR14, D2=1), these pins serve as DMA Requests for the transmitter.

When used as DMA request lines, the timing for the deactivation Request can be programmed in the added register Write Register 7' (WR7') bit D4. If this bit is set, the /DTR//Request pin will be deactivated with the same timing as the /W//REQ pin. If WR7' D4 is reset, the deactivation timing of /DTR//Req pin will be the same as in the Z85C30.

**W//REQA, W//REQB.** *Wait/Request* (outputs, open drain when programmed for Wait function, driven High or Low when programmed for Ready function). These dual-purpose outputs may be programmed as Request lines for a DMA controller or as Wait lines which synchronize the CPU to the ESCC data rate. The reset state is Wait.

**RxDA, RxDB.** *Receive Data* (inputs, active High). These input signals receive serial data at standard TTL levels.

**/RTxCA, /RTxCB.** *Receive/Transmit Clocks* (inputs, active Low). These pins can be programmed to several modes of operation. In each channel, RTxC may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the digital phase-locked loop. These pins can also be programmed for use with the respective SYNC pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

---

**TxD<sub>A</sub>, TxD<sub>B</sub>.** *Transmit Data* (outputs, active High). These output signals transmit serial data at standard TTL levels.

**/TRxCA, /TRxCB.** *Transmit/Receive Clocks* (inputs or outputs, active Low). These pins can be programmed in several different modes of operation. TRxC may supply the receive clock or the transmit clock in the input mode or supply the output of the digital phase-locked loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

**PCLK.** *Clock* (input). This is the master ESCC clock used to synchronize internal signals. PCLK is a TTL level signal. PCLK is not required to have any phase relationship with the master system clock.

**IEI.** *Interrupt Enable In* (input, active High). IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt driven device. A High IEI indicates that no other higher priority device has an interrupt under service or is requesting an interrupt.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing the ESCC interrupt, or the ESCC is not requesting an interrupt (Interrupt Acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

**/INT.** *Interrupt* (output, open drain, active Low). This signal is activated when the ESCC requests an interrupt. Note that /INT is an open drain output.

**/INTACK.** *Interrupt Acknowledge* (input, active Low). This is a strobe which indicates that an interrupt acknowledge cycle is in progress. During this cycle, the ESCC interrupt

daisy chain is resolved. The device is capable of returning an interrupt vector that may be encoded with the type of interrupt pending. During the acknowledge cycle, if IEI is High the ESCC places the interrupt vector on the databus when /RD goes active. /INTACK is latched by the rising edge of PCLK.

**D7-D0.** *Data bus* (bi-directional, tri-state). These lines carry data and commands to and from the ESCC.

**/CE.** *Chip Enable* (input, active Low). This signal selects the ESCC for a read or write operation.

**/RD.** *Read* (input, active Low). This signal indicates a read operation and when the ESCC is selected, enables the ESCC's bus drivers. During the Interrupt Acknowledge cycle, /RD gates the interrupt vector onto the bus if the ESCC is the highest priority device requesting an interrupt.

**/WR.** *Write* (input, active Low). When the ESCC is selected, this signal indicates a write operation. This indicates that the CPU wants to write command bytes or data to the ESCC write registers. The coincidence of /RD and /WR is interpreted as a reset.

**A/B.** *Channel A/Channel B* (input). This signal selects the channel in which the read or write operation occurs. A High selects channel A and Low selects channel B.

**D/C.** *Data/Control Select* (input). This signal defines the type of information transferred to or from the ESCC. A High means data is being transferred and a Low indicates a command.

## FUNCTIONAL DESCRIPTION

**Architecture.** The architecture of the ESCC is described from two points of view: as a datacommunications device which transmits and receives data in a wide variety of protocols; and as a microprocessor peripheral in which the ESCC offers valuable features such as vectored interrupts and DMA support.

The ESCC's peripheral and datacommunication are described in the following sections. A block diagram is shown in Figure 1. The details of the communications between the receive and transmit logic to the system bus are shown in Figures 5 and 6. The features and data path for each of the ESCC's A and B channels is identical. See the ESCC Technical Manual for full details on using the ESCC.

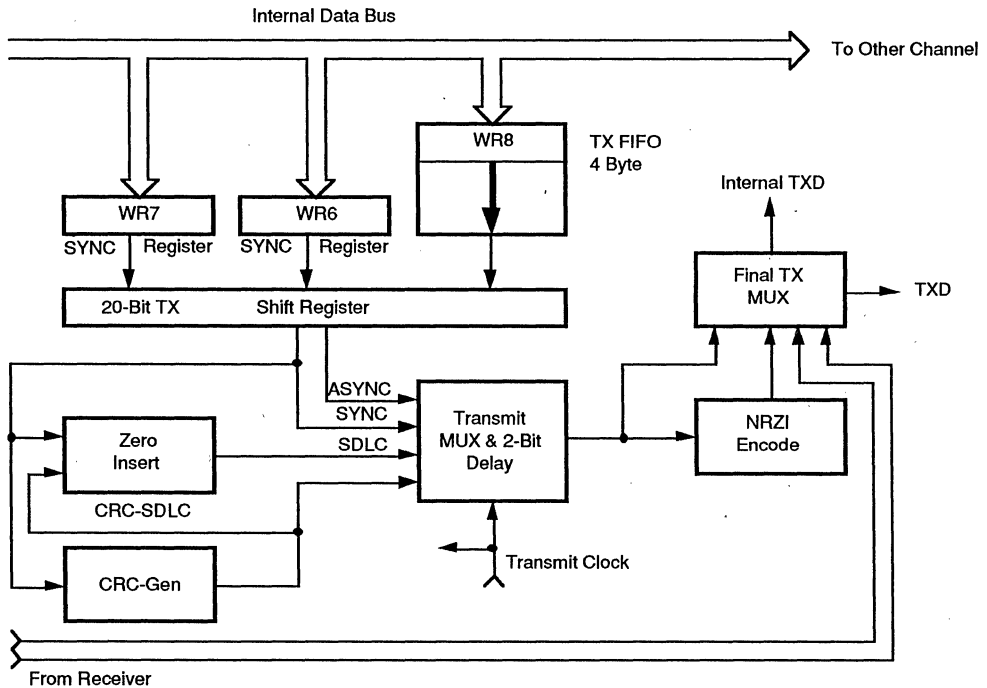


Figure 5. ESCC Transmit Data Path

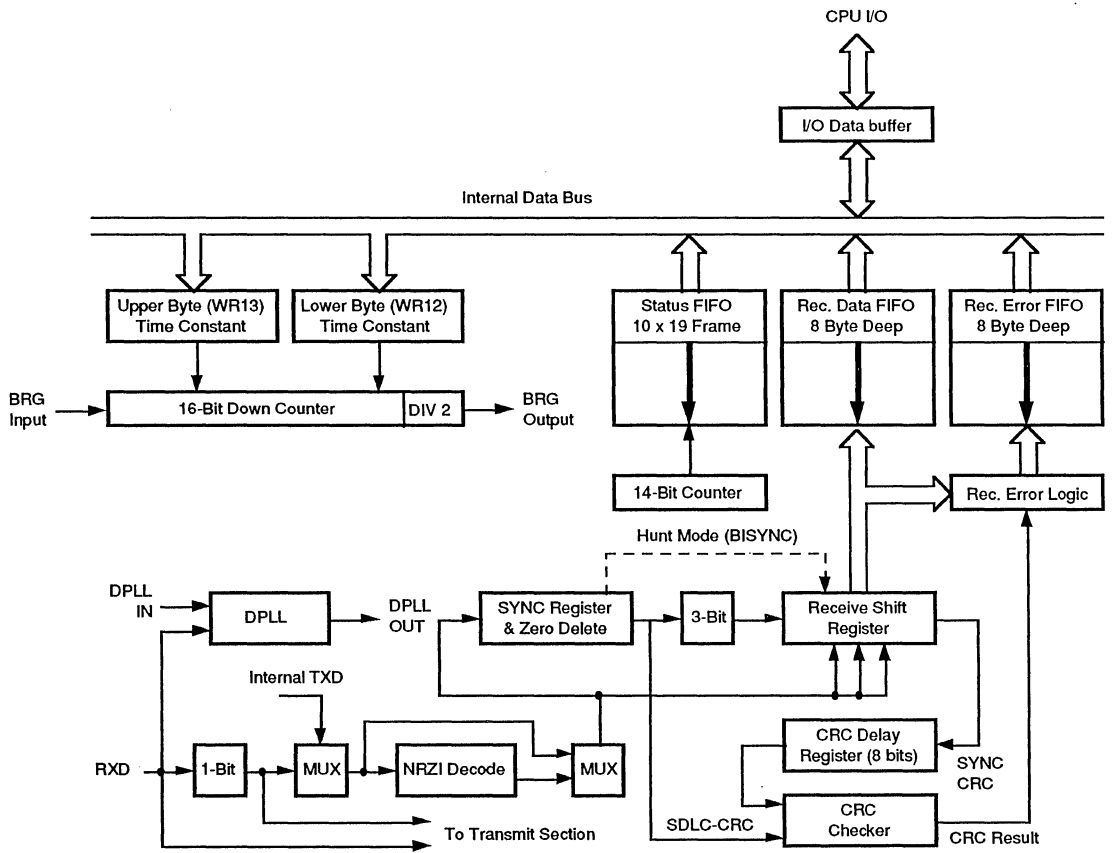


Figure 6. ESCC Receive Data Path

## I/O INTERFACE CAPABILITIES

System communication to and from the ESCC is done through the ESCC's register set. There are seventeen write registers and fifteen read registers. Many of the new features on the ESCC are enabled through a new register in the ESCC: Write Register 7 Prime (WR7'). This new register can be accessed if bit D0 of WR15 is set. Table 1 lists all of the ESCC's registers and a brief description of

their functions. Throughout this document, the write and read registers are referenced with the following notation: "WR" for Write Register and "RR" for Read Register. For example:

WR4A Write Register 4 for channel A  
RR3 Read Register 3 for either/both channels

**Table 1. ESCC Write and Read Registers**

Write Register	Functions
WR0	Command Register: Register Pointers, CRC initialization, and resets for various modes. Interrupt conditions, Wait/DMA request control. Interrupt Vector (accessed through either channel).
WR1	
WR2	
WR3	Receive and miscellaneous control parameters. Transmit and Receive parameters and modes. Transmit parameters and controls. Sync character or SDLC address field.
WR4	
WR5	
WR6	
WR7	Sync character or SDLC flag. SDLC enhancements enable (accessed if WR15 D0 is 1). Transmit FIFO (4 bytes deep). Reset commands and Master INT enable (accessed through either channel).
WR7'	
WR8	
WR9	
WR10	Miscellaneous transmit and receive controls. Clock mode control. Lower byte of BRG time constant.
WR11	
WR12	
WR13	Upper byte of BRG time constant. Miscellaneous controls and DPLL commands. External interrupt control.
WR14	
WR15	
Read Register	Functions
RR0	Transmit, Receive and external status. Special Receive Condition status bits. Unmodified interrupt vector. Modified interrupt vector.
RR1	
RR2A	
RR2B	
RR3A	Interrupt Pending bits. WR4 status (if WR7' D6=1). WR5 status (if WR7' D6=1). SDLC Frame LSB Byte Count (if WR15 D2=1). SDLC Frame 10x19 FIFO Status and MSB Byte Count (if WR15 D2=1). Receive Data FIFO (8 Deep). WR3 status (if WR7' D6=1). Miscellaneous status bits.
RR4	
RR5	
RR6	
RR7	
RR8	WR10 status (if WR7' D6=1). Lower Byte of BRG time constant. Upper byte of BRG time constant. WR7' status (if WR7' D6=1).
RR9	
RR10	
RR11	
RR12	
RR13	
RR14	

There are three choices to move data into and out of the ESCC: Polling, Interrupt (vectored and non-vectored), and Block Transfer. The Block Transfer mode can be implemented under CPU or DMA control.

**Polling.** When polling, all interrupts are disabled. Three status registers in the ESCC are automatically updated whenever any function is performed. For example, end-of-frame in SDLC mode sets a bit in one of these status registers. The purpose of polling is for the CPU to periodically read a status register until the register contents indicate the need for data to be transferred. Only one register needs to be read; depending on its contents, the CPU either writes data, reads data, or continues. Two bits in the register indicate the need for data transfer. An alternative is a poll of the Interrupt Pending register to determine the source of an interrupt. The status for both channels resides in one register.

**Interrupts.** The ESCC's interrupt structure supports vectored and nested interrupts. The fill levels where the transmit and receive FIFOs interrupt the CPU are programmable. This allows the ESCC's requests for data transfers to be tuned to the system interrupt response time.

Nested interrupts are supported with the interrupt acknowledge feature (/INTACK pin) of the ESCC. This allows the CPU to recognize the occurrence of an interrupt, and

re-enable higher priority interrupts. Because an INTACK cycle will release the /INT pin from the active state, a higher priority ESCC interrupt or another higher priority device can interrupt the CPU. When an ESCC responds to an Interrupt Acknowledge signal (INTACK) from the CPU, an interrupt vector may be placed on the data bus. This vector is written in WR2 and may be read in RR2. To speed interrupt response time, the ESCC can modify three bits in this vector to indicate status. If the vector is read in Channel A, status is never included; if it is read in Channel B, status is always included.

Each of the six sources of interrupts in the ESCC (Transmit, Receive, and External/Status interrupts in both channels) has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt Under Service (IUS), and Interrupt Enable (IE). Operation of the IE bit is straightforward. If the IE bit is set for a given interrupt source, then that source can request interrupts. The exception is when the MIE (Master Interrupt Enable) bit in WR9 is reset and no interrupts can be requested. The IE bits are write only. The other two bits are related to the interrupt priority chain (Figure 7). As a microprocessor peripheral, the ESCC may request an interrupt only when no higher priority device is requesting one, e.g., when IEI is High. If the device in question requests an interrupt, it pulls down /INT. The CPU then responds with /INTACK, and the interrupting device places the vector on the data bus.

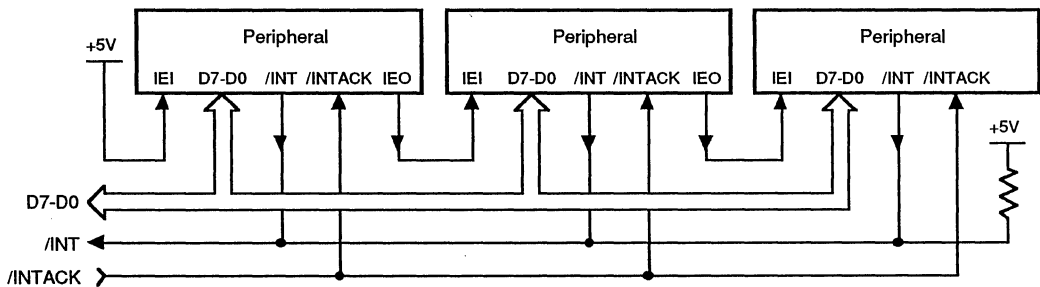


Figure 7. ESCC Interrupt Priority Schedule



---

## I/O INTERFACE CAPABILITIES (Continued)

The ESCC can also execute an interrupt acknowledge cycle through software. In some CPU environments it is difficult to create the /INTACK signal with the necessary timing to acknowledge interrupts and allow the nesting of interrupts. In these cases, the /INTACK signal can be created with a software command to the ESCC. See the Z85230 Enhancements section for more details on this enhancement.

In the ESCC, the Interrupt Pending (IP) bit signals a need for interrupt servicing. When an IP bit is 1 and the IEI input is High, the /INT output is pulled Low, requesting an interrupt. In the ESCC, if the IE bit isn't set by enabling interrupts, then the IP for that source is never set. The IP bits are readable in RR3A.

The IUS bits signal that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority in the ESCC and external to the ESCC are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the ESCC being pulled Low and propagated to subsequent peripherals. An IUS bit is set during an Interrupt Acknowledge cycle if there are no higher priority devices requesting interrupts.

There are three types of interrupts: Transmit, Receive, and External/Status. Each interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receiver, Transmit, and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled (WR1 D1=1), the occurrence of the interrupt depends on the state of WR7' D5. If this bit is reset, the CPU is interrupted when the top byte of the transmit FIFO becomes empty. If WR7' D5 is set, the CPU is interrupted when the transmit FIFO is completely empty. (This implies that the transmitter must have had a data character written into it so that it can become empty.)

When enabled, the receiver can interrupt the CPU in one of three ways:

1. Interrupt on First Receive Character or Special Receive Condition.
2. Interrupt on All Receive Characters or Special Receive Conditions.
3. Interrupt on Special Receive Conditions Only.

If WR7' bit D3 is set, the Receive character interrupt occurs when there are four bytes available in the receive FIFO. This is most useful in synchronous applications as the data is in consecutive bytes. Interrupt on First Character or Special Condition and Interrupt on Special Condition Only are typically used with the Block Transfer mode. A special Receive Condition is one of the following: receiver overrun; framing error in Asynchronous mode, end-of-frame in SDLC mode and, optionally, a parity error. The Special Receive Condition interrupt is different from an ordinary receive character available interrupt only by the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt occurs from Special Receive Conditions any time after the first receive character interrupt.

The main function of the External/Status interrupt is to monitor the signal transitions of the /CTS, /DCD, and /SYNC pins, however, an External/Status interrupt is also caused by a Transmit Underrun condition; a zero count in the baud rate generator; by the detection of a Break (Asynchronous mode), ABORT (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the ABORT or EOP has a special feature allowing the ESCC to interrupt when the ABORT or EOP sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the ABORT condition by external logic in SDLC mode. In SDLC Loop mode, this feature allows secondary stations to recognize the primary station wishes to regain control of the loop during a poll sequence.

**CPU/DMA Block Transfer.** The ESCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode used the /WAIT//REQUEST output in conjunction with the Wait/Request bits in WR1. The /WAIT//REQUEST output can be defined under software control as a WAIT line in the CPU Block Transfer mode or as a REQUEST line in the DMA Block Transfer mode.

To a DMA controller, the ESCC REQUEST output indicates that the ESCC is ready to transfer data to or from memory. To the CPU, the WAIT line indicates that the ESCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The /DTR//REQUEST line allows full-duplex operation under DMA control. The ESCC can be programmed to deassert the /DTR//REQUEST pin with the same timing as the /WAIT//REQUEST pin if WR7' D4 is set.

## ESCC DATA COMMUNICATIONS CAPABILITIES

The ESCC provides two independent full-duplex programmable channels for use in any common asynchronous or synchronous data communication protocols (Figure 8).

Each of the datacommunication channels has identical features and capabilities.

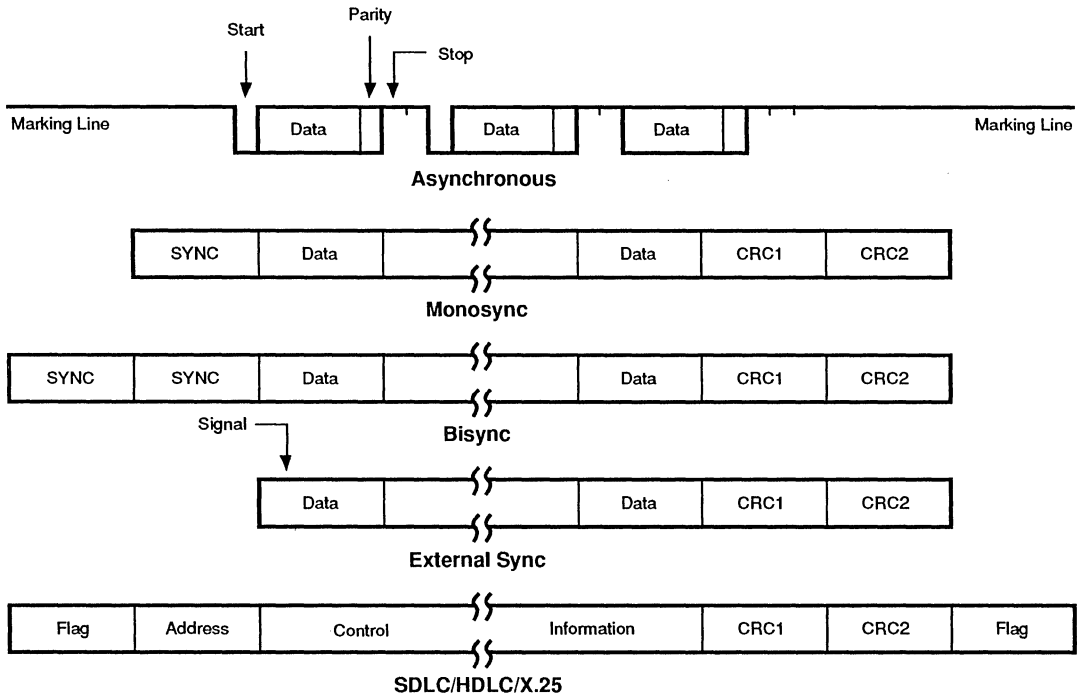


Figure 8. Some ESCC Protocols

The ESCC has significant improvements to its data communications capacity over that of the standard SCC. The addition of the deeper data FIFOs allows for data to be moved in strings instead of on a byte-by-byte basis. The ability to handle data in strings allows for significant improvements in data handling, and consequently, more efficient use of bus bandwidth. The programmability of the INT/DMA level of the FIFOs allows the system designer to determine fill levels as the FIFO's request the system to move data. The deeper data FIFOs are accessible regardless of the protocol used. They do not need to be enabled. For more details on these improvements, see the Z85230 Enhancements section of this specification.

**Asynchronous Modes.** Send and Receive is accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half, or two stop bits per

character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxD<sub>A</sub> or RxD<sub>B</sub> pins). If the Low does not persist (e.g., a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur. Vectored interrupts allow fast servicing or error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

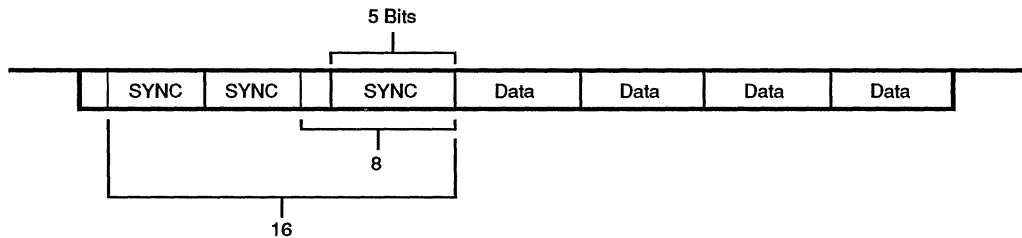
## ESCC DATA COMMUNICATIONS CAPABILITIES (Continued)

The ESCC does not require symmetric transmit and receive clock signals - a feature allowing use of the wide variety of clock sources. The transmitter and receiver handle data at a rate supplied to the receive and transmit clock inputs. In Asynchronous modes, the SYNC pin may be programmed as an input used for functions such as monitoring a ring indicator.

**Synchronous Modes.** The ESCC supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols are handled in several

modes. They allow character synchronization with a 6-bit or 8-bit sync character (Monosync), and a 12-bit or 16-bit synchronization pattern (Bisync), or with an external sync signal. Leading sync characters are removed without interrupting the CPU.

Five or 7-bit synchronous characters are detected with 8- or 16-bit patterns in the ESCC by overlapping the larger pattern across multiple incoming synchronous characters as shown in Figure 9.



**Figure 9. Detecting 5- or 7-Bit Synchronous Characters**

CRC checking for Synchronous byte oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols such as IBM Bisync.

Both CRC-16 ( $X^{16} + X^{15} + X^2 + 1$ ) and CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) error checking polynomials are supported. Either polynomial may be selected in all Synchronous modes. Users may preset the CRC generator and checker to all 1's or all 0's. The ESCC also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows for high-speed transmissions under DMA control, with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 6-, 8-, or 16-bit sync characters, regardless of the programmed character length.

**SDLC Mode.** The ESCC supports Synchronous bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero insertion, and CRC generation. A special command is used to abort a frame in transmission. At the end of a message, the ESCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line consisting of continuous flag characters or a steady marking condition.

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort can be issued. The ESCC may also be programmed to send an ABORT itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent, allowing reception of a message with no prior information about the character structure in the information field of a frame.

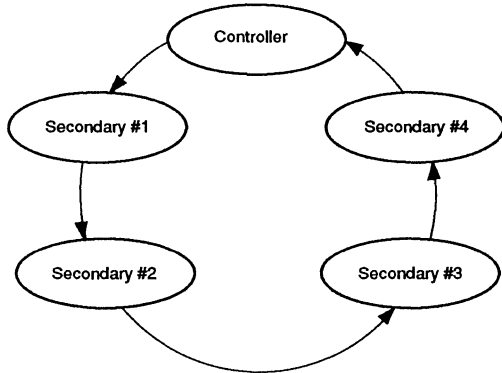
The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the /SYNC pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored.

The number of address bytes are extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all 0's inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. In SDLC mode, the ESCC

must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all 1's or all 0's. The CRC is inverted before transmission and the receiver checks against the bit pattern 0001110100001111.

NRZ, NRZI or FM coding may be used in any 1x mode. The parity options available in Asynchronous modes are available in Synchronous modes.

**SDLC Loop Mode.** The ESCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow on the loop and any number of secondary stations. In SDLC Loop mode, the ESCC performs the functions of a secondary station while an ESCC operating in regular SDLC mode acts as a controller (Figure 10). SDLC loop mode can be selected by setting WR10 bit D1.



**Figure 10. An SDLC Loop**

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop and, in fact, passes these messages to the rest of the loop by retransmitting them with a one-bit time delay. The secondary station places its own message on the loop only at specific times. The controller signals that secondary stations can transmit messages by sending a special character, called an EOP (End Of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages, this bit pattern is unique and easily recognized.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit append their

messages to the message of the first secondary station by the same process. Any secondary stations without messages to send merely echo the incoming message and are prohibited from placing messages on the loop (except upon recognizing an EOP). In SDLC Loop mode, NRZ, NRZI, and FM coding may all be used.

**SDLC FIFO.** The ESCC's ability to receive high speed back-to-back SDLC frames is maximized by a 10-bit deep by 19-bit wide status FIFO. When enabled (through WR15, bit D2), it provides the DMA the ability to continue to transfer data into memory so that the CPU can examine the message later. For each SDLC frame, a 14-bit byte count and 5 status/error bits are stored. The byte count and status bits are accessed through Read Registers 6 and 7. Read Registers 6 and 7 are only accessible when the SDLC FIFO is enabled. The 10x19 status FIFO is separate from the 8-byte receive data FIFO.

**Baud Rate Generator.** Each channel in the ESCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On startup, the flip-flop on the output is set in a High state, the value in the time constant register is loaded into the counter, and the counter starts counting down. The output of the baud rate generator toggles upon reaching 0, the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the digital phase-locked loop (see next section).

If the receive clock or transmit clock is not programmed to come from the TRxC pin, the output of the baud rate generator may be echoed out via the TRxC pin.

The following formula relates the time constant to the baud rate where PCLK or RTxC is the baud rate generator input frequency in Hertz. The clock mode is 1, 16, 32, or 64, as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select 1 and Asynchronous should select 16, 32 or 64.

$$\text{Time Constant} = \frac{\text{PCLK or RTxC Frequency}}{2(\text{Baud Rate}) (\text{Clock Mode})} - 2$$

## ESCC DATA COMMUNICATIONS CAPABILITIES (Continued)

**Digital Phase-Locked Loop.** The ESCC contains a Digital Phase-Locked Loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock is then used as the ESCC receive clock, the transmit clock, or both. When the DPLL is selected as the transmit clock source, it will provide a jitter free clock output that is the DPLL input frequency divided by the appropriate divisor for the selected encoding technique.

For NRZI encoding, the DPLL counts the 32x clock to create nominal bit times. As the 32x clock is counted, the DPLL is searching the incoming data stream for edges (either 1 to 0, or 0 to 1). Whenever an edge is detected, the DPLL makes a count adjustment (during the next counting cycle), producing a terminal count closer to the center of the bit cell.

For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16 and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15 to 16 counting transition.

The 32x clock for the DPLL can be programmed to come from either the RTxC input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the ESCC via the TRxC pin (if this pin is not being used as an input).

**Data Encoding.** The ESCC may be programmed to encode and decode the serial data in four different ways (Figure 11). In NRZ encoding, a 1 is represented by a High level and a 0 is represented by a Low level. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. In FM1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A 1 is represented by an additional transition at the center of the bit cell and a 0 is represented by no additional transition at the center of the bit cell. In FM0 (bi-phase space), a transition occurs at the beginning of every bit cell. A 0 is represented by an additional transition at the center of the bit cell, and a 1 is represented by no additional transition at the center of the bit cell. In addition to these four methods, the ESCC can be used to decode

Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0 to 1, the bit is a 0. If the transition is 1 to 0, the bit is a 1.

**Auto Echo and Local Loopback.** The ESCC is capable of automatically echoing everything it receives. This feature is useful mainly in Asynchronous modes, but works in Synchronous and SDLC modes as well. Auto Echo mode (TxD is RxD) is used with NRZI or FM encoding with no additional delay because the data stream is not decoded before retransmission. In Auto Echo mode, the /CTS input is ignored as a transmitter enable (although transitions on this input can still cause interrupts if programmed to do so). In this mode, the transmitter is actually bypassed and the programmer is responsible for disabling transmitter interrupts and /WAIT//REQUEST on transmit.

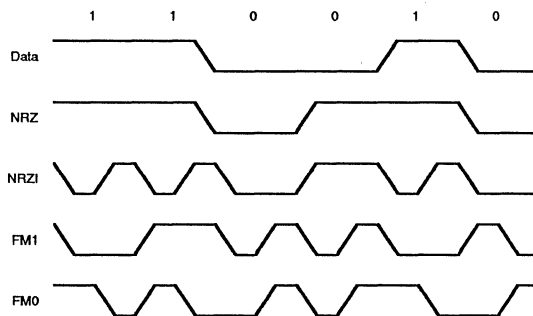


Figure 11. Data Encoding Methods

The ESCC is also capable of Local Loopback. In this mode, TxD or RxD is just like Auto Echo mode. However, in Local Loopback mode the internal transmit data is tied to the internal receive data and RxD is ignored (except to be echoed out via TxD). The /CTS and /DCD inputs are also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in Asynchronous, Synchronous and SDLC modes with NRZ, NRZI or FM coding of the data stream.

## Z85230 ENHANCEMENTS

The following is a detailed description of the enhancements to the Z85230, ESCC from the standard SCC.

### 4-Byte Deep Transmit FIFO

The ESCC has a 4-byte transmit buffer with programmable interrupt and DMA request levels. It is not necessary to enable the FIFO as it is always available. The user can choose to have the Transmit Buffer Empty (TBE) interrupt and DMA Request on Transmit be generated either when the top byte of transmit FIFO is empty or only when the FIFO is completely empty. A hardware or channel reset will reset the transmit shift register, flush the transmit FIFO, and set WR7' D5=1.

If the transmitter generates the Interrupt or DMA request for data when the top byte of the FIFO is empty (WR7' D5=0), the system can allow for a long response time to the data request without underflowing. The interrupt service routine can write one byte and then test RR0 D2 if more data may be written. The DMA Request in this mode will go inactive after each data write and then go active again until the FIFO is filled. The Transmit Buffer Empty status bit (TBE), RR0 bit D2, is set when the top byte of the FIFO is empty. Note that this IS NOT the reset state.

For applications where the frequency of interrupts is important, the transmit interrupt service routine can be optimized by programming the ESCC to generate the TBE interrupt only when the FIFO is completely empty (WR7' D5=1) and then writing four bytes to fill the FIFO. When WR7' D5=1, only one DMA request is generated (filling the bottom of the FIFO). However, this may be preferred for some applications where the possible reassertion of the DMA request is not desired. The Transmit Buffer Empty status bit (TBE), RR0 bit D2, is set when the top byte of the FIFO is empty. (Note that WR7' D5=1 after a hardware or channel reset).

### 8-Byte Receive FIFO

The ESCC has an 8-byte receive FIFO with programmable interrupt levels. The receive character available interrupt is generated as selected by WR7' bit D3. The Receive Character Available bit, RR0 D0, is set when at least one byte is available in the top of the FIFO (independent of WR7' D3). It is not necessary to enable the 8-byte FIFO as it is always available. A hardware or channel reset resets the receive shift register and flushes the receive FIFO.

A DMA Request on Receive, if enabled, is generated whenever one byte is available in the receive FIFO independent of WR7' D3. If more than one byte is available in the FIFO, the /Wait//Request pin goes inactive and then goes active again until the FIFO is emptied.

By resetting WR7' D3=0, applications which have a long latency to interrupts can generate the request to read data from the FIFO when one byte is available, and then test the Receive Character Available bit to determine if more data is available.

By setting WR7' D3=1, the ESCC can be programmed to interrupt when the receive FIFO is half full (4 bytes available) and, therefore, allowing the frequency of receive interrupt to be reduced. If WR7' D3 is set, the receive character available interrupt is generated when there are 4 bytes available. Therefore, if the interrupt service routine reads 4 bytes during each routine, the frequency of interrupts is reduced.

If WR7' D3=1 and "Receive Interrupt on All Characters and Special Conditions" is enabled, the receive character available interrupt is generated when four characters are available. However, when a character is detected to have a special condition, a special condition interrupt is generated when the character is loaded into the top four bytes of the FIFO. Therefore, the special condition interrupt service routine should read RR1 before reading the data to determine which byte has the special condition.

### Write Register 7' (7 prime)

A new register, WR7', has been added to the ESCC to facilitate the programming of six new features. The format of this register is shown in Figure 12.

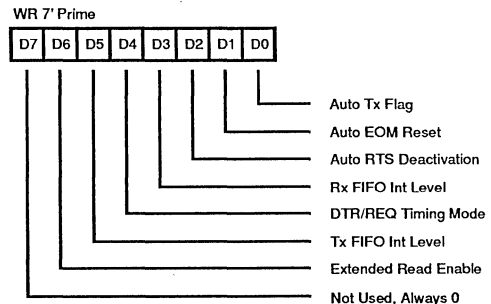


Figure 12. Write Register 7' (7 prime)

WR7' is written to by first setting bit D0 of Write Register 15 (WR15 D0) to one, and then addressing WR7 as normal. All writes to register 7 are to WR7' while WR15 D0 is set. WR15 bit D0 must be reset to 0 to address the sync character register WR7. If bit D6 of WR7' is set, then WR7' can be read by doing a read cycle to RR14. The WR7' features remain enabled until specifically disabled or by a hardware or software reset. Note that bit D5 is set after a reset. All other bits are reset to zero following reset.

## Z85230 ENHANCEMENTS (Continued)

For applications which may use either the Zilog Z85C30 or Z85230, these two device types can be identified in software with the following test. Write a 01 hex to Write Register 15. Then read Read Register 15 and if D0 is reset it is a Z85C30 and, if D0 is set it is a Z85230. Note that if the device is Z85C30, a write to WR15 resetting D0 should be done before proceeding. Also, if the device is Z85230, the result in all writes to address seven will be to WR7' until WR15 D0 is reset.

**Bit 7.** *Not used.* This bit must always be written zero (0).

**Bit 6.** *Extended Read Enable.* Setting this bit enables the ability to read WR3, WR4, WR5, WR7' and WR10. These registers are read by reading RR9 (WR3), RR4, RR5, RR14 (WR7'), and RR11 (WR10), respectively.

**Bit 5.** *Transmit FIFO Interrupt Level.* If this bit is set, the transmit buffer empty interrupt is generated when the

transmit FIFO is completely empty. If this bit is reset, the transmit buffer empty interrupt is generated when the top byte of the transmit FIFO is empty. This bit is set following a hardware or channel reset.

In DMA Request on Transmit mode, when using either the /W//REQ or /DTR//REQ pins, the request is asserted when the Tx FIFO is completely empty if WR7' D5 is set. The request is asserted when the top byte of the FIFO is empty if D5 is reset.

**Bit 4.** */DTR//REQ timing.* If this bit is set and the /DTR//REQ pin is used for Request mode (WR14 D2=1), the deactivation of the /DTR//REQ pin will be identical to the /W//REQ pin as shown in Figure 13. If this bit is reset, the deactivation time is 4TcPc.

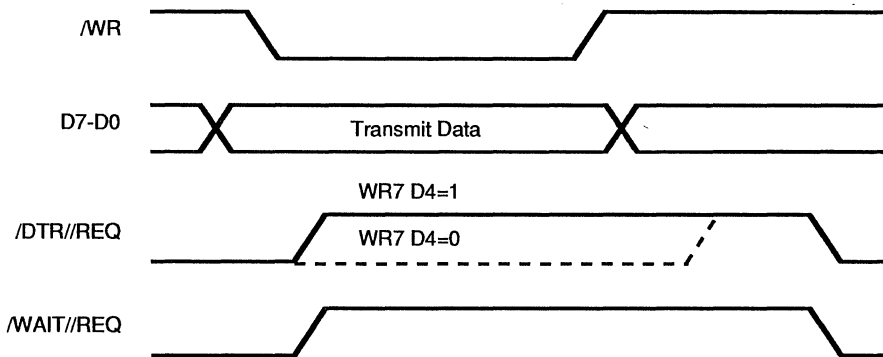


Figure 13. DMA Request on Transmit Deactivation Timing

**Bit 3.** *Receive FIFO Interrupt Level.* This bit sets the interrupt level of the receive FIFO. If this bit is set, the receive data available bit is asserted when the receive FIFO is half full (4 bytes available). If the RFF bit is reset, the receive data available interrupt is generated when a byte reaches the top of the FIFO. See the description of the 8-byte receive FIFO for more details.

**Bit 2.** *Automatic /RTS Pin Deassertion.* This bit controls the timing of the deassertion of the /RTS pin in SDLC mode. If this bit is set and WR5 D1 is reset during the transmission of a SDLC frame, the deassertion of the /RTS pin is delayed until the last bit of the closing flag clears the Tx D pin. The /RTS pin is pulled high after the rising edge of the transmit clock cycle from the last bit of the closing flag. This implies

that the ESCC should be programmed for "Flag on Underrun" (WR10 D2=0) for the /RTS pin to deassert at the end of the frame. This feature works independently of the programmed transmitter idle state. In synchronous modes other than SDLC, the /RTS pin will immediately follow the state programmed into WR5 D1. When WR7' D2 is reset, the /RTS follows the state of WR5 D1.

**Bit 1.** *Automatic EOM Reset.* If this bit is set, the ESCC automatically resets the Tx Underrun/EOM latch and pre-sets the transmit CRC generator to its programmed preset state (per values set in WR5 D2 and WR10 D7). Therefore, it is not necessary to issue the Reset Tx Underrun/EOM latch command when this feature is enabled.

**Bit 0. Automatic Tx SDLC Flag.** If this bit is set, the ESCC will automatically transmit an SDLC flag before transmitting data. This removes the requirement to reset the mark idle bit (WR10 D3) before writing data to the transmitter.

**Modified Databus Timing**

The ESCC's latching of the databus has been modified to simplify the CPU interface. The Z85C30 AC Timing parameter #29, Write Data to /WR falling minimum, has been changed for the Z85230 to: /WR falling to Write Data Valid maximum. See the AC Timing Characteristic section for the specified time at each clock speed. The databus must be valid no later than 20 ns after the falling edge of /WR regardless of the system (PCLK) clock rate. The databus hold time, spec #30, remains at 0ns.

Historically, the SCC has latched the databus on the falling edge of /WR. However, as many CPUs do not guarantee that the databus is valid when the /WR pin goes low, Zilog has modified the databus timing to allow a maximum delay from the /WR signal going active Low to the latching of the databus.

**Complete CRC Reception in SDLC Mode**

In SDLC mode, the entire CRC is clocked into the receive FIFO. The ESCC completes clocking in the CRC to allow it to be retransmitted, unaltered, or manipulated in software. In the SCC when the closing flag is recognized, the contents of the receive shift register are immediately transferred to the receive FIFO resulting in the last two bits of the CRC being lost. In the ESCC, it is not necessary to program this feature. When the closing flag is detected, the last two bits of the CRC are clocked into the receive FIFO. In all other synchronous modes, the ESCC does not clock in the last two CRC bits (same as SCC).

**TxD Forced High in SDLC with NRZI**

**Encoding When Marking Idle**

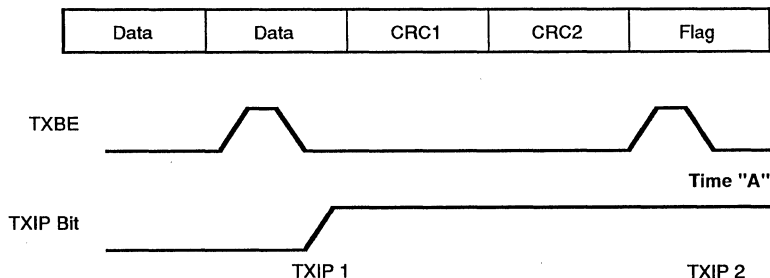
When the ESCC is programmed for SDLC mode with NRZI data encoding and mark idle (WR10 D6=0, D5=1, D3=1),

the TxD pin is automatically forced high when the transmitter goes to the mark idle state. There are several different ways for the transmitter to go into the idle state. In each of the following cases the TxD pin is forced high when the mark idle condition is reached: data, CRC, flag and idle; data, flag and idle; data, abort (on underrun) and idle; data, abort (command) and idle; idle flag and command to idle mark. The force high feature is disabled when the mark idle bit is reset.

This feature is used in combination with the automatic SDLC opening flag transmission feature, WR7' D0=1, to assure that data packets are properly formatted. Therefore, when these features are used together, it is not necessary for the CPU to issue any commands when using the force idle mode in combination with NRZI data encoding. If WR7' D0 is reset, like in the SCC, it is necessary to reset the mark idle bit (WR10 D3) to enable flag transmission before an SDLC packet is transmitted.

**Improved Transmit Interrupt Handling in Synchronous Modes**

The ESCC latches the Transmit Buffer Empty (TBE) interrupt due to the CRC being loaded to the transmit shift register even if the TBE interrupt, due at the last data byte, has not yet been reset. Therefore, the end of a synchronous frame is guaranteed to generate two TBE interrupts even if a reset transmit buffer interrupt command for the data created interrupt is issued after (time "A" in Figure 14) the CRC interrupt had occurred. In this case, two reset TBE commands are required. The TxIP is latched if the EOM latch has been reset before the end of the frame.



**Figure 14. TxIP Latching**



## NEW FEATURE DESCRIPTION (Continued)

### DPLL Counter Tx Clock Source

When DPLL output is selected as the transmit clock source, the DPLL counter output is the DPLL source clock divided by the appropriate divisor for the programmed data encoding format. Therefore, in FM mode (FM0 or FM1), the DPLL counter output is the input frequency divided by 16.

In NRZI mode, the DPLL counter frequency is the input divided by 32. This feature provides a jitter-free output and replaces the DPLL transmit clock output being available as the transmit clock source. This has no effect on the use of the DPLL as the receive clock source (Figure 15).

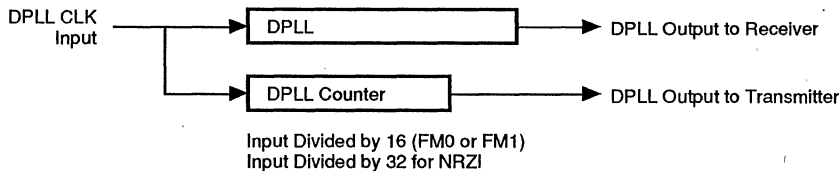


Figure 15. DPLL Outputs

### Read Register 0 Status Latched During Read Cycle

The contents of Read Register 0, RR0, are latched during a read to this register. The ESCC prevents the contents of RR0 to change while the Read cycle is active. The SCC allows the status of RR0 to change while reading the register and, therefore, it is necessary to read RR0 twice to detect changes that otherwise may be missed. The contents of RR0 are updated after the rising edge of /RD.

### Software Interrupt Acknowledge

The Z85230 interrupt acknowledge cycle can be initiated through software. If Write Register 9 (WR9) bit D5 is set, reading register 2 (RR2) results in an interrupt acknowledge cycle to be executed internally. Like a hardware INTACK cycle, a software acknowledge causes the INT pin to return high, the IEO pin to go Low and set the IUS latch for the highest priority interrupt pending.

Similar to when the hardware INTACK signal can be used, a software acknowledge cycle requires that a Reset Highest IUS command be issued in the interrupt service routine. Whenever an interrupt acknowledge cycle is used, hardware or software, a reset highest IUS command is required. If RR2 is read from channel A, the unmodified vector is returned. If RR2 is read from channel B, then the vector is modified to indicate the source of the interrupt. The Vector Includes Status (VIS) and No Vector (NV) bits in WR9 are ignored when bit D5 is set to 1.

When the INTACK and IEI pins are not being used, they should be pulled up to  $V_{cc}$  through a resistor (10k Ohm typical).

### Fast SDLC Transmit Data Interrupt Response

To more easily facilitate the transmission of back-to-back SDLC frames with a single shared flag between frames, the ESCC allows data for a second frame to be written to the transmit FIFO after the Tx Underrun/EOM interrupt has occurred. This allows application software more time to write the data to the transmitter while allowing the current frame to be properly concluded with CRC and flag. The SCC historically has required that data not be written to the transmitter until a transmit buffer empty interrupt was generated after the CRC has completed transmission. If data is written to the transmit FIFO after the Transmit Underrun/EOM interrupt and before the transmit buffer empty interrupt, the Automatic EOM Reset feature should be enabled (WR7' D1=1). Consequently, the commands "Reset Tx/Underrun EOM" latch and "Reset Tx CRC Generator" should not be used.

### SDLC FIFO Frame Status FIFO Enhancement

When used with a DMA controller, the Z85230 SDLC Frame Status FIFO enhancement maximizes the ESCC's ability to receive high speed, back-to-back SDLC messages. It minimizes frame overruns due to CPU latencies in responding to interrupts. Additional logic was added to the industry standard SCC consisting of a 10-bit deep by 19-bit wide status FIFO, 14-bit receive byte counter, and control logic as shown in Figure 16. The 10 x 19 bits status FIFO is separate from the 8-byte receive data FIFO.

When the enhancement is enabled, the status in Read Register 1 (RR1) and byte count for the SDLC frame are stored in the 10 x 19-bit status FIFO. This allows the DMA controller to transfer the next frame into memory while the CPU verifies that the message was properly received.

---

Summarizing the operation; data is received, assembled, and loaded into the 8-byte FIFO before being transferred to memory by the DMA controller. When a flag is received at the end of an SDLC frame, the frame byte count from the 14-bit counter and five status bits are loaded into the status FIFO for verification by the CPU. The CRC checker is automatically reset in preparation for the next frame which can begin immediately. Since the byte count and status are saved for each frame, the message integrity is verified at a later time. Status information for up to 10 frames is stored before a status FIFO overrun can occur.

If a frame is terminated with an ABORT, the byte count and status will be loaded to the status FIFO and the counter reset for the next frame.

**FIFO Detail.** For a better understanding of details of the FIFO operation, refer to the block diagram in Figure 16.

**Enable/Disable.** This FIFO is implemented so that it is enabled when WR15, bit D2, is set and the ESCC is in the SDLC/HDLC mode. Otherwise, the status register contents bypass the FIFO and go directly to the bus interface (the FIFO pointer logic is reset either when disabled or via a channel or power-on reset). When the FIFO mode is disabled, the ESCC is completely downward compatible with the NMOS Z8530. The FIFO mode is disabled on power-up (WR15 D2 is set to 0 on reset). The effects of backward compatibility on the register set are that RR4 is an image of RR0, RR5 is an image of RR1, RR6 is an image of RR2 and RR7 is an image of RR3. For details on the added registers, refer to Figure 18. The status of the FIFO Enable signal is obtained by reading RR15, bit D2. If the FIFO is enabled, the bit will be set to 1; otherwise, it will be reset.

**Read Operation.** When WR15 bit D2 is set and the FIFO is not empty, the next read to status register RR1 or the additional registers RR7 and RR6, are from the FIFO. Reading status register RR1 causes one location of the FIFO to be emptied, so status is read after reading the byte count, otherwise the count is incorrect. Before the FIFO underflows, it is disabled. In this case, the multiplexer is switched to allow status to read directly from the status register. Reads from RR7 and RR6 contain bits that are undefined. Bit D6 of RR7 (FIFO Data Available) is used to determine if status data is coming from the FIFO or directly from the status register, since it is set to 1 whenever the FIFO is not empty.

Since not all status bits are stored in the FIFO, the All Sent, Parity, and EOF bits bypass the FIFO. The status bits sent through the FIFO are Residue Bits (3), Overrun, and CRC Error.

The sequence for proper operation of the byte count and FIFO logic is to read the registers in the following order: RR7, RR6, and RR1 (reading RR6 is optional). Additional logic prevents the FIFO from being emptied by multiple reads from RR1. The read from RR7 latches the FIFO empty/full status bit (D6) and steers the status multiplexer to read from the ESCC megacell instead of the status FIFO (since the status FIFO is empty). The read from RR1 allows an entry to be read from the FIFO (if the FIFO was empty, logic was added to prevent a FIFO underflow condition).

**Write Operation.** When the end of an SDLC frame (EOF) has been received and the FIFO is enabled, the contents of the status and byte-count registers are loaded into the FIFO. The EOF signal is used to increment the FIFO. If the FIFO overflows, the RR7 bit D7 (FIFO Overflow) is set to indicate the overflow. This bit and the FIFO control logic is reset by disabling and re-enabling the FIFO control bit (WR15 bit D2). For details of FIFO control timing during an SDLC frame, refer to Figure 17.

**SDLC Status FIFO Anti-Lock Feature.** When the Frame Status FIFO is enabled and the ESCC is programmed for "Special Receive Condition Only" (WR1 D4=D3=1), the data FIFO is not locked when a character with End of Frame status is read (Figure 16). When a character with the EOF status is at the top of the FIFO, an interrupt with a vector for receive data is generated. The command "Reset Highest IUS" must be issued at the end of the interrupt service routine regardless if an interrupt acknowledge cycle had been executed (hardware or software). This allows a DMA to complete transfer of the received frame to memory and then interrupt the CPU that a frame has been completed without locking the FIFO. Since in the "Receive Interrupt on Special Condition Only" mode the interrupt vector for receive data is not used, it is used to indicate that the last byte of a frame has been read out the receive FIFO. Reading the frame status (CRC, byte count and other status stored in the status FIFO) to determine EOF is not required.

When a character with a special receive condition other than EOF is received (receiver overrun, or parity), a special receive condition interrupt is generated after the character is read from the FIFO and the receive FIFO is locked until the "Error Reset" command is issued.

NEW FEATURE DESCRIPTION (Continued)

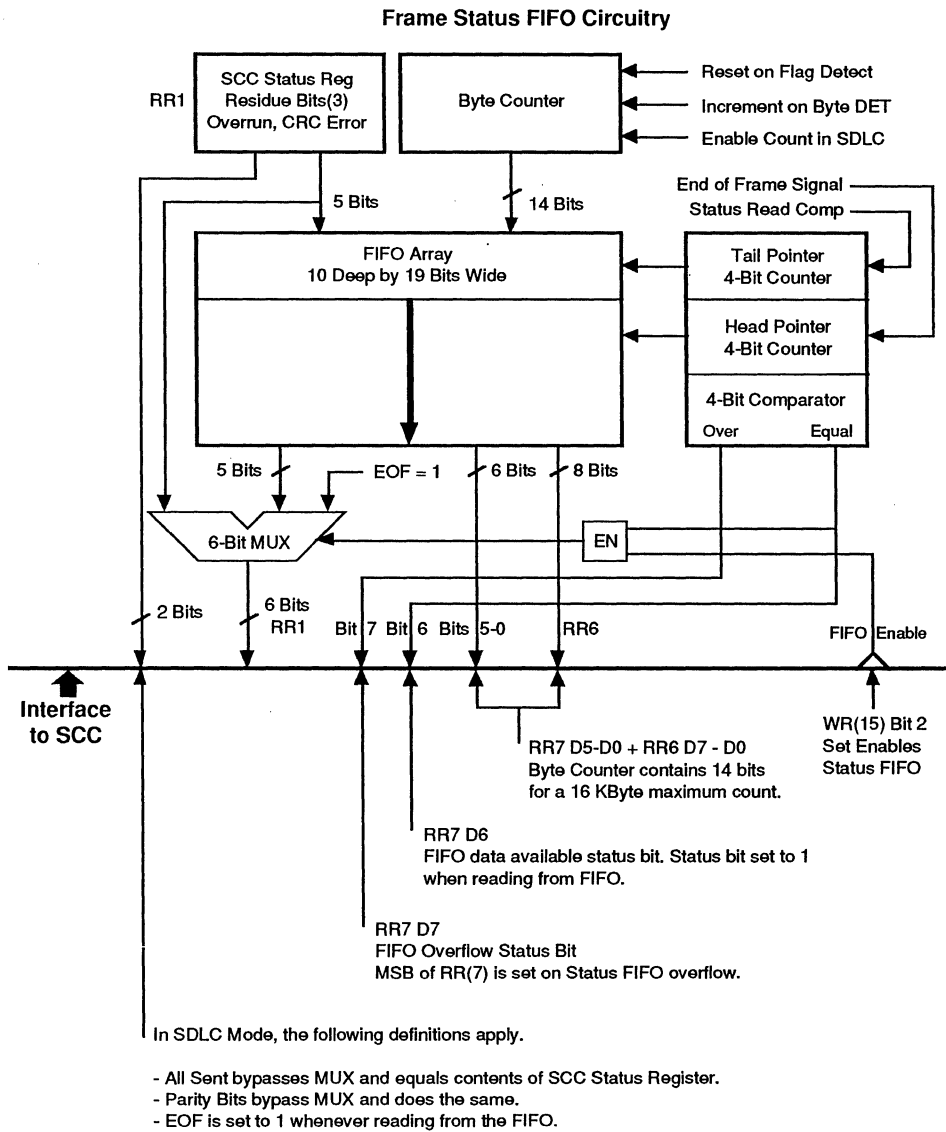


Figure 16. SDLC Frame Status FIFO

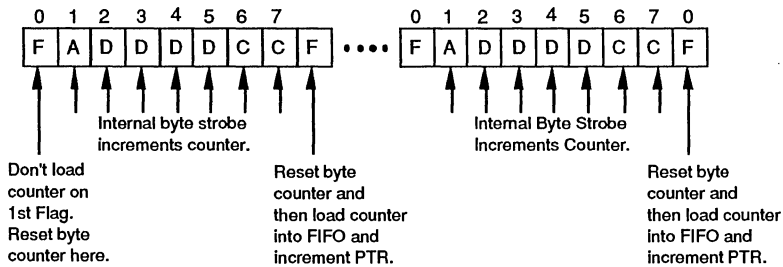


Figure 17. SDLC Byte Counting Detail

## PROGRAMMING

The ESCC contains write registers in each channel that are programmed by the system separately to configure the functional uniqueness of the channels.

In the ESCC, the data registers are directly addressed by selecting a High on the D//C pin. With all other registers (with the exception of WR0 and RR0), programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to WR0 and contains three bits that point to the selected register. The second write is the actual control word for the selected register, and if the second operation is read, the selected read register is accessed. All of the ESCC registers, including the data registers, may be accessed in this fashion. The pointer bits are automatically cleared after the read or write operation so that WR0 (or RR0) is addressed again.

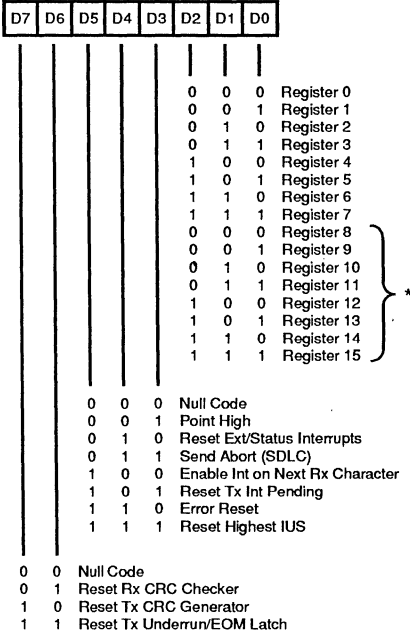
**Initialization.** The system program first issues a series of commands to initialize the basic mode of operation. This is followed by other commands to qualify conditions within the selected mode. For example, in the Asynchronous mode, character length, clock rate, number of stop bits, and even or odd parity should be set first. Then the interrupt mode is set, and finally, the receiver and transmitter are enabled.

**Write Registers.** The ESCC contains 16 write registers (17 counting the transmit buffer) in each channel. These write registers are programmed separately to configure the functional "personality" of the channels. There are two registers (WR2 and WR9) shared by the two channels that are accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits and reset commands. A new register, WR7', was added to the ESCC and may be written to if WR15 D0 is set. Figure 18 shows the format of each write register.

**Read Registers.** The ESCC contains ten read registers (eleven, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers (RR12 and RR13) are read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A only). RR6 and RR7 contain the information in the SDLC Frame Status FIFO, but is only read when WR15 D2 is set. If WR7' D6 is set, Write Registers WR3, WR4, WR5, WR7', and WR10 can be read as RR9, RR4, RR5, and RR14, respectively. Figure 19 shows the format of each Read register.

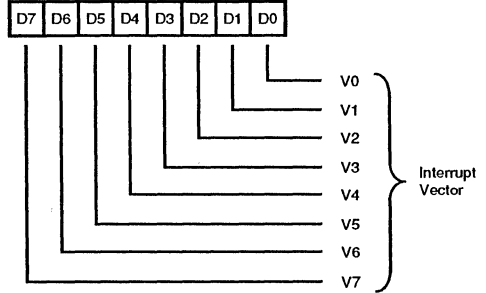
# CONTROL REGISTERS

Write Register 0 (non-multiplexed bus mode)

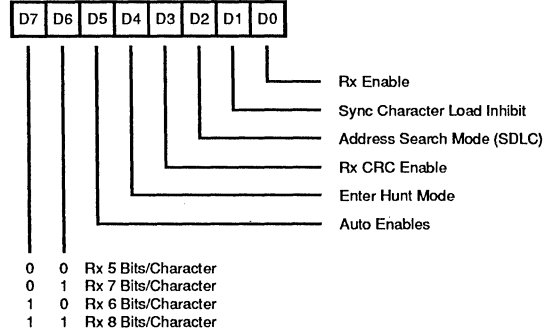


\* With Point High Command

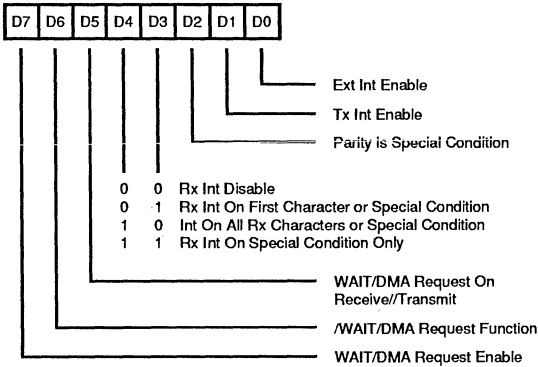
Write Register 2



Write Register 3



Write Register 1



Write Register 4

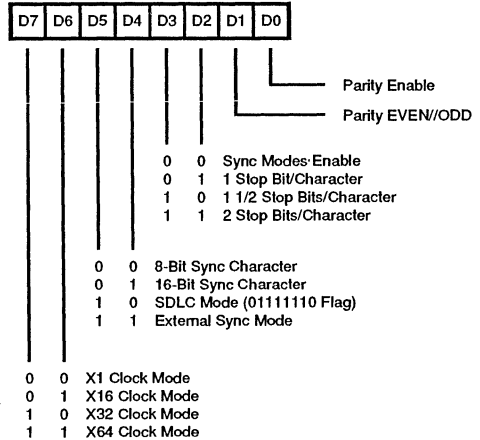


Figure 18. Write Register Bit Functions

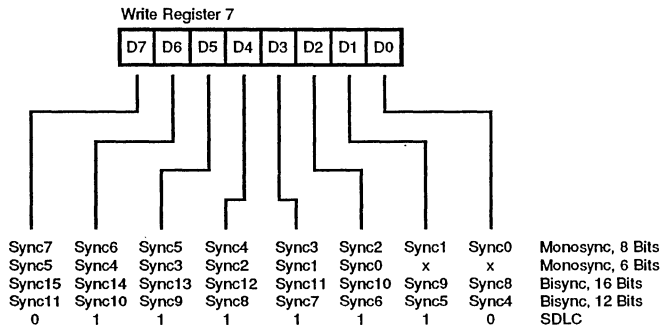
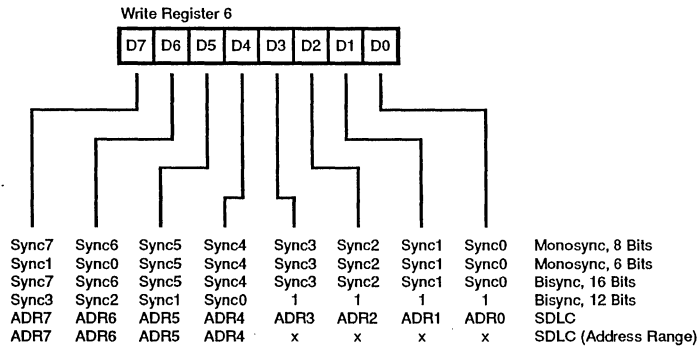
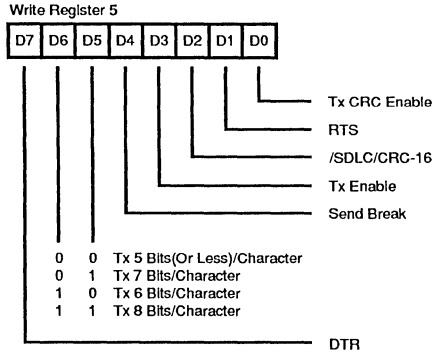
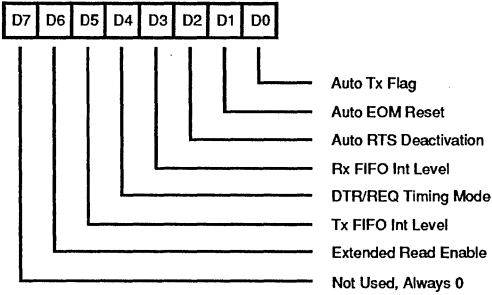


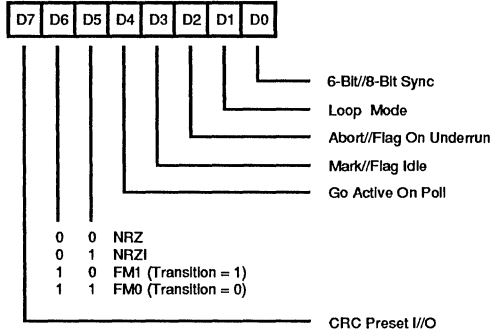
Figure 18. Write Register Bit Functions (Continued)

## CONTROL REGISTERS (Continued)

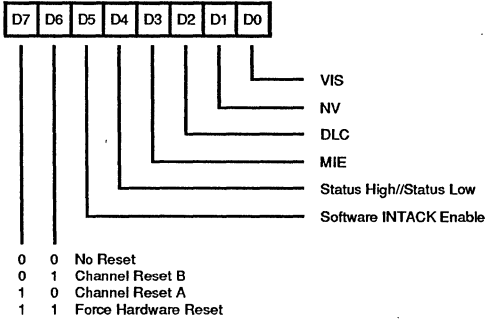
WR 7' Prime



Write Register 10



Write Register 9



Write Register 11

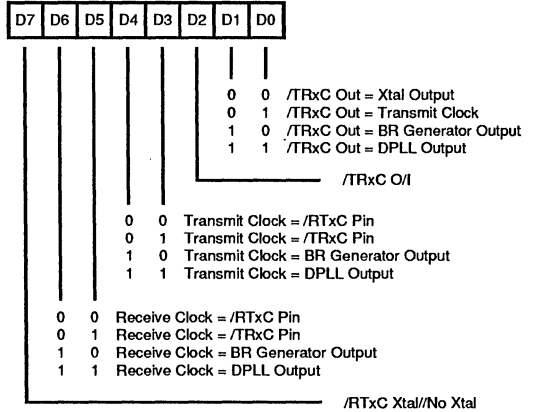
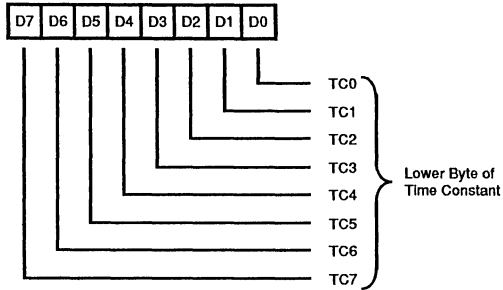
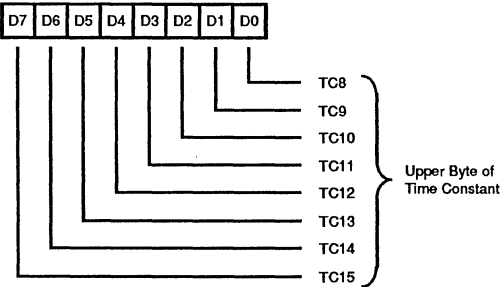


Figure 18. Write Register Bit Functions (Continued)

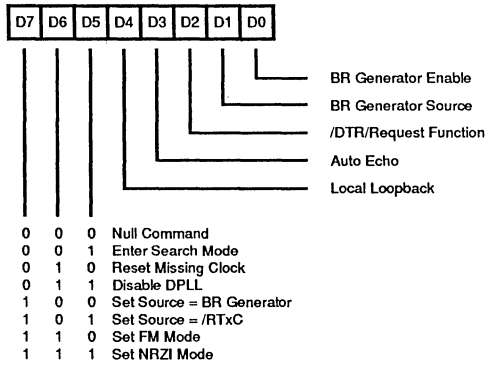
Write Register 12



Write Register 13



Write Register 14



Write Register 15

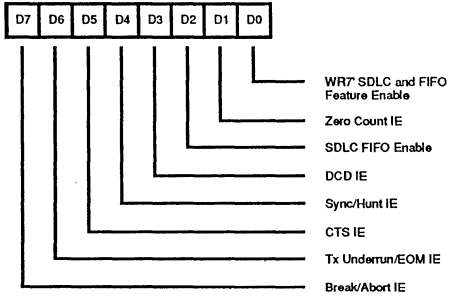
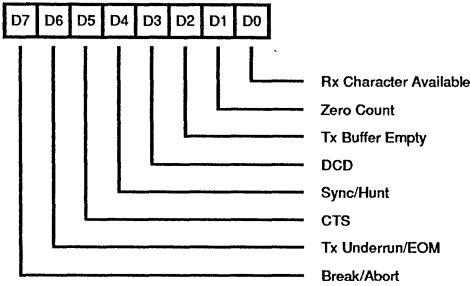


Figure 18. Write Register Bit Functions (Continued)

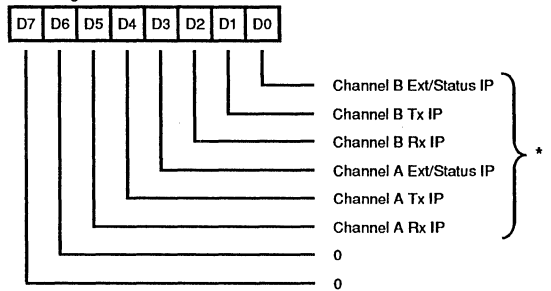


# CONTROL REGISTERS (Continued)

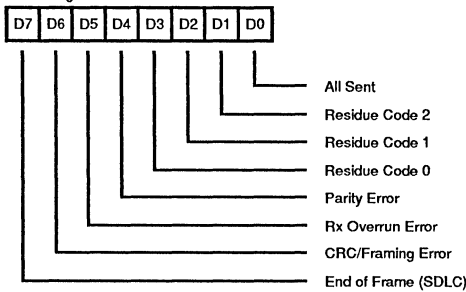
Read Register 0



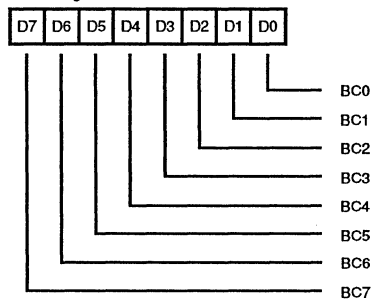
Read Register 3



Read Register 1



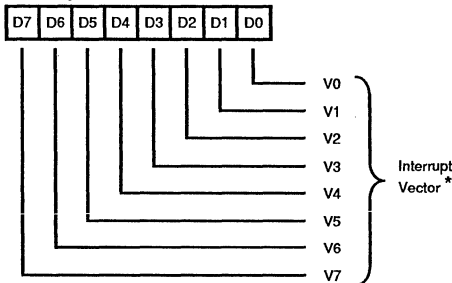
Read Register 6 \*



\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

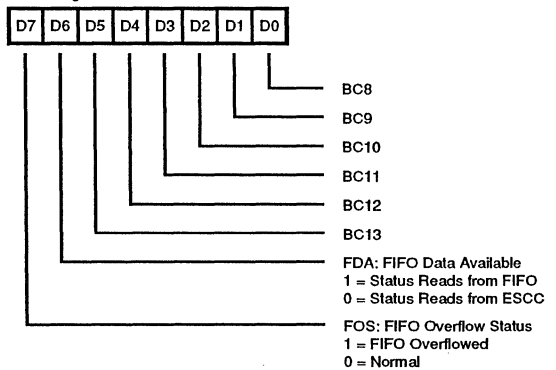
### SDLC FIFO Status and Byte Count (LSB)

Read Register 2



\* Modified In B Channel

Read Register 7 \*

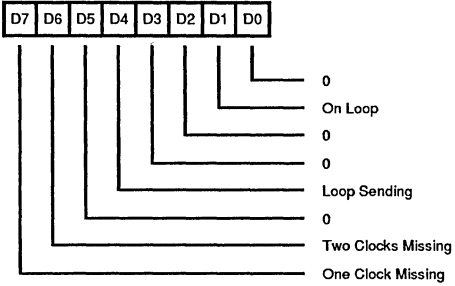


\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

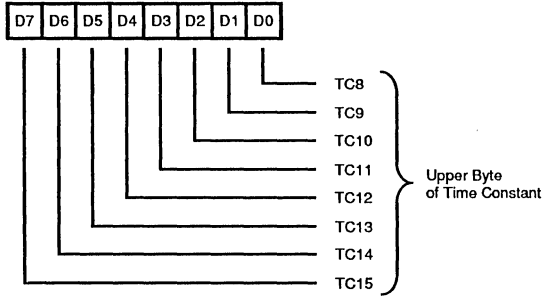
### SDLC FIFO Status and Byte Count (MSB)

Figure 19. Read Register Bit Functions

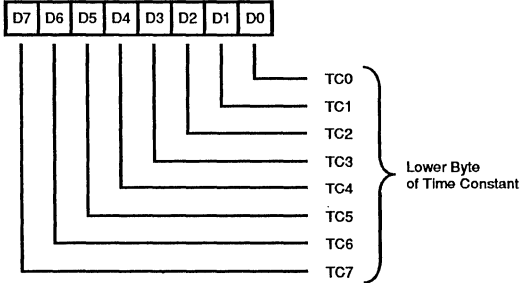
Read Register 10



Read Register 13



Read Register 12



Read Register 15

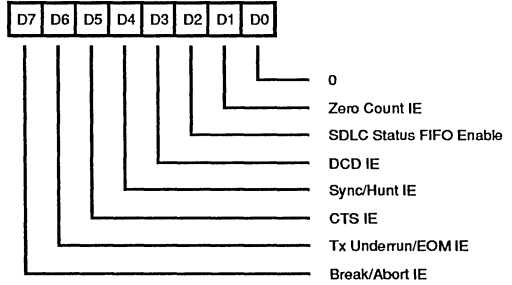


Figure 19. Read Register Bit Functions (Continued)

## Z85230 TIMING

The ESCC generates internal control signals from the  $/WR$  and  $/RD$  that are related to PCLK. Since PCLK has no phase relationship with  $/WR$  and  $/RD$ , the circuitry generating the internal control signals provides time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK. The recovery time applies only between bus transactions involving the ESCC. The recovery time required for proper operation is specified from the falling edge of  $/WR$  or  $/RD$  in the first transaction involving the ESCC to the falling edge of  $/WR$  or  $/RD$  in the second.

transaction involving the ESCC. This time must be at least 4 PCLKs regardless of which register or channel is being accessed.

**Read Cycle Timing.** Figure 20 illustrates Read cycle timing. Addresses on A//B and D//C and the status on  $/INTACK$  must remain stable throughout the cycle. If  $/CE$  falls after  $/RD$  falls, or if it rises before  $/RD$  rises, the effective  $/RD$  is shortened.

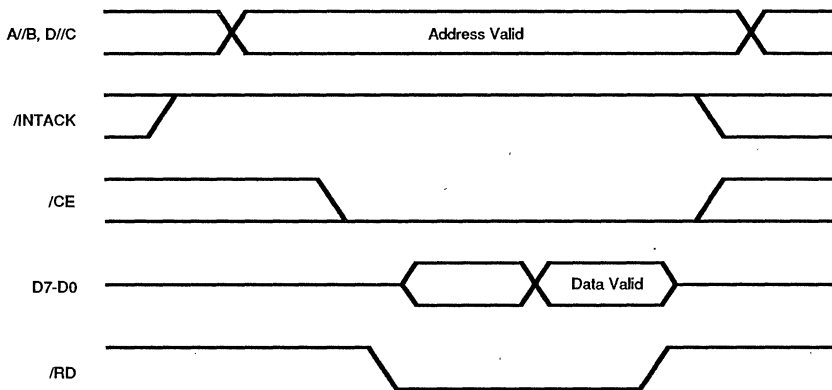


Figure 20. Read Cycle Timing

**Write Cycle Timing.** Figure 21 illustrates Write cycle timing. Addresses on A//B and D//C and the status on  $/INTACK$  must remain stable throughout the cycle. If  $/CE$  falls after  $/WR$  falls, or if it rises before  $/WR$  rises, the effective  $/WR$  is shortened. Because many popular CPUs do not guaran-

tee that the databus is valid when  $/WR$  is driven Low, the databus timing requirements of the ESCC have been modified so that the databus does not have to be valid when the  $/WR$  pin goes Low. See AC Characteristic #29 for details.

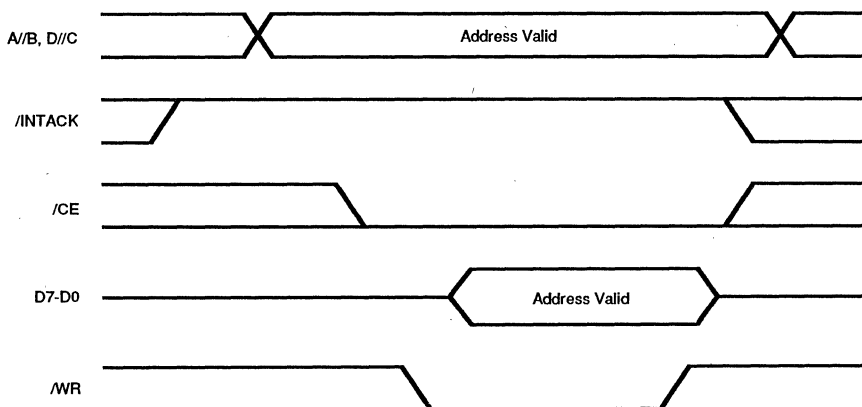
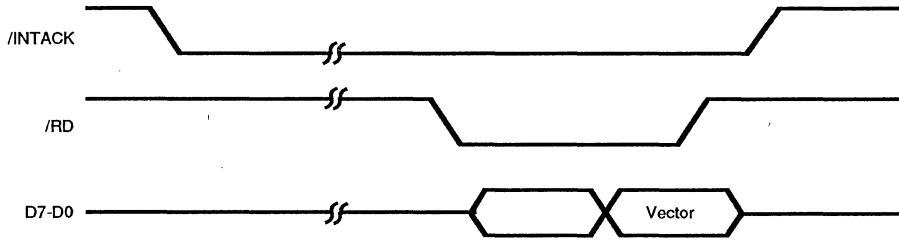


Figure 21. Write Cycle Timing

**Interrupt Acknowledge Cycle Timing.** Figure 22 illustrates Interrupt Acknowledge cycle timing. Between the time /INTACK goes Low and the falling edge of /RD, the internal and external IEI/IEO daisy chains settle. If there is an interrupt pending in the ESCC and IEI is High when /RD falls, the Acknowledge cycle is intended for the ESCC. In this case, the ESCC may be programmed to respond to /RD Low by placing its interrupt vector on D7-D0. It then

sets the appropriate Interrupt-Under-Service latch internally. If the external daisy chain is not used, then AC parameter #38 is required to settle the interrupt priority daisy chain internal to the ESCC. If the external daisy chain is used, the user should follow the equation in AC Characteristics Note 5 for calculating the required daisy-chain settle time.



**Figure 22. Interrupt Acknowledge Cycle Timing**

## OTHER ZILOG DATA COMMUNICATIONS PRODUCTS

### SIO Family

Z84C40 SIO	Dual channel multiprotocol USART.
Z84C13 IPC	Z80 CPU with integrated SIO, CTC and WDT.
Z84C15 IPC	Z80 CPU with integrated SIO, CTC, WDT and PIO.

### SCC Family

Z08530 SCC	NMOS SCC low cost with speeds up to 8 MHz.
Z08030 SCC	NMOS SCC for multiplexed buses.
Z85C30 SCC	CMOS SCC at speeds up to 16 MHz. NMOS compatible.
Z80C30 SCC	CMOS SCC for multiplexed buses.
Z16C35 ISCC	SCC with 4 channel DMA and advanced CPU interface.
Z80181 SAC	Z180 CPU with integrated single channel SCC.

### USC Family

Z16C30 USC	Dual channel high performance multi-protocol data communications up to 10 Megabits/second.
Z16C33 MUSC	Single channel USC w/ ISDN Time Slot Assigner.
Z16C31 IUSC	MUSC with high performance dual channel DMA.
Z16C50 DDPLL	Dual channel DPLL cell from the USC.

## ABSOLUTE MAXIMUM RATINGS

$V_{CC}$ Supply Voltage range	-0.3V to +7.0V
Voltages on all pins with respect to GND	
Operating Ambient Temperature	-0.3V to $V_{CC} + 0.3V$
Storage Temperature	See Ordering Information
	-65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin.

Standard conditions are as follows:

- $+4.50\text{ V} \leq V_{CC} \leq +5.50\text{ V}$
- $\text{GND} = 0\text{ V}$
- $T_A$  as specified in Ordering Information

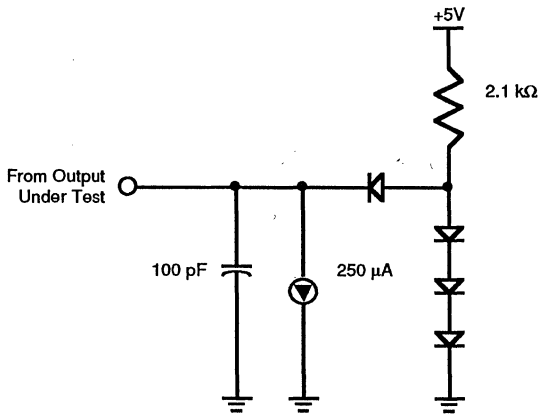


Figure 23. Standard Test Load

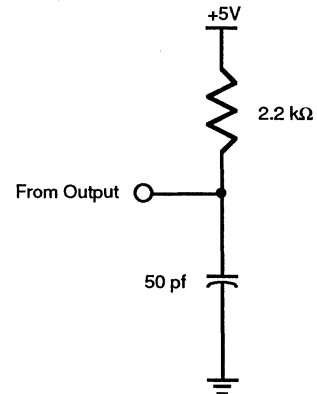


Figure 24. Open-Drain Test Load

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit	Test Condition
$C_{IN}$	Input Capacitance		10	pF	Unmeasured pins returned to Ground.
$C_{OUT}$	Output Capacitance		15	pF	
$C_{IO}$	Bidirectional Capacitance		20	pF	

**Note:**

$f = 1\text{ MHz}$ , over specified temperature range.

## MISCELLANEOUS

Gate Count - 11,000

## DC CHARACTERISTICS

Z85230

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC}+0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6\text{mA}$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$			V	$I_{OH} = -250\mu\text{A}$
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = 2.0\text{mA}$
$I_{IL}$	Input Leakage			$\pm 10.0$	$\mu\text{A}$	$0.4 < V_{IN} < +2.4\text{V}$
$I_{OL}$	Output Leakage			$\pm 10.0$	$\mu\text{A}$	$0.4 < V_{OUT} < +2.4\text{V}$
$I_{CC1}$	$V_{CC}$ Supply Current		4	10 (8.5 MHz)	mA	$V_{CC}=5\text{V}$ $V_{IH}=4.8$ $V_{IL}=0.2\text{V}$ Crystal Oscillators off
			5	12 (10 MHz)	mA	
			7	15 (16 MHz)	mA	
			9	20 (20 MHz)	mA	
$I_{CC(\text{OSC})}$	Crystal OSC Current		6		mA	Current for each osc. in addition to $I_{CC1}$

### Notes:

[1]  $V_{CC} = 5\text{V} \pm 10\%$  unless otherwise specified, over specified temperature range.

[2] Typical  $I_{CC}$  was measured with oscillator off.

[3] No  $I_{CC(\text{osc})}$  max is specified due to dependency on the external circuit.

**AC CHARACTERISTICS**  
Z85230 Timing Diagrams

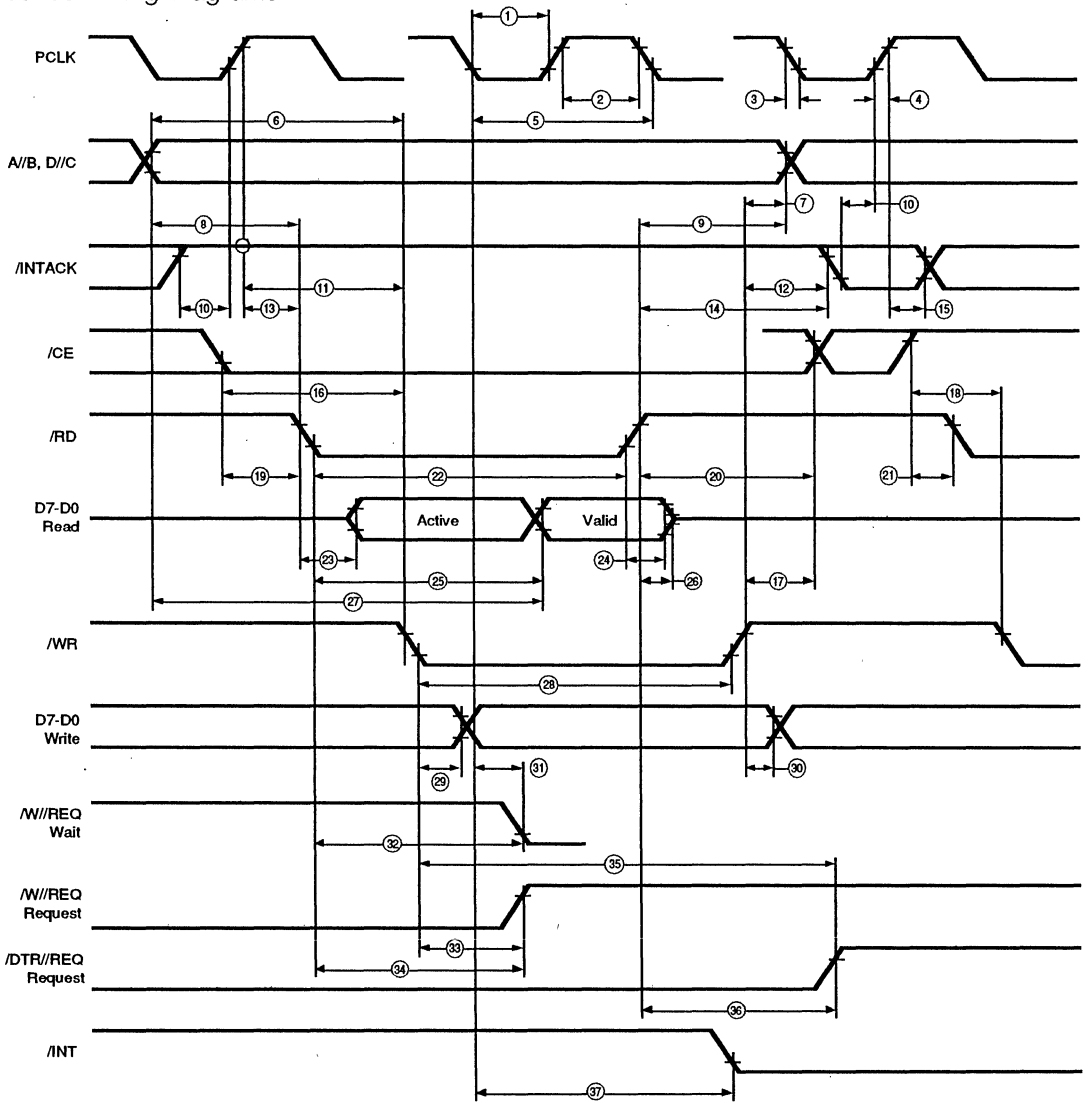


Figure 25. Read and Write Timing Diagram

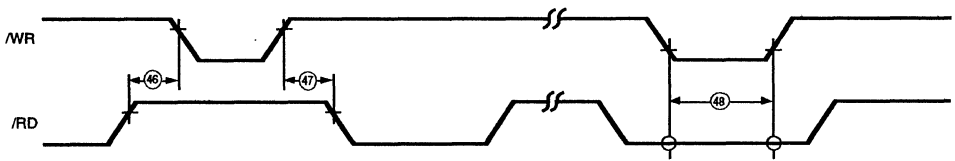


Figure 26. Reset Timing Diagram

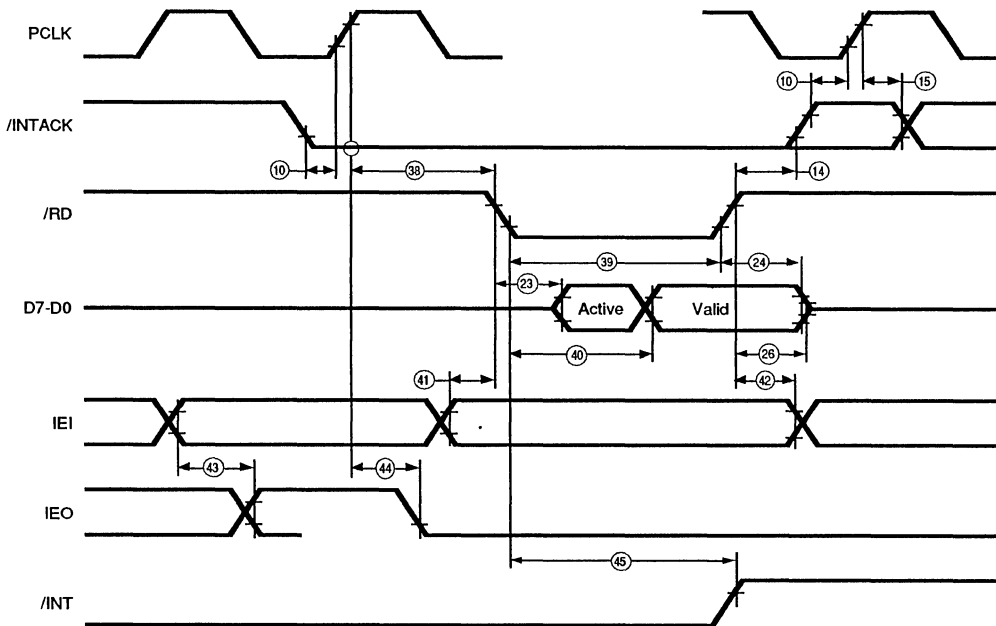


Figure 27. Interrupt Acknowledge Timing Diagram

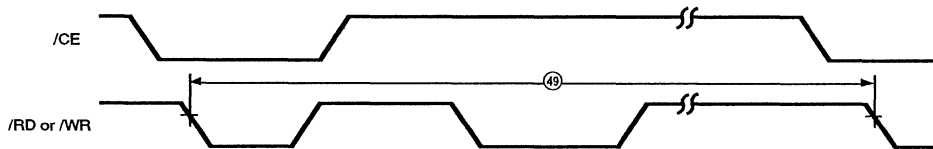


Figure 28. Cycle Timing Diagram

## AC CHARACTERISTICS

### Z85230 Read and Write Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16 MHz		20 MHz		Notes
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TwPCL	PCLK Low Width	45	1000	40	1000	26	1000	22	1000	
2	TwPCh	PCLK High Width	45	1000	40	1000	26	1000	22	1000	
3	TfPC	PCLK Fall Time		10		10		5		5	
4	TrPC	PCLK Rise Time		10		10		5		5	
5	TcPC	PCLK Cycle Time	118	2000	100	2000	61	2000	50	2000	
6	TsA(WR)	Address to /WR Fall Setup Time	66		50		35		30		
7	ThA(WR)	Address to /WR Rise Hold Time	0		0		0		0		
8	TsA(RD)	Address to /RD Fall Setup Time	66		50		35		30		
9	ThA(RD)	Address to /RD Rise Hold Time	0		0		0		0		
10	TsIA(PC)	/INTACK to PCLK Rise Setup Time	20		20		15		15		



## AC CHARACTERISTICS

### Z85230 Read and Write Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16 MHz		20 MHz		Notes
			Min	Max	Min	Max	Min	Max	Min	Max	
11	TsIAi(WR)	/INTACK To /WR Fall Setup Time	140		130		70		65		[1]
12	ThIA(WR)	/INTACK To /WR Rise Hold Time	0		0		0		0		
13	TsIAi(RD)	/INTACK To /RD Fall Setup Time	140		130		70		65		[1]
14	ThIA(RD)	/INTACK To /RD Rise Hold Time	0		0		0		0		
15	ThIA(PC)	/INTACK To PCLK Rise Hold Time	38		30		15		15		
16	TsCEI(WR)	/CE Low To /WR Fall Setup Time	0		0		0		0		
17	ThCE(WR)	/CE To /WR Rise Hold Time	0		0		0		0		
18	TsCEh(WR)	/CE High To /WR Fall Setup Time	58		50		30		25		
19	TsCEI(RD)	/CE Low To /RD Fall Setup Time	0		0		0		0		[1]
20	ThCE(RD)	/CE To /RD Rise Hold Time	0		0		0		0		[1]
21	TsCEh(RD)	/CE High To /RD Fall Setup Time	58		50		30		25		[1]
22	TwRDI	/RD Low Width	145		125		70		65		[1]
23	TdRD(DRA)	/RD Fall To Read Data Active Delay	0		0		0		0		
24	TdRDr(DR)	/RD Rise To Data Not Valid Delay	0		0		0		0		
25	TdRD(DR)	/RD Fall To Read Data Valid Delay		135		120		70		65	
26	TdRD(DRz)	/RD Rise To Read Data Float Delay		38		35		30		30	
27	TdA(DR)	Addr To Read Data Valid Delay		210		180		100		90	
28	TwWRI	/WR Low Width	145		125		75		65		
29	TdWR(DW)	/WR Fall To Write Data Valid Delay		20		20		20		20	
30	ThDW(WR)	Write Data To /WR Rise Hold Time	0		0		0		0		
31	TdWR(W)	/WR Fall To Wait Valid Delay		168		100		50		50	[4]
32	TdRD(W)	/RD Fall To Wait Valid Delay		168		100		50		50	[4]
33	TdWRr(REQ)	/WR Fall To /W//REQ Not Valid Delay		168		120		70		65	
34	TdRDl(REQ)	/RD Fall To /W//REQ Not Valid Delay		168		120		70		65	[6]
35a	TdWRr(REQ)	/WR Fall To /DTR//REQ Not Valid		4TcPc		4TcPc		4TcPc		4TcPc	
35b	TdWRr(REQ)	/WR Fall To /DTR//REQ Not Valid		168		100		70		65	[6]
36	TdRDr(REQ)	/RD Rise To /DTR//REQ Not Valid Delay		NA		NA		NA		NA	
37	TdPC(INT)	PCLK Fall To /INT Valid Delay		500		320		175		160	
38	TdIAi(RD)	/INTACK To /RD Fall (Ack) Delay	145		90		50		45		[5]
39	TwRDA	/RD (Acknowledge) Width	145		125		75		65		
40	TdRDA(DR)	/RD Fall(Ack) To Read Data Valid Delay		135		120		70		60	
41	TsIEI(RDA)	IEI To /RD Fall (Ack) Setup Time	95		95		50		45		
42	ThIEI(RDA)	IEI To /RD Rise (Ack) Hold Time	0		0		0		0		
43	TdIEI(IEO)	IEI To IEO Delay Time		95		90		45		40	
44	TdPC(IEO)	PCLK Rise To IEO Delay		195		175		80		80	
45	TdRDA(INT)	/RD Fall To /INT Inactive Delay		480		320		200		180	[4]
46	TdRD(WRQ)	/RD Rise To /WR Fall Delay For No Reset	15		15		10		10		
47	TdWRQ(RD)	/WR Rise To /RD Fall Delay For No Reset	15		15		10		10		
48	TwRES	/WR And /RD Low For Reset	145		100		75		65		
49	Trc	Valid Access Recovery Time	4TcPc		4TcPc		4TcPc		4TcPc		[3]

#### Notes:

- [1] Parameter does not apply to Interrupt Acknowledge transactions.
- [3] Parameter applies only between transactions involving the ESCC.
- [4] Open-drain output, measured with open-drain test load.
- [5] Parameter is system dependent. For any ESCC in the daisy chain, TdIAi(RD) must be greater than the sum of TdPC(IEO) for the highest priority device in the daisy chain. TsIEI(RDA) for the ESCC and TdIEI(IEO) for each device separating them in the daisy chain.
- [6] Parameter applies to enhanced Request mode only (WR7' D4=1).

**AC CHARACTERISTICS**  
Z85230 General Timing Diagram

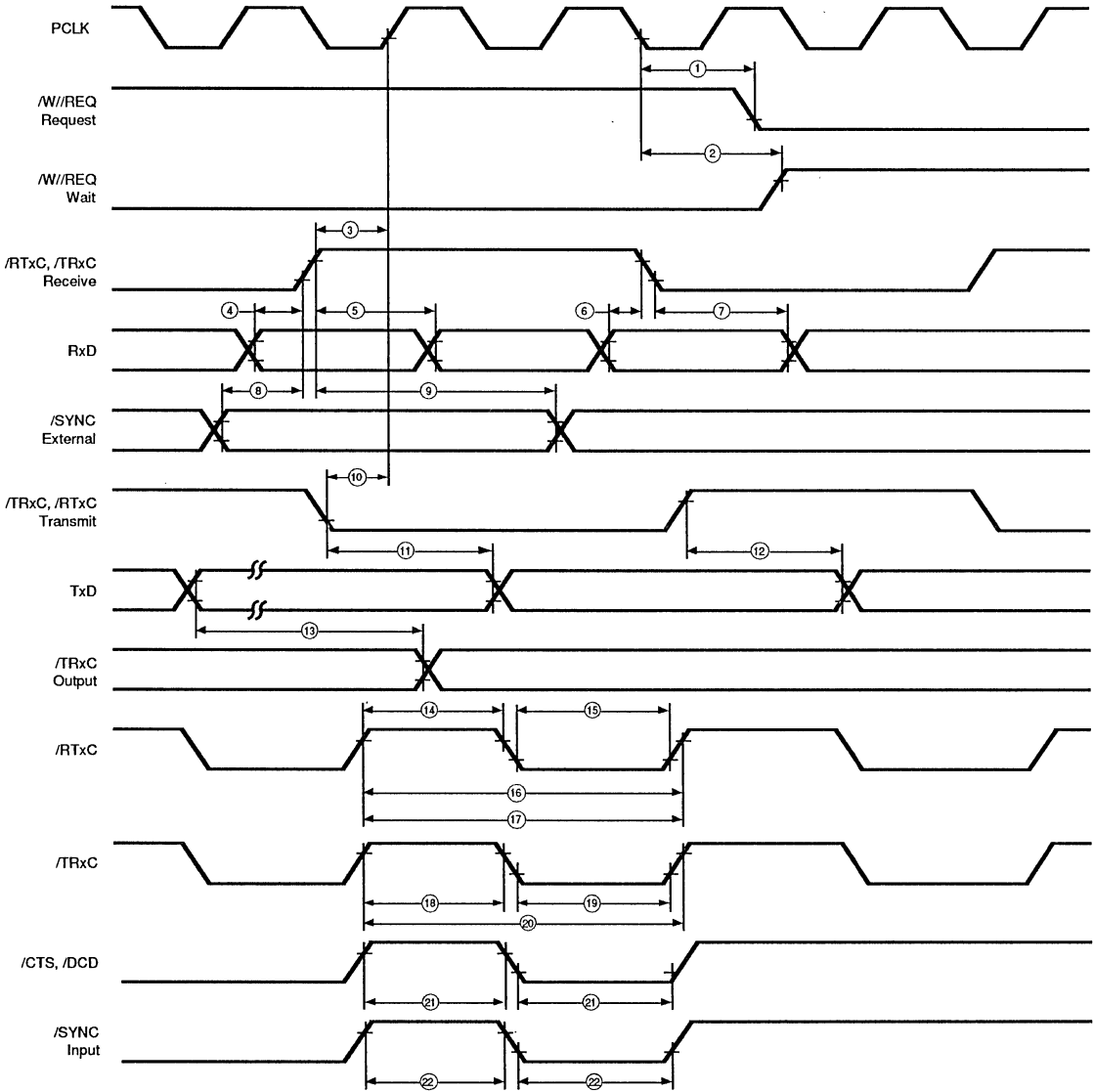


Figure 29. General Timing Diagram

## AC CHARACTERISTICS

### Z85230 General Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16 MHz		20 MHz		Notes
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TdPC(REQ)	/PCLK to W/REQ Valid		250		200		80		70	
2	TdPC(W)	/PCLK to Wait Inactive		350		300		180		170	
3	TsRXC(PC)	/RxC to /PCLK Setup Time	NA		NA		NA		NA		[1,4]
4	TsRXD(RXCr)	RxD to /RxC Setup Time		0		0		0		0	[1]
5	ThRXD(RXCr)	RxD to /RxC Hold Time	150		125		50		45		[1]
6	TsRXD(RXCf)	RxD to /RxC Setup Time	0		0		0		0		[1,5]
7	ThRXD(RXCf)	RxD to /RxC Hold Time	150		125		50		45		[1,5]
8	TsSY(RXC)	/SYNC to /RxC Setup Time	-200		-150		-100		-90		[1]
9	ThSY(RXC)	/SYNC to /RxC Hold Time	5TcPc		5TcPc		5TcPc		5TcPc		[1]
10	TsTXC(PC)	/TxC to /PCLK Setup Time	NA		NA		NA		NA		[2,4]
11	TdTXC(TXD)	/TxC to TxD Delay		190		150		80		70	[2]
12	TdTxCr(TXD)	/TxC to TxD Delay		190		150		80		70	[2,5]
13	TdTXD(TRX)	TxD to TRxC Delay		200		140		80		70	
14	TwRTXh	RTxC High Width	130		120		80		70		[6]
15	TwRTXI	TRxC Low Width	130		120		80		70		[6]
16a	TcRTX	RTxC Cycle Time	472		400		244		200		[6,7]
16b	TxRX(DPLL)	DPLL Cycle Time Min	50		50		31		31		[7,8]
17	TcRTXX	Crystal Osc. Period	125	1000	100	1000	61	1000	61	1000	[3]
18	TwTRXh	TRxC High Width	130		120		80		70		[6]
19	TwTRXI	TRxC Low Width	130		120		80		70		[6]
20	TcTRX	TRxC Cycle Time	472		400		244		200		[6,7]
21	TwEXT	DCD or CTS Pulse Width	200		120		70		60		
22	TwSY	SYNC Pulse Width	200		120		70		60		

#### Notes:

- [1] RxC is /RTxC or /TRxC, whichever is supplying the receive clock.
- [2] TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.
- [3] Both /RTxC and /SYNC have 30 pF capacitors to ground connected to them.
- [4] Synchronization of RxC to PCLK is eliminated in divide by four operation.
- [5] Parameter applies only to FM encoding/decoding.
- [6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator timing requirements are identical to case PCLK requirements.
- [7] The maximum receive or transmit data rate is 1/4 PCLK.
- [8] Applies to DPLL clock source only. Maximum data rate of 1/4 PCLK still applies. DPLL clock should have a 50% duty cycle.

**AC CHARACTERISTICS**  
Z85230 System Timing Diagram

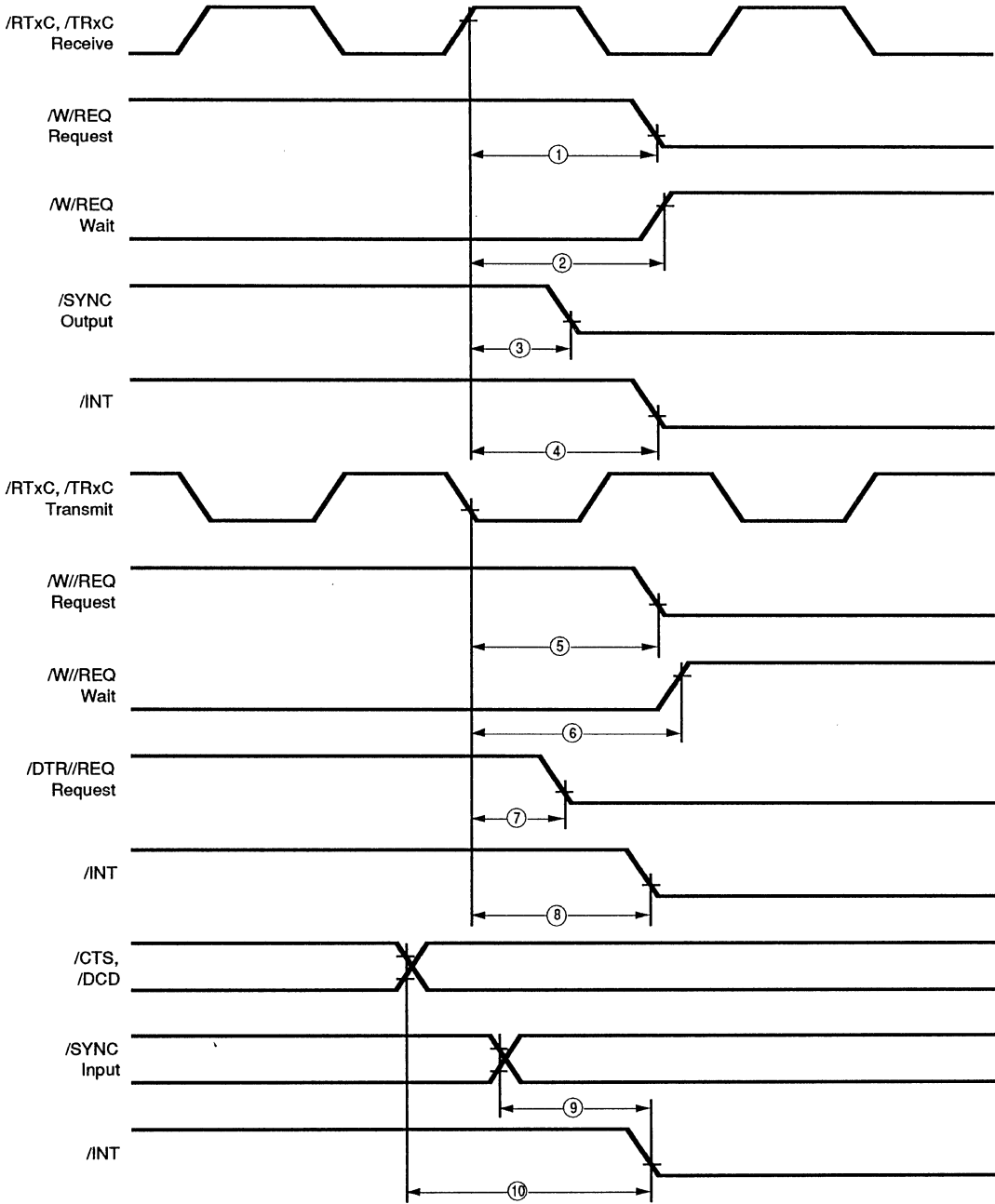


Figure 30. Z85230 System Timing

## AC CHARACTERISTICS

### Z85230 System Timing Table

No	Symbol	Parameter	8.5 MHz		10 MHz		16 MHz		20 MHz		Notes [4]
			Min	Max	Min	Max	Min	Max	Min	Max	
1	TdRXC(REQ)	/RxC to /W//REQ Valid	13	17	13	17	13	17	13	18	[2]
2	TdRXC(W)	/RxC to /Wait Inactive	13	17	13	17	13	17	13	18	[1,2]
3	TdRXC(SY)	/RxC to /SYNC Valid	4	7	4	7	4	7	4	8	[2]
4	TdRXC(INT)	/RxC to /INT Valid	15	21	15	21	15	21	15	22	[1,2]
5	TdTXC(REQ)	/TxC to /W//REQ Valid	8	11	8	11	8	11	8	12	[3]
6	TdTXC(W)	/TxC to /Wait Inactive	8	14	8	14	8	14	8	15	[1,3]
7	TdTXC(DRQ)	/TxC to /DTR//REQ Valid	7	10	7	10	7	10	7	11	[3]
8	TdTXC(INT)	/TxC to /INT Valid	9	13	9	13	9	13	9	14	[1,3]
9	TdSY(INT)	/SYNC to /INT Valid	2	6	2	6	2	6	2	7	[1]
10	TdEXT(INT)	/DCD or /CTS to /INT Valid	3	8	3	8	3	8	3	9	[1]

#### Notes:

[1] Open drain-output, measured with open-drain test load.

[2] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[3] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[4] Units equal to TcPc.



## Z80230

### Z-BUS™ ESCC™ ENHANCED SERIAL COMMUNICATION CONTROLLER

#### FEATURES

- Deeper Data FIFOs
  - 4-byte transmit FIFO
  - 8-byte receive FIFO
- Programmable FIFO interrupt levels provide flexible interrupt response
- Pin and function compatible to CMOS and NMOS Z80C30 SCC
- Many improvements to support SDLC/HDLC transfers:
  - Deactivation of /RTS pin after closing flag
  - Automatic transmission of the opening flag
  - Automatic reset of Tx Underrun/EOM latch
  - Complete CRC reception
  - TxD pin automatically forced High with NRZI encoding when using mark idle.
  - Receive FIFO automatically unlocked for special receive interrupts when using the SDLC status FIFO.
  - Back-to-back frame transmission simplified
- Easier interface to popular CPUs
- Fast speeds:
  - 10.0 MHz for data rates up to 2.5 Mbit/sec.
  - 16.384 MHz for data rates up to 4.096 Mbit/sec.
- Improved SDLC frame status FIFO
- Low power CMOS
- New programmable features added with Write Register 7'
- Write registers: WR3, WR4, WR5, and WR10 are now readable
- Read Register 0 latched during access
- Software Interrupt Acknowledge Mode
- DPLL counter output available as jitter-free clock source
- /DTR//REQ pin deactivation time reduced
- Two independent full-duplex channels, each with a crystal oscillator, baud rate generator, and digital phase-locked loop.
- Multi-protocol operation under program control
- Asynchronous mode with five to eight bits, and one, one-and-one-half, or two stop bits per character; programmable clock factor; break detection and generation; parity, overrun, and framing error detection.
- Synchronous mode with internal or external character synchronization on one or two synchronous characters and CRC generation and checking with programmable CRC preset values.

#### GENERAL DESCRIPTION

The Zilog Enhanced Serial Communications Controller, Z80230 Z-Bus™ ESCC, is a pin and software compatible CMOS member of the SCC family (The SCC was introduced by Zilog in 1981.). The ESCC is a dual-channel, full-duplex datacommunications controller capable of supporting a wide range of popular protocols. The ESCC is built from Zilog's industry standard SCC core and is compatible with

designs using Zilog's SCC to receive and transmit data. It has many improvements that significantly reduce CPU overhead. The addition of a 4-byte transmit FIFO and an 8-byte receive FIFO significantly reduces the overhead required to provide data to, and get data from, the transmitters and receivers.

---

## GENERAL DESCRIPTION (Continued)

The ESCC also has many features that improve packet handling in SDLC mode. The ESCC will automatically: transmit a flag before the data, reset the Tx Underrun/EOM latch, force the TxD pin High at the appropriate time when using NRZI encoding, deassert the /RTS pin after the closing flag, and better handle ABORTed frames when using the 10x19 status FIFO. The combination of these features along with the deeper data FIFOs significantly simplifies SDLC driver software.

The CPU hardware interface has been simplified by relieving the databus setup time requirement and supporting the software generation of the interrupt acknowledge signal (/INTACK). These changes allow an interface with less external logic to many microprocessor families while maintaining compatibility with existing designs. I/O handling of the ESCC is improved over the SCC with faster response of the /INT and /DTR//REQ pins.

The many enhancements added to the ESCC permits a system design that increases overall system performance with better data handling and less interface logic (Figure 1).

### Notes:

All Signals with a preceding front slash, '/', are active Low, e.g.: B//W (WORD is active Low); /B/W (BYTE is active Low, only).

Power connections follow conventional descriptions below:

Connection	Circuit	Device
Power Ground	$V_{CC}$ GND	$V_{DD}$ $V_{SS}$

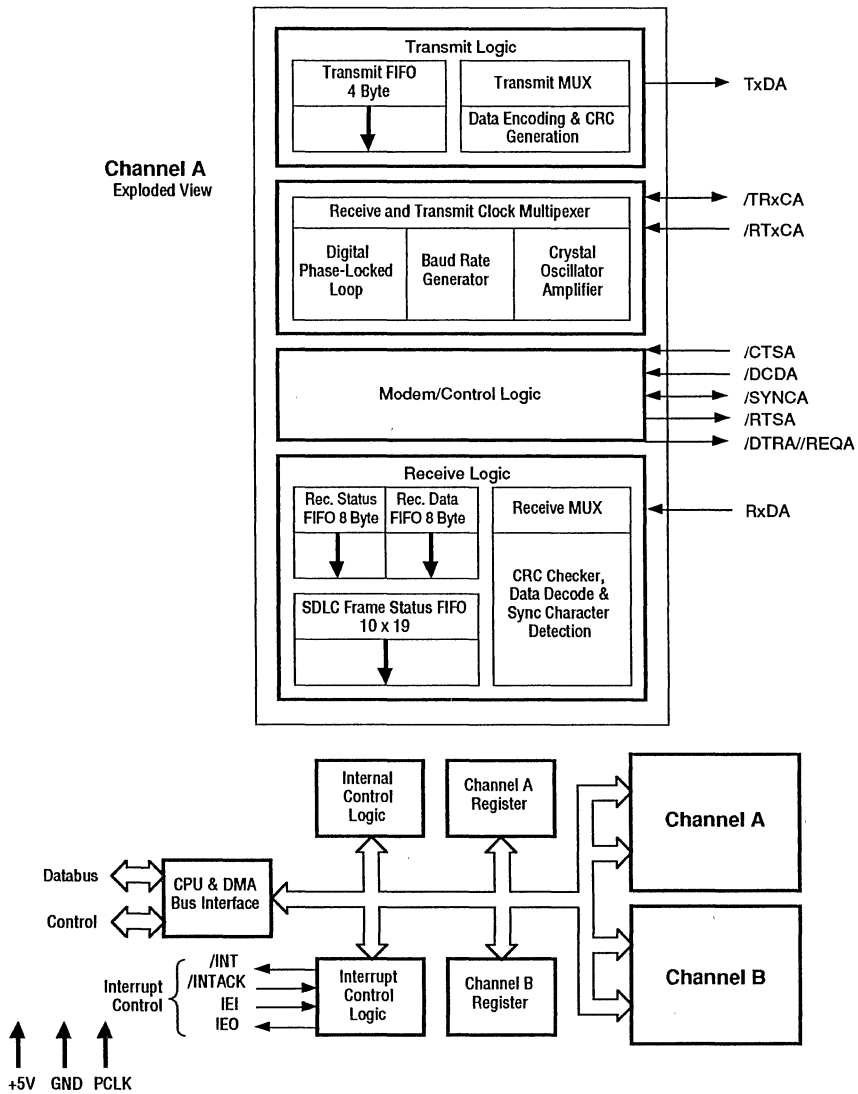


Figure 1. ESCC Block Diagram



## PIN DESCRIPTIONS

The following section describes the Z80230 pin functions. Figures 2 and 3 detail the pin assignments for the 40-pin DIP and 44-pin PLCC packages. The Z80230 ESCC is socket compatible with the Zilog Z8030 and Z80230 as the

pin electrical characteristics and pin assignments are the same. Any unused input pins should be pulled up to the +5V supply.

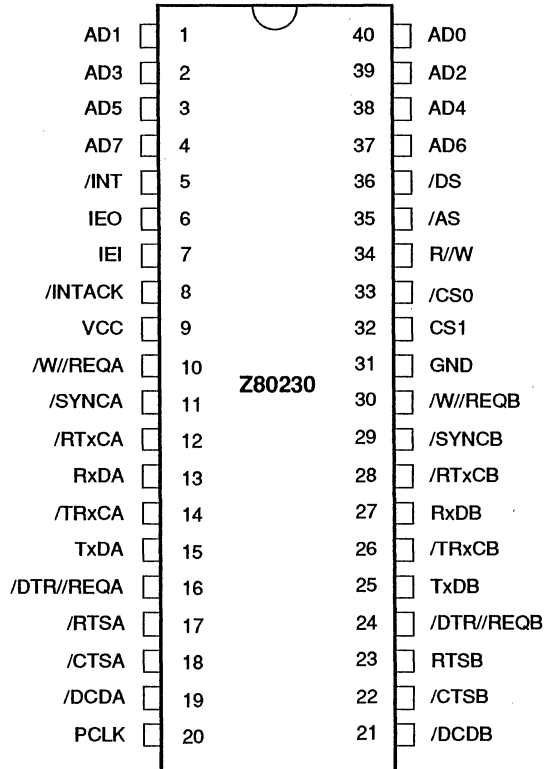


Figure 2. Z80230 DIP Pin Assignments

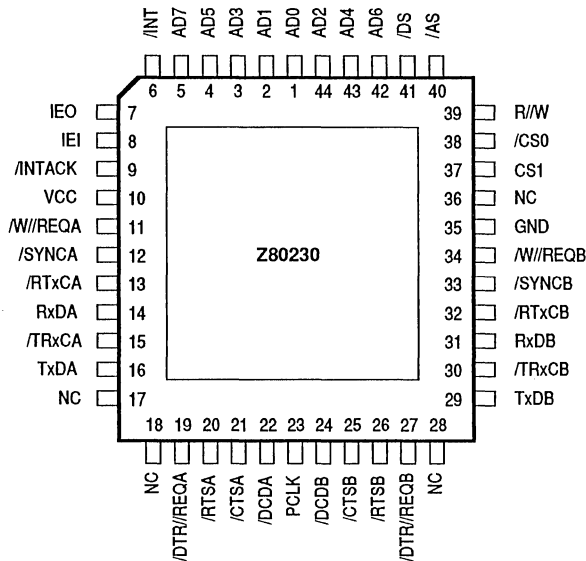


Figure 3. Z80230 PLCC Pin Assignments

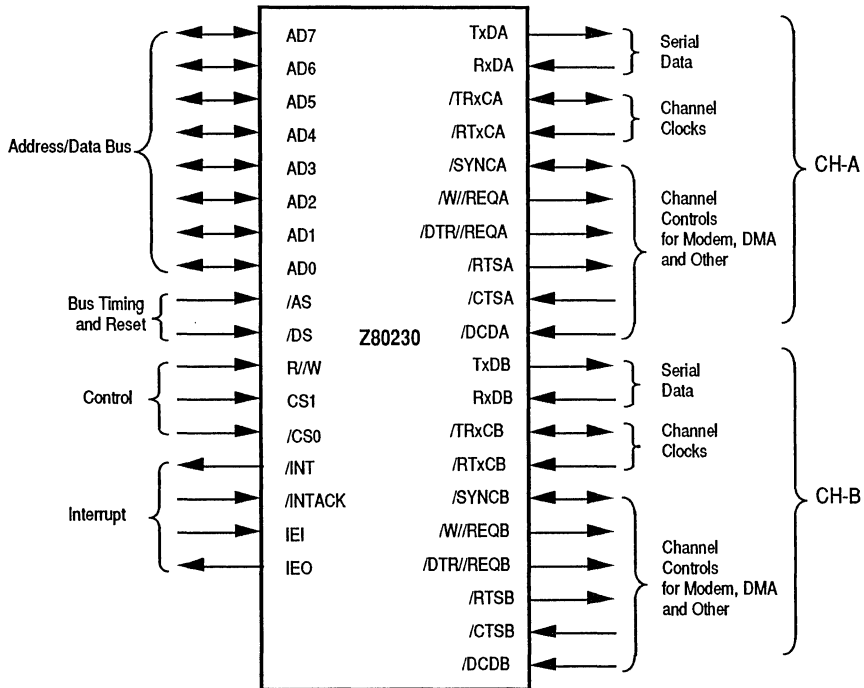


Figure 4. Z80230 Pin Functions

---

## PIN DESCRIPTIONS

**/CTSA, /CTSB.** *Clear To Send* (inputs, active Low). These pins function as transmitter enables if they are programmed for Auto Enables (WR3, D5=1). A Low on the inputs enables the respective transmitters. If not programmed as Auto Enables, they may be used as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The ESCC detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

**/DCDA, /DCDB.** *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if they are programmed for Auto Enables (WR3, D5=1); otherwise they are used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow rise-time signals. The ESCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

**/RTSA, /RTSB.** *Request To Send* (outputs, active Low). The /RTS pins can be used as general-purpose outputs or with the Auto Enables feature. When used with Auto Enables ON (WR3, D5=1) in asynchronous mode, the /RTS pin goes High after the transmitter is empty. When Auto Enable is OFF, the /RTS pins can be used as general-purpose outputs and they strictly follow the inverse state of the RTS bit (WR5 bit D1).

In SDLC mode, the /RTS pins can be programmed to be deasserted when the closing flag of the message clears the TxD pin if WR7' D2 is set.

**/SYNCA, /SYNCB.** *Synchronization* (inputs or outputs, active Low). These pins can act either as inputs, outputs, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to CTS and DCD. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode, /SYNC must be driven Low for two receive clock cycles after the last bit in the synchronous character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of /SYNC.

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which synchronous condition is latched. These outputs are active each time a synchronization pattern is recognized (regardless of character boundaries). In SDLC mode, the pins act as outputs and are valid on receipt of a flag. The /SYNC pins switch from input to output when monosync, bisync, or SDLC is programmed in WR4 and sync modes are enabled.

**/DTR//REQA, /DTR//REQB.** *Data Terminal Ready/Request* (outputs, active Low). These pins are programmed (WR14, D2) to serve either as general-purpose outputs or as DMA Request lines. When programmed for the DTR function (WR14, D2=0), these outputs follow the state programmed into the DTR bit of Write Register 5 (WR5, D7). When programmed for Request mode (WR14, D2=1), these pins serve as DMA Requests for the transmitter.

When used as DMA request lines, the timing for the deactivation Request can be programmed in the added register Write Register 7' (WR7') bit D4. If this bit is set, the /DTR//Request pin will be deactivated with the same timing as the /W//REQ pin. If WR7' D4 is reset, the deactivation timing of /DTR//Req pin will be the same as in the Z85C30.

**W//REQA, W//REQB.** *Wait/Request* (outputs, open drain when programmed for Wait function, driven High or Low when programmed for Ready function). These dual-purpose outputs may be programmed as Request lines for a DMA controller or as Wait lines which synchronize the CPU to the ESCC data rate. The reset state is Wait.

**RxDA, RxDB.** *Receive Data* (inputs, active High). These input signals receive serial data at standard TTL levels.

**/RTxCA, /RTxCB.** *Receive/Transmit Clocks* (inputs, active Low). These pins can be programmed to several modes of operation. In each channel, RTxC may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the digital phase-locked loop. These pins can also be programmed for use with the respective SYNC pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

---

**TxDA, TxDB.** *Transmit Data* (outputs, active High). These output signals transmit serial data at standard TTL levels.

**/TRxCA, /TRxCB.** *Transmit/Receive Clocks* (inputs or outputs, active Low). These pins can be programmed in several different modes of operation. TRxC may supply the receive clock or the transmit clock in the input mode or supply the output of the digital phase-locked loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

**PCLK.** *Clock* (input). This is the master ESCC clock used to synchronize internal signals. PCLK is a TTL level signal. PCLK is not required to have any phase relationship with the master system clock.

**IEI.** *Interrupt Enable In* (input, active High). IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt driven device. A High IEI indicates that no other higher priority device has an interrupt under service or is requesting an interrupt.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing the ESCC interrupt, or the ESCC is not requesting an interrupt (Interrupt Acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

**/INT.** *Interrupt* (output, open drain, active Low). This signal is activated when the ESCC requests an interrupt. Note that /INT is an open drain output.

**/INTACK.** *Interrupt Acknowledge* (input, active Low). This is a strobe which indicates that an interrupt acknowledge cycle is in progress. During this cycle, the ESCC interrupt

daisy chain is resolved. The device is capable of returning an interrupt vector that may be encoded with the type of interrupt pending. During the acknowledge cycle, if IEI is High the ESCC places the interrupt vector on the databus when /RD goes active. /INTACK is latched by the rising edge of PCLK.

## Z80230

**AD7-AD0.** Address/Data Bus (bidirectional, active High, tri-state) These multiplexed lines carry register addresses to the SCC as well as data or control information.

**/AS.** Address Strobe (input, active Low). Addresses on AD7-AD0 are latched by the rising edge of this signal.

**/CS0.** Chip Select 0 (input, active Low). This signal is latched concurrently with the addresses on AD7-AD0 and must be active for the intended bus transaction to occur.

**CS1.** Chip Select 1 (input, active High). This second select signal must also be active before the intended bus transaction can occur. CS1 must remain active throughout the transaction.

**/DS.** Data Strobe (input, active Low). This signal provides timing for the transfer of data into and out of the SCC. If /AS and /DS coincide, this is interpreted as a reset.

**R/W.** Read/Write (input). This signal specifies whether the operation to be performed is a read or a write.

## FUNCTIONAL DESCRIPTION

**Architecture.** The architecture of the ESCC is described from two points of view: as a datacommunications device which transmits and receives data in a wide variety of protocols; and as a microprocessor peripheral in which the ESCC offers valuable features such as vectored interrupts and DMA support.

The ESCC's peripheral and datacommunication functions are described in the following sections. A block diagram is shown in Figure 1. The details of the communications between the receive and transmit logic to the system bus are shown in Figures 5 and 6. The features and data path for each of the ESCC's A and B channels is identical. See the ESCC Technical Manual for full details on using the ESCC.

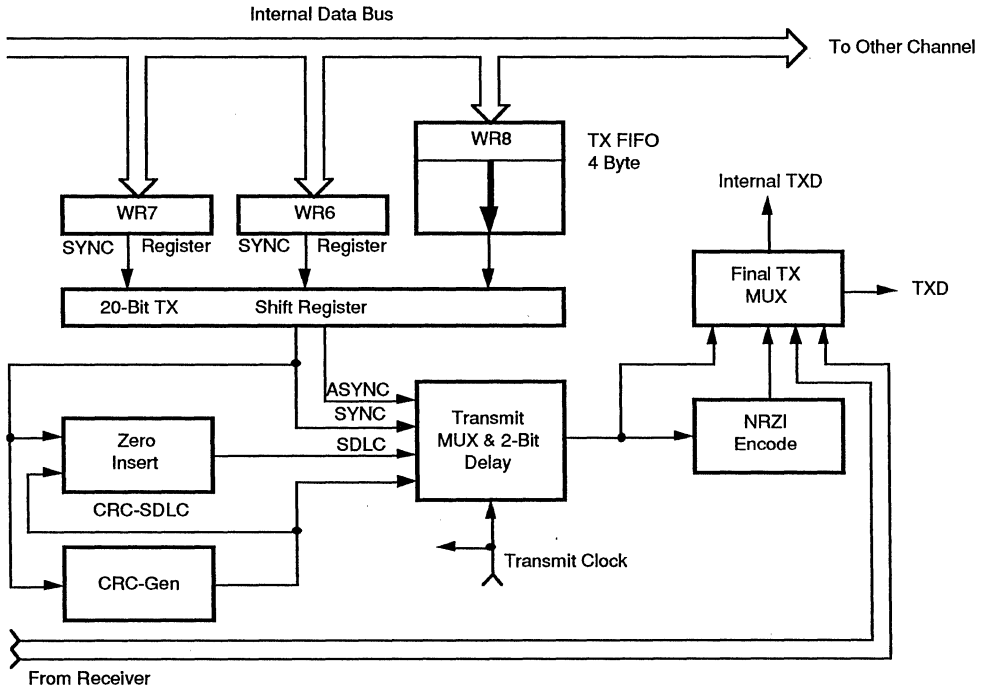


Figure 5. ESCC Transmit Data Path

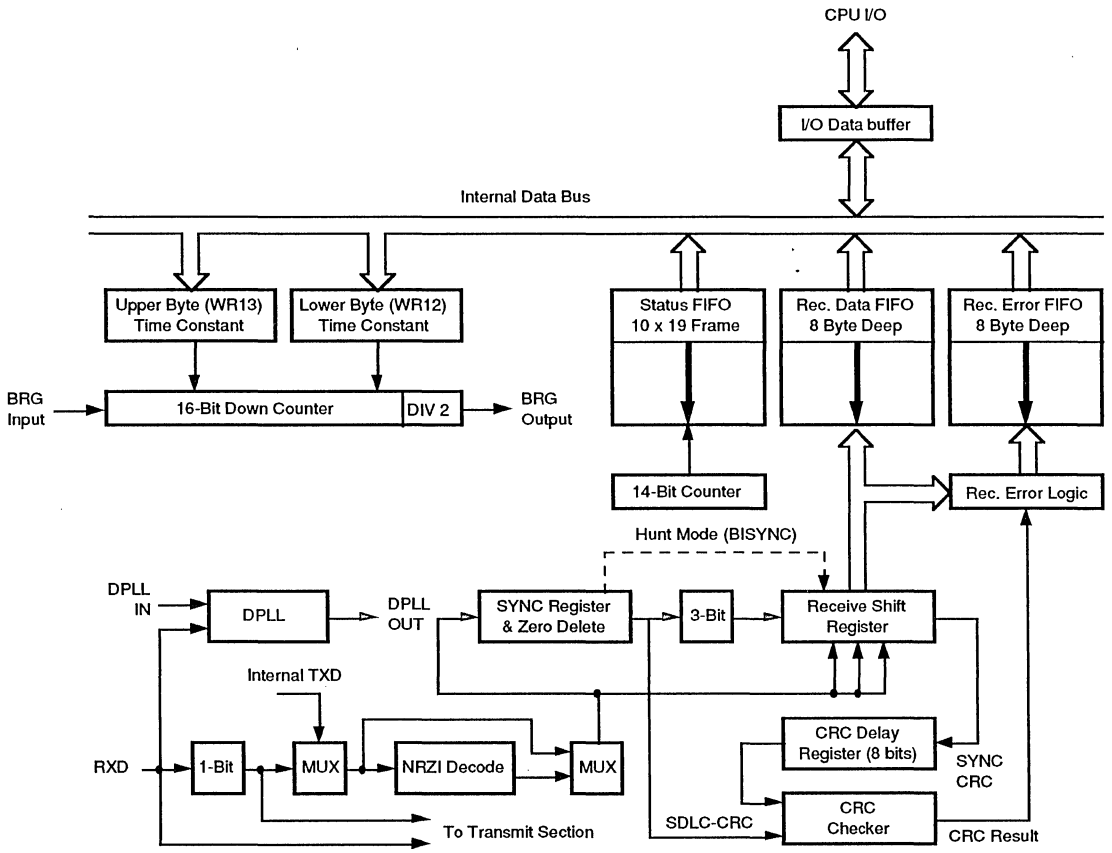


Figure 6. ESCC Receive Data Path

## I/O INTERFACE CAPABILITIES

System communication to and from the ESCC is done through the ESCC's register set. There are 17 write registers and 15 read registers. Many of the new features on the ESCC are enabled through a new register in the ESCC: Write Register 7 Prime (WR7'). This new register can be accessed if bit D0 of WR15 is set. Table 1 lists all of the ESCC's registers and a brief description of their functions.

Throughout this document, the write and read registers are referenced with the following notation: "WR" for Write Register and "RR" for Read Register. For example:

WR4A Write Register 4 for channel A  
 RR3 Read Register 3 for either/both channels

**Table 1. ESCC Write and Read Registers**

Write Register	Functions
WR0	Command Register: Select Shift Left/Right Mode, CRC initialization, and resets for various modes.
WR1	Interrupt conditions, Wait/DMA request control.
WR2	Interrupt Vector (accessed through either channel).
WR3	Receive and miscellaneous control parameters.
WR4	Transmit and Receive parameters and modes.
WR5	Transmit parameters and controls.
WR6	Sync character or SDLC address field.
WR7	Sync character or SDLC flag.
WR7'	SDLC enhancements enable (accessed if WR15 D0 is 1).
WR8	Transmit FIFO (4 bytes deep).
WR9	Reset commands and Master INT enable (accessed through either channel).
WR10	Miscellaneous transmit and receive controls.
WR11	Clock mode control.
WR12	Lower byte of BRG time constant.
WR13	Upper byte of BRG time constant.
WR14	Miscellaneous controls and DPLL commands.
WR15	External interrupt control.
Read Register	Functions
RR0	Transmit, Receive and external status.
RR1	Special Receive Condition status bits.
RR2A	Unmodified interrupt vector.
RR2B	Modified interrupt vector.
RR3A	Interrupt Pending bits.
RR4	WR4 status (if WR7' D6=1).
RR5	WR5 status (if WR7' D6=1).
RR6	SDLC Frame LSB Byte Count (if WR15 D2=1).
RR7	SDLC Frame 10x19 FIFO Status and MSB Byte Count (if WR15 D2=1).
RR8	Receive Data FIFO (8 Deep).
RR9	WR3 status (if WR7' D6=1).
RR10	Miscellaneous status bits.
RR11	WR10 status (if WR7' D6=1).
RR12	Lower Byte of BRG time constant.
RR13	Upper byte of BRG time constant.
RR14	WR7' status (if WR7' D6=1).





---

## I/O INTERFACE CAPABILITIES (Continued)

The ESCC can also execute an interrupt acknowledge cycle through software. In some CPU environments it is difficult to create the /INTACK signal with the necessary timing to acknowledge interrupts and allow the nesting of interrupts. In these cases, the /INTACK signal can be created with a software command to the ESCC. See the Z80230 Enhancements section for more details on this enhancement.

In the ESCC, the Interrupt Pending (IP) bit signals a need for interrupt servicing. When an IP bit is 1 and the IEI input is High, the /INT output is pulled Low, requesting an interrupt. In the ESCC, if the IE bit isn't set by enabling interrupts, then the IP for that source is never set. The IP bits are readable in RR3A.

The IUS bits signal that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority in the ESCC and external to the ESCC are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the ESCC being pulled Low and propagated to subsequent peripherals. An IUS bit is set during an Interrupt Acknowledge cycle if there are no higher priority devices requesting interrupts.

There are three types of interrupts: Transmit, Receive, and External/Status. Each interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receiver, Transmit, and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled (WR1 D1=1), the occurrence of the interrupt depends on the state of WR7' D5. If this bit is reset, the CPU is interrupted when the top byte of the transmit FIFO becomes empty. If WR7' D5 is set, the CPU is interrupted when the transmit FIFO is completely empty. (This implies that the transmitter must have had a data character written into it so that it can become empty.)

When enabled, the receiver can interrupt the CPU in one of three ways:

1. Interrupt on First Receive Character or Special Receive Condition.
2. Interrupt on All Receive Characters or Special Receive Conditions.
3. Interrupt on Special Receive Conditions Only.

If WR7' bit D3 is set, the Receive character interrupt occurs when there are four bytes available in the receive FIFO. This is most useful in synchronous applications as the data is in consecutive bytes. Interrupt on First Character or Special Condition and Interrupt on Special Condition Only are typically used with the Block Transfer mode. A special Receive Condition is one of the following: receiver overrun, framing error in Asynchronous mode, end-of-frame in SDLC mode and, optionally, a parity error. The Special Receive Condition interrupt is different from an ordinary receive character available interrupt only by the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt occurs from Special Receive Conditions any time after the first receive character interrupt.

The main function of the External/Status interrupt is to monitor the signal transitions of the /CTS, /DCD, and /SYNC pins, however, an External/Status interrupt is also caused by a Transmit Underrun condition; a zero count in the baud rate generator; by the detection of a Break (Asynchronous mode), ABORT (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the ABORT or EOP has a special feature allowing the ESCC to interrupt when the ABORT or EOP sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the ABORT condition by external logic in SDLC mode. In SDLC Loop mode, this feature allows secondary stations to recognize the primary station wishes to regain control of the loop during a poll sequence.

**CPU/DMA Block Transfer.** The ESCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the /WAIT//REQUEST output in conjunction with the Wait/Request bits in WR1. The /WAIT//REQUEST output can be defined under software control as a WAIT line in the CPU Block Transfer mode or as a REQUEST line in the DMA Block Transfer mode.

To a DMA controller, the ESCC REQUEST output indicates that the ESCC is ready to transfer data to or from memory. To the CPU, the WAIT line indicates that the ESCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The /DTR//REQUEST line allows full-duplex operation under DMA control. The ESCC can be programmed to deassert the /DTR//REQUEST pin with the same timing as the /WAIT//REQUEST pin if WR7' D4 is set.

## ESCC DATA COMMUNICATIONS CAPABILITIES

The ESCC provides two independent full-duplex programmable channels for use in any common asynchronous or synchronous data communication

protocols (Figure 8). Each of the data communication channels has identical features and capabilities.

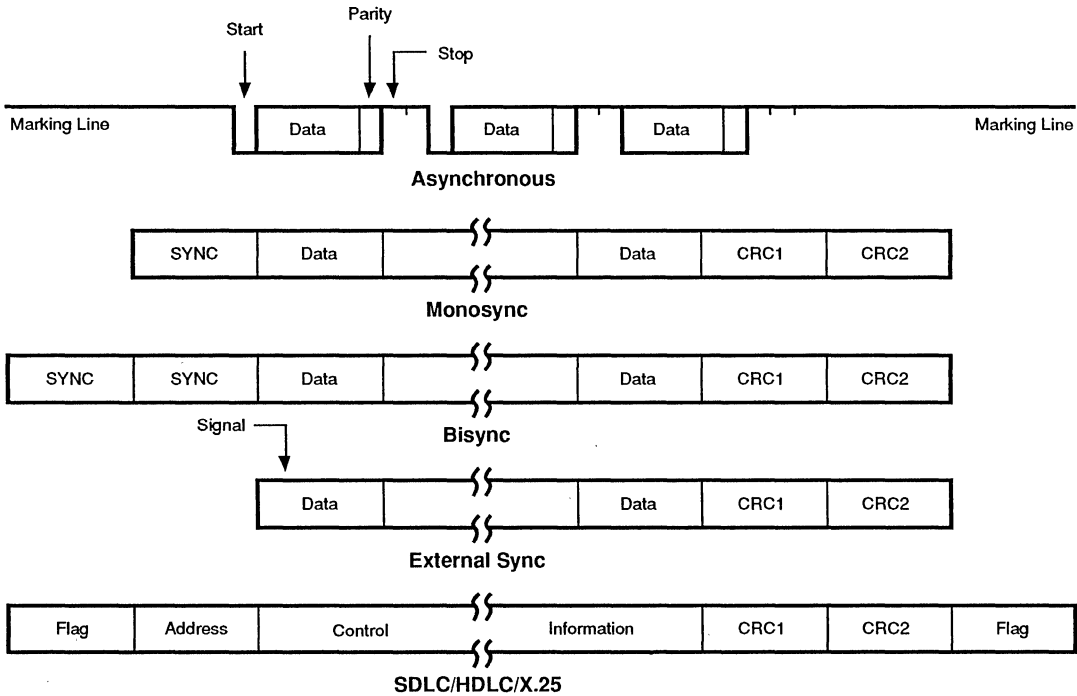


Figure 8. Some ESCC Protocols

The ESCC has significant improvements to its data communications capacity over that of the standard SCC. The addition of the deeper data FIFOs allows for data to be moved in strings instead of on a byte-by-byte basis. The ability to handle data in strings allows for significant improvements in data handling, and consequently, more efficient use of bus bandwidth. The programmability of the INT/DMA level of the FIFOs allows the system designer to determine fill levels as the FIFOs request the system to move data. The deeper data FIFOs are accessible regardless of the protocol used. They do not need to be enabled. For more details on these improvements, see the Z80230 Enhancements section of this specification.

**Asynchronous Modes.** Send and Receive is accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half, or two stop bits per

character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxD<sub>A</sub> or RxD<sub>B</sub> pins). If the Low does not persist (e.g., a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur. Vectored interrupts allow fast servicing or error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

## ESCC DATA COMMUNICATIONS CAPABILITIES (Continued)

The ESCC does not require symmetric transmit and receive clock signals - a feature allowing use of the wide variety of clock sources. The transmitter and receiver handle data at a rate supplied to the receive and transmit clock inputs. In Asynchronous modes, the SYNC pin may be programmed as an input used for functions such as monitoring a ring indicator.

**Synchronous Modes.** The ESCC supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols are handled in several

modes. They allow character synchronization with a 6-bit or 8-bit sync character (Monosync), and a 12-bit or 16-bit synchronization pattern (Bisync), or with an external sync signal. Leading sync characters are removed without interrupting the CPU.

Five or 7-bit synchronous characters are detected with 8- or 16-bit patterns in the ESCC by overlapping the larger pattern across multiple incoming synchronous characters as shown in Figure 9.

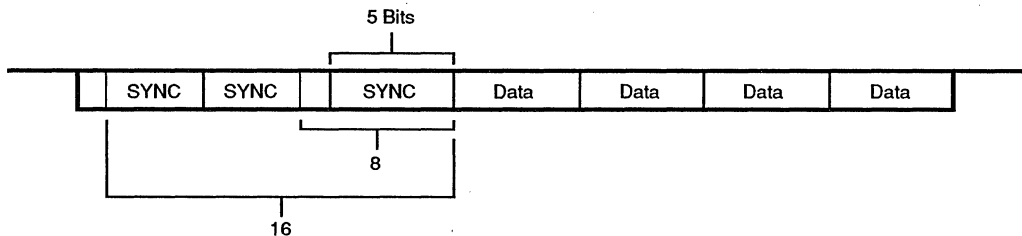


Figure 9. Detecting 5- or 7-Bit Synchronous Characters

CRC checking for Synchronous byte oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols such as IBM Bisync.

Both CRC-16 ( $X^{16} + X^{15} + X^2 + 1$ ) and CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) error checking polynomials are supported. Either polynomial may be selected in all Synchronous modes. Users may preset the CRC generator and checker to all 1s or all 0s. The ESCC also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows for high-speed transmissions under DMA control, with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 6-, 8-, or 16-bit sync characters, regardless of the programmed character length.

**SDLC Mode.** The ESCC supports Synchronous bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero insertion, and CRC generation. A special command is used to abort a frame in transmission. At the end of a message, the ESCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line consisting of continuous flag characters or a steady marking condition.

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort can be issued. The ESCC may also be programmed to send an ABORT itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent, allowing reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the /SYNC pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored.

The number of address bytes are extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all 0s inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. In SDLC mode, the ESCC

must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all 1s or all 0s. The CRC is inverted before transmission and the receiver checks against the bit pattern 0001110100001111.

NRZ, NRZI or FM coding may be used in any 1x mode. The parity options available in Asynchronous modes are available in Synchronous modes.

**SDLC Loop Mode.** The ESCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow on the loop and any number of secondary stations. In SDLC Loop mode, the ESCC performs the functions of a secondary station while an ESCC operating in regular SDLC mode acts as a controller (Figure 10). SDLC loop mode can be selected by setting WR10 bit D1.

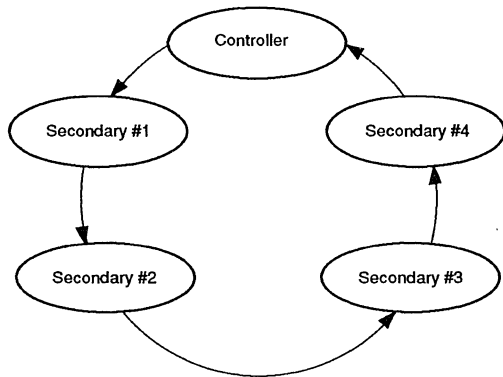


Figure 10. An SDLC Loop

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop and, in fact, passes these messages to the rest of the loop by retransmitting them with a one-bit time delay. The secondary station places its own message on the loop only at specific times. The controller signals that secondary stations can transmit messages by sending a special character, called an EOP (End Of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages, this bit pattern is unique and easily recognized.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit appends their

messages to the message of the first secondary station by the same process. Any secondary stations without messages to send merely echo the incoming message and are prohibited from placing messages on the loop (except upon recognizing an EOP). In SDLC Loop mode, NRZ, NRZI, and FM coding may all be used.

**SDLC FIFO.** The ESCC's ability to receive high speed back-to-back SDLC frames is maximized by a 10-bit deep by 19-bit wide status FIFO. When enabled (through WR15, bit D2), it provides the DMA the ability to continue to transfer data into memory so that the CPU can examine the message later. For each SDLC frame, a 14-bit byte count and 5 status/error bits are stored. The byte count and status bits are accessed through Read Registers 6 and 7. Read Registers 6 and 7 are only accessible when the SDLC FIFO is enabled. The 10x19 status FIFO is separate from the 8-byte receive data FIFO.

**Baud Rate Generator.** Each channel in the ESCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On startup, the flip-flop on the output is set in a High state, the value in the time constant register is loaded into the counter, and the counter starts counting down. The output of the baud rate generator toggles upon reaching 0, the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the digital phase-locked loop (see next section).

If the receive clock or transmit clock is not programmed to come from the TRxC pin, the output of the baud rate generator may be echoed out via the TRxC pin.

The following formula relates the time constant to the baud rate where PCLK or RTxC is the baud rate generator input frequency in Hertz. The clock mode is 1, 16, 32, or 64, as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select 1 and Asynchronous should select 16, 32 or 64.

$$\text{Time Constant} = \frac{\text{PCLK or RTxC Frequency}}{2(\text{Baud Rate}) (\text{Clock Mode})} - 2$$

## ESCC DATA COMMUNICATIONS CAPABILITIES (Continued)

**Digital Phase-Locked Loop.** The ESCC contains a Digital Phase-Locked Loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock is then used as the ESCC receive clock, the transmit clock, or both. When the DPLL is selected as the transmit clock source, it will provide a jitter free clock output that is the DPLL input frequency divided by the appropriate divisor for the selected encoding technique.

For NRZI encoding, the DPLL counts the 32x clock to create nominal bit times. As the 32x clock is counted, the DPLL is searching the incoming data stream for edges (either 1 to 0, or 0 to 1). Whenever an edge is detected, the DPLL makes a count adjustment (during the next counting cycle), producing a terminal count closer to the center of the bit cell.

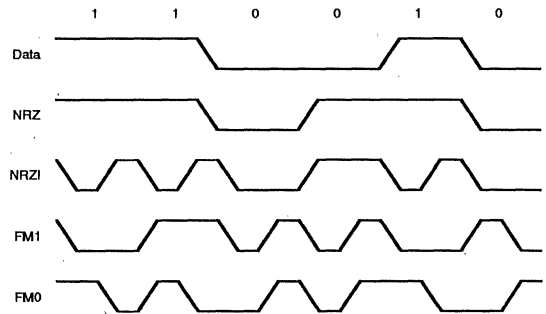
For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16 and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15 to 16 counting transition.

The 32x clock for the DPLL can be programmed to come from either the RTxC input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the ESCC via the TRxC pin (if this pin is not being used as an input).

**Data Encoding.** The ESCC may be programmed to encode and decode the serial data in four different ways (Figure 11). In NRZ encoding, a 1 is represented by a High level and a 0 is represented by a Low level. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. In FM1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A 1 is represented by an additional transition at the center of the bit cell and a 0 is represented by no additional transition at the center of the bit cell. In FM0 (bi-phase space), a transition occurs at the beginning of every bit cell. A 0 is represented by an additional transition at the center of the bit cell, and a 1 is represented by no additional transition at the center of the bit cell. In addition to these four methods, the ESCC can be used to decode

Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0 to 1, the bit is a 0. If the transition is 1 to 0, the bit is a 1.

**Auto Echo and Local Loopback.** The ESCC is capable of automatically echoing everything it receives. This feature is useful mainly in Asynchronous modes, but works in Synchronous and SDLC modes as well. Auto Echo mode (TxD is RxD) is used with NRZI or FM encoding with no additional delay because the data stream is not decoded before retransmission. In Auto Echo mode, the /CTS input is ignored as a transmitter enable (although transitions on this input can still cause interrupts if programmed to do so). In this mode, the transmitter is actually bypassed and the programmer is responsible for disabling transmitter interrupts and /WAIT//REQUEST on transmit.



**Figure 11. Data Encoding Methods**

The ESCC is also capable of Local Loopback. In this mode, TxD or RxD is just like Auto Echo mode. However, in Local Loopback mode the internal transmit data is tied to the internal receive data and RxD is ignored (except to be echoed out via TxD). The /CTS and /DCD inputs are also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in Asynchronous, Synchronous and SDLC modes with NRZ, NRZI or FM coding of the data stream.

## Z80230 ENHANCEMENTS

The following is a detailed description of the enhancements to the Z80230, ESCC from the standard SCC.

### 4-Byte Deep Transmit FIFO

The ESCC has a 4-byte transmit buffer with programmable interrupt and DMA request levels. It is not necessary to enable the FIFO as it is always available. The user can choose to have the Transmit Buffer Empty (TBE) interrupt and DMA Request on Transmit be generated either when the top byte of transmit FIFO is empty or only when the FIFO is completely empty. A hardware or channel reset will reset the transmit shift register, flush the transmit FIFO, and set WR7' D5=1.

If the transmitter generates the Interrupt or DMA request for data when the top byte of the FIFO is empty (WR7' D5=0), the system can allow for a long response time to the data request without underflowing. The interrupt service routine can write one byte and then test RR0 D2 if more data may be written. The DMA Request in this mode will go inactive after each data write and then go active again until the FIFO is filled. The Transmit Buffer Empty status bit (TBE), RR0 bit D2, is set when the top byte of the FIFO is empty. Note that this IS NOT the reset state.

For applications where the frequency of interrupts is important, the transmit interrupt service routine can be optimized by programming the ESCC to generate the TBE interrupt only when the FIFO is completely empty (WR7' D5=1) and then writing four bytes to fill the FIFO. When WR7' D5=1, only one DMA request is generated (filling the bottom of the FIFO). However, this may be preferred for some applications where the possible reassertion of the DMA request is not desired. The Transmit Buffer Empty status bit (TBE), RR0 bit D2, is set when the top byte of the FIFO is empty. (Note that WR7' D5=1 after a hardware or channel reset).

### 8-Byte Receive FIFO

The ESCC has an 8-byte receive FIFO with programmable interrupt levels. The receive character available interrupt is generated as selected by WR7' bit D3. The Receive Character Available bit, RR0 D0, is set when at least one byte is available in the top of the FIFO (independent of WR7' D3). It is not necessary to enable the 8-byte FIFO as it is always available. A hardware or channel reset resets the receive shift register and flushes the receive FIFO.

A DMA Request on Receive, if enabled, is generated whenever one byte is available in the receive FIFO independent of WR7' D3. If more than one byte is available in the FIFO, the /Wait//Request pin goes inactive and then goes active again until the FIFO is emptied.

By resetting WR7' D3=0, applications which have a long latency to interrupts can generate the request to read data from the FIFO when one byte is available, and then test the Receive Character Available bit to determine if more data is available.

By setting WR7' D3=1, the ESCC can be programmed to interrupt when the receive FIFO is half full (4 bytes available) and, therefore, allowing the frequency of receive interrupt to be reduced. If WR7' D3 is set, the receive character available interrupt is generated when there are 4 bytes available. Therefore, if the interrupt service routine reads 4 bytes during each routine, the frequency of interrupts is reduced.

If WR7' D3=1 and "Receive Interrupt on All Characters and Special Conditions" is enabled, the receive character available interrupt is generated when four characters are available. However, when a character is detected to have a special condition, a special condition interrupt is generated when the character is loaded into the top four bytes of the FIFO. Therefore, the special condition interrupt service routine should read RR1 before reading the data to determine which byte has the special condition.

### Write Register 7' (7 prime)

A new register, WR7', has been added to the ESCC to facilitate the programming of six new features. The format of this register is shown in Figure 12.

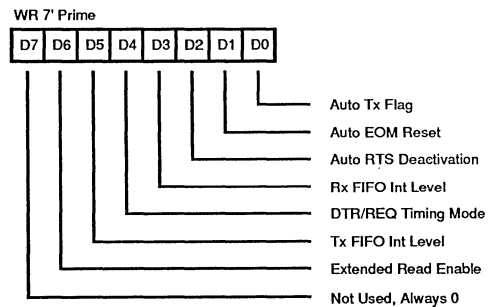


Figure 12. Write Register 7' (7 prime)

WR7' is written to by first setting bit D0 of Write Register 15 (WR15 D0) to one, and then addressing WR7 as normal. All writes to register 7 are to WR7' while WR15 D0 is set. WR15 bit D0 must be reset to 0 to address the sync character register WR7. If bit D6 of WR7' is set, then WR7' can be read by doing a read cycle to RR14. The WR7' features remain enabled until specifically disabled or by a hardware or software reset. Note that bit D5 is set after a reset. All other bits are reset to zero following reset.

## Z85230 ENHANCEMENTS (Continued)

For applications which may use either the Zilog Z80C30 or Z80230, these two device types can be identified in software with the following test. Write a 01 hex to Write Register 15. Then read Read Register 15 and if D0 is reset it is a Z80C30 and, if D0 is set it is a Z80230. Note that if the device is Z80C30, a write to WR15 resetting D0 should be done before proceeding. Also, if the device is Z80230, the result in all writes to address seven will be to WR7' until WR15 D0 is reset.

**Bit 7.** *Not used.* This bit must always be written zero (0).

**Bit 6.** *Extended Read Enable.* Setting this bit enables the ability to read WR3, WR4, WR5, WR7' and WR10. These registers are read by reading RR9 (WR3), RR4, RR5, RR14 (WR7'), and RR11 (WR10), respectively.

**Bit 5.** *Transmit FIFO Interrupt Level.* If this bit is set, the transmit buffer empty interrupt is generated when the

transmit FIFO is completely empty. If this bit is reset, the transmit buffer empty interrupt is generated when the top byte of the transmit FIFO is empty. This bit is set following a hardware or channel reset.

In DMA Request on Transmit mode, when using either the /W//REQ or /DTR//REQ pins, the request is asserted when the Tx FIFO is completely empty if WR7' D5 is set. The request is asserted when the top byte of the FIFO is empty if D5 is reset.

**Bit 4.** */DTR//REQ timing.* If this bit is set and the /DTR//REQ pin is used for Request mode (WR14 D2=1), the deactivation of the /DTR//REQ pin will be identical to the /W//REQ pin as shown in Figure 13. If this bit is reset, the deactivation time is 4TcPc.

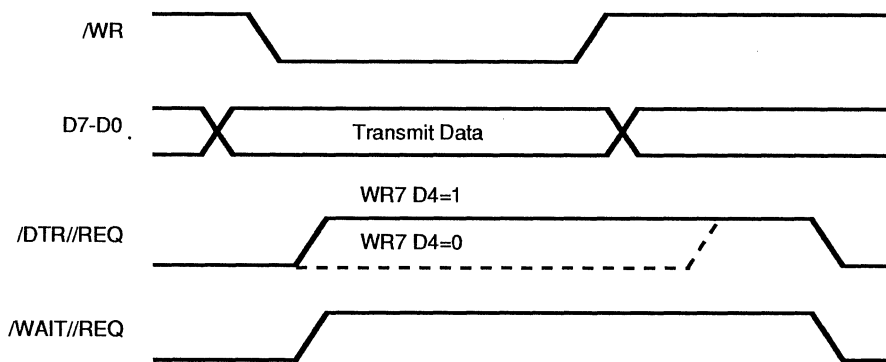


Figure 13. DMA Request on Transmit Deactivation Timing

**Bit 3.** *Receive FIFO Interrupt Level.* This bit sets the interrupt level of the receive FIFO. If this bit is set, the receive data available bit is asserted when the receive FIFO is half full (4 bytes available). If the RFF bit is reset, the receive data available interrupt is generated when a byte reaches the top of the FIFO. See the description of the 8-byte receive FIFO for more details.

**Bit 2.** *Automatic /RTS Pin Deassertion.* This bit controls the timing of the deassertion of the /RTS pin in SDLC mode. If this bit is set and WR5 D1 is reset during the transmission of a SDLC frame, the deassertion of the /RTS pin is delayed until the last bit of the closing flag clears the TxD pin. The /RTS pin is pulled high after the rising edge of the transmit clock cycle from the last bit of the closing flag. This implies

that the ESCC should be programmed for "Flag on Underrun" (WR10 D2=0) for the /RTS pin to deassert at the end of the frame. This feature works independently of the programmed transmitter idle state. In synchronous modes other than SDLC, the /RTS pin will immediately follow the state programmed into WR5 D1. When WR7' D2 is reset, the /RTS follows the state of WR5 D1.

**Bit 1.** *Automatic EOM Reset.* If this bit is set, the ESCC automatically resets the Tx Underrun/EOM latch and pre-sets the transmit CRC generator to its programmed preset state (per values set in WR5 D2 and WR10 D7). Therefore, it is not necessary to issue the Reset Tx Underrun/EOM latch command when this feature is enabled.

**Bit 0. Automatic Tx SDLC Flag.** If this bit is set, the ESCC will automatically transmit an SDLC flag before transmitting data. This removes the requirement to reset the mark idle bit (WR10 D3) before writing data to the transmitter.

Historically, the SCC has latched the databus on the falling edge of /WR. However, as many CPUs do not guarantee that the databus is valid when the /WR pin goes low, Zilog has modified the databus timing to allow a maximum delay from the /WR signal going active Low to the latching of the databus.

**Complete CRC Reception in SDLC Mode**

In SDLC mode, the entire CRC is clocked into the receive FIFO. The ESCC completes clocking in the CRC to allow it to be retransmitted, unaltered, or manipulated in software. In the SCC when the closing flag is recognized, the contents of the receive shift register are immediately transferred to the receive FIFO resulting in the last two bits of the CRC being lost. In the ESCC, it is not necessary to program this feature. When the closing flag is detected, the last two bits of the CRC are clocked into the receive FIFO. In all other synchronous modes, the ESCC does not clock in the last two CRC bits (same as SCC).

**TxD Forced High in SDLC with NRZI Encoding When Marking Idle**

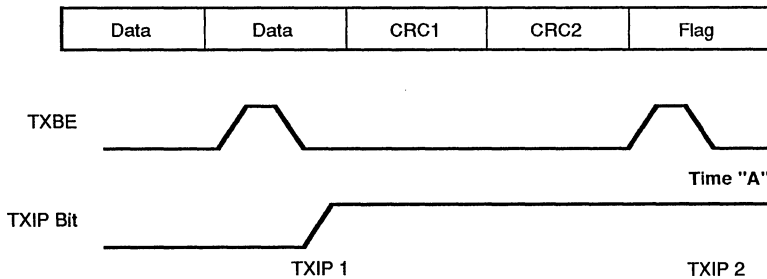
When the ESCC is programmed for SDLC mode with NRZI data encoding and mark idle (WR10 D6=0, D5=1, D3=1), the TxD pin is automatically forced high when the transmitter goes to the mark idle state. There are several different

ways for the transmitter to go into the idle state. In each of the following cases the TxD pin is forced high when the mark idle condition is reached: data, CRC, flag and idle; data, flag and idle; data, abort (on underrun) and idle; data, abort (command) and idle; idle flag and command to idle mark. The force high feature is disabled when the mark idle bit is reset.

This feature is used in combination with the automatic SDLC opening flag transmission feature, WR7' D0=1, to assure that data packets are properly formatted. Therefore, when these features are used together, it is not necessary for the CPU to issue any commands when using the force idle mode in combination with NRZI data encoding. If WR7' D0 is reset, like in the SCC, it is necessary to reset the mark idle bit (WR10 D3) to enable flag transmission before an SDLC packet is transmitted.

**Improved Transmit Interrupt Handling in Synchronous Modes**

The ESCC latches the Transmit Buffer Empty (TBE) interrupt due to the CRC being loaded to the transmit shift register even if the TBE interrupt, due at the last data byte, has not yet been reset. Therefore, the end of a synchronous frame is guaranteed to generate two TBE interrupts even if a reset transmit buffer interrupt command for the data created interrupt is issued after (time "A" in Figure 14) the CRC interrupt had occurred. In this case, two reset TBE commands are required. The TxIP is latched if the EOM latch has been reset before the end of the frame.



**Figure 14. TxIP Latching**



---

## NEW FEATURE DESCRIPTION (Continued)

### DPLL Counter Tx Clock Source

When DPLL output is selected as the transmit clock source, the DPLL counter output is the DPLL source clock divided by the appropriate divisor for the programmed data encoding format. Therefore, in FM mode (FM0 or FM1), the DPLL counter output is the input frequency divided by 16.

In NRZI mode, the DPLL counter frequency is the input divided by 32. This feature provides a jitter-free output and replaces the DPLL transmit clock output being available as the transmit clock source. This has no effect on the use of the DPLL as the receive clock source (Figure 15).

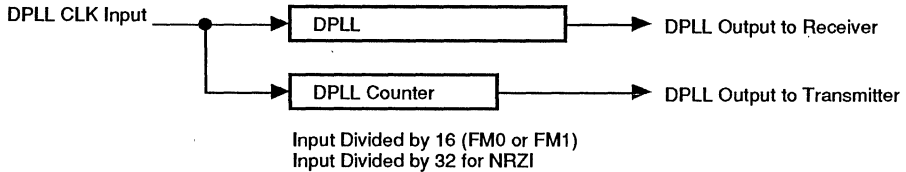


Figure 15. DPLL Outputs

---

### Read Register 0 Status Latched During Read Cycle

The contents of Read Register 0, RR0, are latched during a read to this register. The ESCC prevents the contents of RR0 to change while the Read cycle is active. The SCC allows the status of RR0 to change while reading the register and, therefore, it is necessary to read RR0 twice to detect changes that otherwise may be missed. The contents of RR0 are updated after the rising edge of /RD.

### Software Interrupt Acknowledge

The Z80230 interrupt acknowledge cycle can be initiated through software. If Write Register 9 (WR9) bit D5 is set, reading register 2 (RR2) results in an interrupt acknowledge cycle to be executed internally. Like a hardware INTACK cycle, a software acknowledge causes the INT pin to return high, the IEO pin to go Low and set the IUS latch for the highest priority interrupt pending.

Similar to when the hardware INTACK signal can be used, a software acknowledge cycle requires that a Reset Highest IUS command be issued in the interrupt service routine. Whenever an interrupt acknowledge cycle is used, hardware or software, a reset highest IUS command is required. If RR2 is read from channel A, the unmodified vector is returned. If RR2 is read from channel B, then the vector is modified to indicate the source of the interrupt. The Vector Includes Status (VIS) and No Vector (NV) bits in WR9 are ignored when bit D5 is set to 1.

When the INTACK and IEI pins are not being used, they should be pulled up to  $V_{cc}$  through a resistor (2.2k Ohm typical).

### Fast SDLC Transmit Data Interrupt Response

To more easily facilitate the transmission of back-to-back SDLC frames with a single shared flag between frames, the ESCC allows data for a second frame to be written to the transmit FIFO after the Tx Underrun/EOM interrupt has occurred. This allows application software more time to write the data to the transmitter while allowing the current frame to be properly concluded with CRC and flag. The SCC historically has required that data not be written to the transmitter until a transmit buffer empty interrupt was generated after the CRC has completed transmission. If data is written to the transmit FIFO after the Transmit Underrun/EOM interrupt and before the transmit buffer empty interrupt, the Automatic EOM Reset feature should be enabled (WR7' D1=1). Consequently, the commands "Reset Tx/Underrun EOM" latch and "Reset Tx CRC Generator" should not be used.

### SDLC FIFO Frame Status FIFO Enhancement

When used with a DMA controller, the Z80230 SDLC Frame Status FIFO enhancement maximizes the ESCC's ability to receive high speed, back-to-back SDLC messages. It minimizes frame overruns due to CPU latencies in responding to interrupts. Additional logic was added to the industry standard SCC consisting of a 10-bit deep by 19-bit wide status FIFO, 14-bit receive byte counter, and control logic as shown in Figure 16. The 10 x 19 bits status FIFO is separate from the 8-byte receive data FIFO.

When the enhancement is enabled, the status in Read Register 1 (RR1) and byte count for the SDLC frame are stored in the 10 x 19-bit status FIFO. This allows the DMA controller to transfer the next frame into memory while the CPU verifies that the message was properly received.

---

Summarizing the operation; data is received, assembled, and loaded into the 8-byte FIFO before being transferred to memory by the DMA controller. When a flag is received at the end of an SDLC frame, the frame byte count from the 14-bit counter and five status bits are loaded into the status FIFO for verification by the CPU. The CRC checker is automatically reset in preparation for the next frame which can begin immediately. Since the byte count and status are saved for each frame, the message integrity is verified at a later time. Status information for up to 10 frames is stored before a status FIFO overrun can occur.

If a frame is terminated with an ABORT, the byte count and status will be loaded to the status FIFO and the counter reset for the next frame.

**FIFO Detail.** For a better understanding of details of the FIFO operation, refer to the block diagram in Figure 16.

**Enable/Disable.** This FIFO is implemented so that it is enabled when WR15, bit D2, is set and the ESCC is in the SDLC/HDLC mode. Otherwise, the status register contents bypass the FIFO and go directly to the bus interface (the FIFO pointer logic is reset either when disabled or via a channel or power-on reset). When the FIFO mode is disabled, the ESCC is completely downward compatible with the NMOS Z8030. The FIFO mode is disabled on power-up (WR15 D2 is set to 0 on reset). The effects of backward compatibility on the register set are that RR4 is an image of RR0, RR5 is an image of RR1, RR6 is an image of RR2 and RR7 is an image of RR3. For details on the added registers, refer to Figure 18. The status of the FIFO Enable signal is obtained by reading RR15, bit D2. If the FIFO is enabled, the bit will be set to 1; otherwise, it will be reset.

**Read Operation.** When WR15 bit D2 is set and the FIFO is not empty, the next read to status register RR1 or the additional registers RR7 and RR6, are from the FIFO. Reading status register RR1 causes one location of the FIFO to be emptied, so status is read after reading the byte count, otherwise the count is incorrect. Before the FIFO underflows, it is disabled. In this case, the multiplexer is switched to allow status to read directly from the status register. Reads from RR7 and RR6 contain bits that are undefined. Bit D6 of RR7 (FIFO Data Available) is used to determine if status data is coming from the FIFO or directly from the status register, since it is set to 1 whenever the FIFO is not empty.

Since not all status bits are stored in the FIFO, the All Sent, Parity, and EOF bits bypass the FIFO. The status bits sent through the FIFO are Residue Bits (3), Overrun, and CRC Error.

The sequence for proper operation of the byte count and FIFO logic is to read the registers in the following order: RR7, RR6, and RR1 (reading RR6 is optional). Additional logic prevents the FIFO from being emptied by multiple reads from RR1. The read from RR7 latches the FIFO empty/full status bit (D6) and steers the status multiplexer to read from the ESCC megacell instead of the status FIFO (since the status FIFO is empty). The read from RR1 allows an entry to be read from the FIFO (if the FIFO was empty, logic was added to prevent a FIFO underflow condition).

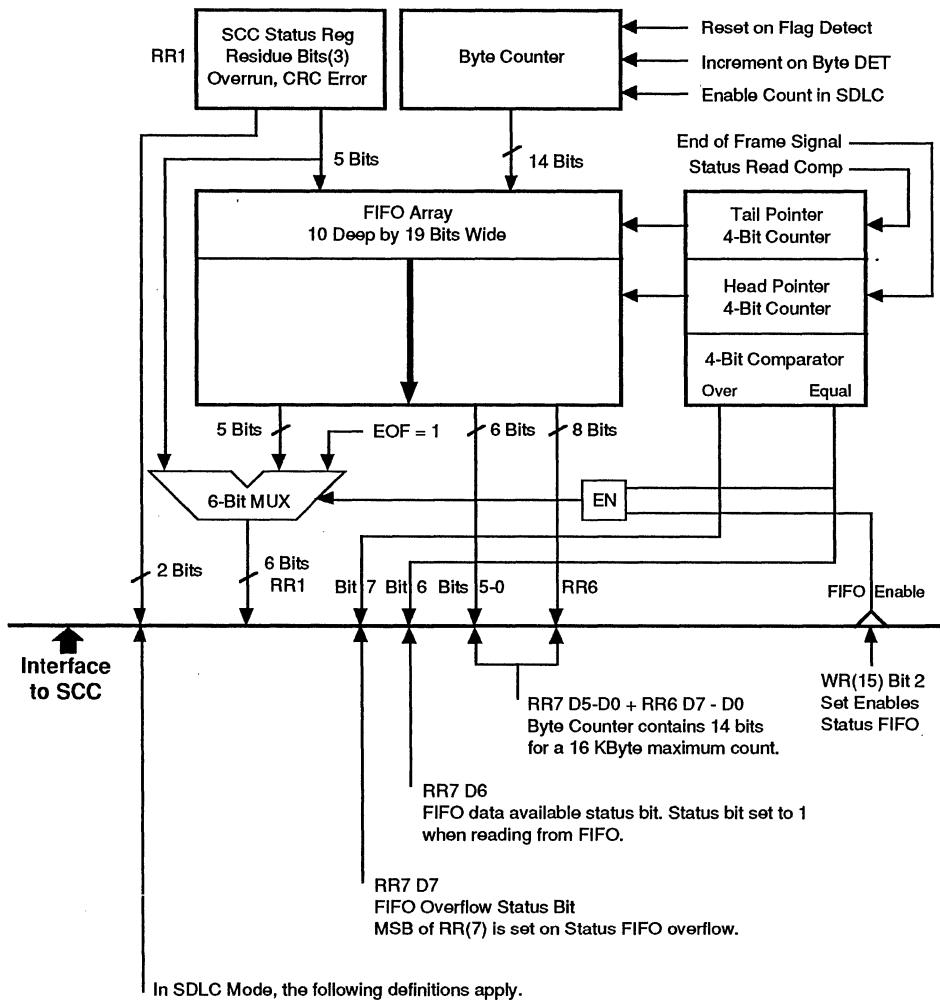
**Write Operation.** When the end of an SDLC frame (EOF) has been received and the FIFO is enabled, the contents of the status and byte-count registers are loaded into the FIFO. The EOF signal is used to increment the FIFO. If the FIFO overflows, the RR7 bit D7 (FIFO Overflow) is set to indicate the overflow. This bit and the FIFO control logic is reset by disabling and re-enabling the FIFO control bit (WR15 bit D2). For details of FIFO control timing during an SDLC frame, refer to Figure 17.

**SDLC Status FIFO Anti-Lock Feature.** When the Frame Status FIFO is enabled and the ESCC is programmed for "Special Receive Condition Only" (WR1 D4=D3=1), the data FIFO is not locked when a character with End of Frame status is read (Figure 16). When a character with the EOF status is at the top of the FIFO, an interrupt with a vector for receive data is generated. The command "Reset Highest IUS" must be issued at the end of the interrupt service routine regardless if an interrupt acknowledge cycle had been executed (hardware or software). This allows a DMA to complete transfer of the received frame to memory and then interrupt the CPU that a frame has been completed without locking the FIFO. Since in the "Receive Interrupt on Special Condition Only" mode the interrupt vector for receive data is not used, it is used to indicate that the last byte of a frame has been read out the receive FIFO. Reading the frame status (CRC, byte count and other status stored in the status FIFO) to determine EOF is not required.

When a character with a special receive condition other than EOF is received (receiver overrun, or parity), a special receive condition interrupt is generated after the character is read from the FIFO and the receive FIFO is locked until the "Error Reset" command is issued.

**NEW FEATURE DESCRIPTION** (Continued)

**Frame Status FIFO Circuitry**



**Figure 16. SDLC Frame Status FIFO**

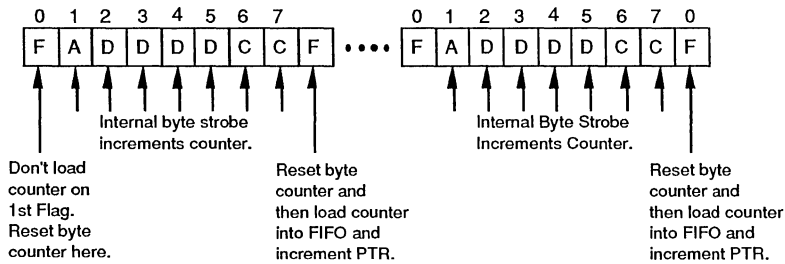


Figure 17. SDLC Byte Counting Detail

## PROGRAMMING

The ESCC contains write registers in each channel that are programmed by the system separately to configure the functional uniqueness of the channels.

In the ESCC, the data registers are directly addressed by selecting a High on the D/C pin. With all other registers (with the exception of WR0 and RRO), programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to WR0 and contains three bits that point to the selected register. The second write is the actual control word for the selected register, and if the second operation is read, the selected read register is accessed. All of the ESCC registers, including the data registers, may be accessed in this fashion. The pointer bits are automatically cleared after the read or write operation so that WR0 (or RRO) is addressed again.

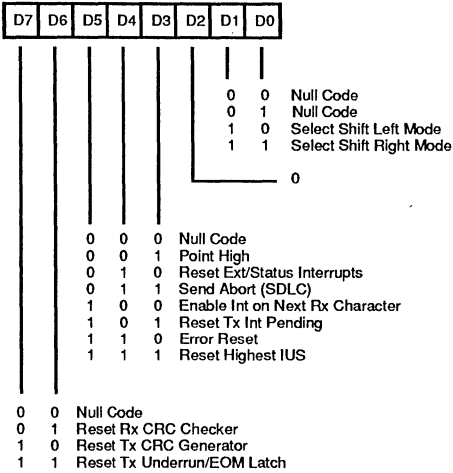
**Initialization.** The system program first issues a series of commands to initialize the basic mode of operation. This is followed by other commands to qualify conditions within the selected mode. For example, in the Asynchronous mode, character length, clock rate, number of stop bits, and even or odd parity should be set first. Then the interrupt mode is set, and finally, the receiver and transmitter are enabled.

**Write Registers.** The ESCC contains 16 write registers (17 counting the transmit buffer) in each channel. These write registers are programmed separately to configure the functional "personality" of the channels. There are two registers (WR2 and WR9) shared by the two channels that are accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits and reset commands. A new register, WR7', was added to the ESCC and may be written to if WR15 D0 is set. Figure 18 shows the format of each write register.

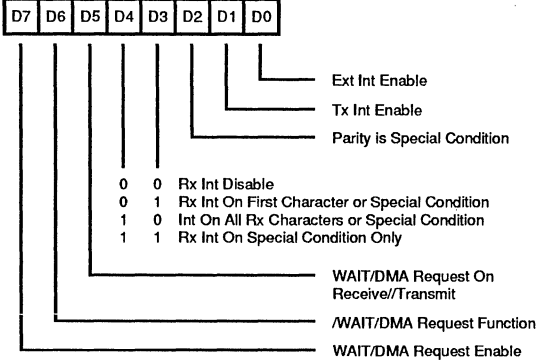
**Read Registers.** The ESCC contains ten read registers (eleven, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers (RR12 and RR13) are read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A only). RR6 and RR7 contain the information in the SDLC Frame Status FIFO, but is only read when WR15 D2 is set. If WR7' D6 is set, Write Registers WR3, WR4, WR5, WR7', and WR10 can be read as RR9, RR4, RR5, and RR14, respectively. Figure 19 shows the format of each Read register.

# CONTROL REGISTERS

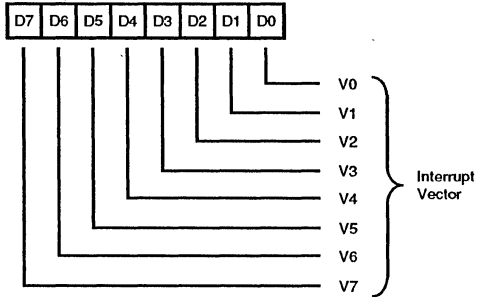
Write Register 0



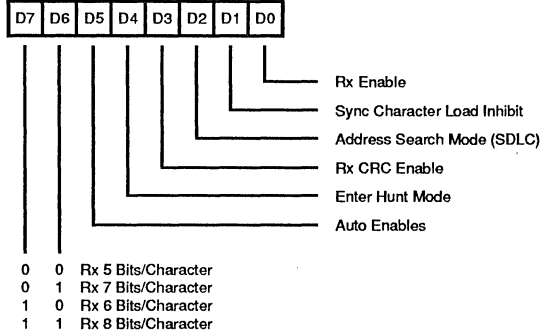
Write Register 1



Write Register 2



Write Register 3



Write Register 4

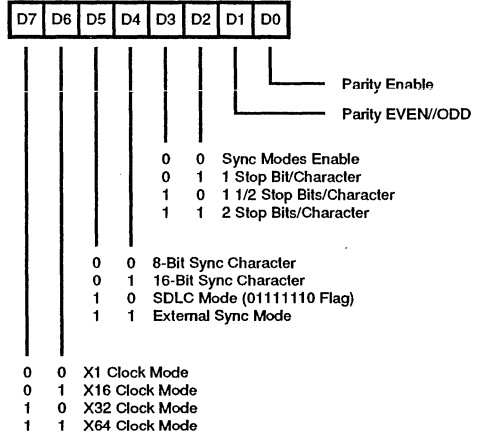
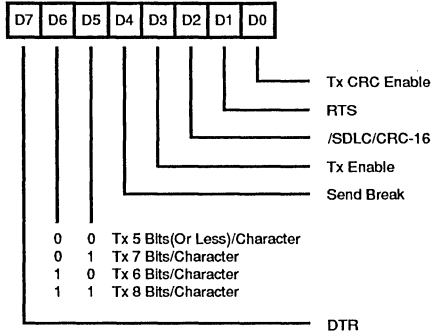
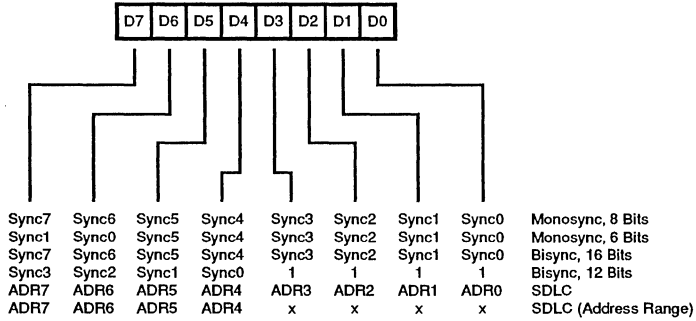


Figure 18. Write Register Bit Functions

Write Register 5



Write Register 6



Write Register 7

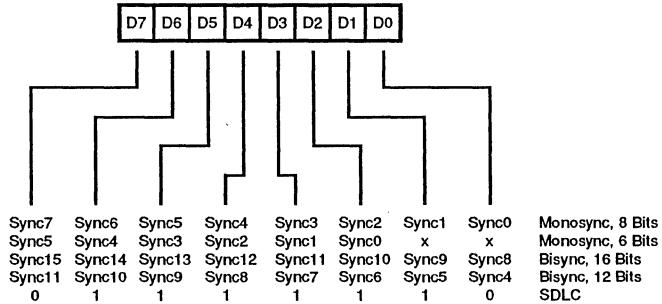
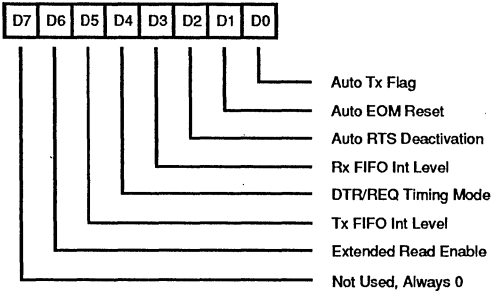


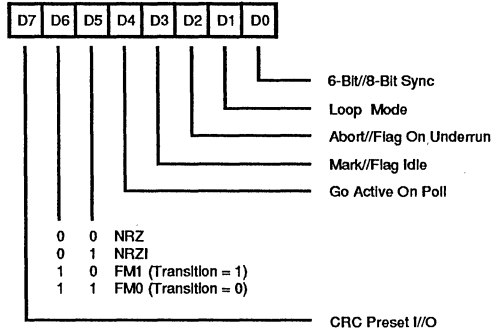
Figure 18. Write Register Bit Functions (Continued)

# CONTROL REGISTERS (Continued)

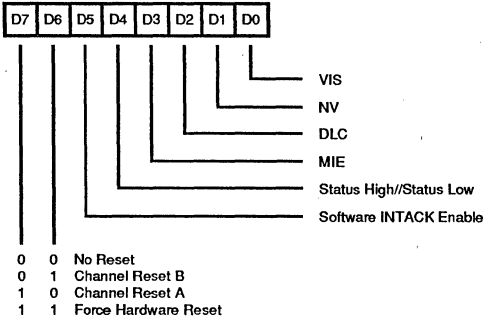
WR 7' Prime



Write Register 10



Write Register 9



Write Register 11

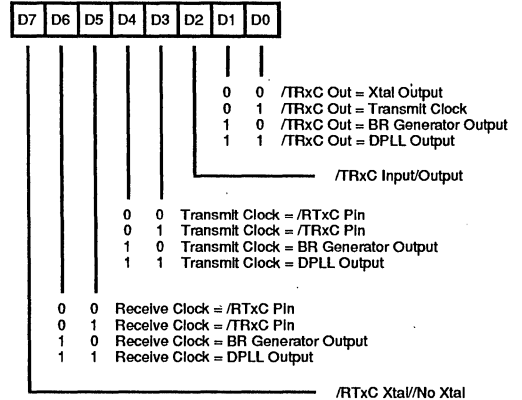
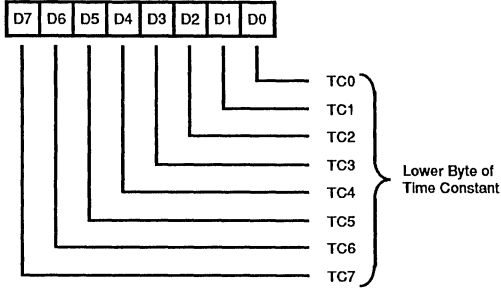
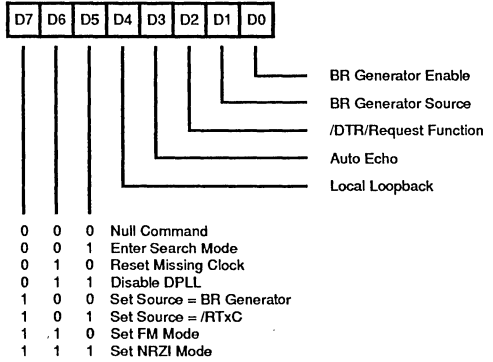


Figure 18. Write Register Bit Functions (Continued)

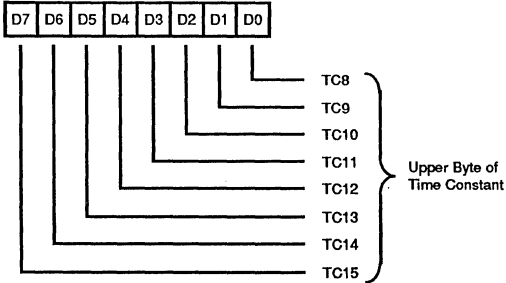
Write Register 12



Write Register 14



Write Register 13



Write Register 15

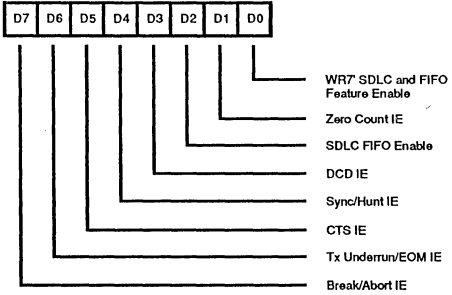
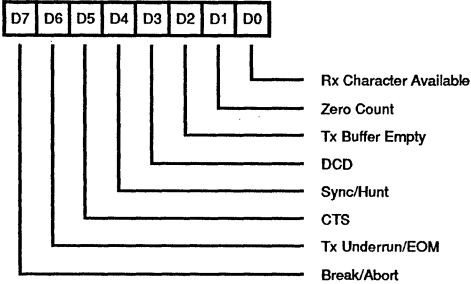


Figure 18. Write Register Bit Functions (Continued)

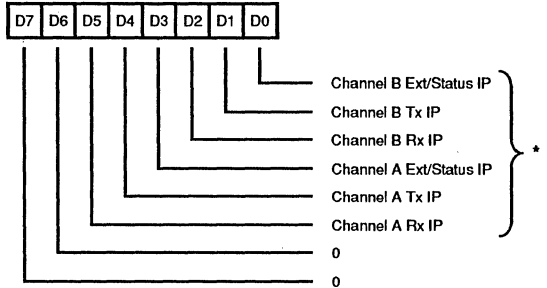


# CONTROL REGISTERS (Continued)

Read Register 0

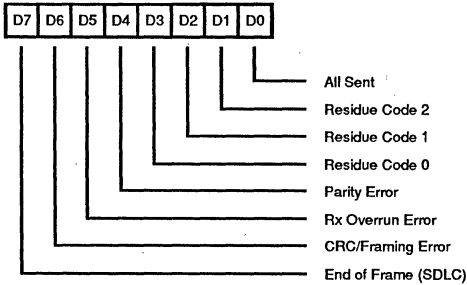


Read Register 3

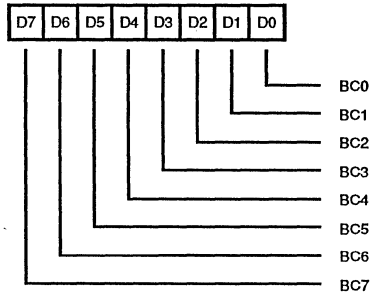


\* Always 0 In B Channel

Read Register 1



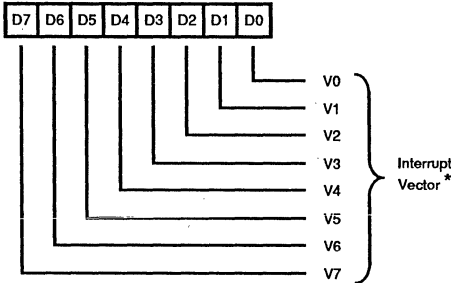
Read Register 6 \*



\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

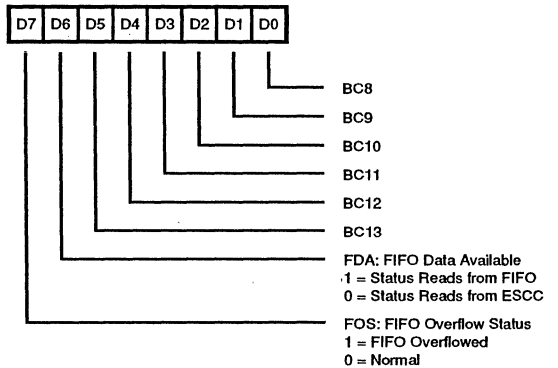
### SDLC FIFO Status and Byte Count (LSB)

Read Register 2



\* Modified In B Channel

Read Register 7 \*

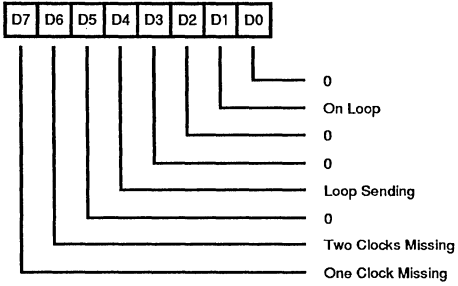


\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

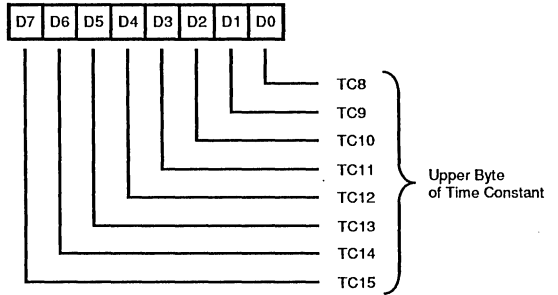
### SDLC FIFO Status and Byte Count (MSB)

Figure 19. Read Register Bit Functions

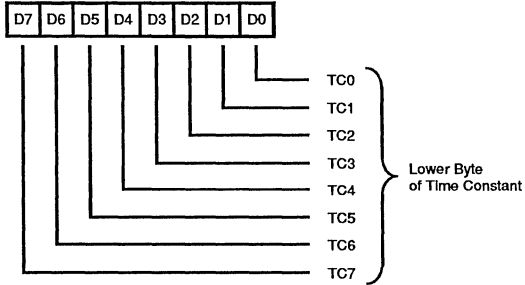
Read Register 10



Read Register 13



Read Register 12



Read Register 15

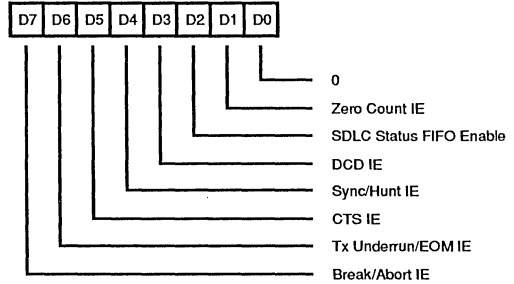


Figure 19. Read Register Bit Functions (Continued)

## Z80230 INTERFACE TIMING

The Z-Bus compatible ESCC is suited for system applications with multiplexed address/data buses similar to the Z8<sup>®</sup>, Z8000<sup>®</sup>, Z280<sup>®</sup>.

Two control signals, /AS and /DS, are used by the Z80230 to time bus transactions. In addition, four other control signals (/CS0, CS1, R/W, and /INTACK) are used to control the type of bus transaction that occurs. A bus transaction is initiated by /AS; the rising edge latches the register address on the Address/Data bus and the state of /INTACK and /CS0.

In addition to timing bus transactions, /AS is used by the interrupt section to set the Interrupt Pending (IP) bits. Because of this, /AS must be kept cycling for the interrupt section to function properly.

The Z80230 generates internal control signals in response to a register access. Since /AS and /DS have no phase

relationship with PCLK, the circuitry generating these internal control signals provide time for metastable condition to disappear. This results in a recovery time related to PCLK.

This recovery time applies only to transactions involving the Z80230, and any intervening transactions are ignored. This recovery time is four PCLK cycles, measure from the falling edge of /DS of one access to the ESCC, to the falling edge of /DS for a subsequent access.

### Z80230 Read Cycle Timing

The Read cycle timing for the Z80230 is shown in Figure 20. The register address on AD7-AD0, as well as the state of /CS0 and /INTACK, are latched by the rising edge of /AS. R/W must be HIGH before /DS falls to indicate a Read cycle. The Z80230 data bus drivers are enabled while CS1 is HIGH and /DS is LOW.

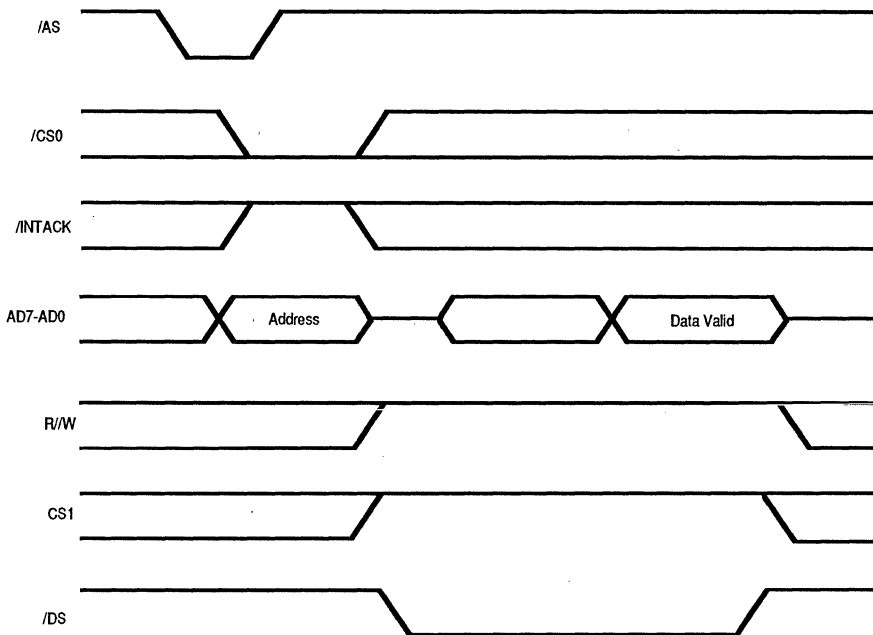


Figure 20. Read Cycle Timing

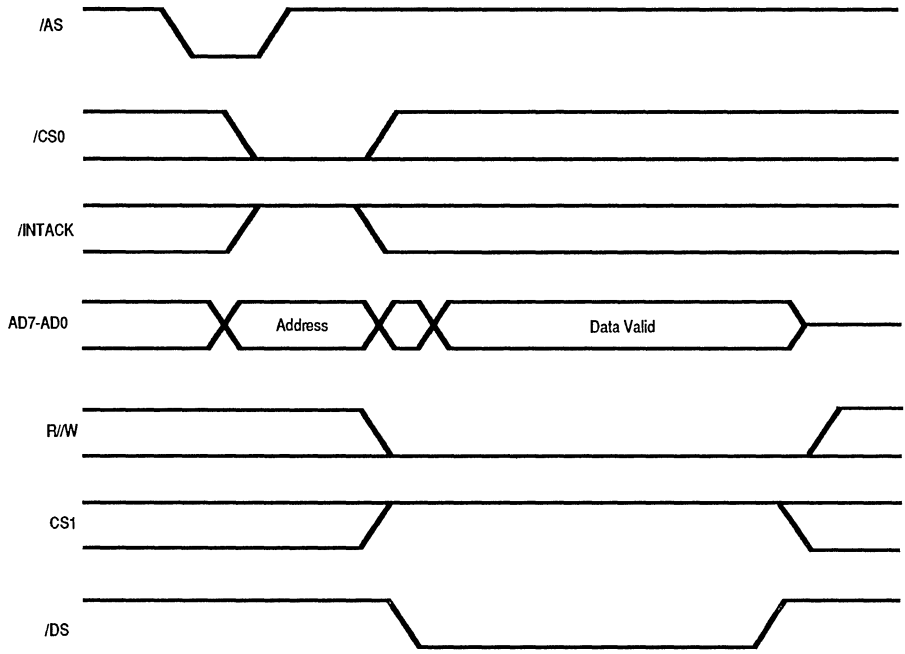


Figure 21. Write Cycle Timing

## Z80230 Interrupt Acknowledge Cycle Timing

The Interrupt Acknowledge cycle timing for the Z80230 is shown in Figure 22. The address on AD7-AD0 and the state of /CS0 and /INTACK are latched by the rising edge of /AS. However, if /INTACK is LOW, the address /CS0, CS1 and R/W are ignored for the duration of the interrupt acknowledge cycle.

The Z80230 samples the state of /INTACK on the rising edge of /AS and AC parameters #7 and #8 specify the setup and hold time requirements. Between the rising edge of /AS and the falling edge of /DS, the internal and external daisy chains settle (AC parameter #29). A system with no external daisy chain should provide the time priority internal to the ESCC. Systems using an external daisy chain should refer to Note 5 referenced in the Z80230

Read/Write and Interrupt Acknowledge Timing for the time required to settle the daisy chain.

If there is an interrupt pending in the ESCC, and IEI is HIGH when /DS falls, the acknowledge cycle was intended for the ESCC. This being the case, the Z80230 sets the Interrupt Under Service (IUS) latch for the highest priority pending interrupt, as well as placing an interrupt vector on AD7-AD0. The placing of a vector on the bus can be disabled by setting WR9, D1=1. The /INT pin also goes inactive in response to the falling edge of /DS. Note that there should be only one /DS per acknowledge cycle. Another important fact is that the IP bits in the Z80230 are updated by /AS, which may delay interrupt requests if the processor does not supply /AS strobes during the time between accesses of the Z80230.

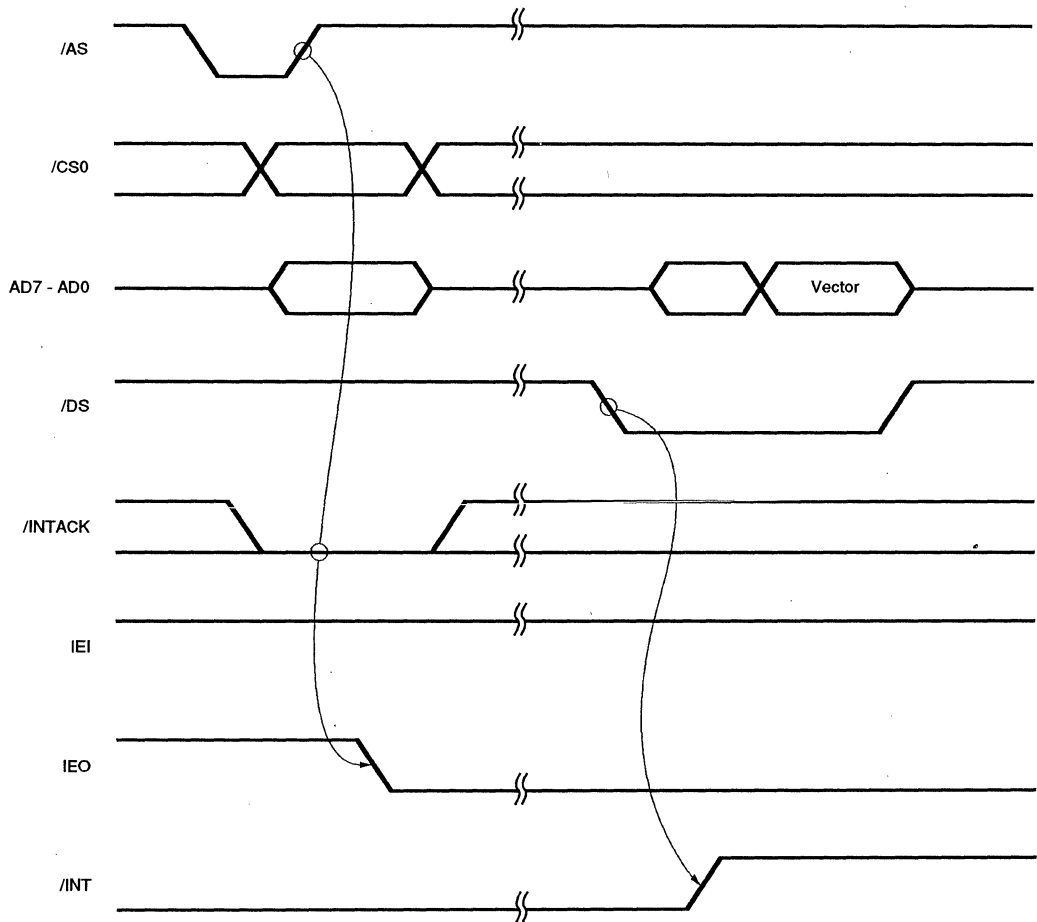


Figure 22. Interrupt Acknowledge Cycle Timing

---

---

## OTHER ZILOG DATA COMMUNICATIONS PRODUCTS

---

### SIO Family

---

Z84C40 SIO	Dual channel multiprotocol USART.
Z84C13 IPC	Z80 CPU with integrated SIO, CTC and WDT.
Z84C15 IPC	Z80 CPU with integrated SIO, CTC, WDT and PIO.

---

### SCC Family

---

Z08530 SCC	NMOS SCC low cost with speeds up to 8 MHz.
Z08030 SCC	NMOS SCC for multiplexed buses.
Z85C30 SCC	CMOS SCC at speeds up to 16 MHz. NMOS compatible.
Z80C30 SCC	CMOS SCC for multiplexed buses.
Z85230 ESCC™	CMOS ESCC for non-multiplexed buses.
Z16C35 ISCC™	SCC with 4 channel DMA and advanced CPU interface.
Z80181 SAC™	Z180 CPU with integrated single channel SCC.

---

### USC Family

---

Z16C30 USC™	Dual channel high performance multi-protocol data communications up to 10 Megabits/second.
Z16C33 MUSC™	Single channel USC with ISDN Time Slot Assigner.
Z16C31 IUSC™	MUSC with high performance dual channel DMA.
Z16C50 DDPLL™	Dual channel DPLL cell from the USC.

---

## ABSOLUTE MAXIMUM RATINGS

$V_{CC}$  Supply Voltage range ..... -0.3V to +7.0V  
 Voltages on all pins  
 with respect to GND ..... -0.3V to  $V_{CC} + 0.3V$   
 Operating Ambient  
 Temperature ..... See Ordering Information  
 Storage Temperature ..... -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and capacitance sections below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin.

Standard conditions are as follows:

- $+4.50\text{ V} \leq V_{CC} \leq +5.50\text{ V}$
- $GND = 0\text{ V}$
- $T_A$  as specified in Ordering Information

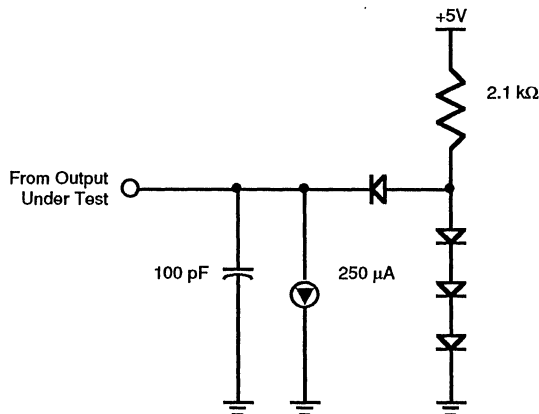


Figure 23. Standard Test Load

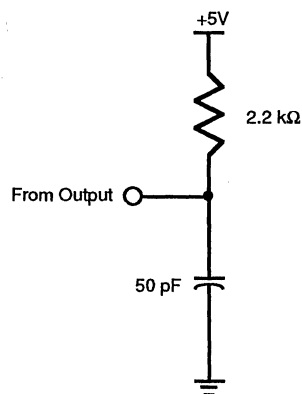


Figure 24. Open-Drain Test Load

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit	Test Condition
$C_{IN}$	Input Capacitance		10	pF	Unmeasured pins returned to Ground.
$C_{OUT}$	Output Capacitance		15	pF	
$C_{I/O}$	Bidirectional Capacitance		20	pF	

**Note:**

$f = 1\text{ MHz}$ , over specified temperature range.

## MISCELLANEOUS

Gate Count - 11,000

## DC CHARACTERISTICS

Z80230

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC}+0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6 \text{ mA}$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$			V	$I_{OH} = -250 \mu\text{A}$
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = 2.0 \text{ mA}$
$I_{IL}$	Input Leakage			$\pm 10.0$	$\mu\text{A}$	$0.4 \leq V_{IN} \leq +2.4\text{V}$
$I_{OL}$	Output Leakage			$\pm 10.0$	$\mu\text{A}$	$0.4 \leq V_{OUT} \leq +2.4\text{V}$
$I_{CC1}$	$V_{CC}$ Supply Current		4	10 (8.5 MHz)	mA	$V_{CC}=5\text{V}$ $V_{IH}=4.8$ $V_{IL}=0.2\text{V}$ Crystal Oscillators off
			5	12 (10 MHz)	mA	
			7	15 (16 MHz)	mA	
			9	20 (20 MHz)	mA	
$I_{CC(OSC)}$	Crystal OSC Current		6		mA	Current for each osc. in addition to $I_{CC1}$

### Notes:

[1]  $V_{CC} = 5\text{V} \pm 10\%$  unless otherwise specified, over specified temperature range.

[2] Typical  $I_{CC}$  was measured with oscillator off.

[3] No  $I_{CC(OSC)}$  max is specified due to dependency on the external circuit.



# AC CHARACTERISTICS

## Z80230 Read and Write Timing Diagrams

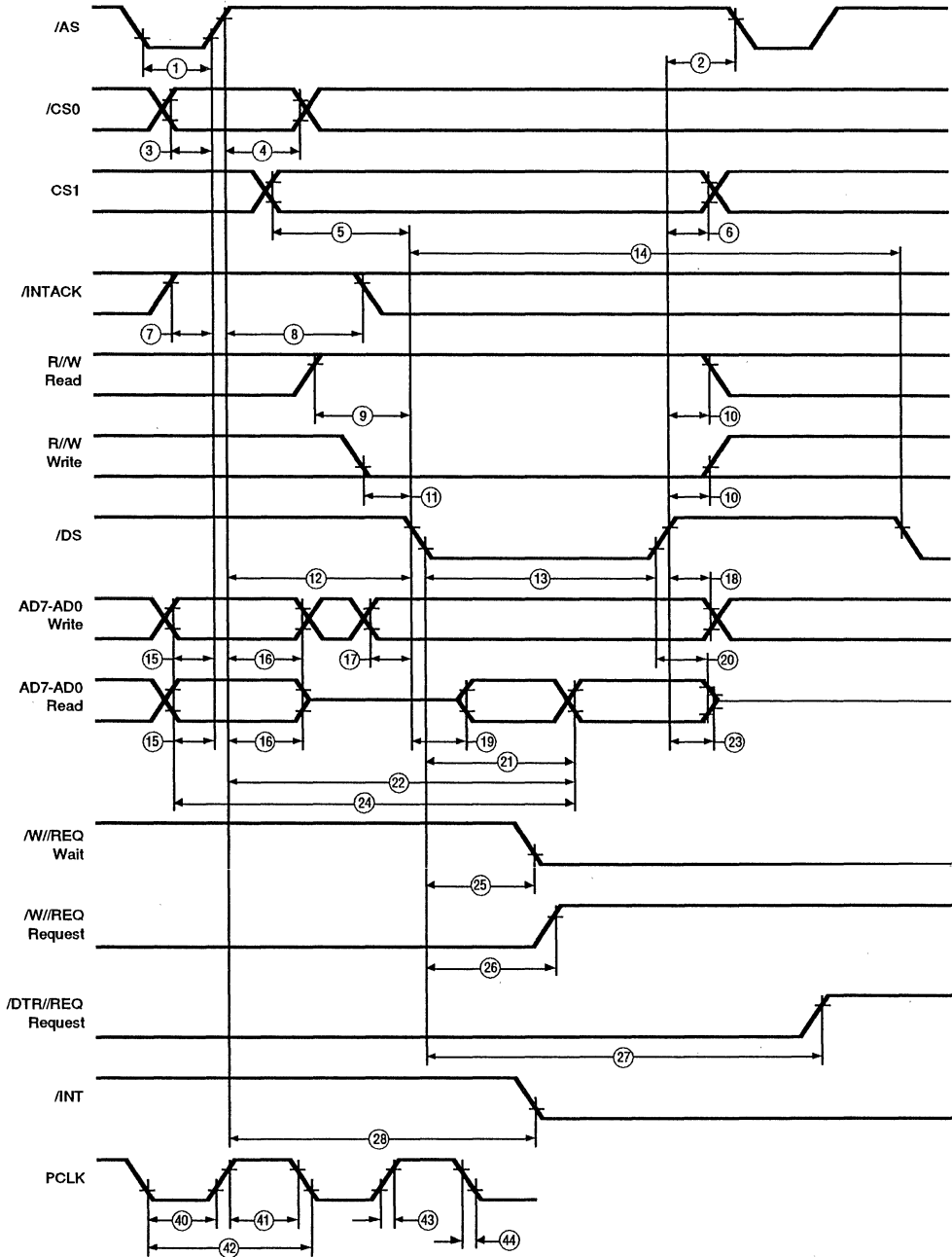


Figure 25. Read and Write Timing Diagram

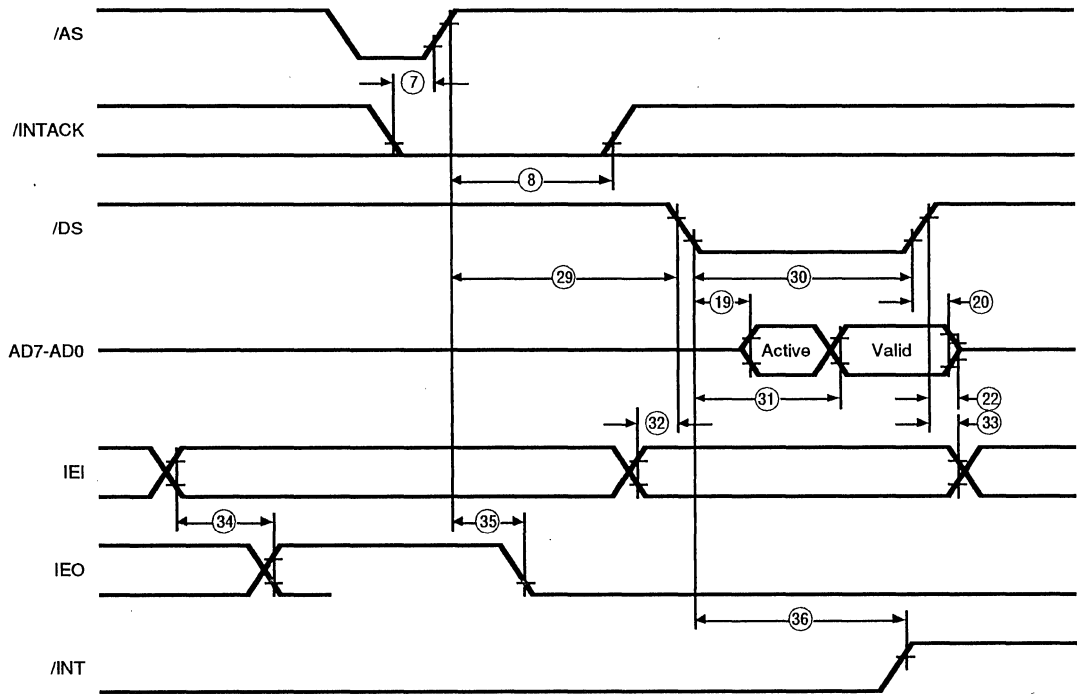


Figure 26. Interrupt Acknowledge Timing Diagram

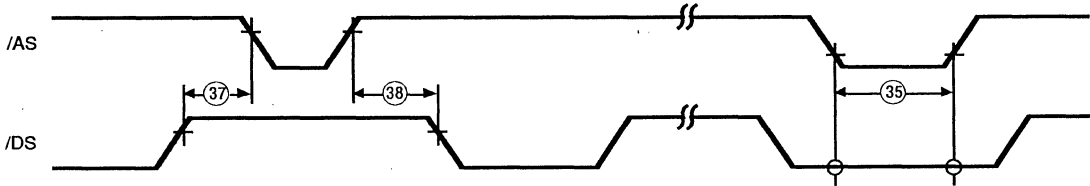


Figure 27. Reset Timing Diagram

## AC CHARACTERISTICS

### Z80230 Read/Write Timing Table

No	Symbol	Parameter	10 MHz		16 MHz		Notes*
			Min	Max	Min	Max	
1	TwAS	/AS Low Width	30		20		
2	TdDS(AS)	/DS Rise to /AS Fall Delay	10		10		[1]
3	TsCSO(AS)	/CS0 to /AS Rise Setup Time	0		0		[1]
4	ThCSO(AS)	/CS0 to /AS Rise Hold Time	20		15		[1]
5	TsCS1(DS)	CS1 to /DS Fall Setup Time	50		35		[1]
6	ThCS1(DS)	CS1 to /DS Rise Hold Time	20		10		[1]
7	TsIA(AS)	/INTACK to /AS Rise Setup Time	10		10		
8	ThIA(AS)	/INTACK to /AS Rise Hold Time	125		100		
9	TsRWR(DS)	R/W (Read) to /DS Fall Setup Time	50		30		
10	ThRW(DS)	R/W to /DS Rise Hold Time	0		0		
11	TsRWW(DS)	R/W (Write) to /DS Fall Setup Time	0		0		
12	TdAS(DS)	/AS Rise to /DS Fall Delay	20		15		
13	TwDSI	/DS Low Width	125		80		
14	TrC	Valid Access Recovery Time	4TcPc		4TcPc		[2]
15	TsA(AS)	Address to /AS Rise Setup Time	10		10		[1]
16	ThA(AS)	Address to /AS Rise Hold Time	20		10		[1]
17	TsDW(DS)	Write Data to /DS Fall Setup Time	10		10		
18	ThDW(DS)	Write Data to /DS Rise Hold Time	0		0		
19	TdDS(DA)	/DS Fall to Data Active Delay	0		0		
20	TdDSr(DR)	/DS Rise to Read Data Not Valid Delay	0		0		
21	TdDSI(DR)	/DS Fall to Read Data Valid Delay		120	70		
22	TdAS(DR)	/AS Rise to Read Data Valid Delay		190	110		
23	TdDS(DRz)	/DS Rise to Read Data Float Delay		35	20		[3]
24	TdA(DR)	Address Required Valid to Read Data Valid Delay		210	100		
25	TdDS(W)	/DS Fall to Wait Valid Delay		160	60		[4]
26	TdDSI(REQ)	/DS Fall to M//REQ Not Valid Delay		160	60		
27	TdDSr(REQ)	/DS Fall to /DTR//REQ Not Valid Delay		4TcPc	4TcPc		
28	TdAS(INT)	/AS Rise to /INT Valid Delay		500	175		[4]
29	TdAS(DSA)	/AS Rise to /DS Fall (Acknowledge) Delay	225		50		[5]
30	TwDSA	/DS (Acknowledge) Low Width	125		75		
31	TdDSA(DR)	/DS Fall (Acknowledge) to Read Data Valid Delay	120		70		
32	TsIEI(DSA)	IEI to /DS Fall (Acknowledge) Setup Time	80		50		
33	ThIEI(DSA)	IEI to /DS Rise (Acknowledge) Hold Time	0		0		
34	TdIEI(IEO)	IEI to IEO Delay		90	45		
35	TdAS(IEO)	/AS Rise to IEO Delay		175	80		[6]
36	TdDSA(INT)	/DS Fall (Acknowledge) to /INT Inactive Delay		450	200		[4]
37	TdDS(ASQ)	/DS Rise to /AS Fall Delay for No Reset	15		10		
38	TdASQ(DS)	/AS Rise to /DS Fall Delay for No Reset	15		10		
39	TwRES	/AS and /DS Coincident Low for Reset	100		75		[7]
40	TwPCI	PCLK Low Width	40	100	26	1000	

---

## AC CHARACTERISTICS

### Z80230 Read/Write Timing Table (Continued)

---

No	Symbol	Parameter	10 MHz		16 MHz		Notes*
			Min	Max	Min	Max	
41	TwPCh	PCLK High Width	40	1000	26	1000	
42	TcPC	PCLK Cycle Time	100	2000	61	2000	
43	TrPC	PCLK Rise Time		10		5	
44	TfPC	PCLK Fall Time		10		5	

---

#### Notes:

[1] Parameter does not apply to Interrupt Acknowledge transactions.

[2] Parameter applies only between transactions involving the ESCC.

[3] Float delay is defined as the time required for a  $\pm 0.5V$  change in the output with a maximum DC load and a minimum AC load.

[4] Open-drain output, measured with open-drain test load.

[5] Parameter is system dependent. For any Zilog ESCC in the daisy chain, TdAS(DSA) must be greater than the sum of TdAS(IEO) for the highest priority device in the daisy chain, TslE(DSA) for the Zilog ESCC, and TdIEI(IEO) for each device separating them in the daisy chain.

[6] Parameter applies only to a Zilog ESCC pulling INT Low at the beginning of the Interrupt Acknowledge transaction.

[7] Internal circuitry allows for the reset provided by the Z8<sup>®</sup> to be recognized as a reset by the Z-ESCC. All timing references assume 2.0V for a logic 1 and 0.8V for a logic 0.

\* Units in nanoseconds (ns).

**AC CHARACTERISTICS**  
Z80230 General Timing Diagram

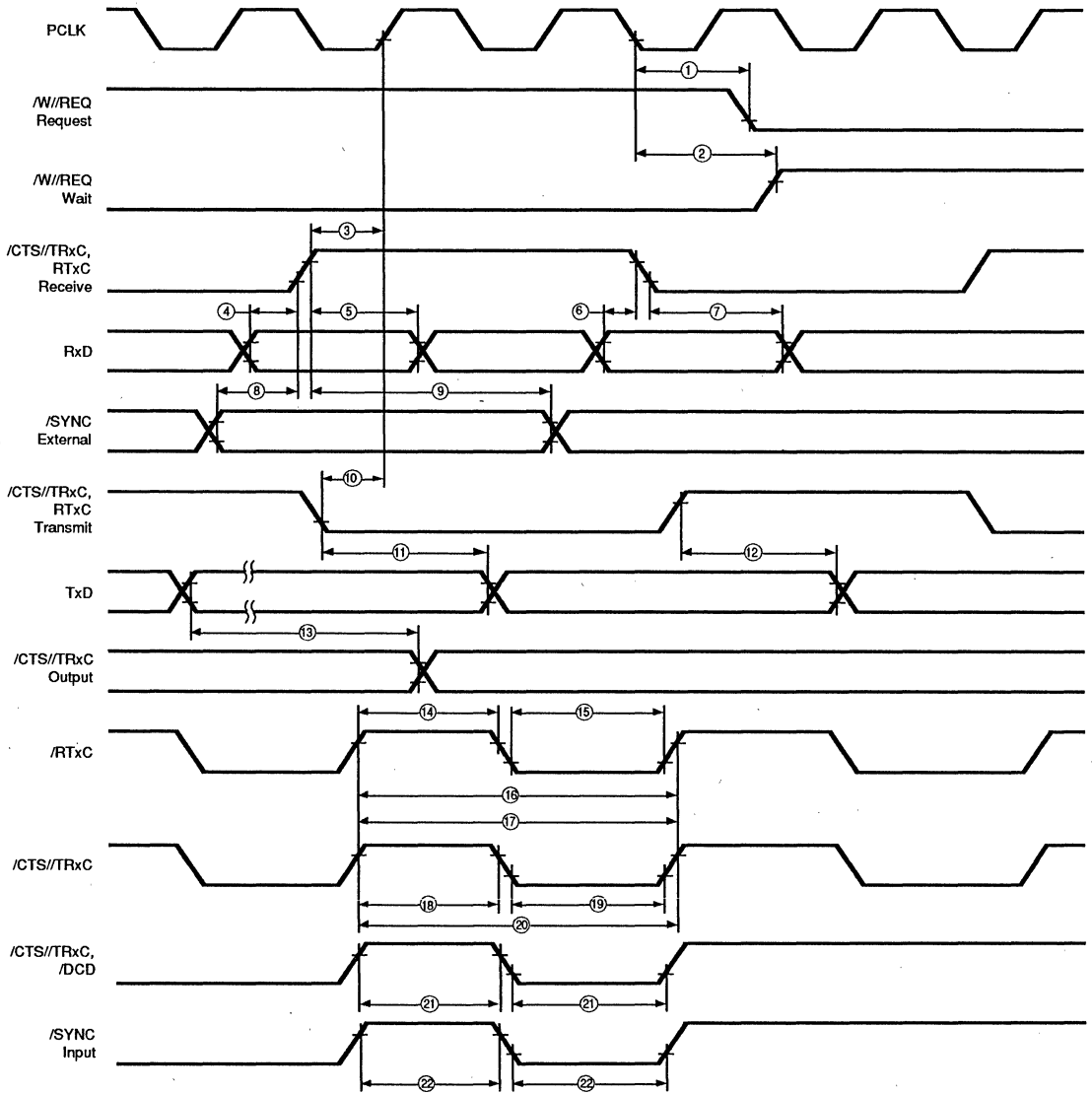


Figure 28. General Timing Diagram

## AC CHARACTERISTICS

### Z80230 General Timing Table

No	Symbol	Parameter	10 MHz		16 MHz		Notes*
			Min	Max	Min	Max	
1	TdPC(REQ)	/PCLK Low to W/REQ Valid		200		110	
2	TsPC(W)	/PCLK Low to Wait Inactive		300		180	
3	TsRXC(PC)	/RxC High to /PCLK High Setup Time	NA		NA		[1,4]
4	TsRXD(RXCr)	RxD to /RxC High Setup Time	0		0		[1]
5	ThRXD(RxCr)	RxD to /RxC High Hold Time	125		60		[1]
6	TsRXD(RXCf)	RxD to /RxC Low Setup Time	0		0		[1,5]
7	ThRXD(RXCf)	RxD to /RxC Low Hold Time	125		60		[1,5]
8	TsSY(RXC)	SYNC to /RxC High Setup Time	-150		-100		[1]
9	ThSY(RXC)	SYNC to /RxC High Hold Time	5TcPc		5TcPc		[1]
10	TsTXC(PC)	/TxC Low to /PCLK High Setup Time	NA		NA		[2,4]
11	TdTXCf(TXD)	/TxC Low to TxD Delay		150		85	[2]
12	TdTxCr(TXD)	/TxC High to TxD Delay		150		85	[2,5]
13	TdTXD(TRX)	TxD to TRxC Delay		140		80	
14	TwRTXh	RTxC High Width	120		80		[6]
15	TwRTXI	TRxC Low Width	120		80		[6]
16a	TcRTX	RTxC Cycle Time	400		244		[6,7]
16b	TxRX(DPLL)	DPLL Cycle Time Min	50		31		[7,8]
17	TcRTXX	Crystal Osc. Period	100	1000	100	1000	[3]
18	TwTRXh	TRxC High Width	120		80		[6]
19	TwTRXI	TRxC Low Width	120		80		[6]
20	TcTRX	TRxC Cycle Time	400		244		[6,7]
21	TwEXT	DCD or CTS Pulse Width	120		70		
22	TwSY	SYNC Pulse Width	120		70		

#### Notes:

- [1] RxC is /RTxC or /TRxC, whichever is supplying the receive clock.
- [2] TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.
- [3] Both /RTxC and /SYNC have 30 pF capacitors to ground connected to them.
- [4] Synchronization of RxC to PCLK is eliminated in divide-by-four operation.
- [5] Parameter applies only to FM encoding/decoding.
- [6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator timing requirements are identical to case PCLK requirements.
- [7] The maximum receive or transmit data rate is 1/4 PCLK.
- [8] Applies to DPLL clock source only. Maximum data rate of 1/4 PCLK still applies. DPLL clock should have a 50% duty cycle.

\* Units in nanoseconds (ns).

## AC CHARACTERISTICS

### Z80230 System Timing Table

No	Symbol	Parameter	10 MHz		16 MHz		Notes*
			Min	Max	Min	Max	
1	TdRXC(REQ)	/RxC High to W/REQ Valid	13	17	13	17	[2]
2	TdRXC(W)	/RxC High to Wait Inactive	13	19	13	19	[1,2]
3	TdRdXC(SY)	/RxC High to SYNC Valid	9	12	9	12	[2]
4b	TdRXC(INT), Z80230	/RxC High to INT Valid	13	17	13	17	[1,2]
			2	3	+2	+3	[4]
5	TdTXC(REQ)	/TxC Low to W/REQ Valid	8	11	8	11	[3]
6	TdTXC(W)	/TxC Low to Wait Inactive	8	14	8	14	[1,3]
7	TdTXC(DRQ)	/TxC Low to DTR/REQ Valid	7	10	7	10	[3]
8b	TdTXC(INT), Z80230	/TxC Low to /INT Valid	7	9	7	9	[1,3]
			+2	+3	+2	+3	[4]
9a	TdSY(INT)	SYNC to INT Valid	2	6	2	6	[1]
9b	TdSY(INT)	SYNC to INT Valid	2	3	2	3	[1,4]
10b	TdEXT(INT), Z80230		2	3	3	8	[1,4]

#### Notes:

[1] Open drain-output, measured with open-drain test load.

[2] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.

[3] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[4] Units equal to /AS.

\* Units equal to TcPc.



## Z16C35/Z85C35

### CMOS ISCC™ INTEGRATED SERIAL COMMUNICATIONS CONTROLLER

#### FEATURES

- Two general-purpose SCC channels, four DMA channel; and a Universal Bus Interface Unit.
  - Software compatible to the Zilog CMOS SCC
  - Four DMA channels; two transmit and two receive channels to and from the SCC.
  - Four gigabyte address range per DMA channel
  - Flyby DMA transfer mode
  - Programmable DMA channel priorities
  - Independent DMA register set
  - A Universal Bus Interface Unit providing a simple interface to most CPUs with a multiplexed or non-multiplexed bus; compatible with 680x0 and 8x86 CPUs.
  - 32-bit addresses multiplexed to 16-pin address/data lines
  - 8-bit data supporting high/low byte swapping
  - 10 and 16 MHz timing
  - 68-pin PLCC
- Supports all Zilog CMOS SCC features:
- Two independent, 0 to 4.0 Mbit/second, full-duplex channels, each with a separate crystal oscillator, baud rate generator, and digital phase-locked loop circuit for clock recovery.
  - Multi-protocol operation under program control; programmable for NRZ, NRZI, or FM data encoding.
  - Asynchronous mode with five to eight bits and one, one and one-half, or two stop bits per character; programmable clock factor; break detection and generation; parity, overrun, and framing error detection.
  - Synchronous mode with internal or external character synchronization on one or two synchronous characters and CRC generation and checking with CRC-16 or CRC-CCITT preset to either 1's or 0's.
  - SDLC/HDLC mode with comprehensive frame-level control, automatic zero insertion and deletion, I-field residue handling, abort generation and detection, CRC generation and checking, and SDLC Loop mode operation.
  - Local Loopback and Auto Echo modes
  - Supports T1 digital trunk
  - Enhanced SDLC 10x19 Status FIFO for DMA support

#### GENERAL DESCRIPTION

The Z85C35, directly equivalent to the Z16C35 ISCC, is a CMOS superintegrated device with a flexible Bus Interface Unit (BIU) connecting a built-in Direct Memory Access (DMA) cell to the CMOS Serial Communications Control (SCC) cell.

The ISCC is a dual-channel, multi-protocol data communications peripheral which easily interfaces to CPU's with

either multiplexed or non-multiplexed address and data buses. The advanced CMOS process offers lower power consumption, higher performance, and superior noise immunity. The programming flexibility of the internal registers allow the ISCC to be configured for a wide variety of serial communications applications. The many on-chip features such as, streamlined bus interface, four channel



## GENERAL DESCRIPTION (Continued)

DMA, baud rate generators, digital phase-locked loops, and crystal oscillators dramatically reduce the need for external logic. Additional features, including a 10x19 bit status FIFO, are added to support high speed SDLC transfers using on-chip DMA controllers (Figure 1).

The ISCC can address up to four gigabytes per DMA channel by using the /UAS and /AS signals to strobe out 32-bit multiplexed addresses.

The ISCC handles asynchronous formats, synchronous byte-oriented protocols such as IBM Bisync, and synchronous bit-oriented protocols such as HDLC and IBM SDLC. This versatile device supports virtually any serial data transfer application (terminals, printers, diskette, tape drives, etc.).

The device can generate and check CRC codes in any synchronous mode and can be programmed to check data integrity in various modes. The ISCC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

The standard Zilog interrupt daisy chain is supported for interrupt hierarchy control. Internally, the SCC cell has higher interrupt priority than the DMA cell.

The DMA cell consists of four DMA channels; one for transmit and one for receive to and from each SCC channel, respectively. The cycle time for each DMA transfer is 400 ns for the 10 MHz version. There is no idle cycle between DMA transfers.

The DMA cell adopts a simple fly-by mode DMA transfer, allowing easy programming of the DMA cell and yet providing a powerful and efficient DMA access. The cell does not support memory-to-memory transfer.

Priorities between the four DMA channels are programmable to custom-fit user applications. Arbitration of Bus priority control signals between the ISCC DMA and other system DMA's should be handled outside the ISCC.

The BIU has a universal interface to most system/CPU bus structures and timing. The first write to the ISCC after a hardware reset will confirm the bus interface type being implemented.

**Note:** All Signals with a preceding front slash, "/", are active Low, e.g.: B/W (WORD is active Low); /B/W (BYTE is active Low, only); /N/S (NORMAL and SYSTEM are both active Low).

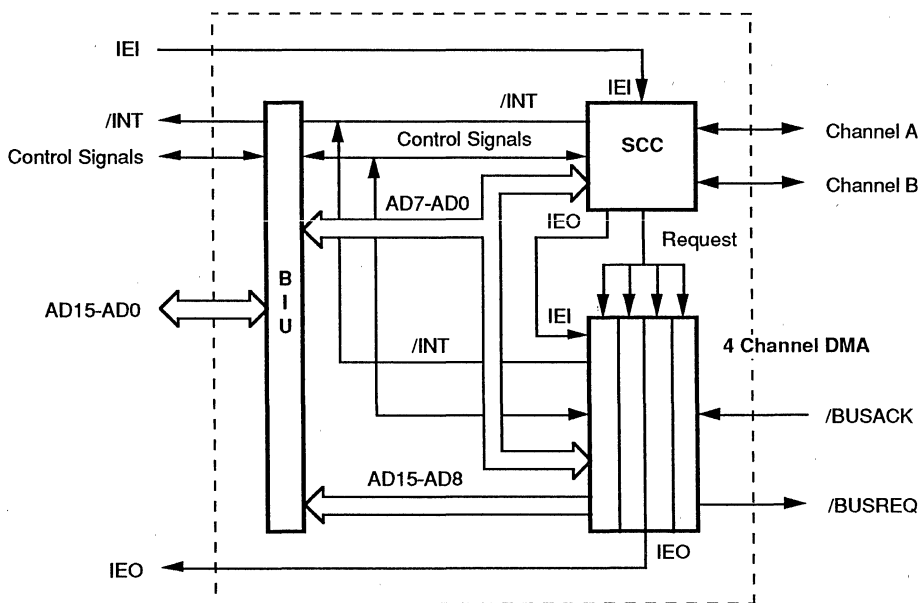


Figure 1. Block Diagram

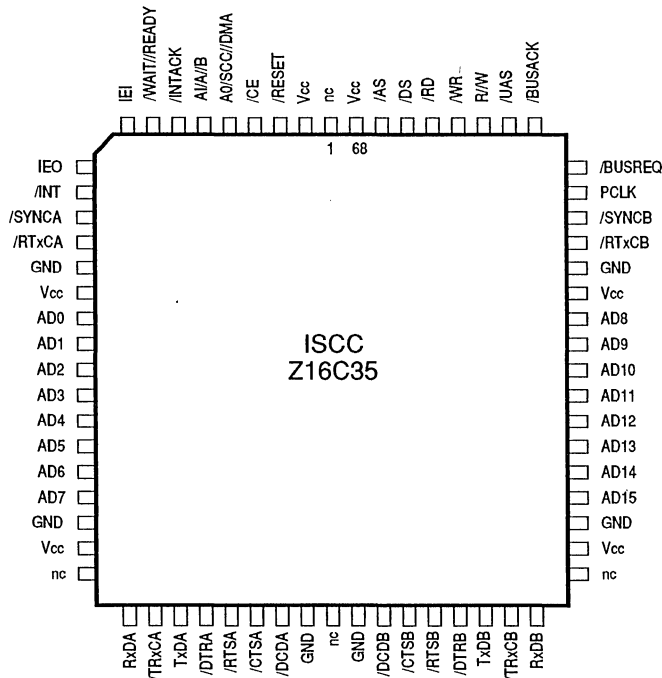


Figure 2. Pin Assignments

## PIN DESCRIPTION

The following section describes the Z16C35 pin functions. Figure 2 details the respective pin functions and pin assignments. All references to DMA are internal.

**/CTSA, /CTSB.** *Clear To Send* (inputs, active Low). If these pins are programmed as Auto Enables, a Low on the inputs enables the respective transmitters. If not programmed as Auto Enables, they may be used as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The SCC cell detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

**/DCDA, /DCDB.** *Data Carrier Detect* (inputs, active Low). These pins function as receiver enables if they are programmed for Auto Enables; otherwise they are used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow rise time signals. The SCC cell detects pulses on these pins and can interrupt the CPU on both logic level transitions.

**/DTRA, /DTRB.** *Data Terminal Ready* (outputs, active Low). These outputs follow the state programmed into the DTR bit.

**IEI.** *Interrupt Enable In* (input, active High). IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt driven device. A high IEI indicates that no other higher priority device has an interrupt under service or is requesting an interrupt. The SCC cell has a higher interrupt priority than the DMA cell.

**IEO.** *Interrupt Enable Out* (output, active High). IEO is High only if IEI is High and the CPU is not servicing the ISCC (SCC or DMA) interrupt, or the ISCC is not requesting an interrupt (Interrupt Acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

**/INT.** *Interrupt* (output, active Low). This signal is activated when the SCC or DMA requests an interrupt. Note that /INT is pulled high and is not an open-drain output. This signal tristates at reset.

---

## PIN DESCRIPTION (Continued)

**/INTACK.** *Interrupt Acknowledge* (input, active Low). This is a strobe which indicates that an interrupt acknowledge cycle is in progress. During this cycle, the SCC and DMA interrupt daisy chain is resolved. The device is capable of returning an interrupt vector that may be encoded with the type of interrupt pending during this acknowledge cycle when RD or DS become high. INTACK may be programmed to accept a status acknowledge, a single pulse acknowledge, or a double pulse acknowledge. This is programmed in the Bus Configuration Register (BCR). The double pulse acknowledge is compatible with 8x86 family microprocessors.

**PCLK.** *Clock* (input). This is the master SCC and DMA clock used to synchronize internal signals. PCLK is a TTL level signal. PCLK is not required to have any phase relationship with the master system clock.

**RxDA, RxDB.** *Receive Data* (inputs, active High). These input signals receive serial data at standard TTL levels.

**/RTxCA, /RTxCB.** *Receive/Transmit Clocks* (inputs, active Low). These pins can be programmed to several modes of operation. In each channel, RTxC may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the Digital Phase-Locked Loop. These pins can also be programmed for use with the respective SYNC pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

**/RTSA, /RTSB.** *Request To Send* (outputs, active Low). When the Request To Send (RTS) bit in Write Register 5 is set, the RTS signal goes Low. When the RTS bit is reset in the Asynchronous mode and Auto Enable is on, the signal goes High after the transmitter is empty. In Synchronous mode or in Asynchronous mode with Auto Enable off, the RTS pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

**/SYNCA, /SYNCB.** *Synchronization* (inputs or outputs, active Low). These pins can act either as inputs, outputs, or part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to CTS and DCD. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode, SYNC must be driven Low to receive clock cycles after the last bit in the synchronous character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of SYNC.

In the Internal Synchronization mode (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which synchronous condition is not latched. These outputs are active each time a synchronization pattern is recognized (regardless of character boundaries). In SDLC mode, the pins act as outputs and are valid on receipt of a flag.

**TxDA, TxDB.** *Transmit Data* (outputs, active high). These output signals transmit serial data at standard TTL levels.

**/TRxCA, /TRxCB.** *Transmit/Receive Clocks* (inputs or outputs, active Low). These pins can be programmed in several different modes of operation. TRxC may supply the receive clock or the transmit clock in the input mode or supply the output of the Digital Phase-Locked Loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode.

**/CE.** *Chip Enable* (input, active Low). This signal selects the ISCC for a peripheral read or write operation. This signal is not used when the ISCC is bus master.

**AD15-AD0.** *Data bus* (bidirectional, 3-state). These lines carry data and commands to and from the ISCC.

**/RD.** *Read* (bidirectional, active Low). When the ISCC is a peripheral (i.e. bus slave), this signal indicates a read operation and when the ISCC is selected, enables the ISCC's bus drivers. As an input, /RD indicates that the CPU wants to read from the ISCC read registers. During the Interrupt Acknowledge cycle, /RD gates the interrupt vector onto the bus if the ISCC is the highest priority device requesting an interrupt. When the ISCC is the bus master, this signal is used to read data. As an output, after the ISCC has taken control of the system buses, /RD indicates a DMA-controlled read from a memory or I/O port address.

**/WR.** *Write* (bidirectional, active Low). When the ISCC is selected, this signal indicates a write operation. As an input, this indicates that the CPU wants to write control or command bytes to the ISCC write registers. As an output, after the ISCC has taken control of the system buses /WR indicates a DMA-controlled write to a memory or I/O port address.

**/DS.** *Data Strobe* (bidirectional, active Low). A Low on this signal indicates that the AD15-AD0 bus is used for data transfer. When the ISCC is not in control of the system bus and the external system is transferring information to or from the ISCC, /DS is a timing input used by the ISCC to move data to or from the AD15-AD0 bus. Data is written into the ISCC by the external system on the Low to High /DS transition. Data is read from the ISCC by the external

system while /DS is Low. There are no timing requirements between /DS as an input and ISCC clock; this allows use of the ISCC with a system bus which does not have a bussed clock.

During a DMA operation when the ISCC is in control of the system, DS is an output generated by the ISCC and used by the system to move data to or from the AD15-AD0 bus. When the ISCC has bus control, it writes to the external system by placing data on the AD15-AD0 bus before the High-to-Low DS transition and holds the data stable until after the Low-to-High DS transition; while reading from the external system, the Low-to-High transition of DS inputs data from the AD15-AD0 bus into the ISCC.

**R/W.** *Read/Write* (bidirectional). Read polarity is High and write polarity is Low. When the ISCC is bus master, R/W indicates the data direction of the current bus transaction, and is stable from when AS is High until the bus transaction ends. When the ISCC is not in control of the system bus and the external system is transferring information to or from the ISCC, R/W is a status input used by the ISCC to determine if data is entering or leaving on the AD15-AD0 bus during /DS time. In such a case, Read (High) indicates that the system is requesting data from the ISCC and Write (Low) indicates that the system is presenting data to the ISCC. The only timing requirements for R/W as an input are defined relative to DS. When the ISCC is in control of the system bus, R/W is an output generated by the ISCC, with Read indicating that data is being requested from the addressed location or device, and Write indicating that data is being presented to the addressed location or device.

**/UAS.** *Upper Address Strobe* (Output, active Low). This signal is used if the address is more than 16-bit. The upper address, A31-A16, can be latched externally by the rising edge of this signal. /UAS is active first before AS becomes active. This signal and AS are used by the DMA cell.

**/IAS.** *Lower Address Strobe* (Bidirectional, active Low). When the ISCC is bus master, this signal when an output, is used as a lower address strobe for AD15-AD0. It is used in conjunction with UAS since the address is 32-bits. This signal and /UAS are used by the DMA cell when it is bus master. When ISCC is not bus master, this signal is used in the multiplexed bus modes to latch the address on the AD lines. The /IAS signal is not used in the non-multiplexed bus modes and should be tied to Vcc in these cases.

**/WAIT//RDY.** *Wait/Ready* (bidirectional, active Low). It may be programmed to function either as a Wait signal or Ready signal during the BCR write. When the BCR is written to Channel A (A1/A//B High during the BCR write), this signal functions as a WAIT and thus supports the READY function of 8X86 microprocessors family. When

the BCR writes to Channel B (A1/A//B Low), this signal functions as a READY and supports the DTACK function of the 680X0 microprocessor family.

This signal is an output when the ISCC is not bus master. In this case, the Wait/RDY signal indicates when the data is available during a read cycle; when the device is ready to receive data during a write cycle; and when a valid vector is available during an interrupt acknowledge cycle.

When the ISCC is the bus master (the DMA cell has taken control of the bus), the /Wait//RDY signal functions as a WAIT or READY input. Slow memories and peripheral devices can assert WAIT to extend /DS during bus transfers. Similarly, memories and peripherals use READY to indicate that its output is valid or that it is ready to latch input data.

**/BUSACK.** *Bus Acknowledge* (input, active Low). Signals the bus has been released to the DMA. If the /BUSACK is inactive before the DMA transfer is completed, the current DMA transfer is aborted. This signal tristates at reset.

**/BUSREQ.** *Bus Request* (output, active Low). This signal is used by the DMA to obtain the bus from the CPU.

**A0/SCC//DMA.** *DMA Channel/SCC Select/DMA Select* (bidirectional). When this pin is used as input, a high selects the SCC cell and a low selects the DMA cell. When this pin is used as output, the signal on this pin is used in conjunction with A1/A//B pin output to identify which DMA channel is active. This information can be used by the user to determine whether to issue a DMA abort command. A0/SCC//DMA and A1/A//B output encoding is shown below:

A1/A//B	A0/SCC//DMA	DMA channel
1	1	RxA
1	0	TxA
0	1	RxB
0	0	TxB

**A1/A//B.** *DMA Channel/Channel A/Channel B* (bidirectional). This signal, when used as input, selects the SCC channel in which the read and write operation occurs. Note that A0/SCC//DMA pin must be held high to select this feature. When this pin is used as an output, it is used in conjunction with the A0/SCC//DMA pin output to identify which DMA channel is active. During a DMA peripheral access, the A1/A//B pin is ignored.

**/RESET.** (input, active Low). This signal resets the device to a known state. The first write to the ISCC after a reset accesses the BCR to select additional bus options for the device.

## FUNCTIONAL DESCRIPTION

The functional capabilities of the ISCC are described in three blocks: the SCC cell, the DMA cell, and the Bus Interface Unit (BIU). Each of the blocks are described independently in the following sections with the ISCC

architecture shown in Figure 3. Please refer to the ISCC Technical Manual for a detailed description of the functions outlined here.

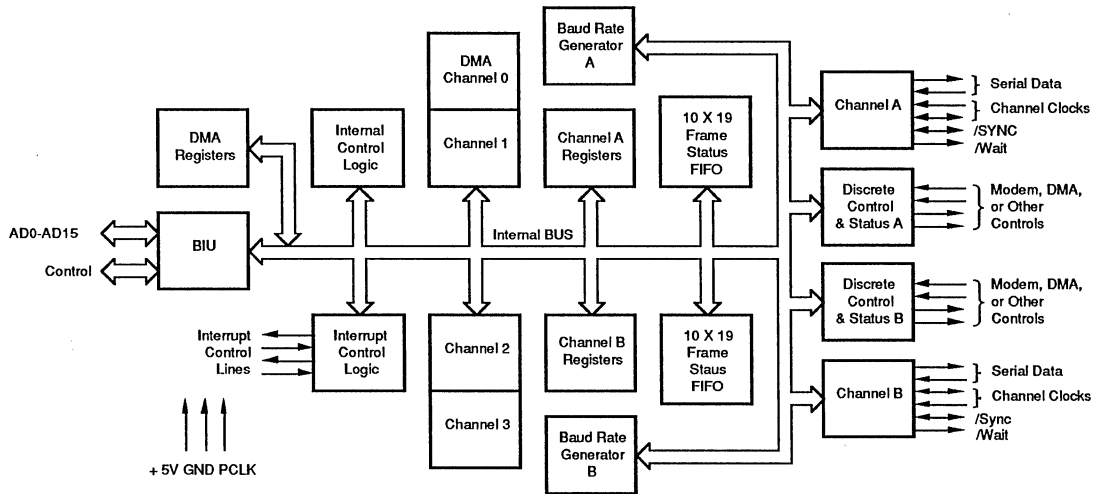


Figure 3. Block Diagram of ISCC Architecture

**SCC Cell Data Communications Capabilities.** The ISCC provides two independent full-duplex programmable channels for use in any common asynchronous or synchronous data communications protocol. The ISCC is built from Zilog's industry standard SCC core and is compatible with designs using Zilog's SCC to receive and transmit data (Figure 4).

**Asynchronous Modes.** Send and Receive can be accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one-and-a-half, or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB in Figure 2). If the Low does not persist (e.g., a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur.

Vectored interrupts allow fast servicing of error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

The ISCC does not require symmetric transmit and receive clock signals - a feature allowing use of the wide variety of clock sources. The transmitter and receiver can handle data at a rate supplied to the receive and transmit clock inputs. In Asynchronous modes, the SYNC pin may be programmed as an input used for functions such as monitoring a ring indicator.

**Synchronous Modes.** The ISCC supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols can be handled in several modes, allowing character synchronization with a 6-bit or 8-bit synchronous character (Monosync), and 12-bit synchronization pattern (Bisync), or with an external synchronous signal. Leading sync characters can be removed without interrupting the CPU.

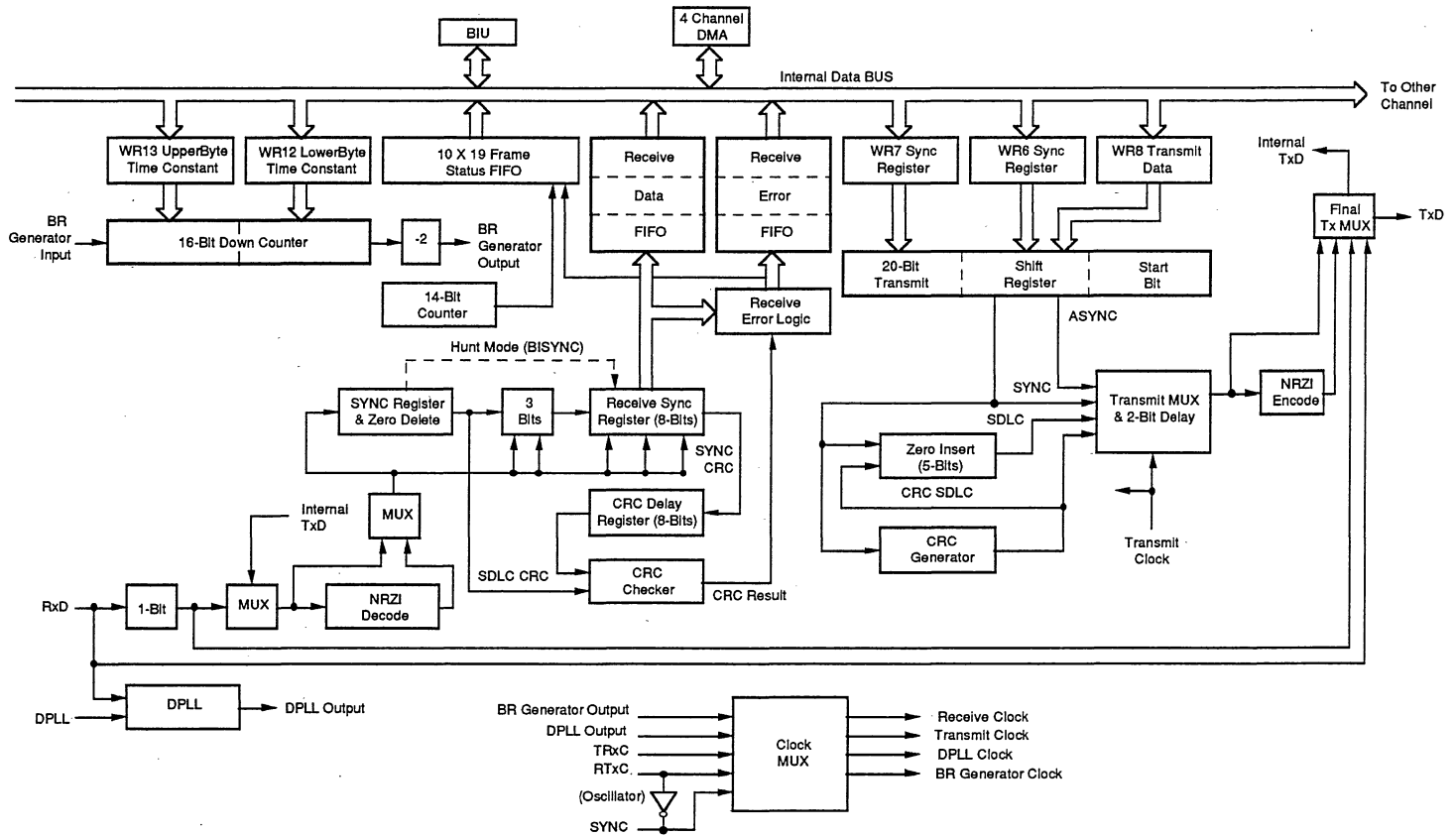


Figure 4. ISCC Data Path

## FUNCTIONAL DESCRIPTION (Continued)

Five or 7-bit synchronous characters are detected with 8- or 16-bit patterns in the ISCC by overlapping the larger

pattern across multiple incoming synchronous characters as shown in Figure 5.

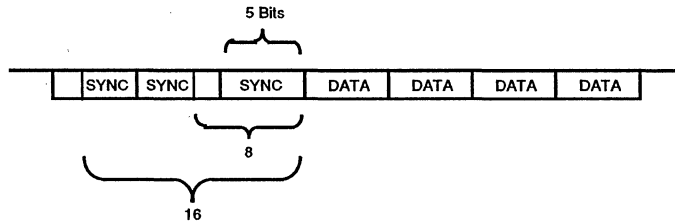


Figure 5. Detecting 5- or 7-Bit Synchronous Characters

CRC checking for Synchronous byte oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols such as IBM Bisync.

Both CRC-16 ( $X^{16} + X^{15} + X^2 + 1$ ) and CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) error checking polynomials are supported. Either polynomial may be selected in all Synchronous modes. Users may preset the CRC generator and checker to all 1's or all 0's. The ISCC also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows for high speed transmissions under DMA control, with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 6-, 8-, or 16-bit synchronous characters, regardless of the programmed character length.

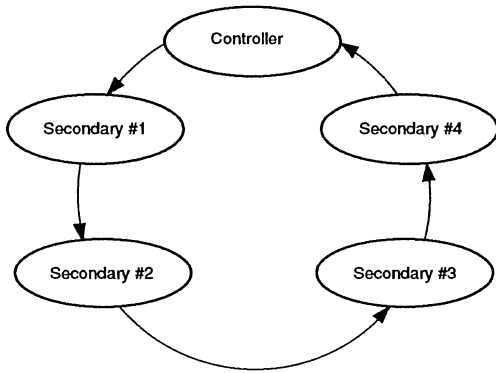
The ISCC supports Synchronous bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero insertion, and CRC generation. A special command is used to abort a frame in transmission. At the end of a message, the ISCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line consisting of continuous flag characters or a steady marking condition.

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. The ISCC may also be programmed to send an abort itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent, allowing reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the SYNC pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all 0's inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. In SDLC mode, the ISCC must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all 1's or all 0's. The CRC is inverted before transmission and the receiver checks against the bit pattern 0001110100001111.

NRZ, NRZI or FM coding may be used in any 1x mode. The parity options available in Asynchronous modes are available in Synchronous modes.

**SDLC Loop Mode.** The ISCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow on the loop and any number of secondary stations. In SDLC Loop mode, the ISCC performs the functions of a secondary station while an ISCC operating in regular SDLC mode acts as a controller (Figure 6).



**Figure 6. An SDLC Loop**

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop and, in fact, passes these messages to the rest of the loop by retransmitting them with a one-bit-time delay. The secondary station places its own message on the loop only at specific times. The controller signals that secondary stations can transmit messages by sending a special character, called an EOP (End Of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages, this bit pattern is unique and easily recognized.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit append their messages to the message of the first secondary station by the same process. Any secondary stations without messages to send merely echo the incoming message and are prohibited from placing messages on the loop (except upon recognizing an EOP.)

SDLC Loop mode is a programmable option in the ISCC. NRZ, NRZI, and FM coding may all be used in SDLC Loop mode.

**SDLC FIFO.** The ISCC's ability to receive high speed back-to-back SDLC frames is maximized by a 10-bit deep by 19-bit wide status FIFO. When enabled (through WR15, bit D2), it provides the DMA the ability to continue to transfer data into memory so that the CPU can examine the message later. For each SDLC frame, a 14-bit byte count

and 5 status/error bits are stored. The byte count and status bits are accessed through Read Registers 6 and 7. Read Registers are only accessible when the SDLC FIFO is enabled. The 10x19 status FIFO is separate from the 3 byte receive data FIFO.

**Notes on the SDLC FIFO.** When using the SDLC FIFO enhancement in channel B, it is necessary to enable the enhancement in channel A. There is no special requirement to enable the enhancement in channel A only, or to use it in both channels. Designs using only one channel should, therefore, use channel A.

When an SDLC frame is received with an abort condition, the byte counter in the FIFO enhancement is not reset. Therefore, after the abort is received, a dummy frame consisting of a flag should be sent by the transmitter. This resets the byte counter for the next frame. The aborted frame has a byte count which includes the byte count of the next dummy frame.

**Baud Rate Generator.** Each channel in the ISCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On startup, the flip-flop on the output is set in a High state, the value in the time constant register is loaded into the counter, and the counter starts counting down. The output of the baud rate generator toggles upon reaching 0, the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the Digital Phase-Locked Loop (see next section).

If the receive clock or transmit clock is not programmed to come from the TRxC pin, the output of the baud rate generator may be echoed out via the TRxC pin.

The following formula relates the time constant to the baud rate where PCLK or RTxC is the baud rate generator input frequency in Hertz. The clock mode is 1, 16, 32, or 64, as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select 1 and Asynchronous should select 16, 32 or 64.

$$\text{Time Constant} = \frac{\text{PCLK or RTxC Frequency}}{2(\text{Baud Rate})(\text{Clock Mode})} - 2$$



---

## FUNCTIONAL DESCRIPTION (Continued)

**Digital Phase-Locked Loop.** The ISCC contains a Digital Phase-Locked Loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock is then used as the ISCC receive clock, the transmit clock, or both.

For NRZI encoding, the DPLL counts the 32x clock to create nominal bit times. As the 32x clock is counted, the DPLL is searching the incoming data stream for edges (either 1 to 0, or 0 to 1). Whenever an edge is detected, the DPLL makes a count adjustment (during the next counting cycle), producing a terminal count closer to the center of the bit cell.

For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16 and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15 to 16 counting transition.

The 32x clock for the DPLL can be programmed to come from either the RTxC input or the output of the baud rate

generator. The DPLL output may be programmed to be echoed out of the ISCC via the TRxC pin (if this pin is not being used as an input).

**Data Encoding.** The ISCC may be programmed to encode and decode the serial data in four different ways (Figure 7). In NRZ encoding, a 1 is represented by a High level and a 0 is represented by a Low level. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. In FM1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A 1 is represented by an additional transition at the center of the bit cell and a 0 is represented by no additional transition at the center of the bit cell. In FM0 (bi-phase space), a transition occurs at the beginning of every bit cell. A 0 is represented by an additional transition at the center of the bit cell, and a 1 is represented by no additional transition at the center of the bit cell. In addition to these four methods, the ISCC can be used to decode Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0 to 1, the bit is a 0. If the transition is 1 to 0, the bit is a 1.

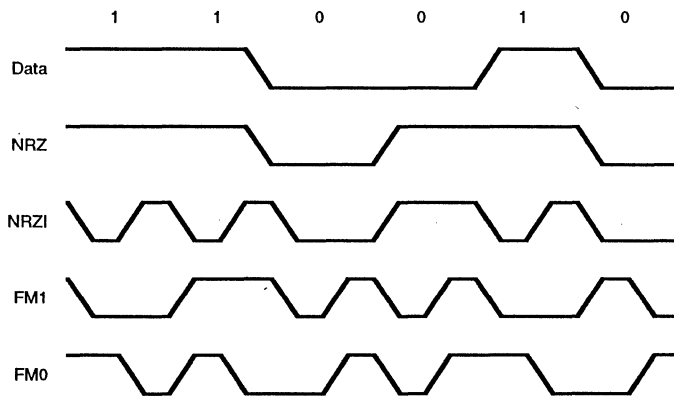


Figure 7. Data Encoding Methods

---

**Auto Echo and Local Loopback.** The ISCC is capable of automatically echoing everything it receives. This feature is useful mainly in Asynchronous modes, but works in Synchronous and SDLC modes as well. In Auto Echo mode, TxD is RxD. Auto Echo mode can be used with NRZI or FM encoding with no additional delay because the data stream is not decoded before retransmission. In Auto Echo

mode, the /CTS input is ignored as a transmitter enable (although transitions on this input can still cause interrupts if programmed to do so). In this mode, the transmitter is actually bypassed and the programmer is responsible for disabling transmitter interrupts and /WAIT//REQUEST on transmit.

---

The ISCC is also capable of local loopback. In this mode TxD is RxD is just like Auto Echo mode. However, in Local Loopback mode the internal transmit data is tied to the internal receive data and RxD is ignored (except to be echoed out via TxD). The /CTS and /DCD inputs are also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in Asynchronous, Synchronous and SDLC modes with NRZ, NRZI or FM coding of the data stream.

**DMA Core.** The ISCC contains four independent fly-by mode DMA channels. Each of the ISCC's transmit and

receive channels has a DMA channel dedicated to it to move data to-and-from memory. The DMA channels are dedicated to the transmit and receive FIFO's, and therefore, can not be used for device initialization. Each DMA has a 32-bit address and a 16-bit byte counter. The DMA address may be incremented or decremented providing flexibility in doing block transfers.

See the I/O Interface Capabilities Section for more details on the DMA features.

---

## BUS INTERFACE UNIT (BIU) DESCRIPTION

The ISCC contains a flexible bus interface that is compatible with a variety of microprocessors and microcontrollers. The device is designed to work with 8- or 16-bit bus systems and may be used with address/data multiplexed busses or non-multiplexed busses. The multiplexed bus is selected for the ISCC if there is an Address Strobe prior to or during the transaction which writes the BCR. If no Address Strobe is present prior to or during the transaction which writes the BCR, a non-multiplexed bus is selected.

When the ISCC is initialized for non-multiplexed operation, register addressing for the ISCC cell is (with the exception of WRO and RRO), accomplished as follows. Programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to WRO which contains four bits that point to the selected register (note point high command). The second write is the actual control word for the selected register. If the second operation is a read, the selected register is accessed. When in the non-multiplexed mode, all of the registers in the SCC cell of the ISCC, including the data registers, are accessed in this fashion. The pointer register is automatically cleared after the second read or write operation so that WRO (or RRO) is addressed again. Note that when the DMA is not used to address the data, the data registers must be accessed by pointing to Register 8. This is in contrast to the Z8530 which allows direct addressing of the data registers through the C/D pin.

When the ISCC is initialized for non-multiplexed operation, register addressing for the DMA cell (with the exception of CSAR) is accomplished as follows and is completely independent of the SCC cell register addressing. Programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to the Command Status Address Register (CSAR) which contains five bits that point to the selected register (CSAR bits 4 - 0). The second write is the actual control word for the selected

register. If the second operation is a read, the selected register is accessed. When in the non-multiplexed mode, all of the registers in the DMA cell of the ISCC may be accessed in this fashion. The pointer bits are automatically cleared after the second read or write operation so that CSAR is addressed again.

When the ISCC is initialized for multiplexed bus operation, all registers in the SCC cell are directly addressable with the register address occupying AD5 through AD1, or AD4 through AD0 (Shift Left / Shift Right modes). Two additional pins, A0/SCC//DMA and A1/A//B control the channel A/B register selection and the SCC channel /DMA selection. Refer to the A0/SCC//DMA and A1/A//B pin descriptions for the encoding of these signals.

The Shift Left / Shift Right modes for the address decoding for the internal registers (multiplexed bus) are separately programmable for the SCC cell and for the DMA cell. For the SCC cell the programming and operation is identical to that in the SCC; programming is accomplished through Write Register 0 (WRO), bits 1 and 0 (Figure 9-1).

The programming of the Shift Left/Shift Right modes for the DMA cell is accomplished in the BCR, bit 0. In this case, the shift function is similar to that for the SCC cell; with Shift left, the internal register addresses are decoded from bits AD5 through AD1 and with Shift Right, the internal register addresses are decoded from bits AD4 through AD0.

When the multiplexed bus mode is selected, Write Register 0 (WRO) takes on the form of WRO in the Z8030 (Figure 9).

All data transfers to and from the ISCC are done in bytes even though the data can, at special times, occupy the lower or upper byte of the 16-bit bus. When accessed as a peripheral device (i.e., when the ISCC is not a bus master performing DMA transfers), all bus transactions are on the lower 8 bits of the bus with the following exception:

---

## FUNCTIONAL DESCRIPTION (Continued)

When the ISCC registers are read, the byte data is present on both the lower 8 bits of the bus and the upper 8 bits of the bus. Data is accepted only on the lower 8 bits of the bus except in certain DMA transfers.

During DMA transfers, data may be transferred to or from the ISCC on the upper 8 bits of the bus for odd or even byte transfers. During DMA transfers to memory from the ISCC, byte data only is transferred and the data appears on both the lower 8 bits and is replicated on the upper 8 bits of the bus.

During DMA transfers to the ISCC from memory, byte data only is transferred and normally data is accepted only on the lower 8 bits of the bus. However, the byte swapping

feature may be used to elect on which byte of the bus the data is accepted. The byte swapping feature is enabled by programming the Byte Swap Enable bit to a 1 in the BCR. The odd/even byte transfer selection is made by programming the Byte Swap Select bit in the BCR. If Byte Swap Select is a 1, then even address bytes (transfers where the DMA address has A0 equal 0) are transferred on the lower 8 bits of the bus and odd address bytes (transfers where the DMA address has A0 equal 1) are transferred on the upper 8 bits of the bus. If Byte Swap Select is a 0, then even address bytes (transfers where the DMA address has A0 equal 0) are transferred on the upper 8 bits of the bus and odd address bytes (transfers where the DMA address has A0 equal 1) are transferred on the lower 8 bits of the bus.

---

## I/O INTERFACE CAPABILITIES

The ISCC offers the choice of Polling, Interrupt (vectored or non-vectored), and DMA Transfer modes to transfer data, status, and control information to and from the CPU.

**Polling.** In this mode all interrupts and the DMA's are disabled. Three status registers in the SCC are automatically updated whenever any function is performed. For example, end-of-frame in SDLC mode sets a bit in one of these status registers. With polling, the CPU must periodically read a status register until the register contents indicate the need for some CPU action to be taken. Only one register in the SCC needs to be read; depending on the contents of the register, the CPU either reads data, writes data, or satisfies an error condition. Two bits in the register indicate the need for data transfer. An alternative is to poll the Interrupt Pending register to determine the source of an interrupt. The status for both SCC channels resides in one register.

**Interrupts.** When the ISCC responds to an Interrupt Acknowledge signal (INTACK) from the CPU, an interrupt vector is placed on the data bus. Both the SCC and the DMA contain vector registers. Depending on the source of interrupt, one of these vectors is returned, either unmodified or modified by the interrupt status to indicate the exact cause of the interrupt.

Each of the six sources in interrupts in the SCC (Transmit, Receive, and External/Status interrupts in both channels) and each DMA channel has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt Under Service (IUS), and Interrupt Enable (IE). If the IE bit is set for any given source of interrupt, then that source can request interrupts. The only exception to this rule is when

the associate Master Interrupt Enable (MIE) bit is reset, then no interrupts are requested. Both the SCC and the DMA have an associated MIE bit. The IE bits in the SCC are write only, but the IE bits in the DMA are read write.

The ISCC provides for nesting of interrupt sources with an interrupt daisy chain using the IEI, IEO, and INTACK pins. As a microprocessor peripheral, the ISCC may request an interrupt only when no higher priority device is requesting one, e.g., when IEI is High. If the device in question requests an interrupt, it enables the /INT signal. The CPU then responds with /INTACK, and the interrupting device places the vector on the data bus.

In the ISCC, the IP bit signals a need for interrupt servicing. When an IP bit is 1 and the IEI input is High, the /INT signal is activated, requesting an interrupt. In the SCC, if the IE bit is not set, then the IP for that source can never be set. The IP bits in the DMA are set independent of the IE bit.

The IUS bits signal that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority in the ISCC and external to the ISCC are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the ISCC being pulled Low and propagated to subsequent peripherals. Internally, the SCC is higher priority than the DMA. An IUS bit is set during an Interrupt Acknowledge cycle if there are no higher priority devices requesting interrupts.

Within the SCC portion of the ISCC there are three types of interrupts: Transmit, Receive, and External/Status. Each

---

interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receive, Transmit, and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted when the transmit buffer becomes empty. This implies that the transmitter had a data character written into it to make it empty. When enabled, the receiver interrupts the CPU in one of three ways:

1. Interrupt on First Receive Character or Special Receive Condition
2. Interrupt on All Receive Characters or Special Receive Condition
3. Interrupt on Special Condition Only

Interrupt on First Character or Special Condition, and Interrupt on Special Condition Only, are typically used when doing block transfers with the DMA. A Special Receive Condition is one of the following: receiver overrun, framing error in Asynchronous mode, end-of-frame in SDLC mode and, optionally, a parity error. The Special Receive Condition interrupt is different from an Ordinary Receive Character Available interrupt only by the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt occurs from Special Receive Conditions any time after the First Receive Character interrupt.

The main function of the External/Status interrupt is to monitor the signal transitions of the /CTS, /DCD, and /SYNC pins; however, an External/Status interrupt is also caused by a Transmit Underrun condition, or a zero count in the baud rate generator, or by the detection of a Break (Asynchronous mode), Abort (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the Abort or EOP has a special feature allowing the ISCC to interrupt when the Abort or EOP sequence is detected or terminated. This feature facilitates the proper termination of the current message, correct initialization of the next message, and the accurate timing of the Abort condition in external logic.

Each DMA in the ISCC has two sources of interrupt, which share an IP bit and an IUS bit, but have independent enables: Terminal Count and Abort. The Abort interrupt is

generated when an active DMA channel is forced to terminate its transfers because /BUSACK is de-asserted during a transfer. The Terminal Count interrupt is generated when the DMA transfer count reaches zero. The DMA channels themselves are prioritized in a fixed order: Receive A, Transmit A, Receive B, and Transmit B.

**DMA Transfer.** In this mode, the on-chip DMA channels transfer data directly to the transmit buffers or directly from the receive buffers. No other transfers are possible (for initialization, for example). The request signals from the receivers and transmitters are hard-wired to the request inputs of the DMA channels internally. Each DMA channel provides a 32-bit address which is either incremented or decremented with a 16-bit transfer length. Whenever a DMA channel receives a request from its associated receiver or transmitter and the DMA channel is enabled, the ISCC activates the /BUSREQ signal. Upon receipt of an active /BUSACK, the DMA channel transfers data between memory and the SCC. This transfer continues until the receiver or transmitter stops requesting a transfer, until the terminal count is reached, or /BUSACK is deactivated. The four DMA channels operate independently when the Request Per Channel option is selected; otherwise, all requests pending at the time of bus acquisition will be serviced before the bus is released. Each DMA channel is independently enabled and disabled.

**Bus Interface.** The ISCC contains a flexible bus interface that provides the resources necessary to interface the ISCC to virtually any type of bus. The ISCC directly supports either an 8-bit or a 16-bit bus, although all transfers to and from the device are limited to 8-bits at a time. The control signals provided allow connection to either a multiplexed address/data type bus or to a separate address and data type bus. While the ISCC is bus master, the upper address, lower address, and data are multiplexed on AD15-0. Interrupt Acknowledge is signaled through the /INTACK signal, which may be programmed as either a status input, a pulsed input, or a double-pulsed input. The ISCC also contains a /WAIT/RDY input for synchronizing CPU or DMA and memory accesses. This pin may be programmed to act as either a /WAIT signal or a /READY signal. The appropriate signal is provided by the ISCC when it is not bus master, and is sampled by the ISCC when it is bus master. The ISCC requests the bus via a /BUSREQ signal and assumes bus mastership upon receipt of a /BUSACK signal.

## CONTROL REGISTERS

The ISCC contains separate register sets for the SCC core and the DMA core. Access to each set is controlled by the A0/SCC//DMA pin. When this pin is an input, a High selects the SCC core and a Low selects the DMA core. The first write to the ISCC after reset is always to the Bus Configuration Register (BCR), see Figure 8. If an /AS is present before the BCR is written to, a multiplexed bus is selected. If no /AS is present before the BCR write, a non-multiplexed bus is selected. The BCR cannot be changed without resetting the ISCC.

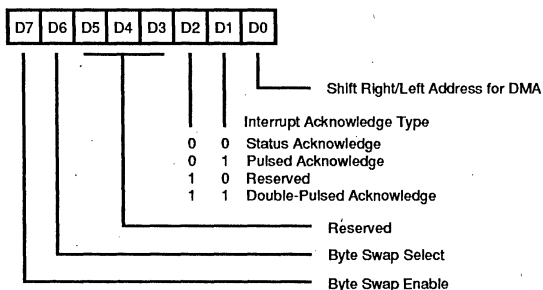


Figure 8. Bus Configuration Register (BCR)

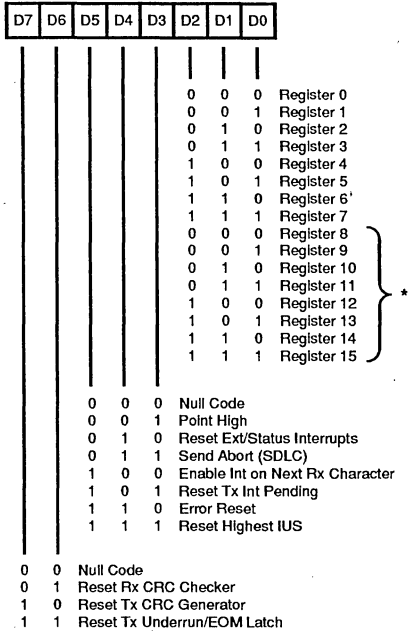
**SCC Cell.** The SCC core contains 13 write registers (14 counting the transmit buffer) and ten read registers (11 counting the receive buffer) in each channel. Two of the write registers are shared (WR2 and WR9) and are accessed by both channels. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits. Table 1 is a list of the SCC write registers and Table 2 is a list of the SCC read registers. Figures 9 and 10 show the write and read register formats. Read Registers 6 and 7 are only accessible when the SDLC FIFO is enabled. When the SDLC FIFO is not enabled, Read Registers 6 and 7 are images of Read Registers 2 and 3, respectively.

**DMA Cell.** The DMA cell contains 17 registers (counting the BCR). All of the registers are write/read except the BCR, CCAR and ICSR. The ISCC also has two status registers, the DMA status register (DSR) and the Interrupt Status Register (ISR), which are addressed by reading the CCAR and ICSR. The DMA also reserves two addresses for future use and should not be addressed or should be written with all zeros to prevent unexpected operation and maintain compatibility with future products. Each DMA channel has a 32-bit wide address register providing an addressing range of 4 gigabytes. Each channel also has a 16-bit count register for up to 64K byte data packet sizes (Reference Figures 11-26 and Table 3).

Table 1. SCC Write Registers

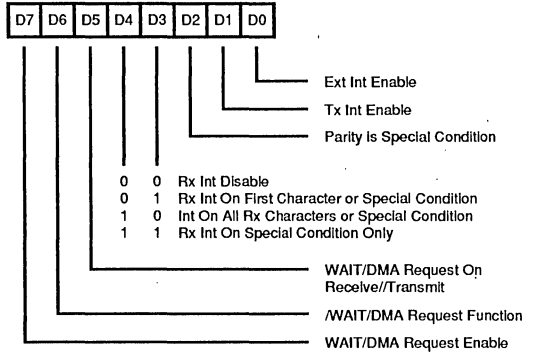
Bit	Description
WR0	Register Pointers, various initialization commands
WR1	Transmit and Receive interrupt enables, WAIT/DMA commands
WR2	Interrupt Vector
WR3	Receive parameters and control modes
WR4	Transmit and Receive modes and parameters
WR5	Transmit parameters and control modes
WR6	Sync Character or SDLC address
WR7	Sync Character or SDLC flag
WR8	Transmit buffer
WR9	Master Interrupt control and reset commands
WR10	Miscellaneous transmit and receive control bits
WR11	Clock mode controls for receive and transmit
WR12	Lower byte of baud rate generator
WR13	Upper byte of baud rate generator
WR14	Miscellaneous control bits
WR15	External status interrupt enable control

Write Register 0 (non-multiplexed bus mode)

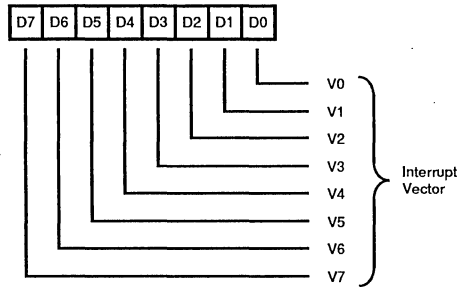


\* With Point High Command

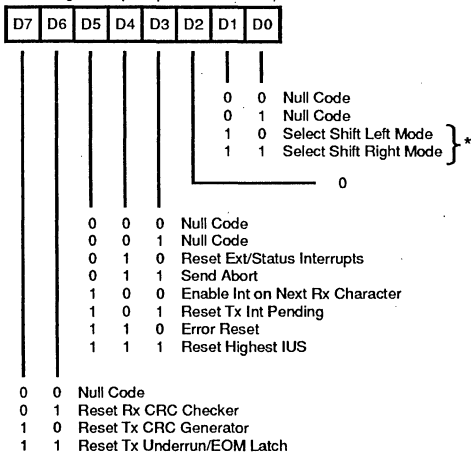
Write Register 1



Write Register 2



Write Register 0 (multiplexed bus mode)



\* B Channel Only

Write Register 3

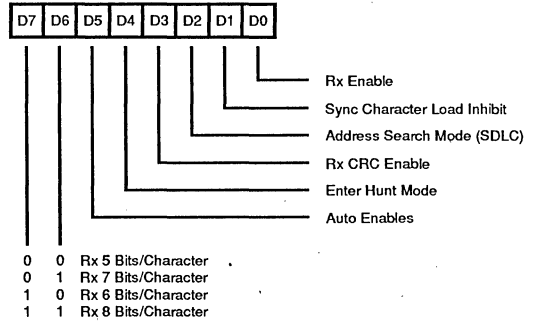
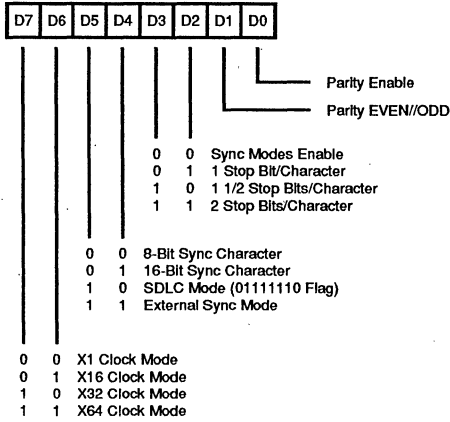
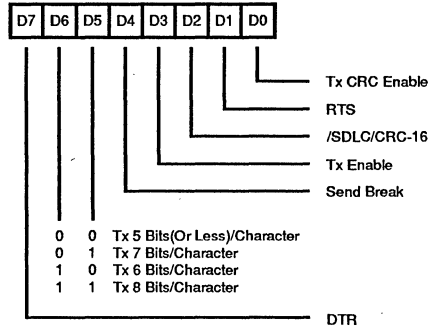


Figure 9. Write Register Bit Functions

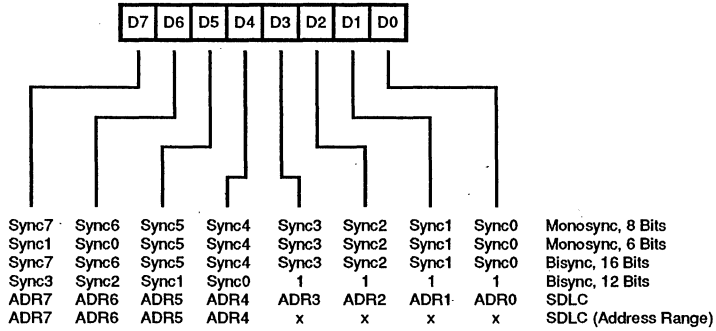
Write Register 4



Write Register 5



Write Register 6



Write Register 7

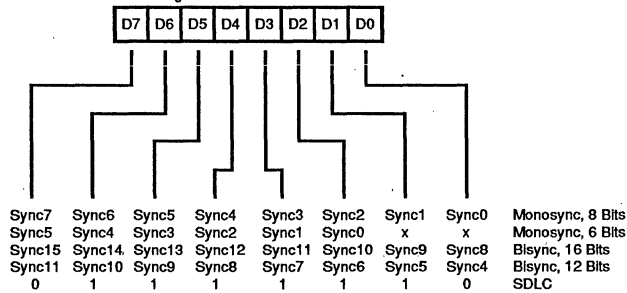
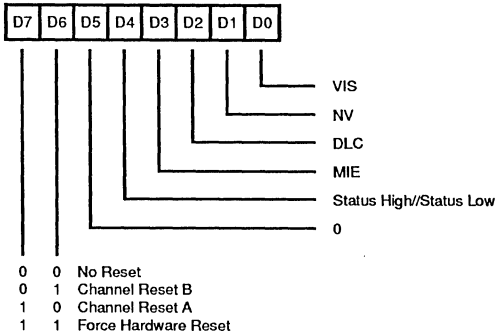
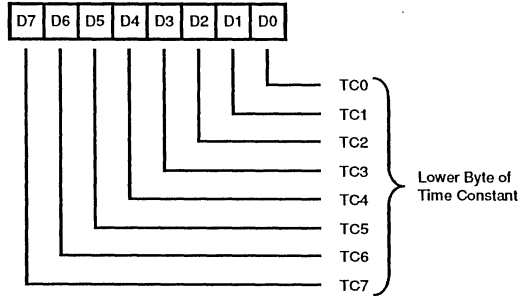


Figure 9. Write Register Bit Functions (Continued)

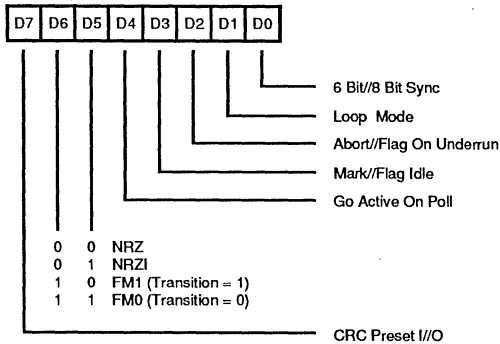
Write Register 9



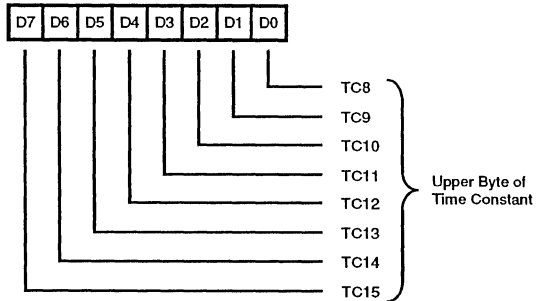
Write Register 12



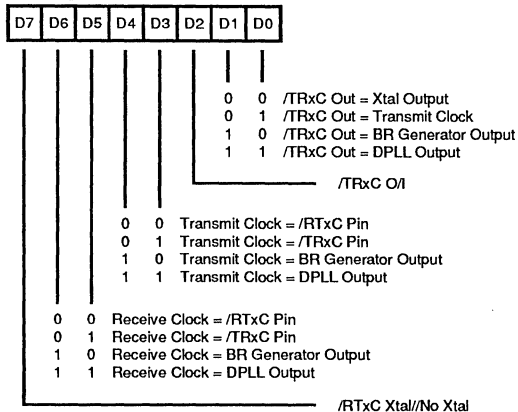
Write Register 10



Write Register 13



Write Register 11



Write Register 14

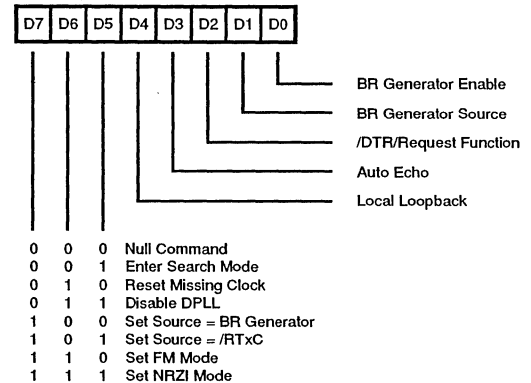
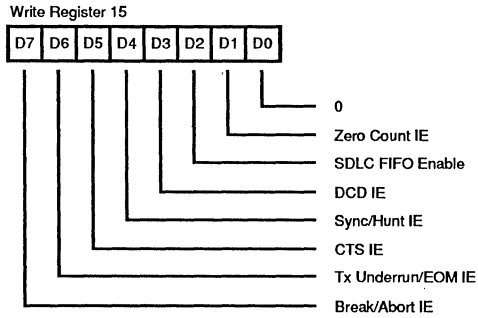


Figure 9. Write Register Bit Functions (Continued)



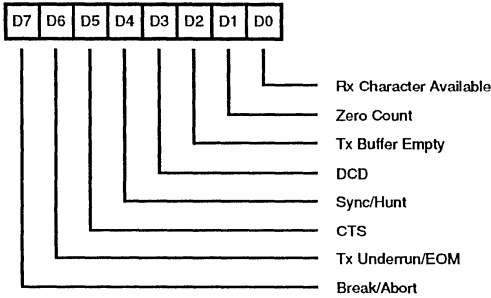


**Figure 9. Write Register Bit Functions** (Continued)

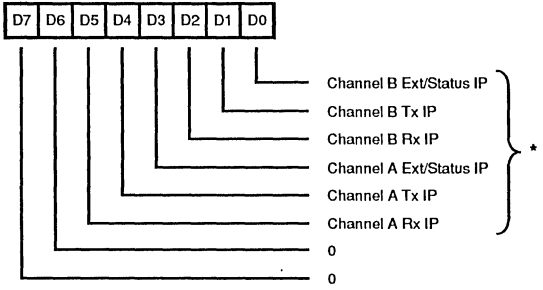
**Table 2. SCC Read Registers**

Bit	Description
RR0	Transmit and Receive buffer status and external status
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only), Unmodified interrupt vector (Channel A only)
RR3	Interrupt pending bits (Channel A only)
RR6	SDLC FIFO byte counter lower byte (only when enabled)
RR7	SDLC FIFO byte count and status (only when enabled)
RR8	Receive buffer
RR10	Miscellaneous status bits
RR12	Lower byte of baud rate generator time constant
RR13	Upper byte of baud rate generator time constant
RR15	External Status interrupt information

Read Register 0

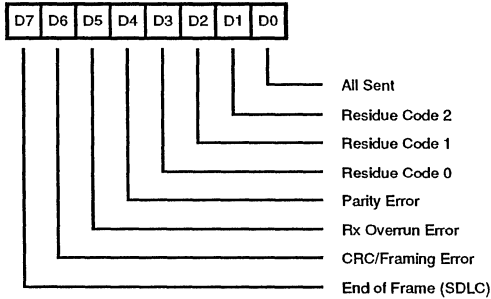


Read Register 3

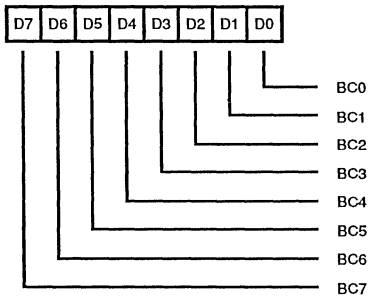


\* Always 0 In B Channel

Read Register 1

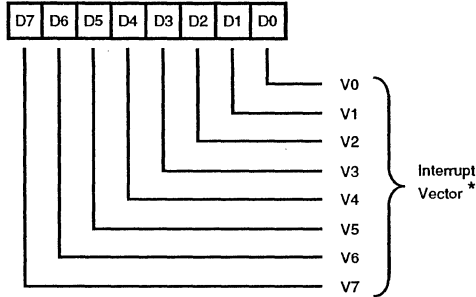


Read Register 6 \*



\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

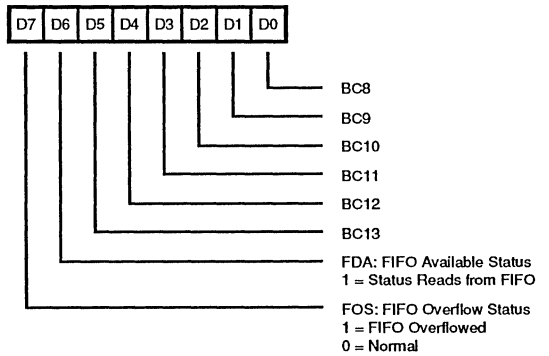
Read Register 2



\* Modified In B Channel

SDLC FIFO Status and Byte Count (LSB)

Read Register 7 \*



\* Can only be accessed if the SDLC FIFO enhancement is enabled (WR15 bit D2 set to 1)

SDLC FIFO Status and Byte Count (MSB)

Figure 10. Read Register Bit Functions

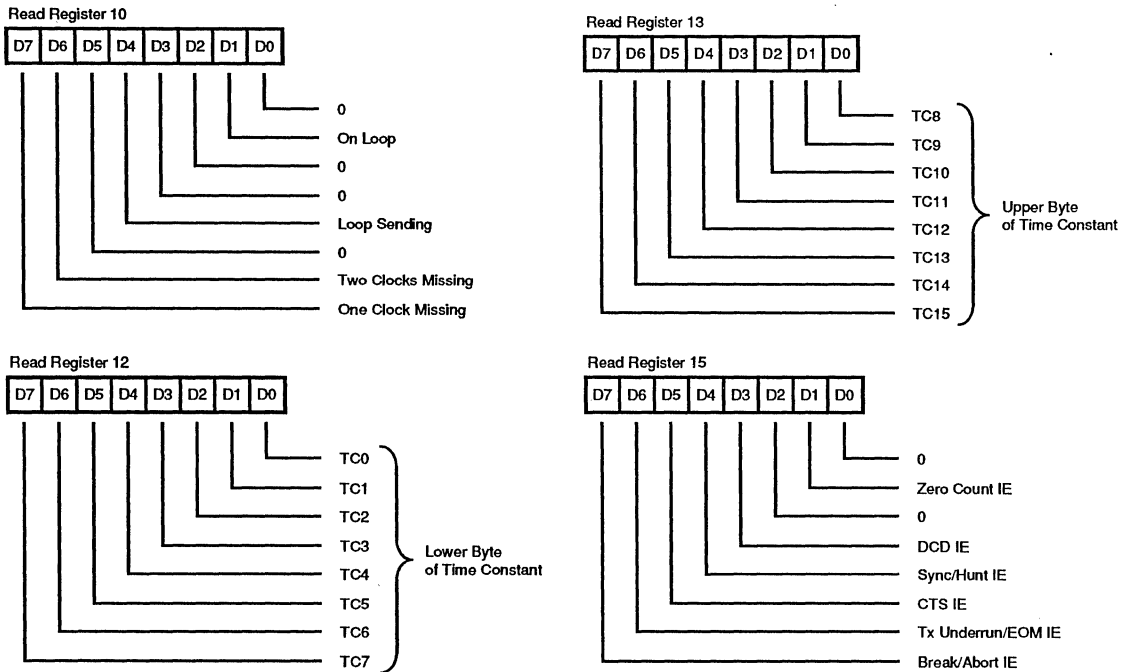


Figure 10. Read Register Bit Functions (Continued)

Table 3. DMA Cell Register Description

Address	Name	Description
xxxxx	BCR	Bus Configuration Register
00000	CCAR	Channel Command/Address Register (WRITE)
00000	DSR	DMA Status Register (READ)
00001	ICR	Interrupt Control Register
00010	IVR	Interrupt Vector Register
00011	ICSR	Interrupt Command Register (WRITE)
00011	ISR	Interrupt Status Register (READ)
00100	DER	DMA Enable/Disable Register
00101	DCR	DMA Control Register
00110		Reserved Address
00111		Reserved Address
01000-01001	RDCRA	Receive DMA Count Register Channel A (Low-high byte)
01010-01011	TDCRA	Transmit DMA Count Register Channel A
01100-01101	RDCRB	Receive DMA Count Register Channel B
01110-01111	TDCRB	Transmit DMA Count Register Channel B
10000-10011	RDARA	Receive DMA Address Register Channel A
10100-10111	TDARA	Transmit DMA Address Register Channel A
11000-11011	RDARB	Receive DMA Address Register Channel B
11100-11111	TDARB	Transmit DMA Address Register Channel B

Address: 00000 (Write)

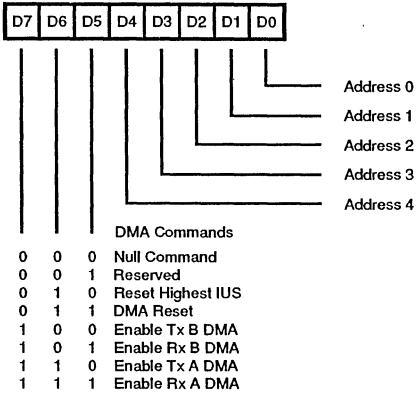


Figure 11. Channel Command/Address Register

Address: 00000 (Read)

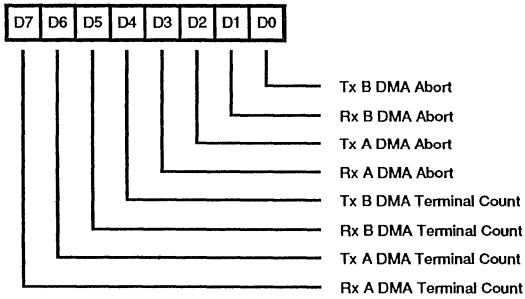


Figure 12. DMA Status Register

Address: 00001

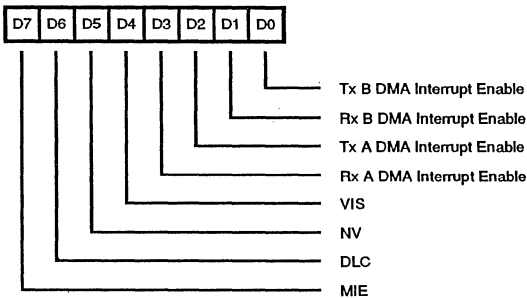


Figure 13. Interrupt Control Register

Address: 00010

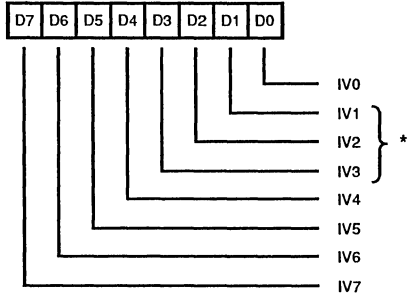


Figure 14. Interrupt Vector Register

Address: 00011 (Write)

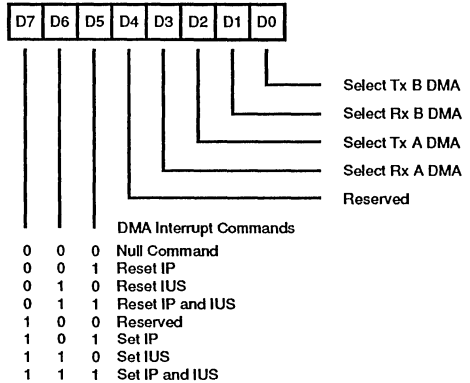


Figure 15. Interrupt Command/Register

Address: 00011 (Read)

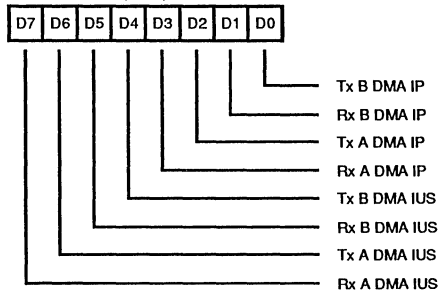


Figure 16. Interrupt Status Register

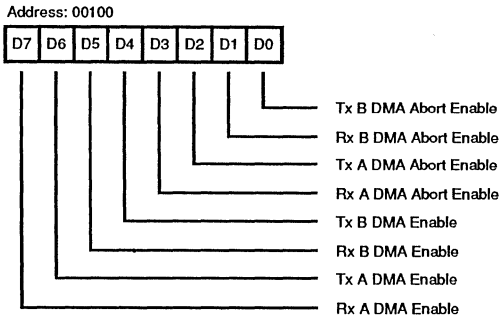


Figure 17. DMA Enable Register

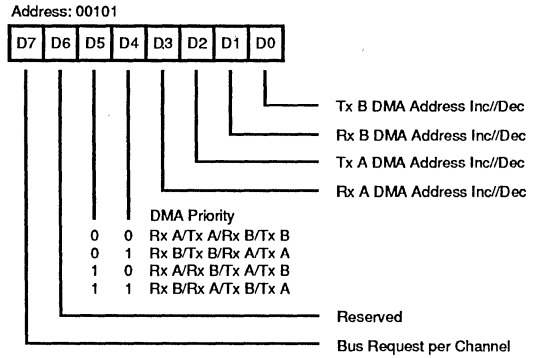
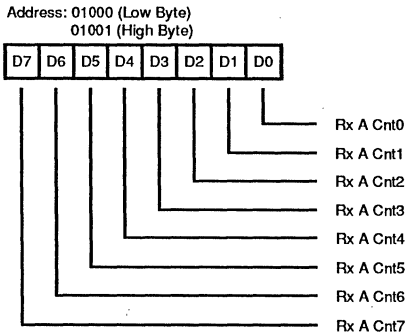
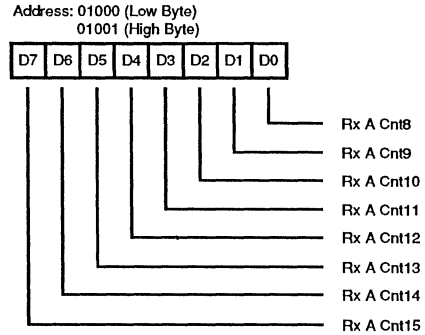


Figure 18. DMA Control Register

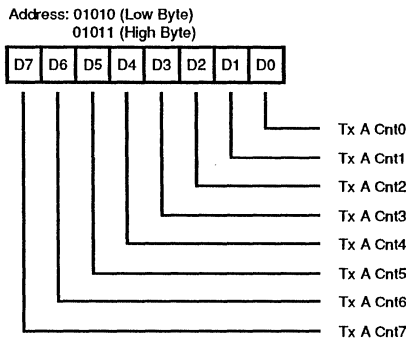


A) LSB

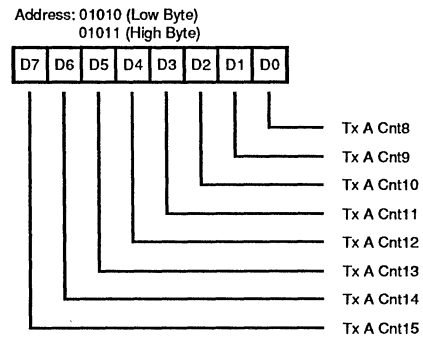


B) MSB

Figure 19. Receive DMA Count Register Channel A



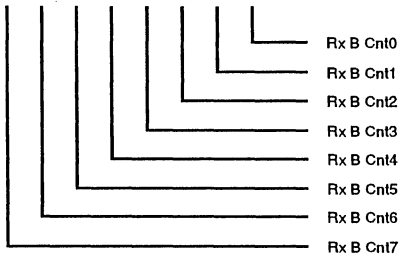
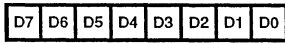
A) LSB



B) MSB

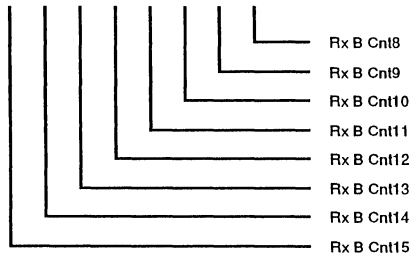
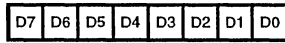
Figure 20. Transmit DMA Count Register Channel A

Address: 01100 (Low Byte)  
01101 (High Byte)



A) LSB

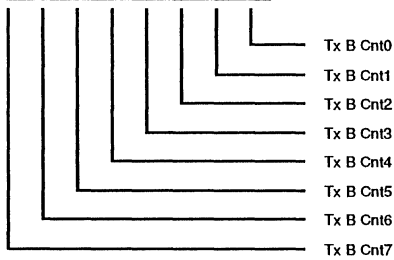
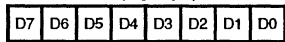
Address: 01100 (Low Byte)  
01101 (High Byte)



B) MSB

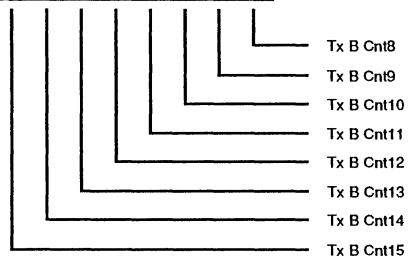
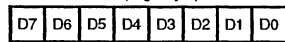
Figure 21. Receive DMA Count Register Channel B

Address: 01110 (Low Byte)  
01111 (High Byte)



A) LSB

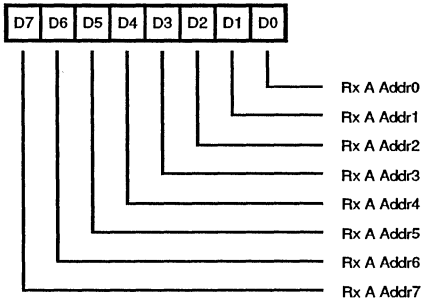
Address: 01110 (Low Byte)  
01111 (High Byte)



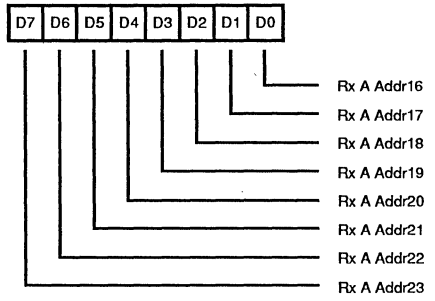
B) MSB

Figure 22. Transmit DMA Count Register Channel B

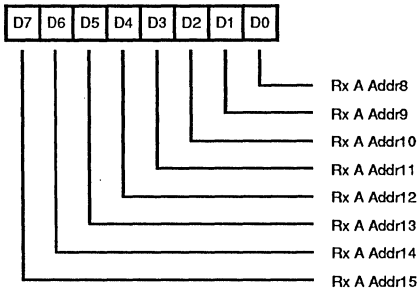
Address: 10000 (Bits 0-7)



Address: 10010 (Bits 16-23)



Address: 10001 (Bits 8-15)



Address: 10011 (Bits 24-31)

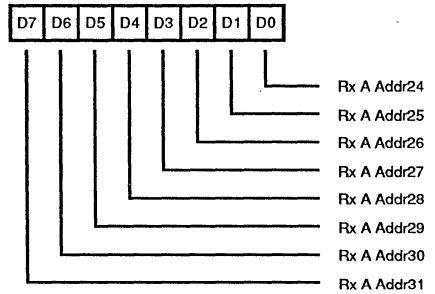
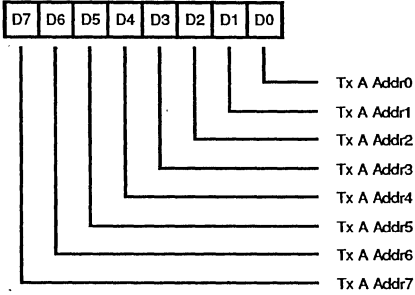
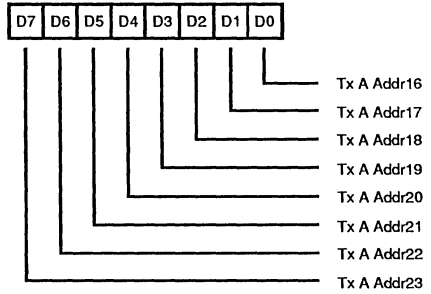


Figure 23. Receive DMA Address Register Channel A

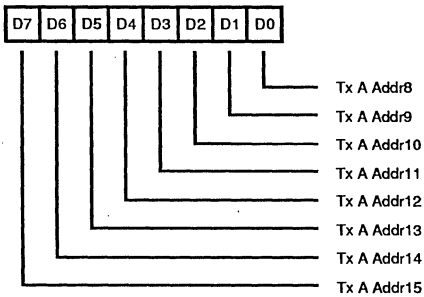
Address: 10100 (Bits 0-7)



Address: 10110 (Bits 16-23)



Address: 10101 (Bits 8-15)



Address: 10111 (Bits 24-31)

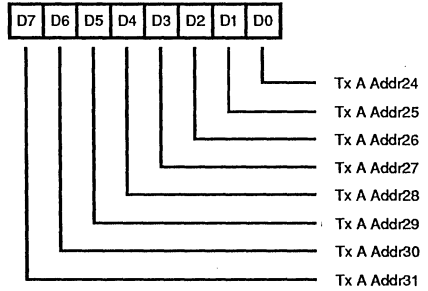
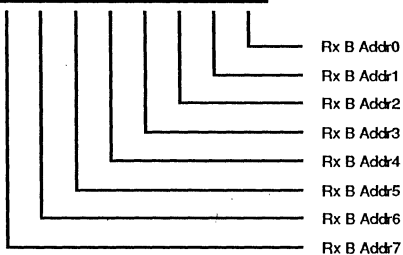
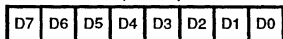


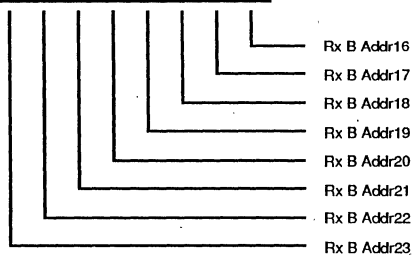
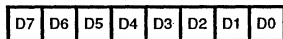
Figure 24. Transmit DMA Address Register Channel A



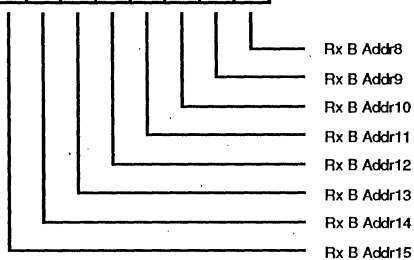
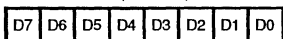
Address: 11000 (Bits 0-7)



Address: 11010 (Bits 16-23)



Address: 11001 (Bits 8-15)



Address: 11011 (Bits 24-31)

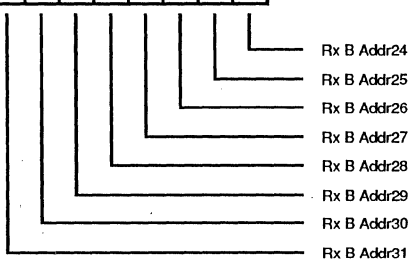
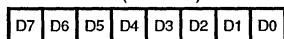
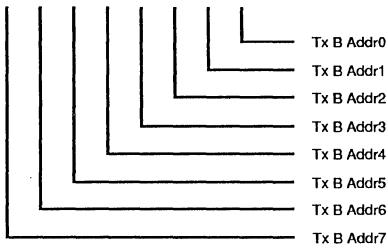
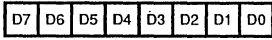
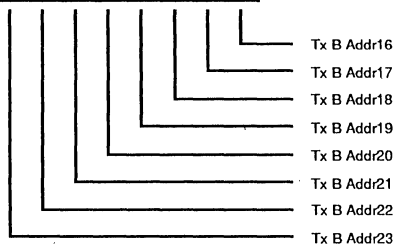
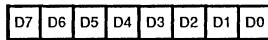


Figure 25. Receive DMA Address Register Channel B

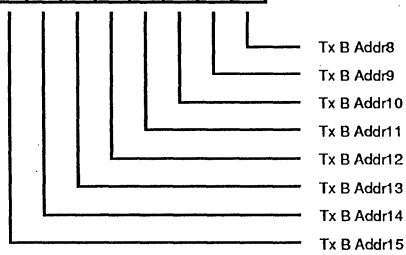
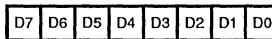
Address: 11100 (Bits 0-7)



Address: 11110 (Bits 16-23)



Address: 11101 (Bits 8-15)



Address: 11111 (Bits 24-31)

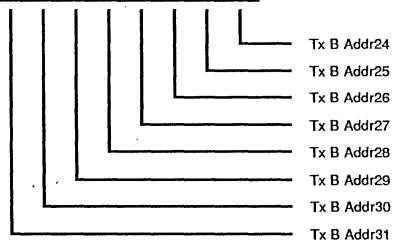
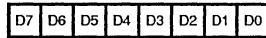


Figure 26. Transmit DMA Address Register Channel B

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins,  
with respect to GND .....-0.3 V to +7.0 V  
Operating Ambient  
Temperature .....See Ordering Information  
Storage Temperature ..... -85°C to 150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

- $+4.75\text{ V} \leq V_{CC} \leq 5.25\text{ V}$
- $\text{GND} = 0\text{ V}$
- $T_A$  as specified in Ordering Information

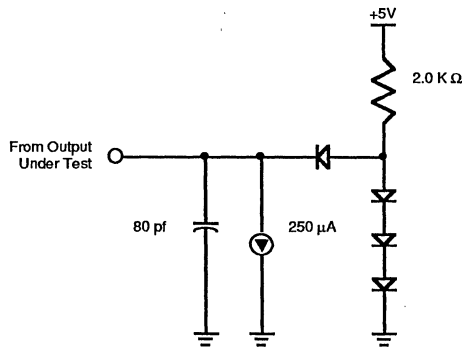


Figure 27. Standard Test Load

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit	Condition
$C_{IN}$	Input Capacitance		10	pF	Unmeasured Pins Returned to Ground
$C_{OUT}$	Output Capacitance		15	pF	
$C_{IO}$	Bidirectional Capacitance		20	pF	

**Note:**

$f = 1\text{ MHz}$  over specified temperature range.  
Unmeasured pins returned to ground.

## MISCELLANEOUS

Transistor Count      52,047

## DC CHARACTERISTICS

### Z16C35

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC} + 0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6$ mA
$V_{OH2}$	Output High Voltage	$V_{CC} - 0.8$			V	$I_{OH} = -250$ $\mu$ A
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = +2.0$ mA
$I_{IL}$	Input Leakage			$\pm 10.00$ $\mu$ A		$0.4 < V_{IN} < +2.4$ V
$I_{OL}$	Output Leakage			$\pm 10.00$ $\mu$ A		$0.4 < V_{OUT} < +2.4$ V
$I_{CC1}$	$V_{CC}$ Supply Current		7	50	mA	$V_{CC} = 5$ V, $V_{IH} = 4.8$ V, $V_{IL} = 0.2$ V

#### Note:

$V_{CC} = 5$  V  $\pm$  5% unless otherwise specified, over specified temperature range.

## AC CHARACTERISTICS

Note: See the corresponding figures following this table (Figures 28-49).

No	Symbol	Parameter	10 MHz		16 MHz		Notes
			Min	Max	Min	Max	
1	Tcyc	Bus Cycle Time	4TcPC		4TcPC		
2	TwASl	/AS Low Width	40		20		
3	TwASh	/AS High Width	90		55		
4	TwDSl	/DS Low Width	70		50		
5	TwDSH	/DS High Width	60		30		
6	TdAS(DS)	/AS Rise to /DS Fall Delay Time	5		5		
7	TdDS(AS)	/DS Rise to /AS Fall Delay Time	5		5		
8	TdDS(DRa)	/DS Fall to Data Active Delay	0		0		
9	TdDS(DRv)	/DS Fall to Data Valid Delay			85	75	
10	TdDS(DRn)	/DS Rise to Data Not Valid Delay	0		0		
11	TdDS(DRz)	/DS Rise to Data Float Delay			20	15	
12	TsCS(AS)	/CS to /AS Rise Setup Time	15		12		
13	ThCS(AS)	/CS to /AS Rise Hold Time	0		0		
14	TsADD(AS)	Direct Address to /AS Rise Setup Time	15		12		[1]
15	ThADD(AS)	Direct Address to /AS Rise Hold Time	5		5		[1]
16	TsSIA(AS)	Status /INTACK to /AS Rise Setup Time	15		10		
17	ThSIA(AS)	Status /INTACK to /AS Rise Hold Time	5		5		
18	TsAD(AS)	Address to /AS Rise Setup Time	15		10		
19	ThAD(AS)	Address to /AS Rise Hold Time	5		5		
20	TsRW(DS)	R/W to /DS Fall Setup Time	0		0		
21	ThRW(DS)	R/W to /DS Fall Hold Time	25		15		
22	TdDSf(RDY)	/DS Fall to /READY Fall Delay			50	40	
23	TdDSr(RDY)	/DS Rise to /READY Rise Delay			40	20	
24	TsDW(DS)	Write Data to /DS Fall Setup Time	0		0		
25	ThDW(DS)	Write Data to /DS Fall Hold Time	25		15		
26	TdRDY(DRv)	/READY Fall to Data Valid Delay			40	40	
28	TwRDl	/RD Low Width	70		50		
29	TwRDh	/RD High Width	60		30		

## AC CHARACTERISTICS (Continued)

No	Symbol	Parameter	10 MHz		16 MHz		Notes
			Min	Max	Min	Max	
30	TdAS(RD)	/AS Rise to /RD Fall Delay Time	5		5		
31	TdRD(AS)	/RD Rise to /AS Fall Delay Time	5		5		
32	TdRD(DRa)	/RD Fall to Data Active Delay	0		0		
33	TdRD(DRv)	/RD Fall to Data Valid Delay		85		75	
34	TdRD(DRn)	/RD Rise to Data Not Valid Delay	0		0		
35	TdRD(DRz)	/RD Rise to Data Float Delay		20		15	
36	TdRDf(RDY)	/RD Fall to /READY Fall Delay		50		40	
37	TdRDr(RDY)	/RD Rise to /READY Rise Delay		40		20	
38	TwWRI	/WR Low Width	70		50		
39	TwWRh	/WR High Width	60		30		
40	TdAS(WR)	/AS Rise to /WR Fall Delay Time	5		5		
41	TdWR(AS)	/WR Rise to AS Fall Delay Time	5		5		
42	TsDW(WR)	Write Data to /WR Fall Setup Time	0		0		
43	ThDW(WR)	Write Data to /WR Fall Hold Time	25		15		
44	TdWRf(RDY)	/WR Fall to /READY Fall Delay		50		40	
45	TdWRr(RDY)	/WR Rise to /READY Fall Delay		40		20	
46	TsCS(DS)	/CS to /DS Fall Setup Time	0		0		[2]
47	ThCS(DS)	/CS to /DS Fall Hold Time	25		15		[2]
48	TsADD(DS)	Direct Address to /DS Fall Setup Time	0		0		[1,2]
49	ThADD(DS)	Direct Address to /DS Fall Hold Time	25		15		[1,2]
50	TsSIA(DS)	Status /INTACK to /DS Fall Setup Time	0		0		[2]
51	ThSIA(DS)	Status /INTACK to /DS Fall Hold Time	25		15		[2]
52	TsCS(RD)	/CS to /RD Fall Setup Time	0		0		[2]
53	ThCS(RD)	/CS to /RD Fall Hold Time	25		15		[2]
54	TsADD(RD)	Direct Address to /RD Fall Setup Time	0		0		[1,2]
55	ThADD(RD)	Direct Address to /RD Fall Hold Time	25		15		[1,2]
56	TsSIA(RD)	Status /INTACK to /RD Fall Setup Time	0		0		[2]
57	ThSIA(RD)	Status /INTACK to /RD Fall Hold Time	25		15		[2]
58	TsCS(WR)	/CS to /WR Fall Setup Time	0		0		[2]
59	ThCS(WR)	/CS to /WR Fall Hold Time	25		15		[2]
60	TsADD(WR)	Direct Address to /WR Fall Setup Time	0		0		[1,2]
61	ThADD(WR)	Direct Address to /WR Fall Hold Time	25		15		[1,2]
62	TsSIA(WR)	Status /INTACK to /WR Fall Setup Time	0		0		[2]
63	ThSIA(WR)	Status /INTACK to /WR Fall Hold Time	25		15		[2]
78	TdDSI(RDY)	/DS Fall (INTACK) to /READY Fall Delay		300		250	[4]
81	TsIEI(DSI)	IEI to /DS Fall (INTACK) Setup Time	60		40		
82	ThIEI(DSI)	IEI to /DS Rise (INTACK) Hold Time	0		0		
83	TdIEI(IEO)	IEI to IEO Delay		60		40	
84	TdAS(IEO)	/AS Rise or Status INTACK to IEO Delay		60		40	
85	TdDSI(INT)	/DS Fall (INTACK) to /INT Inactive Delay		200		170	
86	TdDSI(Wf)	/DS Fall (INTACK) to /WAIT Fall Delay		40		35	
87	TdDSI(Wr)	/DS Fall (INTACK) to /WAIT Rise Delay		300		175	[4]
88	TdW(DRy)	/WAIT Rise to Data Valid Delay		40		35	
89	TdRDI(RDY)	/RD Fall (INTACK) to /READY Fall Delay		300		175	[4]

## AC CHARACTERISTICS (Continued)

No	Symbol	Parameter	10 MHz		16 MHz		Notes
			Min	Max	Min	Max	
91	TsIEI(RDI)	IEI to /RD Fall (INTACK) Setup Time	60		50		
92	ThIEI(RDI)	IEI to /RD Rise (INTACK) Hold Time	0		0		
93	TdRDI(INT)	/RD Fall (INTACK) to /INT Inactive Delay		200		170	
94	TdRDI(Wf)	/RD Fall (INTACK) to /WAIT Fall Delay		40		35	
95	TdRDI(Wr)	/RD Fall (INTACK) to /WAIT Rise Delay		300		175	[4]
96	TwPIA1	Pulsed /INTACK Low Width	70		55		
97	TwPIAh	Pulsed /INTACK High Width	60		45		
98	TdAS(PIA)	/AS Rise to Pulsed /INTACK Fall Delay Time	5		5		
99	TdPIA(AS)	Pulsed /INTACK Rise to /AS Fall Delay Time	5		5		
100	TdPIA(DRa)	Pulsed /INTACK Fall to Data Active Delay	0		0		
101	TdPEA(DRn)	Pulsed /INTACK Rise to Data Not Valid Delay	0		0		
102	TdPIA(DRz)	Pulsed /INTACK Rise to Data Float Delay		20		15	
103	TsIEI(PIA)	IEI to Pulsed /INTACK Fall Setup Time	60		50		
104	ThIEI(PIA)	IEI to Pulsed /INTACK Rise Hold Time	0		0		
105	TdPIA(IEO)	Pulsed /INTACK Fall to IEO Delay		60		50	
106	TdPIA(INT)	Pulsed /INTACK Fall to /INT Inactive Delay		200		170	
107	TdPIA(RDY)	Pulsed /INTACK Fall to /READY Fall Delay		300		200	[4]
108	TdPIAr(RDY)	Pulsed /INTACK Rise to /READY Rise Delay		40		35	
109	TdPIA(Wf)	Pulsed /INTACK Fall to /WAIT Fall Delay		40		35	
110	TdPIA(Wr)	Pulsed /INTACK Fall to /WAIT Rise Delay		300		175	[4]
111	TdSIA(INT)	Status /INTACK Fall to /INT Inactive Delay		200		200	[2]
113	TwRESI	/RESET Low Width	170		140		
114	TwRESH	/RESET High Width	60		40		
115	TdRES(STB)	/RESET Rise to /Strobe Fall	60		40		[3]
116	TdPC(BUSa)	PCLK Rise to Bus Active Delay		40		35	[5]
117	TdPC(BRQ)	PCLK Rise to /BUSREQ Delay		40		35	
118	TsBAK(PC)	/BUSACK to PCLK Rise Setup Time	10		10		
119	ThBAK(PC)	/BUSACK to PCLK Rise Hold Time	30		20		
120	TwPCI	PCLK Low Width	35		26		
121	TwPCh	PCLK High Width	35		26		
122	TcPC	PCLK Cycle Time	100		61		
123	TfPC	PCLK Fall Time		10		5	
124	TrPC	PCLK Rise Time		10		5	
125	TdPCr(UAS)	PCLK Rise to /UAS Delay		30		25	[5]
126	TwUASI	/UAS Low Width	30		25		[5,6]
127	TdPCf(UAS)	PCLK Fall to /UAS Delay		30		25	[5]
128	TdPCr(AS)	PCLK Rise to /AS Delay		30		25	[5]
129	TwASI	/AS Low Width	30		25		[5,6]
130	TdPCf(AS)	PCLK Fall to /AS Delay		30		25	[5]
131	TdAS(DSr)	/AS Rise to /DS Fall (READ) Delay	30		25		[5,7]
132	TdDS(PCr)	PCLK Rise to /DS Delay		30		25	[5]
133	TwDSIr	/DS Low Width (READ)	135		90		[5,8]
134	TdPCf(DS)	PCLK Fall to /DS Delay		30		25	[5]
135	TsDR(DS)	Read Data to /DS Rise Setup Time	30		25		[5]

## AC CHARACTERISTICS (Continued)

No	Symbol	Parameter	10 MHz		16 MHz		Notes
			Min	Max	Min	Max	
136	ThDR(DS)	Read Data to /DS Rise Hold Time	0		0		[5]
137	TdPC(RW)	PCLK Rise to R/W Delay		30		25	[5]
138	TdAS(RD)	/AS Rise to /RD Fall Delay	30		25		[5,7]
139	TdPCr(RD)	PCLK Rise to /RD Delay		30		25	[5]
140	TwRDI	/RD Low Width	135		90		[5,8]
141	TdPCf(RD)	PCLK Fall to /RD Delay		30		25	[5]
142	TsDR(RD)	Read Data to /RD Rise Setup Time	30		25		[5]
143	ThDR(RD)	Read Data to /RD Rise Hold Time	0		0		[5]
144	TdPC(ADD)	PCLK Rise to Direct Address Delay		30		25	[1,5]
145	TdPC(AD)	PCLK Rise to Address Delay		40		40	[5]
146	ThAD(PC)	Address to PCLK Rise Hold Time	0		0		[5]
147	TdPC(ADz)	PCLK Rise to Address Float Delay		50		45	[5]
148	TdPC(ADa)	PCLK Rise to Address Active Delay		40		35	[5]
149	TsAD(UAS)	Address to /UAS Rise Setup Time	20		10		[5]
150	ThAD(UAS)	Address to /UAS Rise Hold Time	20		10		[5]
151	TsAD(AS)	Address to /AS Rise Setup Time	20		10		[5]
152	ThAD(AS)	Address to /AS Rise Hold Time	20		10		[5]
153	TsW(PC)	/WAIT to PCLK Fall Setup Time	10		10		[5]
154	ThW(PC)	/WAIT to PCLK Fall Hold Time	30		20		[5]
155	TsRDY(PC)	/READY to PCLK Fall Setup Time	10		10		[5]
156	ThRDY(PC)	/READY to PCLK Fall Hold Time	30		20		[5]
157	ThDW(PC)	Write Data to PCLK Rise Hold Time	0		0		[5]
158	TdAS(DSw)	/AS Rise to /DS Fall (WRITE) Delay	85		45		[5,9]
159	TsDW(DS)	Write Data to /DS Fall Setup Time	30		25		[5,6]
160	TwDSlw	/DS Low Width (WRITE)	90		70		[5,10]
161	ThDW(DS)	Write Data to /DS Rise Hold Time	30		25		[5, 7]
162	TdAS(WR)	/AS Rise to /WR Fall Delay	85		55		[5,9]
163	TsDW(WR)	Write Data to /WR Fall Setup Time	30		25		[5,6]
164	TwWRI	/WR Low Width	90		55		[5,10]
165	ThDW(WR)	Write Data to /WR Rise Hold Time	30		25		[5,7]
166	TdPC(WR)	PCLK Fall to /WR Delay		30		25	[5]
167	TdPC(BUSz)	PCLK Rise to Bus Float Delay		50		40	[5]

### Notes:

- [1] Direct address is A1/A/B or A0/SCC/DMA.
- [2] The parameter applies only when /AS is not present.
- [3] /Strobe is any of /DS, /RD, /WR or Pulsed /INTACK.
- [4] Clock-cycle dependent, 2T<sub>oPC</sub> + Tw<sub>oPC1</sub> + T<sub>oPC</sub> + 55.
- [5] Parameter applies only while ISCC is bus master.
- [6] Clock-cycle dependent, Tw<sub>oPC</sub>h + T<sub>oPC</sub> - 15.
- [7] Clock-cycle dependent, Tw<sub>oPC1</sub> + Tr<sub>oPC</sub> - 15.
- [8] Clock-cycle dependent, T<sub>oPC</sub> + Tw<sub>oPC</sub>h + Tr<sub>oPC</sub> - 10.
- [9] Clock-cycle dependent, T<sub>oPC</sub> - 15.
- [10] Clock-cycle dependent, T<sub>oPC</sub> - 10.
- [11] Timings in nanoseconds

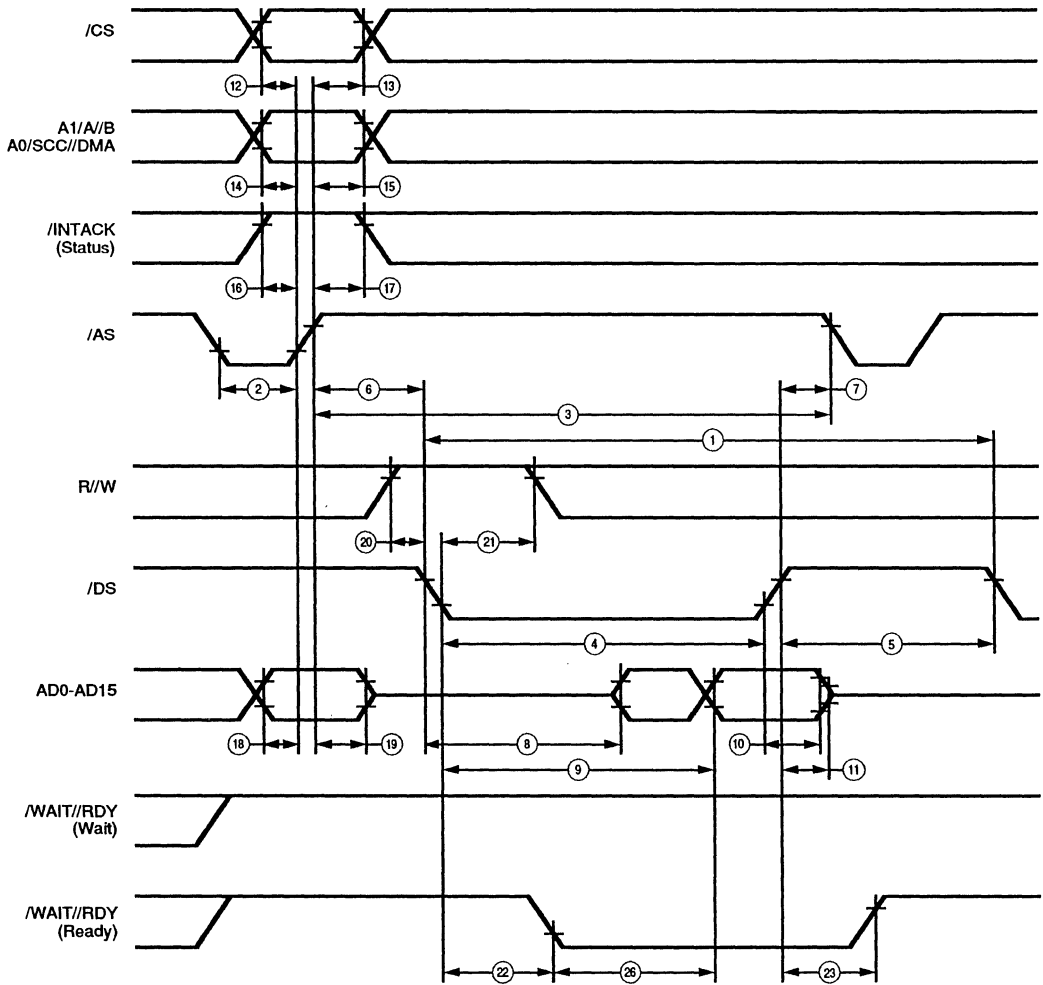


Figure 28. Multiplexed /DS Read Cycle



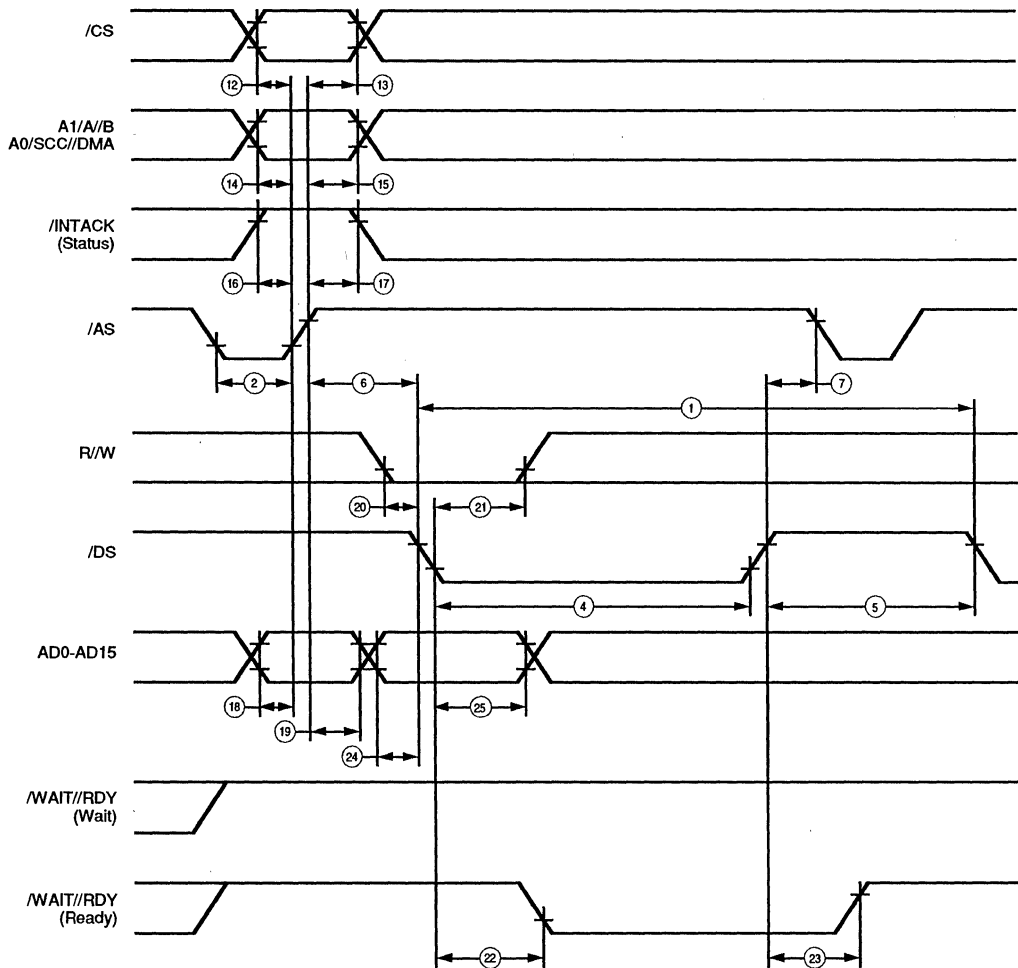


Figure 29. Multiplexed /DS Write Cycle

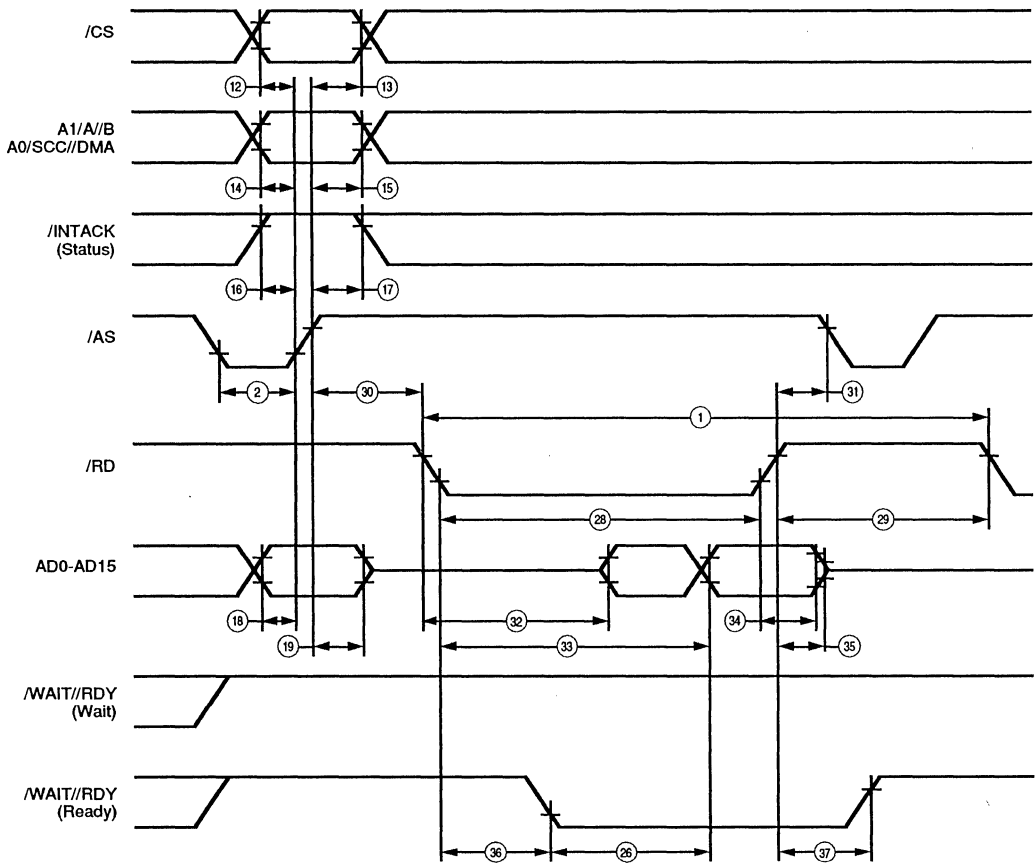


Figure 30. Multiplexed /RD Read Cycle

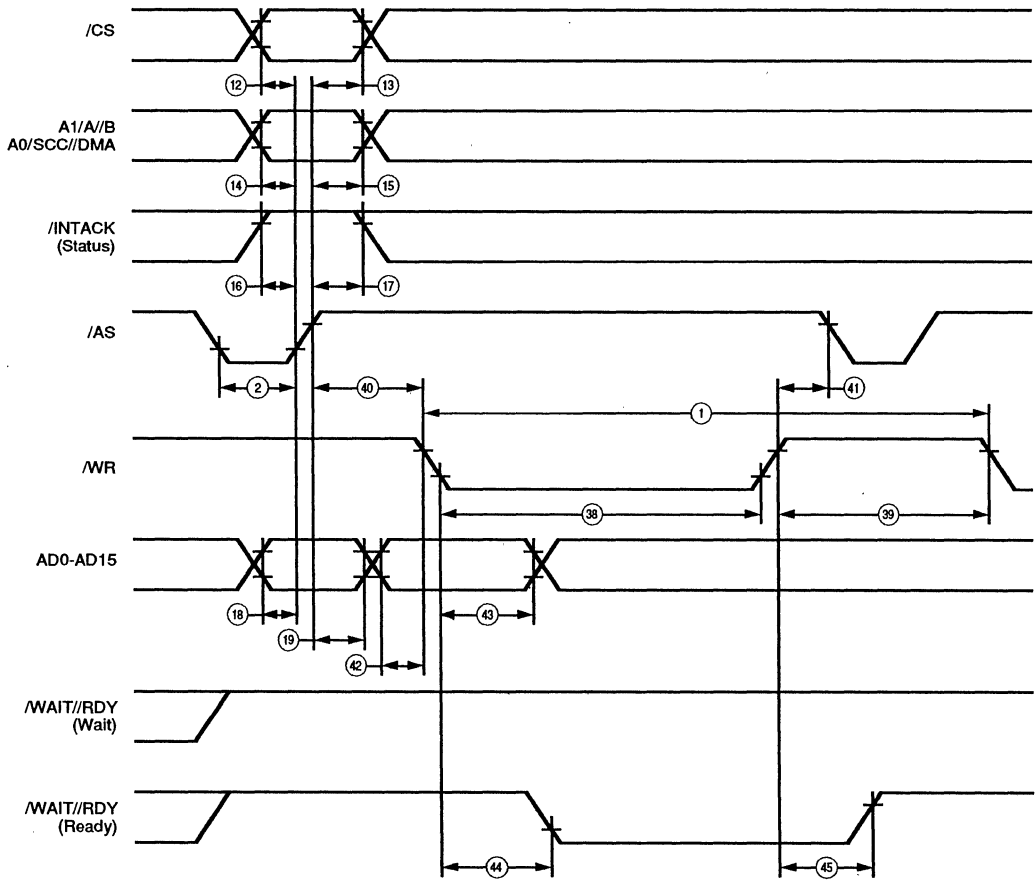


Figure 31. Multiplexed /WR Write Cycle

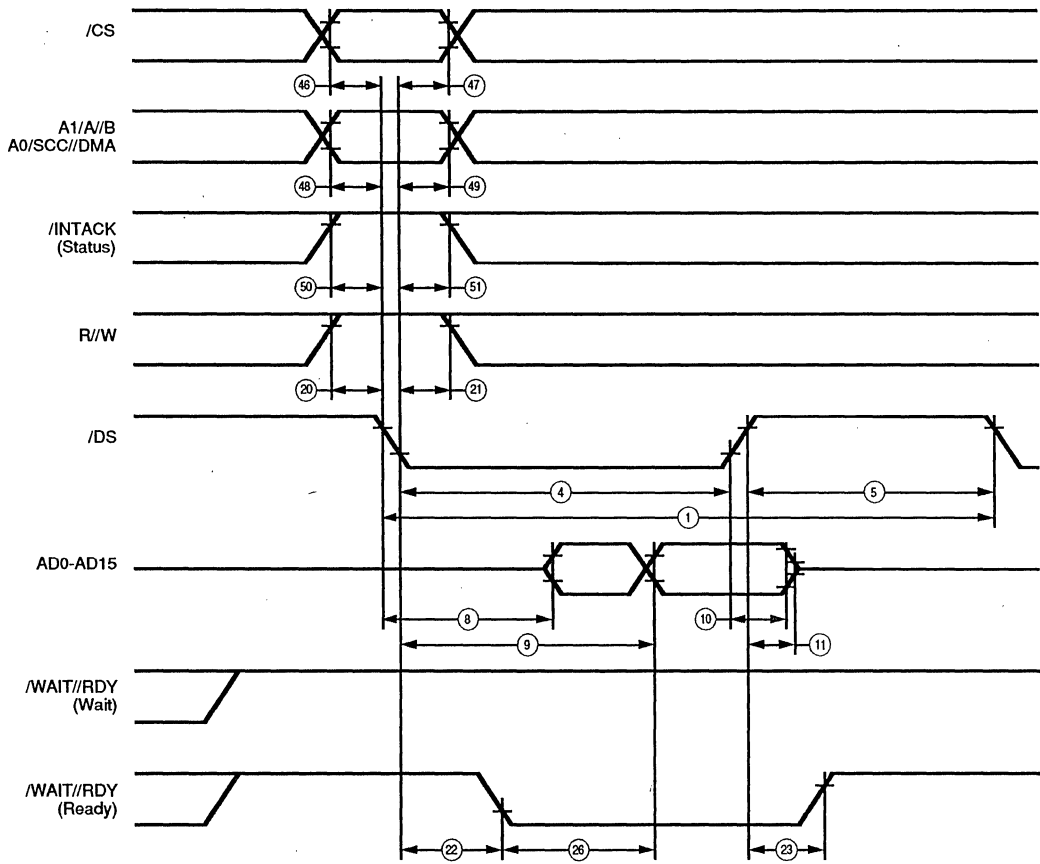


Figure 32. Non-multiplexed /DS Read Cycle

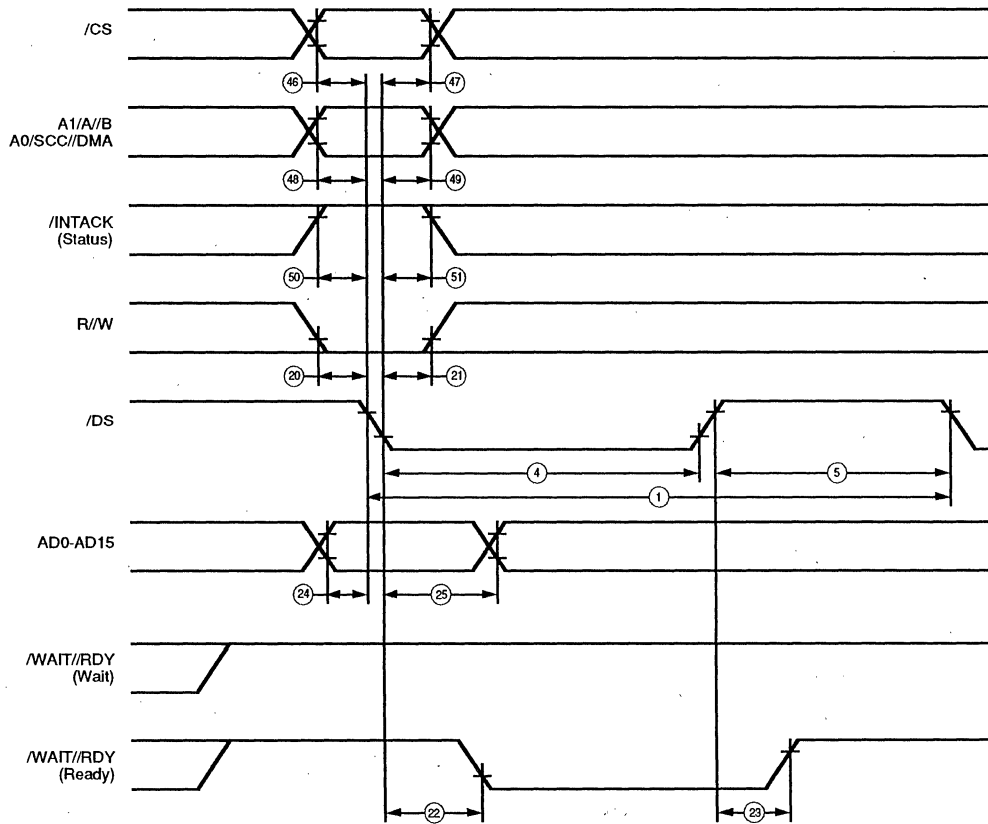


Figure 33. Non-multiplexed /DS Write Cycle

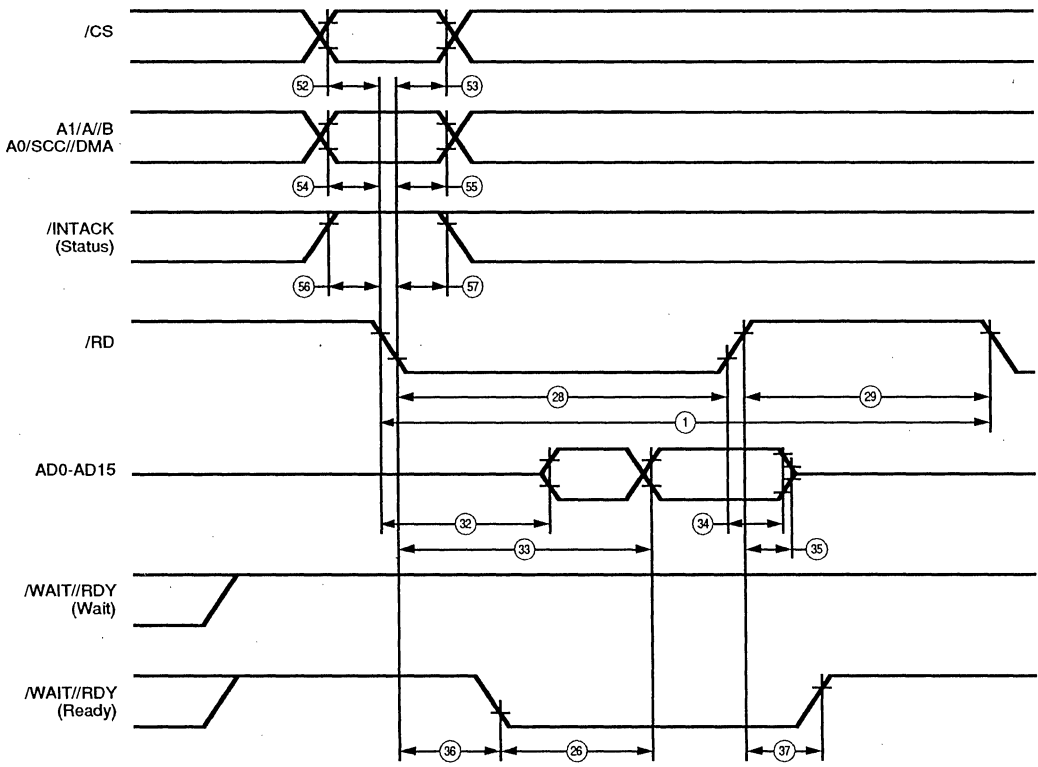


Figure 34. Non-multiplexed /RD Read Cycle

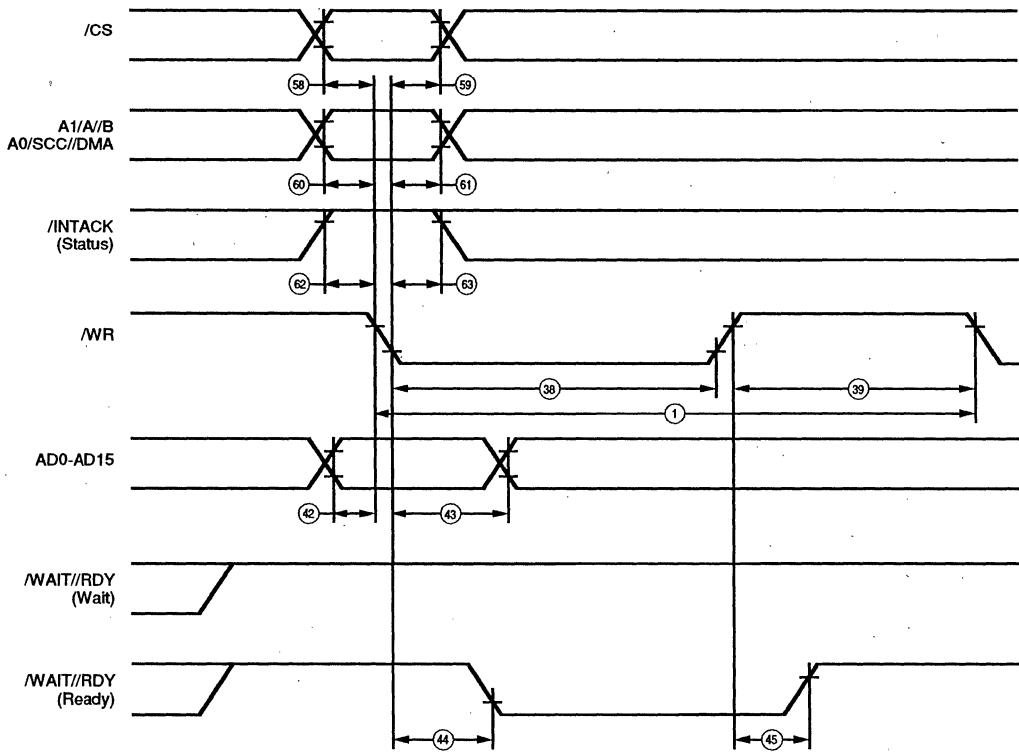


Figure 35. Non-multiplexed /WR Write Cycle

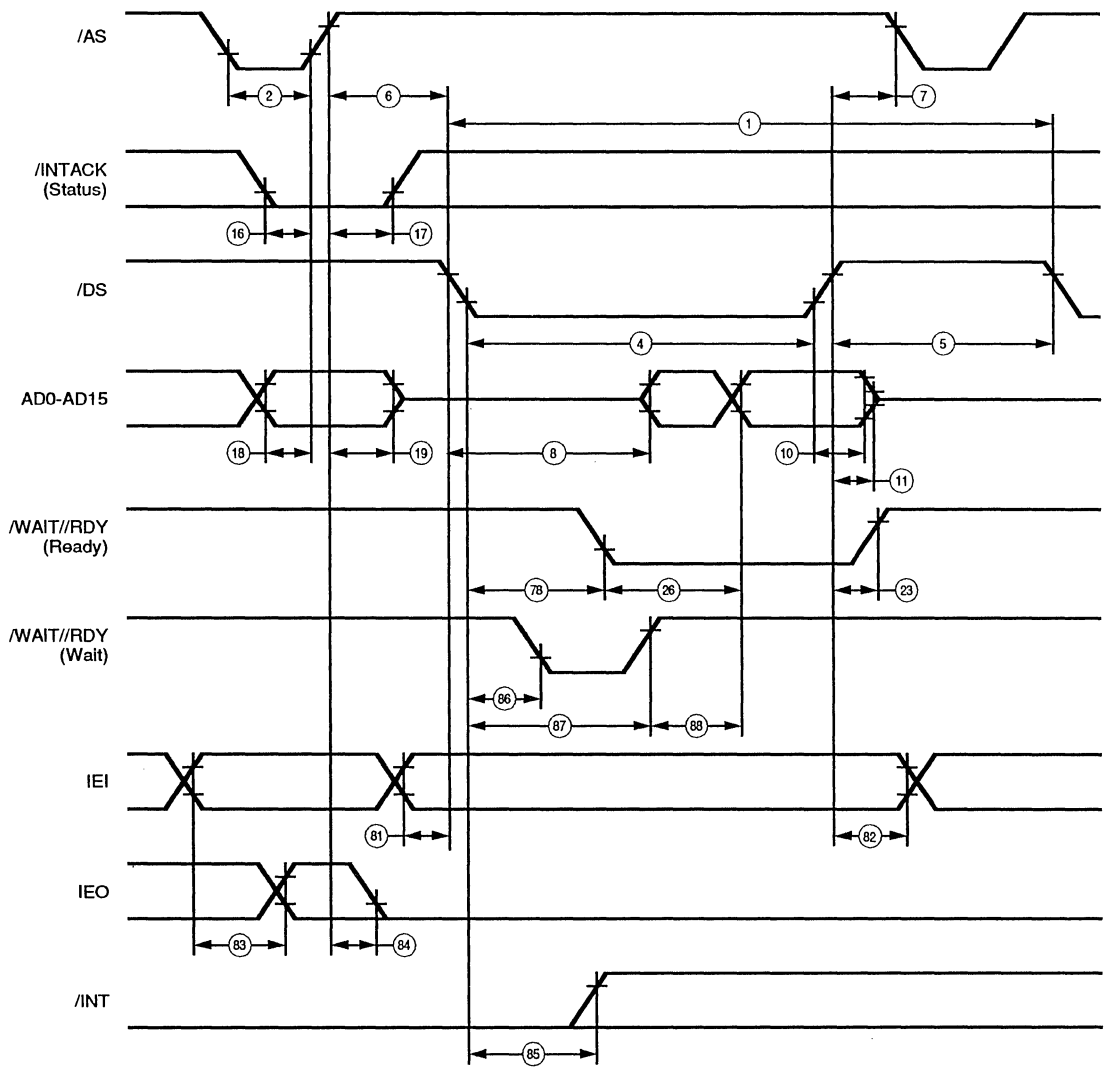


Figure 36. Multiplexed /DS Status INTACK Cycle



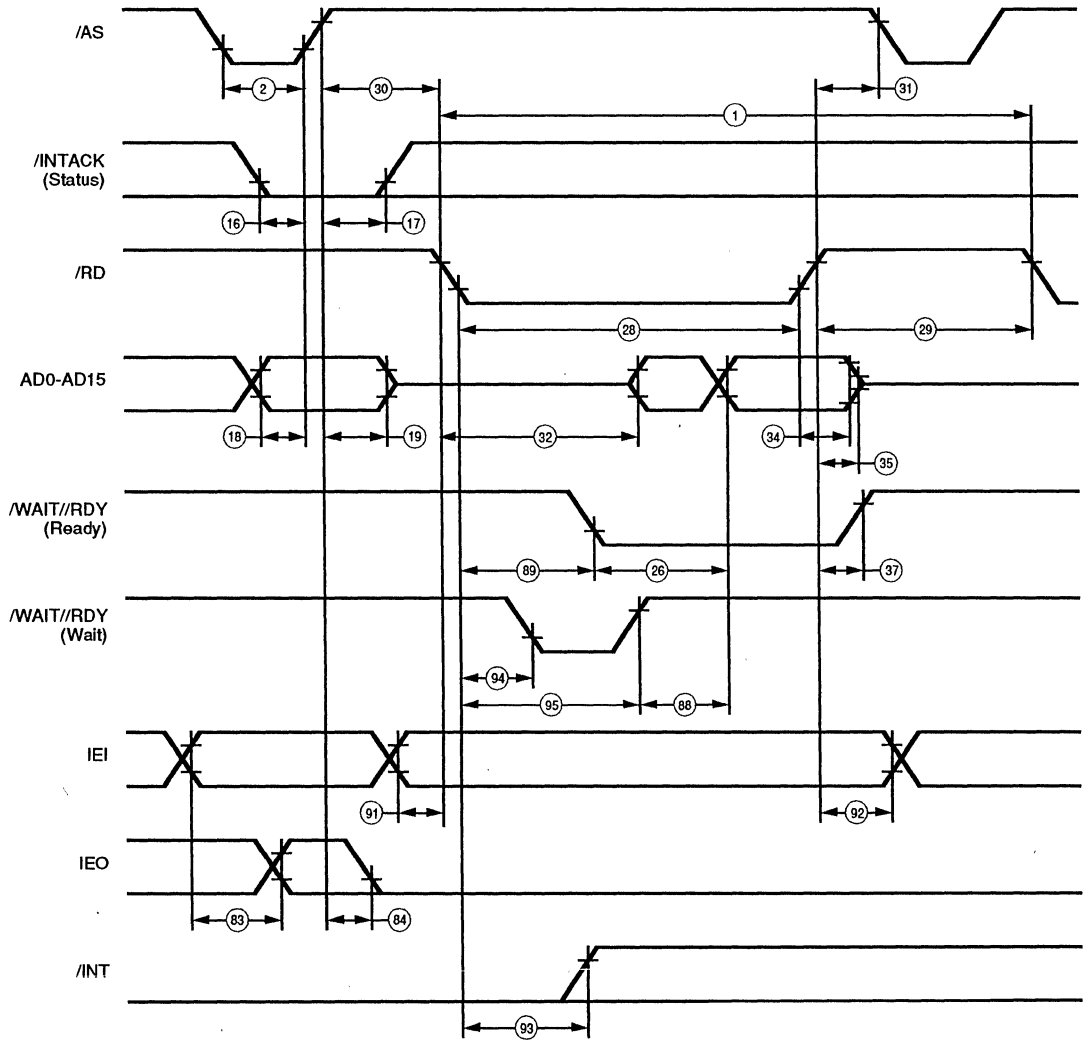


Figure 37. Multiplexed /RD Status INTACK Cycle

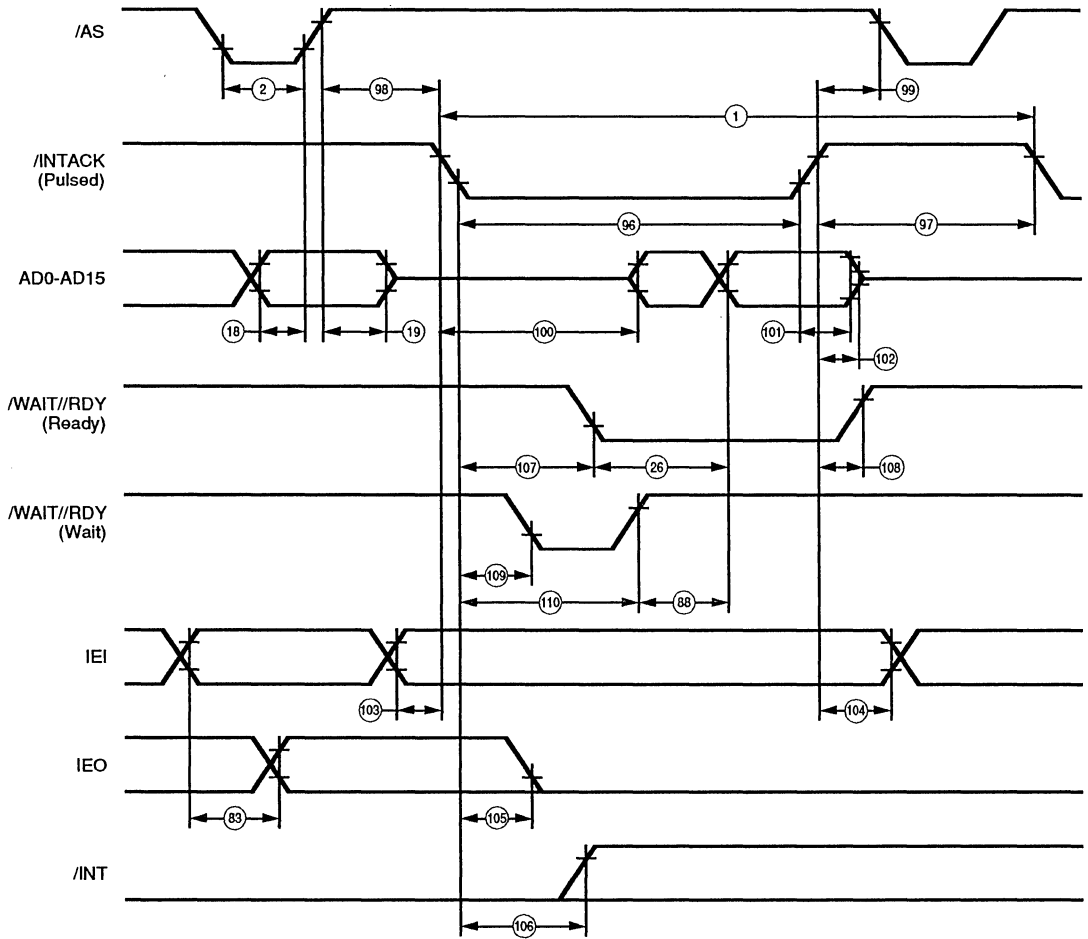


Figure 38. Multiplexed Pulsed INTACK Cycle

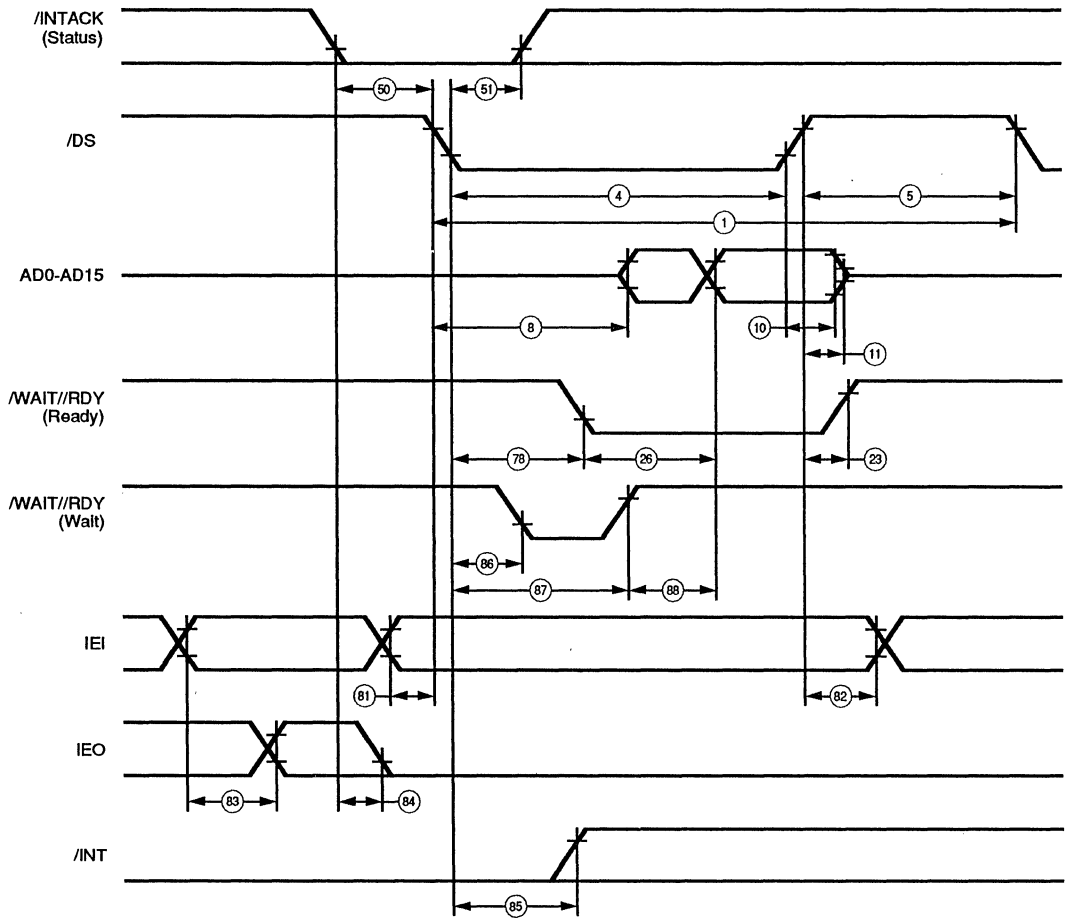


Figure 39. Non-multiplexed /DS INTACK Cycle

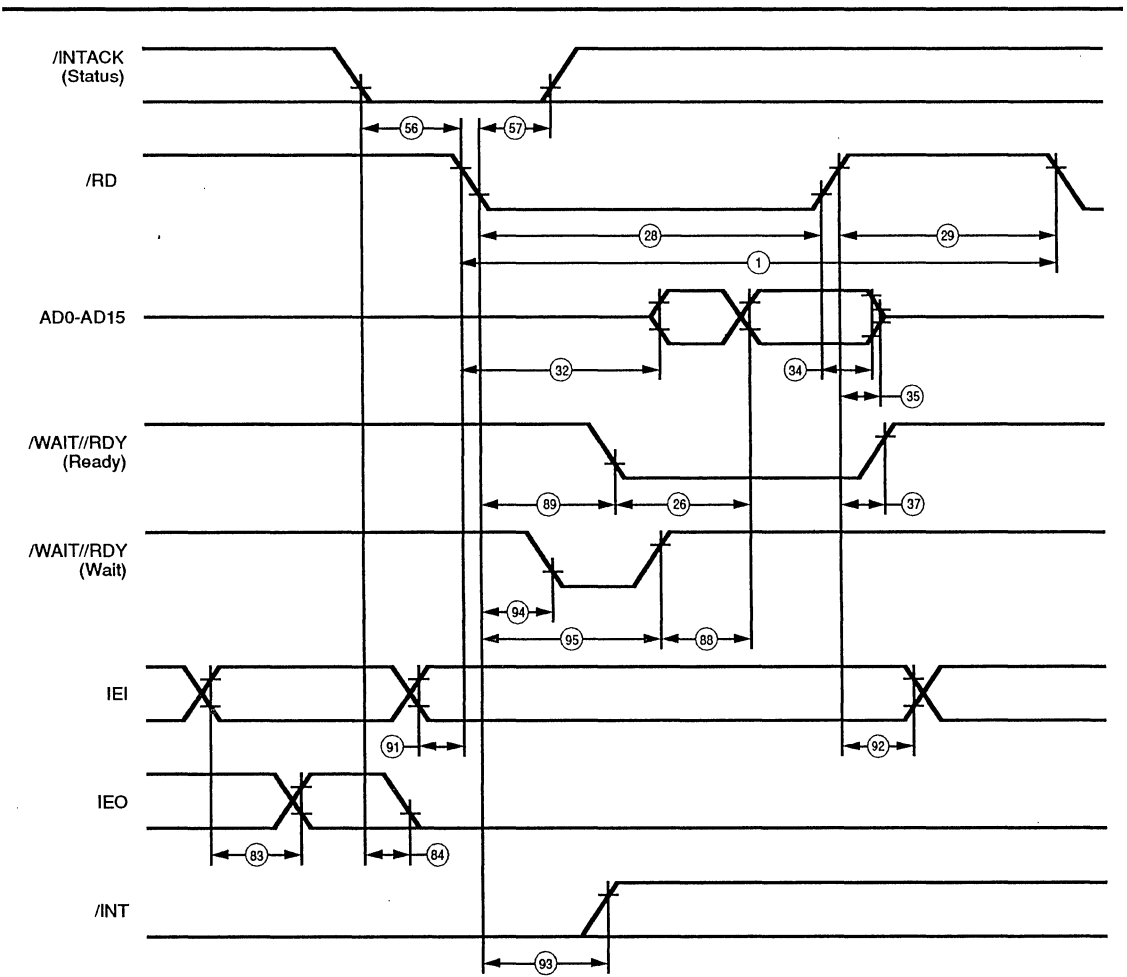


Figure 40. Non-multiplexed /RD Status INTACK Cycle

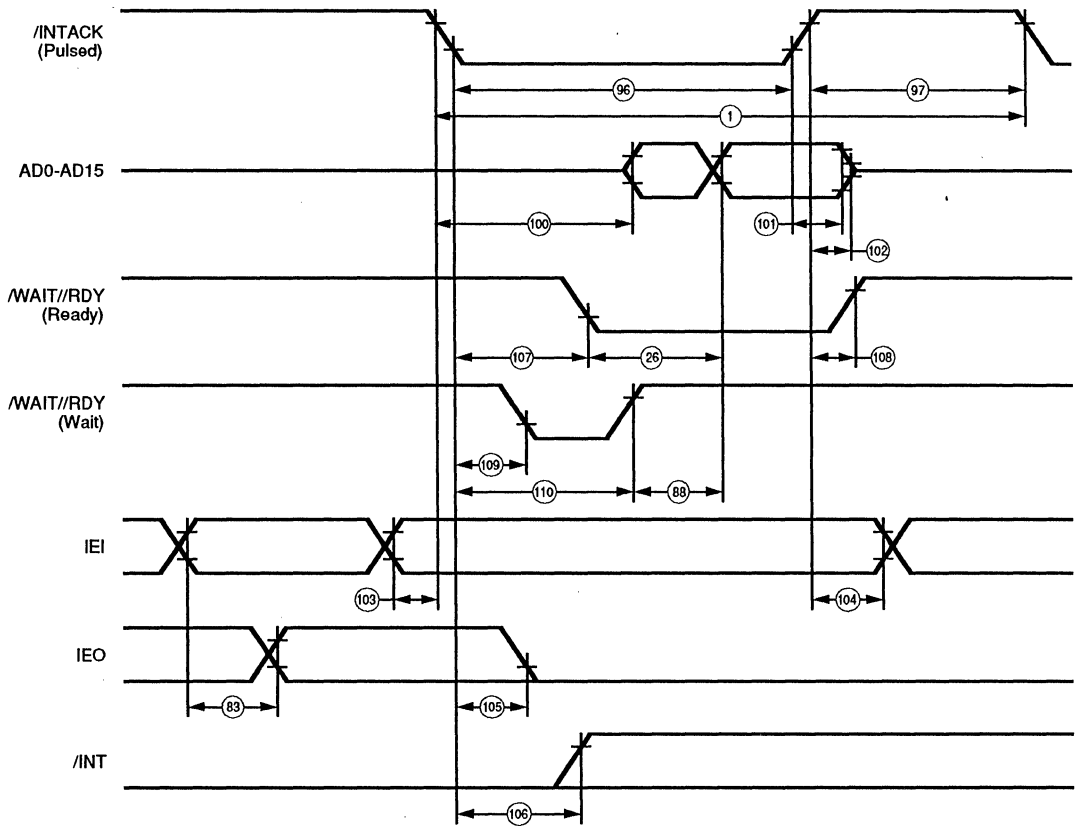


Figure 41. Non-multiplexed Pulsed INTACK Cycle



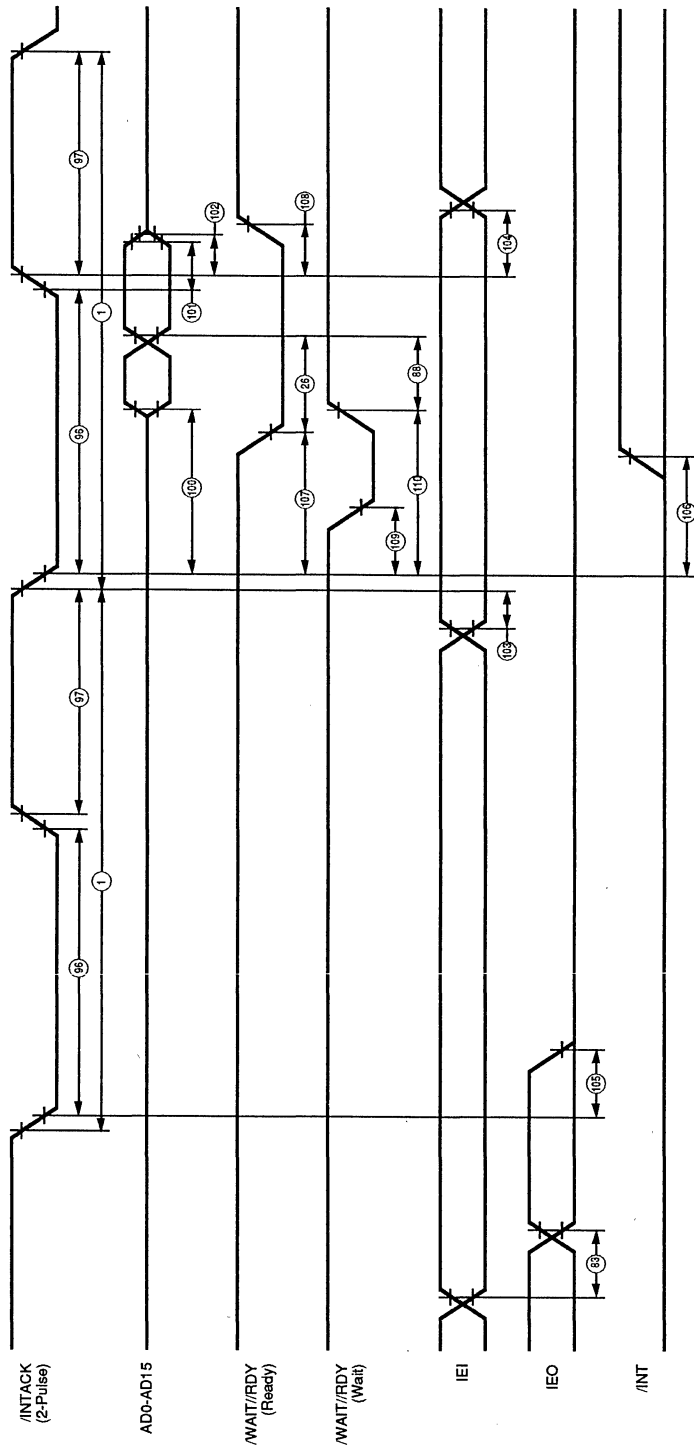


Figure 43. Non-multiplexed Double-Pulsed INTACK Cycle

/RESET

/STB

(13)

(14)

(15)

Figure 44. Reset



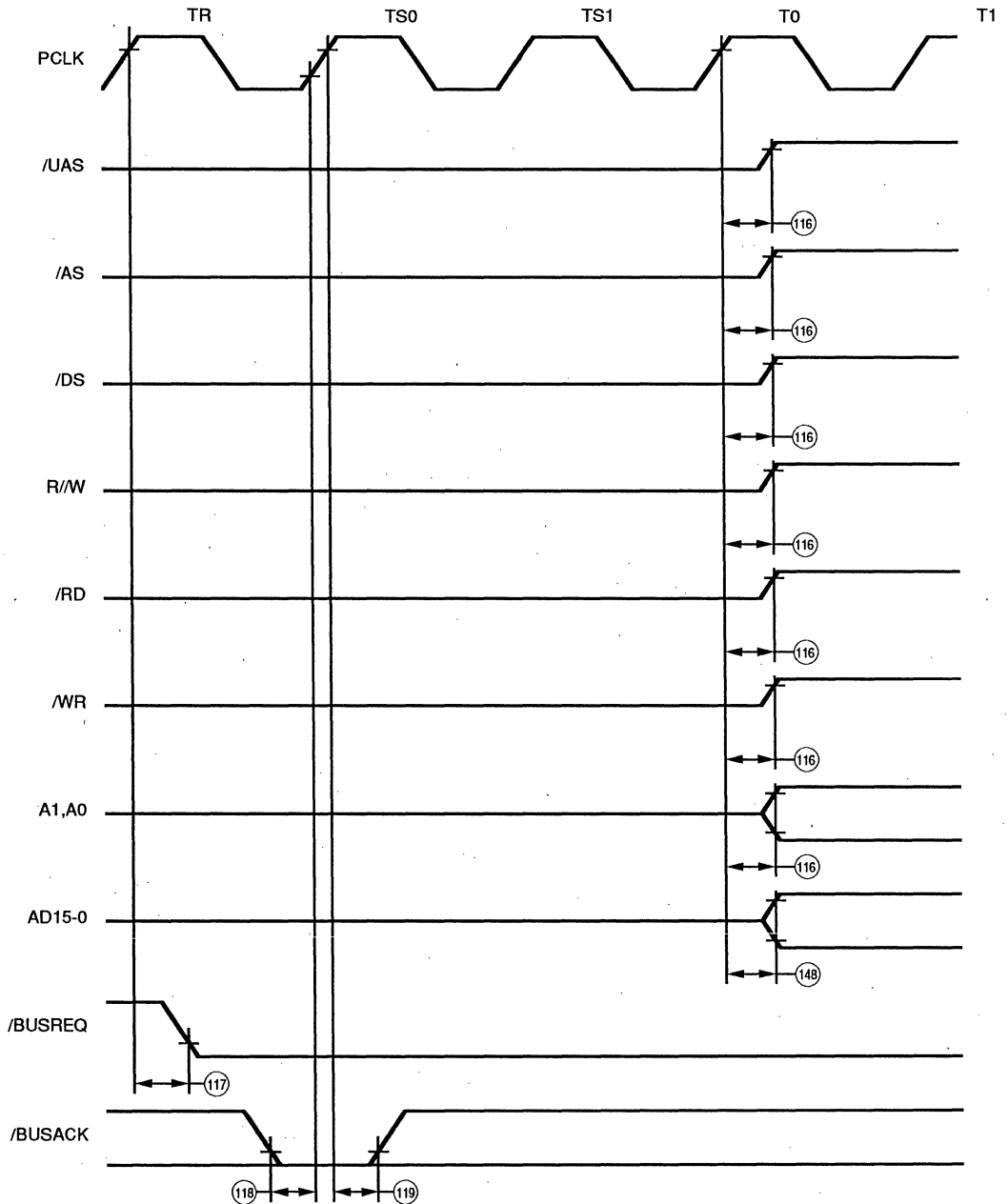


Figure 45. Z16C35 Start-up

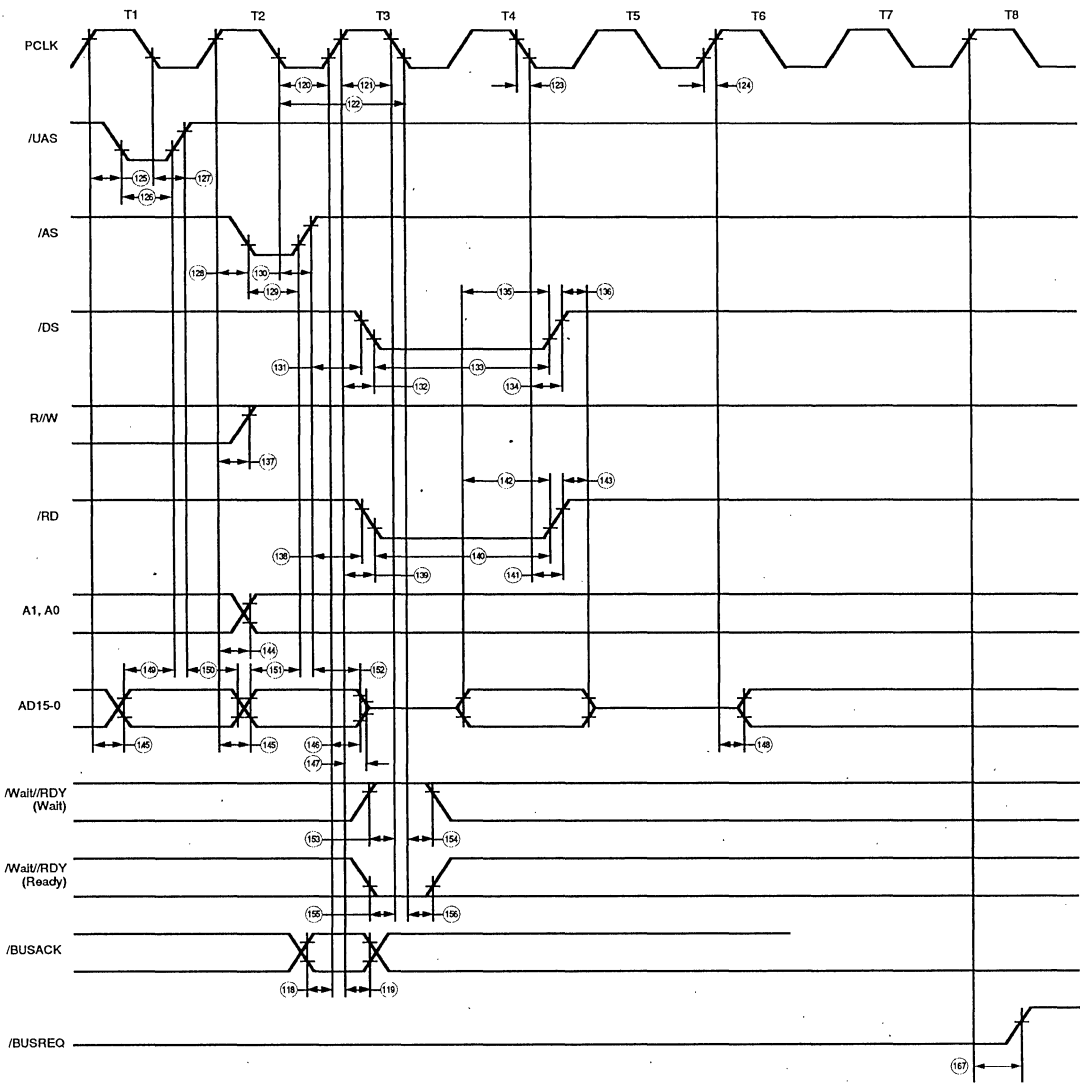


Figure 46. Z16C35 Memory Read

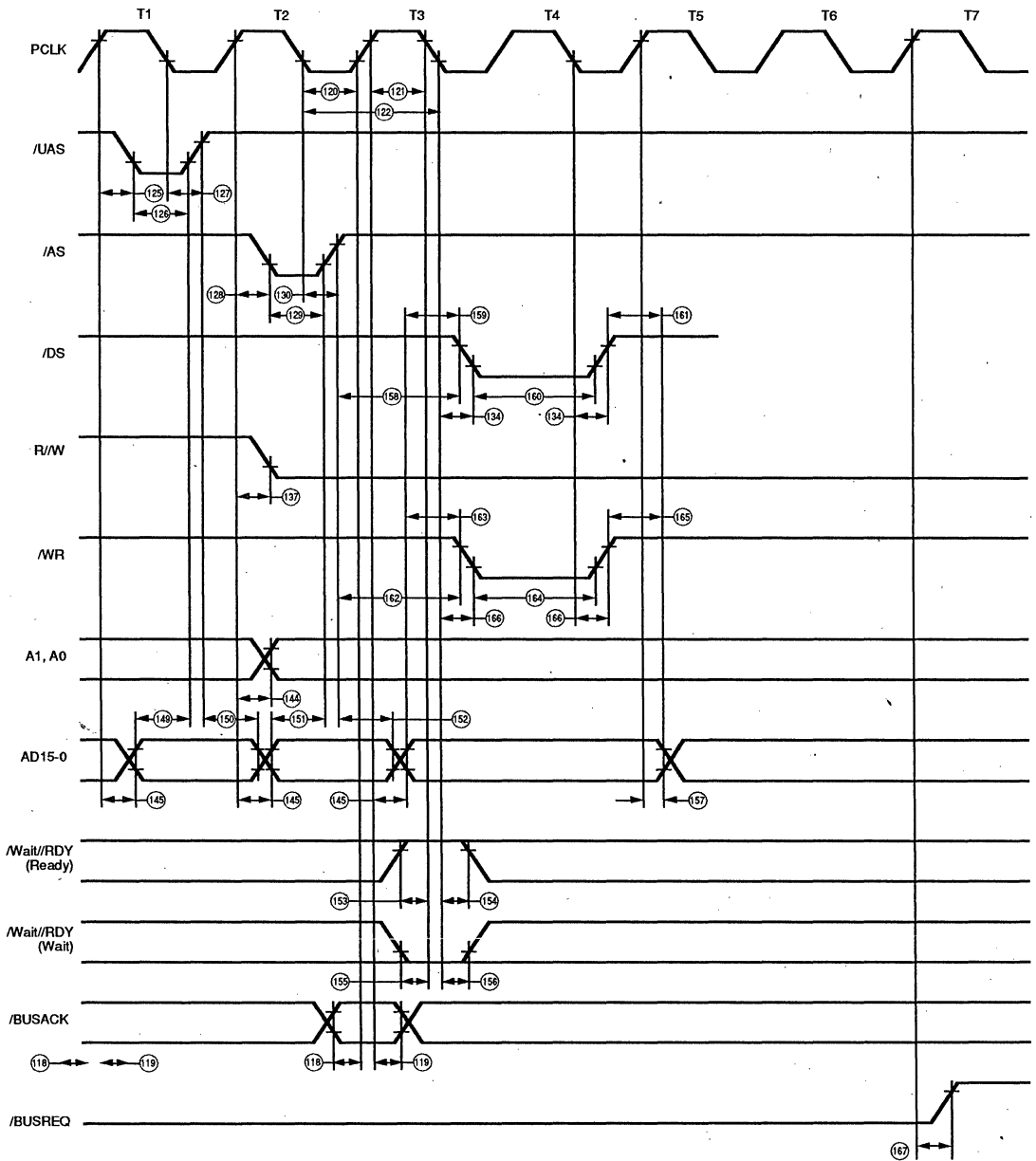


Figure 47. Z16C35 Memory Write

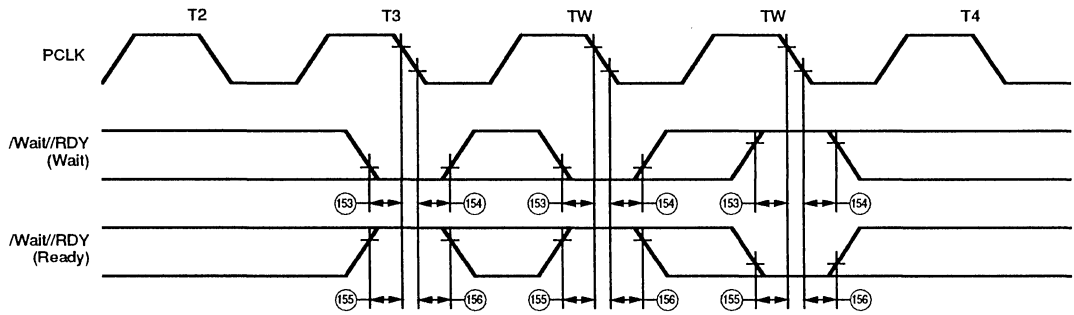


Figure 48. Wait and Ready Timing

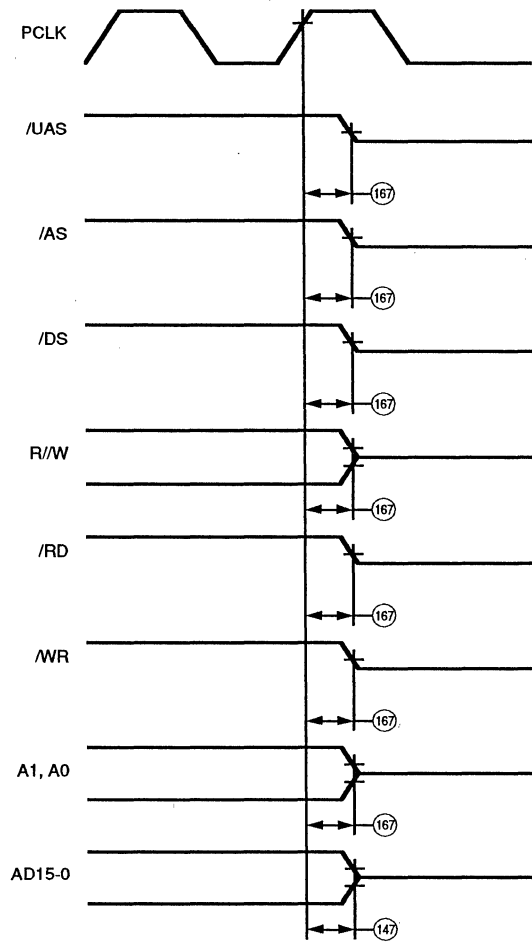


Figure 49. BUS Release

# AC CHARACTERISTICS

## General Timing

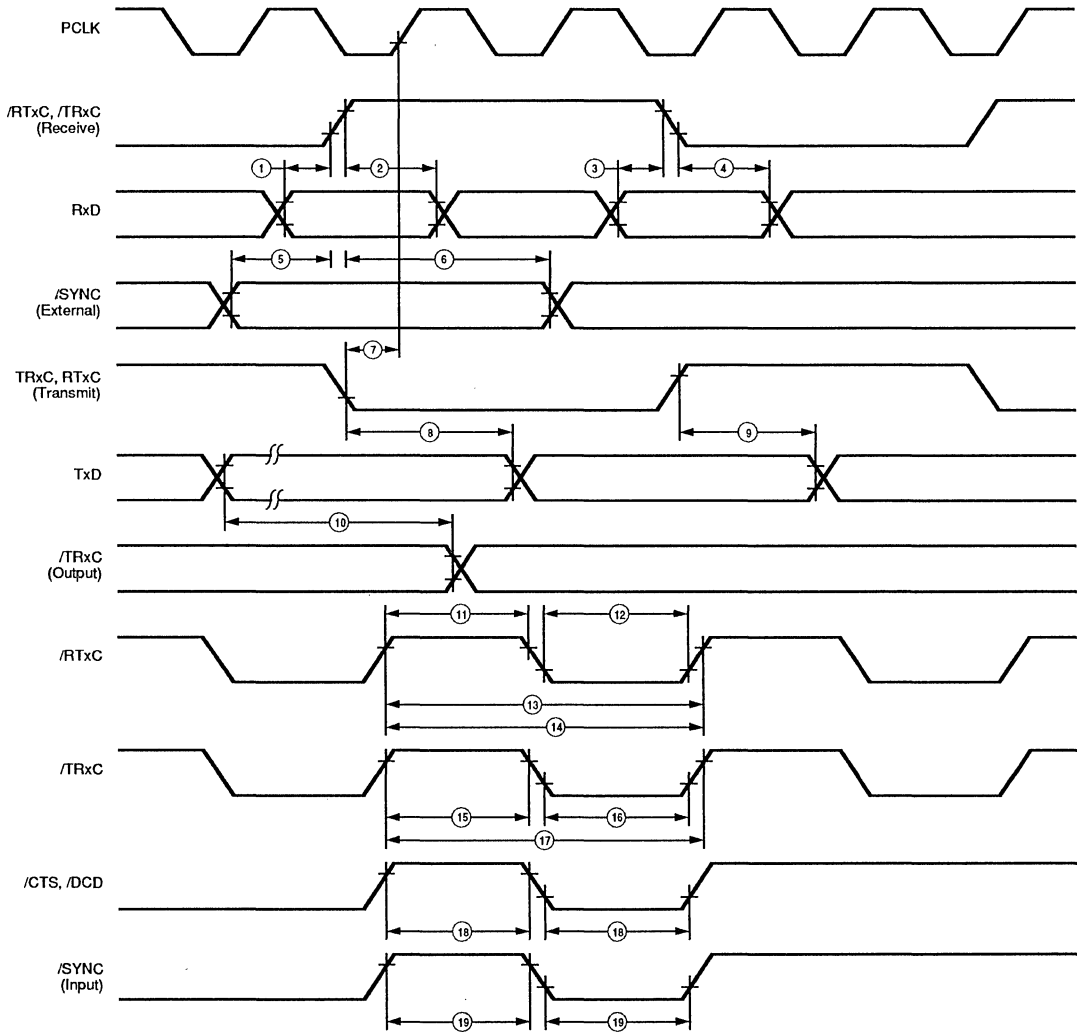


Figure 50. Z16C35 General Timing

## AC CHARACTERISTICS

### General Timing

No	Symbol	Parameter	10 MHz		16 MHz		Notes
			Min	Max	Min	Max	
1	TsRXD(RXCr)	RxD to /RxC Rise Setup Time (x1 mode)	0		0		[1]
2	ThRXD(RXCr)	RxD to /RxC Rise Hold Time (x1 mode)	150		60		[1]
3	TsRXD(RXCf)	RxD to /RxC Fall Setup Time (x1 mode)	0		0		[1,5]
4	ThRXD(RXCf)	RxD to /RxC Fall Hold Time (x1 mode)	150		60		[1,5]
5	TsSY(RXC)	/SYNC to /RxC Rise Setup Time	-200		-100		[1]
6	ThSY(RXC)	/SYNC to RxC Rise Hold Time	5TcPc		5TcPc		[1]
7	TsTXC(PC)	/TxC to PCLK Setup Time	0		0		[2,4]
8	TdTXCf(TXD)	/TxC Fall to TxD Delay (x1 mode)		150		85	[2]
9	TdTxCr(TXD)	/TxC Rise to TxD Delay (x1 mode)		150		85	[2,5]
10	TdTXD(TRX)	TxD to /TRxC Delay (Send Clock Echo)		200		80	
11	TwRTXh	/RTxC High Width	150		80		[6]
12	TwRTXI	/RTxC Low Width	150		80		[6]
13	TcRTX	/RTxC Cycle Time (RxD, TxD)	400		244		[6,7]
14	TcRTXX	Crystal Oscillator Period	100	1000	100	1000	[3]
15	TwTRXh	/TRxC High Width	150		80		[6]
16	TwTRXI	/TRxC Low Width	150		80		[6]
17	TcTRX	/TRxC Cycle Time (RxD, TxD)	400		244		[6,7]
18	TwEXT	/DCD or /CTS Pulse Width	200		70		
19	TwSY	/SYNC Pulse Width	200		70		

#### Notes:

- [1] /RxC is /RTxC or /TRxC, whichever is supplying the receive clock.  
 [2] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.  
 [3] Both /RTxC and /SYNC have 30 pf capacitors to ground connected to them.  
 [4] Parameter applies only if the data rate is one-fourth the PCLK rate. In all other cases, no phase relationship between /RxC and PCLK or/TxC and PCLK is required.  
 [5] Parameter applies only to FM encoding/decoding.  
 [6] Parameter applies only for transmitter and receiver; DPLL and baud rate generator requirements are identical to case PCLK requirements.  
 [7] The maximum receive or transmit data rate is one-fourth PCLK.  
 [8] Timings in nanoseconds.

# AC CHARACTERISTICS

## System Timing

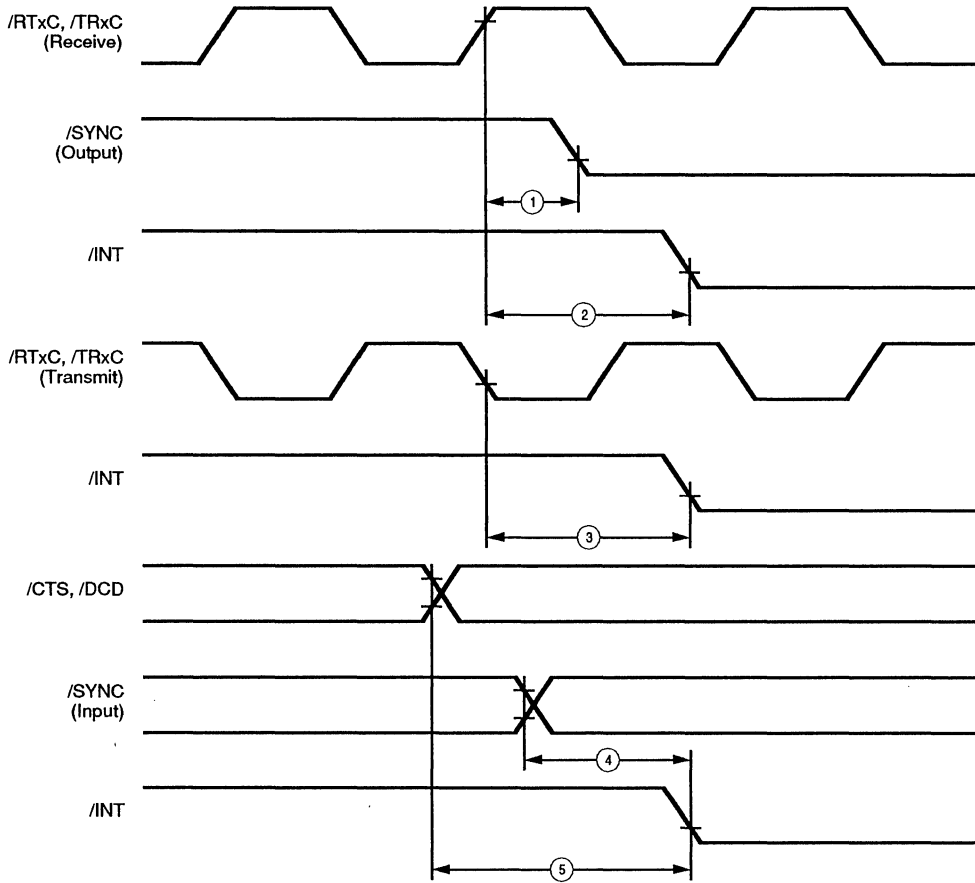


Figure 51. Z16C35 System Timing



---

## AC CHARACTERISTICS

### System Timing

---

No	Symbol	Parameter	10 MHz		16 MHz		Notes[3]
			Min	Max	Min	Max	
1	TdRXC(SY)	/RxC Rise to /SYNC	4	7	4	7	[1]
2	TdRXC(INT)	RxC Rise to /INT Valid Delay	10	16	10	16	[1]
3	TdTXC(INT)	/TxC Fall to /INT Valid Delay	6	10	6	10	
4	TdSY(INT)	/SYNC Transition to /INT Valid Delay	2	6	2	6	
5	TdEXT(INT)	/DCD or /CTS Transition to /INT Valid Delay	2	6	2	6	

---

**Notes:**

[1] /RxC is /RTXC or /TRxC, whichever is supplying the receive clock.

[2] /TxC is /TRxC or /RTxC, whichever is supplying the transmit clock.

[3] Units equal to TcPc.



## Z5380 SCSI

### SMALL COMPUTER SYSTEM INTERFACE

#### FEATURES

- Compatible 5380 pinout
- Low power CMOS
- Asynchronous interface, supports 1.5 MB/s
- Direct SCSI Bus interface with on-board 48 mA drivers
- Supports Target and Initiator roles
- Arbitration support
- DMA or programmed I/O data transfers
- Supports Normal or Block Mode DMA
- Memory or I/O Mapped CPU interface

#### GENERAL DESCRIPTION

The Z5380 SCSI (Small Computer System Interface) controller is a 40-pin DIP or 44-pin PLCC CMOS device (Figure 1). It is designed to implement the SCSI protocol as defined by the ANSI X3.131-1986 standard, and is fully compatible with the industry standard 5380. It is capable of operating both as a Target and as an Initiator. Special high-current open-drain outputs enable it to directly interface to, and drive, the SCSI bus. The Z5380 has the necessary interface hook-ups so the system CPU can communicate with it like with any other peripheral device. The CPU can read from, or write to, the SCSI registers which are addressed as standard or memory-mapped I/Os.

The Z5380 increases the system performance by minimizing the CPU intervention in DMA operations which the SCSI controls. The CPU is interrupted by the SCSI when it detects a bus condition that requires attention. It also supports arbitration and reselection. The Z5380 has the proper hand-shake signals to support normal and block mode DMA operations with most DMA controllers available (Figure 2).

**Note: All Signals with a preceding front slash, "/", are active Low, e.g.: B/W (WORD is active Low); /B/W (BYTE is active Low, only); /N/S (NORMAL and SYSTEM are both active Low).**

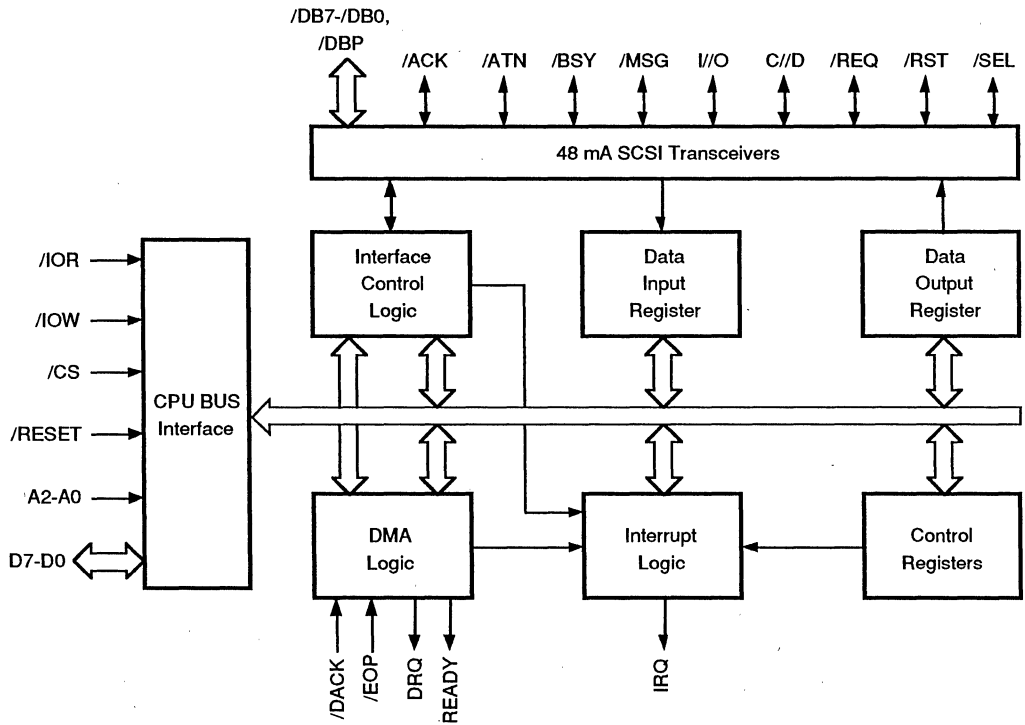


Figure 1. Block Diagram

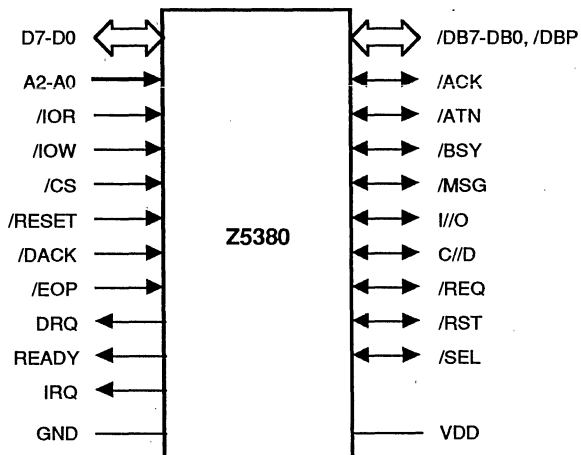


Figure 2. Logic Symbol

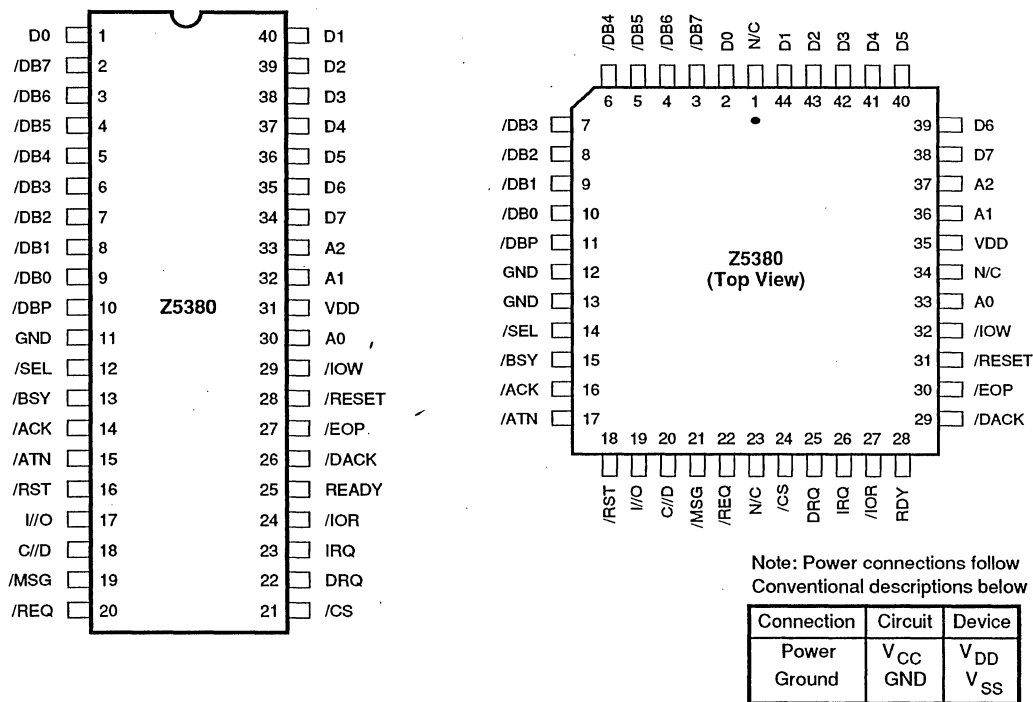


Figure 3. Pin Diagrams

## PIN DESCRIPTION

### Microprocessor Bus

Figure 3 shows the pins and their respective functions for both the DIP and PLCC.

**A2-A0. Address Lines (Input).** Address lines are used with /CS, /IOR, or /IOW to address all internal registers.

**/CS. Chip Select (Input, Active Low).** This signal, in conjunction with /IOR or /IOW, enables the internal register selected by A2-A0, to be read from or written to.

**/DACK. DMA Acknowledge (Input, Active Low).** /DACK resets DRQ and selects the data register for input or output data transfers. /DACK is used by DMA controller instead of /CS.

**DRQ. DMA Request (Output, Active High).** DRQ indicates that the data register is ready to be read or written. DRQ is asserted only if DMA mode is set in the Command Register. DRQ is cleared by /DACK.

**D7-D0. Data Lines (Bidirectional, three-state, Active High).** Bidirectional microprocessor data bus lines. D0 is the Least Significant Bit of the bus. Data bus lines carry data and commands to and from the SCSI.

**/EOP. End of Process (Input, Active Low).** /EOP is used to terminate a DMA transfer. If asserted during a DMA cycle, the current byte will be transferred, but no additional bytes will be requested.

**/IOR. I/O Read (Input, Active Low).** /IOR is used in conjunction with /CS and A2-A0 to read an internal register. It also selects the Input Data Register when used with /DACK.

**/IOW. I/O Write (Input, Active Low).** /IOW is used in conjunction with /CS and A2-A0 to write an internal register. It also selects the Output Data Register when used with /DACK.

---

## PIN DESCRIPTION (Continued)

**IRQ.** *Interrupt Request* (Output, Active High). IRQ alerts a microprocessor of an error condition or an event completion.

**READY.** *Ready* (Output, Active High). Ready is used to control the speed of Block Mode DMA transfers. This signal goes active to indicate the chip is ready to send/receive data and remains Low after a transfer until the last byte is sent or until the DMA Mode bit is reset.

**/RESET.** *Reset* (Input, Active Low). /RESET clears all registers. It has no effect upon the SCSI /RST signal.

### SCSI Bus

The following signals are all bidirectional, active Low, open-drain, with 48 mA sink capability. All pins interface directly with the SCSI bus.

**/ACK.** *Acknowledge* (Bidirectional, Open-drain, Active Low). Driven by an Initiator, /ACK indicates an acknowledgement for a /REQ//ACK data-transfer handshake. In the Target role, /ACK is received as a response to the /REQ signal.

**/ATN.** *Attention* (Bidirectional, Open-drain, Active Low). Driven by an Initiator, received by the Target, /ATN indicates an Attention condition.

**/BSY.** *Busy* (Bidirectional, Open-drain, Active Low). This signal indicates that the SCSI bus is being used and can be driven by both the Initiator and the Target device.

**C//D.** *Control/Data* (Bidirectional, Open-drain). Driven by the Target and received by the Initiator, C//D indicates whether Control or Data information is on the Data Bus. True indicates Control.

**/DB7-/DB0, /DBP.** *Data Bus Bits, Data Bus Parity Bit* (Bidirectional, Open-drain). These eight data bits (/DB7-/DB0), plus a parity bit (/DBP) form the data bus. /DB7 is the most significant bit (MSB) and has the highest priority during the Arbitration phase. Data parity is odd. Parity is always generated and optionally checked. Parity is not valid during Arbitration.

**/I/O.** *Input/Output* (Bidirectional, Open-drain). I/O is a signal driven by a Target which controls the direction of data movement on the SCSI bus. True indicates input to the Initiator. This signal is also used to distinguish between Selection and Reselection phases.

**/MSG.** *Message* (Bidirectional, Open-drain, Active Low). This signal is driven by the Target during the Message phase. This signal is received by the Initiator.

**/REQ.** *Request* (Bidirectional, Open-drain, Active Low). Driven by the Target and received by the Initiator, this signal indicates a request for a /REQ//ACK data-transfer handshake.

**/RST.** *SCSI Bus Reset* (Bidirectional, Open-drain, Active Low). This signal indicates a SCSI bus Reset condition.

**/SEL.** *Select* (Bidirectional, Open-drain, Active Low). This signal is used by an Initiator to select a Target, or by a Target to reselect an Initiator.

### Power Signals:

GND. Ground (0V)

VDD. VDD Supply (+5V)

---

## FUNCTIONAL DESCRIPTION

The Z5380 Small Computer System Interface (SCSI) has a set of eight registers that are controlled by the CPU. By reading and writing the appropriate registers, the CPU may initiate any SCSI Bus activity or may sample and assert any signal on the SCSI Bus. This allows the user to

implement all or any of the SCSI protocol in software. These registers are read (written) by activating /CS with an address on A2-A0 and then issuing an /IOR (/IOW) pulse. This section describes the operation of the internal registers (Table 1).

**Table 1. Register Summary**

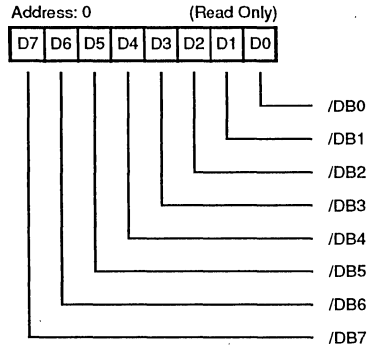
Address				Register Name
A2	A1	A0	R/W	
0	0	0	R	Current SCSI Data
0	0	0	W	Output Data
0	0	1	R/W	Initiator Command
0	1	0	R/W	Mode
0	1	1	R/W	Target Command
1	0	0	R	Current SCSI Bus Status
1	0	0	W	Select Enable
1	0	1	R	Bus and Status
1	0	1	W	Start DMA Send
1	1	0	R	Input Data
1	1	0	W	Start DMA Target Receive
1	1	1	R	Reset Parity/Interrupt
1	1	1	W	Start DMA Initiator Receive

**Data Registers**

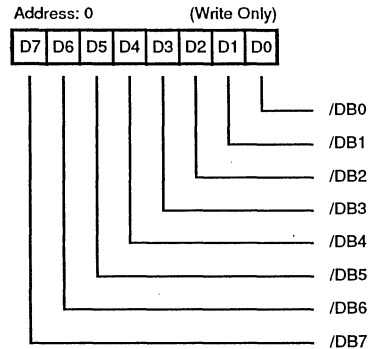
The data registers are used to transfer SCSI commands, data, status, and message bytes between the microprocessor Data Bus and the SCSI Bus. The Z5380 does not interpret any information that passes through the data registers. The data registers consist of the transparent Current SCSI Data Register, the Output Data Register, and the Input Data Register.

**Current SCSI Data Register.** *Address 0 (Read Only).* The Current SCSI Data Register (Figure 4) is a read-only register which allows the microprocessor to read the active SCSI Data Bus. This is accomplished by activating /CS with an address on A2-A0 of 000 and issuing an /IOR pulse. If parity checking is enabled, the SCSI Bus parity is checked at the beginning of the read cycle. This register is used during a programmed I/O data read or during Arbitration to check for higher priority arbitrating devices. Parity is not guaranteed valid during Arbitration.

**Output Data Register.** *Address 0 (Write Only).* The Output Data Register (Figure 5) is a write-only register that is used to send data to the SCSI Bus. This is accomplished by either using a normal CPU write, or under DMA control, by using /IOW and /DACK. This register also asserts the proper ID bits on the SCSI Bus during the Arbitration and Selection phases.



**Figure 4. Current SCSI Data Register**



**Figure 5. Output Data Register**

**Initiator Command Register.** *Address 1 (Read/Write).* The Initiator Command Register (Figures 6 and 7) are read and write registers which assert certain SCSI Bus signals, monitors those signals, and monitors the progress of bus arbitration. Many of these bits are significant only when being used as an Initiator; however, most can be used during Target role operation.

## FUNCTIONAL DESCRIPTION (Continued)

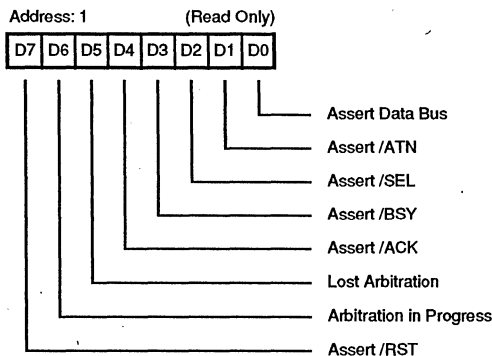


Figure 6. Initiator Command Register

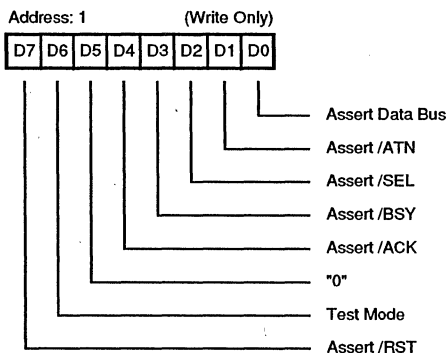


Figure 7. Initiator Command Register

The following describes the operation of all bits in the Initiator Command Register:

**Bit 0. Assert Data Bus.** This bit, when set, allows the contents of the Output Data Register to be enabled as chip outputs on the signals /DB7-DB0. Parity is also generated and asserted on /DBP.

When connected as an Initiator, the outputs are only enabled if the Target Mode bit (Mode Register, bit 6) is 0, the received signal I/O is False, and the phase signals C//D, I//O, and /MSG match the contents of the Assert C//D, Assert I//O, and Assert /MSG in the Target Command Register.

This bit should also be set during DMA send operations.

**Bit 1. Assert /ATN.** /ATN may be asserted on the SCSI Bus by setting this bit to a one (1) if the Target Mode bit (Mode Register, bit 6) is False. /ATN is normally asserted by the initiator to request a Message Out bus phase. Note that since Assert /SEL and Assert /ATN are in the same register, a select with /ATN may be implemented with one CPU write. /ATN may be deasserted by resetting this bit to zero. A read of this register simply reflects the status of this bit.

**Bit 2. Assert /SEL.** Writing a one (1) into this bit position asserts /SEL onto the SCSI Bus. /SEL is normally asserted after Arbitration has been successfully completed. /SEL may be disabled by resetting bit 2 to a zero. A read of this register reflects the status of this bit.

**Bit 3. Assert /BSY.** Writing a one (1) into this bit position asserts /BSY onto the SCSI Bus. Conversely, a zero resets the /BSY signal. Asserting /BSY indicates a successful selection or reselection. Resetting this bit creates a Bus-Disconnect condition. Reading this register reflects bit status.

**Bit 4. Assert /ACK.** Bit 4 is used by the bus initiator to assert /ACK on the SCSI Bus. In order to assert /ACK, the Target Mode bit (Mode Register, bit 6) must be False. Writing a zero to this bit deasserts /ACK. Reading this register reflects bit status.

**Bit 5. "0" (Write Bit).** Bit 5 should be written with a zero for proper operation.

**Bit 5. LA (Lost Arbitration - Read Bit).** Bit 5, when active, indicates that the SCSI detected a Bus-Free condition, arbitrated for use of the bus by asserting /BSY and its ID on the Data Bus, and lost Arbitration due to /SEL being asserted by another bus device. This bit is active only when the Arbitrate bit (Mode Register, bit 0) is active.

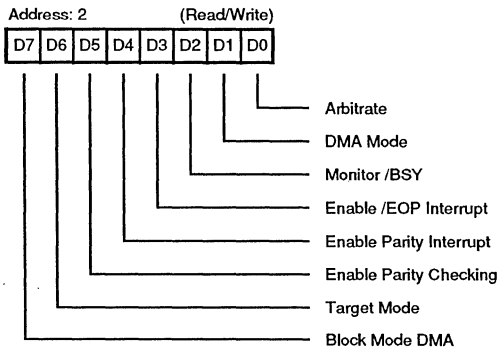
**Bit 6. Test Mode (Write Bit).** Bit 6 is written during a test environment to disable all output drivers, effectively removing the Z5380 from the circuit. Resetting this bit returns the part to normal operation.

**Bit 6. AIP (Arbitration in Process - Read Bit).** Bit 6 is used to determine if Arbitration is in progress. For this bit to be active, the Arbitrate bit (Mode Register, bit 0) must have been set previously. It indicates that a Bus-Free condition has been detected and that the chip has asserted /BSY and put the contents of the Output Data Register onto the SCSI Bus. AIP will remain active until the Arbitrate bit is reset.

**Bit 7. Assert /RST.** Whenever a one is written to bit 7 of the Initiator Command Register, the /RST signal is asserted on the SCSI Bus. The /RST signal will remain asserted until this bit is reset or until an external /RESET occurs. After this bit

is set (1), IRQ goes active and all internal logic and control registers are reset (except for the interrupt latch and the Assert /RST bit). Writing a zero to bit 7 of the Initiator Command Register deasserts the /RST signal. The status of this bit is monitored by reading the Initiator Command Register.

**Mode Register.** Address 2 (Read/Write). The Mode Register controls the operation of the chip. This register determines whether the Z5380 operates as an Initiator or a Target, whether DMA transfers are being used, whether parity is checked, and whether interrupts are generated on various external conditions. This register is read to check the value of these internal control bits (Figure 8).



**Figure 8. Mode Register**

The following describes the operation of all bits in the Initiator Command Register:

**Bit 0. Arbitrate.** The Arbitrate bit is set (1) to start the Arbitration process. Prior to setting this bit, the Output Data Register should contain the proper SCSI device ID value. Only one data bit should be active for SCSI Bus Arbitration. The Z5380 waits for a Bus-Free condition before entering the Arbitration phase. The results of the Arbitration phase is determined by reading the status bits LA and AIP (Initiator Command Register, bits 5 and 6, respectively).

**Bit 1. DMA Mode.** The DMA Mode bit is normally used to enable a DMA transfer and must be set (1) prior to writing Start DMA Send Register, Start DMA Target Register, and Start DMA Initiator Receiver Register. These three registers are used to start DMA transfers. The Target Mode bit (Mode Register, bit 6) must be consistent with writes to Start DMA Target Receive and Start DMA Initiator Receive Registers; i.e., set (1) for a write to Start DMA Target Receive Register and set (0) for Start DMA Initiator Receive

Register. The control bit Assert Data Bus (Initiator Command Register, bit 0) must be True (1) for all DMA send operations. In the DMA mode, /REQ and /ACK are automatically controlled.

The DMA Mode bit is not reset upon the receipt of an /EOP signal. Any DMA transfer is stopped by writing a zero into this bit location; however, care must be taken not to cause /CS and /DACK to be active simultaneously.

**Bit 2. Monitor Busy.** The Monitor Busy bit, when True (1), causes an interrupt to be generated for an unexpected loss of /BSY. When the interrupt is generated due to loss of /BSY, the lower six bits of the Initiator Command Register are reset (0) and all signals are removed from the SCSI Bus.

**Bit 3. Enable /EOP interrupt.** The enable /EOP interrupt bit, when set (1), causes an interrupt to occur when the /EOP (End Of Process) signal is received from the DMA controller logic.

**Bit 4. Enable Parity Interrupt.** The Enable Parity Interrupt bit, when set (1), will cause an interrupt (IRQ) to occur if a parity error is detected. A parity interrupt will only be generated if the Enable Parity Checking bit (bit 5) is also enabled (1).

**Bit 5. Enable Parity Checking.** The Enable Parity Checking bit determines whether parity errors are ignored or saved in the parity error latch. If this bit is reset (0), parity is ignored. Conversely, if this bit is set (1), parity errors are saved.

**Bit 6. Target Mode.** The Target Mode bit allows the Z5380 to operate as a SCSI Bus Initiator or Target. With this bit reset (0), the Z5380 operates as a SCSI Bus Initiator. Setting Target Mode bit to 1 programs the Z5380 to operate as a SCSI Bus Target device. If the signals /ATN and /ACK are to be asserted on the SCSI Bus, the Target Mode bit must be reset (0). If the signals C//D, I//O, /MSG, and /REQ are to be asserted on the SCSI Bus, the Target Mode bit must be set (1).

**Bit 7. Block Mode DMA.** The Block Mode DMA bit controls the characteristics of the DMA DRQ-/DACK handshake. When this bit is reset (0) and the DMA Mode bit is active (1), the DMA handshake uses the normal interlocked handshake, and the rising edge of /DACK indicates the end of each byte being transferred. In Block Mode operation, when the Block Mode DMA bit is set (1) and DMA Mode bit is active (1), the end of /IOR or /IOW signifies the end of each byte transferred and /DACK is allowed to remain active throughout the DMA operation. Ready can then be used to request the next transfer.



## FUNCTIONAL DESCRIPTION (Continued)

**Target Command Register.** *Address 3(Read/Write).* When connected as a target device, the Target Command Register (Figure 9) allows the CPU to control the SCSI Bus Information Transfer phase and/or to assert /REQ by writing this register. The Target Mode bit (Mode Register, bit 6) must be True (1) for bus assertion to occur. The SCSI Bus phases are described in Table 2.

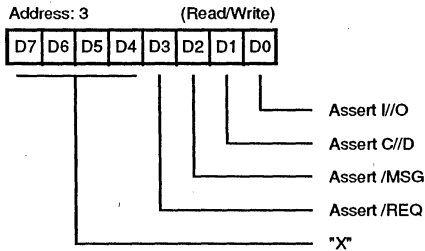


Figure 9. Target Command Register

Table 2. SCSI Information Transfer Phases

Bus Phase	Assert I/O	Assert C//D	Assert /MSG
Data Out	0	0	0
Unspecified	0	0	1
Command	0	1	0
Message Out	0	1	1
Data In	1	0	0
Unspecified	1	0	1
Status	1	1	0
Message In	1	1	1

When connected as an Initiator with DMA Mode bit True, if the phase lines (/I/O, C//D, and /MSG) do not match the phase bits in the Target Command Register, a phase mismatch interrupt is generated when /REQ goes active. To send data as an Initiator, the Assert I/O, Assert C//D, and Assert /MSG bits must match the corresponding bits in the Current SCSI Bus Status Register. The Assert /REQ bit (bit 3) has no meaning when operating as an Initiator.

Bits 4, 5, 6, and 7 are not used.

**Current SCSI Bus Status Register.** *Address 4(Read Only).* The Current SCSI Bus Register is a read-only register which is used to monitor seven SCSI Bus control signals, plus the Data Bus parity bit. For example, an Initiator

device can use this register to determine the current bus phase and to poll /REQ for pending data transfers. This register may also be used to determine why a particular interrupt occurred. Figure 10 describes the Current SCSI Bus Status Register.

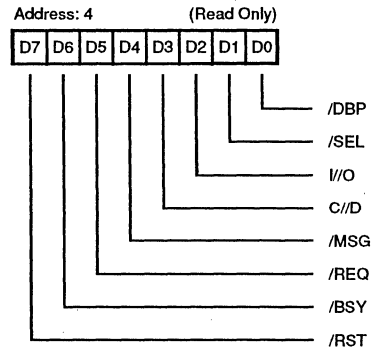


Figure 10. Current SCSI Bus Status Register

**Select Enable Register.** *Address 4(Write Only).* The Select Enable Register (Figure 11) is a write-only register which is used as a mask to monitor a signal ID during a selection attempt. The simultaneous occurrence of the correct ID bit, /BSY False, and /SEL True causes an interrupt. This interrupt can be disabled by resetting all bits in this register. If the Enable Parity Checking bit (Mode Register, bit 5) is active (1), parity is checked during selection.

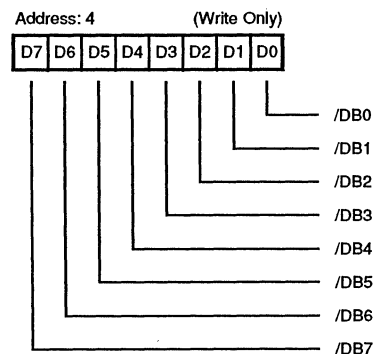
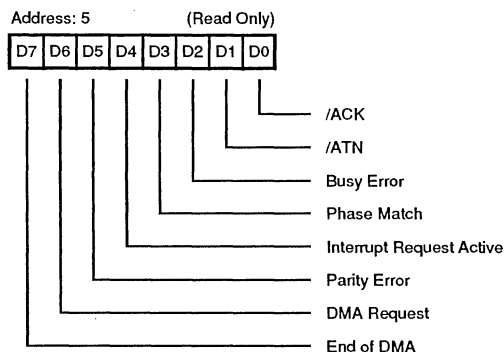


Figure 11. Select Enable Register

**Bus and Status Register.** Address 5 (Read Only). The Bus and Status Register (Figure 12) is a read-only register which can be used to monitor the remaining SCSI control signals not found in the Current SCSI Bus Status Registers (/ATN and /ACK), as well as six other status bits. The following describes each bit of the Bus and Status Register individually.



**Figure 12. Bus and Status Register**

**Bit 0. /ACK.** Bit 0 reflects the condition of the SCSI Bus control signal /ACK. This signal is normally monitored by the Target device.

**Bit 1. /ATN.** Bit 1 reflects the condition of the SCSI Bus control signal /ATN. This signal is normally monitored by the Target device.

**Bit 2. Busy Error.** The Busy Error bit is active if an unexpected loss of the /BSY signal has occurred. This latch is set whenever the Monitor Busy bit (Mode Register, bit 2) is True and /BSY is False. An unexpected loss of /BSY disables any SCSI outputs and resets the DMA Mode bit (Mode Register, bit 1).

**Bit 3. Phase Match.** The SCSI signals /MSG, C//D, and I//O, represent the current information Transfer phase. The Phase Match bit indicates whether the current SCSI Bus phase matches the lower 3 bits of the Target Command Register. Phase Match is continuously updated and is only significant when operating as a Bus Initiator. A phase match is required for data transfers to occur on the SCSI Bus.

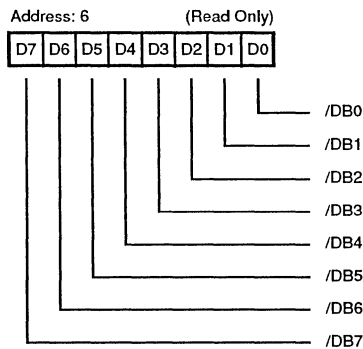
**Bit 4. Interrupt Request ACTIVE.** Bit 4 is set if an enabled interrupt condition occurs. It reflects the current state of the IRQ output and can be cleared by reading the Reset Parity/Interrupt Register.

**Bit 5. Parity Error.** Bit 5 is set if a parity error occurs during a data receive or a device selection. The Parity Error bit can only be set (1) if the Enable Parity Check bit (Mode Register, bit 5) is active (1). This bit may be cleared by reading the Reset Parity/Interrupt Register.

**Bit 6. DMA Request.** The DMA Request bit allows the CPU to sample the output pin DRQ. DRQ can be cleared by asserting /DACK or by resetting the DMA Mode bit (bit 1) in the Mode Register. The DRQ signal does not reset when a phase-mismatch interrupt occurs.

**Bit 7. End of DMA Transfer.** The End of DMA Transfer bit is set if /EOP, /DACK, and either /I/O or /I/O are simultaneously active for at least 100ns. Since the /EOP signal can occur during the last byte sent to the Output Data Register, the /REQ and /ACK signals should be monitored to ensure that the last byte has been transferred. This bit is reset when the DMA Mode bit is reset (0) in the Mode Register.

**Input Data Register.** Address 6 (Read Only). The input Data Register (Figure 13) is a read-only register that is used to read latched data from the SCSI Bus. Data is latched either during a DMA Target receive operation when /ACK goes active or during a DMA Initiator receive when /REQ goes active. The DMA Mode bit (bit 1) must be set before data can be latched in the Input Data Register. This register is read under DMA control using /I/O and /DACK. Parity is optionally checked when the Input Data Register is loaded.



**Figure 13. Input Data Register**

---

## FUNCTIONAL DESCRIPTION (Continued)

### DMA Registers

Three write-only registers are used to initiate all DMA activity. They are: Start DMA Send, Start DMA Target Receive, and Start DMA Initiator Receive. Performing a write operation into one of these registers starts the desired type of DMA transfer. Data presented to the Z5380 on signals D7-D0 during the register write is meaningless and has no effect on the operation. Prior to writing these registers, the Block Mode DMA bit (bit 7), the DMA Mode bit (bit 1), and the Target Mode bit (bit 6) in the Mode Register must be appropriately set. The individual registers are briefly described as follows:

**Start DMA Send.** *Address 5 (Write Only).* This register is written to initiate a DMA send, from the DMA to the SCSI Bus, for either Initiator or Target role operations. The DMA Mode bit (Mode Register, bit 1) is set prior to writing this register.

**Start DMA Target Receive.** *Address 6 (Write Only).* This register is written to initiate a DMA receive - from the SCSI Bus to the DMA, for Target operation only. The DMA Mode bit (bit 1) and the Target Mode bit (bit 6) in the Mode Register must both be set (1) prior to writing this register.

**Start DMA Initiator Receive.** *Address 7 (Write Only).* This register is written to initiate a DMA receive - from the SCSI Bus to the DMA, for Initiator operation only. The DMA Mode bit (bit 6) must be False (0) in the Mode Register prior to writing this register.

**Reset Parity/Interrupt.** *Address 7 (Read Only).* Reading this register resets the Parity Error bit (bit 5), the Interrupt Request bit (bit 4), and the Busy Error bit (bit 2) in the Bus and Status Register.

### On-Chip SCSI Hardware Support

The Z5380 is easy to use because of its simple architecture. The chip allows direct control and monitoring of the SCSI Bus by providing a latch for each signal. However, portions of the protocol define timings which are much too quick for traditional microprocessors to control. Therefore, hardware support has been provided for DMA transfers, bus arbitration, phase change monitoring, bus disconnection, bus reset, parity generation, parity checking, and device selection/reselection.

Arbitration is accomplished using a bus-free filter to continuously monitor /BSY. If /BSY remains inactive for at least 1.2 $\mu$ s, the SCSI Bus is considered free and Arbitration may begin. Arbitration will begin if the bus is free, /SEL is inactive, and the Arbitrate bit (Mode Register, bit 0) is

active. Once arbitration has begun (/BSY asserted), an arbitration delay of 2.2 $\mu$ s must elapse before the Data Bus can be examined to determine if Arbitration is enabled. This delay is implemented in the controlling software driver.

The Z5380 is a clockwise device. Delays such as bus-free delay, bus-set delay, and bus-settle delay are implemented using gate delays. These delays may differ between devices because of inherent process variations, but are well within the proposed ANSI X3.131 - 1986 specification.

### Interrupts

The Z5380 provides an interrupt output (IRQ) to indicate a task completion or an abnormal bus occurrence. The use of interrupts is optional and may be disabled by resetting the appropriate bits in the Mode Register or the Select Enable Register.

When an interrupt occurs, the Bus and Status Register and the Current SCSI Bus Status Register (Figures 12 and 10) must be read to determine which condition created the interrupt. IRQ can be reset simply by reading the Reset Parity/Interrupt Register or by an external chip reset /RESET active for 200ns.

Assuming the Z5380 has been properly initialized, an interrupt is generated if the chip is selected or reselected; if an /EOP signal occurs during a DMA transfer; if a SCSI Bus reset occurs; if a parity error occurs during a data transfer; if a bus phase mismatch occurs; or if a SCSI Bus disconnection occurs.

### Selection/Reselection Interrupt

The Z5380 generates a select interrupt if /SEL is active (0), its device ID is True and /BSY is False for at least a bus-settle delay. If I/O is active, this is considered a reselect interrupt. The correct ID bit is determined by a match in the Select Enable Register. Only a single bit match is required to generate an interrupt. This interrupt may be disabled by writing zeros into all bits of the Select Enable Register.

If parity is supported, parity should be good during the selection phase. Therefore, if the Enable Parity bit (Mode Register, bit 5) is active, the Parity Error bit is checked to ensure that a proper selection has occurred. The Enable Parity Interrupt bit need not be set for this interrupt to be generated.

The proposed SCSI specification also requires that no more than two device ID's be active during the selection process. To ensure this, the Current SCSI Data Register is read.

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 14 and 15, respectively.

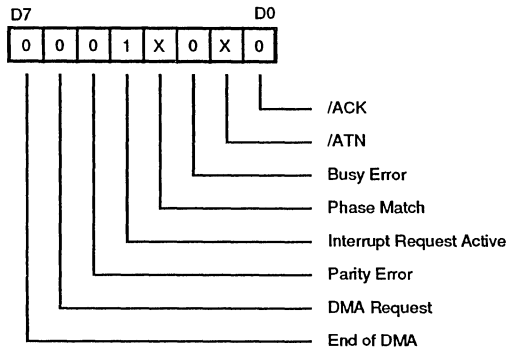


Figure 14. Bus and Status Register

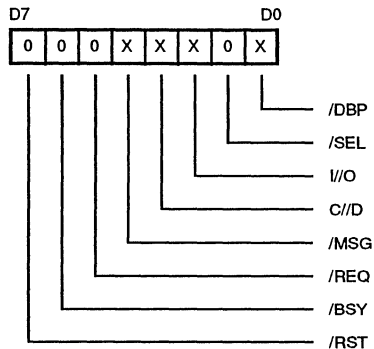


Figure 15. Current SCSI Bus Status Register

**End Of Process (EOP) Interrupt**

An End Of Process signal (EOP) which occurs during a DMA transfer (DMA Mode True) will set the End of DMA Status bit (bit 7) and will optionally generate an interrupt if Enable EOP Interrupt bit (Mode Register, bit 3) is True. The /EOP pulse will not be recognized (End of DMA bit set) unless /EOP, /DACK, and either /IOR or /IOW are concurrently active for at least 100 ns. DMA transfers can still occur if /EOP was not asserted at the correct time. This

interrupt is disabled by resetting the Enable EOP Interrupt bit.

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register for this interrupt are shown in Figures 16 and 17.

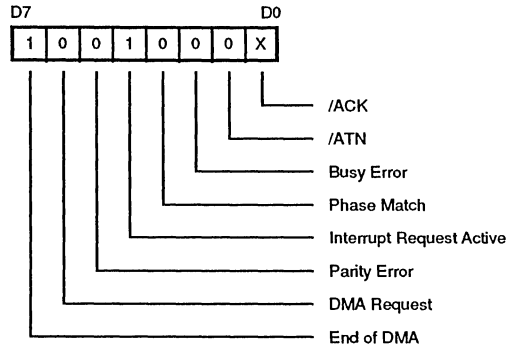


Figure 16. Bus and Status Register

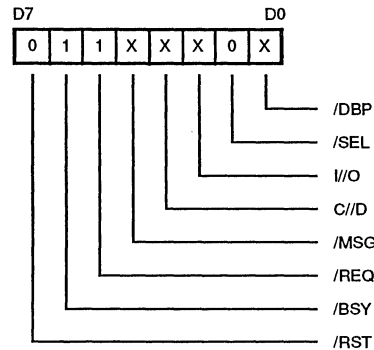


Figure 17. Current SCSI Bus Status Register

The End of DMA bit is used to determine when a block transfer is complete. Receive operations are complete when there is no data left in the chip and no additional handshakes occurring. The only exception to this is receiving data as an Initiator and the Target opts to send additional data for the same phase. In this /REQ goes active and the new data is present in the Input Data Register. Since a phase-mismatch interrupt will not occur, /REQ and /ACK need to be sampled to determine that the Target is attempting to send more data.

## FUNCTIONAL DESCRIPTION (Continued)

For send operations, the End of DMA bit is set when the DMA finishes its transfers, but the SCSI transfer may still be in progress. If connected as a Target, /REQ and /ACK should be sampled until both are False. If connected as an Initiator, a phase change interrupt is used to signal the completion of the previous phase. It is possible for the Target to request additional data for the same phase. In this case, a phase change will not occur and both /REQ and /ACK are sampled to determine when the last byte was transferred.

### SCSI Bus Reset Interrupt

The Z5380 generates an interrupt when the /RST signal transitions to True. The device releases all bus signals within a bus-clear delay of this transition. This interrupt also occurs after setting the Assert /RST bit (Initiator Command Register, bit 7). This interrupt cannot be disabled. (Note: /RST is not latched in bit 7 of the Current SCSI Bus Status Register and is not active when this port is read. For this case, the Bus Reset interrupt is determined by default.)

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 18 and 19, respectively.

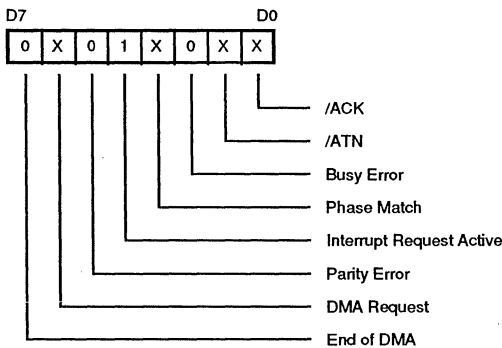


Figure 18. Bus and Status Register

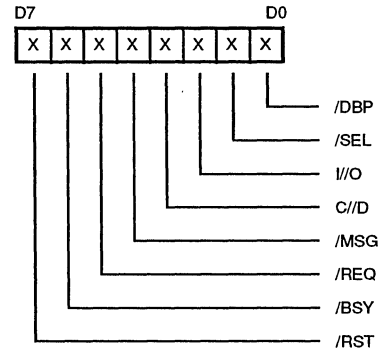


Figure 19. Current SCSI Bus Status Register

### Parity Error Interrupt

An Interrupt is generated for a received parity error if the Enable Parity Check (bit 5) and the Enable Parity Interrupt (bit 4) bits are set (1) in the Mode Register. Parity is checked during a read of the Current SCSI Data Register and during a DMA receive operation. A parity error can be detected without generating an interrupt by disabling the Enable Parity Interrupt bit and checking the Parity Error flag (Bus and Status Register, bit 5).

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 20 and 21, respectively.

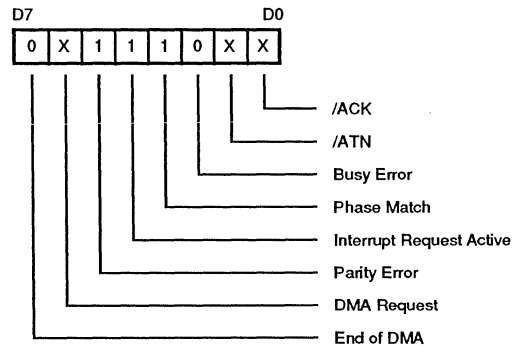


Figure 20. Bus and Status Register

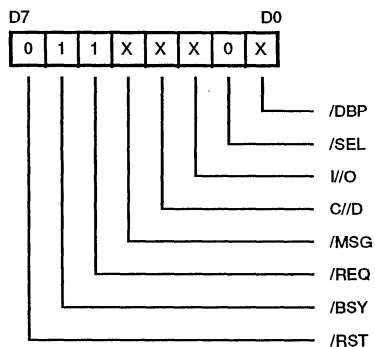


Figure 21. Current SCSI Bus Status Register

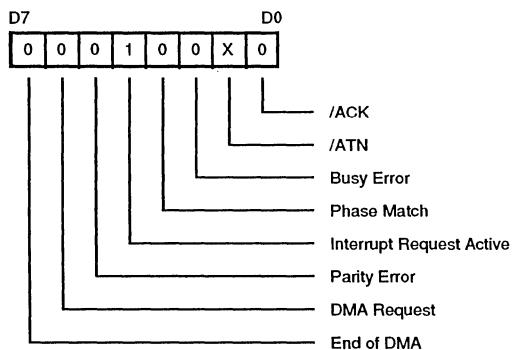


Figure 22. Bus and Status Register

### Bus Phase Mismatch Interrupt

The SCSI phase lines are comprised of the signals I//O, C//D, and /MSG. These signals are compared with the corresponding bits in the Target Command Register: Assert I//O (bit 0), Assert C//D (bit 1), and Assert /MSG (bit 2). The comparison occurs continually and is reflected in the Phase Match bit (bit 3) of the Bus and Status Register. If the DMA Mode bit (Mode Register, bit 1) is active and a phase mismatch occurs when /REQ transitions from False to True, an interrupt (IRQ) is generated.

A phase mismatch prevents the recognition of /REQ and removes the chip from the bus during an Initiator send operation (/DB7-/DB0 and /DBP will not be driven even through the Assert Data Bus bit (Initiator Command Register, bit 0) is active). This may be disabled by resetting the DMA Mode bit (Note: It is possible for this interrupt to occur when connected as a Target if another device is driving the phase lines to a different state).

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 22 and 23, respectively.

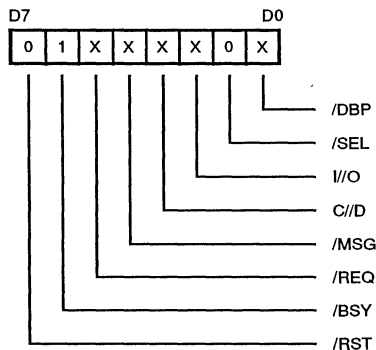


Figure 23. Current SCSI Bus Status Register

### Loss of BSY Interrupt

If the Monitor Busy bit (bit 2) in the Mode Register is active, an interrupt is generated if the BSY signal goes False for at least a bus-settle delay. This interrupt is disabled by resetting the Monitor Busy bit. Register values are displayed in Figures 24 and 25.

## FUNCTIONAL DESCRIPTION (Continued)

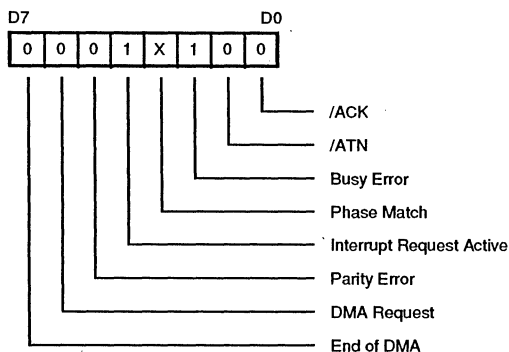


Figure 24. Bus and Status Register

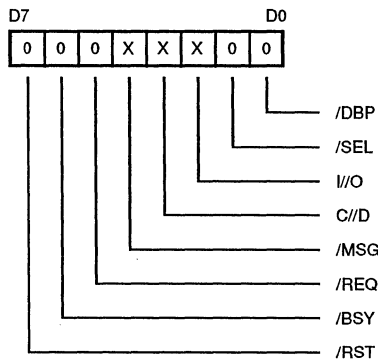


Figure 25. Current SCSI Bus Status Register

### Reset Conditions

Three possible reset situations exist with the Z5380, as follows:

#### Hardware Chip Reset

When the signal /RST is active for at least 200 ns, the Z5380 device is re-initialized and all internal logic and control registers are cleared. This is a chip reset only and does not create a SCSI Bus-Reset condition.

#### SCSI Bus Reset (/RST) Received

When a SCSI /RST signal is received, an IRQ interrupt is generated and a chip reset is performed. All internal logic and registers are cleared, except for the IRQ interrupt latch and the Assert /RST bit (bit 7) in the Initiator Command Register. (Note: The /RST signal may be sampled by

reading the Current SCSI Bus Status Register; however, this signal is not latched and may not be present when this port is read).

#### SCSI Bus Reset (/RST) Issued

If the CPU sets the Assert /RST bit (bit 7) in the Initiator Command Register, the /RST signal goes active on the SCSI Bus and an internal reset is performed. Again, all internal logic and registers are cleared except for the IRQ interrupt latch and the Assert /RST bit (bit 7) in the Initiator Command Register. The /RST signal will continue to be active until the Assert /RST bit is reset or until a hardware reset occurs.

#### Data Transfers

Data is transferred between SCSI Bus devices in one of four modes (Reference Figures 26-41):

1. Programmed I/O
2. Normal DMA
3. Block Mode DMA
4. Pseudo DMA

The following sections describe these modes in detail (**Note:** For all data transfer operations, /DACK and /CS should never be active simultaneously).

#### Programmed I/O Transfers

Programmed I/O is the most primitive form of data transfer. The /REQ and /ACK handshake signals are individually monitored and asserted by reading and writing the appropriate register bits. This type of transfer is normally used when transferring small blocks of data such as command blocks or message and status bytes. An Initiator send operation would begin by setting the C//D, I/O, and /MSG bits in the Target Command Register to the correct state so that a phase match exists. In addition to the phase match condition, it is necessary for the Assert Data Bus bit (Initiator Command Register, bit 0) to be True and the received I/O signal to be False for the Z5380 to send data. For each transfer, the data is loaded into the Output Data Register. The CPU then waits for the /REQ bit (Current SCSI Bus Status Register, bit 5) to become active. Once /REQ goes active, the Phase Match bit (Bus and Status Register, bit 3) is checked and the Assert /ACK bit (Initiator Command Register, bit 4) is set. The /REQ bit is sampled until it becomes False and the CPU resets the Assert /ACK bit to complete the transfer.

#### Normal DMA Mode

DMA transfers are normally used for large block transfers. The SCSI chip outputs a DMA request (DRQ) whenever it is ready for a byte transfer. External DMA logic uses this

---

DRQ signal to generate /DACK and an /IOR or an /IOW pulse to the Z5380. DRQ goes inactive when /DACK is asserted and /DACK goes inactive some time after the minimum read or write pulse width. This process is repeated for every byte. For this mode, /DACK should not be allowed to cycle unless a transfer is taking place.

#### **Block Mode DMA**

Some popular DMA Controllers, such as the 9517A, provide a Block Mode DMA transfer. This type of transfer allows the DMA controller to transfer blocks of data without relinquishing the use of the Data Bus to the CPU after each byte is transferred; thus, faster transfer rates are achieved by eliminating the repetitive access and release of the CPU Bus. If the Block Mode DMA bit (Mode Register, bit 7) is active, the Z5380 begins the transfer by asserting DRQ. The DMA controller then asserts /DACK for the remainder of the block transfer. DRQ goes inactive for the duration of the transfer. The Ready output is used to control the transfer rate. Non-Block Mode DMA transfers end when /DACK goes False, whereas Block Mode DMA transfers end when /IOR or /IOW becomes inactive. Since this is the case, DMA transfers may be started sooner in a Block Mode transfer. To obtain optimum performance in Block Mode operation, the DMA logic optionally uses the normal DMA mode interlocking handshake. Ready is still available to throttle the DMA transfer, but DRQ is 30 to 40 ns faster than Ready and is used to start the cycle sooner. The methods described under "Halting a DMA Operation" apply for all DMA operations.

#### **Pseudo DMA Mode**

To avoid the tedium of monitoring and asserting the request/acknowledgement handshake signals for programmed I/O transfers, the system can be designed to implement a pseudo DMA mode. This mode is implemented by programming the Z5380 to operate in the DMA mode, but using the CPU to emulate the DMA handshake. DRQ may be detected by polling the DMA Request bit (bit 6) in the Bus and Status Register, by sampling the signal through an external port, or by using it to generate a CPU interrupt. Once DRQ is detected, the CPU can perform a read or write data transfer. This CPU read/write is externally decoded to generate the appropriate /DACK and /IOR or /IOW signals.

Often, external decoding logic is necessary to generate the Z5380 /CS signal. This same logic may be used to generate /DACK at no extra cost and provide an increased performance in programmed I/O transfers.

#### **Halting a DMA Operation**

The /EOP signal is not the only way to halt a DMA transfer. A bus phase mismatch or a reset of the DMA Mode bit (Mode Register, bit 1) can also terminate a DMA cycle for the current bus phase.

#### **Using the /EOP Signal**

If /EOP is used, it should be asserted for at least 100ns while /DACK and /IOR or /IOW are simultaneously active. Note, however, that if /IOR or /IOW is not active, an interrupt is generated, but the DMA activity continues. The /EOP signal does not reset the DMA Mode bit. Since the /EOP signal can occur during the last byte sent to the Output Data Register, the /REQ and /ACK signals are monitored to ensure that the last byte has transferred.

#### **Bus Phase Mismatch Interrupt**

A bus phase mismatch interrupt is used to halt the transfer if operating as an Initiator. Using this method frees the host from maintaining a data length counter and frees the DMA logic from providing the /EOP signal. If performing an Initiator send operation, the Z5380 requires /DACK to cycle before /ACK goes inactive. Since phase changes cannot occur if /ACK is active, either /DACK must be cycled after the last byte is sent or the DMA Mode bit must be reset in order to receive the phase mismatch interrupt.

#### **Resetting the DMA Mode Bit**

A DMA operation may be halted at any time simply by resetting the DMA Mode bit. It is recommended that the DMA Mode bit be reset after receiving an /EOP or bus phase-mismatch interrupt. The DMA Mode bit must then be set before writing any of the start DMA registers for subsequent bus phases.

If resetting the DMA Mode bit is used instead of /EOP for Target role operation, then care must be taken to reset this bit at the proper time. If receiving data as a Target device, the DMA Mode bit must be reset once the last DRQ is received and before /DACK is asserted to prevent an additional /REQ from occurring. Resetting this bit causes DRQ to go inactive. However, the last byte received remains in the Input Data Register and may be obtained either by performing a normal CPU read or by cycling /DACK and /IOR. In most cases, /EOP is easier to use when operating as a Target device.



# READ REGISTERS

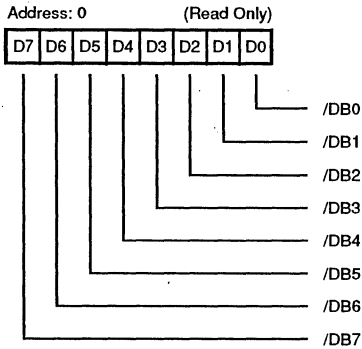


Figure 26. Current SCSI Data Register

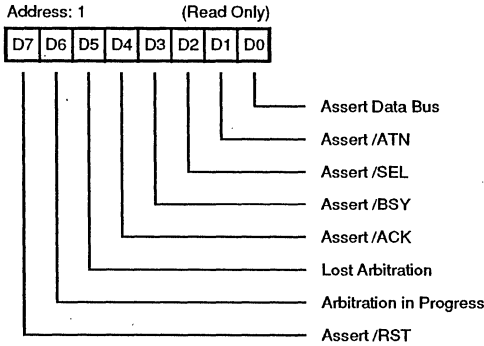


Figure 27. Initiator Command Register

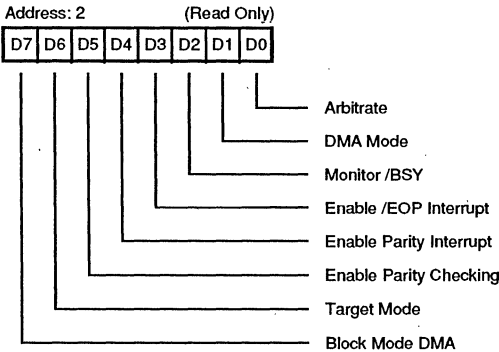


Figure 28. Mode Register

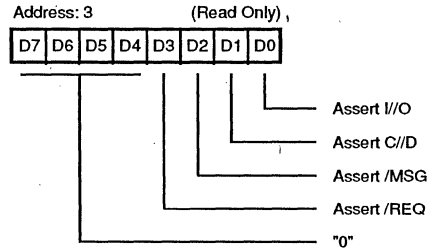


Figure 29. Target Command Register

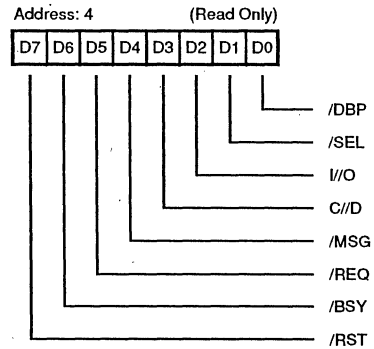


Figure 30. Current SCSI Bus Status Register

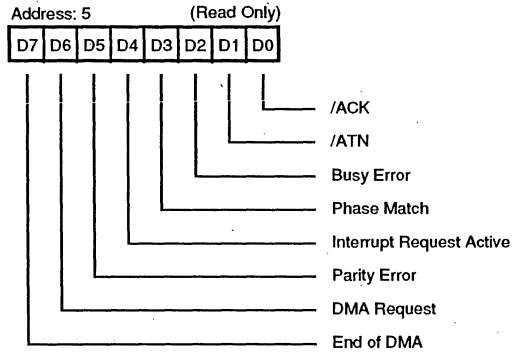


Figure 31. Bus and Status Register

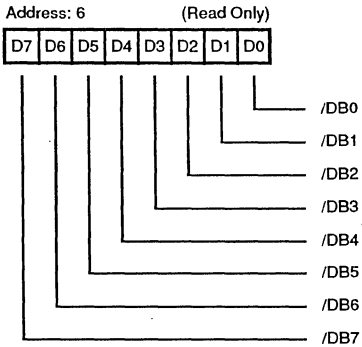


Figure 32. Input Data Register

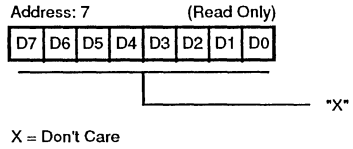


Figure 33. Reset Parity/Interrupt

WRITE REGISTERS

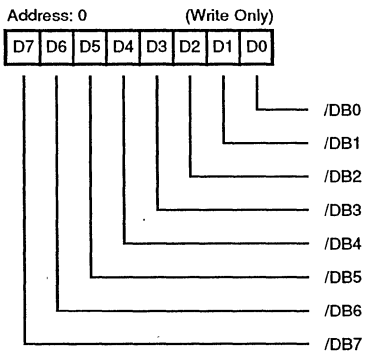


Figure 34. Output Data Register

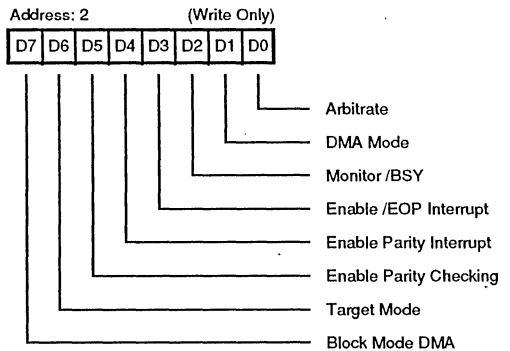


Figure 36. Mode Register

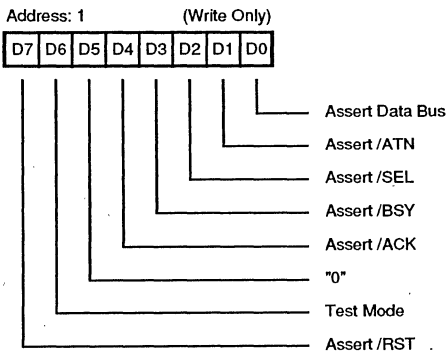


Figure 35. Initiator Command Register

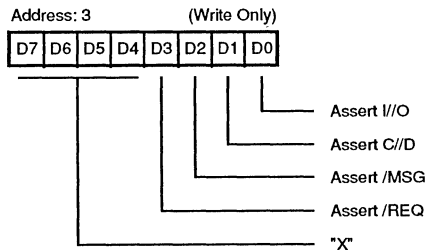


Figure 37. Target Command Register

## WRITE REGISTERS (Continued)

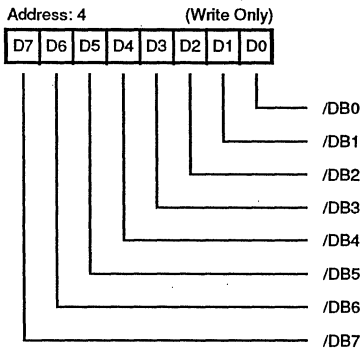


Figure 38. Select Enable Register

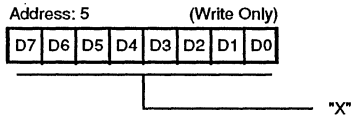


Figure 39. Start DMA Send

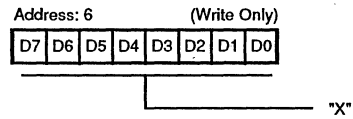


Figure 40. Start DMA Target Receive

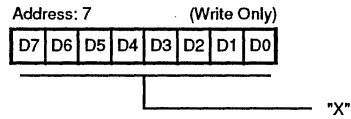


Figure 41. Start DMA Initiator Receive

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins with respect to GND .....	-0.3V to +7.0V
Operating Ambient Temperature .....	†
Storage Temperature .....	-65°C to +150°C

**Note:**

† See Ordering Information

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows (Figures 42 and 43):

- $+4.75V < V_{CC} < +5.25V$
- $GND = 0V$
- $T_A$  as specified in Ordering Information

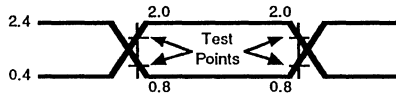


Figure 42. Switching Test Circuit

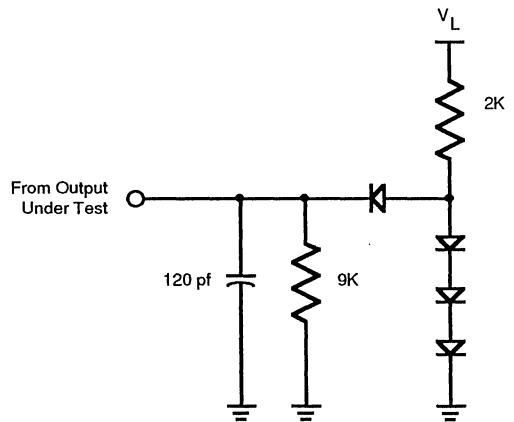


Figure 43. Standard Test Load

## DC CHARACTERISTICS

Z5380

Symbol	Parameter	Conditions	Min	Max	Units
$V_{DD}$	Supply Voltage		4.75	5.25	V
$V_{IH}$	High-Level Input Voltage		2.0	5.25	V
$V_{IL}$	Low-Level Input Voltage		-0.3	0.8	V
$I_{IH1}$	High-Level Input Current SCSI Bus Pins	$V_{IH} = 5.25V$ $V_{IL} = 0V$		50	$\mu A$
$I_{IH2}$	High-Level Input Current All Other Pins	$V_{IH} = 5.25V$ $V_{IL} = 0V$		10	$\mu A$
$I_{IL1}$	Low-Level Input Current SCSI Bus Pins	$V_{IH} = 5.25V$ $V_{IL} = 0V$	-50		$\mu A$
$I_{IL2}$	Low-Level Input Current All Other Pins	$V_{IH} = 5.25V$ $V_{IL} = 0V$	-10		$\mu A$
$V_{OH}$	High-Level Output Voltage	$I_{OH} = -3mA$ $V_{DD} = 4.75V$	2.4		V
$V_{OL1}$	Low-Level Output Voltage SCSI Bus Pins	$I_{OL} = 48mA$ $V_{DD} = 4.75V$	0.5		V
$V_{OL2}$	Low-Level Output Voltage All Other Pins	$I_{OL} = 7mA$ $V_{DD} = 4.75V$	0.5		V
$I_{DD}$	Supply Current	15 mA			
$T_A$	Operating Free-Air		0	70	C

## AC CHARACTERISTICS

### CPU Write Cycle Timing Diagram

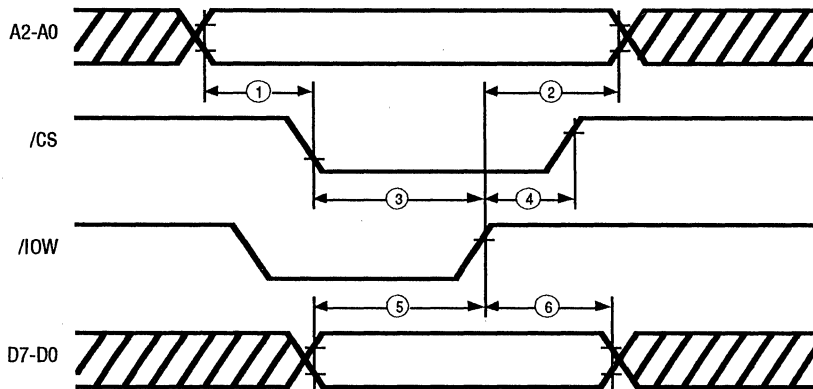


Figure 44. CPU Write Cycle

## AC CHARACTERISTICS

### CPU Write Cycle Timing Table

No	Description	Min	Max	Units
1	Address Setup to Write Enable[1]	20		ns
2	Address Hold from End Write Enable[1]	20		ns
3	Write Enable Width[1]	70		ns
4	Chip Select Hold from End of /IOW	0		ns
5	Data Setup to end of Write Enable[1]	50		ns
6	Data Hold Time from End of /IOW	30		ns

**Note:**

[1] Write Enable is the occurrence of /IOW and /CS

## AC CHARACTERISTICS

### CPU Read Cycle Timing Diagram

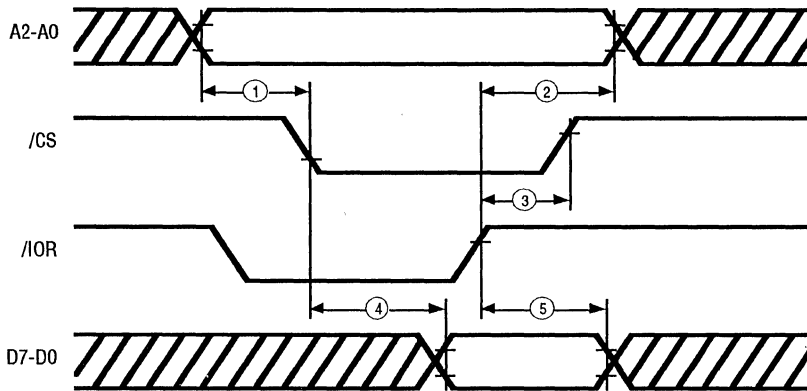


Figure 45. CPU Read Cycle

## AC CHARACTERISTICS

### CPU Read Cycle Timing Table

No	Description	Min	Max	Units
1	Address Setup to Read Enable[1]	20		ns
2	Address Hold from End Read Enable[1]	20		ns
3	Chip Select Hold from End of /IOR	0		ns
4	Data Access Time from Read Enable[1]	130		ns
5	Data Hold Time from End of Read Enable[1]	20		ns

**Note:**

[1] Read Enable is the occurrence of /IOR and /CS.

**AC CHARACTERISTICS**

DMA Write (Non-Block Mode) Target Send Cycle Timing Diagram

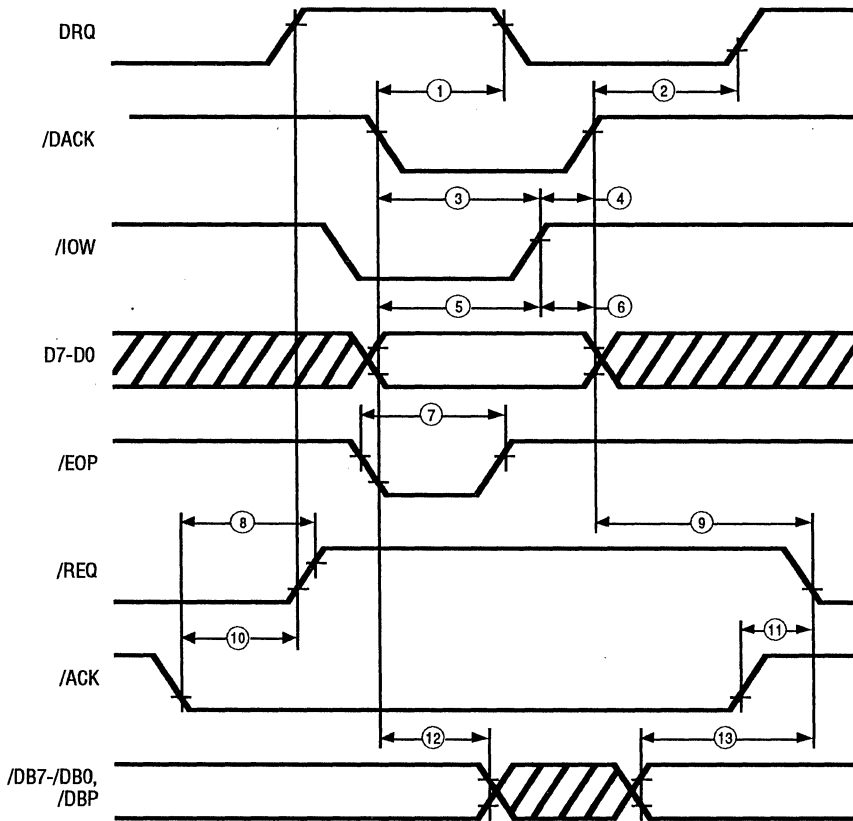


Figure 46. DMA Write (Non-Block Mode) Target Send Cycle

---

## AC CHARACTERISTICS

### DMA Write (Non-Block Mode) Target Send Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low	130		ns
2	/DACK High to DRQ High	30		ns
3	Write Enable Width[1]	100		ns
4	/DACK Hold from /IOW High	0		ns
5	Data Setup to End of Write Enable[1]	50		ns
6	Data Hold Time from End of /IOW	40		ns
7	Width of /EOP Pulse[2]	100		ns
8	/ACK Low to /REQ High	25	125	ns
9	/REQ from End of /DACK (/ACK High)	30	150	ns
10	/ACK Low to DRQ High (Target)	15	110	ns
11	/ACK High to /REQ Low (/DACK High)	20	150	ns
12	Data Hold from Write Enable	15		ns
13	Data Setup to /REQ Low (Target)	60		ns

**Notes:**

[1] Write Enable is the occurrence of /IOW and /DACK.

[2] /EOP, /IOW, and /DACK must be concurrently Low for at least T7 for proper recognition of the /EOP pulse.



## AC CHARACTERISTICS

### DMA Write (Non-Block Mode) Initiator Send Cycle Timing Diagram

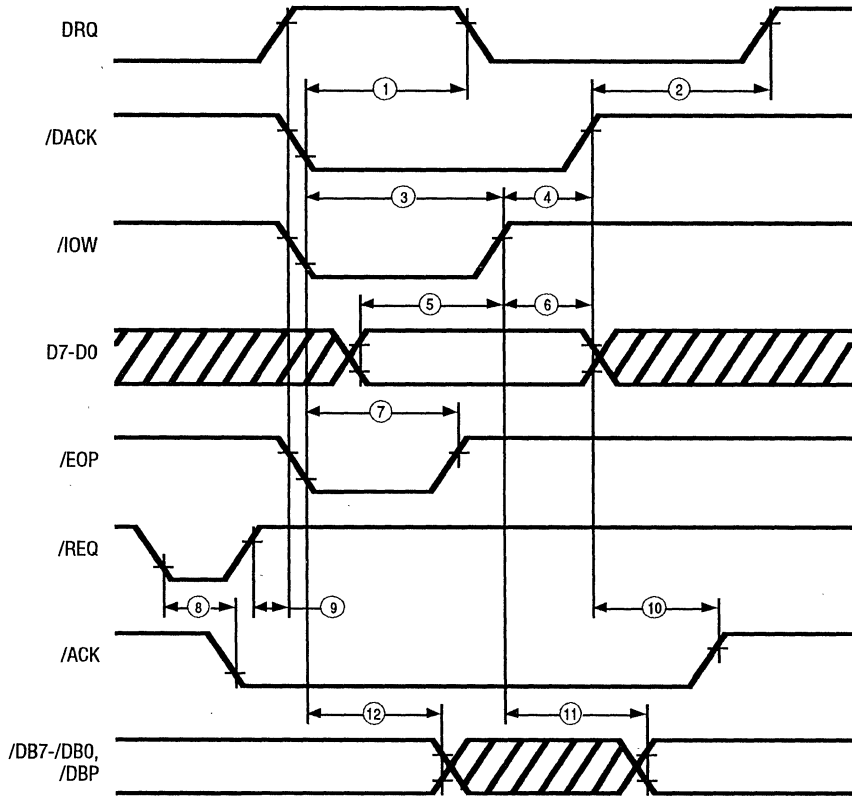


Figure 47. DMA Write (Non-Block Mode) Initiator Send Cycle

---

## AC CHARACTERISTICS

### DMA Write (Non-Block Mode) Initiator Send Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low	130		ns
2	/DACK High to DRQ High	30		ns
3	Write Enable Width[1]	100		ns
4	/DACK Hold from End of /IOW	0		ns
5	Data Setup to End of Write Enable[1]	50		ns
6	Data Hold Time from End of /IOW	40		ns
7	Width of /EOP Pulse[2]	100		ns
8	/REQ Low to /ACK Low	20	160	ns
9	/REQ High to DRQ High	20	110	ns
10	/DACK High to /ACK High	25	150	ns
11	/IOW High to Valid SCSI Data	100		ns
12	Data Hold from Write Enable[1]	15		ns

**Notes:**

[1] Write Enable is the occurrence of /IOW and /DACK.

[2] /EOP, /IOW, and /DACK must be concurrently Low for at least T7 for proper recognition of the /EOP pulse.

# AC CHARACTERISTICS

## DMA Read (Non-Block Mode) Target Receive Cycle Timing Diagram

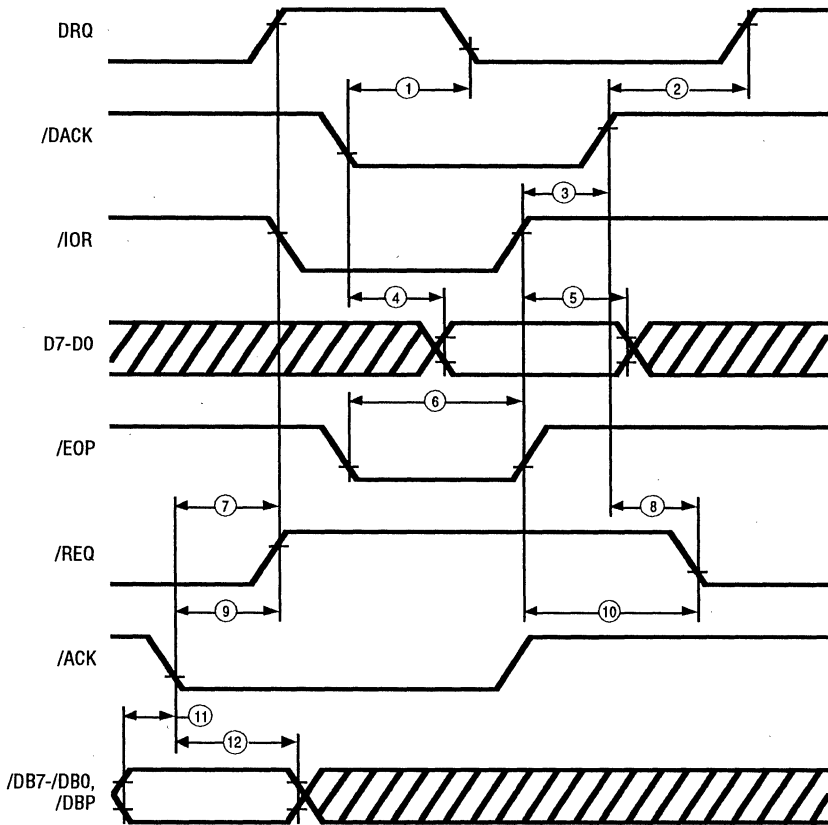


Figure 48. DMA Read (Non-Block Mode) Target Receive Cycle

---

## AC CHARACTERISTICS

### DMA Read (Non-Block Mode) Target Receive Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low	130		ns
2	/DACK High to DRQ High	30		ns
3	/DACK Hold Time from End of /IOR	0		ns
4	Data Access Time from Read Enable[1]	115		ns
5	Data Hold Time from End of /IOR	20		ns
6	Width of /EOP Pulse[2]	100		ns
7	/ACK Low to DRQ High	15	110	ns
8	/DACK High to /REQ Low (/ACK High)	30	150	ns
9	/ACK Low to /REQ High	25	125	ns
10	/ACK High to /REQ Low (/DACK High)	20	150	ns
11	Data Setup Time to /ACK	20		ns
12	Data Hold Time from /ACK	50		ns

**Notes:**

[1] Read Enable is the occurrence of /IOR and /DACK.

[2] /EOP, /IOR, and /DACK must be concurrently Low for at least T6 for proper recognition of the /EOP pulse.

## AC CHARACTERISTICS

### DMA Read (Non-Block Mode) Initiator Receive Cycle Timing Diagram

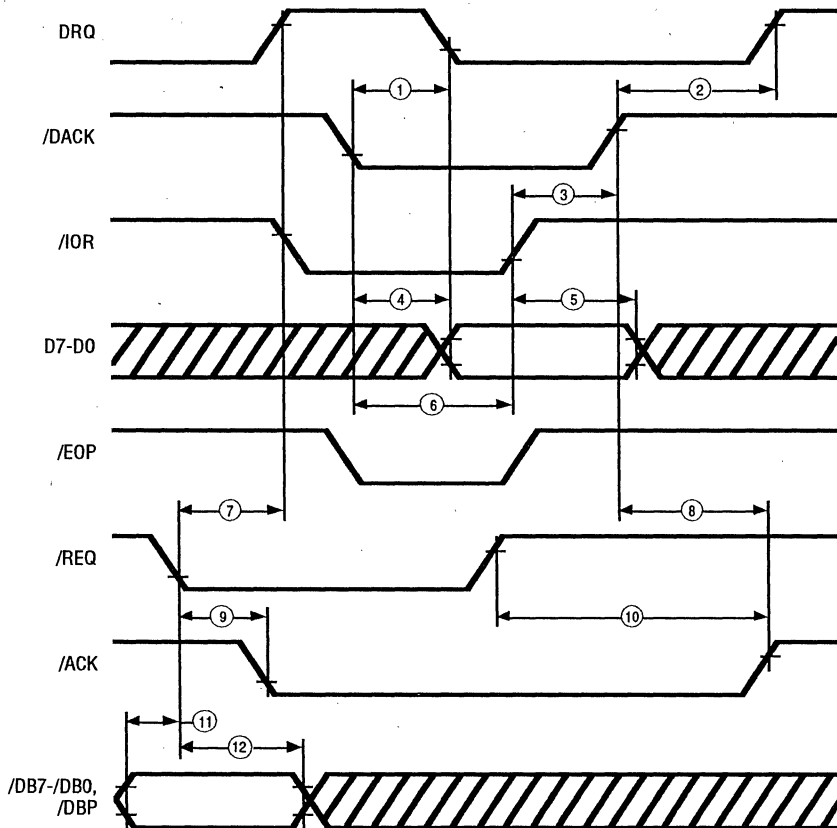


Figure 49. DMA Read (Non-Block Mode) Initiator Receive Cycle

---

## AC CHARACTERISTICS

### DMA Read (Non-Block Mode) Initiator Receive Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low	130		ns
2	/DACK High to DRQ High	30		ns
3	/DACK Hold Time from End of /IOR	0		ns
4	Data Access Time from Read Enable[1]	115		ns
5	Data Hold Time from End of /IOR	20		ns
6	Width of /EOP Pulse[2]	100		ns
7	/REQ Low to DRQ High	20		ns
8	/DACK High to /ACK High (/REQ High)	25	160	ns
9	/REQ Low to /ACK Low	20	160	ns
10	/REQ High to /ACK High (/DACK High)	15	140	ns
11	Data Setup Time to /REQ	20		ns
12	Data Hold Time from /REQ	50		ns

**Notes:**

[1] Read Enable is the occurrence of /IOR and /DACK.

[2] /EOP, /IOR, and /DACK must be concurrently Low for at least T6 for proper recognition of the /EOP pulse.

# AC CHARACTERISTICS

## DMA Write (Block Mode) Target Send Cycle Timing Diagram

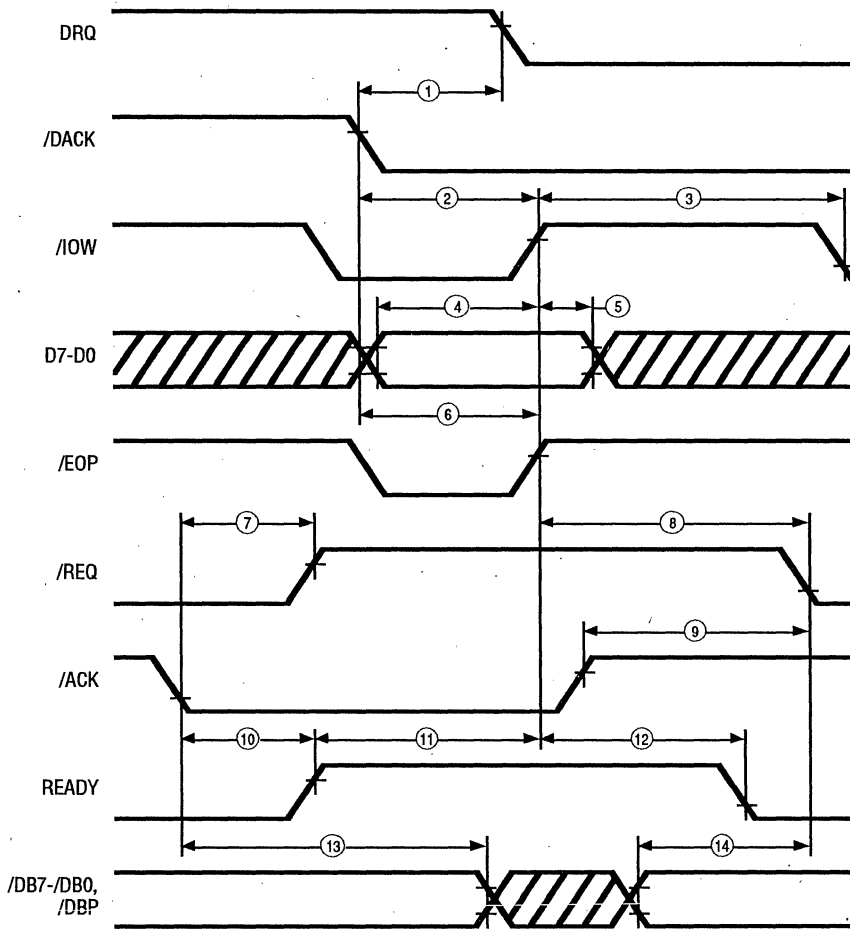


Figure 50. DMA Write (Block Mode) Target Send Cycle

---

## AC CHARACTERISTICS

### DMA Write (Block Mode) Target Send Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low	130		ns
2	Write Enable Width[1]	100		ns
3	Write Recovery Time	120		ns
4	Data Setup to End of Write Enable[1]	50		ns
5	Data Hold Time from End of /IOW	40		ns
6	Width of /EOP Pulse[2]	100		ns
7	/ACK Low to /REQ High	25	125	ns
8	/REQ from End of /IOW (/ACK High)	40	180	ns
9	/REQ from End of /ACK (/IOW High)	20	170	ns
10	/ACK Low to READY High	20	140	ns
11	READY High to /IOW High	70		ns
12	/IOW High to READY Low	20	140	ns
13	Data Hold from /ACK Low	40		ns
14	Data Setup to /REQ Low	60		ns

**Notes:**

[1] Write Enable is the occurrence of /IOW and /DACK.

[2] /EOP, /IOW, and /DACK must be concurrently Low for at least T6 for proper recognition of the /EOP pulse.



# AC CHARACTERISTICS

## DMA Read (Block Mode) Target Receive Cycle Timing Diagram

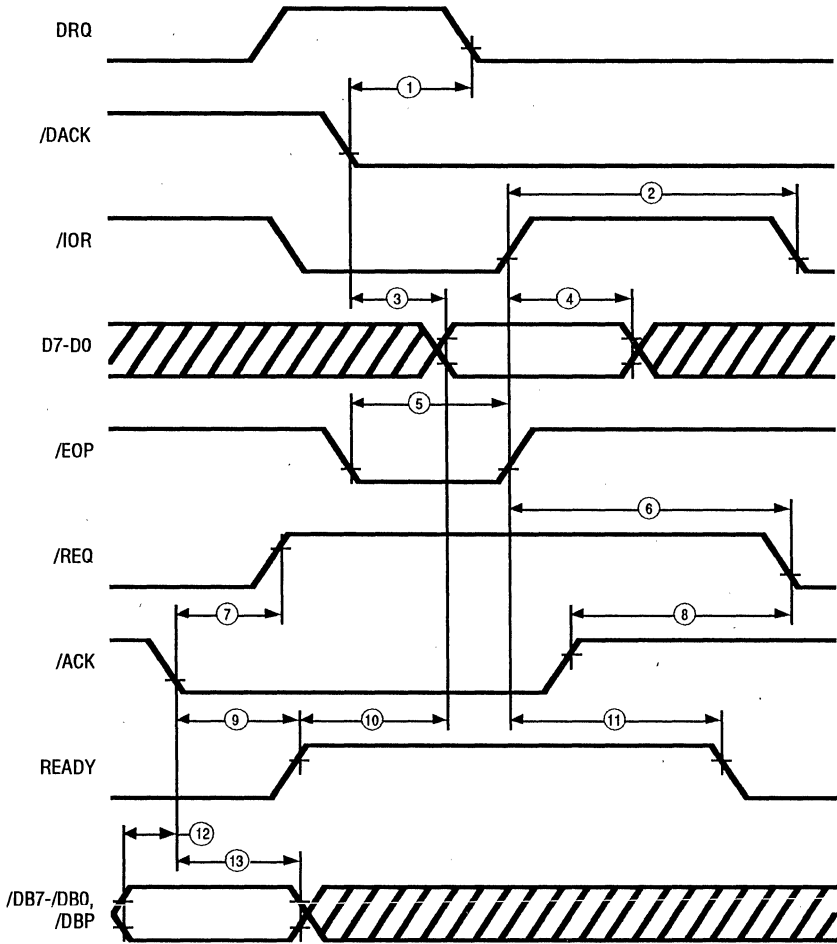


Figure 51. DMA Read (Block Mode) Target Receive Cycle

---

## AC CHARACTERISTICS

### DMA Read (Block Mode) Target Receive Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low	130		ns
2	/IOR Recovery Time	120		ns
3	Data Access Time from Read Enable[1]	110		ns
4	Data Hold Time from End of /IOR	20		ns
5	Width of /EOP Pulse[2]	100		ns
6	/IOR High to /REQ Low	30	190	ns
7	/ACK Low to /REQ High	25	125	ns
8	/ACK High to /REQ Low (/IOR High)	20	170	ns
9	/ACK Low to READY High	20	140	ns
10	READY High to Valid Data	50		ns
11	/IOR High to READY Low	20	140	ns
12	Data Setup Time to /ACK	20		ns
13	Data Hold Time from /ACK	50		ns

**Notes:**

[1] Read Enable is the occurrence of /IOR and /DACK.

[2] /EOP, /IOR, and /DACK must be concurrently Low for at least T5 for proper recognition of the /EOP pulse.

## AC CHARACTERISTICS

### Arbitration

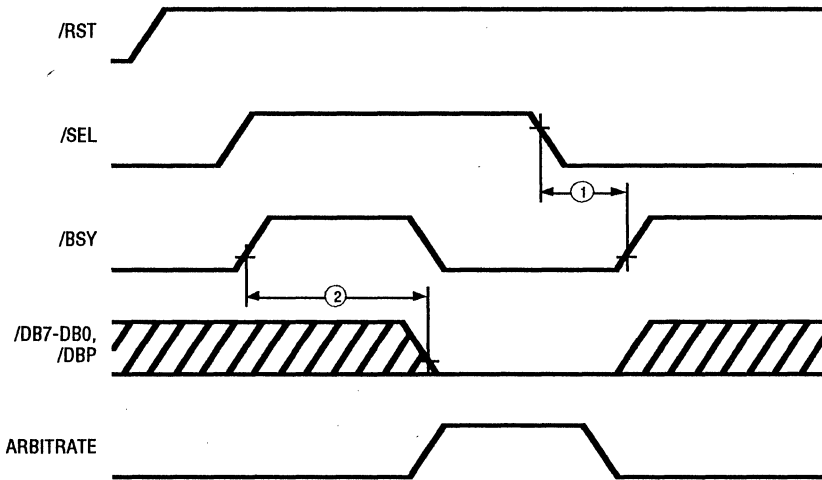


Figure 52. Arbitration

No	Description	Min	Max	Units
1	Bus Clear from /SEL Low		600	ns
2	Arbitrate Start from /BSY High	1200	2200	ns

## AC CHARACTERISTICS

### Reset

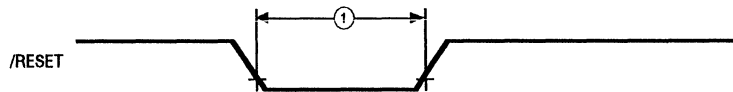


Figure 53. Reset

No	Description	Min	Max	Units
1	Minimum Width of /RESET	200		ns

---

## Z5380 NOTES

1. Edge-triggered /RST Interrupt - If the SCSI Bus is not terminated, the /RST interrupt is continually generated.

2. True End of DMA Interrupt - The Z5380 generates an interrupt when it receives the last byte from the DMA, not when the last byte is transferred to the SCSI Bus.

3. Return to Ready after /EOP Interrupt - When operating in Block Mode DMA, the Z5380 does not return the Ready signal to a Ready condition. This locks up the bus and prevents the CPU from executing.

4. SCSI handshake after /EOP occurs - If an EOP occurs when receiving data, a subsequent request will cause /ACK to be asserted even though no DRQ is issued.

5. Reselection Interrupt - During reselection, if the Target Command Register does not reflect the current bus phase (most likely Data Out), the reselection interrupt may get reset.

6. Phase Mismatch Interrupt - A phase mismatch interrupt is not guaranteed after a reselection for the following reasons:

DMA Mode bit must be set in order to receive a phase mismatch interrupt.

DMA Mode bit can not be set unless /BSY is active.

/BSY can not be asserted until after the reselection has occurred.

Once /BSY is asserted, the Target may assert /REQ in less than 500ns.

The phase mismatch interrupt is generated on the active edge of /REQ. If the DMA Mode bit is not set before the /REQ goes active, the phase mismatch interrupt will not occur.





# Z53C80

## SMALL COMPUTER SYSTEM INTERFACE (SCSI)

### FEATURES

- DMA or programmed I/O data transfers
- Arbitration support
- Supports normal or block mode DMA
- Memory or I/O mapped CPU interface
- Asynchronous interface, supports 3 Mbytes/sec
- Direct SCSI bus interface with on-board 48 mA drivers
- Supports target and initiator roles
- Meets SCSI protocol as defined in ANSI X3.131-1986 standard
- Added "Glitch Eater" enhancement to minimize bus reflection

### GENERAL DESCRIPTION

The Z53C80 SCSI (Small Computer System Interface) controller is a 44-pin PLCC, or 48-pin DIP CMOS device. It is designed to implement the SCSI protocol as defined by the ANSI X3.131-1986 standard, and is fully compatible with the industry standard 5380. It is capable of operating both as a target and as an initiator. Special high-current open-drain outputs enable it to directly interface to the SCSI bus. The Z53C80 has the necessary interface hook-ups so the system CPU can communicate with it as with any other peripheral device. The CPU can read from, or write to, the SCSI registers which are addressed as standard or memory-mapped I/Os.

The Z53C80 increases the system performance by minimizing the CPU intervention in DMA operations which the SCSI controls. The CPU is interrupted by the SCSI when it detects a bus condition that requires attention. It also supports arbitration and reselection. The Z53C80 has the proper handshake signals to support normal and block mode DMA operations with most DMA controllers available.

The added enhancement known as the "Glitch Eater" is used to minimize effects of bus reflection on improperly terminated SCSI bus applications. The high frequency

reflections that can occur on the SCSI bus are filtered out, reducing the sensitivity of the inputs, specifically /REQ and /ACK to bus signal reflections. Figure 1 shows a worst case input waveform (labeled A), along with the filtered input (labeled B) and the output of a Schmitt trigger used to provide the hysteresis required on SCSI inputs (labeled C). This enhancement is a requirement for the device to function properly in a Apple Macintosh® environment.

#### Notes:

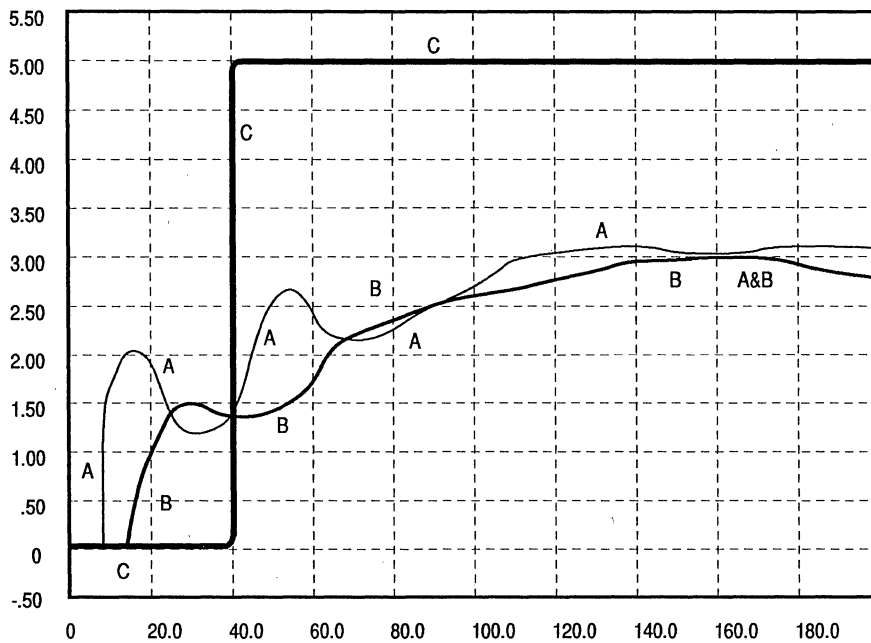
All Signals with a preceding front slash, '/', are active Low, e.g.: B/W (WORD is active Low); /B/W (BYTE is active Low, only).

Power connections follow conventional descriptions below:

Connection	Circuit	Device
Power	V <sub>CC</sub>	V <sub>DD</sub>
Ground	GND	V <sub>SS</sub>

® Apple Macintosh is a registered trademark of Apple Computer, Inc.

**GENERAL DESCRIPTION** (Continued)



**Figure 1. Worst Case Unfiltered Input (A), Filtered Input (B), Output of Schmitt Trigger Used to Provide Hysteresis (C).**

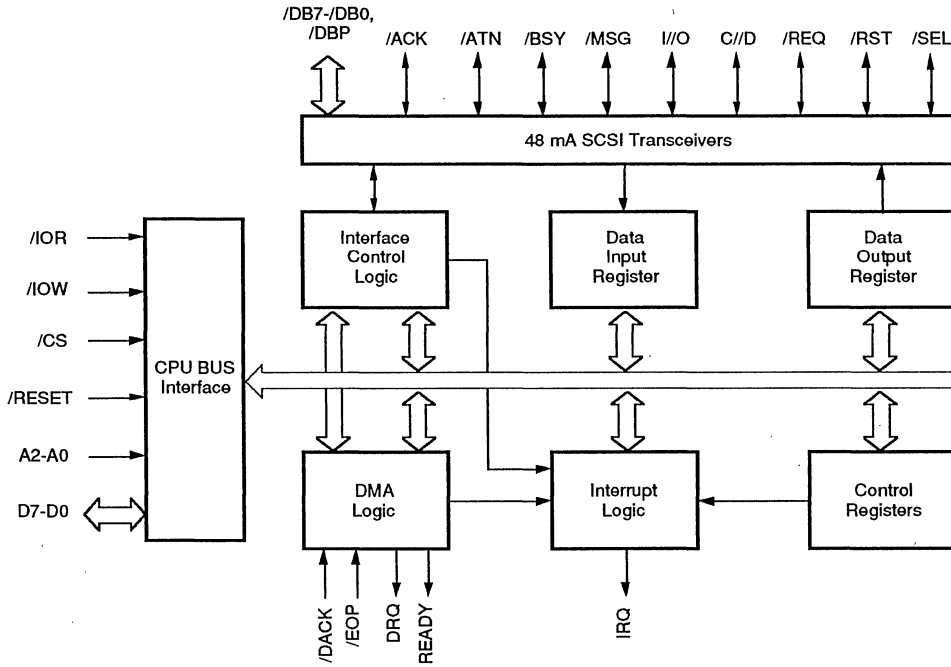


Figure 2a. SCSI Block Diagram

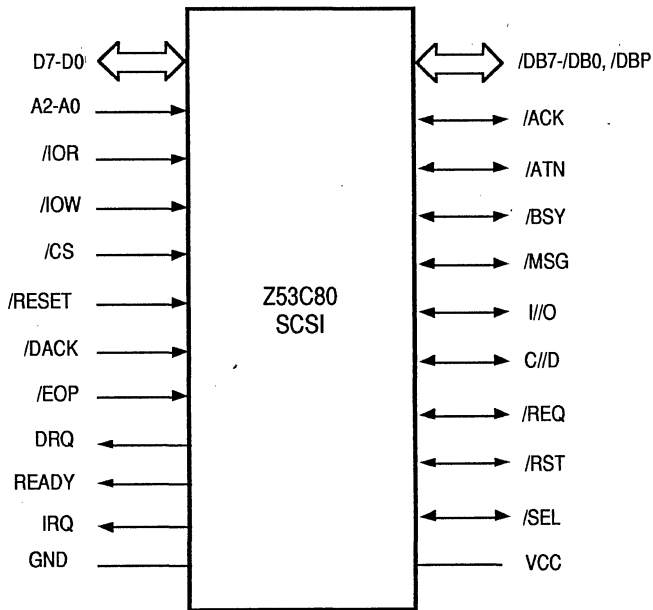
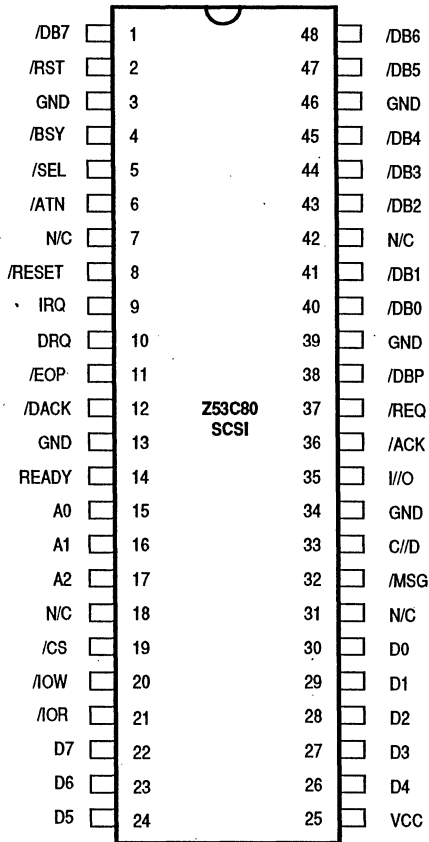


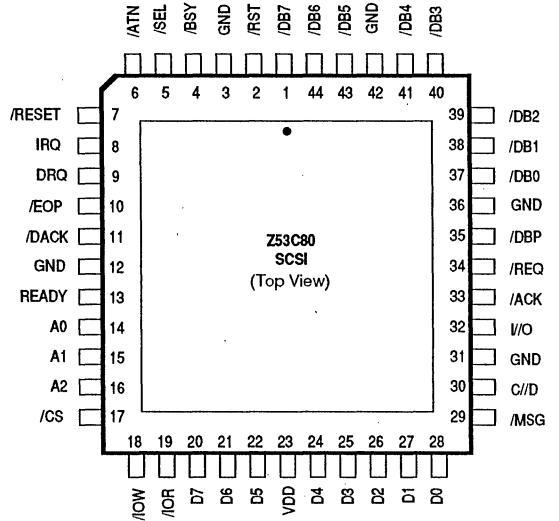
Figure 2b. SCSI Pin Functions



**GENERAL DESCRIPTION** (Continued)



**Figure 3. 48-Pin DIP Pin Assignments**



**Figure 4. 44-Pin PLCC Pin Assignments**

---

## PIN DESCRIPTION

### Microprocessor Bus

**A2-A0. Address Lines (Input).** Address lines are used to access all internal registers with /CS, /IOR, and /IOW.

**/CS. /Chip Select (Input, active Low).** /CS, in conjunction with /RD or /WR, enables the internal register selected by A2-A0, to be read from or write to. /CS and /DACK must never be active simultaneously.

**/DACK. /DMA Acknowledge (Input, active Low).** /DACK, in conjunction with /IOR and /IOW, is used to enable reading or writing the SCSI I/O Data Registers when in the DMA Mode. When the DRQ has acknowledged that the byte has been successfully transferred to or from the DMA controller, this signal is asserted. /DACK and /CS must never be active simultaneously.

**DRQ. DMA Request (Output, active High).** This signal is asserted when the chip is ready to transfer a data byte to and from the DMA controller. The DMA Request will be asserted only if the DMA Mode bit (Register 2, Bit 1) is set. The transfer is complete upon reception of /DACK.

**D7-D0. Data Lines (Bidirectional; Tri-State, active High).** The Data Bus lines carry data and commands to and from SCSI. D7 is the most significant bit of this bus.

**/EOP. /End of Process (Input, active Low).** To terminate a DMA transfer, this signal is asserted. The current byte will be transferred but no additional bytes will be requested if asserted during a DMA cycle. /EOP can be used to generate an interrupt when it is received from a DMA Controller.

**/IOR. /I/O Read (Input, active Low).** This signal is used to read an internal register selected by /CS and A2-A0. The Input Data Register can also be selected by this signal when /DACK is active during DMA transfers.

**/IOW. /I/O Write (Input, active Low).** This signal is used to write to an internal register selected by /CS and A2-A0. The Output Data Register can also be selected by this signal when used with /DACK during DMA transfers.

**IRQ. Interrupt Request (Output, active High).** IRQ alerts the microprocessor of an error condition or an event completion. Most of the interrupts are individually maskable.

**READY. Ready (Output, active High).** This signal can be used to control the data transfer handshaking of block mode DMA transfers. READY is asserted to indicate that the chip is ready to transfer data and remains false after a transfer until the chip is ready for another DMA transfer. READY is always asserted when the DMA Mode Bit is a zero.

**/RESET. /Reset (Input, active Low).** /RESET clears all registers and has no effect upon the SCSI /RST signal. Therefore it does not reset the SCSI bus.

---

#### Power Signals

---

V <sub>CC</sub>	+5 Volt Power Supply
GND	Ground
N/C	No Connect

---

---

## SCSI BUS

The following signals are all bidirectional, active Low, open-drain, with 48 mA sink capacity. All pins interface directly with the SCSI Bus.

**/ACK.** */Acknowledge* (Bidirectional, Open Drain, Active Low). /ACK is driven by the Initiator and indicates an acknowledgement for a SCSI data transfer. /ACK is received as a response to the /REQ Signal in the Target role.

**/ATN.** */Attention* (Bidirectional, Open Drain, Active Low). /ATN is driven by the Initiator and indicates an attention condition. /ATN is received and is responded to by entering the Message Out Phase in the Target role.

**/BSY.** */Busy* (Bidirectional, Open Drain, Active Low). /BSY indicates that the SCSI Bus is being occupied. /BSY can be driven by both the Target and the Initiator device.

**/C//D.** */Control//Data* (Bidirectional, Open Drain, Active Low). /C//D indicates Control or Data information is on the SCSI Bus. This signal is driven by a Target and is received by the Initiator.

**/I/O.** */Input//Output* (Bidirectional, Open Drain, Active Low). /I/O is driven by a Target and controls the direction of data transfer on the SCSI Bus. When asserted, this signal indicates input to the Initiator. When not asserted, this signal indicates output from the Initiator. This signal is also used to recognize the difference between the Selection and Reselection Phases.

**/MSG.** */Message* (Bidirectional, Open Drain, Active Low). The Target drives /MSG active during the Message Phase and is received by the Initiator.

**/REQ.** */Request* (Bidirectional, Open Drain, Active Low). Received by the Initiator and driven by a Target, /REQ indicates a request for an SCSI data-transfer handshake.

**/RST.** *SCSI Bus RESET* (Bidirectional, Open Drain, Active Low). The /RST signal shows a SCSI Bus RESET condition has occurred.

**/DB7-/DB0,/DBP.** */Data Bits, /Parity Bits* (Bidirectional, Open Drain, Active Low). These eight data bits (/DB7-/DB0), plus a parity bit (/DBP) form the SCSI Data Bus. /DB7 has the highest priority during the Arbitration phase and is the most significant bit (MSB). Data parity is odd and is always generated and optionally checked, which is not valid during Arbitration.

**/SEL.** */Select* (Bidirectional, Open Drain, Active Low). /SEL is used by a Target to select an Initiator, or by an Initiator to reselect a Target.

## FUNCTIONAL DESCRIPTION

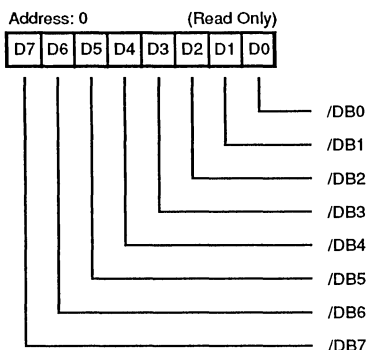
**General.** The Small Computer System interface (SCSI) device has a set of eight registers that are controlled by the CPU. By reading and writing the appropriate registers, the CPU may initiate any SCSI Bus activity or may sample and assert any signal on the SCSI Bus. This allows the user to implement all or any of the SCSI protocol in software. These registers are read (written) by activating /CS with an address on A2-A0 and then issuing a /RD (/WR) pulse. This section describes the operation of the internal registers (Table 1).

**Table 1. Register Summary**

Address			R/W	Register Name
A2	A1	A0		
0	0	0	R	Current SCSI Data
0	0	0	W	Output Data
0	0	1	R/W	Initiator Command
0	1	0	R/W	Mode
0	1	1	R/W	Target Command
1	0	0	R	Current SCSI Bus Status
1	0	0	W	Select Enable
1	0	1	R	Bus and Status
1	0	1	W	Start DMA Send
1	1	0	R	Input Data
1	1	0	W	Start DMA Target Receive
1	1	1	R	Reset Parity/Interrupt
1	1	1	W	Start DMA Initiator Receive

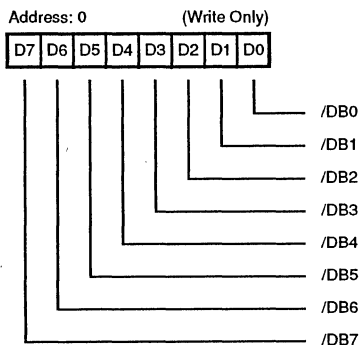
**Data Registers.** The data registers are used to transfer SCSI commands, data, status, and message bytes between the microprocessor Data Bus and the SCSI Bus. The SCSI does not interpret any information that passes through the data registers. The data registers consist of the transparent Current SCSI Data Register, the Output Data Register, and the Input Data Register.

**Current SCSI Data Register.** *Address 0 (Read Only).* The Current SCSI Data Register (Figure 5) is a read-only register which allows the microprocessor to read the active SCSI Data Bus. This is accomplished by activating /CS with an address on A2-A0 and issuing a /RD pulse. If parity checking is enabled, the SCSI Bus parity is checked at the beginning of the read cycle. This register is used during a programmed I/O data read or during Arbitration to check for higher priority arbitrating devices. Parity is not guaranteed valid during Arbitration.



**Figure 5. Current SCSI Data Register**

**Output Data Register.** *Address 0 (Write Only).* The Output Data Register (Figure 6) is a write-only register that is used to send data to the SCSI Bus. This is accomplished by either using a normal CPU write, or under DMA control, by using /WR and /DACK. This register also asserts the proper ID bits on the SCSI Bus during the Arbitration and Selection phases.



**Figure 6. Output Data Register**

## FUNCTIONAL DESCRIPTION (Continued)

**Input Data Register.** *Address 6 (Read Only).* The input Data Register (Figure 7) is a read-only register that is used to read latched data from the SCSI Bus. Data is latched either during a DMA Target receive operation when /ACK goes active or during a DMA Initiator receive when /REQ goes active. The DMA Mode bit (Mode Register bit 1) must be set before data can be latched in the Input Data Register. This register is read under DMA control using /RD and /DACK. Parity is optionally checked when the Input Data Register is loaded.

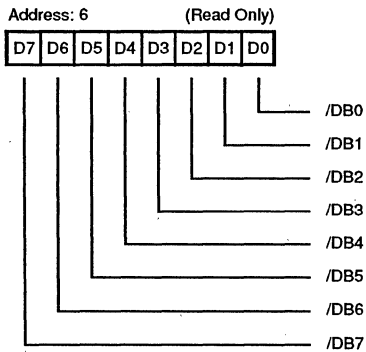


Figure 7. Input Data Register

**Initiator Command Register.** *Address 1 (Read/Write).* The Initiator Command Register (Figures 8 and 9) are read and write registers which assert certain SCSI Bus signals, monitors those signals, and monitors the progress of bus arbitration. Many of these bits are significant only when being used as an Initiator; however, most can be used during Target role operation.

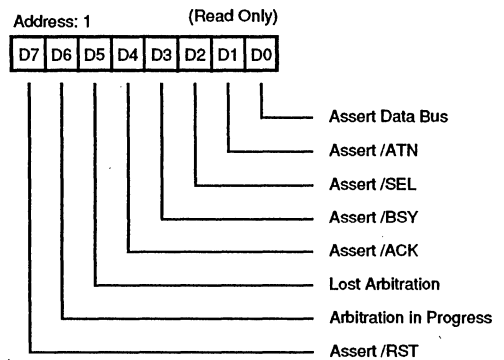


Figure 8. Initiator Command Register (Register Read)

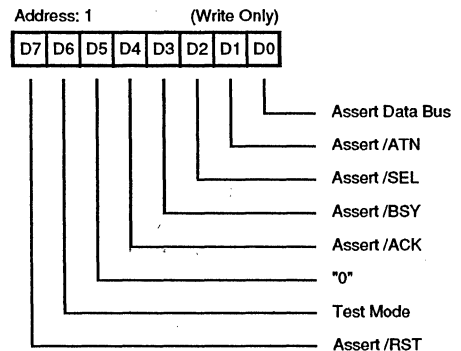


Figure 9. Initiator Command Register (Register Write)

The following describes the operation of all bits in the Initiator Command Register.

**Bit 0. Assert Data Bus.** The Assert Data Bus bit, when set, allows the contents of the Output Data Register to be enabled as chip outputs on the signals /DB7-/DB0. Parity is also generated and asserted on /DBP.

When connected as an Initiator, the outputs are only enabled if the Target Mode bit (Mode Register, bit 6) is False, the received signal I/O is False, and the phase signals C/D, I/O, and /MSG match the contents of the Assert C/O, Assert I/O and Assert /MSG in the Target Command Register.

This bit should also be set during DMA send operations.

**Bit 1. Assert/ATN/.** Bit 1 may be asserted on the SCSI Bus by setting this bit to a 1 if the Target Mode bit (Mode Register, bit 6) is False, /ATN is normally asserted by the initiator to request a Message Out bus phase. Note that since Assert/SEL and Assert/ATN are in the same register, a select with /ATN may be implemented with one CPU write /ATN may be deasserted by resetting this bit to zero. A read on this register simply reflects the status of this bit.

**Bit 2. Assert/SEL.** Writing a 1 into this bit position asserts /SEL onto the SCSI Bus. /SEL is normally asserted after Arbitration has been successfully completed /SEL may be disabled by resetting bit 2 to a 0. A read of this register reflects the status of this bit.

**Bit 3. Assert/BSY.** Writing a 1 into this bit position asserts /BSY onto the SCSI Bus. Conversely, a 0 resets the /BSY signal. Asserting /BSY indicates a successful selection or reselection. Resetting this bit creates a Bus-Disconnect condition. Reading this register reflects bit status.

**Bit 4. Assert/ACK.** Bit 4 is used by the bus initiator to assert /ACK on the SCSI Bus. In order to assert /ACK, the Target Mode bit (Mode Register, bit 6) must be False. Writing a zero to this bit deasserts /ACK. Reading this register reflects bit status.

**Bit 5. "0" (Write Bit).** Bit 5 should be written with a 0 for proper operation.

**Bit 5. LA (Lost Arbitration - Read Bit).** Bit 5, when active, indicates that the SCSI detected a Bus-Free condition, arbitrated for use of the bus by asserting /BSY and its ID on the Data Bus, and lost Arbitration due to /SEL being asserted by another bus device. This bit is active only when the Arbitrate bit (Mode Register, bit 0) is active.

**Bit 6. Test Mode (Write Bit).** Bit 6 is written during a test environment to place all output drivers, in the high impedance state.

**Bit 6. AIP (Arbitration in Process - Read Bit).** Bit 6 is used to determine if Arbitration is in progress. For this bit to be active, the Arbitrate bit (Mode Register, bit 0) must have been set previously. It indicates that a Bus-Free condition has been detected and that the chip has asserted /BSY and put the contents of the Output Data Register onto the SCSI Bus. AIP will remain active until the Arbitrate bit is reset.

**Bit 7. Assert/RST.** Whenever a one is written to bit 7 of the Initiator Command Register, the /RST signal is asserted on the SCSI Bus. The /RST signal will remain asserted until this bit is reset or until an external /RESET occurs. After this bit is set (1), IRQ goes active and all internal logic and control registers are reset (except for the interrupt latch and the Assert/RST bit). Writing a zero to bit 7 of the Initiator Command Register deasserts the /RST signal. The status of this bit is monitored by reading the Initiator Command Register.

**Mode Register. Address 2 (Read/Write).** The Mode Register controls the operation of the chip. This register determines whether the SCSI operates as an Initiator or a Target, whether DMA transfers are being used, whether parity is checked, and whether interrupts are generated on various external conditions. This register is read to check the value of these internal control bits (Figure 10).

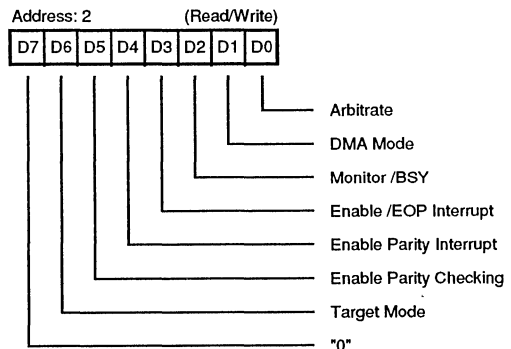


Figure 10. Mode Register

## FUNCTIONAL DESCRIPTION (Continued)

**Bit 0. Arbitrate.** The Arbitrate bit is set (1) to start the Arbitration process. Prior to setting this bit, the Output Data Register should contain the proper SCSI device ID value. Only one data bit should be active for SCSI Bus Arbitration. The SCSI waits for a Bus-Free condition before entering the Arbitration phase. The results of the Arbitration phase is determined by reading the status bits LA and AIP (Initiator Command Register, bits 5 and 6, respectively).

**Bit 1. DMA Mode.** The DMA Mode bit is normally used to enable a DMA transfer and must be set (1) prior to writing Start DMA Send Register, Start DMA Target Receive Register, and Start DMA Initiator Receiver Register. These three registers are used to start DMA transfers. The Target Mode bit (Mode Register, bit 6) must be consistent with writes to Start DMA Target Receive and Start DMA Initiator Receive Registers [i.e., set (1) for a write to start DMA Target Receive Register and set (0) for a write to Start DMA Initiator Receive Register]. The control bit Assert Data Bus (Initiator Command Register, bit 0) must be True (1) for all DMA send operations. In the DMA mode, /REQ and /ACK are automatically controlled.

The DMA Mode bit is not reset upon the receipt of an /EOP signal. Any DMA transfer is stopped by writing a zero into this bit location; however, care must be taken not to cause /CS and /DACK to be active simultaneously.

**Bit 2. Monitor Busy.** The Monitor Busy bit, when True (1), causes an interrupt to be generated for an unexpected loss of /BSY. When the interrupt is generated due to loss of /BSY, the lower six bits of the Initiator Command Register are reset (0) and all signals are removed from the SCSI Bus.

**Bit 3. Enable/EOP interrupt.** The enable /EOP interrupt, when set (1), causes an interrupt to occur when the /EOP (End of Process) signal is received from the DMA controller logic.

**Bit 4. Enable Parity Interrupt.** The Enable Parity Interrupt bit, when set (1), will cause an interrupt (IRQ) to occur if a parity error is detected. A parity interrupt will only be generated if the Enable Parity Checking bit (bit 5) is also enabled (1).

**Bit 5. Enable Parity Checking.** The Enable Parity Checking bit determines whether parity errors are ignored or saved in the parity error latch. If this bit is reset (0), parity is ignored. Conversely, if this bit is set (1), parity errors are saved.

**Bit 6. Targetmode.** The Targetmode bit allows the SCSI to operate as either a SCSI Bus Initiator, bit reset (0), or as a SCSI Bus Target device, bit set (1). If the signals /ATN and /ACK are to be asserted on the SCSI Bus, the Targetmode bit must be reset (0). If the signals C//D, I//O, /MSG, and /REQ are to be asserted on the SCSI Bus, the Targetmode bit must be set (1).

**Bit 7. 0.** Bit 7 should be written with a zero for proper operation.

**Target Command Register. Address 3(Read/Write).** When connected as a target device, the Target Command Register (Figure 11) allows the CPU to control the SCSI Bus Information Transfer phase and/or to assert /REQ by writing this register. The Targetmode bit (Mode Register, bit 6) must be True (1) for bus assertion to occur. The SCSI Bus phases are described in Table 2.

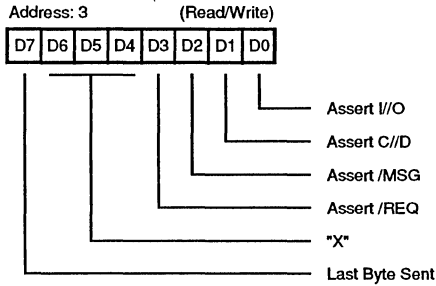
**Table 2. SCSI Information Transfer Phase**

Bus Phase	ASSERT I/O	ASSERT C/D	ASSERT /MS
Data Out	0	0	0
Unspecified	0	0	1
Command	0	1	0
Message Out	0	1	1
Data In	1	0	0
Unspecified	1	0	1
Status	1	1	0
Message In	1	1	1

When connected as an Initiator with DMA Mode True, if the phase lines I//O, C//D, and /MSG do not match the phase bits in the Target Command Register, a phase mismatch interrupt is generated when /REQ goes active. To send data as an Initiator, the Assert I//O, Assert C//D, and Assert /MSG bits must match the corresponding bits in the Current SCSI Bus Status Register. The Assert /REQ bit (bit 3) has no meaning when operating as an Initiator.

Bits 4, 5, and 6 are not used.

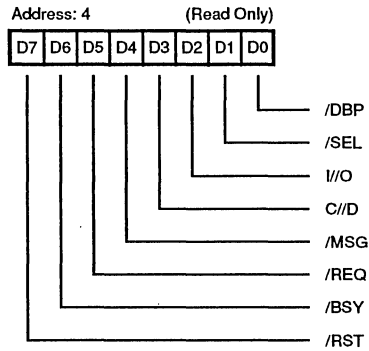
**Bit 7. Last Byte Sent (Read Only).** The End Of DMA Transfer bit (Bus and Status Register, bit 7) only indicates when the last byte was received from the DMA controller. The Last Byte Sent bit can be used to flag that the last byte of the DMA send operation has been transferred on the SCSI Data Bus.



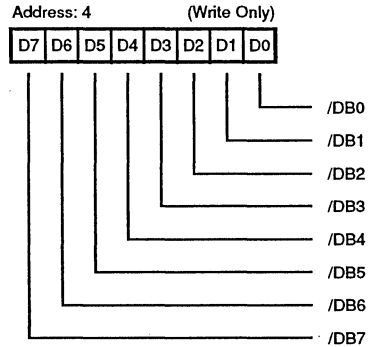
**Figure 11. Target Command Register**

**Current SCSI Bus Status Register. Address 4 (Read Only).** The Current SCSI Bus Register is a read-only register which is used to monitor seven SCSI Bus control signals, plus the Data Bus parity bit. For example, an Initiator device can use this register to determine the current bus phase and to poll /REQ for pending data transfers. This register may also be used to determine why a particular interrupt occurred. Figure 12 describes the Current SCSI Bus Status Register.

**Select Enable Register. Address 4 (Write Only).** The Select Enable Register (Figure 13) is a write-only register which is used as a mask to monitor a signal ID during a selection attempt. The simultaneous occurrence of the correct ID bit, /BSY FALSE, and /SEL TRUE will cause an interrupt. This interrupt can be disabled by resetting all bits in this register. If the Enable Parity Checking bit (Mode Register, bit 5) is active (1), parity is checked during selection.



**Figure 12. Current SCSI Bus Status Register**



**Figure 13. Select Enable Register**

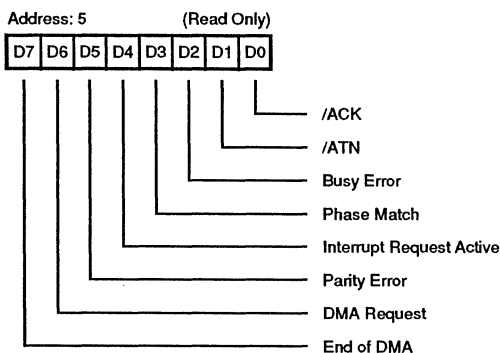


## FUNCTIONAL DESCRIPTION (Continued)

**Bus and Status Register.** *Address 5 (Read Only).* The Bus and Status Register (Figure 14) is a read-only register which can be used to monitor the remaining SCSI control signals not found in the Current SCSI Bus Status Registers (*/ATN* and */ACK*), as well as six other status bits. The following describes each bit of the Bus Status Register individually.

**Bit 0. */ACK.*** Bit 0 reflects the condition of the SCSI Bus control signal */ACK*. This signal is normally monitored by the Target device.

**Bit 1. */ATN.*** Bit 1 reflects the condition of the SCSI Bus control signal */ATN*. This signal is normally monitored by the Target device.



**Figure 14. Bus and Status Register**

**Bit 2. *Busy Error.*** The Busy Error bit is active if an unexpected loss of the */BSY* signal has occurred. This latch is set whenever the Monitor Busy bit (Mode Register, bit 2) is True and */BSY* is False. An unexpected loss of */BSY* disables any SCSI outputs and resets the DMA Mode bit (Mode Register, bit 1).

**Bit 3. *Phase Match.*** The SCSI signals */MSG*, *C//D*, and *I//O*, represent the current information transfer phase. The Phase Match bit indicates whether the current SCSI Bus phase matches the lower three bits of the Target Command Register. Phase Match is continuously updated and is only significant when operating as a Bus Initiator. A phase match is required for data transfers to occur on the SCSI Bus.

**Bit 4. *Interrupt Request Active.*** Bit 4 is set if an enabled interrupt condition occurs. It reflects the current state of the IRQ output and can be cleared by reading the Reset Parity/Interrupt Register.

**Bit 5. *Parity Error.*** Bit 5 is set if a parity error occurs during a data receive or a device selection. The Parity Error bit can only be set (1) if the Enable Parity Check bit (Mode Register, bit 5) is active (1). This bit may be cleared by reading the Reset Parity/Interrupt Register.

**Bit 6. *DMA Request.*** The DMA Request bit allows the CPU to sample the output pin DRQ. DRQ can be cleared by asserting */DACK* or by resetting the DMA MODE bit (bit 1) in the Mode Register. The DRQ signal does not reset when a phase-mismatch interrupt occurs.

**Bit 7. *End Of DMA Transfer.*** The End Of DMA Transfer bit is set if */EOP*, */DACK*, and either */RD* or */WR* are simultaneously active for at least 100 ns. Since the */EOP* signal can occur during the last byte sent to the Output Data Register, the */REQ* and */ACK* signals should be monitored to ensure that the last byte has been transferred. This bit is reset when the DMA MODE bit is reset (0) in the Mode Register.

**DMA Registers.** Three write-only registers are used to initiate all DMA activity. They are: Start DMA Send, Start DMA Target Receive, and Start DMA Initiator Receive. Performing a write operation into one of these registers starts the desired type of DM transfer. Data presented to the SCSI on signals D7-D0 during the register write is meaningless and has no effect on the operation. Prior to writing these registers, the DMA Mode bit (bit 1), and the Target mode bit (bit 6) in the Mode Register must be appropriately set. The individual registers are briefly described as follows:

**Start DMA Send. Address 5 (Write Only).** This register is written to initiate a DMA send, from the DMA to the SCSI Bus, for either Initiator or Target role operations. The DMA Mode bit (Mode Register, bit 1) is set prior to writing this register.

**Start DMA Target Receive. Address 6 (Write Only).** This register is written to initiate a DMA receive - from the SCSI Bus to the DMA, for Target operation only. The DMA Mode bit (bit 1) and the Targetmode bit (bit 6) in the Mode Register must both be set (1) prior to writing this register.

**Start DMA Initiator Receive.** *Address 7 (Write Only).* This register is written to initiate a DMA receive from the SCSI Bus to the DMA, for Initiator operation only. The DMA Mode bit (bit 6) must be False (0) in the Mode Register prior to writing this register.

**Reset Parity/Interrupt.** *Address 7 (Read Only).* Reading this register resets the Parity Error bit (bit 5), the Interrupt Request bit (bit 4), and the Busy Error bit (bit 2) in the Bus and Status Register.

**On-Chip SCSI Hardware Support.** The SCSI is easy to use because of its simple architecture. The chip allows direct control and monitoring of the SCSI Bus by providing a latch for each signal. However, portions of the protocol define timings which are much too quick for traditional microprocessors to control. Therefore, hardware support has been provided for DMA transfers, bus arbitration, phase change monitoring, bus disconnection, bus reset, parity generation, parity checking, and device selection/reselection.

Arbitration is accomplished using a Bus-Free filter to continuously monitor /BSY. If /BSY remains inactive for at least 400 ns, the SCSI is considered free and Arbitration may begin. Arbitration will begin if the bus is free, /SEL is inactive, and the Arbitrate bit (Mode Register, bit 0) is active. Once arbitration has begun (/BSY asserted), an arbitration delay of 2.2  $\mu$ s must elapse before the Data Bus can be examined to determine if Arbitration is enabled. This delay is implemented in the controlling software driver.

The Z53C80 is a clockwise device. Delays such as bus-free delay, bus-set delay, and bus-settle delay are implemented using gate delays. These delays may differ between devices because of inherent process variations, but are well within the proposed ANSI X3.131 - 1986 specification.

**Interrupts.** The Z53C80 provides an interrupt output (IRQ) to indicate a task completion or an abnormal bus occurrence. The use of interrupts is optional and may be disabled by resetting the appropriate bits in the Mode Register or the Select Enable Register.

When an interrupt occurs, the Bus and Status Register and the Current SCSI Bus Status Register (Figures 12 and 14) must be read to determine which condition created the interrupt. IRQ can be reset simply by reading the Reset Parity/Interrupt Register or by an external chip reset /RESET active for 100 ns.

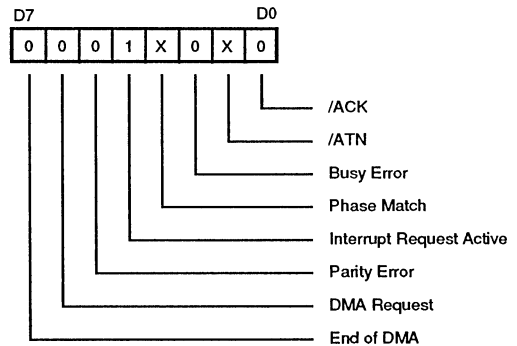
Assuming the Z53C80 has been properly initialized, an interrupt will be generated if the chip is selected or reselected, if an /EOP signal occurs, if a parity error occurs during a data transfer, if a bus phase mismatch occurs, or if a SCSI Bus disconnection occurs.

**Selection Reselection.** The Z53C80 generates a select interrupt if SEL is active (0), its device ID is True and /BSY is False for at least a bus-settle delay. If I/O is active, this is considered a reselect interrupt. The correct ID bit is determined by a match in the Select Enable Register. Only a single bit match is required to generate an interrupt. This interrupt may be disabled by writing zeros into all bits of the Select Enable Register.

If parity is supported, parity should be good during the selection phase. Therefore, if the Enable Parity bit (Mode Register, bit 5) is active, the Parity Error bit is checked to ensure that a proper selection has occurred. The Enable Parity Interrupt bit need not be set for this interrupt to be generated.

The proposed SCSI specification also requires that no more than two device IDs be active during the selection process. To ensure this, the Current SCSI Data Register is read.

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 15 and 16, respectively.



**Figure 15. Bus and Status Register**

## FUNCTIONAL DESCRIPTION (Continued)

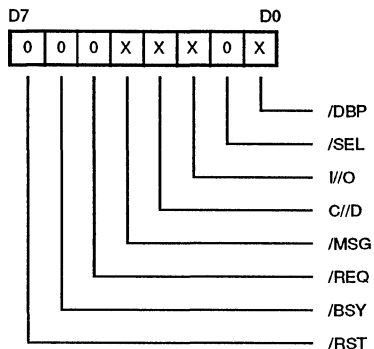


Figure 16. Current SCSI Bus Status Register

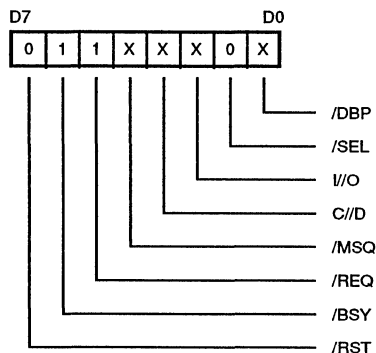


Figure 18. Current SCSI Bus Status Register

**End of Process (EOP) Interrupt.** An End Of Process signal (EOP) which occurs during a DMA transfer (DMA Mode True) will set the End of DMA Status bit (Bus and Status Register bit 7) and will optionally generate an interrupt if Enable EOP Interrupt bit (Mode Register, bit 3) is True. The /EOP pulse will not be recognized (End of DMA bit set) unless /EOP, /DACK, and either /RD or /WR are concurrently active for at least 50 ns. DMA transfers can still occur if /EOP was not asserted at the correct time. This interrupt is disabled by resetting the Enable EOP Interrupt bit.

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register for this interrupt are shown in Figures 17 and 18.

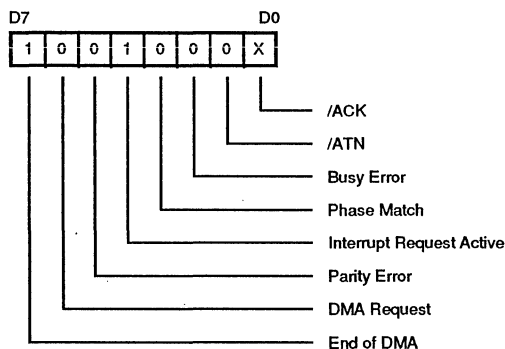


Figure 17. Bus and Status Register

The End of DMA bit is used to determine when a block transfer is complete. Receive operations are complete when there is no data left in the chip and no additional handshakes occurring. The only exception to this is receiving data as an Initiator and the Target opts to send additional data for the same phase. In this /REQ goes active and the new data is present in the Input Data Register. Since a phase-mismatch interrupt will not occur, /REQ and /ACK need to be sampled to determine that the Target is attempting to send more data.

For send operations, the End of DMA bit is set when the DMA finishes its transfers, but the SCSI transfer may still be in progress. If connected as a Target, /REQ and /ACK should be sampled until both are False. If connected as an Initiator, a phase change interrupt is used to signal the completion of the previous phase. It is possible for the Target to request additional data for the same phase. In this case, a phase change will not occur and both /REQ and /ACK are sampled to determine when the last byte was transferred.

**SCSI Bus Reset.** The SCSI generates an interrupt when the /RST signal transitions to True. The device releases all bus signals within a bus-clear delay of this transition. This interrupt also occurs after setting the Assert /RST bit (Initiator Command Register, bit 7). This interrupt cannot be disabled. (Note: /RST is not latched in bit 7 of the Current SCSI Bus Status Register and is not active when this port is read. For this case, the Bus Reset interrupt is determined by default.)

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 19 and 20, respectively.

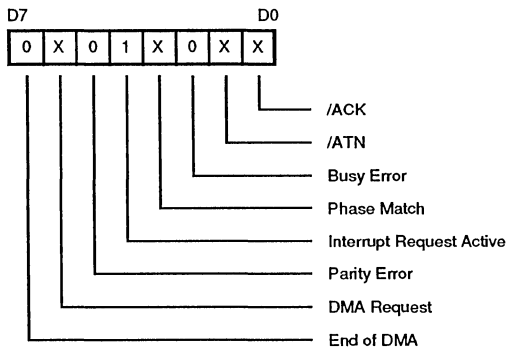


Figure 19. Bus and Status Register

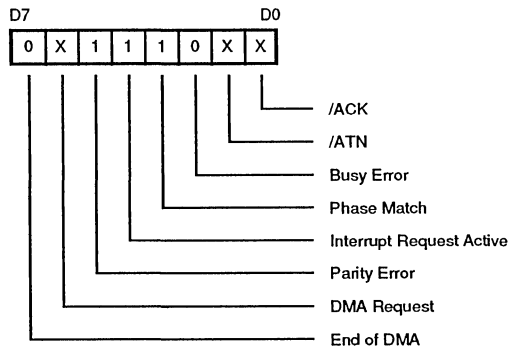


Figure 21. Bus and Status Register

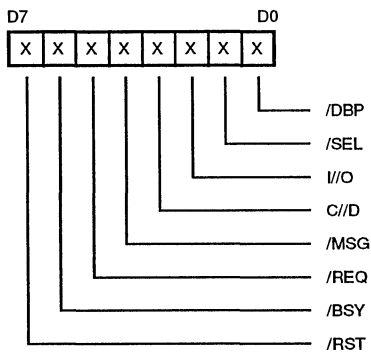


Figure 20. Current SCSI Bus Status Register

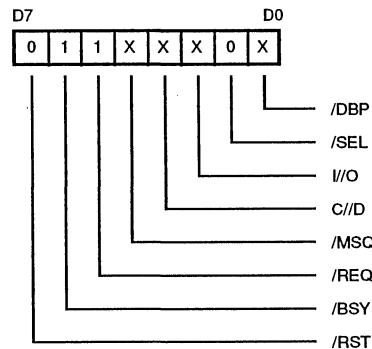


Figure 22. Current SCSI Bus Status Register

**Parity Error.** An interrupt is generated for a received parity error if the Enable Parity Check (bit 5) and the Enable Parity Interrupt (bit 4) bits are set (1) in the Mode Register. Parity is checked during a read of the Current SCSI Data Register and during a DMA receive operation. A parity error can be detected without generating an interrupt by disabling the Enable Parity Interrupt bit and checking the Parity Error flag (Bus and Status Register, bit 5).

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 21 and 22, respectively.

**Bus Phase Mismatch.** The SCSI phase lines have the signals I/O, C//D, and /MSG. These signals are compared with the corresponding bits in the Target Command Register: Assert I/O (bit 0), Assert C//D (bit 1), and Assert /MSG (bit 2). The comparison occurs continually and is reflected in the Phase Match bit (bit 3) of the Bus and Status Register. If the DMA Mode bit (Mode Register, bit 1) is active and a phase mismatch occurs when /REQ transitions from False to True, an interrupt (IRQ) is generated.

## FUNCTIONAL DESCRIPTION (Continued)

A phase mismatch prevents the recognition of /REQ and removes the chip from the bus during an Initiator send operation (/DB7-/DB0 and /DBP will not be driven even through the Assert Data Bus bit (Initiator Command Register, bit 0). This may be disabled by resetting the DMA Mode bit (Note: It is possible for this interrupt to occur when connected as a Target if another device is driving the phase lines to a different state).

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 23 and 24, respectively.

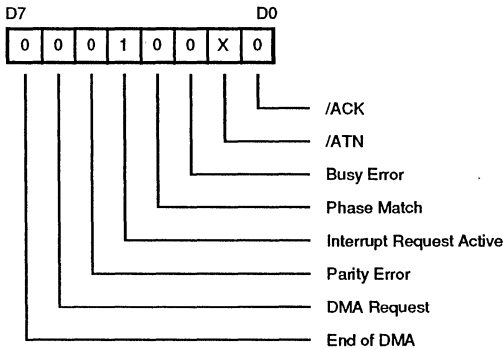


Figure 23. Bus and Status Register

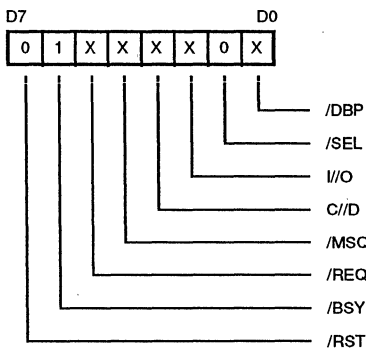


Figure 24. Current SCSI Bus Status Register

**Loss of BSY.** If the Monitor Busy bit (bit 2) in the Mode Register is active, an interrupt is generated if the BSY signal goes FALSE for at least a bus-settle delay. This interrupt is disabled by resetting the Monitor Busy bit. Register values are displayed in Figures 25 and 26.

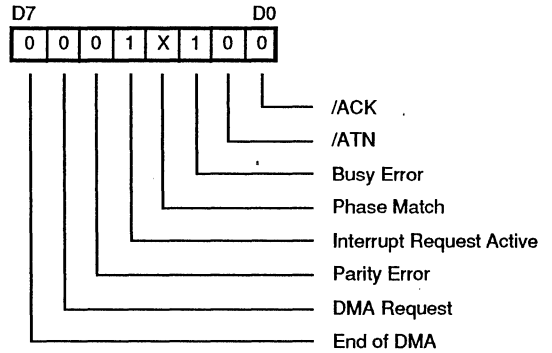


Figure 25. Bus and Status Register

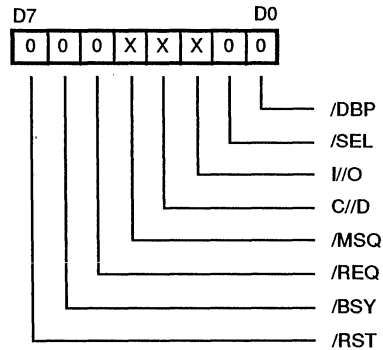


Figure 26. Current SCSI Bus Status Register

---

**Reset Conditions.** Three possible reset situations exist with the Z53C80, as follows:

**Hardware Chip Reset.** When the signal RST is active for at least 100 ns, the Z53C80 device is re-initialized and all internal logic and control registers are cleared. This is a chip reset only and does not create a SCSI Bus-Reset condition.

**SCSI Bus Reset (/RST) Received.** When a SCSI /RST signal is received, an IRQ interrupt is generated and a chip reset is performed. All internal logic and registers are cleared, except for the IRQ interrupt latch and the Assert /RST bit (bit 7) in the Initiator Command Register. (Note: The /RST signal may be sampled by reading the Current SCSI Bus Status Register, however, this signal is not latched and may not be present when this port is read.)

**SCSI Bus Reset (/RST) Issued.** If the CPU sets the Assert /RST bit (+ 7) in the Initiator Command Register, the /RST signal goes active on the SCSI Bus and an internal reset is performed. Again, all internal logic and registers are cleared except for the IRQ interrupt latch and the Assert /RST bit (bit 7) in the Initiator Command Register. The /RST signal will continue to be active until the Assert /RST bit is reset or until a hardware reset occurs.

**Data Transfers.** Data is transferred between SCSI Bus devices in one of four modes: 1) Programmed I/O, 2) Normal DMA, 3) Block Mode DMA, or 4) Pseudo DMA. The following sections describe these modes in detail (Note: for all data transfer operations /DACK and /CS should never be active simultaneously.)

**Programmed I/O Transfers.** Programmed I/O is the most primitive form of data transfer. The /REQ and /ACK handshake signals are individually monitored and asserted by reading and writing the appropriate register bits. This type of transfer is normally used when transferring small blocks of data such as command blocks or message and status bytes. An Initiator send operation would begin by setting the C//D, I//O, and /MSG bits in the Target Command Register to the correct state so that a phase match exists. In addition to the phase match condition, it is necessary for the Assert Data Bus bit (Initiator Command Register, bit 0) to be True and the received I/O signal to be False for the Z53C80 to send data. For each transfer, the data is loaded into the Output Data Register. The CPU then waits for the /REQ bit (Current SCSI Bus Status Register, bit 5) to become active. Once /REQ goes active, the Phase Match bit (Initiator Command Register, bit 4) is set. The /REQ bit is sampled until it becomes FALSE and the CPU resets the Assert /ACK bit to complete the transfer.

**Normal DMA Mode.** DMA transfers are normally used for large block transfers. The SCSI chip outputs a DMA

request (DRQ) whenever it is ready for a byte transfer. External DMA logic uses this DRQ signal to generate /DACK and a /RD or a /WR pulse to the Z53C80. DRQ goes inactive when /DACK is asserted and /DACK goes inactive some time after the minimum read or write pulse width. This process is repeated for every byte. For this mode, /DACK should not be allowed to cycle unless a transfer is taking place.

**Block Mode Transfers.** The Block Mode DMA transfers allow an external DMA controller, such as the Intel 8237, to perform successive DMA transfers without abandoning the data bus to the microprocessor. Keeping an active /DACK prevents the (Intel-type) CPUs from gaining control of the system bus. The Block Mode handshaking method does not increase the transfer rate. Preventing the CPU from multiplexing the system bus does not have any speed advantages. Therefore, this is not recommended for initiator use.

In the Block Mode, the SCSI chip asserts the DRQ signal to initiate the transfer. The DMA controller responds to the DRQ signal by asserting the /DACK and remains asserted throughout the transfer. The 53C80 asserts the READY signal after the /IOR or /IOW signals deassert, effectively replacing the DRQ signal. The READY signal for Intel-type DMA controllers extends the memory read and write signals. Therefore, D7-D0 is available to be read or written on the system bus until the SCSI chip is ready for the next transfer. This transfer method prevents the CPU from executing any action, such as a refresh cycle on the system bus. In the non-block DMA mode, the system bus is unoccupied until the 53C80 asserts DRQ. This indicates that the chip is ready for the next byte transfer. The advantage of this mode is that it allows the CPU to use the system bus while the 53C80 is transferring data across the SCSI bus.

Caution must be taken when executing this mode due to the operation of READY. For example, if a phase mismatch interrupt occurs, the READY signal will stay inactive and IRQ will be active. Then, the DMA controller cannot give the system bus back to the CPU for the 53C80 interrupt to be serviced since READY remains inactive. READY must be asserted to continue the bus cycle. Therefore, /EOP should be used in Block Mode so that the CPU can regain control of the bus after the last byte has been transferred. To make READY active again, reset the DMA Mode Bit.

Block Mode transfers are stopped in the same fashion as in the Block Mode. This is executed by resetting the DMA Mode Bit or using the /EOP signal. (See the previous section, Normal DMA Mode, for more information on stopping a DMA transfer.)

---

## FUNCTIONAL DESCRIPTION (Continued)

**Pseudo DMA Mode.** To avoid monitoring and asserting the request/acknowledgment handshake signals for programmed I/O transfers, the system may be designed to implement a pseudo DMA mode. This mode is implemented by programming the Z53C80 to operate in the DMA mode, but using the CPU to emulate the DMA handshake. DRQ may be detected by polling the DMA Request bit (bit 6) in the Bus and Status Register, by sampling the signal through an external port, or by using it to generate a CPU interrupt. Once DRQ is detected, the CPU can perform a read or write data transfer. This CPU read/write is externally decoded to generate the appropriate /DACK and /RD or /WR signals.

Often, external decoding logic is necessary to generate the /CS signal. This same logic may be used to generate /DACK at no extra cost and provide an increased performance in programmed I/O transfers.

**Halting a DMA Operation.** The EOP signal is not the only way to halt a DMA transfer. A bus phase mismatch or a reset of the DMA MODE bit (Mode Register, bit 1) can also terminate a DMA cycle for the current bus phase.

**Using the /EOP Signal.** If /EOP is used, it should be asserted for at least 50 ns while /DACK and /RD or /WR are simultaneously active. Note, however, that if /RD or /WR is not active, an interrupt is generated, but the DMA activity continues. The /EOP signal does not reset the DMA MODE bit. Since the /EOP signal can occur during the last byte sent to the Output Data Register, the /REQ and /ACK signals are monitored to ensure that the last byte has transferred.

**Bus Phase Mismatch Interrupt.** A bus phase mismatch interrupt is used to halt the transfer if operating as an Initiator. Using this method frees the host from maintaining a data length counter and frees the DMA logic from providing the /EOP signal. If performing an Initiator send operation, the Z53C80 requires /DACK to cycle before /ACK goes inactive. Since phase changes cannot occur if /ACK is active, either /DACK must be cycled after the last byte is sent or the DMA Mode bit must be reset in order to receive the phase mismatch interrupt.

**Resetting the DMA MODE Bit.** A DMA operation may be halted at any time simply by resetting the DMA Mode bit. It is recommended that the DMA Mode bit be reset after receiving an /EOP or bus phase-mismatch interrupt. The DMA Mode bit must then be set before writing any of the start DMA registers for subsequent bus phases.

If resetting the DMA Mode bit is used instead of /EOP for Target role operation, then care must be taken to reset this bit at the proper time. If receiving data as a Target device, the DMA Mode bit must be reset once the last DRQ is received and before /DACK is asserted to prevent an additional /REQ from occurring. Resetting this bit causes DRQ to go inactive. However, the last byte received remains in the Input Data Register and may be obtained either by performing a normal CPU read or by cycling /DACK and /RD. In most cases, /EOP is easier to use when operating as a Target device.

# READ REGISTERS

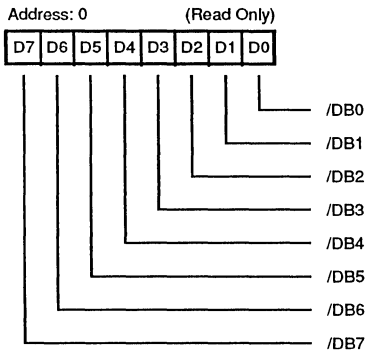


Figure 27. Current SCSI Data Register

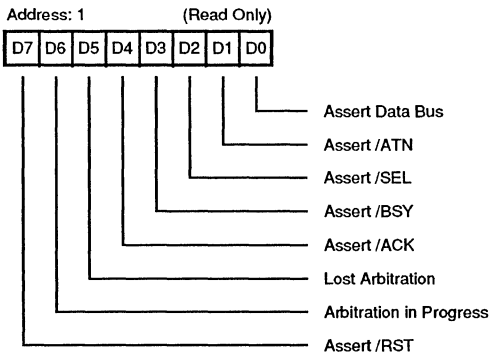


Figure 28. Initiator Command Register

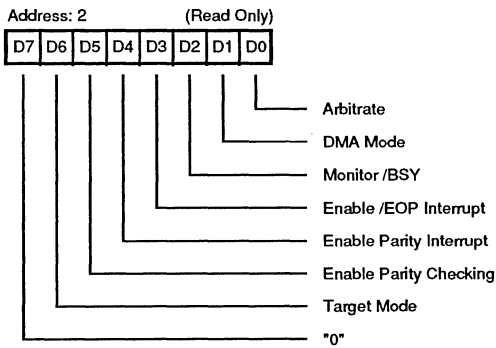


Figure 29. Mode Register

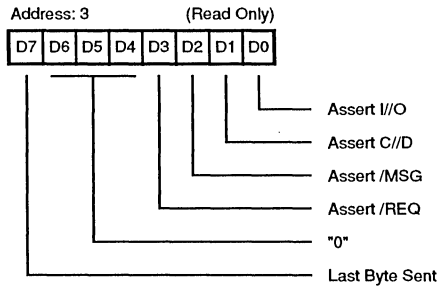


Figure 30. Target Command Register

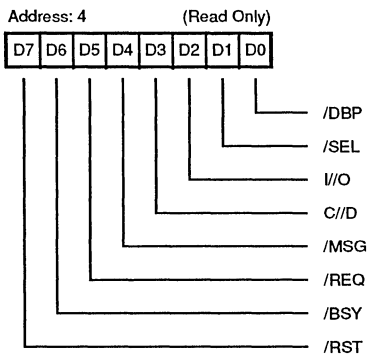


Figure 31. Current SCSI Bus Status Register

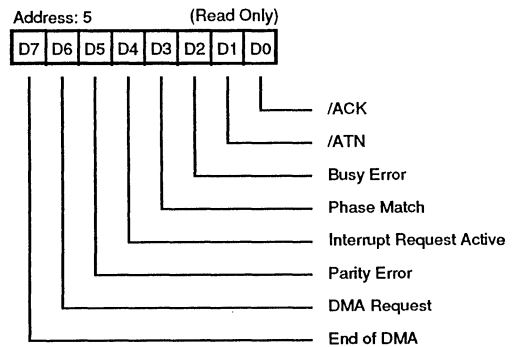


Figure 32. Bus and Status Register



## READ REGISTERS (Continued)

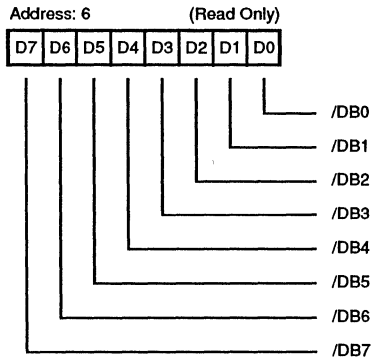


Figure 33. Input Data Register

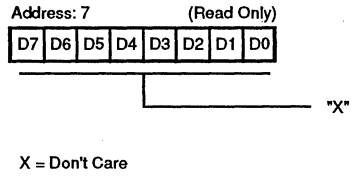


Figure 34. Reset Parity/Interrupt

## WRITE REGISTERS

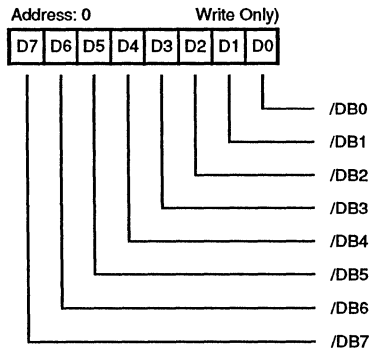


Figure 35. Output Data Register

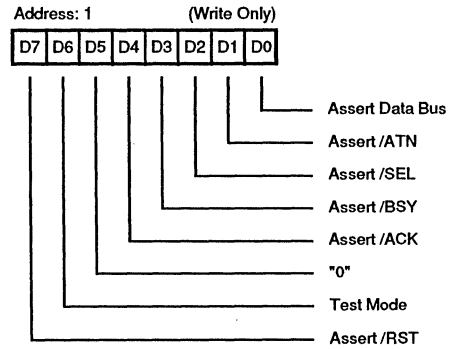


Figure 36. Initiator Command Register

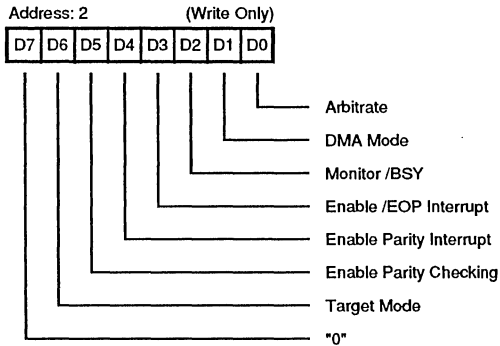


Figure 37. Mode Register

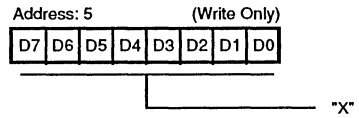


Figure 40. Start DMA Send

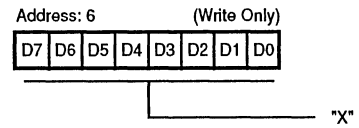


Figure 41. Start DMA Target Receive

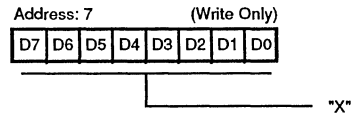


Figure 42. Start DMA Initiator Receive

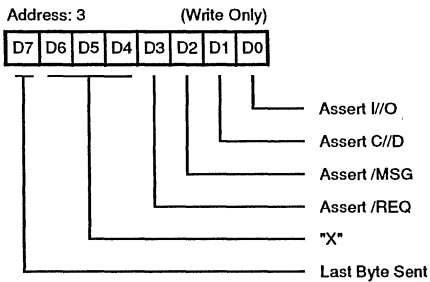


Figure 38. Target Command Register

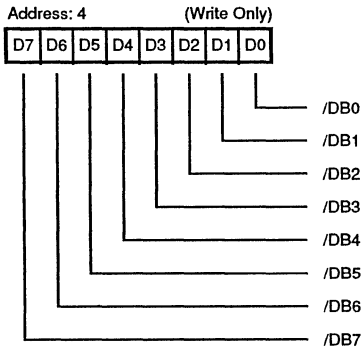


Figure 39. Select Enable Register

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins with respect to GND	-0.3V to +7.0V
Operating Ambient Temperature	See Ordering Information
Storage Temperature	-65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to this device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC characteristics and capacitance section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

- $+4.75V \leq V_{CC} \leq +5.25V$
- $GND = 0V$
- $T_A$  as specified in Ordering Information

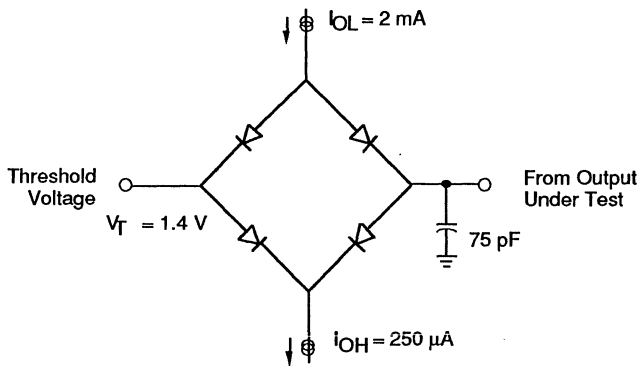


Figure 43. Standard Test Dynamic Load Circuit

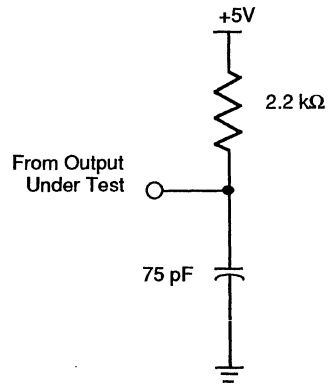


Figure 44. Open-Drain Test Load

## DC CHARACTERISTICS

Symbol	Parameter	Condition	Min	Max	Units
$V_{DD}$	Supply Voltage		4.75	5.25	V
$V_{IH}$	High-Level Input Voltage		2.0	$V_{DD} + 0.5$	V
$V_{IL}$	Low-Level Input Voltage		$V_{SS} - 0.5$	0.8	V
$I_{IH1}$	High-Level Input Current SCSI Bus Pins	$V_{IH} = 5.25V$ $V_{IL} = 0V$		50	$\mu A$
$I_{IH2}$	High-Level Input Current All Other Pins	$V_{IH} = 5.25V$		10	$\mu A$
$I_{IL1}$	Low-Level Input Current SCSI Bus Pins (Except /RST)	$V_{IL} = V_{SS}$		-50	$\mu A$
$I_{IL2}$	Low-Level Input Current All Other Pins	$V_{IL} = V_{SS}$		-10	$\mu A$
$V_{OH1}$	High-Level Output Voltage	$I_{OH} = -4mA$	2.4	$V_{DD}$	V
$V_{OL1}$	Low-Level Output Voltage SCSI Bus Pins	$I_{OL} = 48 mA$	$V_{SS}$	0.5	V
$V_{OL2}$	Low-Level Output Voltage All Other Pins	$I_{OL} = 8 mA$	$V_{SS}$	0.4	V
$I_{DD}$	Supply Current			15	mA
$T_A$	Operating Free-Air Temperature		0	70	$^{\circ}C$

## AC CHARACTERISTICS

### CPU Write Cycle Timing Diagram

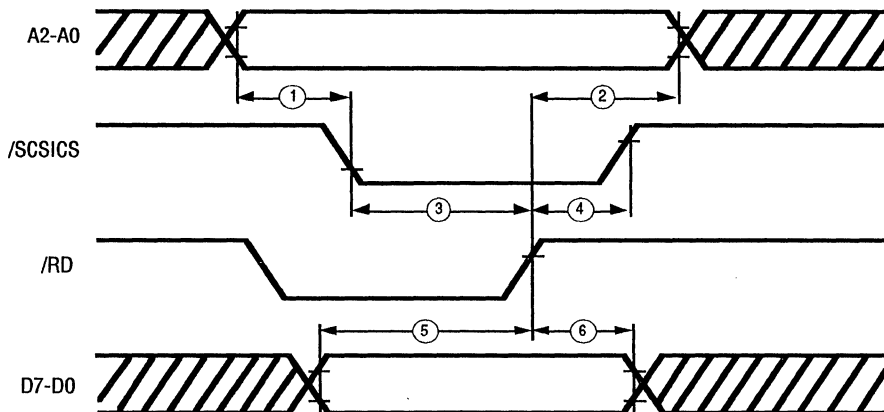


Figure 45. CPU Write Cycle

## AC CHARACTERISTICS

### CPU Write Cycle Table

No	Description	Min	Max	Units
1	Address Setup to Write Enable*	10		ns
2	Address Hold from End Write Enable*	10		ns
3	Write Enable Width*	40		ns
4	Chip Select Hold from End of /IOW	0		ns
5	Data Setup to end of Write Enable*	20		ns
6	Data Hold Time from End of /IOW	20		ns

**Note:**

\* Write Enable is the occurrence of /WR and /CS

## AC CHARACTERISTICS

### CPU Read Cycle Timing Diagram

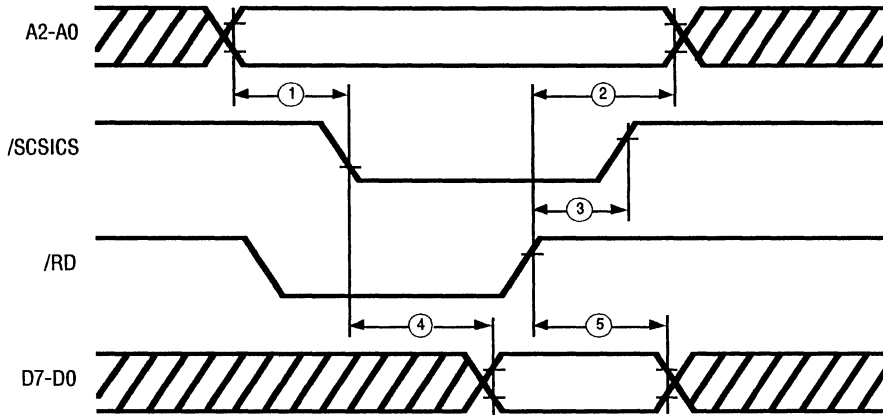


Figure 46. CPU Read Cycle

## AC CHARACTERISTICS

### CPU Read Cycle Table

No	Description	Min	Max	Units
1	Address Setup to Read Enable*	10		ns
2	Address Hold from End Read Enable*	10		ns
3	Chip Select Hold from End of /RD	0		ns
4	Data Access Time from Read Enable*		70	ns
5	Data Hold Time from End of Read Enable*	10		ns

**Note:**

\* Read Enable is the occurrence of /RD and /CS

## AC CHARACTERISTICS

### DMA Write (Non-Block Mode) Initiator Send Cycle Timing Diagram

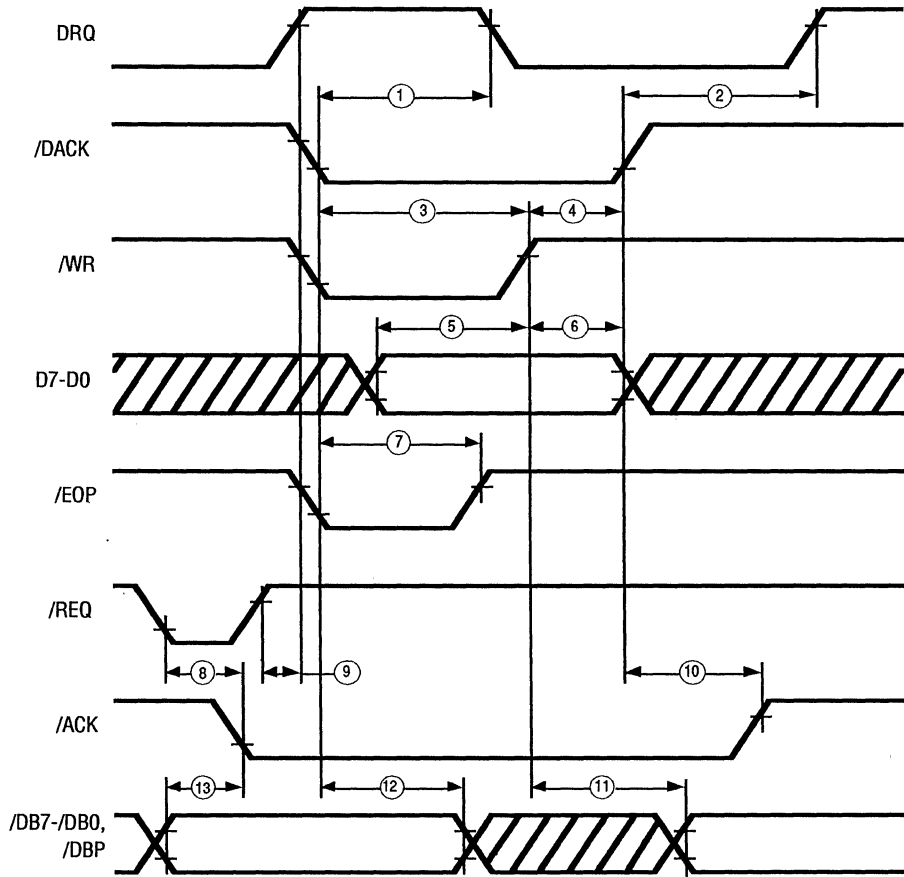


Figure 47. DMA Write (Non-Block Mode) Initiator Send Cycle

---

## AC CHARACTERISTICS

### DMA Write Initiator Send Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low		60	ns
2	/DACK High to DRQ High	30		ns
3	Write Enable Width*	50		ns
4	/DACK Hold from End of /WR	0		ns
5	Data Setup to End of Write Enable*	50		ns
6	Data Hold Time from End of /WR	25		ns
7	Width of /EOP Pulse [1]	50		ns
8	/REQ Low to /ACK Low		70	ns
9	/REQ High to DRQ High		70	ns
10	/DACK High to /ACK High		90	ns
11	/WR High to Valid SCSI Data		50	ns
12	Data Hold from Write Enable*	15		ns
13	Data Setup to /ACK Low	55		ns

**Notes:**

[1] /EOP, /WR, and /DACK must be concurrently Low for at least T7 for proper recognition of the /EOP pulse.

\* Write Enable is the occurrence of /WR and /DACK.



# AC CHARACTERISTICS

## DMA Read (Non-Block Mode) Initiator Receive Timing Diagram

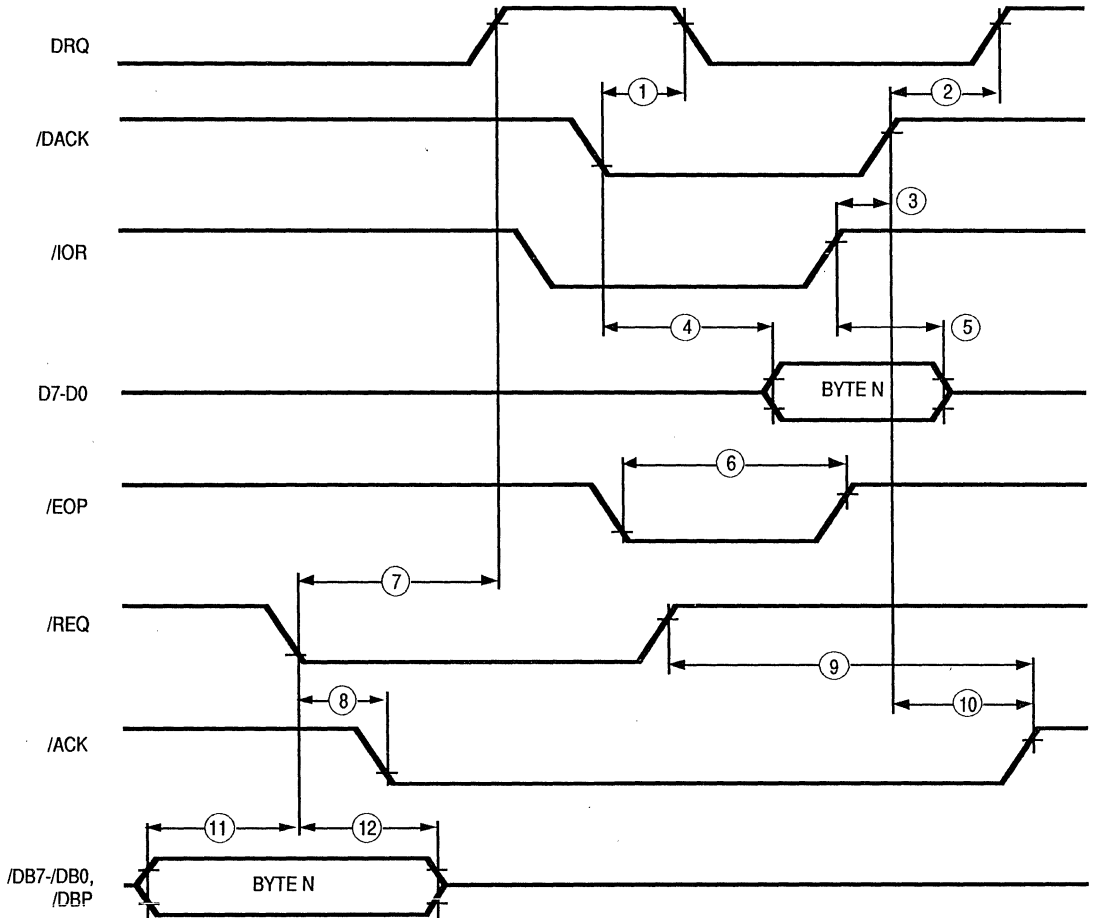


Figure 48. DMA Read (Non-Block Mode) Initiator Receive

---

## AC CHARACTERISTICS

### DMA Read (Non-Block Mode) Initiator Receive Table

Name	Description	Min	Max	Units
1	DRQ False from /DACK True		60	ns
2	/DACK False to DRQ True	30		ns
3	/DACK Hold Time from End of /IOR	0		ns
4	Data Access Time from Read Enable*		70	ns
5	Data Hold Time from End of /IOR	10		ns
6	Width of /EOP Pulse [1]	50		ns
7	/REQ True to DRQ True		70	ns
8	/REQ True to /ACK True		70	ns
9	/REQ False to /ACK False (/DACK False)		80	ns
10	/DACK False to /ACK False (/REQ False)		90	ns
11	DATA Setup Time to /REQ	20		ns
12	DATA Hold Time from /REQ True	50		ns

**Notes:**

[1] /EOP, /IOR and /DACK must be concurrently true for at least T6 for proper recognition of the /EOP pulse.

\*Read enable is the occurrence of both /IOR and /DACK.

## AC CHARACTERISTICS

### DMA Write (Non-Block Mode) Target Send Cycle Timing Diagram

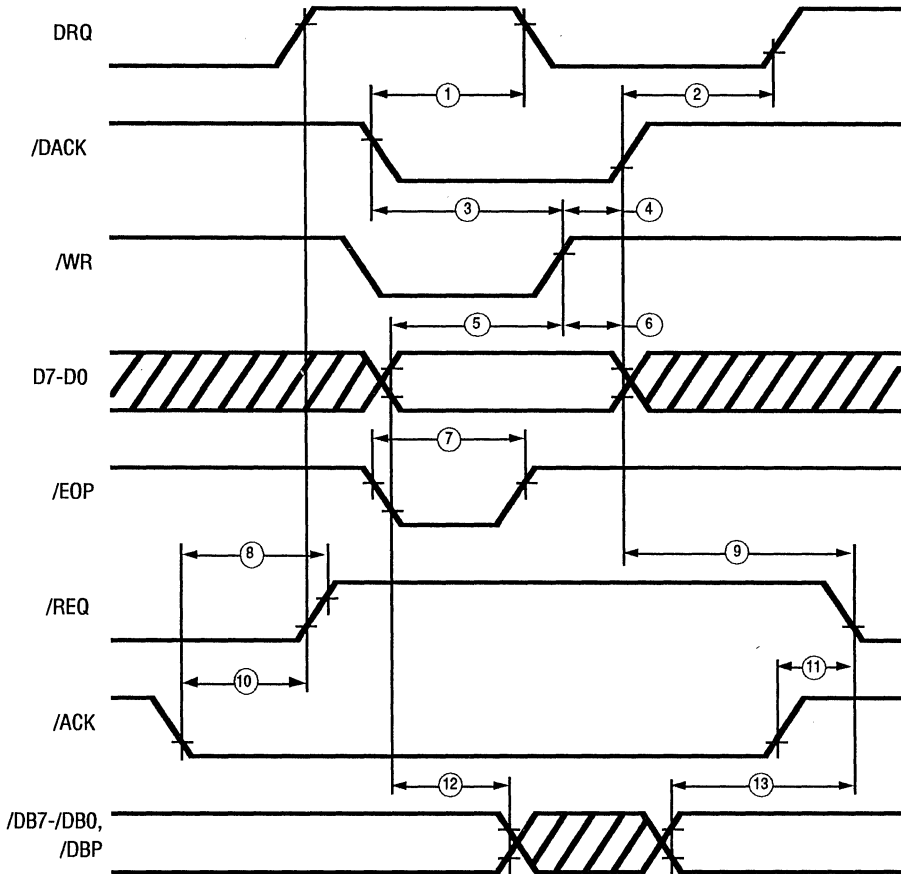


Figure 49. DMA Write (Non-Block Mode) Target Send Cycle

## AC CHARACTERISTICS

### DMA Write Target Send Cycle Table

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low		60	ns
2	/DACK High to DRQ High	30		ns
3	Write Enable Width*	50		ns
4	/DACK Hold from /WR High	0		ns
5	Data Setup to End of Write Enable*	50		ns
6	Data Hold Time from End of /WR	25		ns
7	Width of /EOP Pulse [1]	50		ns
8	/ACK Low to /REQ High		80	ns
9	/REQ from End of /DACK (/ACK High)		90	ns
10	/ACK Low to DRQ High (Target)		70	ns
11	/ACK High to /REQ Low (/DACK High)		100	ns
12	Data Hold from Write Enable	15		ns
13	Data Setup to /REQ Low (Target)	55		ns

#### Notes:

[1] /EOP, /WR, and /DACK must be concurrently Low for at least T7 for proper recognition of the /EOP pulse.

\* Write Enable is the occurrence of /IOW and /DACK

## AC CHARACTERISTICS

### DMA Read (Non-Block Mode) Target Receive Timing Diagram

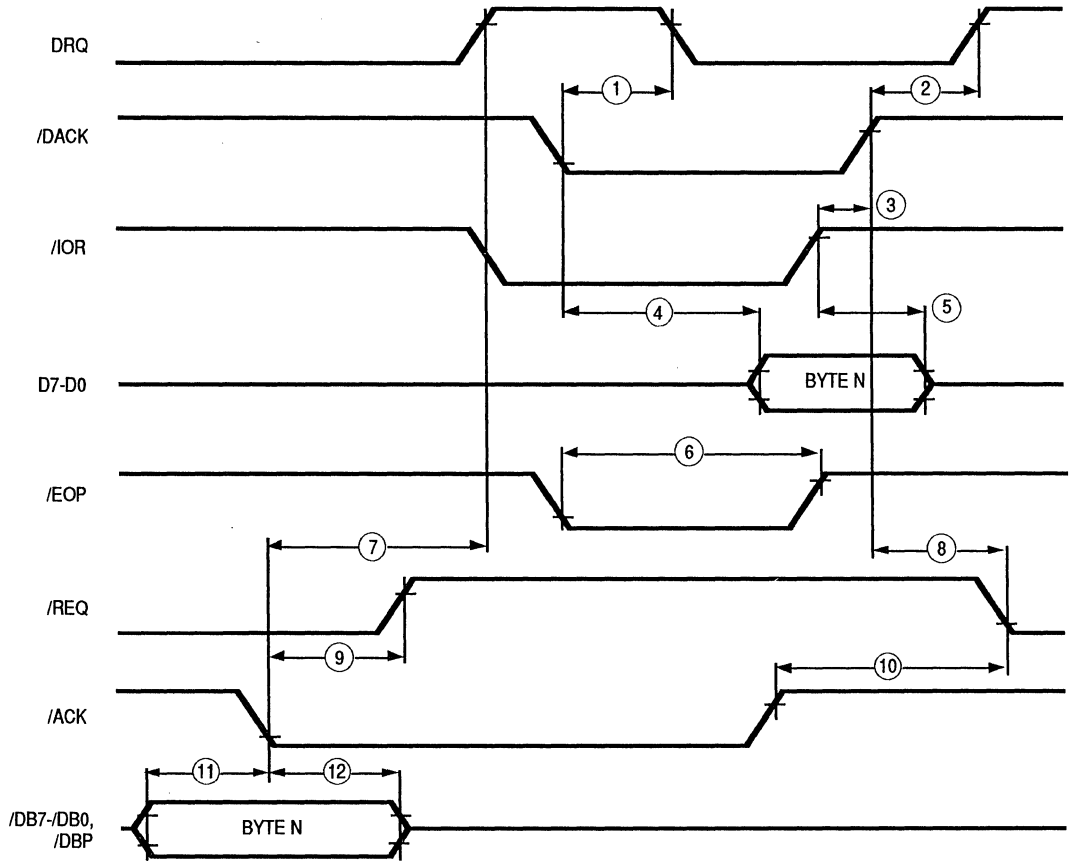


Figure 50. DMA Read (Non-Block Mode) Target Receive

---

## AC CHARACTERISTICS

### DMA Read (Non-Block Mode) Target Receive Table

Name	Description	Min	Max	Units
1	DRQ False from /DACK True		60	ns
2	/DACK False to DRQ True	30		ns
3	/DACK Hold Time from End of /IOR	0		ns
4	Data Access Time from Read Enable*		70	ns
5	Data Hold Time from End of /IOR	10		ns
6	Width of /EOP Pulse [1]	50		
7	/ACK True to DRQ True		70	
8	/DACK False to /REQ True (/ACK False)		90	
9	/ACK True to /REQ False		80	
10	/ACK False to /REQ True (/DACK False)		100	
11	DATA Setup Time to /ACK	20		
12	DATA Hold Time from /ACK True	30		

**Notes:**

[1] /EOP, /IOR and /DACK must be concurrently true for a least T6 for proper recognition of the /EOP pulse.

\*Read enable is the occurrence of both /IOR and /DACK.

# AC CHARACTERISTICS

## DMA Write (Block Mode) Target Send Timing Diagram

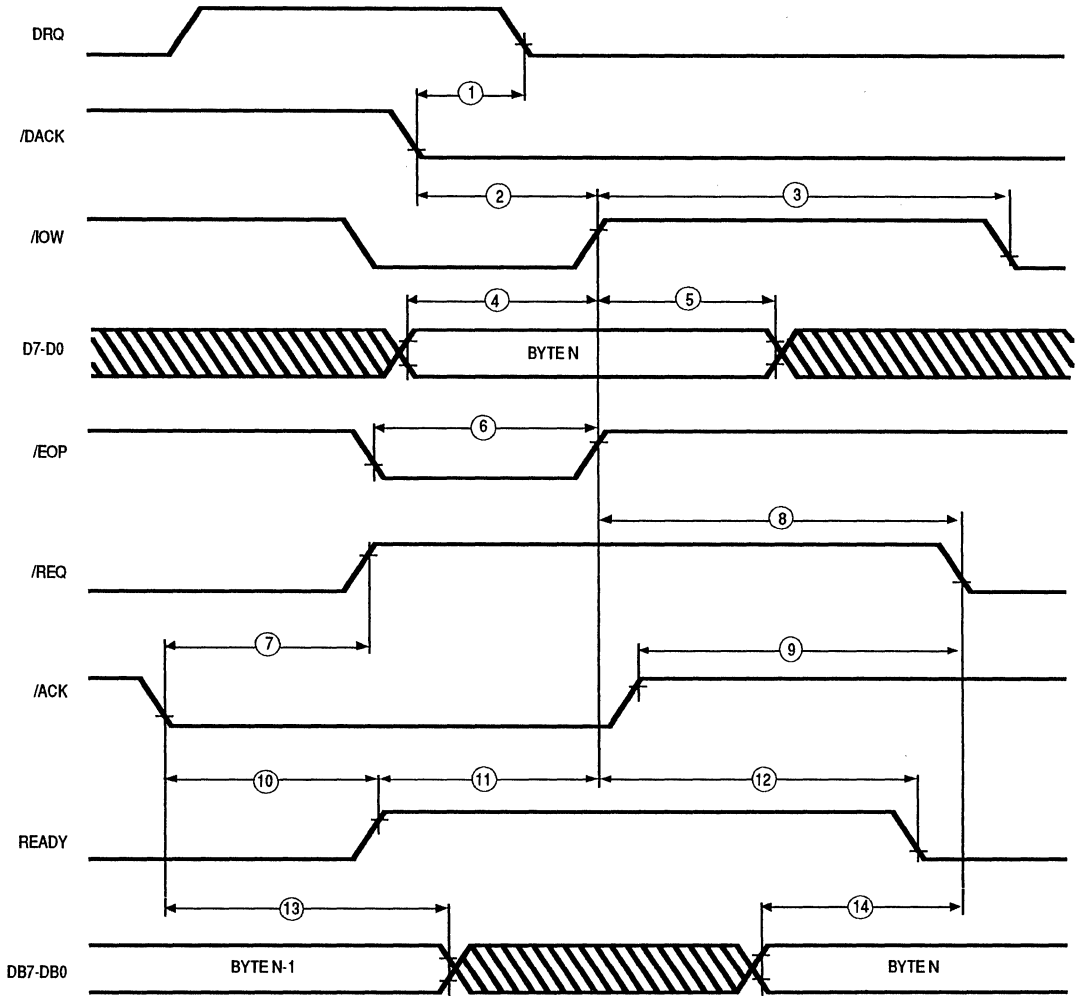


Figure 51. DMA Write (Block Mode) Target Send

---

## AC CHARACTERISTICS

### DMA Write (Block Mode) Target Send Table

---

Name	Description	Min	Max	Units
1	DRQ False from /DACK True		60	ns
2	Write Enable Width*	50		ns
3	Write Recovery Time	120		ns
4	Data Setup to End of Write Enable*	50		ns
5	Data Hold Time from End of /IOW	25		ns
6	Width of /EOP Pulse [1]	50		ns
7	/ACK True to /REQ False		80	ns
8	/REQ from End of /IOW (/ACK False)		90	ns
9	/REQ from End of ACK (/IOW False)		100	ns
10	/ACK True to READY True		70	ns
11	READY True to /IOW False	70		ns
12	/IOW False to READY False		70	ns
13	DATA Hold from /ACK True	40		ns
14	Data Setup to /REQ True	55		ns

---

**Notes:**

[1] /EOP, /IOW, and /DACK must be concurrently true for at least T6 for proper recognition of the /EOP pulse.

\*Read enable is the occurrence of both /IOR and /DACK.



# AC CHARACTERISTICS

## DMA Read (Block Mode) Target Receive Timing Diagram

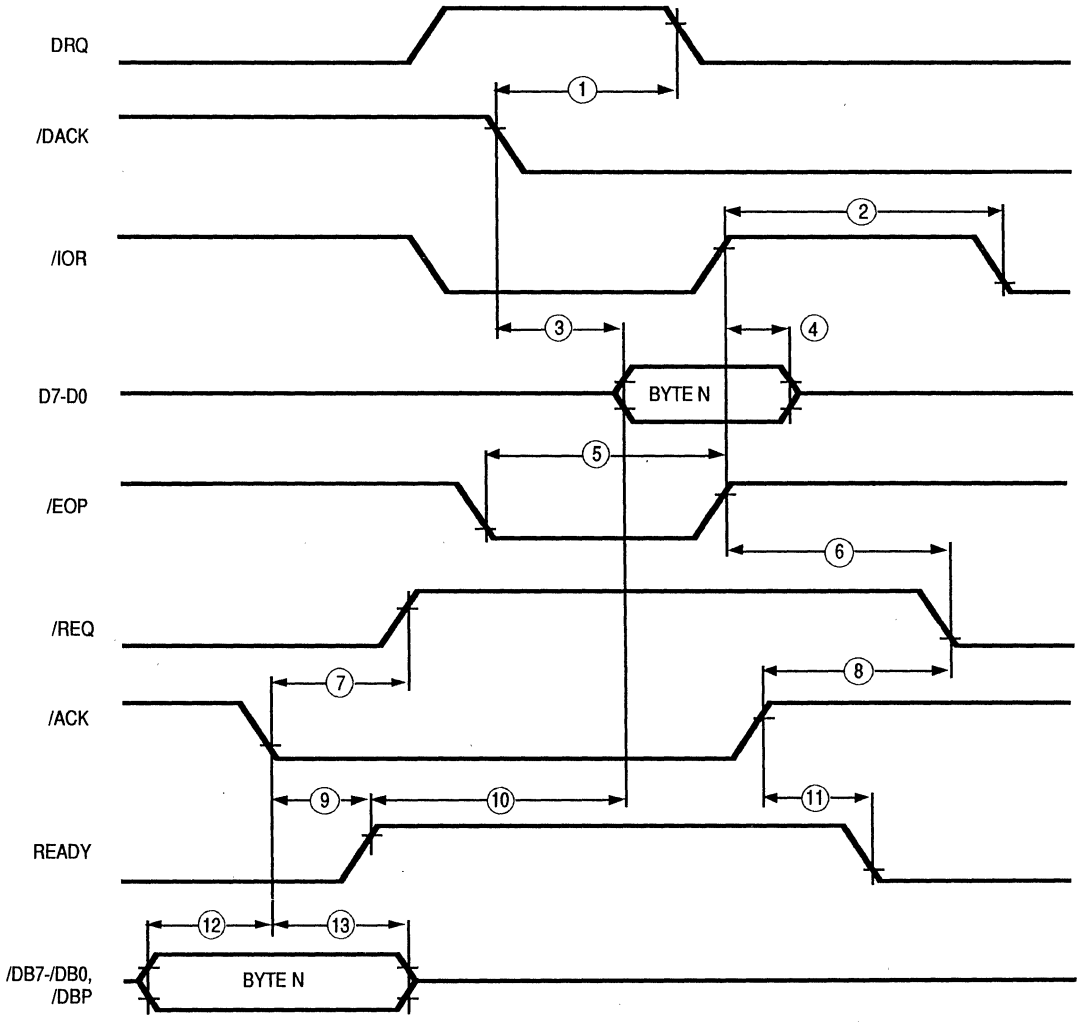


Figure 52. DMA Read (Block Mode) Target Receive

## AC CHARACTERISTICS

### DMA Read (Block Mode) Target Receive Table

Name	Description	Min	Max	Units
1	DRQ False from /DACK True		60	ns
2	/IOR Recovery Time	120		ns
3	Data Access Time from Read Enable*		70	ns
4	Data Hold Time from End of /IOR	10		
5	Width of /EOP Pulse [1]	50		ns
6	/IOR False to /REQ True (/ACK False)		90	ns
7	/ACK True to /REQ False		80	ns
8	/ACK False to /REQ True (/IOR False)		100	ns
9	/ACK True to READY True		70	ns
10	READY True to Valid Data		50	ns
11	/IOR False to READY False		70	ns
12	DATA Setup time to /ACK	20		
13	DATA Hold Time from /ACK	30		

#### Notes:

[1] /EOP, /IOR, and /DACK must be concurrently true for at least T5 for proper recognition of the /EOP pulse.

\*Read enable is the occurrence of both /IOR and /DACK.

## AC CHARACTERISTICS

### Reset Timing Diagram

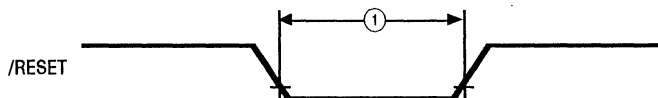


Figure 53. Reset

## AC CHARACTERISTICS

### Reset Table

No	Description	Min	Max	Units
1	Minimum Width of /RESET	100		ns

**AC CHARACTERISTICS**  
Arbitration Timing Diagram

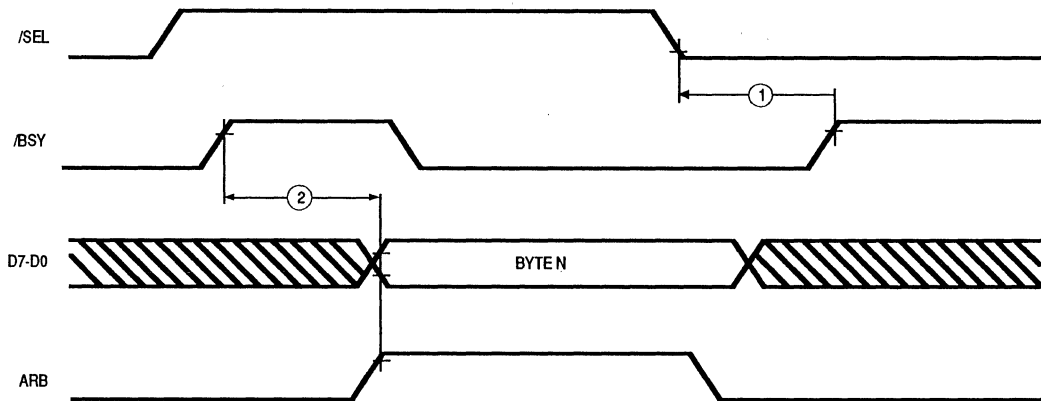


Figure 54. Arbitration

**AC CHARACTERISTICS**  
Arbitration Table

Name	Description	Min	Max	Units
1	Bus Clear from SEL True		600	ns
2	ARBITRATE Start from BSY False	1200	2200	ns



## Z85C80

### SCSCI SERIAL COMMUNICATIONS AND SMALL COMPUTER INTERFACE

#### FEATURES

- Low power CMOS
- Two independent, 0 to 2.5M bit/second, full-duplex channels, each with a separate crystal oscillator, baud rate generator, and Digital Phase-Locked Loop for clock recovery.
- Multi-protocol operation under program control; programmable for NRZ, NRZI, or FM data encoding.
- Asynchronous mode with five to eight bits and one, one and one-half, or two stop bits per character, programmable clock factor, break detection and generation; parity, overrun, and framing error detection.
- Synchronous mode with internal or external character synchronization on one or two synchronous characters and CRC generation and checking with CRC-16 or CRC-CCITT preset to either 1s or 0s.
- SDLC/HDLC mode with comprehensive frame-level control, automatic zero insertion and deletion, I-field residue handling, abort generation and detection, CRC generation and checking, and SDLC Loop mode operation.
- Local Loopback and Auto Echo modes.
- Supports T1 digital trunk.
- Enhanced DMA support
  - 10 X 19-bit status FIFO
  - 14-bit byte counter
- Arbitration Support
- DMA or Programmed I/O Data Transfers
- Supports Normal DMA
- Memory or I/O Mapped CPU Interface.
- Asynchronous Interface, Supports 3 MB/s
- Direct SCSI Bus Interface with On-Board 48 mA Drivers
- Supports Target and Initiator Roles

#### GENERAL DESCRIPTION

The Z85C80 CMOS SCSCI is an industry standard 85C30 dual channel Serial Communication Controller (SCC) and an industry standard 53C80 Small Computer System Interface (SCSI) integrated into one monolithic Integrated Circuit. The internal SCC and SCSI share the 8-bit data bus (D7 through D0) and read and write inputs (/RD and /WR).

**Note: All Signals with a preceding front slash, "/", are active Low, e.g.: B//W (WORD is active Low); /B/W (BYTE is active Low, only); /N//S (NORMAL and SYSTEM are both active Low).**

The Z85C80 is offered in a 68-pin PLCC package. With a few exceptions, all of the internal SCC and SCSI signals are connected to the outside pins.

---

## GENERAL DESCRIPTION (Continued)

The exceptions are:

- IEI input to SCC is internally connected to VDD.
- IEO output from SCC is not internally connected (N/C).
- READY output from SCSI is not internally connected (N/C).
- /SYNCB output from the SCC is not internally connected (N/C).
- /TRXCA and /CTSA inputs to the SCC are internally connected.
- /TRXCB and /CTSB inputs to the SCC are internally connected.

The internal SCC is a dual channel, multi-protocol data communications peripheral that easily interfaces to CPUs with non-multiplexed address/data buses. The programming flexibility of the internal registers allows the SCC to be configured to satisfy a wide variety of serial communications applications. The many on-chip features such as baud rate generators, digital phase locked loops, and crystal oscillators dramatically reduce the need for external logic. Additional features, including a 10x19-bit status FIFO and 14-bit byte counter, were added to support high speed SDLC transfers using DMA controllers.

The SCC handles asynchronous formats, synchronous byte-oriented protocols such as IBM Bisync, and synchronous bit-oriented protocols such as HDLC and IBM

SDLC. The internal SCC can generate and check CRC codes in any synchronous mode and can be programmed to check data integrity in various modes. It also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls are used for general-purpose I/O. The daisy-chain interrupt hierarchy is also supported and is standard for Zilog peripheral components.

The internal SCSI is designed to implement the SCSI protocol as defined by the ANSI X3.131-1986 standard, and is fully compatible with the industry standard 53C80. It is capable of operating both as a target and as an initiator. Special high current open-drain outputs enable it to directly interface to, and drive, the SCSI bus. The internal SCSI has the necessary interface hook-ups so the system CPU can communicate with it like with any other peripheral device. The CPU can read from, or write to, the SCSI registers which are addressed as standard or memory-mapped I/Os.

The Internal SCSI increases the system performance by minimizing the CPU intervention in DMA operations which the SCSI controls. The CPU is interrupted by the SCSI when it detects a bus condition that requires attention. It also supports arbitration and reselection. The internal SCSI has the proper hand-shake signals to support normal DMA operations with most DMA controllers available.

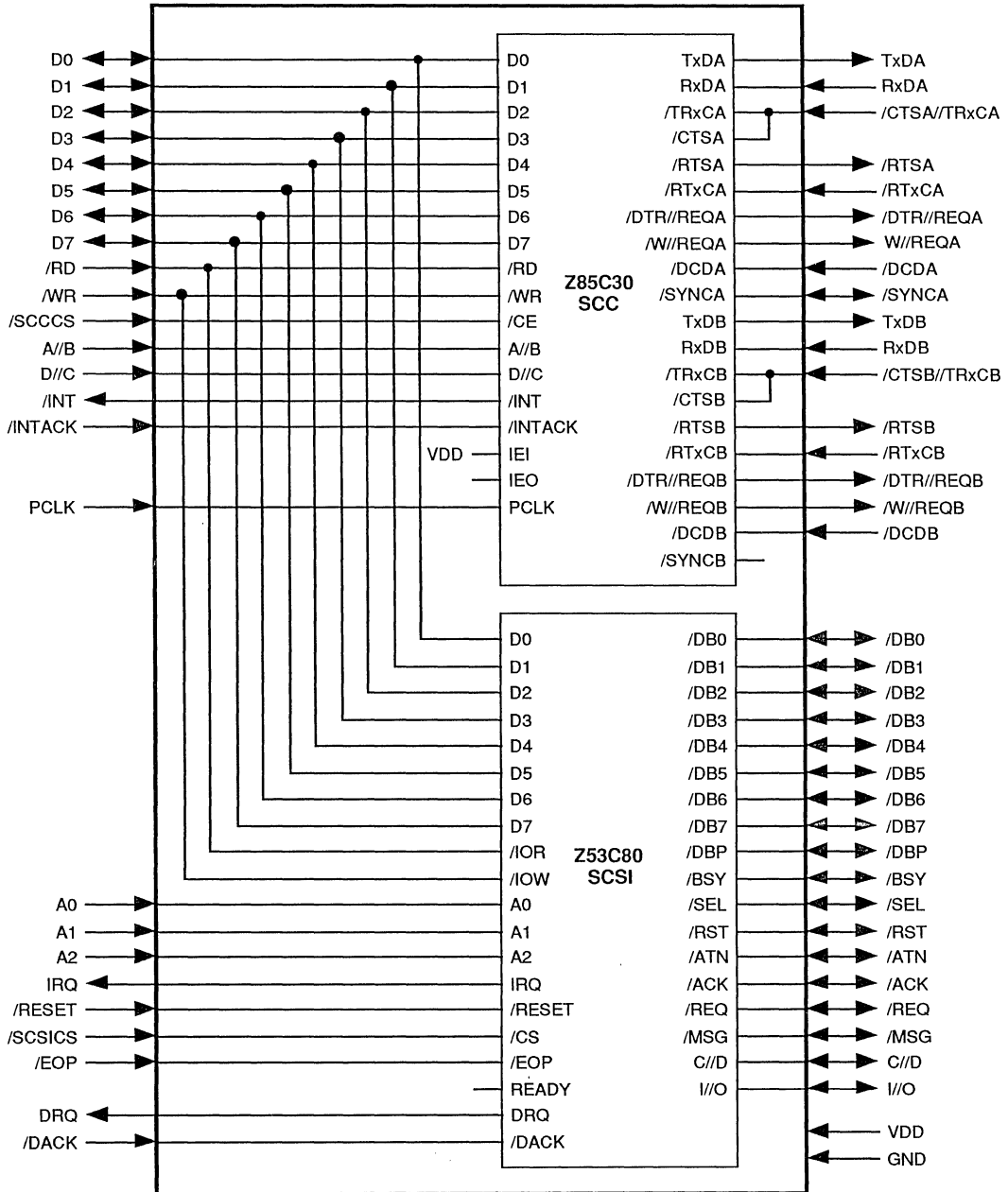
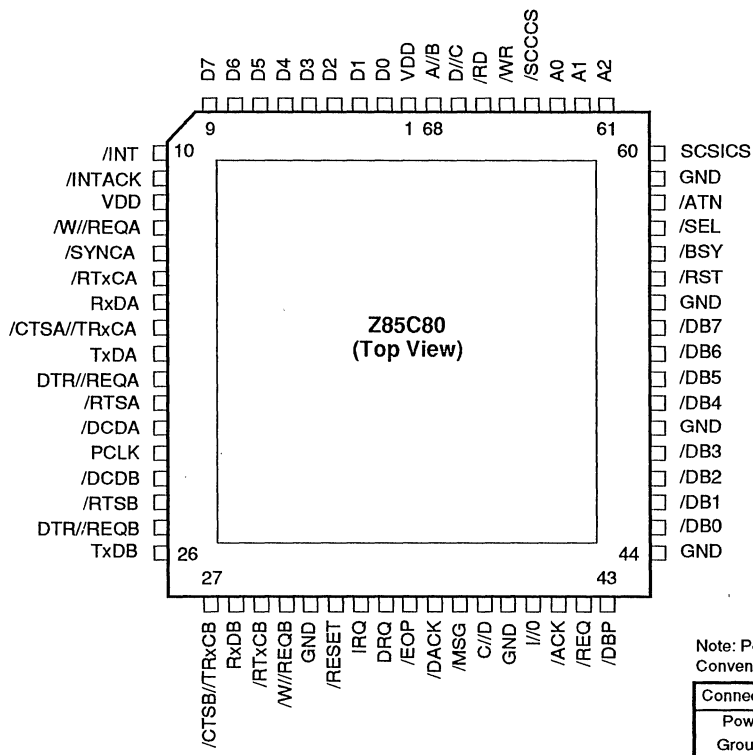


Figure 1. Z85C80 SCSI Block Diagram



Note: Power connections follow Conventional descriptions below

Connection	Circuit	Device
Power	V <sub>CC</sub>	V <sub>DD</sub>
Ground	GND	V <sub>SS</sub>

Figure 2. 68-Pin PLCC Pin Diagram

### PIN DESCRIPTION

Signal	Pin	Type	Description
A0	63	I	<b>SCSI Address Line Bit 0 (SCSI)</b> . Address lines are used with /SCSICS, /RD, or /WR to address all internal registers.
A1	62	I	<b>SCSI Address Line Bit 1 (SCSI)</b> . Address lines are used with /SCSICS, /RD, or /WR to address all internal registers.
A2	61	I	<b>SCSI Address Line Bit 2 (SCSI)</b> . Address lines are used with /SCSICS, /RD, or /WR to address all internal registers.
A/B	68	I	<b>Channel A/Channel B (SCC)</b> . This signal selects the SCC channel in which the read or write operation occurs.
/ACK	41	I/O	<b>Acknowledge</b> (open-drain, active low, SCSI). Driven by an Initiator, /ACK indicates an acknowledgement for a /REQ//ACK data-transfer handshake. In the Target role, /ACK is received as a response to the /REQ signal.
/ATN	58	I/O	<b>Attention</b> (open-drain, active low, SCSI). Driven by an Initiator, received by the Target. /ATN indicates an Attention condition.

## PIN DESCRIPTION (Continued)

Signal	Pin	Type	Description
/BSY	56	I/O	<b>Busy</b> (open-drain, active low, SCSI). This signal indicates that the SCSI bus is being used and can be driven by both the Initiator and the Target device.
C//D	38	I/O	<b>Control/Data</b> (open-drain, SCSI). Driven by the Target and received by the Initiator. C//D indicates whether Control or Data information is on the Data Bus. True indicates control.
/CTS <sub>A</sub> //TRXCA	17	I	<b>Clear To Send for channel A; Transmit/Receive Clock for channel A</b> (active low, SCC). This pin is internally connected to SCC's A Channel /CTS and /TRXC. Receive clock or the transmit clock is supplied via this pin to the SCC's A channel. When programmed as Auto Enables, a low on this pin enables the A-channel transmitter.
/CTSB//TRXCB	27	I	<b>Clear To Send for channel B/Transmit/Receive Clock for channel B</b> (active low, SCC). This pin is internally connected to SCC's B-channel /CTS and /TRXC. Receive clock or the transmit clock is supplied via this pin to the SCC's B channel. When programmed as Auto Enables, a low on this pin enables the B-channel transmitter.
D0	2	I/O	<b>Data bus bit 0</b> (tri-state, active high, SCC and SCSI). This is the Least Significant Bit of the bus. Data bus lines carry data and commands to and from the SCSCI.
D1	3	I/O	<b>Data bus bit 1</b> (tri-state, active high, SCC and SCSI). Data bus lines carry data and commands to and from the SCSCI.
D2	4	I/O	<b>Data bus bit 2</b> (tri-state, active high, SCC and SCSI). Data bus lines carry data and commands to and from the SCSCI.
D3	5	I/O	<b>Data bus bit 3</b> (tri-state, active high, SCC and SCSI). Data bus lines carry data and commands to and from the SCSCI.
D4	6	I/O	<b>Data bus bit 4</b> (tri-state, active high, SCC and SCSI). Data bus lines carry data and commands to and from the SCSCI.
D5	7	I/O	<b>Data bus bit 5</b> (tri-state, active high, SCC and SCSI). Data bus lines carry data and commands to and from the SCSCI.
D6	8	I/O	<b>Data bus bit 6</b> (tri-state, active high, SCC and SCSI). Data bus lines carry data and commands to and from the SCSCI.
D7	9	I/O	<b>Data bus bit 7</b> (tri-state, active high, SCC and SCSI). This is the most significant bit of the bus. Data bus lines carry data and commands to and from the SCSCI.
/DACK	36	I	<b>DMA Acknowledge</b> (active low, SCSI). /DACK resets DRQ and selects the data register for input or output data transfers. /DACK is used by DMA controller instead of /SCSICS.
/DB0	45	I/O	<b>SCSI Data Bus bit 0</b> (open-drain, active low, SCSI). Least significant bit in the SCSI data bus.
/DB1	46	I/O	<b>SCSI Data Bus bit 1</b> (open-drain, active low, SCSI).
/DB2	47	I/O	<b>SCSI Data Bus bit 2</b> (open-drain, active low, SCSI).



---

**PIN DESCRIPTION** (Continued)

Signal	Pin	Type	Description
/DB3	48	I/O	<b>SCSI Data Bus bit 3</b> (open-drain, active low, SCSI).
/DB4	50	I/O	<b>SCSI Data Bus bit 4</b> (open-drain, active low, SCSI).
/DB5	51	I/O	<b>SCSI Data Bus bit 5</b> (open-drain, active low, SCSI).
/DB6	52	I/O	<b>SCSI Data Bus bit 6</b> (open-drain, active low, SCSI).
/DB7	53	I/O	<b>SCSI Data Bus bit 7</b> (open-drain, active low, SCSI). This is the most significant bit in the SCSI data bus.
/DBP	43	I/O	<b>SCSI Data Bus Parity bit</b> (open-drain, active low, SCSI). Data parity is odd. Parity is always generated and optionally checked. Parity is not valid during Arbitration.
D//C	67	I	<b>Data/Control Select (SCC)</b> . This signal defines the type of information transferred to and from the SCC.
/DCDA	21	I	<b>Data Carrier Detect for A channel</b> (active low, SCC). This pin functions as receive enable if it is programmed for Auto Enable; otherwise, it may be used as general-purpose input pin. The SCC detects pulses on this pin and can interrupt the CPU on both logic level transactions.
/DCDB	23	I	<b>Data Carrier Detect for B channel</b> (active low, SCC). This pin functions as receive enable if it is programmed for Auto Enable; otherwise, it may be used as general-purpose input pin. The SCC detects pulses on this pin and can interrupt the CPU on both logic level transactions.
DRQ	34	0	<b>DMA Request</b> (active high, SCSI). DRQ indicates that the data register is ready to be read or written. DRQ is asserted only if DMA mode is set in the Command Register. DRQ is cleared by /DACK.
/DTR//REQA	19	0	<b>Data Terminal Ready/Request for channel A</b> (active low, SCC). This output follows the state programmed into the DTR bit. It can also be used as general-purpose output or as Request line for a DMA controller.
/DTR//REQB	25	0	<b>Data Terminal Ready/Request for channel B</b> (active low, SCC). This output follows the state programmed into the DTR bit. It can also be used as general-purpose output or as Request line for a DMA controller.
/EOP	35	I	<b>End of process</b> (active low, SCSI). EOP is used to terminate a DMA transfer. If asserted during a DMA cycle, the current byte will be transferred, but no additional bytes will be requested.
GND	31, 39,44, 45,54, 59	S	<b>Ground supply</b> (SCC and SCSI).
/INT	10	0	<b>SCC Interrupt Request</b> (open-drain, active low, SCC). This signal is activated when the SCC requests an interrupt.

---

---

## PIN DESCRIPTION (Continued)

Signal	Pin	Type	Description
/INTACK	11	I	<b>Interrupt acknowledge</b> (active low, SCC). This signal indicates an active Interrupt Acknowledge cycle. /INTACK is latched by the rising edge of PCLK.
I/O	40	I/O	<b>Input/Output</b> (open-drain, SCSI). I/O is a signal driven by a Target which controls the direction of data movement on the SCSI bus. TRUE indicates input to the Initiator. This signal is also used to distinguish between Selection and Reselection phases.
IRQ	33	0	<b>SCSI Interrupt Request</b> (active high, SCSI). This signal alerts a microprocessor of an error condition or an event completion.
/MSG	37	I/O	<b>Message</b> (open-drain, SCSI). This signal is driven by the Target during the Message phase. This signal is received by the Initiator.
PCLK	22	I	<b>Clock</b> (SCC). This is the master SCC clock used to synchronize internal signals. PCLK is not required to have any phase relationship with the master system clock.
/RD	66	I	<b>Read</b> (active low, SCC and SCSI). When the SCC is selected, it enables the SCC's bus drivers. When the SCSI is selected, it is used in conjunction with /SCSICS and A2-A0 to read an internal register. It also selects the Input Data Register in SCSI when used with /DACK.
/REQ	42	I/O	<b>Request</b> (open-drain, active low, SCSI). Driven by a Target and received by the Initiator, this signal indicates a request for a /REQ/ /ACK data-transfer handshake.
/RESET	32	I	<b>SCSI Reset</b> (active low, SCSI). This signal clears all registers in the SCSI. It has no effect upon the SCSI /RST signal.
/RST	55	I/O	<b>SCSI bus Reset</b> (open-drain, active low, SCSI). This signal indicates a SCSI bus Reset condition.
/RTSA	20	0	<b>Request To Send for channel A</b> (active low, SCC). When the RTS bit in Write Register 5 is set, the /RTS signal goes low. When the RTS bit is reset in the Asynchronous mode and Auto Enable is on, the signal goes high after the transmitter is empty. In Synchronous mode or in Asynchronous mode with Auto Enable off, the /RTS pin strictly follows the state of the RTS bit. This pin can be used as a general-purpose output.
/RTSB	24	0	<b>Request To Send for channel B</b> (active low, SCC). When the RTS bit in Write Register 5 is set, the /RTS signal goes low. When the RTS bit is reset in the Asynchronous mode and Auto Enable is on, the signal goes high after the transmitter is empty. In Synchronous mode or in Asynchronous mode with Auto Enable off, the /RTS pin strictly follows the state of the RTS bit. This pin can be used as a general-purpose output.

---

## PIN DESCRIPTION (Continued)

Signal	Pin	Type	Description
/RTXCA	15	I	<b>Receive/Transmit Clock for channel A</b> (active low, SCC). This pin can be programmed in several modes of operation. It may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the digital phase-locked loop. This pin can also be programmed for use with the /SYNCA pin as a crystal oscillator. The receive clock can be 1, 16, 32, or 64 times the data rate in Asynchronous modes.
/RTXCB	29	I	<b>Receive/Transmit Clock for channel B</b> (active low, SCC). This pin can be programmed in several modes of operation. It may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the digital phase-locked loop. The receive clock can be 1, 16, 32, or 64 times the data rate in Asynchronous modes.
RXDA	16	I	<b>Receive Data for channel A</b> (active high, SCC). This input signal receives serial data.
RXDB	28	I	<b>Receive Data for channel B</b> (active high, SCC). This input signal receives serial data.
/SCCCS	64	I	<b>SCC Chip Select</b> (active low, SCSI). This signal selects SCC for a read or write operation.
/SCSICS	60	I	<b>SCSI Chip Select</b> (active low, SCSI). This signal, in conjunction with /RD or /WR, enables the internal register selected by A2-A0, to be read from or written to.
/SEL	57	I/O	<b>Select</b> (open-drain, active low, SCSI). This signal is used by an Initiator to select a Target, or by a Target to reselect an Initiator.
/SYNCA	14	I/O	<b>Synchronization for channel A</b> (active low, SCC). This pin can act as input, output, or part of the crystal oscillator circuit.
TXDA	18	O	<b>Transmit Data for channel A</b> (active high, SCC). This output signal transmits serial data at standard TTL levels.
TXDB	26	O	<b>Transmit Data for channel B</b> (active high, SCC). This output signal transmits serial data at standard TTL levels.
VDD	1,12	S	<b>VDD supply</b> (SCC and SCSI).
/WR	65	I	<b>Write</b> (active low, SCC and SCSI). When the SCC is selected, this signal indicates a write operation. The coincidence of /RD and /WR is interpreted as a reset. When the SCSI is selected, it is used in conjunction with /SCSICS and A2-A0 to write an internal register. It also selects the Output Data Register in SCSI, when used with /DACK.
/W//REQA	13	O	<b>Wait/Request for channel A</b> (open-drain when programmed for a Wait function, driven high or low when programmed for a Request function, SCC). This dual purpose output may be programmed as request line for a DMA controller or as a Wait line to synchronize the CPU to the SCC data rate. The reset state is Wait.

---

## PIN DESCRIPTION (Continued)

Signal	Pin	Type	Description
/W//REQB	30	0	<b>Wait/Request for channel B</b> (open-drain when programmed for a Wait function, driven high or low when programmed for a Request function, SCC). This dual purpose output may be programmed as request line for a DMA controller or as a Wait line to synchronize the CPU to the SCC data rate. The reset state is Wait.

---

---

## FUNCTIONAL DESCRIPTION

The Z85C80 consists of an industry standard Z85C30 Serial Communication Controller (SCC) and an industry standard Z53C80 Small Computer System Interface (SCSI), sharing the data bus and read and write signals. With the exception of the following special configurations, the internal SCC and SCSI can be used as standard devices.

### SCC Configuration

- IEI (Interrupt Enable In) is hardwired to VDD. Thus no external interrupt daisy-chain can be used.
- IEO (Interrupt Enable Out) is not bonded out. Since no daisy-chain interrupt is used, this pin is left unbonded.
- /TRXC and /CTS are connected together in each of the two channels to form /CTS//TRXC. In this configuration, the pin in each channel is used as receive or transmit clock input.
- /SYNCB (channel B Synchronization) is not bonded.

### SCSI Configuration

- Data lines of the SCSI are shared with the SCC's data bus (D7 through D0 on both devices). Care must be taken not to cause bus contention by inappropriately selecting the two internal devices using their respective /CS.
- /IOR of SCSI connected to /RD of SCC to generate Z85C80's /RD pin.
- /IOW of SCSI is connected to /WR of SCC to generate Z85C80's /WR pin.
- READY (Ready) is not bonded out. READY is normally used to control the speed of Block Mode DMA transfers. It goes active to indicate the SCSI is ready to send/receive data.

### SCC Functional Description

The functional capabilities of the SCC are described from two different points of view: as a data communications device, it transmits and receives data in a wide variety of data communications protocols; as a microprocessor peripheral, the SCC offers valuable features such as vectored interrupts, polling, and simple handshake capability.

**Data Communications Capabilities.** The SCC provides two independent full-duplex channels programmable for use in any common Asynchronous or Synchronous data communication protocol. Figure 3 and the following description briefly detail these protocols.

**Asynchronous Modes.** Transmission and reception can be accomplished independently on each channel with five to eight bits per character, plus optional even or odd parity. The transmitters can supply one, one and one half, or two stop bits per character and can provide a break output at any time. The receiver break-detection logic interrupts the CPU both at the start and at the end of a received break. Reception is protected from spikes by a transient spike-rejection mechanism that checks the signal one-half a bit time after a Low level is detected on the receive data input (RxDA or RxDB in Figure 1). If the Low does not persist (as in the case of a transient), the character assembly process does not start.

Framing errors and overrun errors are detected and buffered together with the partial character on which they occur. Vectored interrupts allow fast servicing or error conditions using dedicated routines. Furthermore, a built-in checking process avoids the interpretation of a framing error as a new start bit: a framing error results in the addition of one-half a bit time to the point at which the search for the next start bit begins.

## FUNCTIONAL DESCRIPTION (Continued)

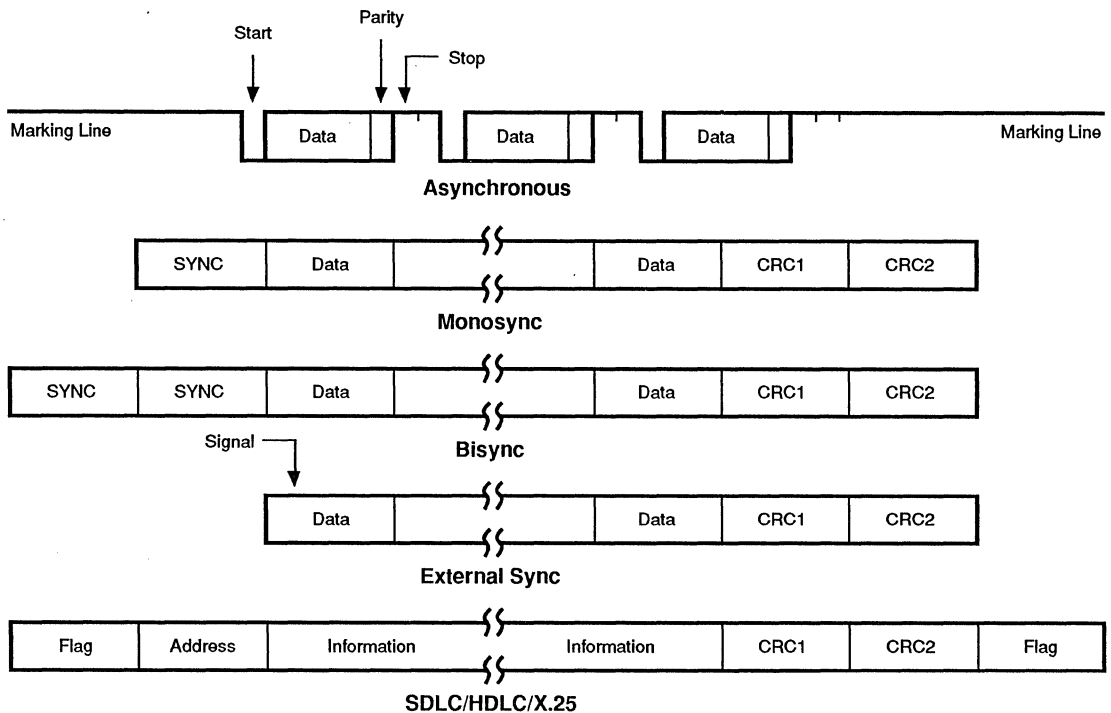


Figure 3. Some SCC Protocols

The SCC does not require symmetric transmit and receive clock signals - a feature allowing the use of a wide variety of clock sources. The transmitter and receiver can handle data at a rate of 1, 1/16, 1/32, or 1/64 of the clock rate supplied to the receive and transmit clock inputs. In Asynchronous modes, the /SYNC pin may be programmed as an input used for functions such as monitoring a ring indicator.

**Synchronous Modes.** The SCC supports both byte-oriented and bit-oriented synchronous communication. Synchronous byte-oriented protocols can be handled in several modes, allowing character synchronization with a 6-bit or 8-bit synchronous character (Monosync), any 12-bit synchronization pattern (Bisync), or with an external synchronous signal. Leading sync characters can be removed without interrupting the CPU.

Five- or 7-bit synchronous characters are detected with 8- or 16-bit patterns in the SCC by overlapping the larger pattern across multiple incoming synchronous characters as shown in Figure 4.

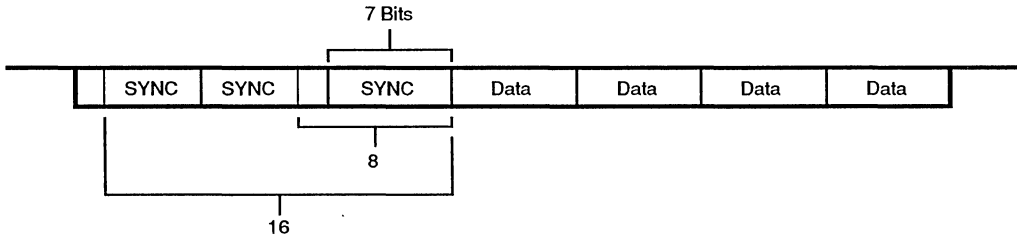
CRC checking for Synchronous byte-oriented modes is delayed by one character time so that the CPU may disable CRC checking on specific characters. This permits the implementation of protocols such as IBM Bisync.

Both CRC-16 ( $X^{16} + X^{15} + X^{12} + 1$ ) and CCITT ( $X^{16} + X^{15} + X^{12} + 1$ ) error checking polynomials are supported. Either polynomial may be selected in all Synchronous modes. Users may preset the CRC generator and checker to all 1s or all 0s. The SCC also provides a feature that automatically transmits CRC data when no other data is available for transmission. This allows for high speed transmission under DMA control, with no need for CPU intervention at the end of a message. When there is no data or CRC to send in Synchronous modes, the transmitter inserts 6-, 8-, or 16-bit synchronous characters, regardless of the programmed character length.

The SCC supports Synchronous bit-oriented protocols, such as SDLC and HDLC, by performing automatic flag sending, zero insertion, and CRC generation. A special command can be used to abort a frame in transmission. At

the end of a message, the SCC automatically transmits the CRC and trailing flag when the transmitter underruns. The transmitter may also be programmed to send an idle line

consisting of continuous flag characters or a steady marking condition.



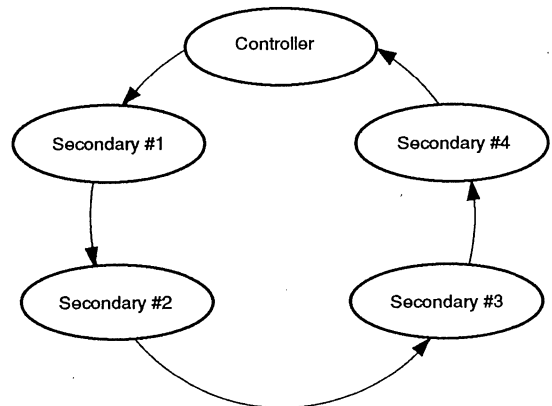
**Figure 4. Detecting 5 - or 7-Bit Synchronous Characters**

If a transmit underrun occurs in the middle of a message, an external/status interrupt warns the CPU of this status change so that an abort may be issued. The SCC may also be programmed to send an abort itself in case of an underrun, relieving the CPU of this task. One to eight bits per character can be sent, allowing reception of a message with no prior information about the character structure in the information field of a frame.

The receiver automatically acquires synchronization on the leading flag of a frame in SDLC or HDLC and provides a synchronization signal on the /SYNC pin (an interrupt can also be programmed). The receiver can be programmed to search for frames addressed by a single byte (or four bits within a byte) of a user-selected address or to a global broadcast address. In this mode, frames not matching either the user-selected or broadcast address are ignored. The number of address bytes can be extended under software control. For receiving data, an interrupt on the first received character, or an interrupt on every character, or on special condition only (end-of-frame) can be selected. The receiver automatically deletes all 0s inserted by the transmitter during character assembly. CRC is also calculated and is automatically checked to validate frame transmission. At the end of transmission, the status of a received frame is available in the status registers. In SDLC mode, the SCC must be programmed to use the SDLC CRC polynomial, but the generator and checker may be preset to all 1s or all 0s. The CRC is inverted before transmission and the receiver checks against the bit pattern 0001110100001111.

tion, for example, the SCC can interrupt the CPU when the first character of a message is received. The CPU then enables the DMA to transfer the message to memory. The SCC then issues an end-of-frame interrupt and the CPU can check the status of the received message. Thus, the CPU is freed for other service while the message is being received. The CPU may also enable the DMA first and have the SCC interrupt only on end-of-frame. This procedure allows all data to be transferred via the DMA.

**SDLC Loop Mode.** The SCC supports SDLC Loop mode in addition to normal SDLC. In an SDLC Loop, there is a primary controller station that manages the message traffic flow on the loop and any number of secondary stations. In SDLC Loop mode, the SCC performs the functions of a secondary station while an SCC operating in regular SDLC mode can act as a controller (Figure 5).



**Figure 5. An SDLC Loop**

NRZ, NRZI or FM coding may be used in any 1x mode. The parity options available in Asynchronous modes are available in Synchronous modes.

The SCC can be conveniently used under DMA control to provide high speed reception or transmission. In recep-

---

## FUNCTIONAL DESCRIPTION (Continued)

A secondary station in an SDLC Loop is always listening to the messages being sent around the loop, and in fact must pass these messages to the rest of the loop by re-transmitting them with a one-bit-time delay. The secondary station can place its own message on the loop only at specific times. The controller signals that secondary stations may transmit messages by sending a special character, called an EOP (End of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages, this bit pattern is unique and easily recognized.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary 1 of the EOP to a 0 before transmission. This has the effect of turning the EOP into a flag sequence. The secondary station now places its message on the loop and terminates the message with an EOP. Any secondary stations further down the loop with messages to transmit can then append their messages to the message of the first secondary station by the same process. Any secondary stations without messages to send merely echo the incoming messages and are prohibited from placing messages on the loop (except upon recognizing an EOP).

SDLC Loop mode is a programmable option in the SCC. NRZ, NRZI, and FM coding may all be used in SDLC Loop mode.

**Baud Rate Generator.** Each channel in the SCC contains a programmable baud rate generator. Each generator consists of two 8-bit time constant registers that form a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output producing a square wave. On startup, the flip-flop on the output is set in a High state, the value in the time constant register is loaded into the counter, and the counter starts counting down. The output of the baud rate generator toggles upon reaching 0, the value in the time constant register is loaded into the counter, and the process is repeated. The time constant may be changed at any time, but the new value does not take effect until the next load of the counter.

The output of the baud rate generator may be used as either the transmit clock, the receive clock, or both. It can also drive the Digital Phase-Locked Loop (see next section).

The following formula relates the time constant to the baud rate where PCLK or /RTxC is the baud rate generator input frequency in Hz. The clock mode is 1, 16, 32, or 64 as selected in Write Register 4, bits D6 and D7. Synchronous operation modes should select 1 and Asynchronous should select 16, 32, or 64.

$$\text{Time Constant} = \frac{\text{PCLK or RTxC Frequency}}{2 \text{ (Baud Rate) (Clock Mode)}} - 2$$

**Digital Phase-Locked Loop.** The SCC contains a Digital Phase-Locked Loop (DPLL) to recover clock information from a data stream with NRZI or FM encoding. The DPLL is driven by a clock that is nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock may then be used as the SCC receive clock, the transmit clock, or both.

For NRZI encoding, the DPLL counts the 32x clock to create nominal bit times. As the 32x clock is counted, the DPLL is searching the incoming data stream for edges (either 1 to 0 or 0 to 1). Whenever an edge is detected, the DPLL makes a count adjustment (during the next counting cycle), producing a terminal count closer to the center of the bit cell.

For FM encoding, the DPLL still counts from 0 to 31, but with a cycle corresponding to two bit times. When the DPLL is locked, the clock edges in the data stream should occur between counts 15 and 16 and between counts 31 and 0. The DPLL looks for edges only during a time centered on the 15 or 16 counting transition.

The 32x clock for the DPLL can be programmed to come from either the /RTxC input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the SCC via the /TRxC pin (if this pin is not being used as an input).

**Data Encoding.** The SCC may be programmed to encode and decode the serial data in four different ways (Figure 6). In NRZ encoding, a 1 is represented by a High level and a 0 is represented by a Low level. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. In FM1 (more properly, bi-phase mark), a transition occurs at the beginning of every bit cell. A 1 is represented by an additional transition at the center of the bit cell and a 0 is represented by no additional transition at the center of the bit cell. In FM0 (bi-phase space), a transition occurs at the beginning of every bit cell. A 0 is represented by an additional transition at the center of the bit cell, and a 1 is represented by no additional transition at the center of the bit cell. In addition to these four methods, the SCC can be used to decode Manchester (bi-phase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is 0 to 1, the bit is a 0. If the transition is 1 to 0, the bit is a 1.

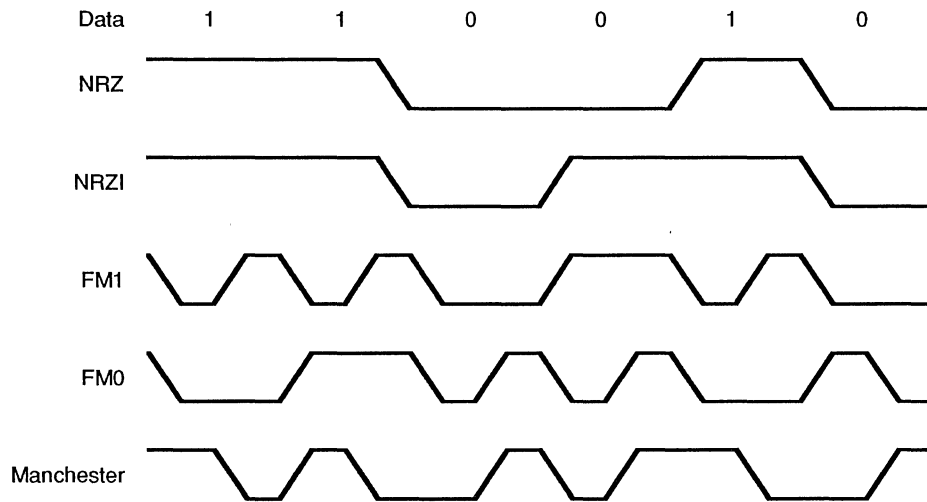


Figure 6. Data Encoding Methods

**Auto Echo and Local Loopback.** The SCC is capable of automatically echoing everything it receives. This feature is useful mainly in Asynchronous modes, but works in Synchronous and SDLC modes as well. In Auto Echo mode, TxD is RxD. Auto Echo mode can be used with NRZI or FM encoding with no additional delay, because the data stream is not decoded before re-transmission. In Auto Echo mode, the /CTS input is ignored as a transmitter enable (although transitions on this input can still cause interrupts if programmed to do so). In this mode, the transmitter is actually bypassed and the programmer is responsible for disabling transmitter interrupts and /W//REQ on transmit.

The SCC is also capable of local loopback. In this mode TxD is RxD, just as in Auto Echo mode. However, in Local Loopback mode, the internal transmit data is tied to the internal receive data and RxD is ignored (except to be echoed out via TxD). The /CTS and /DCD inputs are also ignored as transmit and receive enables. However, transitions on these inputs can still cause interrupts. Local Loopback works in Asynchronous, Synchronous and SDLC modes with NRZ, NRZI, or FM coding of the data stream.

**I/O Interface Capabilities.** The SCC offers the choice of Polling, Interrupt (vectored or nonvectored), and Block Transfer modes to transfer data, status, and control information to and from the CPU. The Block Transfer mode can be implemented under CPU or DMA control.

**Polling.** All interrupts are disabled. Three status registers in the SCC are automatically updated whenever any function is performed. For example, end-of-frame in SDLC mode sets a bit in one of these status registers. The idea behind polling is for the CPU to periodically read a status register until the register contents indicate the need for data to be transferred. Only one register needs to be read; depending on its contents, the CPU either writes data, reads data, or continues. Two bits in the register indicate the need for data transfer. An alternative is a poll of the Interrupt Pending register to determine the source of an interrupt. The status for both channels resides in one register.

## Interrupts

When an SCC responds to an Interrupt Acknowledge signal (/INTACK) from the CPU, an interrupt vector may be placed on the data bus. This vector is written in WR2 and may be read in RR2A or RR2B - Read Register 2, Channel A, or Channel B (Figures 9 and 10).

To speed interrupt response time, the SCC can modify three bits in this vector to indicate status. If the vector is read in Channel A, status is never included; if it is read in Channel B, status is always included.

Each of the six sources of interrupts in the SCC (Transmit, Receive, and External/Status interrupts in both channels)



---

## FUNCTIONAL DESCRIPTION (Continued)

has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt Under Service (IUS), and Interrupt Enable (IE). Operation of the IE bit is straightforward. If the IE bit is set for a given interrupt source, then that source can request interrupts. The exception is when the MIE (Master Interrupt Enable) bit in WR9 is reset and no interrupts may be requested. The IE bits are write only.

In the SCC, the IP bit signals a need for interrupt servicing. When an IP bit is 1, the /INT output is pulled Low, requesting an interrupt. In the SCC, if the IE bit is not set by enabling interrupts, then the IP for that source can never be set. The IP bits are readable in RR3A.

The IUS bits signal that an interrupt request is being serviced. There are three types of interrupts: Transmit, Receive, and External/Status. Each interrupt type is enabled under program control with Channel A having higher priority than Channel B, and with Receiver, Transmit, and External/Status interrupts prioritized in that order within each channel. When the Transmit interrupt is enabled, the CPU is interrupted when the transmit buffer becomes empty. (This implies that the transmitter must have had a data character written into it so that it can become empty.) When enabled, the receiver can interrupt the CPU in one of three ways.

1. Interrupt on First Receive Character or Special Receive Condition.
2. Interrupt on All Receive Characters or Special Receive Condition.
3. Interrupt on Special Receive Condition Only.

Interrupt on First Character or Special Condition and Interrupt on Special Condition Only are typically used with the Block Transfer mode. A Special Receive Condition is one of the following: receiver overrun, framing error in

Asynchronous mode, end-of-frame in SDLC mode and, optionally, a parity error. The Special Receive Condition interrupt is different from an ordinary receive character available interrupt only in the status placed in the vector during the Interrupt Acknowledge cycle. In Interrupt on First Receive Character, an interrupt can occur from Special Receive Conditions any time after the first receive character interrupt.

The main function of the External/Status interrupt is to monitor the signal transactions of the /CTC/TRXC, /DCD, and /SYNC pins; however, an External/Status interrupt is also caused by a Transmit Underrun condition, or a zero count in the baud rate generator, or by the detection, or a zero count in the baud rate generator, or by the detection of a Break (Asynchronous mode), Abort (SDLC mode) or EOP (SDLC Loop mode) sequence in the data stream. The interrupt caused by the Abort or EOP has a special feature allowing the SCC to interrupt when the Abort condition in external logic in SDLC mode. In SDLC Loop mode, this feature allows secondary stations to recognize the wishes of the primary station to regain control of the loop during a poll sequence.

**CPU/DMA Block Transfer.** The SCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the /W//REQ output in conjunction with the Wait/Request bits in WR1. The /W//REQ output can be defined under software control as a /W line in the CPU Block Transfer mode or as a /REQ line in the DMA Block Transfer mode.

To a DMA controller, the SCC /REQ output indicates that the SCC is ready to transfer data to or from memory. To the CPU, the /W line indicates that the SCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The /DTR//REQ line allows full-duplex operation under DMA control.

## ARCHITECTURE

The SCC internal structure includes two full-duplex channels, two baud rate generators, internal control and interrupt logic, and a bus interface to a nonmultiplexed bus. Associated with each channel are a number of read and write registers for mode control and status information, as well as logic necessary to interface to modems or other external devices (Figure 7).

The logic for both channels provides formats, synchronization, and validation for data transferred to and from the channel interface. The modem control inputs are monitored by the control logic under program control. All of the modem control signals are general-purpose in nature and can optionally be used for functions other than modem control.

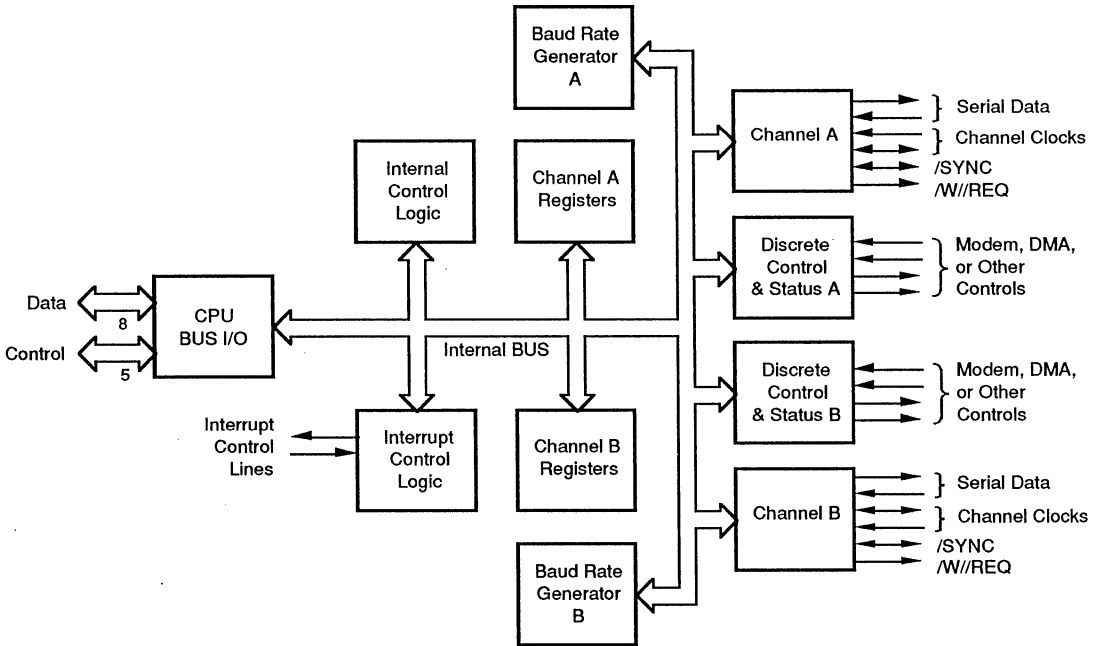


Figure 7. Block Diagram of SCC Architecture

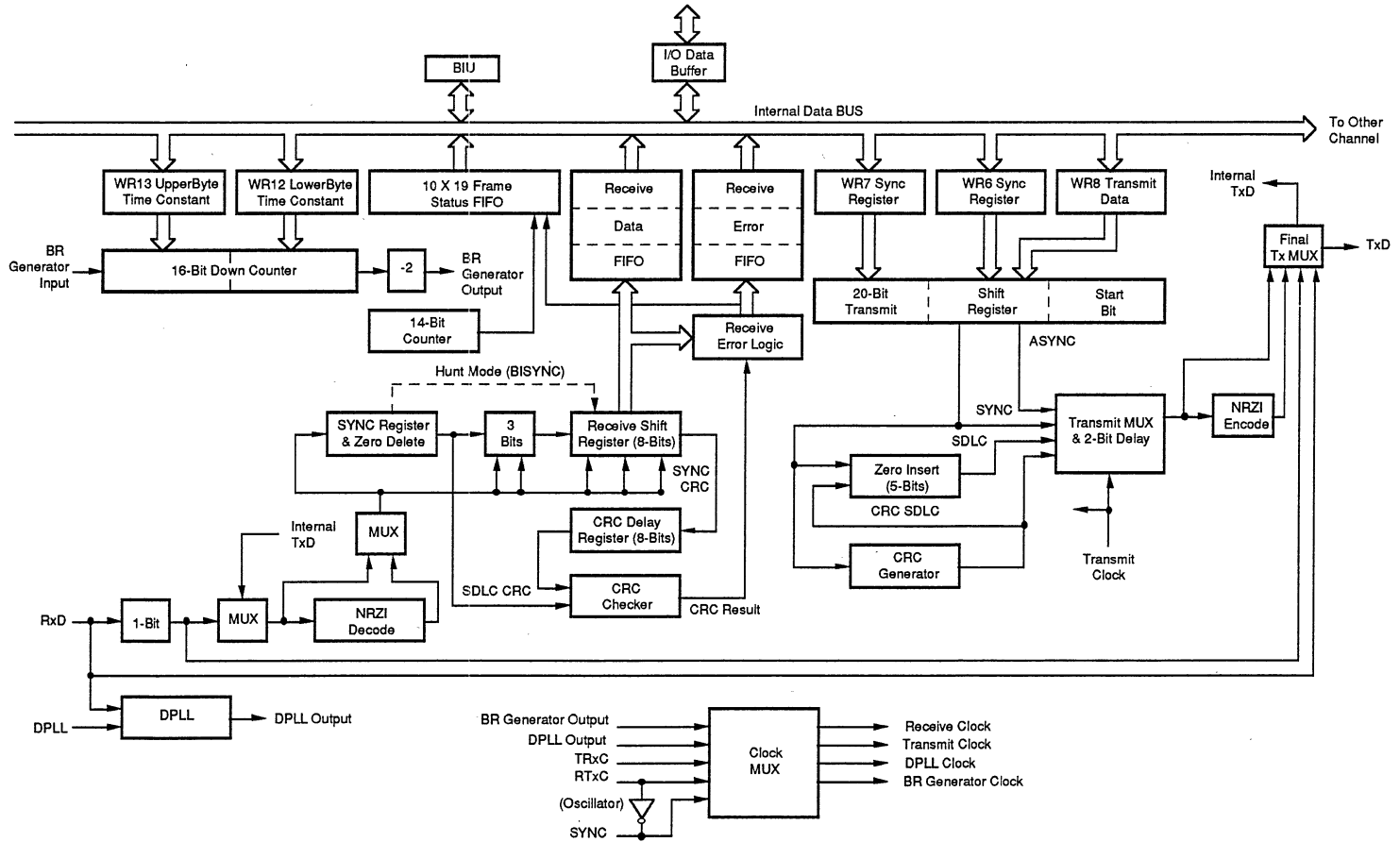


Figure 8. Data Path

---

## ARCHITECTURE (Continued)

The register set for each channel includes ten control (write) registers, two sync-character (write) registers, and four status (read) registers. In addition, each baud rate generator has two (read/write) registers for holding the time constant that determines the baud rate. Finally, associated with the interrupt logic is a write register for the interrupt vector accessible through either channel, a write only Master Interrupt Control register and three read registers: one containing the vector with status information (Channel B only), one containing the vector without status (Channel A only), and one containing the Interrupt Pending bits (Channel A only).

The registers for each channel are designated as follows:

- WRO-WR15 - Write Registers 0 through 15.
- RRO-RR3, RR10, RR12, RR13, RR15 - Read Registers 0 through 3, 10, 12, 13, 15.

Table 1 lists the functions assigned to each read or write register. The SCC contains only one WR2 and WR9, but

they can be accessed by either channel. All other registers are paired (one for each channel).

**Data Path.** The transmit and receive data path illustrated in Figure 8 is identical for both channels. The receiver has three 8-bit buffer registers in a FIFO arrangement, in addition to the 8-bit receive shift register. This scheme creates additional time for the CPU to service an interrupt at the beginning of a block of high speed data. Incoming data is routed through one of several paths (data or CRC) depending on the selected mode (the character length in Asynchronous modes also determines the data path).

The transmitter has an 8-bit Transmit Data buffer register loaded from the internal data bus and a 20-bit Transmit Shift register that can be loaded either from the synchronous character registers or from the Transmit Data register. Depending on the operational mode, outputting data is routed through one of four main paths before it is transmitted from the Transmit Data output (TxD).

---

## ARCHITECTURE (Continued)

**Table 1. Read and Write Register Functions**

---

### Read Register Functions

RR0	Transmit/Receive buffer status and External status.
RR1	Special Receive Condition status
RR2	Modified interrupt vector (Channel B only). Unmodified interrupt vector (Channel A only).
RR3	Interrupt Pending bits (Channel A only).
RR8	Receive buffer
RR10	Miscellaneous status
RR12	Lower byte of baud rate generator time constant.
RR13	Upper byte of baud rate generator time constant.
RR15	External/Status interrupt information.

---

### Write Register Functions

WR0	CRC initialize, initialization commands for the various modes, Register Pointers.
WR1	Transmit/Receive interrupt and data transfer mode definition.
WR2	Interrupt vector (accessed through either channel).
WR3	Receive parameters and control.
WR4	Transmit/Receive miscellaneous parameters and modes.
WR5	Transmit parameters and controls.
WR6	Sync characters or SDLC address field.
WR7	Sync character of SDLC flag.
WR8	Transmit buffer.
WR9	Master interrupt control and reset (accessed through either channel).
WR10	Miscellaneous transmitter/receiver control bits
WR11	Clock mode control.
WR12	Lower byte of baud rate generator time constant.
WR13	Upper byte of baud rate generator time constant.
WR14	Miscellaneous control bits.
WR15	External/Status interrupt control.

---

---

## PROGRAMMING

The SCC contains write registers in each channel that are programmed by the system separately to configure the functional characteristics of the channels.

In the SCC, register addressing is direct for the data registers only, which are selected by a High on the D//C pin. In all other cases (with the exception of WR0 and RRO), programming the write registers requires two write operations and reading the read registers requires both a write and a read operation. The first write is to WR0 and contains

three bits that point to the selected register. The second write is the actual control word for the selected register, and if the second operation is read, the selected read register is accessed.

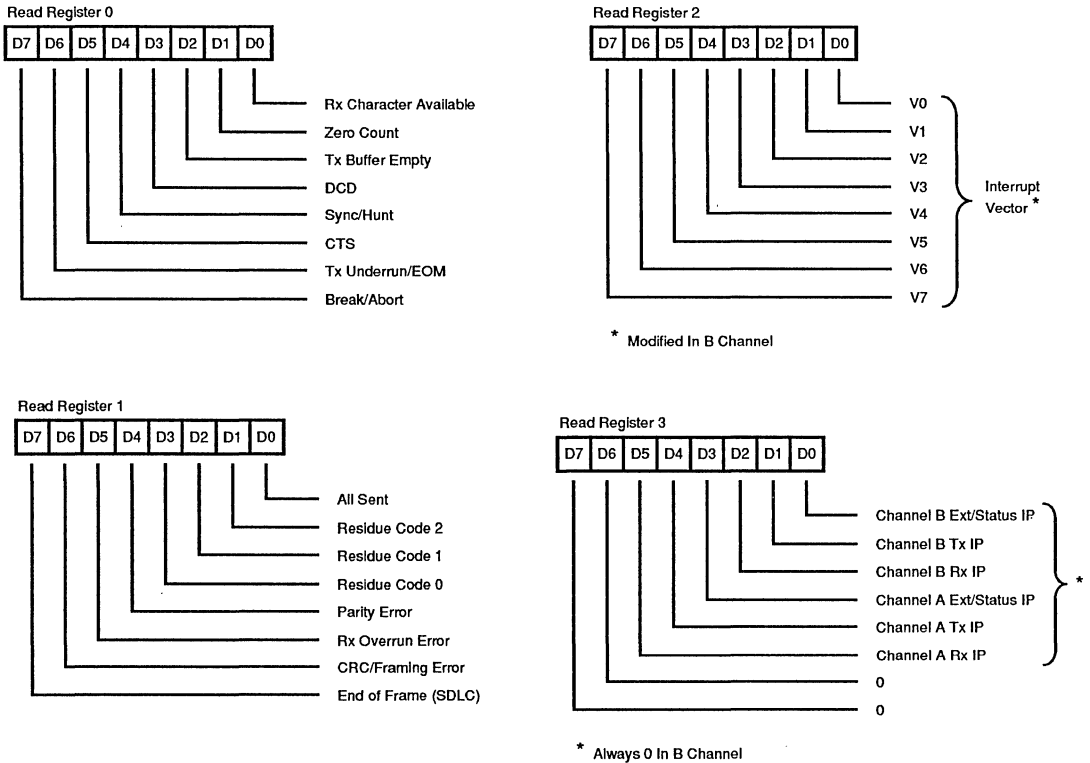
All of the registers in the SCC, including the data registers, may be accessed in this fashion. The pointer bits are automatically cleared after the read or write operation so that WR0 (or RRO) is addressed again.

The system program first issues a series of commands to initialize the basic mode of operation. This is followed by other commands to qualify conditions within the selected mode. For example, the Asynchronous mode, character length, clock rate, number of stop bits, even or odd parity might be set first. Then the interrupt mode would be set, and finally, receiver or transmitter enable.

**Read Registers.** The SCC contains eight read registers (actually nine, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers

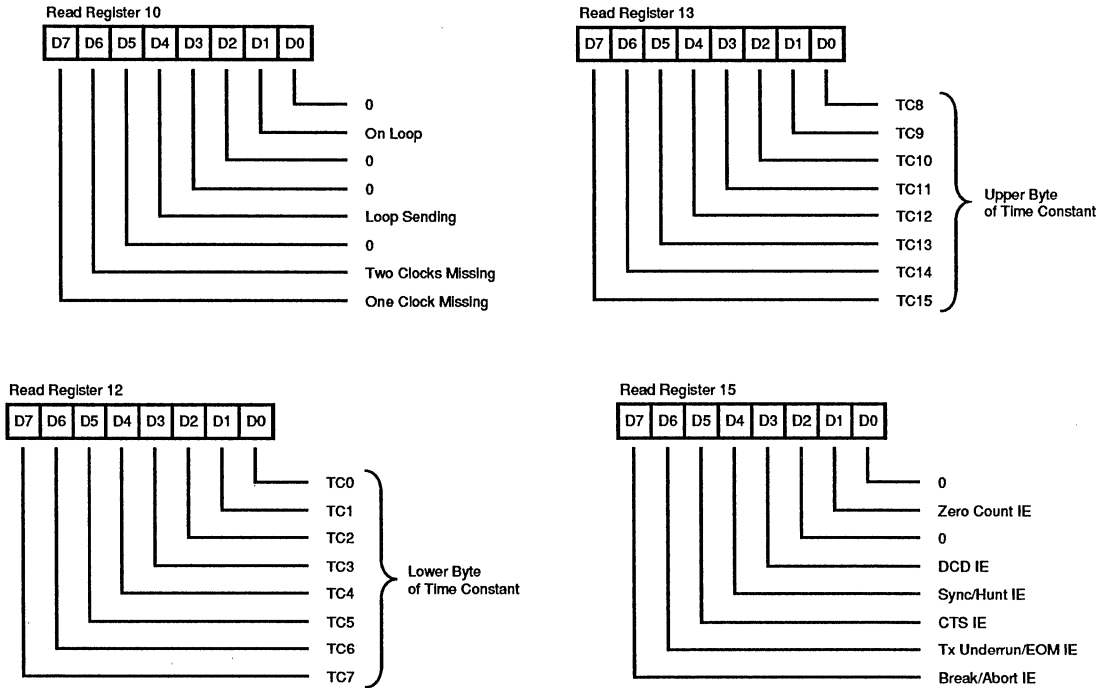
(RR12 and RR13) may be read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A). Figure 9 shows the formats for each read register.

The status bits of RR0 and RR1 are carefully grouped to simplify status monitoring; e.g., when the interrupt vector indicates a Special Receive Condition interrupt, all the appropriate error bits can be read from a single register (RR1).



**Figure 9. Read Register Bit Functions**

## PROGRAMMING (Continued)

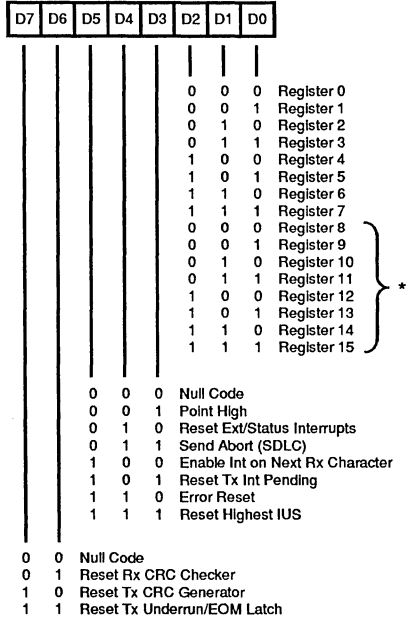


**Figure 9. Read Register Bit Functions (Continued)**

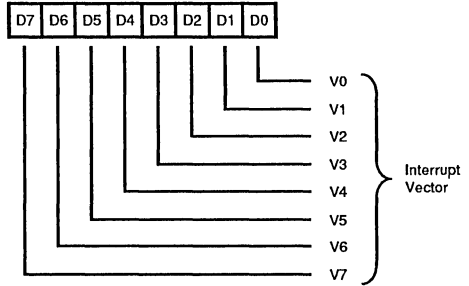
**Write Registers.** The SCC contains 13 write registers (14 counting WR8, the transmit buffer) in each channel. These write registers are programmed separately to configure the functional “personality” of the channels. In addition, there are two registers (WR2 and WR9) shared by

the two channels that may be accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits. Figure 10 shows the format of each write register.

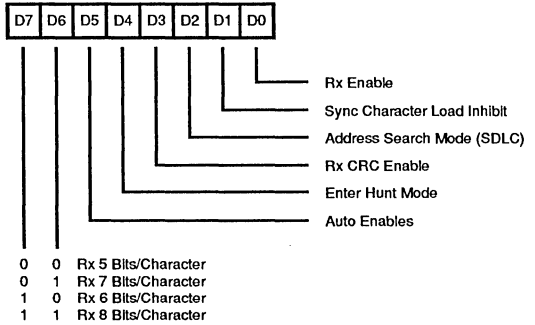
Write Register 0 (non-multiplexed bus mode)



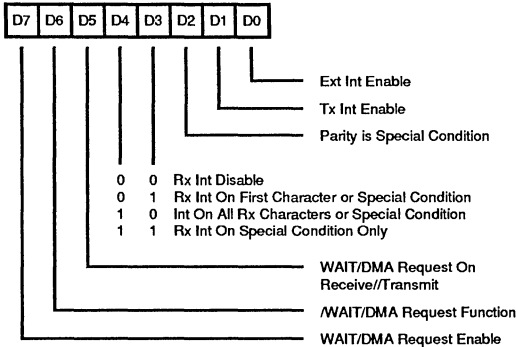
Write Register 2



Write Register 3



Write Register 1



Write Register 4

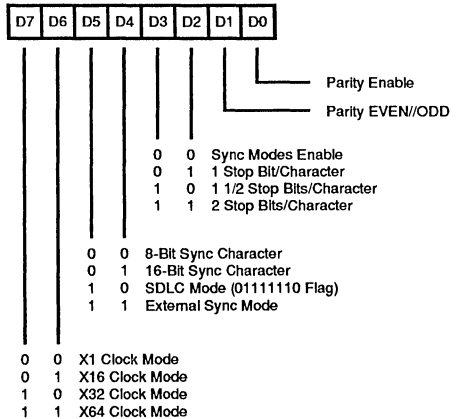
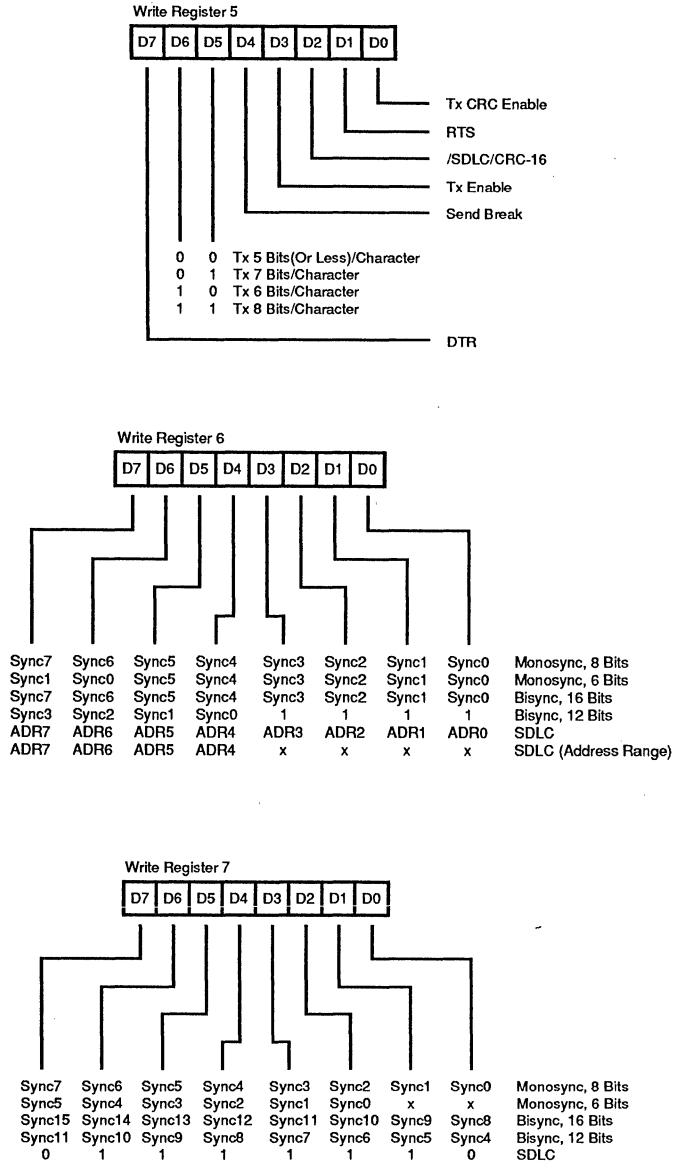


Figure 10. Write Register Bit Functions

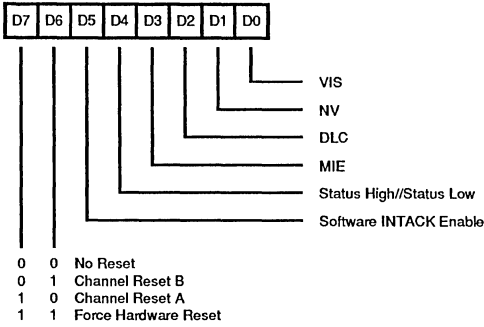


**PROGRAMMING (Continued)**

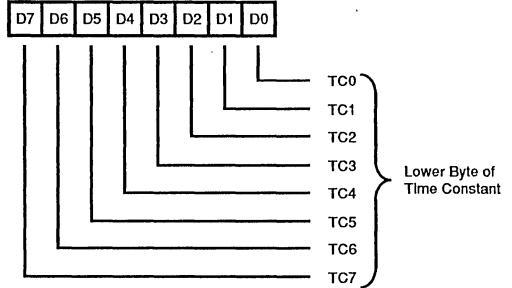


**Figure 10. Write Register Bit Functions (Continued)**

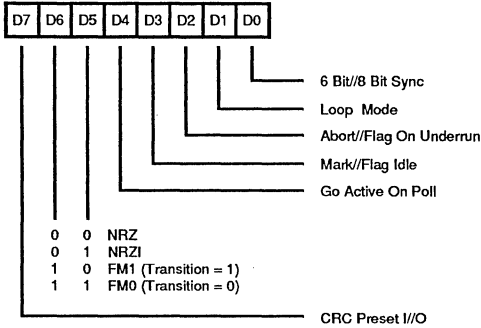
Write Register 9



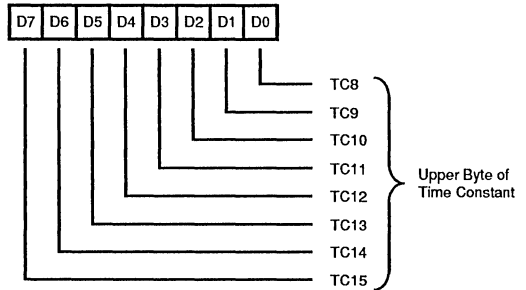
Write Register 12



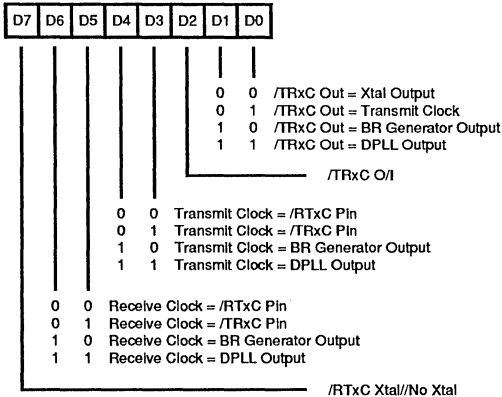
Write Register 10



Write Register 13



Write Register 11



Write Register 14

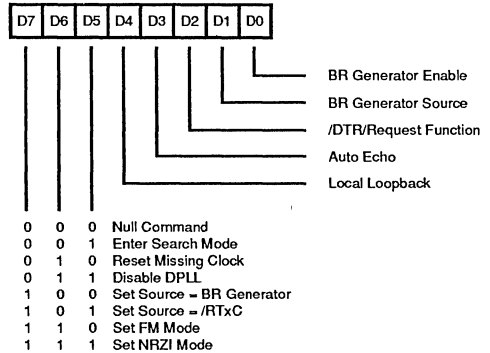
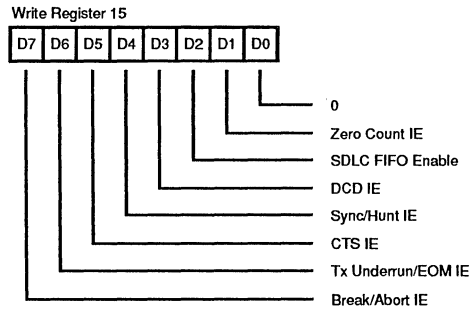


Figure 10. Write Register Bit Functions (Continued)

---

## PROGRAMMING (Continued)



**Figure 10. Write Register Bit Functions (Continued)**

---

## TIMING

The SCC generates internal control signals from  $/WR$  and  $/RD$  that are related to  $PCLK$ . Since  $PCLK$  has no phase relationship with  $/WR$  and  $/RD$ , the circuitry generating these internal control signals must provide time for metastable conditions to disappear. This gives rise to a recovery time related to  $PCLK$ . The recovery time applies only between bus transactions involving the SCC. The recovery time applies only between bus transactions involving the SCC. The recovery time required for proper operation is specified from the falling edge of  $/WR$  or  $/RD$  in the first

transaction involving the SCC to the falling edge of  $/WR$  or  $/RD$  in the second transaction involving the SCC. This time must be at least 4  $PCLK$  regardless of which register or channel is being accessed.

**Read Cycle Timing.** Figure 11 illustrates Read cycle timing. Addresses on  $A//B$  and  $D//C$  and the status on  $/INTACK$  must remain stable throughout the cycle. If  $/SCCCS$  falls after  $/RD$  falls or if it rises before  $/RD$  rises, the effective  $/RD$  is shortened.

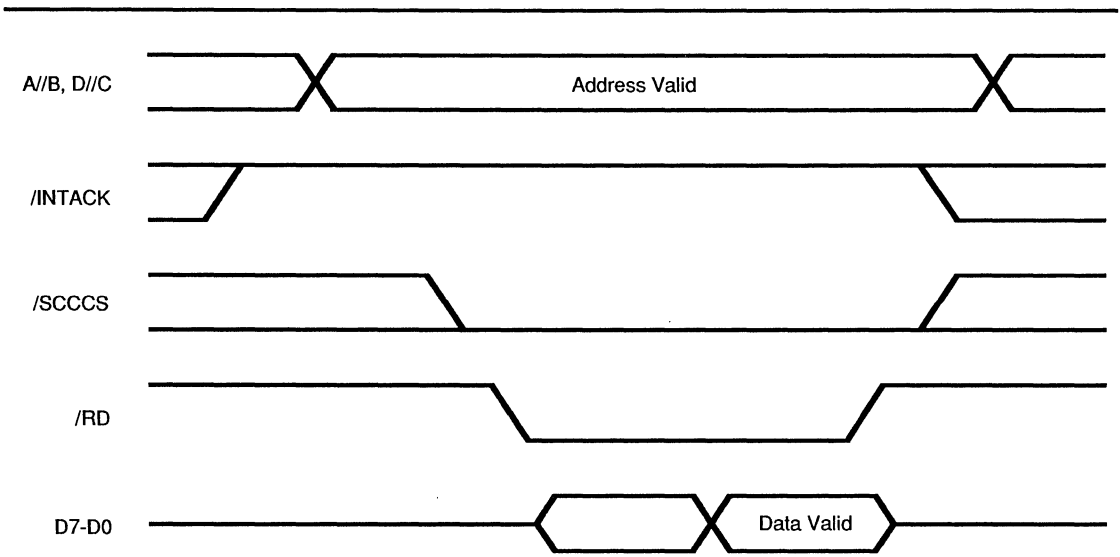


Figure 11. Read Cycle Timing

**Write Cycle Timing.** Figure 12 illustrates Write cycle timing. Addresses on A//B and D//C and the status on /INTACK must remain stable throughout the cycle. If /SCCCS falls

after /WR falls or if it rises before /WR rises, the effective /WR is shortened. Data must be valid before the falling edge of /WR.

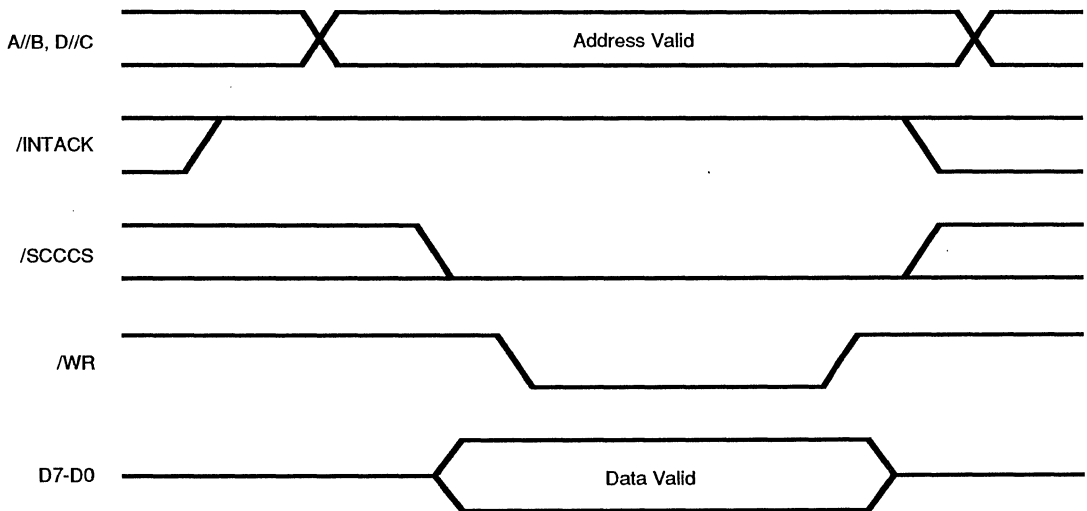
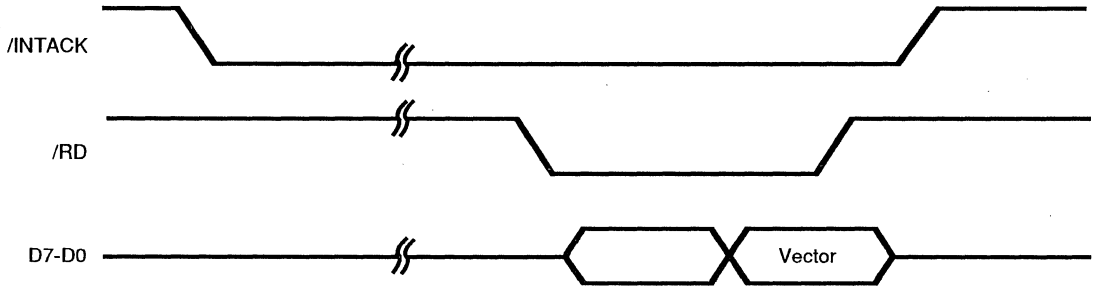


Figure 12. Write Cycle Timing

---

## TIMING (Continued)

**Interrupt Acknowledge Cycle Timing.** Figure 13 illustrates Interrupt Acknowledge cycle timing.



**Figure 13. Interrupt Acknowledge Cycle Timing**

---

## FIFO

The following text explains the functional operations of the FIFO.

**FIFO Enhancements.** When used with a DMA controller, the Z85C30 FIFO enhancement maximizes the SCC's ability to receive high speed back-to-back SDLC messages while minimizing frame overruns due to CPU latencies in responding to interrupts.

Additional logic was added to the industry standard NMOS SCC consisting of a 10 deep by 19 bit status FIFO, 14-bit receive byte counter, and control logic as shown in Figure 14. The 10 x 19 bit status FIFO is separate from the existing three byte receive data FIFO.

When the enhancement is enabled, the status in read register 1 (RR1) and byte count for the SDLC frame will be stored in the 10 x 19 bit status FIFO. This allows the DMA controller to transfer the next frame into memory while the CPU verifies the message was properly received.

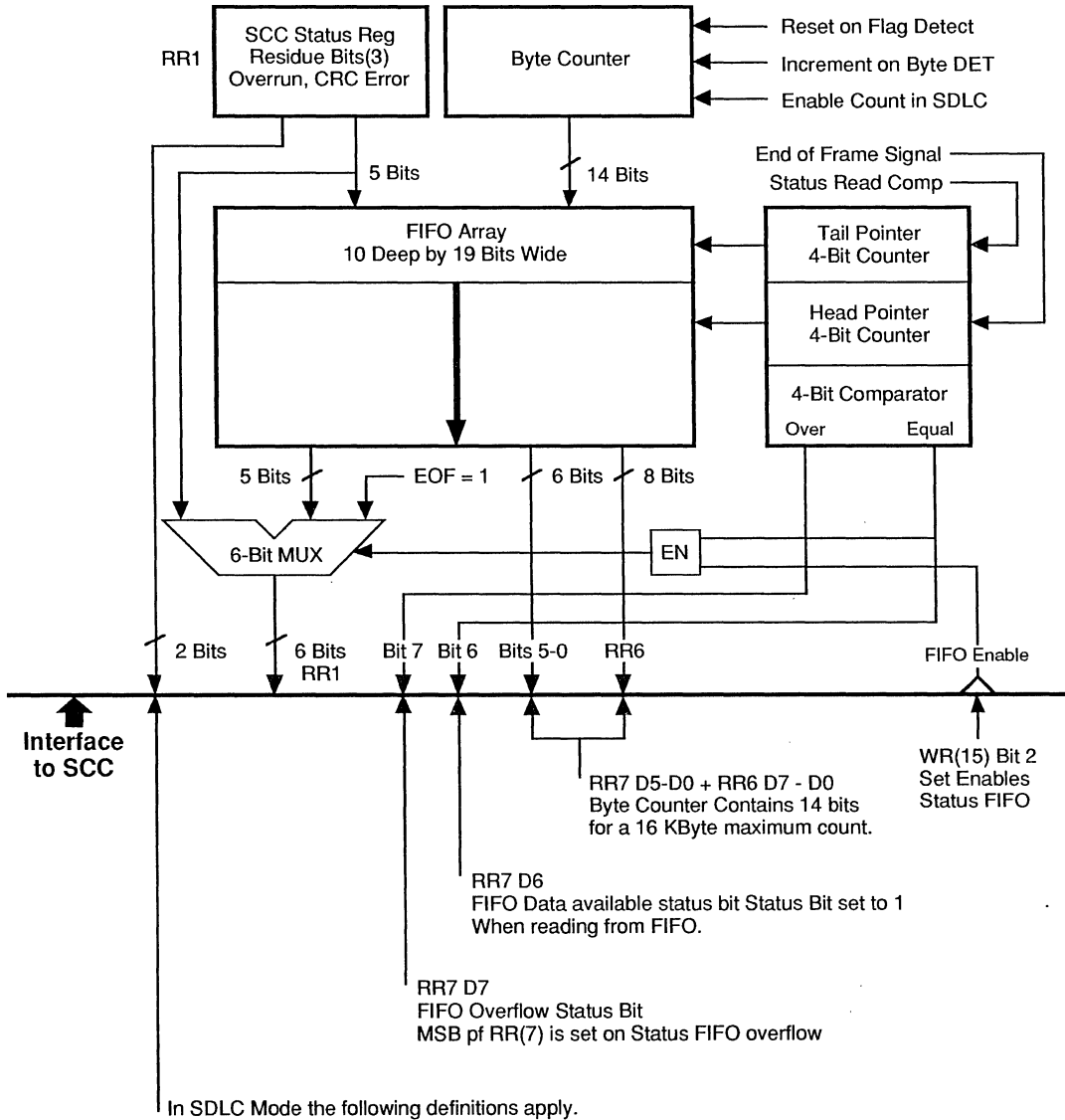
Summarizing the operation, data is received, assembled, loaded into the three byte receive FIFO before being transferred to memory by the DMA controller. When a flag is received at the end of an SDLC frame, the frame byte count from the 14-bit counter and five status bits are loaded into the status FIFO for verification by the CPU.

The CRC checker is automatically reset in preparation for the next frame which can begin immediately. Since the byte count and status are saved for each frame, the message integrity can be verified at a later time. Status information for up to 10 frames can be stored before a status FIFO overrun could occur.

**FIFO Detail.** For a better understanding of details of the FIFO operation, refer to the block diagram contained in Figure 14.

**Enable/Disable.** This FIFO is implemented so that it is enabled when WR15 bit 2 is set and the SCC is in the SDLC/HDLC mode, otherwise the status register contents bypass the FIFO and go directly to the bus interface (the FIFO pointer logic is reset either when disabled or via a channel or power-on reset). When the FIFO mode is disabled, the SCC is completely downward-compatible with the NMOS 8530. The FIFO mode is disabled on power-up (WR15 bit 2 is set to 0 on reset). The effects of backward compatibility on the register set are that RR4 is an image of RR0, RR5 is an image of RR1, RR6 is an image of RR2 and RR7 is an image of RR3. For the details of the added registers, refer to Figure 16. The status of the FIFO Enable signal can be obtained by reading RR15 bit 2. If the FIFO is enabled, the bit will be set to 1; otherwise, it will be reset.

### Frame Status FIFO Circuitry



- All Sent bypasses MUX and equals contents of SCC Status Register.
- Parity Bits bypasses MUX and does the same.
- EOF is set to 1 whenever reading from the FIFO.

Figure 14. SCC Status Register Modifications

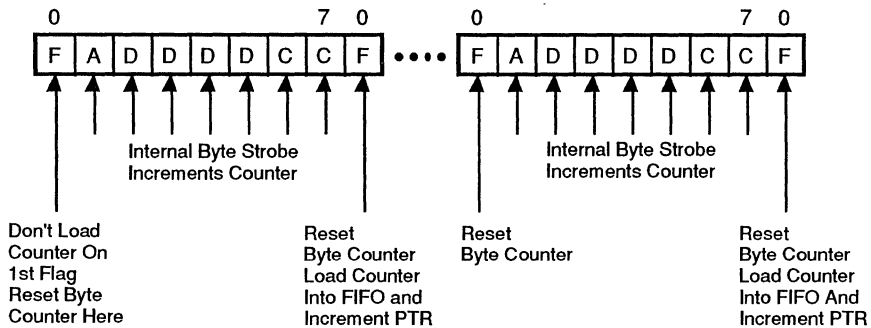
## FIFO (Continued)

**Read Operation.** When WR15 bit 2 is set and the FIFO is not empty, the next read to any of status register RR1 or the additional registers RR7 and RR6 will actually be from the FIFO. Reading status register RR1 causes one location of the FIFO to be emptied, so status should be read after reading the byte count, otherwise the count will be incorrect. Before the FIFO underflows, it is disabled. In this case, the multiplexer is switched to allow status to read directly from the status register, and reads from RR7 and RR6 will contain bits that are undefined. Bit 6 of RR7 (FIFO Data Available) can be used to determine if status data is coming from the FIFO or directly from the status register, since it is set to 1 whenever the FIFO is not empty.

Since not all status bits must be stored in the FIFO, the All Sent, Parity, and EOF bits will bypass the FIFO. The status bits sent through the FIFO will be Residue Bits (3), Overrun, and CRC Error.

The sequence for proper operation of the byte count and FIFO logic is to read the registers in the following order: RR7, RR6, and RR1 (reading RR6 is optional). Additional logic prevents the FIFO from being emptied by multiple reads from RR1. The read from RR7 latches the FIFO empty/full status bit (bit 6) and steers the status multiplexer to read from the SCC megacell instead of the status FIFO (since the status FIFO is empty). The read from RR1 allows an entry to be read from the FIFO (if the FIFO was empty, logic is added to prevent a FIFO underflow condition).

**Write Operation.** When the end of an SDLC frame (EOF) has been received and the FIFO is enabled, the contents of the status and byte-count registers are loaded into the FIFO. The EOF signal is used to increment the FIFO. If the FIFO overflows, the MSB of RR7 (FIFO Overflow) is set to indicate the overflow. This bit and the FIFO control logic is reset by disabling and re-enabling the FIFO control bit (WR15 bit 2). For details of FIFO control timing during an SDLC frame, refer to Figure 15.



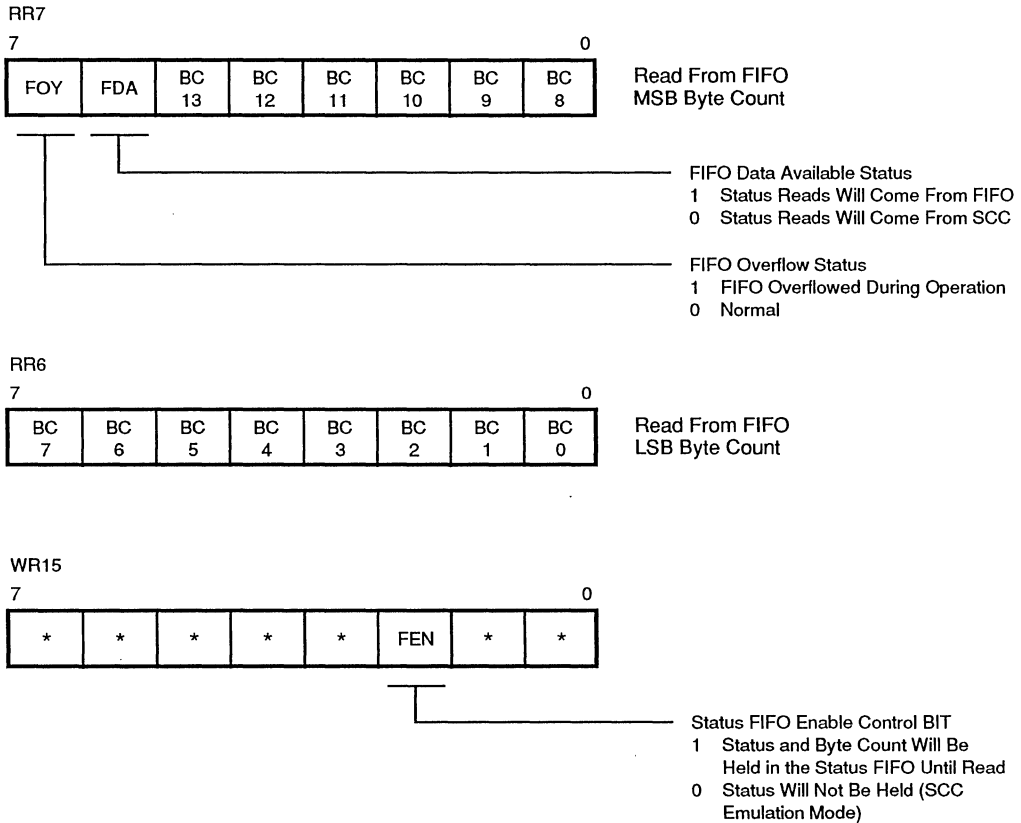
**Figure 15. SDLC Byte Counting Detail**

**Byte Counter Detail.** The 14-bit byte counter allows for packets up to 16K bytes to be received. For a better understanding of its operation refer to Figures 14 and 15.

**Enable.** The byte counter is enabled in the SDLC/HDLC mode.

**Reset.** The byte counter is reset whenever an ADLC flag character is received. The reset is timed so that the contents of the byte counter are successfully written into the FIFO.

**Increment.** The byte counter is incremented by writes to the data FIFO. The counter represents the number of bytes received by the SCC, rather than the number of bytes transferred from the SCC. (These counts may differ by up to the number of bytes in the receive data FIFO contained in the SCC).



\* No change From NMOS SCC DFN

Figure 16. SCC Additional Registers

## SOFTWARE INTERRUPT ACKNOWLEDGE

The SCC can do an interrupt acknowledge cycle through software. In some CPU environments it is difficult to create the /INTACK signal with the necessary timing to acknowledge interrupts and allow the nesting of interrupts. In these cases, it would be desirable to create this signal in software.

If bit 5 of Write Register 9 (WR9) is set, reading register 2 (RR2) will result in an interrupt acknowledge cycle to be executed internally. Like a hardware INTACK cycle, a software acknowledge will cause the /INT pin to return high.

Similarly to when the /INTACK signal is used, when a software acknowledge cycle is used, a Reset Highest IUS command must be issued in the interrupt service routine. If the RR2 is read from channel B, the modified vector will be returned. If the RR2 is read from channel A, then the vector will be returned unmodified. The Vector Includes Status (VIS) and no vector (NV) bits (WR9) and are ignored when bit 5 is set to 1.

When the /INTACK is not being used, it should be pulled up to VDD through a resistor (10K ohm typical).



## SCSI FUNCTIONAL DESCRIPTION

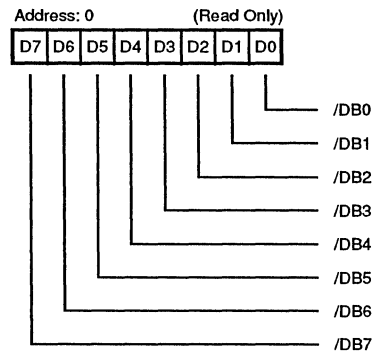
**General.** The Small Computer System interface (SCSI) device has a set of eight registers that are controlled by the CPU. By reading and writing the appropriate registers, the CPU may initiate any SCSI Bus activity or may sample and assert any signal on the SCSI Bus. This allows the user to implement all or any of the SCSI protocol in software. These registers are read (written) by activating /SCSICS with an address on A2-A0 and then issuing a /RD (/WR) pulse. This section describes the operation of the internal registers (Table 2).

**Table 2. Register Summary**

Address			R/W	Register Name
A2	A1	A0		
0	0	0	R	Current SCSI Data
0	0	0	W	Output Data
0	0	1	R/W	Initiator Command
0	1	0	R/W	Mode
0	1	1	R/W	Target Command
1	0	0	R	Current SCSI Bus Status
1	0	0	W	Select Enable
1	0	1	R	Bus and Status
1	0	1	W	Start DMA Send
1	1	0	R	Input Data
1	1	0	W	Start DMA Target Receive
1	1	1	R	Reset Parity/Interrupt
1	1	1	W	Start DMA Initiator Receive

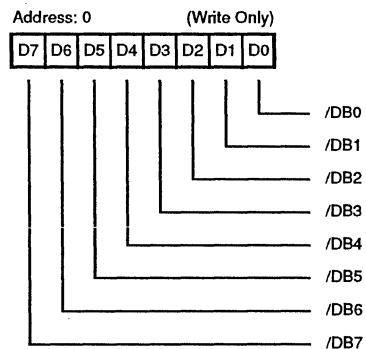
**Data Registers.** The data registers are used to transfer SCSI commands, data, status, and message bytes between the microprocessor Data Bus and the SCSI Bus. The SCSI does not interpret any information that passes through the data registers. The data registers consist of the transparent Current SCSI Data Register, the Output Data Register, and the Input Data Register.

**Current SCSI Data Register.** *Address 0 (Read Only).* The Current SCSI Data Register (Figure 17) is a read-only register which allows the microprocessor to read the active SCSI Data Bus. This is accomplished by activating /SCSICS with an address on A2-A0 and issuing a /RD pulse. If parity checking is enabled, the SCSI Bus parity is checked at the beginning of the read cycle. This register is used during a programmed I/O data read or during Arbitration to check for higher priority arbitrating devices. Parity is not guaranteed valid during Arbitration.



**Figure 17. Current SCSI Data Register**

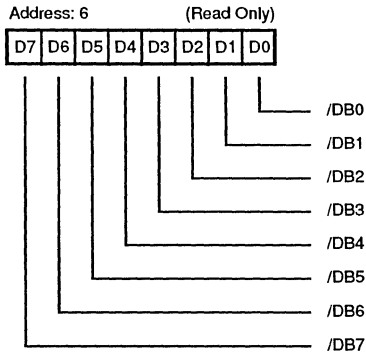
**Output Data Register.** *Address 0 (Write Only).* The Output Data Register (Figure 18) is a write-only register that is used to send data to the SCSI Bus. This is accomplished by either using a normal CPU write, or under DMA control, by using /WR and /DACK. This register also asserts the proper ID bits on the SCSI Bus during the Arbitration and Selection phases.



**Figure 18. Output Data Register**

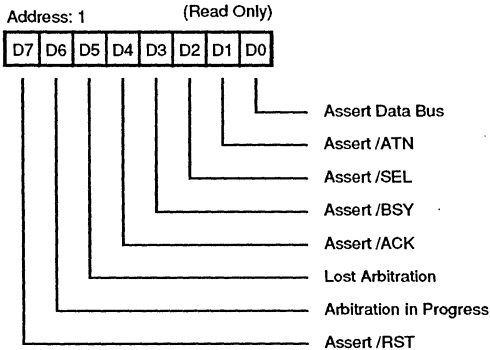
**Input Data Register.** *Address 6 (Read Only).* The Input Data Register (Figure 19) is a read-only register that is used to read latched data from the SCSI Bus. Data is latched either during a DMA Target receive operation

when /ACK goes active or during a DMA Initiator receive when /REQ goes active. The DMA Mode bit (Mode Register bit 1) must be set before data can be latched in the Input Data Register. This register is read under DMA control using /RD and /DACK. Parity is optionally checked when the Input Data Register is loaded.

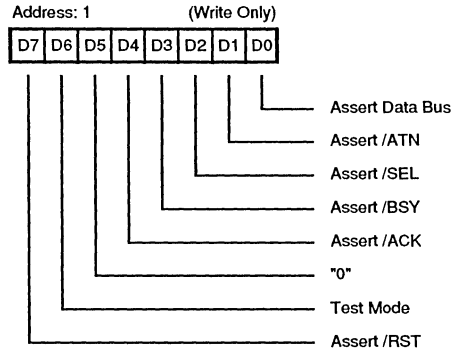


**Figure 19. Input Data Register**

**Initiator Command Register.** Address 1 (read/write). The Initiator Command Register (Figures 20 and 21) are read and write registers which assert certain SCSI Bus signals, monitors those signals, and monitors the progress of bus arbitration. Many of these bits are significant only when being used as an Initiator; however, most can be used during Target role operation.



**Figure 20. Initiator Command Register (Register Read)**



**Figure 21. Initiator Command Register (Register Write)**

The following describes the operation of all bits in the Initiator Command Register.

**Bit 0. Assert Data Bus.** The ASSERT DATA BUS bit, when set, allows the contents of the Output Data Register to be enabled as chip outputs on the signals /DB7-/DB0. Parity is also generated and asserted on /DBP.

When connected as an Initiator, the outputs are only enabled if the TARGETMODE bit (Mode Register, bit 6) is FALSE, the received signal I/O is FALSE, and the phase signals C//D, I//O, and /MSG match the contents of the ASSERT C//O, ASSERT I//O and ASSERT /MSG in the Target Command Register.

This bit should also be set during DMA send operations.

**Bit 1. ASSERT/ATN/ATN.** Bit 1 may be asserted on the SCSI Bus by setting this bit to a one (1) if the TRAGETMODE bit (Mode Register, bit 6) is FALSE /ATN is normally asserted by the initiator to request a Message Out bus phase. Note that since ASSERT/SEL and ASSERT/ATN are in the same register, a select with /ATN may be implemented with one CPU write /ATN may be deasserted by resetting this bit to zero. A read on this register simply reflects the status of this bit.

**Bit 2. ASSERT/SEL.** Writing a one (1) into this bit position asserts /SEL onto the SCSI Bus. /SEL is normally asserted after Arbitration has been successfully completed /SEL may be disabled by resetting bit 2 to a zero. A read of this register reflects the status of this bit.

## SCSI FUNCTIONAL DESCRIPTION (Continued)

**Bit 3. ASSERT/BSY.** Writing a one (1) into this bit position asserts /BSY onto the SCSI Bus. Conversely, a zero resets the /BSY signal. Asserting /BSY indicates a successful selection or reselection. Resetting this bit creates a Bus-Disconnect condition. Reading this register reflects bit status.

**Bit 4. ASSERT/ACK.** Bit 4 is used by the bus initiator to assert /ACK on the SCSI Bus. In order to assert /ACK, the TARGETMODE bit (Mode Register, bit 6) must be FALSE. Writing a zero to this bit deasserts /ACK. Reading this register reflects bit status.

**Bit 5. "0" (Write Bit).** Bit 5 should be written with a zero for proper operation.

**Bit 5. LA (Lost Arbitration - Read Bit).** Bit 5, when active, indicates that the SCSI detected a Bus-Free condition, arbitrated for use of the bus by asserting /BSY and its ID on the Data Bus, and lost Arbitration due to /SEL being asserted by another bus device. This bit is active only when the ARBITRATE bit (Mode Register, bit 0) is active.

**Bit 6. TEST MODE (Write Bit).** Bit 6 is written during a test environment to disable all output drivers, effectively removing the Z53C80 from the circuit. Resetting this bit returns the part to normal operation.

**Bit 6. AIP (Arbitration in Process - Read Bit).** Bit 6 is used to determine if Arbitration is in progress. For this bit to be active, the ARBITRATE bit (Mode Register, bit 0) must have been set previously. It indicates that a Bus-Free condition has been detected and that the chip has asserted /BSY and put the contents of the Output Data Register onto the SCSI Bus. AIP will remain active until the ARBITRATE bit is reset.

**Bit 7. ASSERT/RST.** Whenever a one is written to bit 7 of the Initiator Command Register, the /RST signal is asserted on the SCSI Bus. The /RST signal will remain asserted until this bit is reset or until an external /RESET occurs. After this bit is set (1), IRQ goes active and all internal logic and control registers are reset (except for the interrupt latch and the ASSERT/RST bit). Writing a zero to bit 7 of the Initiator Command Register deasserts the /RST signal. The status of this bit is monitored by reading the Initiator Command Register.

**Mode Register.** Address 2 (Read/Write). The Mode Register controls the operation of the chip. This register determines whether the SCSI operates as an Initiator or a Target, whether DMA transfers are being used, whether

parity is checked, and whether interrupts are generated on various external conditions. This register is read to check the value of these internal control bits (Figure 22).

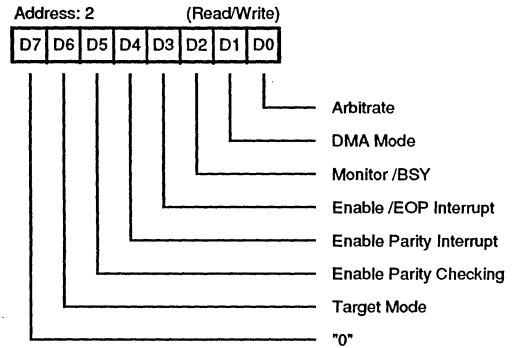


Figure 22. Mode Register

**Bit 0. ARBITRATE.** The ARBITRATE bit is set (1) to start the Arbitration process. Prior to setting this bit, the Output Data Register should contain the proper SCSI device ID value. Only one data bit should be active for SCSI Bus Arbitration. The SCSI waits for a Bus-Free condition before entering the Arbitration phase. The results of the Arbitration phase is determined by reading the status bits LA and AIP (Initiator Command Register, bits 5 and 6, respectively).

**Bit 1. DMA MODE.** The DMA MODE bit is normally used to enable a DMA transfer and must be set (1) prior to writing Start DMA Send Register, Start DMA Target Receive Register, and Start DMA Initiator Receiver Register. These three registers are used to start DMA transfers. The TARGETMODE bit (Mode Register, bit 6) must be consistent with writes to Start DMA Target Receive and Start DMA Initiator Receive Registers [i.e., set (1) for a write to start DMA Target Receive Register and set (0) for a write to Start DMA Initiator Receive Register]. The control bit ASSERT DATA BUS (Initiator Command Register, bit 0) must be TRUE (1) for all DMA send operations. In the DMA mode, /REQ and /JACK are automatically controlled.

The DMA MODE bit is not reset upon the receipt of an /EOP signal. Any DMA transfer is stopped by writing a zero into this bit location; however, care must be taken not to cause /SCSICS and /DACK to be active simultaneously.

**Bit 2. MONITOR BUSY.** The MONITOR BUSY bit, when TRUE (1), causes an interrupt to be generated for an

unexpected loss of /BSY. When the interrupt is generated due to loss of /BSY, the lower six bits of the Initiator Command Register are reset (0) and all signals are removed from the SCSI Bus.

**Bit 3. ENABLE/EOP interrupt.** The enable /EOP interrupt, when set (1), causes an interrupt to occur when the /EOP (End of Process) signal is received from the DMA controller logic.

**Bit 4. ENABLE PARITY INTERRUPT.** The ENABLE PARITY INTERRUPT bit, when set (1), will cause an interrupt (IRQ) to occur if a parity error is detected. A parity interrupt will only be generated if the ENABLE PARITY CHECKING bit (bit 5) is also enabled (1).

**Bit 5. ENABLE PARITY CHECKING.** The ENABLE PARITY CHECKING bit determines whether parity errors are ignored or saved in the parity error latch. If this bit is reset (0), parity is ignored. Conversely, if this bit is set (1), parity errors are saved.

**Bit 6. TARGETMODE.** The TARGETMODE bit allows the SCSI to operate as either a SCSI Bus Initiator, bit reset (0), or as a SCSI Bus Target device, bit set (1). If the signals /ATN and /ACK are to be asserted on the SCSI Bus, the TARGETMODE bit must be reset (0). If the signals C//D, I//O, /MSG, and /REQ are to be asserted on the SCSI Bus, the TARGETMODE bit must be set (1).

**Bit 7. "0".** Bit 7 should be written with a zero for proper operation.

**Target Command Register. Address 3(Read/Write).** When connected as a target device, the Target Command Register (Figure 23) allows the CPU to control the SCSI Bus Information Transfer phase and/or to assert /REQ by writing this register. The TARGETMODE bit (Mode Register, bit 6) must be TRUE (1) for bus assertion to occur. The SCSI Bus phases are described in Table 3.

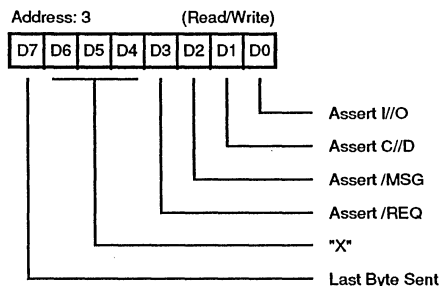
**Table 3. SCSI Information Transfer Phase**

Bus Phase	ASSERT I//O	ASSERT C//D	ASSERT /MS
Data Out	0	0	0
Unspecified	0	0	1
Command	0	1	0
Message Out	0	1	1
Data In	1	0	0
Unspecified	1	0	1
Status	1	1	0
Message In	1	1	1

When connected as an Initiator with DMA MODE TRUE, if the phase lines I//O, C//D, and /MSG do not match the phase bits in the Target Command Register, a phase mismatch interrupt is generated when /REQ goes active. To send data as an Initiator, the ASSERT I//O, ASSERT C//D, and ASSERT /MSG bits must match the corresponding bits in the Current SCSI Bus Status Register. The ASSERT /REQ bit (bit 3) has no meaning when operating as an Initiator.

Bits 4, 5, and 6 are not used.

**Bit 7. LAST BYTE SENT (Read Only).** The END OF DMA TRANSFER bit (Bus and Status Register, bit 7) only indicates when the last byte was received from the DMA controller. The LAST BYTE SENT bit can be used to flag that the last byte of the DMA send operation has been transferred on the SCSI Data Bus.



**Figure 23. Target Command Register**

**Current SCSI Bus Status Register. Address 4(Read Only).** The Current SCSI Bus Register is a read-only register which is used to monitor seven SCSI Bus control signals, plus the Data Bus parity bit. For example, an Initiator device can use this register to determine the current bus phase and to poll /REQ for pending data transfers. This register may also be used to determine why a particular interrupt occurred. Figure 24 describes the Current SCSI Bus Status Register.

**Select Enable Register. Address 4(Write Only).** The Select Enable Register (Figure 25) is a write-only register which is used as a mask to monitor a signal ID during a selection attempt. The simultaneous occurrence of the correct ID bit, /BSY FALSE, and /SEL TRUE will cause an interrupt. This interrupt can be disabled by resetting all bits in this register. If the ENABLE PARITY CHECKING bit (Mode Register, bit 5) is active (1), parity is checked during selection.

## SCSI FUNCTIONAL DESCRIPTION (Continued)

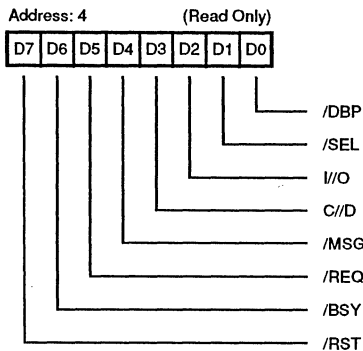


Figure 24. Current SCSI Bus Status Register

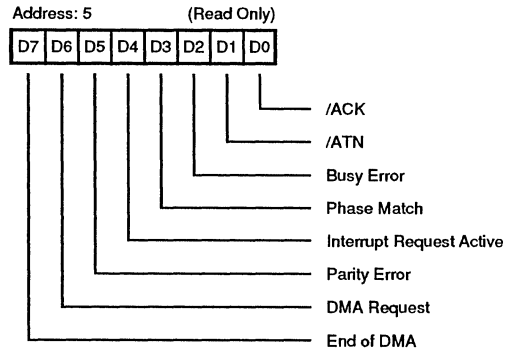


Figure 26. Bus and Status Register

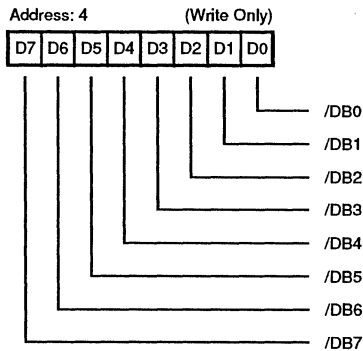


Figure 25. Select Enable Register

**Bus and Status Register.** Address 5 (Read Only). The Bus and Status Register (Figure 26) is a read-only register which can be used to monitor the remaining SCSI control signals not found in the Current SCSI Bus Status Registers (/ATN and /ACK), as well as six other status bits. The following describes each bit of the Bus Status Register individually.

**Bit 0. /ACK.** Bit 0 reflects the condition of the SCSI Bus control signal /ACK. This signal is normally monitored by the Target device.

**Bit 1. /ATN.** Bit 1 reflects the condition of the SCSI Bus control signal /ATN. This signal is normally monitored by the Target device.

**Bit 2. BUSY ERROR.** The BUSY ERROR bit is active if an unexpected loss of the /BSY signal has occurred. This latch is set whenever the MONITOR BUSY bit (Mode Register, bit 2) is TRUE and /BSY is FALSE. An unexpected loss of /BSY disables any SCSI outputs and resets the DMA MODE bit (Mode Register, bit 1).

**Bit 3. PHASE MATCH.** The SCSI signals /MSG, C//D, and I/O, represent the current information Transfer phase. The PHASE MATCH bit indicates whether the current SCSI Bus phase matches the lower 3 bits of the Target Command Register. PHASE MATCH is continuously updated and is only significant when operating as a Bus Initiator. A phase match is required for data transfers to occur on the SCSI Bus.

**Bit 4. INTERRUPT REQUEST ACTIVE.** Bit 4 is set if an enabled interrupt condition occurs. It reflects the current state of the IRQ output and can be cleared by reading the Reset Parity/Interrupt Register.

**Bit 5. PARITY ERROR.** Bit 5 is set if a parity error occurs during a data receive or a device selection. The PARITY ERROR bit can only be set (1) if the ENABLE PARITY CHECK bit (Mode Register, bit 5) is active (1). This bit may be cleared by reading the Reset Parity/Interrupt Register.

**Bit 6. DMA REQUEST.** The DMA REQUEST bit allows the CPU to sample the output pin DRQ. DRQ can be cleared by asserting /DACK or by resetting the DMA MODE bit (bit 1) in the Mode Register. The DRQ signal does not reset when a phase-mismatch interrupt occurs.

---

**Bit 7. END OF DMA TRANSFER.** The END OF DMA TRANSFER bit is set if /EOP, /DACK, and either /RD or /WR are simultaneously active for at least 100ns. Since the /EOP signal can occur during the last byte sent to the Output Data Register, the /REQ and /ACK signals should be monitored to ensure that the last byte has been transferred. This bit is reset when the DMA MODE bit is reset (0) in the Mode Register.

**DMA Registers.** Three write-only registers are used to initiate all DMA activity. They are: Start DMA Send, Start DMA Target Receive, and Start DMA Initiator Receive. Performing a write operation into one of these registers starts the desired type of DM transfer. Data presented to the SCSI on signals D7-D0 during the register write is meaningless and has no effect on the operation. Prior to writing these registers, the DMA MODE bit (bit 1), and the TARGETMODE bit (bit 6) in the Mode Register must be appropriately set. The individual registers are briefly described as follows:

**Start DMA Send.** *Address 5 (Write Only).* This register is written to initiate a DMA send, from the DMA to the SCSI Bus, for either Initiator or Target role operations. The DMA MODE bit (Mode Register, bit 1) is set prior to writing this register.

**Start DMA Target Receive.** *Address 6 (Write Only).* This register is written to initiate a DMA receive - from the SCSI Bus to the DMA, for Target operation only. The DMA MODE bit (bit 1) and the TARGETMODE bit (bit 6) in the Mode Register must both be set (1) prior to writing this register.

**Start DMA Initiator Receive.** *Address 7 (Write Only).* This register is written to initiate a DMA receive - from the SCSI Bus to the DMA, for Initiator operation only. The DMA MODE bit (bit 6) must be FALSE (0) in the Mode Register prior to writing this register.

**Reset Parity/Interrupt.** *Address 7 (Read Only).* Reading this register resets the PARITY ERROR bit (bit 5), the INTERRUPT REQUEST bit (bit 4), and the BUSY ERROR bit (bit 2) in the Bus and Status Register.

**On-Chip SCSI Hardware Support.** The SCSI is easy to use because of its simple architecture. The chip allows direct control and monitoring of the SCSI Bus by providing a latch for each signal. However, portions of the protocol define timings which are much too quick for traditional micro-processors to control. Therefore, hardware support has been provided for DMA transfers, bus arbitration, phase change monitoring, bus disconnection, bus reset, parity generation, parity checking, and device selection/reselection.

Arbitration is accomplished using a Bus-Free filter to continuously monitor /BSY. If /BSY remains inactive for at least 400ns, the SCSI is considered free and Arbitration may begin. Arbitration will begin if the bus is free, /SEL is inactive, and the ARBITRATE bit (Mode Register, bit 0) is active. Once arbitration has begun (/BSY asserted), an arbitration delay of 2.2  $\mu$ s must elapse before the Data Bus can be examined to determine if Arbitration is enabled. This delay is implemented in the controlling software driver.

The Z53C80 is a clockwise device. Delays such as bus-free delay, bus-set delay, and bus-settle delay are implemented using gate delays. These delays may differ between devices because of inherent process variations, but are well within the proposed ANSI X3.131 - 1986 specification.

**INTERRUPTS.** The Z53C80 provides an interrupt output (IRQ) to indicate a task completion or an abnormal bus occurrence. The use of interrupts is optional and may be disabled by resetting the appropriate bits in the Mode Register or the Select Enable Register.

When an interrupt occurs, the Bus and Status Register and the Current SCSI Bus Status Register (Figures 26 and 24) must be read to determine which condition created the interrupt. IRQ can be reset simply by reading the Reset Parity/Interrupt Register or by an external chip reset /RESET active for 100ns.

Assuming the Z53C80 has been properly initialized, an interrupt will be generated if the chip is selected or reselected, if an /EOP signal occurs, if a parity error occurs during a data transfer, if a bus phase mismatch occurs, or if a SCSI Bus disconnection occurs.

**Selection Reselection.** The Z53C80 generates a select interrupt if SEL is active (0), its device ID is TRUE and /BSY is FALSE for at least a bus-settle delay. If I//O is active, this is considered a reselect interrupt. The correct ID bit is determined by a match in the Select Enable Register. Only a single bit match is required to generate an interrupt. This interrupt may be disabled by writing zeros into all bits of the Select Enable Register.

If parity is supported, parity should be good during the selection phase. Therefore, if the ENABLE PARITY bit (Mode Register, bit 5) is active, the PARITY ERROR bit is checked to ensure that a proper selection has occurred. The ENABLE PARITY INTERRUPT bit need not be set for this interrupt to be generated.

## SCSI FUNCTIONAL DESCRIPTION (Continued)

The proposed SCSI specification also requires that no more than two device ID's be active during the selection process. To ensure this, the Current SCSI Data Register is read.

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 27 and 28, respectively.

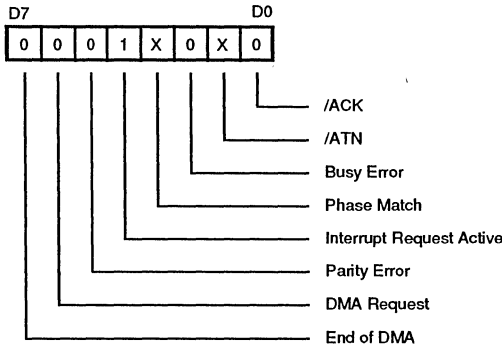


Figure 27. Bus and Status Register

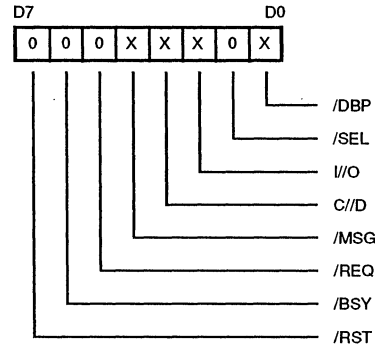


Figure 28. Current SCSI Bus Status Register

**End of Process (EOP) Interrupt.** An End Of Process signal (EOP) which occurs during a DMA transfer (DMA MODE TRUE) will set the END OF DMA Status bit (Bus and Status Register bit 7) and will optionally generate an interrupt if ENABLE EOP INTERRUPT bit (Mode Register, bit 3) is TRUE. The /EOP pulse will not be recognized (END OF DMA bit set) unless /EOP, /DACK, and either /RD or /WR are concurrently active for at least 50 ns. DMA transfers

can still occur if /EOP was not asserted at the correct time. This interrupt is disabled by resetting the ENABLE EOP INTERRUPT bit.

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register for this interrupt are shown in Figures 29 and 30.

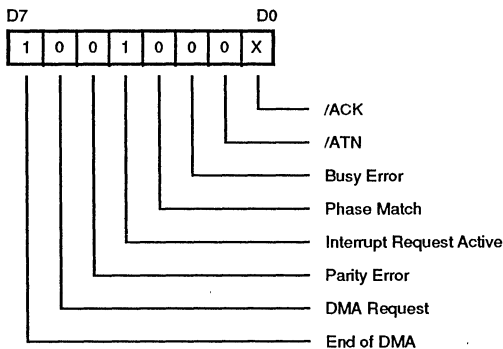


Figure 29. Bus and Status Register

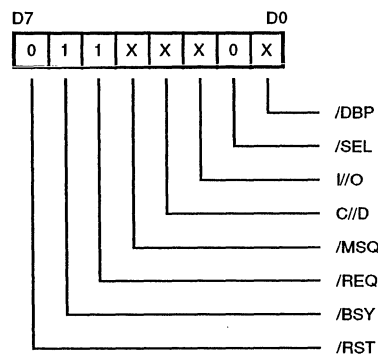


Figure 30. Current SCSI Bus Status Register

The END OF DMA bit is used to determine when a block transfer is complete. Receive operations are complete when there is no data left in the chip and no additional handshakes occurring. The only exception to this is receiving data as an Initiator and the Target opts to send additional data for the same phase. In this /REQ goes active and the new data is present in the Input Data Register. Since a phase-mismatch interrupt will not occur, /REQ and /ACK need to be sampled to determine that the Target is attempting to send more data.

For send operations, the END OF DMA bit is set when the DMA finishes its transfers, but the SCSI transfer may still be in progress. If connected as a Target, /REQ and /ACK should be sampled until both are FALSE. If connected as an Initiator, a phase change interrupt is used to signal the completion of the previous phase. It is possible for the Target to request additional data for the same phase.

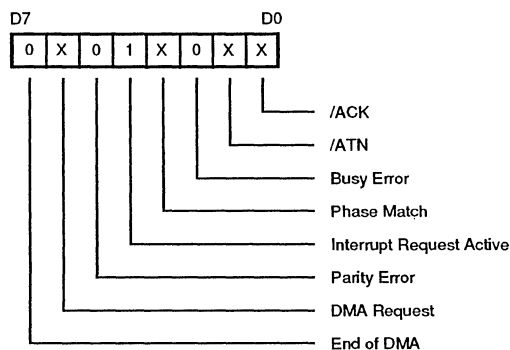


Figure 31. Bus and Status Register

In this case, a phase change will not occur and both /REQ and /ACK are sampled to determine when the last byte was transferred.

**SCSI Bus Reset.** The SCSI generates an interrupt when the /RST signal transitions to TRUE. The device releases all bus signals within a bus-clear delay of this transition. This interrupt also occurs after setting the ASSERT /RST bit (Initiator Command Register, bit 7). This interrupt cannot be disabled. (Note: /RST is not latched in bit 7 of the Current SCSI Bus Status Register and is not active when this port is read. For this case, the Bus Reset interrupt is determined by default.)

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 31 and 32, respectively.

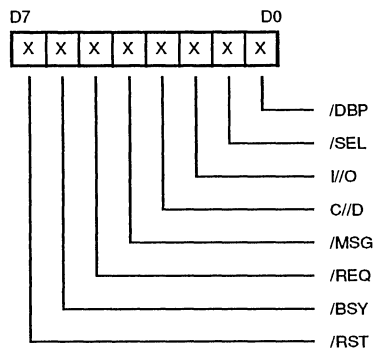


Figure 32. Current SCSI Bus Status Register

**Parity Error.** An interrupt is generated for a received parity error if the ENABLE PARITY CHECK (bit 5) and the ENABLE PARITY INTERRUPT (bit 4) bits are set (1) in the Mode Register. Parity is checked during a read of the Current SCSI Data Register and during a DMA receive operation. A parity error can be detected without generating an interrupt by disabling the ENABLE PARITY INTER-

RUPT bit and checking the PARITY ERROR flag (Bus and Status Register, bit 5).

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 33 and 34, respectively.



## SCSI FUNCTIONAL DESCRIPTION (Continued)

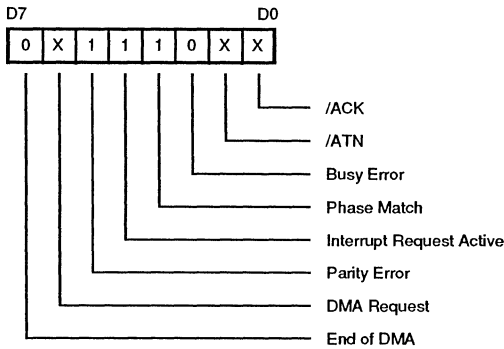


Figure 33. Bus and Status Register

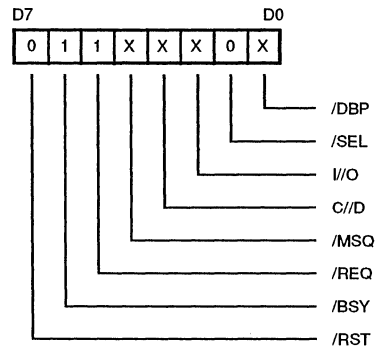


Figure 34. Current SCSI Bus Status Register

**Bus Phase Mismatch.** The SCSI phase lines have the signals I/O, C//D, and /MSG. These signals are compared with the corresponding bits in the Target Command Register: ASSERT I/O (bit 0), ASSERT C//D (bit 1), and ASSERT /MSG (bit 2). The comparison occurs continually and is reflected in the PHASE MATCH bit (bit 3) of the Bus and Status Register. If the DMA MODE bit (Mode Register, bit 1) is active and a phase mismatch occurs when /REQ transitions from FALSE to TRUE, an interrupt (IRQ) is generated.

A phase mismatch prevents the recognition of /REQ and removes the chip from the bus during an Initiator send

operation (/DB7-/DB0 and /DBP will not be driven even through the ASSERT DATA BUS bit (Initiator Command Register, bit 0) is active). This may be disabled by resetting the DMA MODE bit (Note: It is possible for this interrupt to occur when connected as a Target if another device is driving the phase lines to a different state).

The proper values for the Bus and Status Register and the Current SCSI Bus Status Register are displayed in Figures 35 and 36, respectively.

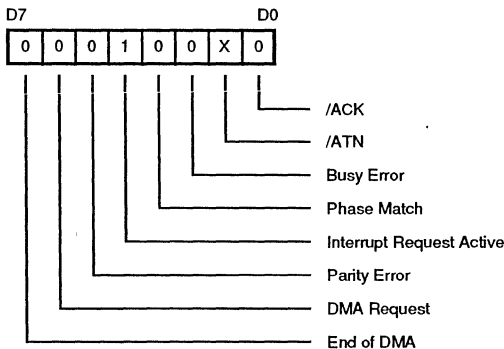


Figure 35. Bus and Status Register

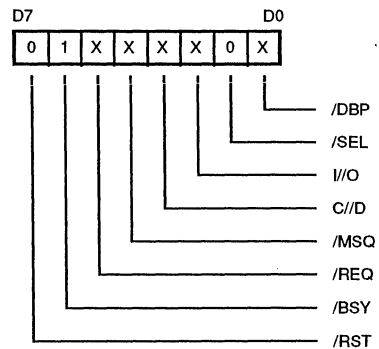


Figure 36. Current SCSI Bus Status Register

**Loss of BSY.** If the MONITOR BUSY bit (bit 2) in the Mode Register is active, an interrupt is generated if the BSY signal goes FALSE for at least a bus-settle delay. This

interrupt is disabled by resetting the MONITOR BUSY bit. Register values are displayed in Figures 37 and 38.

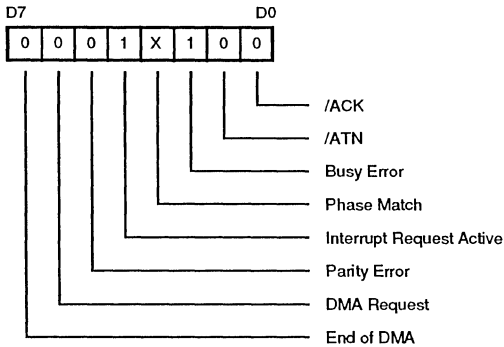


Figure 37. Bus and Status Register

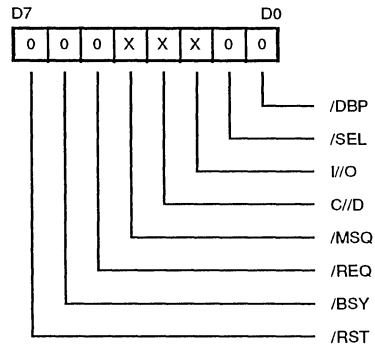


Figure 38. Current SCSI Bus Status Register

**Reset Conditions.** Three possible reset situations exist with the Z85C80, as follows:

**Hardware Chip Reset.** When the signal RST is active for at least 100 ns, the Z53C80 device is re-initialized and all internal logic and control registers are cleared. This is a chip reset only and does not create a SCSI Bus-Reset condition.

**SCSI Bus Reset (/RST) Received.** When a SCSI /RST signal is received, an IRQ interrupt is generated and a chip reset is performed. All internal logic and registers are cleared, except for the IRQ interrupt latch and the ASSERT /RST bit (bit 7) in the Initiator Command Register. (Note: The /RST signal may be sampled by reading the Current SCSI Bus Status Register, however, this signal is not latched and may not be present when this port is read).

**SCSI Bus Reset (/RST) Issued.** If the CPU sets the ASSERT/RST bit (bit 7) in the Initiator Command Register, the /RST signal goes active on the SCSI Bus and an internal reset is performed. Again, all internal logic and registers are cleared except for the IRQ interrupt latch and the ASSERT/RST bit (bit 7) in the Initiator Command Register. The /RST signal will continue to be active until the ASSERT/RST bit is reset or until a hardware reset occurs.

**Data Transfers.** Data is transferred between SCSI Bus devices in one of four modes: 1) Programmed I/O, 2) Normal DMA, or 3) Pseudo DMA. The following sections describe these modes in detail (Note: for all data transfer operations /DACK and /SCSICS should never be active simultaneously).

**Programmed I/O Transfers.** Programmed I/O is the most primitive form of data transfer. The /REQ and /ACK handshake signals are individually monitored and asserted by reading and writing the appropriate register bits. This type of transfer is normally used when transferring small blocks of data such as command blocks or message and status bytes. An Initiator send operation would begin by setting the C//D, I/O, and /MSG bits in the Target Command Register to the correct state so that a phase match exists. In addition to the phase match condition, it is necessary for the ASSERT DATA BUS bit (Initiator Command Register, bit 0) to be TRUE and the received I/O signal to be FALSE for the Z53C80 to send data. For each transfer, the data is loaded into the Output Data Register. The CPU then waits for the /REQ bit (Current SCSI Bus Status Register, bit 5) to become active. Once /REQ goes active, the PHASE MATCH bit (Initiator Command Register, bit 4) is set. The /REQ bit is sampled until it becomes FALSE and the CPU resets the ASSERT /ACK bit to complete the transfer.

**Normal DMA Mode.** DMA transfers are normally used for large block transfers. The SCSI chip outputs a DMA request (DRQ) whenever it is ready for a byte transfer. External DMA logic uses this DRQ signal to generate /DACK and a /RD or a /WR pulse to the Z53C80. DRQ goes inactive when /DACK is asserted and /DACK goes inactive some time after the minimum read or write pulse width. This process is repeated for every byte. For this mode, /DACK should not be allowed to cycle unless a transfer is taking place.

---

## SCSI FUNCTIONAL DESCRIPTION (Continued)

**Pseudo DMA Mode.** To avoid the tedium of monitoring and asserting the request/acknowledgement handshake signals for programmed I/O transfers, the system may be designed to implement a pseudo DMA mode. This mode is implemented by programming the Z53C80 to operate in the DMA mode, but using the CPU to emulate the DMA handshake. DRQ may be detected by polling the DMA REQUEST bit (bit 6) in the Bus and Status Register, by sampling the signal through an external port, or by using it to generate a CPU interrupt. Once DRQ is detected, the CPU can perform a read or write data transfer. This CPU read/write is externally decoded to generate the appropriate /DACK and /RD or /WR signals.

Often, external decoding logic is necessary to generate the /SCSICS signal. This same logic may be used to generate /DACK at no extra cost and provide an increased performance in programmed I/O transfers.

**Halting a DMA Operation.** The EOP signal is not the only way to halt a DMA transfer. A bus phase mismatch or a reset of the DMA MODE bit (Mode Register, bit 1) can also terminate a DMA cycle for the current bus phase.

**Using the /EOP Signal.** If /EOP is used, it should be asserted for at least 50 ns while /DACK and /RD or /WR are simultaneously active. Note, however, that if /RD or /WR is not active, an interrupt is generated, but the DMA activity continues. The /EOP signal does not reset the DMA MODE bit. Since the /EOP signal can occur during the last byte sent to the Output Data Register, the /REQ and /ACK signals are monitored to ensure that the last byte has transferred.

**Bus Phase Mismatch Interrupt.** A bus phase mismatch interrupt is used to halt the transfer if operating as an Initiator. Using this method frees the host from maintaining a data length counter and frees the DMA logic from providing the /EOP signal. If performing an Initiator send operation, the Z53C80 requires /DACK to cycle before /ACK goes inactive. Since phase changes cannot occur if /ACK is active, either /DACK must be cycled after the last byte is sent or the DMA MODE bit must be reset in order to receive the phase mismatch interrupt.

**Resetting the DMA MODE Bit.** A DMA operation may be halted at any time simply by resetting the DMA MODE bit. It is recommended that the DMA MODE bit be reset after receiving an /EOP or bus phase-mismatch interrupt. The DMA MODE bit must then be set before writing any of the start DMA registers for subsequent bus phases.

If resetting the DMA MODE bit is used instead of /EOP for Target role operation, then care must be taken to reset this bit at the proper time. If receiving data as a Target device, the DMA MODE bit must be reset once the last DRQ is received and before /DACK is asserted to prevent an additional /REQ from occurring. Resetting this bit causes DRQ to go inactive. However, the last byte received remains in the Input Data Register and may be obtained either by performing a normal CPU read or by cycling /DACK and /RD. In most cases, /EOP is easier to use when operating as a Target device.

## READ REGISTERS

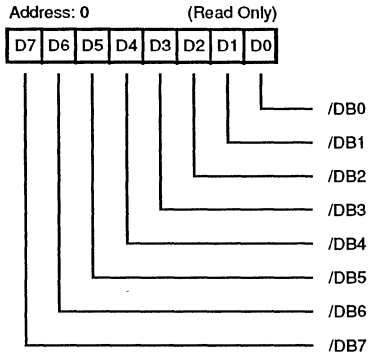


Figure 39. Current SCSI Data Register

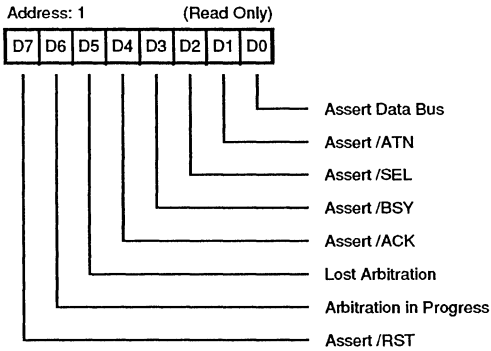


Figure 40. Initiator Command Register

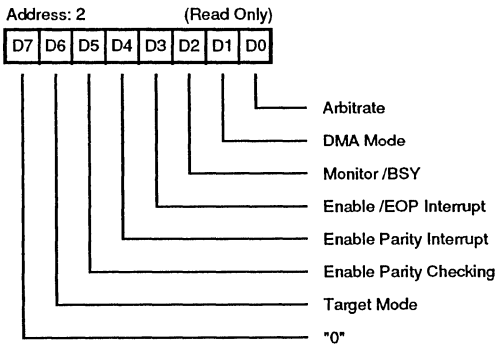


Figure 41. Mode Register

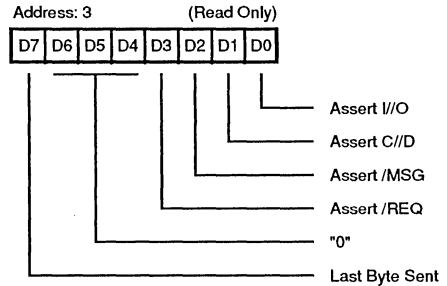


Figure 42. Target Command Register

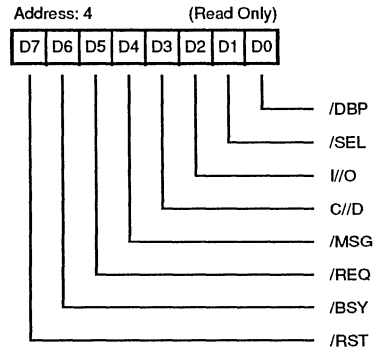


Figure 43. Current SCSI Bus Status Register

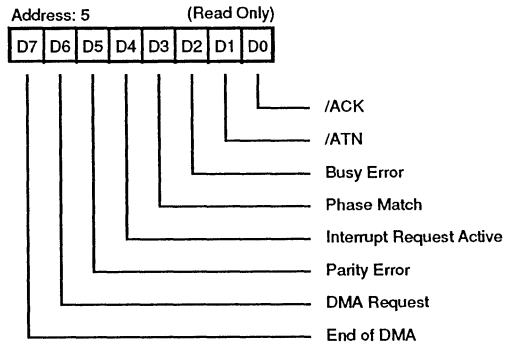
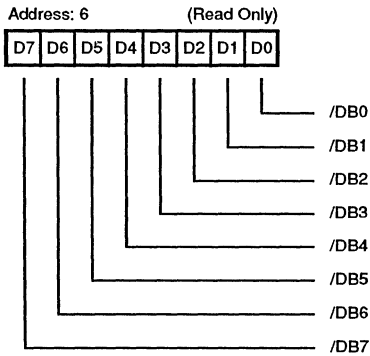
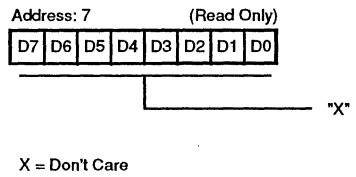


Figure 44. Bus and Status Register

**READ REGISTERS (Continued)**

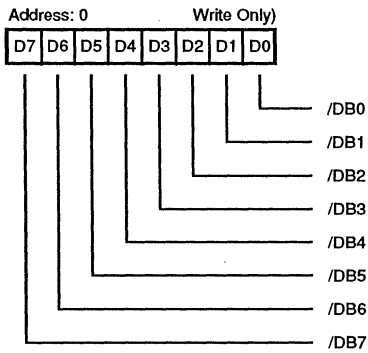


**Figure 45. Input Data Register**

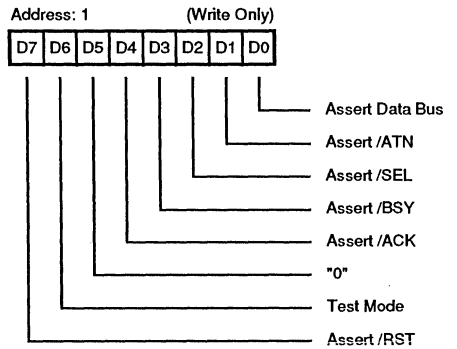


**Figure 46. Reset Parity/Interrupt**

**WRITE REGISTERS**



**Figure 47. Output Data Register**



**Figure 48. Initiator Command Register**

## WRITE REGISTERS (Continued)

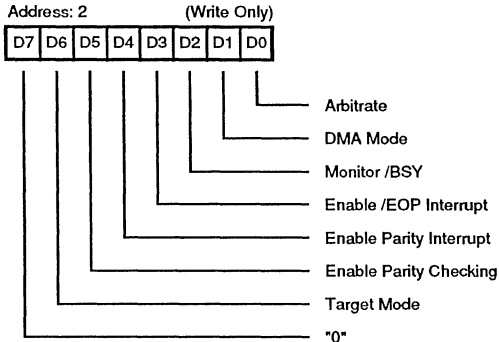


Figure 49. Mode Register

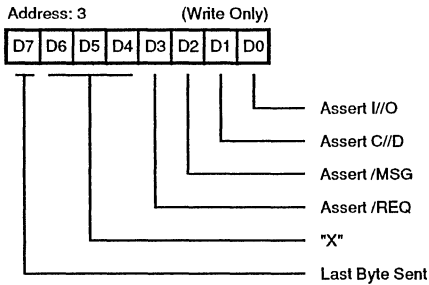


Figure 50. Target Command Register

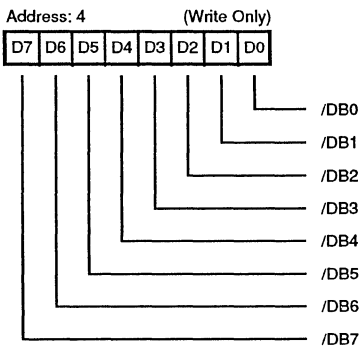


Figure 51. Select Enable Register

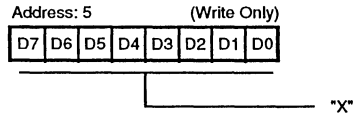


Figure 52. Start DMA Send

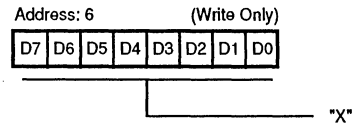


Figure 53. Start DMA Target Receive

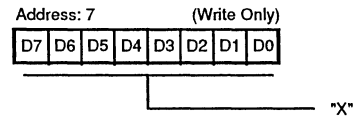


Figure 54. Start DMA Initiator Receive

Note: X = Don't care

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins with respect to GND .....	-0.3V to +7.0V
Operating Ambient Temperature .....	See Ordering Information
Storage Temperature .....	-65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to this device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC characteristics and capacitance section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

- $+4.75V \leq V_{CC} \leq +5.25V$
- $GND = 0V$
- $T_A$  as specified in Ordering Information

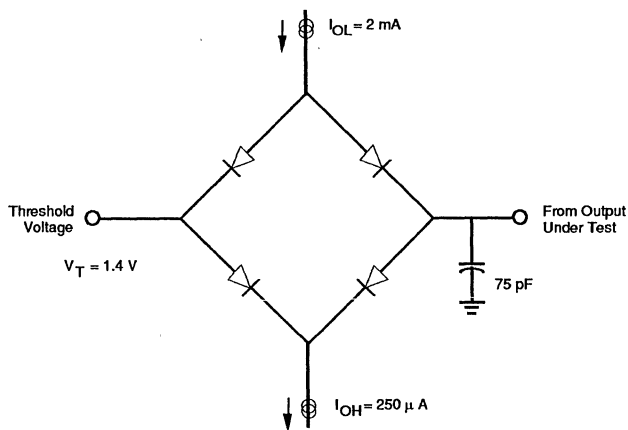


Figure 55. Standard Test Dynamic Load Circuit

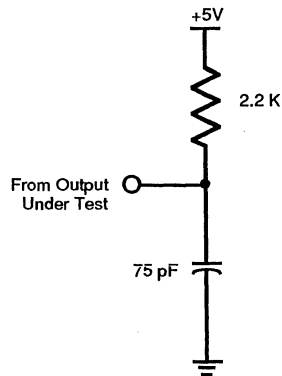


Figure 56. Open-Drain Test Load

## DC CHARACTERISTICS

Symbol	Parameter	Condition	Min	Max	Units
$V_{DD}$	Supply Voltage		4.75	5.25	V
$V_{IH}$	High-Level Input Voltage		2.0	5.5	V
$V_{IL}$	Low-Level Input Voltage		-0.3	0.8	V
$I_{IH1}$	High-Level Input Current SCSI Bus Pins	$V_{IH} = 5.5V$ $V_{IL} = 0V$		50	$\mu A$
$I_{IH2}$	High-Level Input Current All Other Pins	$V_{IH} = 5.5V$ $V_{IL} = 0V$		10	$\mu A$
$I_{IL1}$	Low-Level Input Current SCSI Bus Pins	$V_{IH} = 5.5V$ $V_{IL} = 0V$		-50	$\mu A$
$I_{IL2}$	Low-Level Input Current All Other Pins	$V_{IH} = 5.5V$ $V_{IL} = 0V$		-10	$\mu A$
$V_{OH1}$	High-Level Output Voltage	$I_{OH} = -3mA$	2.4		
$V_{OH2}$	High-Level Output Voltage	$I_{OH} = -250 \mu A$	$V_{DD} - 0.8$	V	
$V_{OL1}$	Low-Level Output Voltage SCSI Bus Pins	$I_{OL} = 48 mA$		0.5	V
$V_{OL2}$	Low-Level Output Voltage All Other Pins	$I_{OL} = 7 mA$		0.5	V
$I_{DD}$	Supply Current			40	mA
$C_{IN}$	Input Capacitance			10	pf
$C_{OUT}$	Output Capacitance			15	pf
$C_{IO}$	Bidirectional Capacitance			20	pf
$T_A$	Operating Free-Air Temperature		0	70	$^{\circ}C$



# AC CHARACTERISTICS

## General Timing

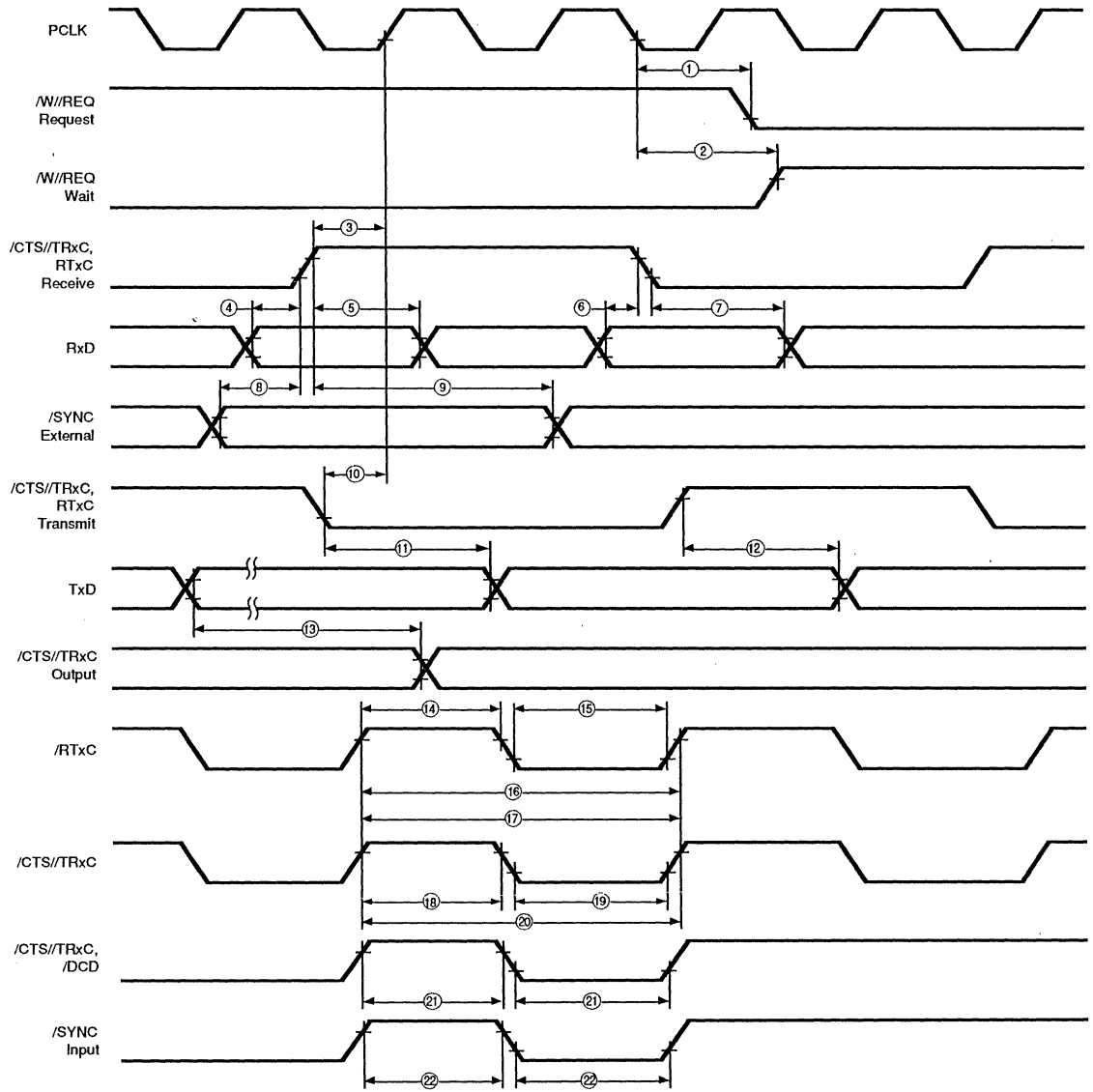


Figure 58. General Timing

**AC CHARACTERISTICS** (Continued)  
Z85C80 General Timing

No	Symbol	Parameter	Min	Max	Notes †
1	TdPC(REQ)	PCLK FALL to /W//REQ Valid Delay		200	
2	TdPC(W)	PCLK FALL to Wait Inactive Delay		300	
3	TsRXC(PC)	/RxC Rise to PCLK Rise Setup Time	N/A	N/A	[1,4]
4	TsRXD(RXCr)	RxD to /RxC Rise Setup Time	0		[1]
5	ThRXD(RXCr)	RxD to /RxC Rise Hold Time	125		[1]
6	TsRXD(RXCf)	RxD to /RxC FALL Setup Time	0		[1,5]
7	ThRXD(RXCf)	RxD to /RxC FALL Hold Time	125		[1,5]
8	TsSY(RXC)	/SYNC to /RxC Rise Setup Time	-150		[1]
9	ThSY(RXC)	/SYNC to /RxC Rise Hold Time	5TcPc		[1]
10	TsTXC(PC)	/TxC FALL to PCLK Rise Setup Time	N/A		[2,4]
11	TdTXC(TXD)	/TxC FALL to TxD Delay		150	[2]
12	TdTxCr(TXD)	/TxC Rise to TxD Delay		150	[2,5]
13	TdTxD(TRX)	TxD to /TRxC Delay		140	
14	TwRTXh	/RTxC High Width	120		[6]
15	TwRTXI	/RTxC Low Width	120		[6]
16a	TcRTX	/RTxC Cycle Time	400		[6,7]
16b	TxRX(DPLL)	DPLL Cycle Time	50		[7,8]
17	TcRTXX	Crystal Oscillator Period	100	1000	[3]
18	TwTRXh	/TRxC High Width	120		[6]
19	TwTRXI	/TRxC Low Width	120		[6]
20	TcTRX	/TRxC Cycle Time	400		[6,7]
21	TwEXT	/DCD or /CTS Pulse Width	120		
22	TwSY	/SYNC Pulse Width	120		

**Notes:**

- [1] /RxC is /RTxC or TRxC, whichever is supplying the receive clock.
- [2] /TxC is /TRxC or RTxC, whichever is supplying the transmit clock.
- [3] Both /RTxC and /SYNC have 300 pF capacitors to ground connected to them.
- [4] Synchronization of /RxC to PCLK is eliminated in divide by four operation.
- [5] Parameter applies only to FM encoding/decoding.
- [6] Parameter applies only for transmitter and receiver; DPLL and baud rate timing requirements are identical to case PCLK requirements.
- [7] The maximum receive or transmit data is 1/4 PCLK.
- [8] Applies to DPLL clock source only. Maximum data rate of 1/4 PCLK still applies. DPLL clock should have a 50% duty cycle.

† Units in nanoseconds (ns)

**AC CHARACTERISTICS**  
Z85C80 System Timing

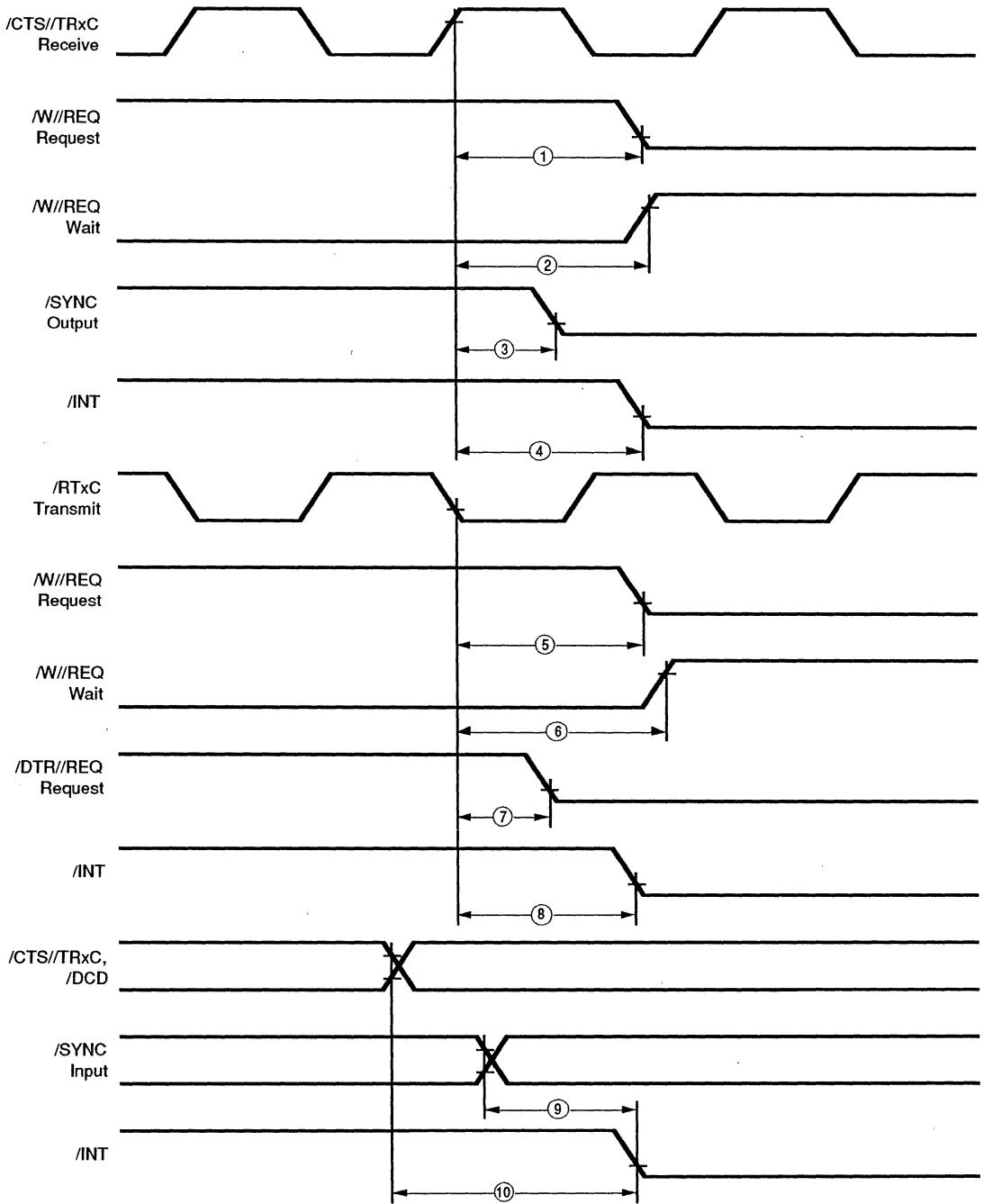


Figure 59. System Timing

**AC CHARACTERISTICS** (Continued)  
Z85C80 System Timing

No	Symbol	Parameter	Min	Max	Notes †
1	TdRXC(REQ)	/RxC Rise to /W//REQ Valid	8	12	[2]
2	TdRXCW)	/RxC Rise to Wait Inactive	8	14	[1,2]
3	TdRXC(SY)	/RxC Rise to /SYNC Valid	4	7	[2]
4	TdRXC(INT)	/RxC Rise to /INT Valid Delay	10	16	[1,2]
5	TdTXC(REQ)	/TxC Fall to /W//REQ	5	8	[3]
6	TdTXC(W)	/TxC Fall to Wait Inactive	5	11	[1,3]
7	TdTXC(DRQ)	/TxC Fall to /DTR//REQ Valid	4	7	[3]
8	TdTXC(INT)	/TxC Fall to /INT Valid	6	10	[1,3]
9	TdSY(INT)	/SYNC to /INT Valid	2	6	[1]
10	TdEXT(INT)	/DCD or /CTS//TRxC to /INT Valid	2	6	[1]

**Notes:**

- [1] Open-drain output measured with open-drain test load.
- [2] /RxC is /RTxC or /CTS//TRxC, whichever is supplying the receive clock.
- [3] /TxC is /CTS//TRxC or RTxC, whichever is supplying the transmit clock.

† Units equal to TcPC

**AC CHARACTERISTICS**  
Z85C80 Additional Timing

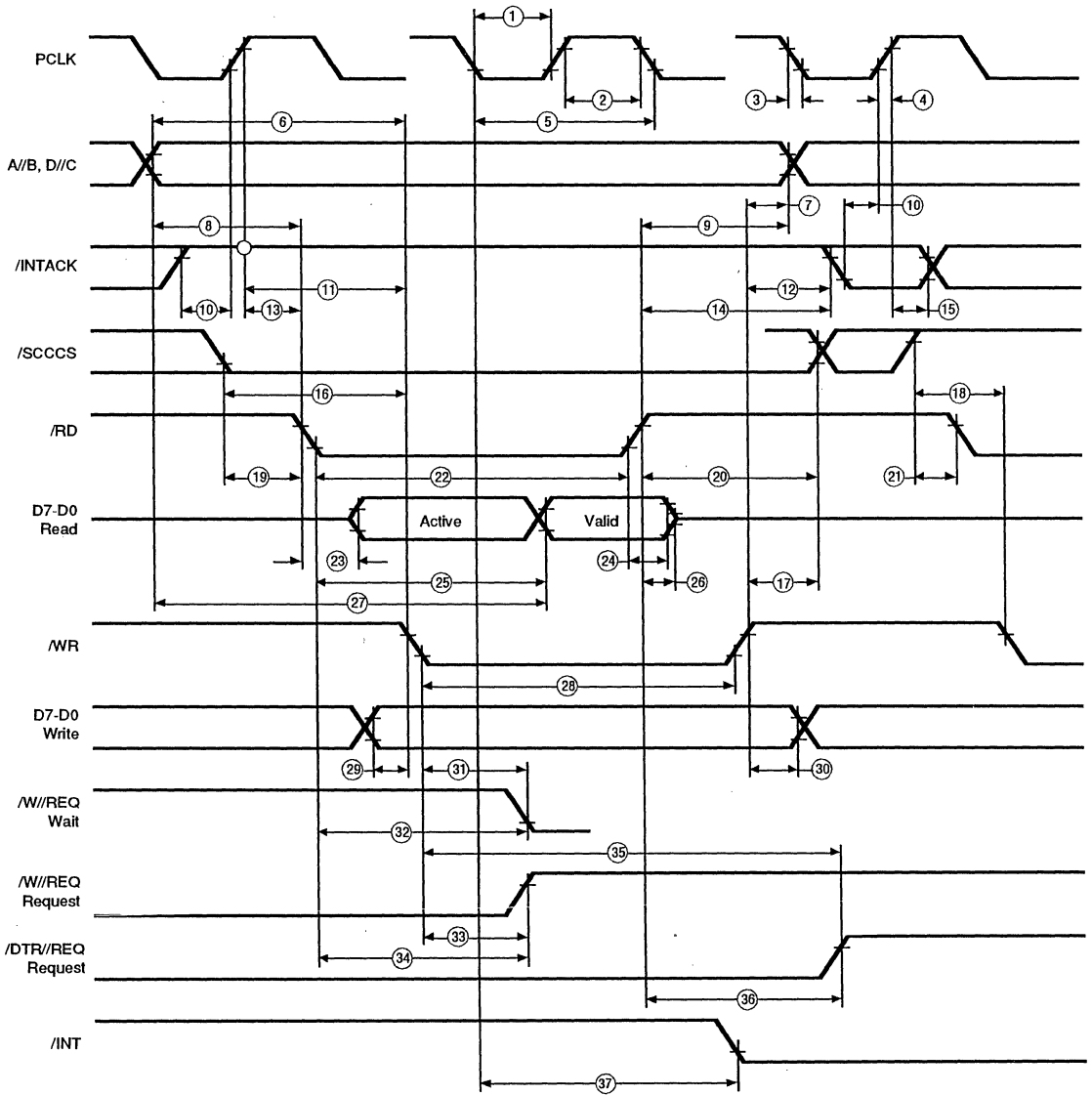


Figure 60. Read/Write Timing

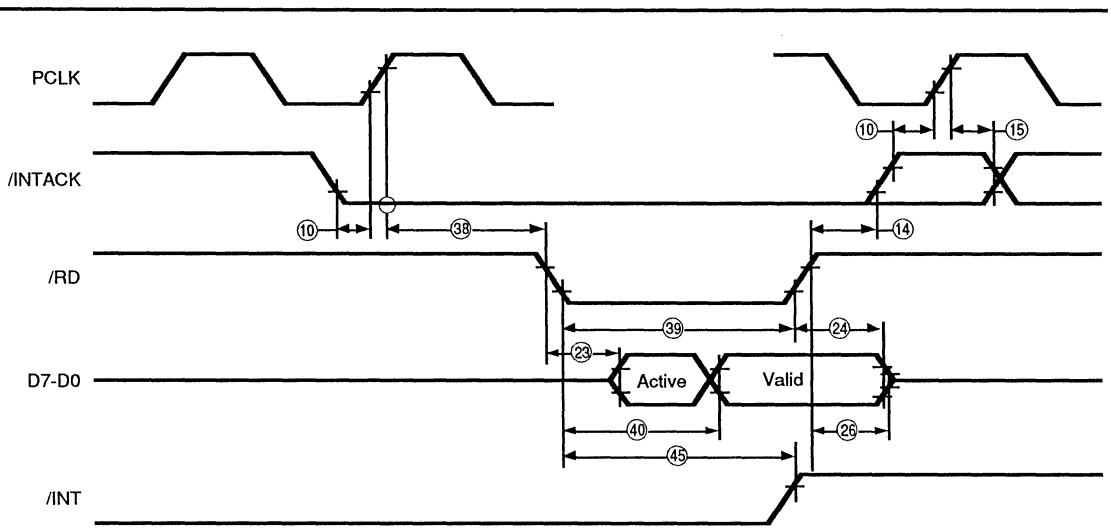


Figure 61. Interrupt Acknowledge Timing

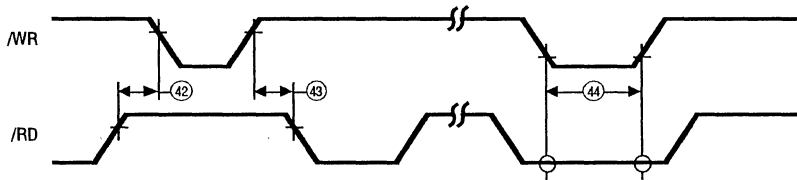


Figure 62. Reset Timing

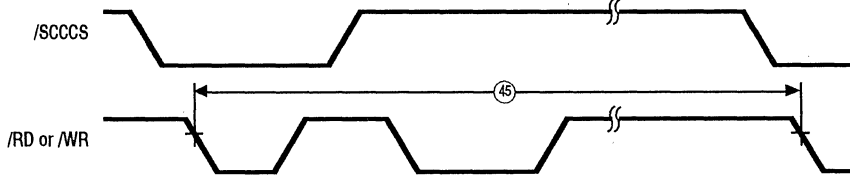


Figure 63. Cycle Timing

## AC CHARACTERISTICS

### Additional Timing

No	Symbol	Parameter	Min	Max	Notes †
1	TwPCI	PCLK Low Width	40	1000	
2	TwPCh	PCLK High Width	40	1000	
3	TfPC	PCLK Fall Time		10	
4	TrPC	PCLK Rise Time		10	
5	TcPC	PCLK Cycle Time	100	2000	
6	TsA(WR)	Address to /WR Fall Setup Time	50		
7	ThA(WR)	Address to /WR Rise Hold Time	0		
8	TsA(RD)	Address to /RD Fall Setup Time	50		
9	ThA(RD)	Address to /RD Rise Hold Time	0		
10	TsIA(PC)	/INTACK to PCLK Rise Setup Time	20		
11	TsIAi(WR)	/INTACK to /WR Fall Setup Time	130		[1]
12	ThIA(WR)	/INTACK to /WR Rise Hold Time	0		
13	TsIAi(RD)	/INTACK to /WR Fall Setup Time	130		[1]
14	ThIA(RD)	/INTACK to /RD Rise Hold Time	0		
15	ThIA(PC)	/INTACK to PCLK Rise Hold Time	30		
16	TsCEI(WR)	/SCCCS Low to /WR Fall Setup Time	0		
17	ThCE(WR)	/SCCCS to /WR Rise Hold Time	0		
18	TsCEh(WR)	/SCCCS High to /WR Fall Setup Time	50		
19	TsCEI(RD)	/SCCCS Low to /RD Fall Setup Time	0		[1]
20	ThCE(RD)	/SCCCS to /RD Rise Hold Time	0		[1]
21	TsCEh(RD)	/SCCCS High to /RD Fall Setup Time	50		[1]
22	TwRDI	/RD Low Width	125		[1]
23	TdRD(DRA)	/RD Fall to Read Data Active Delay	0		
24	TdRD <sub>r</sub> (DR)	/RD Rise to Read Data Not Valid Delay	0		
25	TdRDI(DR)	/RD Fall to Read Data Valid Delay		120	
26	TdRD(DR <sub>z</sub> )	/RD Rise to Read Data Float Delay		35	
27	TdA(DR)	Address to Read Data Valid Delay		180	
28	TwWRI	/WR Low Width	125		
29	TsDW(WR)	Write Data to /WR Fall Setup Time	10		
30	ThDW(WR)	Write Data to /WR Rise Hold Time	0		
31	TdWR(W)	/WR Fall to Wait Valid Delay		160	[2]
32	TdRD(W)	/RD Fall to Wait Valid Delay		160	[2]
33	TdWRI(REQ)	/WR Fall to /W//REQ Not Valid Delay		160	
34	TdRDI(REQ)	/RD Fall to /W//REQ Not Valid Delay		160	
35	TdWRI(REQ)	/WR Fall /DTR//REQ Not Valid Delay		4TcPC	
36	TdRD <sub>r</sub> (REQ)	/RD Rise to /DTR//REQ Not Valid Delay		N/A	
37	TdPC(INT)	PCLK Fall to /INT Valid Delay		450	
38	TdIAiRD)	/INTACK to /RD Fall (Acknowledge) Delay	125		[3]
39	TwRDA	/RD (Acknowledge) Width	125		[3]
40	TdRDA(DR)	/RD Fall (Acknowledge) to Read Data Valid Delay	120		
41	TdRDA(INT)	/RD Fall to /INT Inactive Delay		320	[2]
42	TdRD(WRQ)	/RD Rise to /WR Fall Delay for No Reset	15		

---

**AC CHARACTERISTICS** (Continued)  
Additional Timing

No	Symbol	Parameter	Min	Max	Notes †
43	TdWRQ(RD)	/WR Rise to /RD Fall Delay for No Reset	15		
44	TwRES	/WR and /RD Coincident Low for Reset	100		
45	Trc	Valid Access Recovery Time	3.5TcPc		[1]

**Notes:**

- [1] Parameter is guaranteed by design and does not apply to Interrupt Acknowledge transactions.
- [2] Open-drain output, measured with open-drain test load.
- [3] Parameter is system dependent.

† Units in nanoseconds (ns)



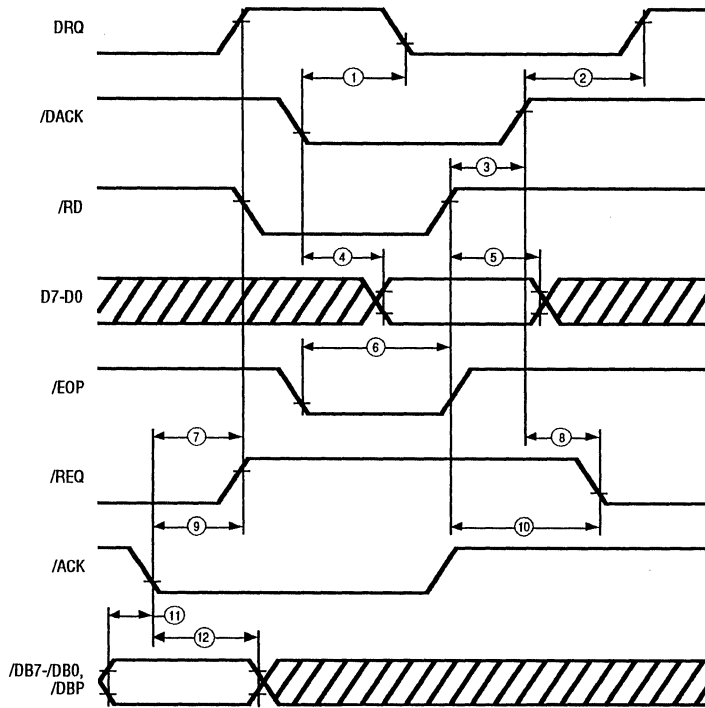


Figure 64. DMA Read Target Receive Cycle

## AC CHARACTERISTICS

### DMA Read Target Receive Cycle

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low		60	ns
2	/DACK High to DRQ High	30		ns
3	/DACK Hold Time from End of /RD	0		ns
4	Data Access Time from Read Enable*		70	ns
5	Data Hold Time from End of /RD	10		ns
6	Width of /EOP Pulse [1]	50		ns
7	/ACK Low to DRQ High		70	ns
8	/DACK High to /REQ Low (/ACK High)		90	ns
9	/ACK Low to /REQ High		80	ns
10	/ACK High to /REQ Low (/DACK High)		100	ns
11	Data Setup Time to /ACK	20		ns
12	Data Hold Time from /ACK	30		ns

#### Notes:

[1] /EOP, /RD, and /DACK must be concurrently Low for at least T6 for proper recognition of the /EOP pulse.

\* Read Enable is the occurrence of /RD and /DACK.

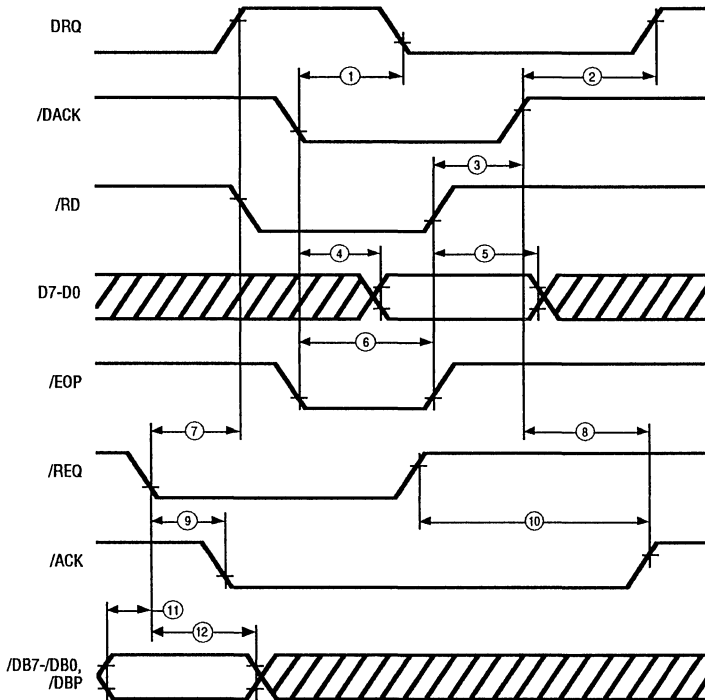


Figure 65. DMA Read Initiator Receive Cycle

## AC CHARACTERISTICS

### DMA Read Initiator Receive Cycle

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low		60	ns
2	/DACK High to DRQ High	30		ns
3	/DACK Hold Time from End of /RD	0		ns
4	Data Access Time from Read Enable*		70	ns
5	Data Hold Time from End of /RD	10		ns
6	Width of /EOP Pulse [1]	50		ns
7	/REQ Low to DRQ High		130	ns
8	/DACK High to /ACK High (/REQ High)		90	ns
9	/REQ Low to /ACK Low		130	ns
10	/REQ High to /ACK High (/DACK High)		80	ns
11	Data Setup Time to /REQ	20		ns
12	Data Hold Time from /REQ	110		ns

#### Notes:

[1] /EOP, /RD, and /DACK must be concurrently Low for at least T6 for proper recognition of the /EOP pulse.

\* Read Enable is the occurrence of /RD and /DACK.

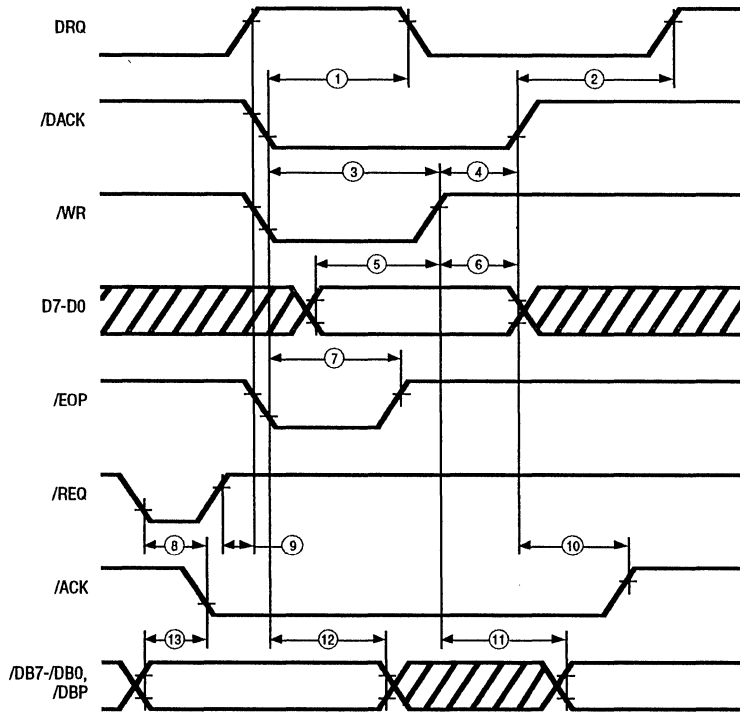


Figure 66. DMA Write Initiator Send Cycle

## AC CHARACTERISTICS

### DMA Write Initiator Send Cycle

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low		60	ns
2	/DACK High to DRQ High	30		ns
3	Write Enable Width*	50		ns
4	/DACK Hold from End of /WR	0		ns
5	Data Setup to End of Write Enable*	50		ns
6	Data Hold Time from End of /WR	25		ns
7	Width of /EOP Pulse [1]	50		ns
8	/REQ Low to /ACK Low		130	ns
9	/REQ High to DRQ High		70	ns
10	/DACK High to /ACK High		90	ns
11	/WR High to Valid SCSI Data		50	ns
12	Data Hold from Write Enable*			ns
13	Data Setup to /ACK Low	15		ns

#### Notes:

[1] /EOP, /WR, and /DACK must be concurrently Low for at least T7 for proper recognition of the /EOP pulse.

\* Write Enable is the occurrence of /WR and /DACK.

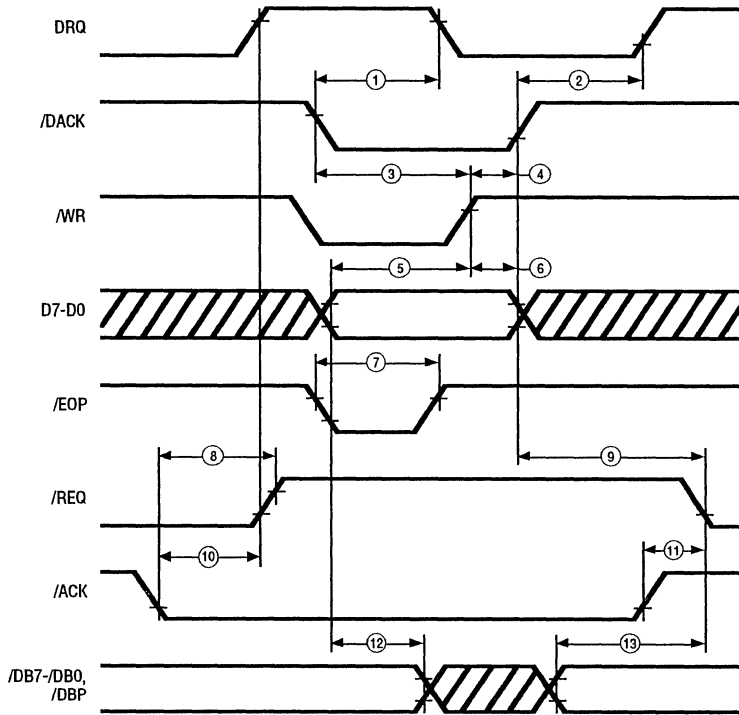


Figure 67. DMA Write Target Send Cycle

## AC CHARACTERISTICS

### DMA Write Target Send Cycle

No	Description	Min	Max	Units
1	DRQ Low from /DACK Low		60	ns
2	/DACK High to DRQ High	30		ns
3	Write Enable Width*	50		ns
4	/DACK Hold from /WR High	0		ns
5	Data Setup to End of Write Enable*	50		ns
6	Data Hold Time from End of /WR	25		ns
7	Width of /EOP Pulse [1]	50		ns
8	/ACK Low to /REQ High		80	ns
9	/REQ from End of /DACK (/ACK High)		90	ns
10	/ACK Low to DRQ High (Target)		70	ns
11	/ACK High to /REQ Low (/DACK High)		100	ns
12	Data Hold from Write Enable	15		ns
13	Data Setup to /REQ Low (Target)	55		ns

#### Notes:

[1] /EOP, /WR, and /DACK must be concurrently Low for at least T7 for proper recognition of the /EOP pulse.

\* Write Enable is the occurrence of /IOW and /DACK

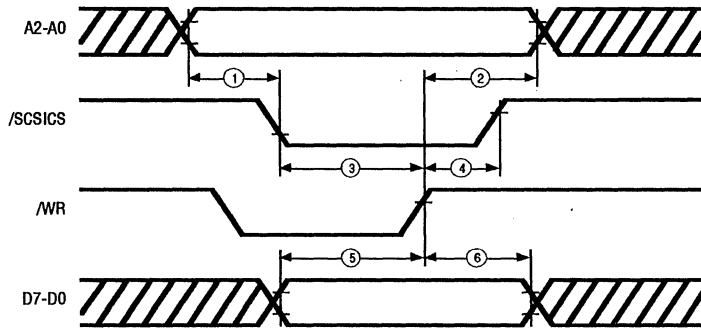


Figure 68. CPU Write Cycle

## AC CHARACTERISTICS

### CPU Write Cycle

No	Description	Min	Max	Units
1	Address Setup to Write Enable*	10		ns
2	Address Hold from End Write Enable*	10		ns
3	Write Enable Width*	40		ns
4	Chip Select Hold from End of /IOW	0		ns
5	Data Setup to end of Write Enable*	20		ns
6	Data Hold Time from End of /IOW	20		ns

**Note:**

\* Write Enable is the occurrence of /WR and /SCSICS

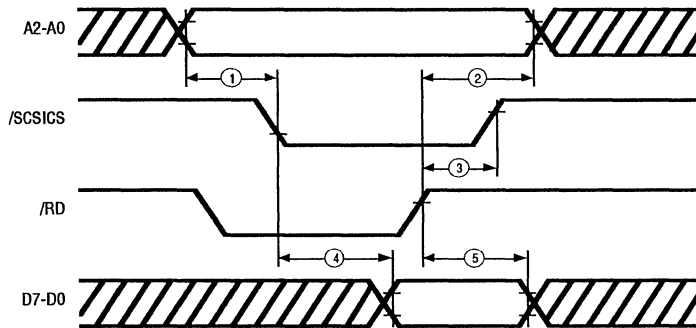


Figure 69. CPU Read Cycle

## AC CHARACTERISTICS

### CPU Read Cycle

No	Description	Min	Max	Units
1	Address Setup to Read Enable*	10		ns
2	Address Hold from End Read Enable*	10		ns
3	Chip Select Hold from End of /RD	0		ns
4	Data Access Time from Read Enable*		70	ns
5	Data Hold Time from End of Read Enable*	10		ns

**Note:**

\* Read Enable is the occurrence of /RD and /SCSICS

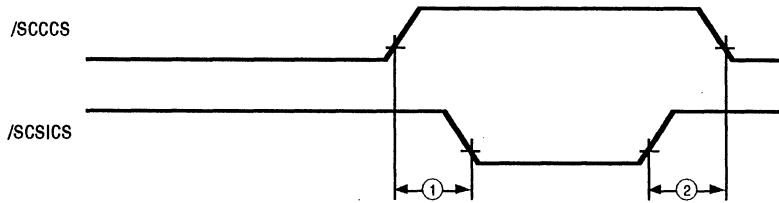


Figure 70. Selection

## AC CHARACTERISTICS

### Selection

No	Description	Min	Max	Units
1	/SCCS to /SCSICS	100		ns
2	/SCSICS to /SCCS	100		ns

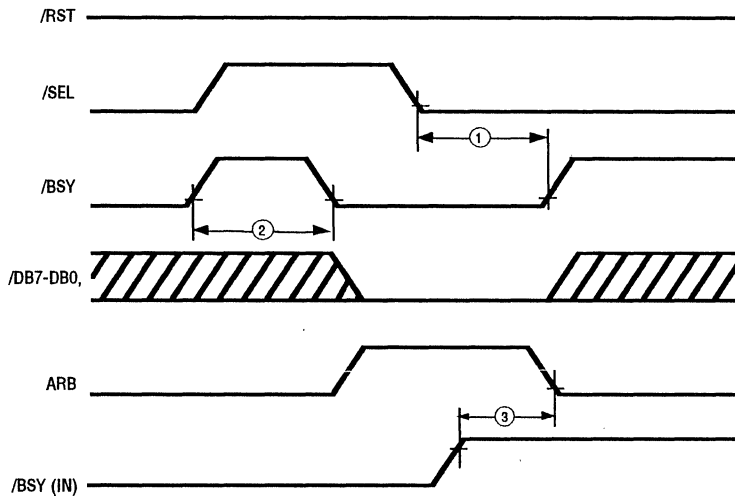


Figure 71. Arbitration

## AC CHARACTERISTICS

### Arbitration

No	Description	Min	Max	Units
1	Bus Clear from /SEL Low		600	ns
2	Arbitrate Start from /BSY False	1200	2200	ns
3	Bus Clear from /BSY High		1100	ns

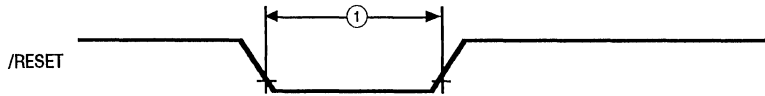


Figure 72. Reset

## AC CHARACTERISTICS

### Reset

No	Description	Min	Max	Units
1	Minimum Width of /RESET	100		ns



---



# Z16C30

## CMOS USC

### UNIVERSAL SERIAL CONTROLLER

#### FEATURES

- Two independent, 0 to 10Mbit/sec, full duplex channels, each with two baud rate generators and one digital phase-locked loop for clock recovery.
- 32-byte data FIFO's for each receiver and transmitter
- 12.5 MByte/sec (16-bit) data bus bandwidth
- Multi-protocol operation under program control with independent mode selection for receiver and transmitter.
- Async mode with one to eight bits/character, 1/16 to 2 stop bits/character in 1/16 bit increments; programmable clock factor; break detect and generation; odd, even, mark, space or no parity and framing error detection. Supports one Address/Data bit and MIL STD 1553B protocols.
- Byte oriented synchronous mode with one to eight bits/character; programmable idle line condition; optional receive sync stripping; optional preamble transmission; 16- or 32-bit CRC and transmit-to-receive slaving (for X.21).
- Bisync mode with 2- to 16-bit programmable sync character; programmable idle line condition; optional receive sync stripping; optional preamble transmission; 16- or 32-bit CRC.
- Transparent Bisync mode with EBCDIC or ASCII character code; automatic CRC handling; programmable idle line condition; optional preamble transmission; automatic recognition of DLE, SYN, SOH, ITX, ETX, ETB, EOT, ENQ and ITB.
- External character sync mode for receive
- HDLC/SDLC mode with eight bit address compare; extended address field option; 16- or 32-bit CRC; programmable idle line condition; optional preamble transmission and loop mode.
- DMA interface with separate request and acknowledge for each receiver and transmitter.
- Channel load command for DMA controlled initialization.
- Flexible bus interface for direct connection to most microprocessors; user programmable for 8 or 16 bits wide. Directly supports 680X0 family or 8X86 family bus interfaces.
- Low power CMOS
- 68-pin PLCC package

#### GENERAL DESCRIPTION

The USC Universal Serial Controller is a dual-channel multi-protocol data communications peripheral designed for use with any conventional multiplexed or non-multiplexed bus. The USC functions as a serial-to-parallel, parallel-to-serial converter/controller and may be software configured to satisfy a wide variety of serial communications applications. The device contains a variety of new, sophisticated internal functions including two baud rate generators per channel, a digital phase-locked loop per channel, character counters for both receive and transmit in each channel and 32-byte data FIFO's for each receiver and transmitter.

Zilog now offers a high speed version of the USC, the 16C3020VSC, with faster clock and data rates. The new USC has improved transmit and receive clocks to 20MHz and doubles data transfer rates from 10Mbits/sec to 20Mbits/sec full duplex. Bus timing has changed to improve bus bandwidth at these data rates. CPU bus accesses have been shortened from 160ns per access to 110ns per access. Zilog will continue to offer the 10 Mbits/sec USC, the Z16C3010VSC.

## GENERAL DESCRIPTION (Continued)

The USC handles asynchronous formats, synchronous byte-oriented formats such as BISYNC and synchronous bit-oriented formats such as HDLC. This device supports virtually any serial data transfer application.

The device can generate and check CRC in any synchronous mode and can be programmed to check data integrity in various modes. The USC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls may be used for general-purpose I/O. The same is true for most of the other pins in each channel.

Interrupts are supported with a daisy-chain hierarchy, with the two channels having completely separate interrupt structures.

High-speed data transfers via DMA are supported by a Request/Acknowledge signal pair for each receiver and

transmitter. The device supports automatic status transfer via DMA and also allows device initialization under DMA control.

To aid the designer in efficiently programming the USC, support tools are available. The Technical Manual describes in detail all features presented in this Product Specification and gives programming sequence hints. The Programmer's Assistant is a MS-DOS disk-based programming initialization tool to be used in conjunction with the Technical Manual. There are also available assorted application notes and development boards to assist the designer in the hardware/software development.

**Note:** All Signals with a preceding front slash, "/", are active Low, e.g.: B/W (WORD is active Low); /B/W (BYTE is active Low, only); /N/S (NORMAL and SYSTEM are both active Low).

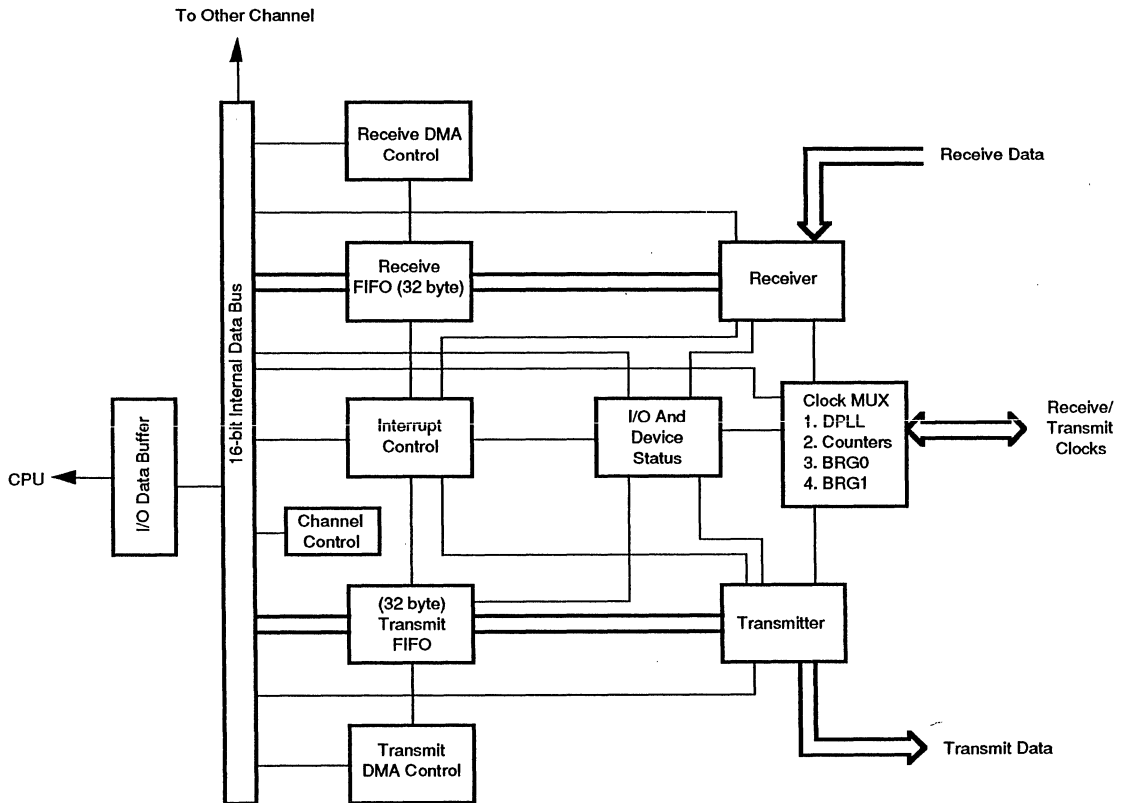


Figure 1. USC Block Diagram

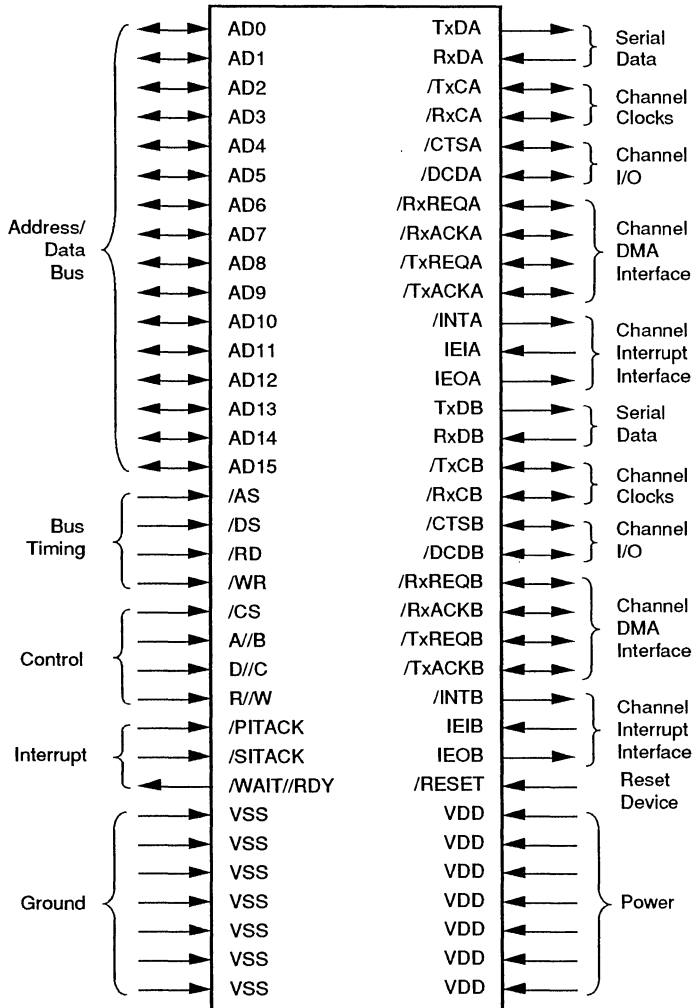
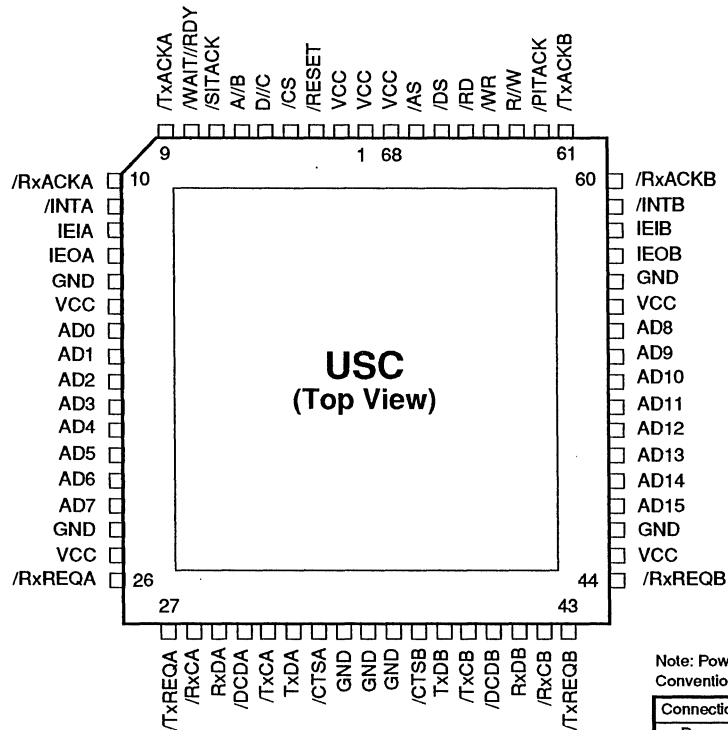


Figure 2. Pin Functions



Note: Power connections follow Conventional descriptions below

Connection	Circuit	Device
Power	V <sub>CC</sub>	V <sub>DD</sub>
Ground	GND	V <sub>SS</sub>

Figure 3. Pin Assignments

## PIN DESCRIPTION

The device contains 13 pins per channel for channel I/O, 16 pins for address and data, 12 pins for CPU handshake and 14 pins for power and ground.

Three separate bus interface types are available for the device. The Bus Configuration Register (BCR) and external connections to the AD bus control selection of the bus type.

A 16-bit bus is selected by setting BCR bit 2 to a 1.

The 8-bit bus is selected by setting BCR bit 2 to zero and tying AD15 - AD8 to VSS.

The 8-bit bus with separate address is selected by setting BCR bit 2 to zero and, during the BCR write, forcing AD15 to a 1 and forcing AD14-AD8 to zero.

The multiplexed bus is selected for the USC if there is an Address Strobe prior to or during the transaction which

writes the BCR. If no Address Strobe is present prior to or during the transaction which writes the BCR, a non-multiplexed bus is selected (See Figure 6).

The section below describes in detail the USC pin assignment.

**/RESET.** *Reset* (input, active Low). This signal resets the device to a known state. The first write to the USC after a reset accesses the BCR to select additional bus options for the device.

**/AS.** *Address Strobe* (input, active Low). This signal is used in the multiplexed bus modes to latch the address on the AD lines. The /AS signal is not used in the non-multiplexed bus modes and should be tied to VDD.

**/DS.** *Data Strobe* (input, active Low). This signal strobes data out of the device during a read and may strobe an interrupt vector out of the device during an interrupt

---

acknowledge cycle. */DS* also strobes data into the device on the state of *R//W*.

*/RD*. *Read Strobe* (input, active Low). This signal strobes data out of the device during a read and may strobe an interrupt vector out of the device during an interrupt acknowledge cycle.

*/WR*. *Write Strobe* (input, active Low). This signal strobes data into the device during a write.

*R//W*. *Read/Write* (input). This signal determines the direction of data transfer for a read or write cycle in conjunction with */DS*.

*/CS*. *Chip Select* (input, active Low). This signal selects the device for access and must be asserted for read and write cycles, but is ignored during interrupt acknowledge and fly-by DMA transfers. In the case of a multiplexed bus interface, */CS* is latched by the rising edge of */AS*.

*A//B*. *Channel A/Channel B Select* (input). This signal selects between the two channels in the device. High selects channel A and Low selects channel B. This signal is sampled and the result is latched during the BCR (Bus Configuration Register) write. It programs the sense of the */WAIT//RDY* signal appropriate for different bus interfaces. See */WAIT//RDY* below.

*D//C*. *Data/Control Select* (input). This signal, when High, provides for direct access to the RDR and TDR. In the case of a multiplexed bus interface, *D//C* High overrides the address provided to the device.

*/SITACK*. *Status Interrupt Acknowledge* (input, active Low). This signal is a status signal that indicates that an interrupt acknowledge cycle is in progress. The device is capable of returning an interrupt vector that may be encoded with the type of interrupt pending during this acknowledge cycle. This signal is compatible with 680X0 family microprocessors.

*/PITACK*. *Pulsed Interrupt Acknowledge* (input, active Low). This signal is a strobe signal that indicates that an interrupt acknowledge cycle is in progress. The device is capable of returning an interrupt vector that may be encoded with the type of interrupt pending during this acknowledge cycle. */PITACK* may be programmed to accept a single pulse or double pulse acknowledge type. This programming is done in the BCR. With the double pulse type selected, the first */PITACK* is recognized but no action takes place. The interrupt vector is returned on the second pulse if the no vector option is not selected. The double pulse type is compatible with 8X86 family microprocessors.

*/WAIT//RDY*. *Wait/Data Ready* (output, active Low). This signal serves to indicate when the data is available during a read cycle, when the device is ready to receive data during a write cycle, and when a valid vector is available during an interrupt acknowledge cycle. It may be programmed to function either as a Wait signal or a Ready signal using the state of the *A//B* pin during the BCR write. When *A//B* is High during the BCR write, this signal functions as a wait output and thus supports the READY function of 8X86 family microprocessors. When *A//B* is Low during the BCR write, this signal functions as a ready output and thus supports the DTACK function of 680X0 family microprocessors.

**AD15-AD0**. *Address/Data Bus* (bidirectional, active High, 3-state). The AD signals carry addresses to, and data to and from, the device. When the 16-bit non-multiplexed bus is selected, AD15-0 carry data to and from the device. Addresses are provided using a pointer within the device that is loaded with the desired register address. When selecting the 8-bit non-multiplexed bus (without separate address) only AD7-0 are used to transfer data. The pointer is used for addressing, with AD15-8 unused. When selecting the 8-bit non-multiplexed bus (with separate address), AD7-0 are used to transfer data with AD15-8 used as address bus. When the 16-bit multiplexed bus is selected, addresses are latched from AD7-0 and data transfers are sixteen bits wide. When selecting the 8-bit multiplexed bus (without separate address) only AD7-0 are used to transfer addresses and data, with AD15-8 unused. When the 8-bit multiplexed bus with separate address is selected, only AD7-0 are used to transfer data, while AD15-8 are used as an address bus.

*/INTA*, */INTB*. *Interrupt Request* (outputs, active Low). These signals indicate that the channel has an interrupt condition pending and is requesting service. These outputs are NOT open-drain.

*IEIA*, *IEIB*. *Interrupt Enable In* (inputs, active High). The IEI signal for each channel is used with the accompanying IEO signal to form an interrupt daisy chain. An active IEI indicates that no device having higher priority is requesting or servicing an interrupt.

*IEOA*, *IEOB*. *Interrupt Enable Out* (outputs, active High). The IEO signal for each channel is used with the accompanying IEI signal to form an interrupt daisy chain. IEO is Low if IEI is Low, an interrupt is under service in the channel, or an interrupt is pending during an interrupt acknowledge cycle.

*/TxACKA*, */TxACKB*. *Transmit Acknowledge* (inputs or outputs, active Low). The primary function of these signals is to perform fly-by DMA transfers to the transmit FIFOs. They may also be used as bit inputs or outputs.

---

**/RxACKA, /RxACKB.** *Receive Acknowledge* (inputs or outputs, active Low). The primary function of these signals is to perform fly-by DMA transfers from the receive FIFOs. They may also be used as bit inputs or outputs.

**TxDA, TxDB.** *Transmit Data* (outputs, active High, 3-state). These signals carry the serial transmit data for each channel.

**RxDA, RxDB.** *Receive Data* (inputs, active High). These signals carry the serial receive data for each channel.

**/TxCA, /TxCB.** *Transmit Clock* (inputs or outputs, active Low). These signals are used as clock inputs for any of the functional blocks within the device. They may also be used as outputs for various transmitter signals or internal clock signals.

**/RxCA, /RxCB.** *Receive Clock* (inputs or outputs, active Low). These signals are used as clock inputs for any of the functional blocks within the device. They may also be used as outputs for various receiver signals or internal clock signals.

**/TxREQA, /TxREQB.** *Transmit Request* (inputs or outputs, active Low). The primary function of these signals is to request DMA transfers to the transmit FIFOs. They may also be used as simple inputs or outputs.

**/RxREQA, /RxREQB.** *Receive Request* (inputs or outputs, active Low). The primary function of these signals is to request DMA transfers from the receive FIFOs. They may also be used as simple inputs or outputs.

**/CTS<sub>A</sub>, /CTS<sub>B</sub>.** *Clear To Send* (inputs or outputs, active Low). These signals are used as enables for the respective transmitters. They may also be programmed to generate interrupts on either transition or used as simple inputs or outputs.

**/DCDA, /DCDB.** *Data Carrier Detect* (inputs or outputs, active Low). These signals are used as enables for the respective receivers. They may also be programmed to generate interrupts on either transition or used as simple inputs or outputs.

---

## ARCHITECTURE

The USC internal structure includes two completely independent full-duplex serial channels, each with two baud rate generators, a digital phase-locked loop for clock recovery, transmit and receive character counters and a full-duplex DMA interface. The two serial channels share a common bus interface. The bus interface is designed to provide easy interface to most microprocessors, whether

they employ a multiplexed or non-multiplexed, 8-bit or 16-bit bus structure. Each channel is controlled by a set of thirty 16-bit registers, nearly all of which are readable and writable. There is one additional 16-bit register in the bus interface used to configure the nature of the bus interface. The BCR functions are shown in Figure 4.

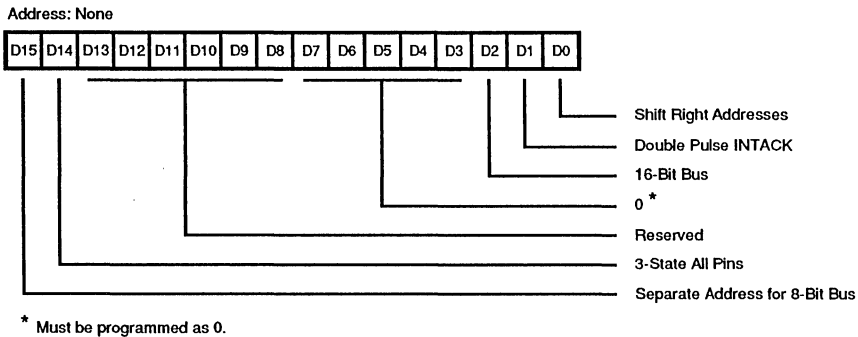


Figure 4. Bus Configuration Register

## DATA PATH

Both the transmitter and the receiver in the channel are actually microcoded serial processors. As the data shifts through the transmit or receive shift register, the microcode watches for specific bit patterns, counts bits, and at

the appropriate time transfers data to or from the FIFOs. The microcode also checks status and generates status interrupts as appropriate.

## FUNCTIONAL DESCRIPTION

The functional capabilities of the USC are described from two different points of view: as a data communications device, it transmits and receives data in a wide variety of data communications protocols; as a microprocessor peripheral, the USC offers such features as read/write registers, a flexible bus interface, DMA interface support and vectored interrupts.

### Data Communications Capabilities

The USC provides two independent full-duplex channels programmable for use in any common data communication protocol. The receiver and transmitter modes are completely independent, as are the two channels. Each receiver and transmitter is supported by a 32-byte deep FIFO and a 16-bit message length counter. All modes allow optional even, odd, mark or space parity. Synchronous modes allow the choice of two 16-bit or one 32-bit CRC polynomial. Selection of from one to eight bits-per-character is available in both receiver and transmitter, independently. Error and status conditions are carried with the data in the receive and transmit FIFOs to greatly reduce the CPU overhead required to send or receive a message. Specific, appropriately timed interrupts are available to signal such conditions as overrun, parity error, framing error, end-of-frame, idle line received, sync acquired,

transmit underrun, CRC sent, closing sync/flag sent, abort sent, idle line sent and preamble sent. In addition, several useful internal signals such as receive FIFO load, received sync, transmit FIFO read and transmission complete may be sent to pins for use by external circuitry.

**Asynchronous Mode.** The receiver and transmitter can handle data at a rate of 1/16, 1/32, or 1/64 the clock rate. The receiver rejects start bits less than one-half a bit time and will not erroneously assemble characters following a framing error. The transmitter is capable of sending one, two, or anywhere in the range of 1/16th to two stop bits per character in 1/16 bit increments.

**External Sync Mode.** The receiver is synchronized to the receive data stream by an externally-supplied signal on a pin for custom protocol applications.

**Isochronous Mode.** Both transmitter and receiver may operate on start-stop (async) data using a 1x clock. The transmitter can send one or two stop bits.

**Asynchronous With Code Violations.** This is similar to Isochronous mode except that the start bit is replaced by a three bit-time code violation pattern as in MIL-STD 1553B. The transmitter can send zero, one or two stop bits.



---

## FUNCTIONAL DESCRIPTION (Continued)

**Monosync Mode.** In this mode, a single character is used for synchronization. The sync character can be either eight bits long with an arbitrary data character length, or programmed to match the data character length. The receiver is capable of automatically stripping sync characters from the received data stream. The transmitter may be programmed to automatically send CRC on either an underrun or at the end of a programmed message length.

**Bisync Mode.** This mode is identical to monosync mode except that character synchronization requires two successive characters for synchronization. The two characters need not be identical.

**HDLC Mode.** In this mode, the receiver recognizes flags, performs optional address matching, accommodates extended address fields, 8- or 16-bit control fields and logical control fields, performs zero deletion and CRC checking. The receiver is capable of receiving shared-zero flags, recognizes the abort sequence and can receive arbitrary length messages. The transmitter automatically sends opening and closing flags, performs zero insertion and can be programmed to send an abort, an extended abort, a flag or CRC and a flag on transmit underrun. The transmitter can also automatically send the closing flag with optional CRC at the end of a programmed message length. Shared-zero flags are selected in the transmitter and a separate character length may be programmed for the last character in the frame.

**Bisync Transparent Mode.** In this mode, the synchronization pattern is DLE-SYN, programmable selected from either ASCII or EBCDIC encoding. The receiver recognizes control character sequences and automatically handles CRC calculation without CPU intervention. The transmitter can be programmed to send either SYN, DLE-SYN, CRC-SYN, or CRC-DLE-SYN upon underrun and can automatically send the closing DLE-SYN with optional CRC at the end of a programmed message length.

**NBIP Mode.** This mode is identical to async except that the receiver checks for the status of an additional address/data bit between the parity bit and the stop bit. The value of this bit is FIFO'ed along with the data. This bit is automatically inserted in the transmitter with the value that is FIFO'ed with the transmit data.

**802.3 Mode.** This mode implements the data format of IEEE 802.3 with 16-bit address compare. In this mode, /DCD and /CTS are used to implement the carrier sense and collision detect interactions with the receiver and transmitter.

**Slaved Monosync Mode.** This mode is available only in the transmitter and allows the transmitter (operating as though it were in monosync mode) to send data that is byte-synchronous to the data being received by the receiver.

**HDLC Loop Mode.** This mode is also available only in the transmitter and allows the USC to be used in an HDLC loop configuration. In this mode, the receiver is programmed to operate in HDLC mode so that the transmitter echos received messages. Upon receipt of a particular bit pattern (actually a sequence of seven consecutive ones) the transmitter breaks the loop and inserts its own frame(s).

### Data Encoding

The USC may be programmed to encode and decode the serial data in any of eight different ways as shown in Figure 5. The transmitter encoding method is selected independently of the receiver decoding method.

**NRZ.** In NRZ, a 1 is represented by a High level for the duration of the bit cell and a 0 is represented by a Low level for the duration of the bit cell.

**NRZB.** Data is inverted from NRZ.

**NRZI-Mark.** In NRZI-Mark, a 1 is represented by a transition at the beginning of the bit cell. That is, the level present in the preceding bit cell is reversed. A 0 is represented by the absence of a transition at the beginning of the bit cell.

**NRZI-Space.** In NRZI-Space, a 1 is represented by the absence of a transition at the beginning of the bit cell. That is, the level present in the preceding bit cell is maintained. A 0 is represented by a transition at the beginning of the bit cell.

**Biphase-Mark.** In Biphase-Mark, a 1 is represented by a transition at the beginning of the bit cell and another transition at the center of the bit cell. A 0 is represented by a transition at the beginning of the bit cell only.

**Biphase-Space.** In Biphase-Space, a 1 is represented by a transition at the beginning of the bit cell only. A 0 is represented by a transition at the beginning of the bit cell and another transition at the center of the bit cell.

**Biphase-Level.** In Biphase-Level, a 1 is represented by a High during the first half of the bit cell and a Low during the second half of the bit cell. A 0 is represented by a Low during the first half of the bit cell and a High during the second half of the bit cell.

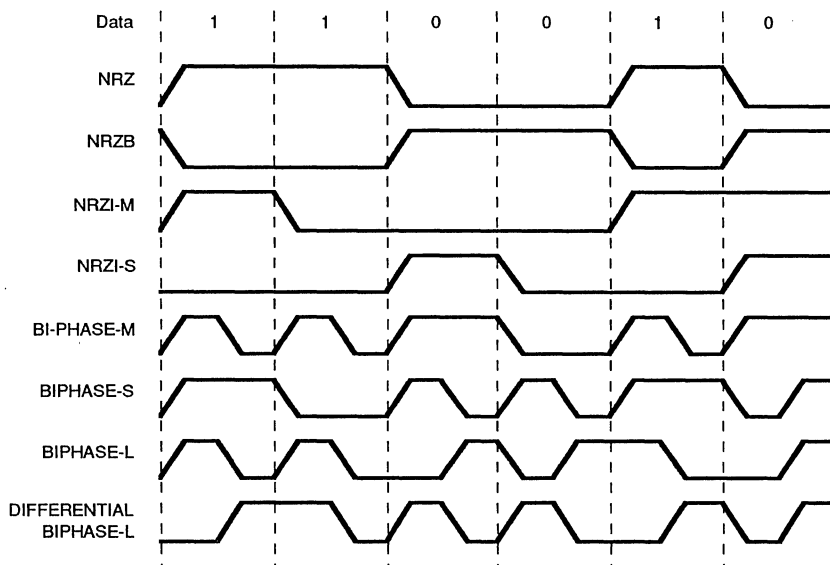


Figure 5. Data Encoding

**Differential Biphase-Level.** In Differential Biphase-Level, a 1 is represented by a transition at the center of the bit cell, with the opposite polarity from the transition at the center of the preceding bit cell. A 0 is represented by a transition at the center of the bit cell with the same polarity as the transition at the center of the preceding bit cell. In both cases there may be transitions at the beginning of the bit cell to set up the level required to make the correct center transition.

### Character Counters

Each channel in the USC contains a 16-bit character counter for both receiver and transmitter. The receive character counter may be preset either under software control or automatically at the beginning of a receive message. The counter decrements with each receive character and at the end of the receive message the current value in the counter is automatically loaded into a four-deep FIFO. This allows DMA transfer of data to proceed without CPU intervention at the end of a received message, as the values in the FIFO allow the CPU to determine message boundaries in memory. Similarly, the transmit character counter is loaded either under software control or automatically at the beginning of a transmit message. The counter is decremented with each write to the transmit FIFO. When the counter has decremented to

zero, and that byte is sent, the transmitter automatically terminates the message in the appropriate fashion (usually CRC and the closing flag or sync character) without requiring CPU intervention.

### Baud Rate Generators

Each channel in the USC contains two baud rate generators. Each generator consists of a 16-bit time constant register and a 16-bit down counter. In operation, the counter decrements with each baud rate generator clock, with the time constant automatically reloaded when the count reaches zero. The output of the baud rate generator toggles when the counter reaches a count of one-half of the time constant and again when the counter reaches zero. A new time constant may be written at any time but the new value will not take effect until the next load of the counter. The outputs of both baud rate generators are sent to the clock multiplexer for use internally or externally. The baud rate generator output frequency is related to the baud rate generator input clock frequency by the following formula:

$$\text{Output frequency} = \frac{\text{Input frequency}}{(\text{time constant} + 1)}$$

This allows an output frequency in the range of 1 to 1/65536 of the input frequency, inclusive.

---

## Digital Phase-Locked Loop

Each channel in the USC contains a Digital Phase-Locked Loop (DPLL) to recover clock information from a data stream with NRZI or Biphasic encoding. The DPLL is driven by a clock that is nominally 8, 16 or 32 times the receive data rate. The DPLL uses this clock, along the data stream, to construct a clock for the data. This clock may then be routed to the receiver, transmitter, or both, or to a pin for use externally. In all modes, the DPLL counts the input clock to create nominal bit times. As the clock is counted, the DPLL watches the incoming data stream for transitions. Whenever a transition is detected, the DPLL makes a count adjustment (during the next counting cycle), to produce an output clock which tracks the incoming bit cells. The DPLL provides properly phased transmit and receive clocks to the clock multiplexer.

## Counters

Each channel contains two 5-bit counters, which are programmed to divide an input clock by 4, 8, 16 or 32. The

inputs of these two counters are sent to the clock multiplexer. The counters are used as prescalers for the baud rate generators, or to provide a stable transmit clock from a common source when the DPLL is providing the receive clock.

## Clock Multiplexer

The clock multiplexer in each channel selects the clock source for the various blocks in the channel and selects an internal clock signal to potentially be sent to either the /RxC or /TxC pin.

## Test Modes

The USC is programmed for local loopback or auto echo operation. In local loopback, the output of the transmitter is internally routed to the input of the receiver. This allows testing of the USC data paths without any external logic. Auto echo connects the RxD pin directly to the TxD pin. This is useful for testing serial links external to the USC.

---

## I/O INTERFACE CAPABILITIES

The USC offers the choice of polling, interrupt (vectored or non-vectored) and block transfer modes to transfer data, status and control information to and from the CPU.

### Polling

All interrupts are disabled. The registers in the USC are automatically updated to reflect current status. The CPU polls the Daisy Chain Control Register (DCCR) to determine status changes and then reads the appropriate status register to find and respond to the change in status. USC status bits are grouped according to function to simplify this software action.

### Interrupt

When a USC responds to an interrupt acknowledge from the CPU, an interrupt vector may be placed on the data bus. This vector is held in the Interrupt Vector Register (IVR). To speed interrupt response time, the USC modifies three bits in this vector to indicate which type of interrupt is being requested.

Each of the six sources of interrupts in each channel of the USC (Receive Status, Receive Data, Transmit Status, Transmit Data, I/O Status and Device Status) has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt-Under-Service (IUS) and Interrupt Enable (IE). If the IE bit for a given source is set, that source can request interrupts. Note that individual sources within the

six groups also have interrupt enable bits which are set for the particular source. In addition, there is a Master Interrupt Enable (MIE) bit in each channel which globally enables or disables interrupts within the channel.

The other two bits are related to the interrupt priority chain. A channel in the USC may request an interrupt only when no higher priority interrupt source is requesting one, e.g., when IEI is High for the channel. In this case the channel activates the /INT signal. The CPU then responds with an interrupt acknowledge cycle, and the interrupting channel places a vector on the data bus.

In the USC, the IP bit signals that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority within the channel and external to the channel are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the channel being pulled Low and propagated to subsequent peripherals. An IUS bit is set during an interrupt acknowledge cycle if there are no higher priority devices requesting interrupts.

There are six sources of interrupt in each channel: Receive Status, Receive Data, Transmit Status, Transmit Data, I/O Status and Device Status, prioritized in that order within the channel. There are six sources of Receive Status interrupt, each individually enabled: exited hunt, idle line, break/abort, code violation/end-of-transmission/end-of-frame,

parity error and overrun error. The Receive Data interrupt is generated whenever the receive FIFO fills with data beyond the level programmed in the Receive Interrupt Control Register (RICR).

There are six sources of Transmit Status interrupt, each individually enabled: preamble sent, idle line sent, abort sent, end-of-frame/end-of-transmission sent, CRC sent and underrun error. The Transmit Data interrupt is generated whenever the transmit FIFO empties below the level programmed in the Transmit Interrupt Control Register (TICR). The I/O Status interrupt serves to report transitions on any of six pins. Interrupts are generated on either or both edges with separate selection and enables for each pin. The pins programmed to generate I/O Status interrupts are /RxC, /TxC, /RxREQ, /TxREQ, /DCD and /CTS. These interrupts are independent of the programmed function of the pins. The Device Status interrupt has four separately enabled sources: receive character count FIFO overflow, DPLL sync acquired, BRG1 zero count and BRGO zero count.

## Block Transfer Mode

The USC accommodates block transfers via DMA through the /RxREQ, /TxREQ, /RxACK and /TxACK pins. The /RxREQ signal is activated when the fill level of the receive FIFO exceeds the value programmed in the RICR. The DMA may respond with either a normal bus transaction or by activating the /RxACK pin to read the data directly (fly-by transfer). The /TxREQ signal is activated when the empty level of the transmit FIFO falls below the value programmed in the TICR. The DMA may respond either with a normal bus transaction or by activating the /TxACK pin to write the data directly (fly-by transfer). The /RxACK and /TxACK pin functions for this mode are controlled by the Hardware Configuration Register (HCR). Then using the /RxACK and /TxACK pins to transfer data, no chip select is necessary; these are dedicated strobes for the appropriate FIFO.

## PROGRAMMING

The Programmers Assistant (MSDOS based) and Technical Manual are available to provide details about programming the USC. Also included are explanations and features of all registers in the USC.

The registers in each USC channel are programmed by the system to configure the channels. Before this can occur, however, the system must program the bus interface by writing to the Bus Configuration Register (BCR). The BCR has no specific address and is only accessible immediately after a hardware reset of the device. The first write to the USC, after a hardware reset, programs the BCR. From that time on the normal channel registers may be accessed. No specific address need be presented to the USC for the BCR write; the USC knows that the first write after a hardware reset is destined for the BCR.

In the multiplexed bus case, all registers are directly addressable via the address latched by /AS at the beginning of a bus transaction. The address is decoded from either AD6-AD0 or AD7-AD1. This is controlled by the Shift Right/Shift Left bit in the BCR. The address maps for these two cases are shown in Table 1. The D//C pin is still used to directly access the receive and transmit data registers (RDR and TDR) in the multiplexed bus; if D//C is High the address latched by /AS is ignored and an access of RDR or TDR is performed.

In the non-multiplexed bus case, the registers in each channel are accessed indirectly using the address pointer in the Channel Command/Address Register (CCAR) in each channel. The address of the desired register is first

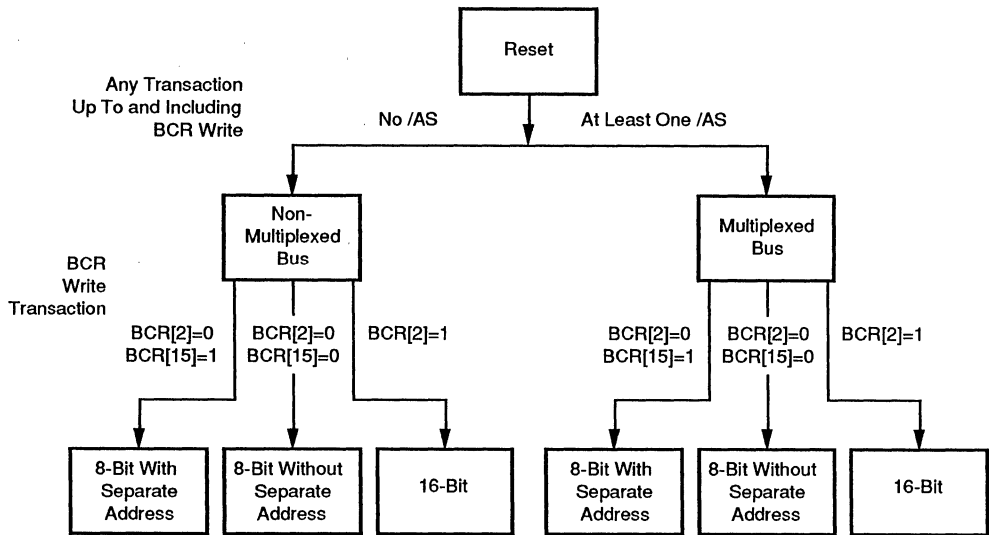
written to the CCAR and then the selected register is accessed; the pointer in the CCAR is automatically cleared after this access. The RDR and TDR are accessed directly using the D//C pin, without disturbing the contents of the pointer in the CCAR.

**Table 1. Multiplexed Bus Address Assignments**

Address Signal	Shift Left	Shift Right
Byte/Word Access	AD7	AD6
Address 4	AD6	AD5
Address 3	AD5	AD4
Address 2	AD4	AD3
Address 1	AD3	AD2
Address 0	AD2	AD1
Upper//Lower Byte Select	AD1	AD0

There are two important things to note about the USC. First, the Channel Reset bit in the CCAR places the channel in the reset state. To exit this reset state either a word of all zeros must be written to the CCAR (16-bit bus) or a byte of all zeros must be written to the lower byte of the CCAR (8-bit bus). The second thing to note is that after reset, the transmit and receive clocks are not connected. The first thing that should be done in any initialization sequence is a write to the Clock Mode Control Register (CMCR) to select a clock source for the receiver and transmitter.

The register addressing is shown in Table 2. and the bit assignments for the registers are shown in Figure 6.



Note:  
The presence of one transaction with an /AS active, between reset up to and including the BCR write, chooses a multiplexed type of bus.

Figure 6. BCR Reset Sequence and Bit Assignments

Table 2. Register Address List

Address A4-A0			Address A4-A0		
00000	CCAR	Channel Command/Address Register	10010	RCSR	Receive Command/Status Register
00001	CMR	Channel Mode Register	10011	RICR	Receive Interrupt Control Register
00010	CCSR	Channel Command/Status Register	10100	RSR	Receive Sync Register
00011	CCR	Channel Control Register	10101	RCLR	Receive Count Limit Register
00110	TMDR	Test Mode Data Register	10110	RCCR	Receive Character Count Register
00111	TMCR	Test Mode Control Register	10111	TCOR	Time Constant 0 Register
01000	CMCR	Clock Mode Control Register	1X000	TDR	Transmit Data Register (Write Only)
01001	HCR	Hardware Configuration Register	11001	TMR	Transmit Mode Register
01010	IVR	Interrupt Vector Register	11010	TCSR	Transmit Command/Status Register
01011	IOCR	I/O Control Register	11011	TICR	Transmit Interrupt Control Register
01100	ICR	Interrupt Control Register	11100	TSR	Transmit Sync Register
01101	DCCR	Daisy-Chain Control Register	11101	TCLR	Transmit Count Limit Register
01110	MISR	Misc Interrupt Status Register	11110	TCCR	Transmit Character Count Register
01111	SICR	Status Interrupt Control Register	11111	TC1R	Time Constant 1 Register
1X000	RDR	Receive Data Register (Read Only)	XXXXX	BCR	Bus Configuration Register
10001	RMR	Receive Mode Register			

# CONTROL REGISTERS

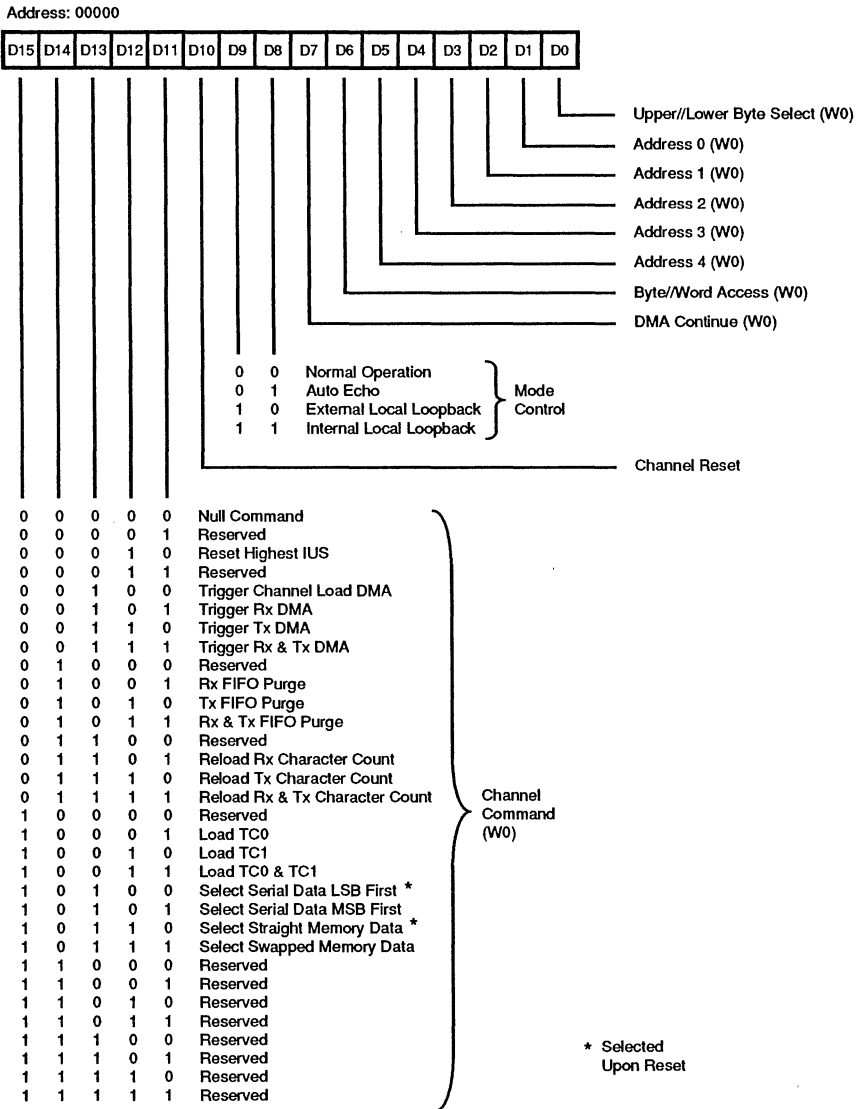


Figure 7. Channel Command/Address Register

Address: 00001

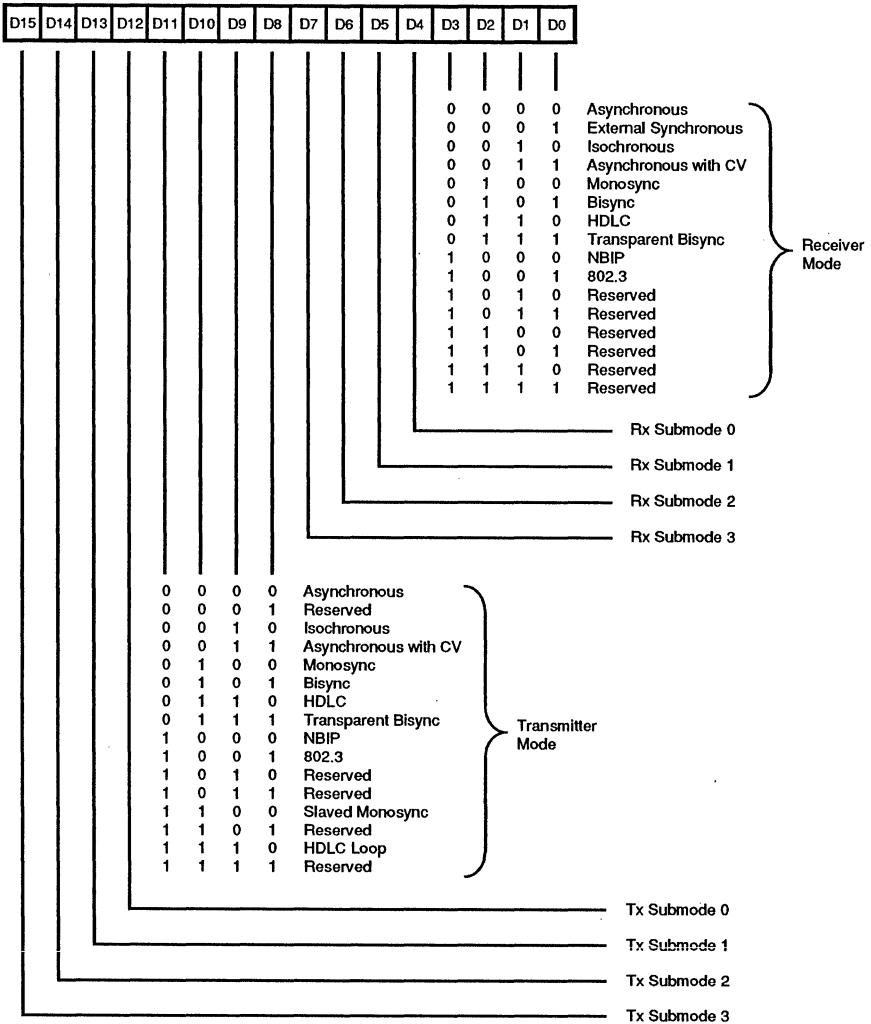


Figure 8. Channel Mode Register

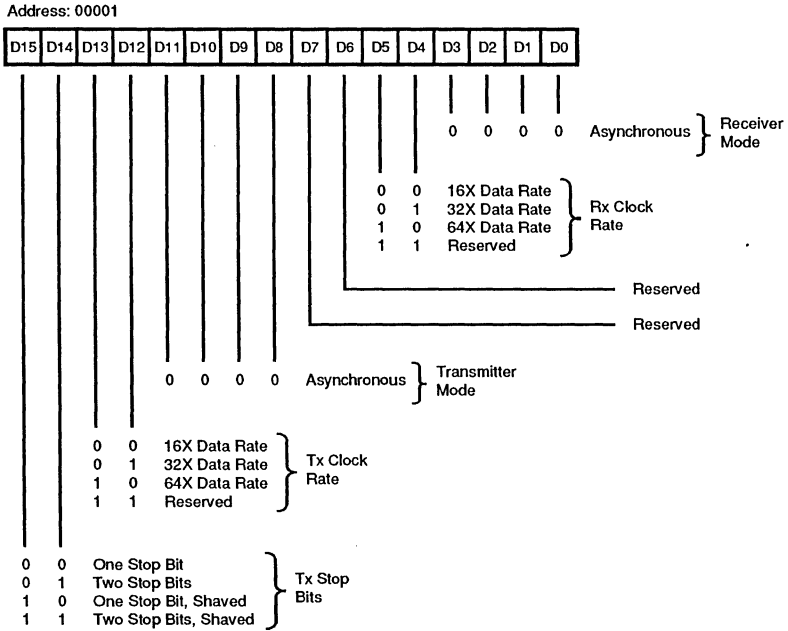


Figure 9. Channel Mode Register, Asynchronous Mode

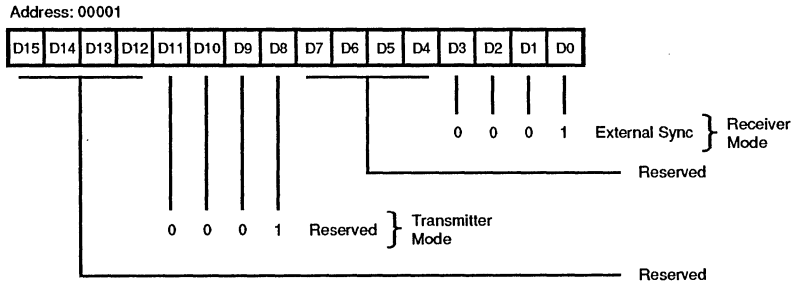


Figure 10. Channel Mode Register, External Sync Mode



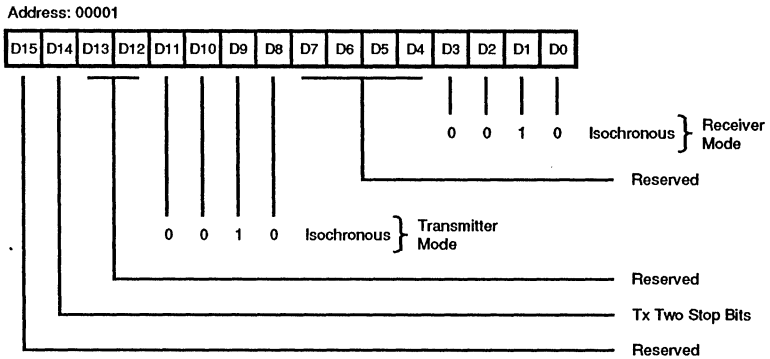


Figure 11. Channel Mode Register, Isochronous Mode

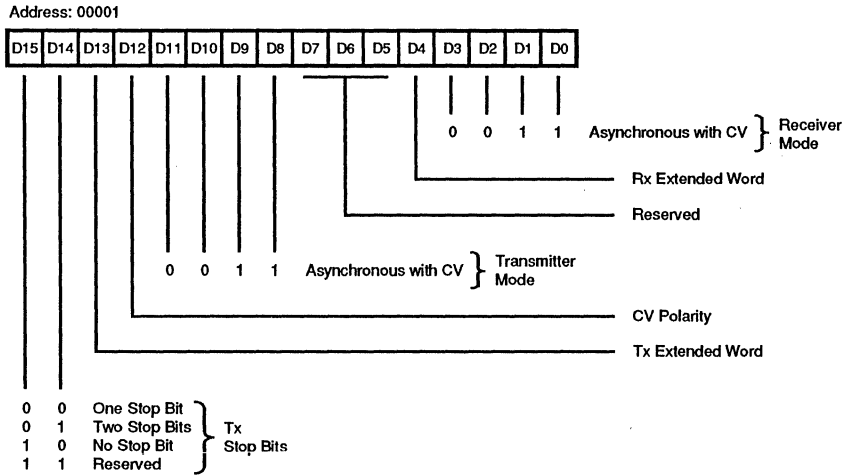


Figure 12. Channel Mode Register, Asynchronous Mode with Code Violation (MIL STD 1553)

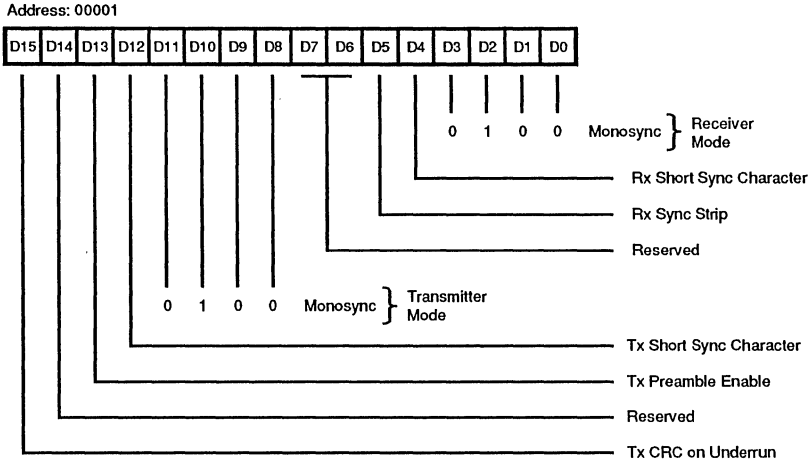


Figure 13. Channel Mode Register, Monosync Mode

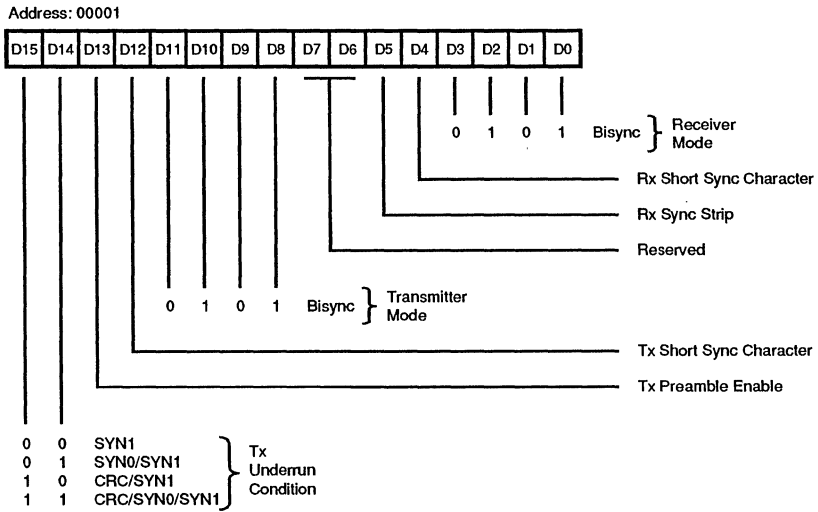


Figure 14. Channel Mode Register, Bisync Mode

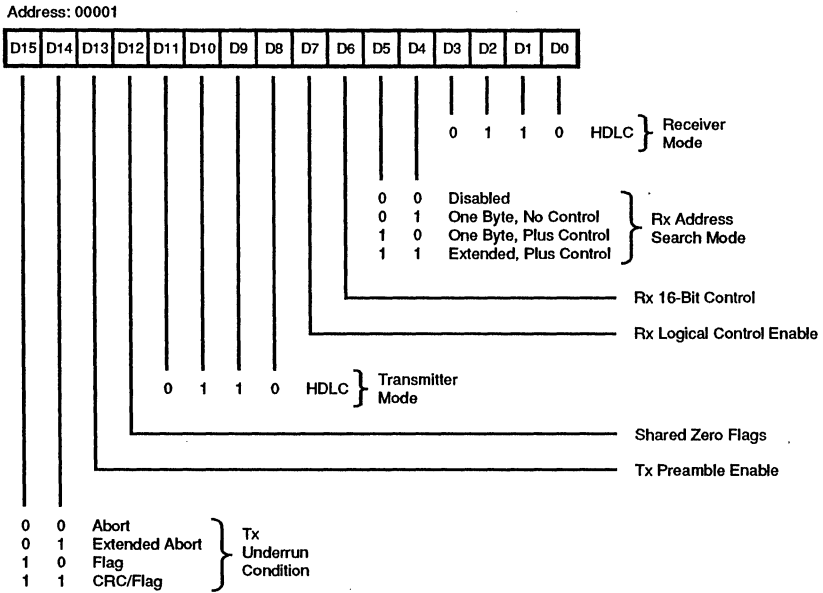


Figure 15. Channel Mode Register, HDLC Mode

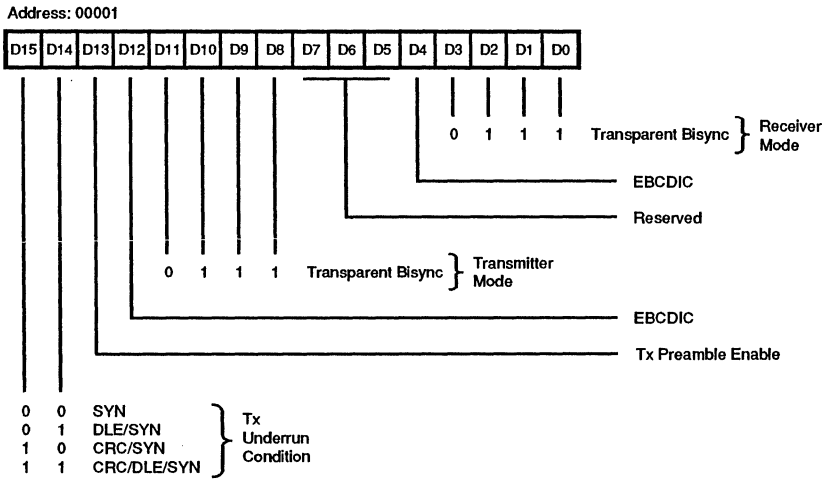


Figure 16. Channel Mode Register, Transparent Bisync Mode

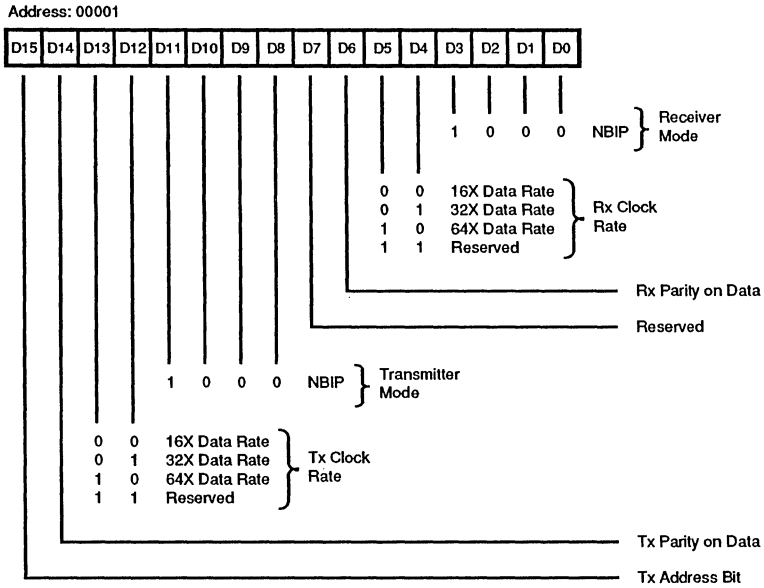


Figure 17. Channel Mode Register, NBIP Mode

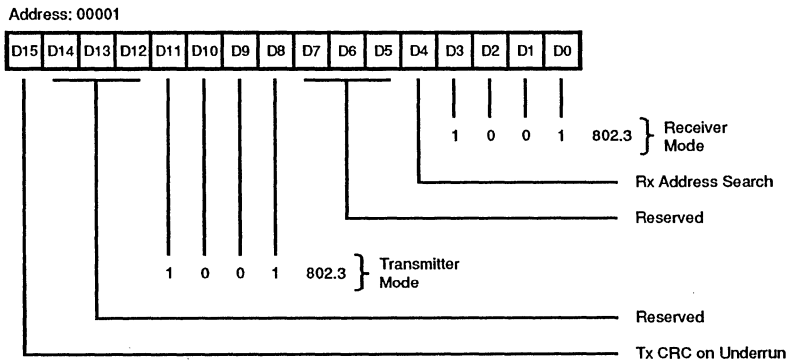


Figure 18. Channel Mode Register, 802.3 Mode

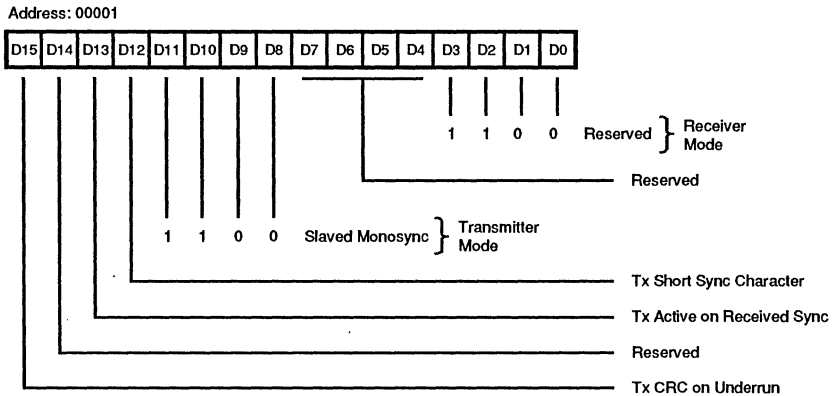


Figure 19. Channel Mode Register, Slaved Monosync Mode

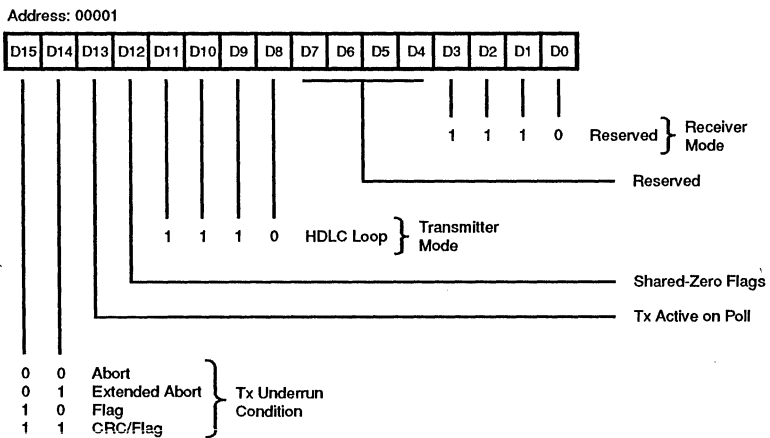


Figure 20. Channel Mode Register, HDLC Loop Mode

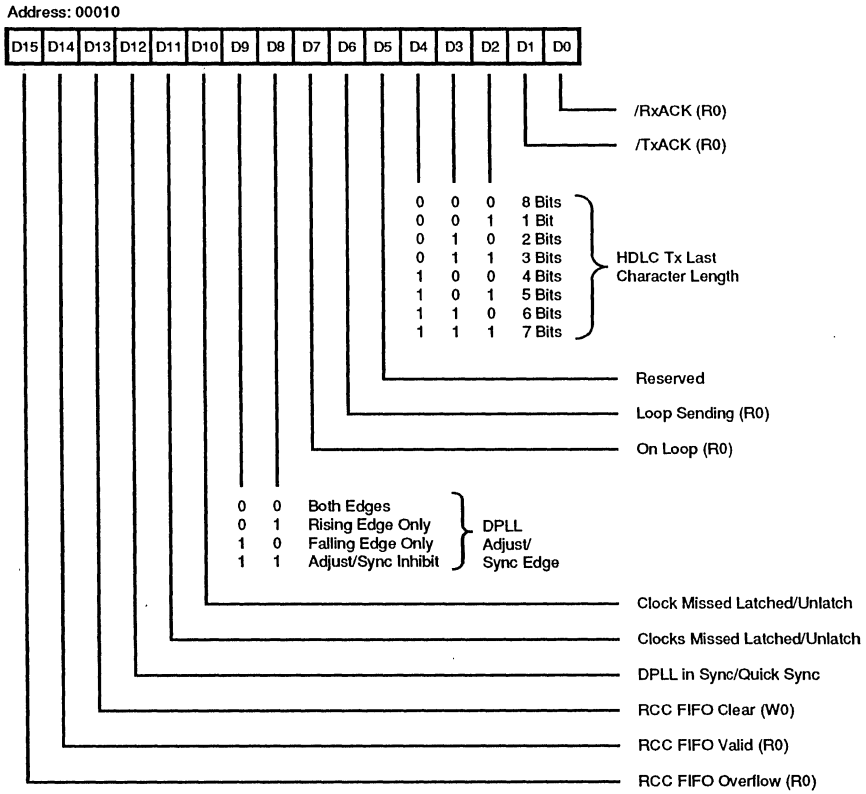


Figure 21. Channel Command/Status Register

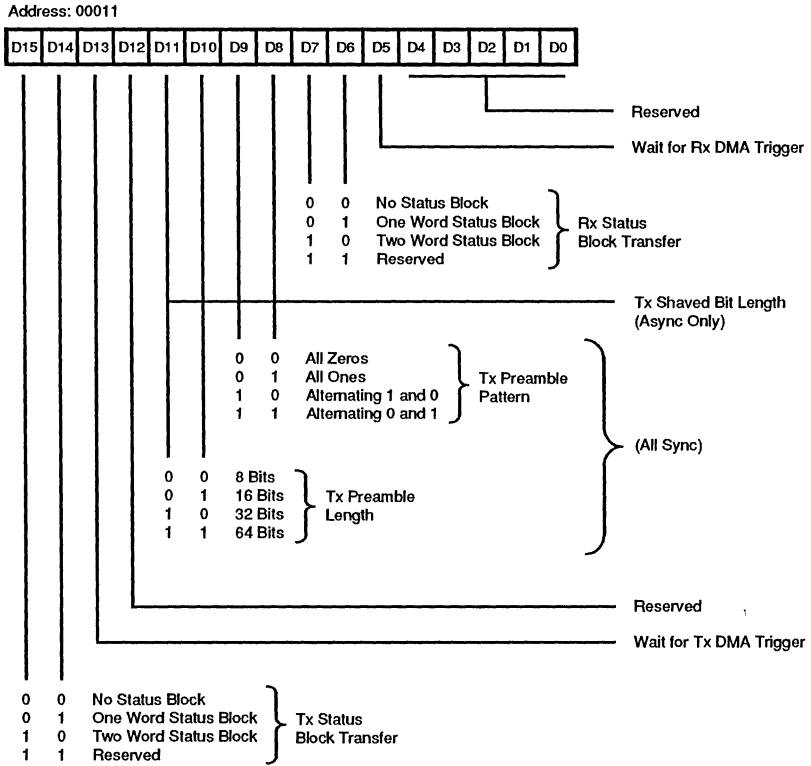


Figure 22. Channel Control Register

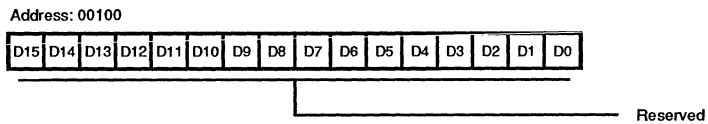


Figure 23. Primary Reserved Register

Address: 00101

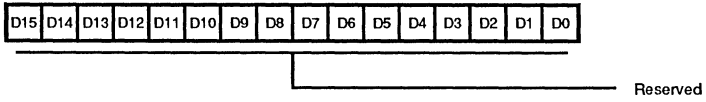


Figure 24. Secondary Reserved Register

Address: 00110

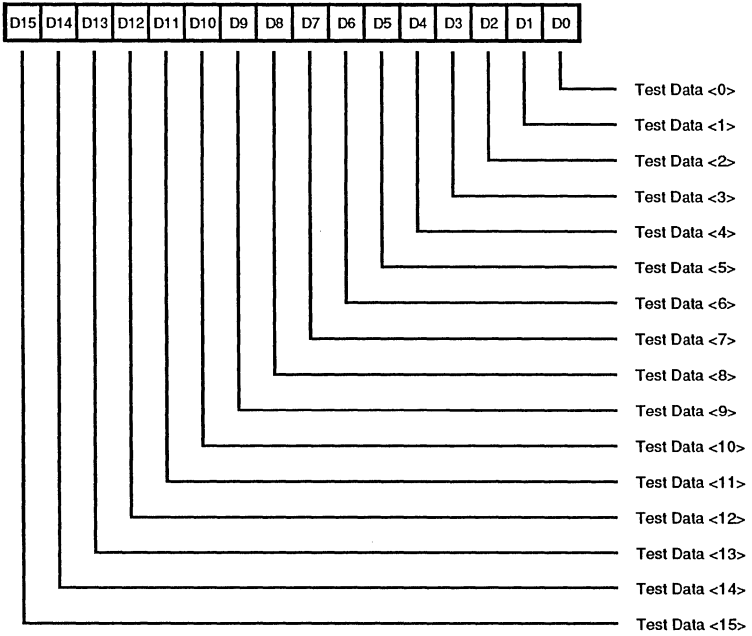


Figure 25. Test Mode Data Register



Address: 00111

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
											0	0	0	0	0	Null Address
											0	0	0	0	1	High Byte of Shifters
											0	0	0	1	0	CRC Byte 0
											0	0	0	1	1	CRC Byte 1
											0	0	1	0	0	Rx FIFO (Write)
											0	0	1	0	1	Clock Multiplexer Outputs
											0	0	1	1	0	CTR0 and CTR1 Counters
											0	0	1	1	1	Clock Multiplexer Inputs
											0	1	0	0	0	DPLL State
											0	1	0	0	1	Low Byte of Shifters
											0	1	0	1	0	CRC Byte 2
											0	1	0	1	1	CRC Byte 3
											0	1	1	0	0	Tx FIFO (Read)
											0	1	1	0	1	Reserved
											0	1	1	1	0	I/O and Device Status Latches
											0	1	1	1	1	Internal Daisy Chain
											1	0	0	0	0	Reserved
											1	0	0	0	1	Reserved
											1	0	0	1	0	Reserved
											1	0	0	1	1	Reserved
											1	0	1	0	0	Reserved
											1	0	1	0	1	Rx Count Holding Register
											1	0	1	1	1	Reserved
											1	1	0	0	0	Reserved
											1	1	0	0	1	Reserved
											1	1	0	1	0	Reserved
											1	1	0	1	1	Reserved
											1	1	1	0	0	Reserved
											1	1	1	0	1	Reserved
											1	1	1	1	0	Reserved
											1	1	1	1	1	Reserved

Figure 26. Test Mode Control Register

Address: 01000

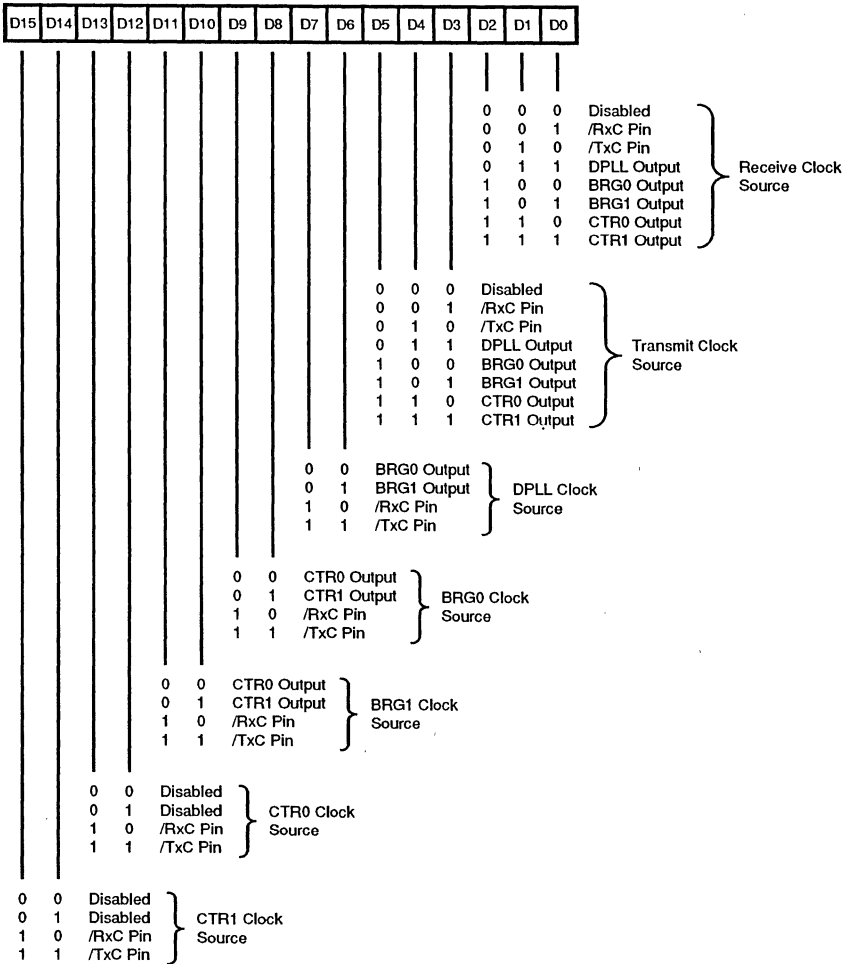


Figure 27. Clock Mode Control Register

Address: 01001

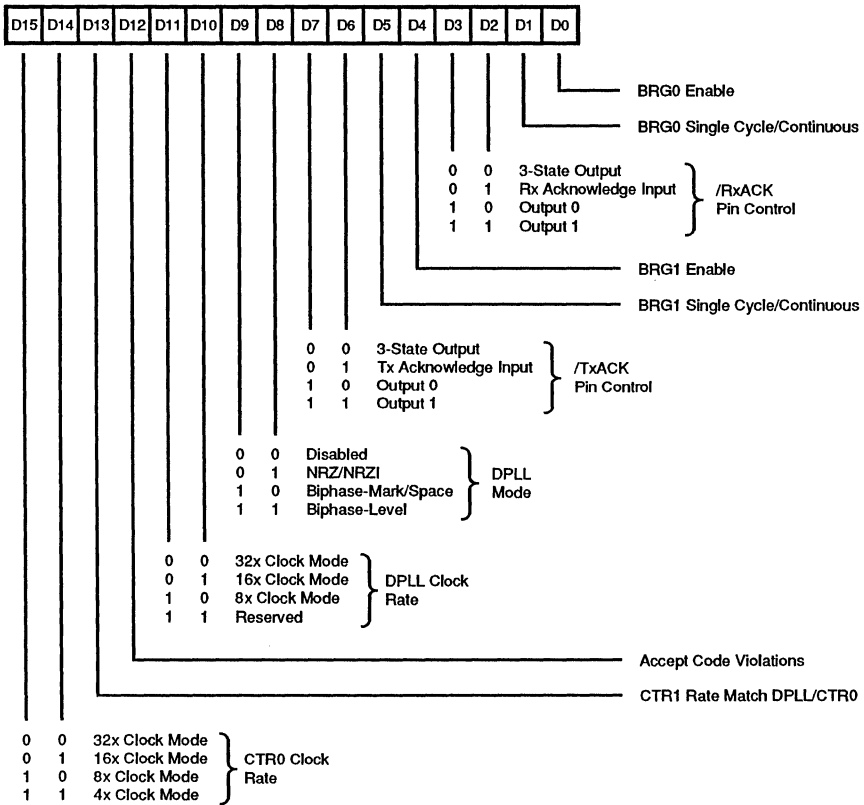


Figure 28. Hardware Configuration Register

Address: 01010

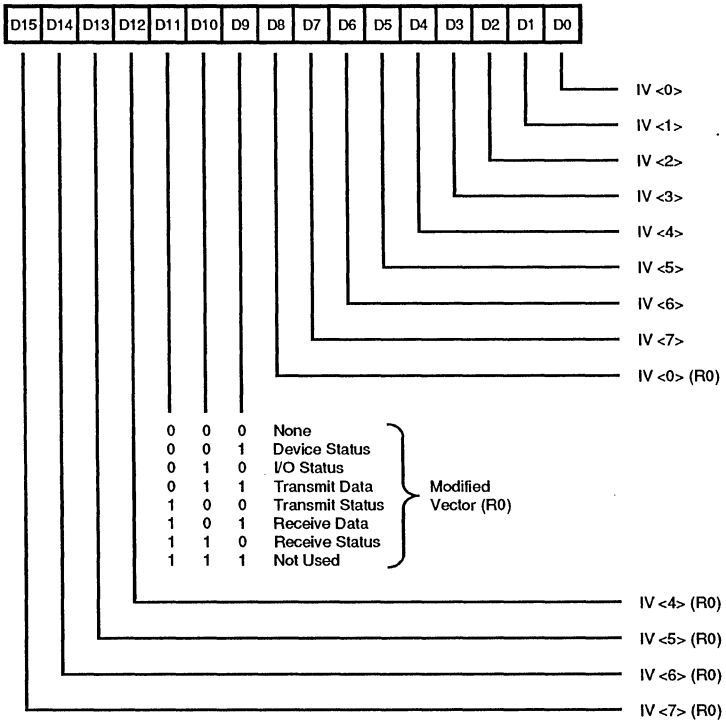


Figure 29. Interrupt Vector Register

Address: 01011

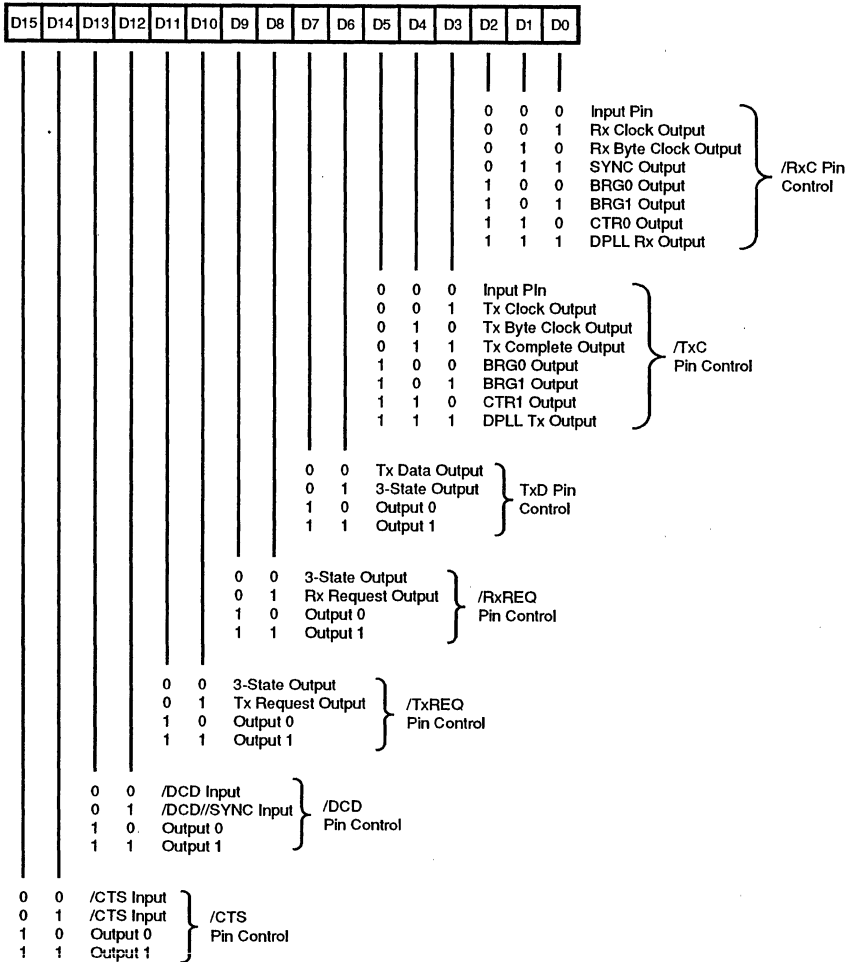


Figure 30. I/O Control Register

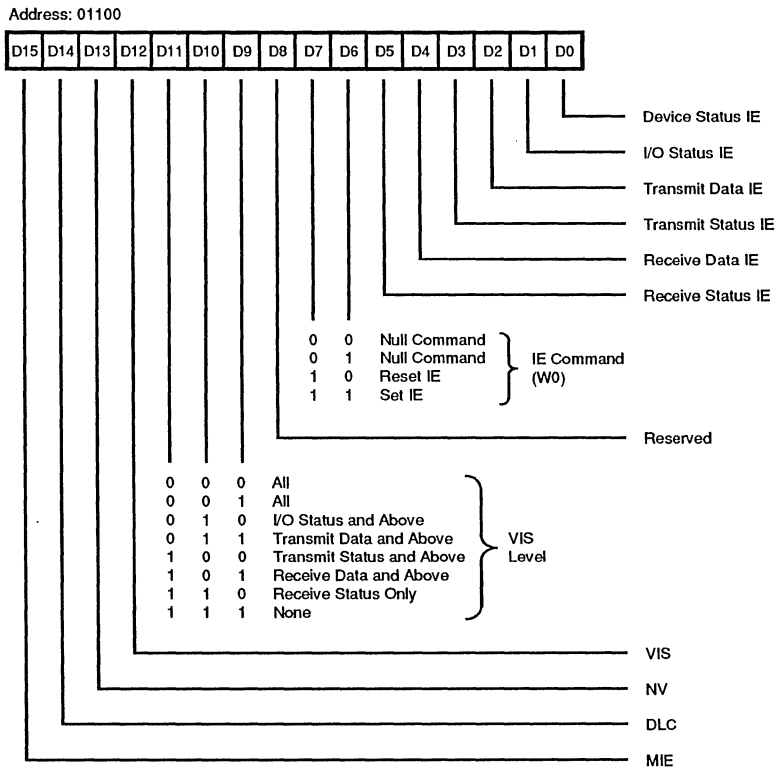


Figure 31. Interrupt Control Register

Address: 01101

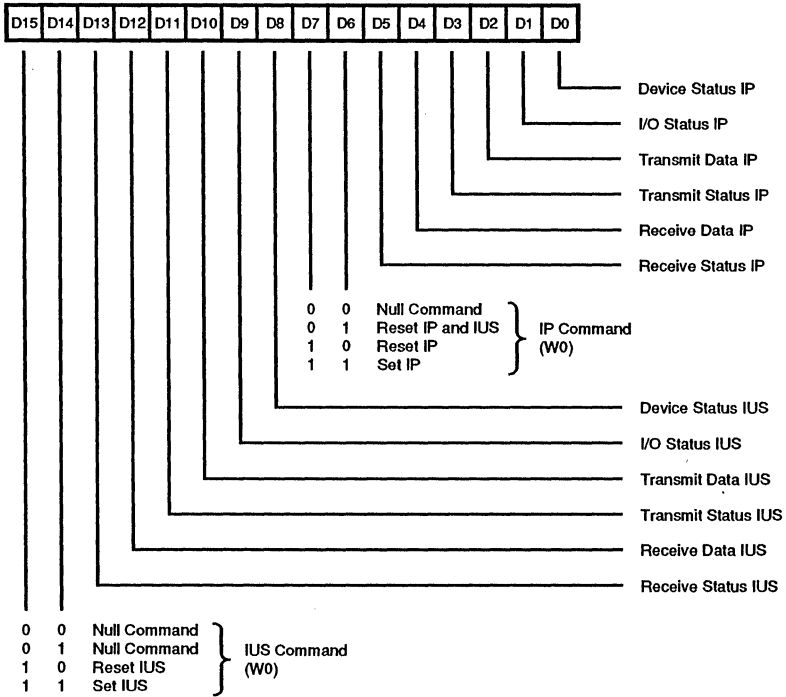


Figure 32. Daisy-Chain Control Register

Address: 01110

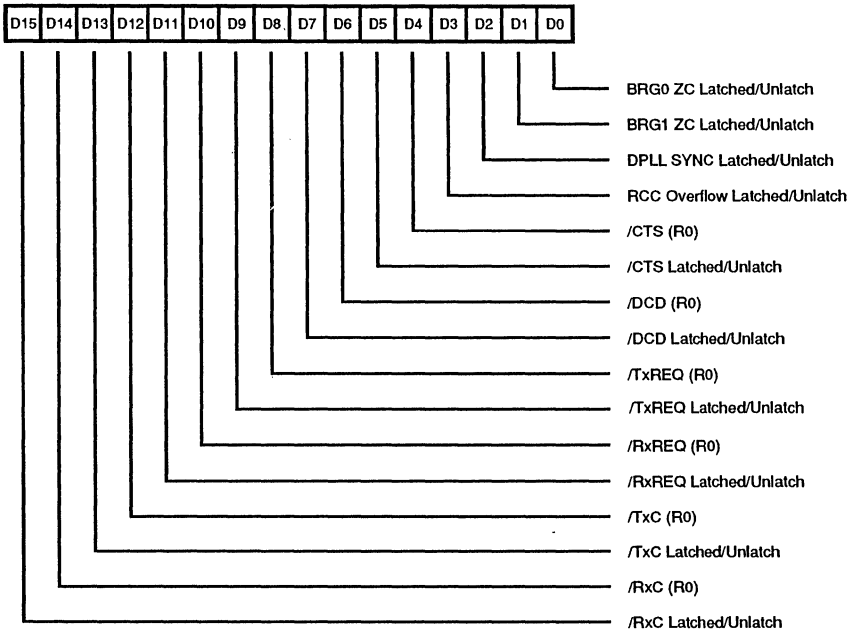


Figure 33. Miscellaneous Interrupt Status Register



Address: 01111

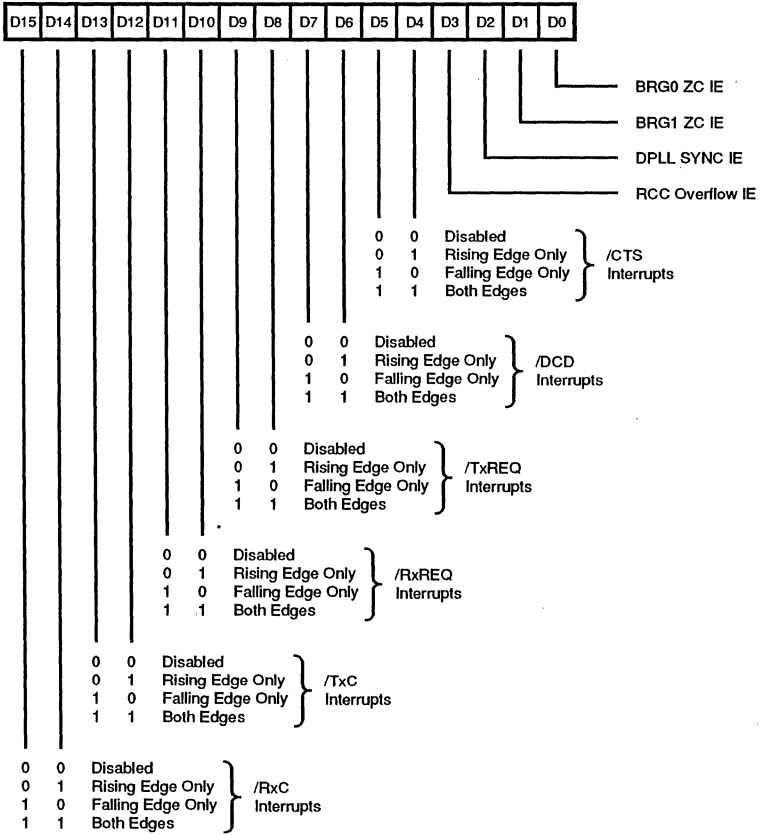


Figure 34. Status Interrupt Control Register

Address: 1x000

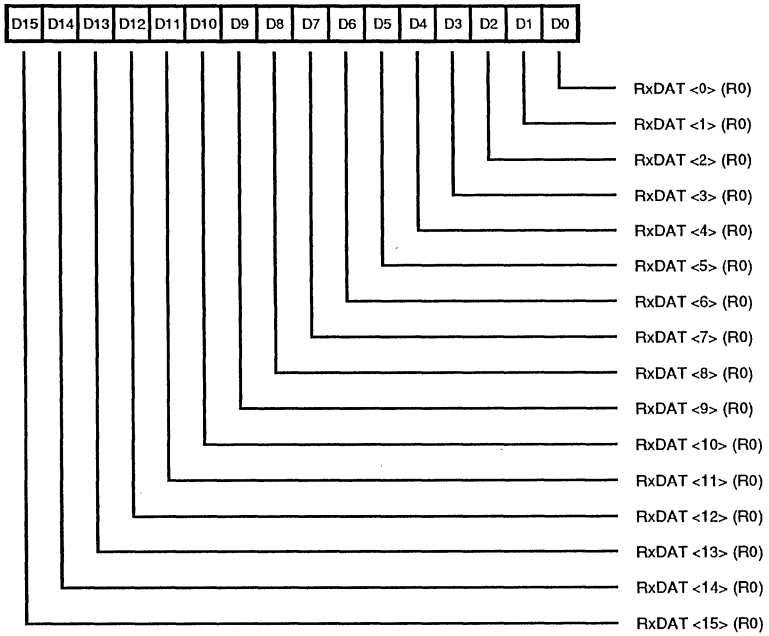


Figure 35. Receive Data Register

Address: 10001

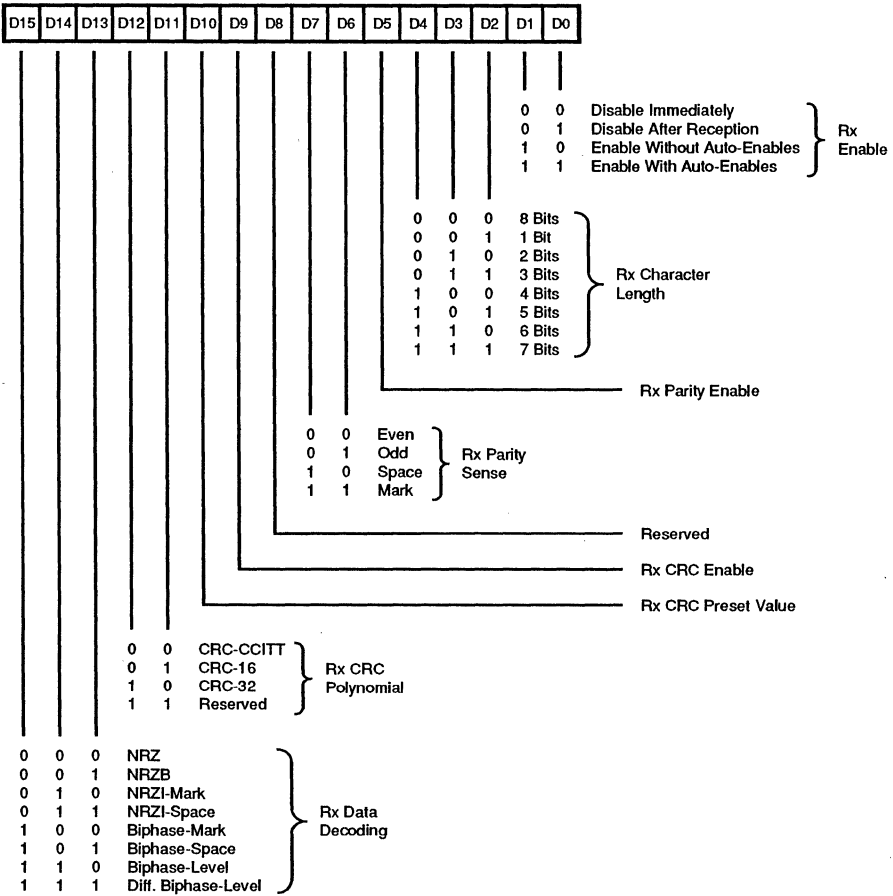


Figure 36. Receive Mode Register

Address: 10010

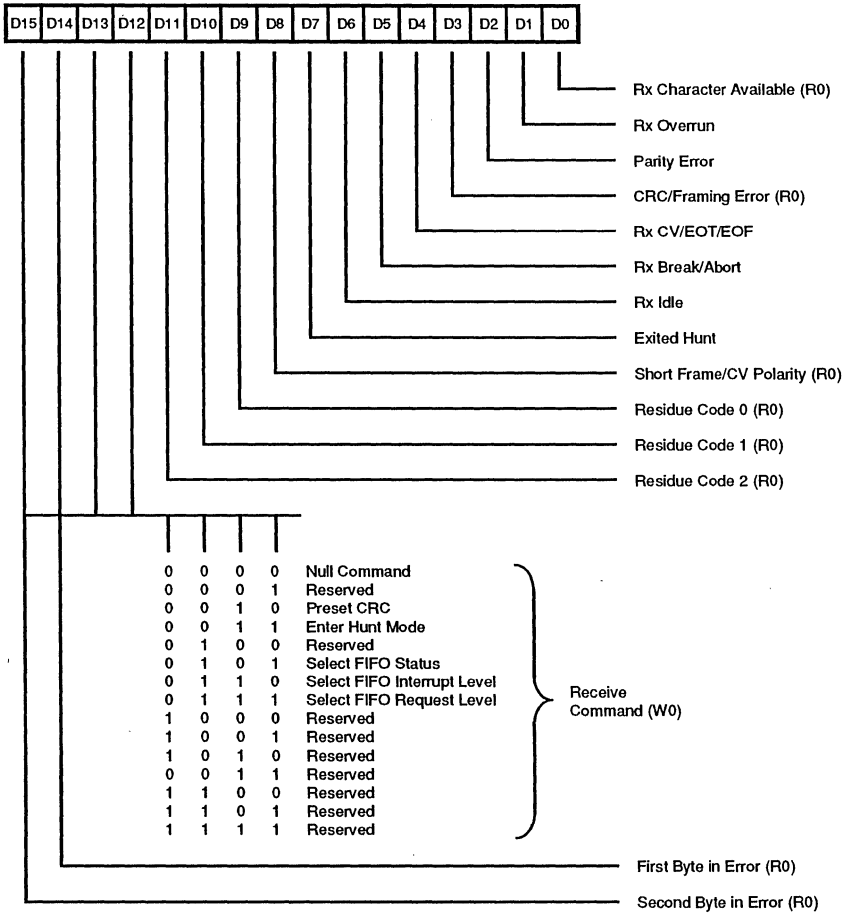


Figure 37. Receive Command Status Register

Address: 10011

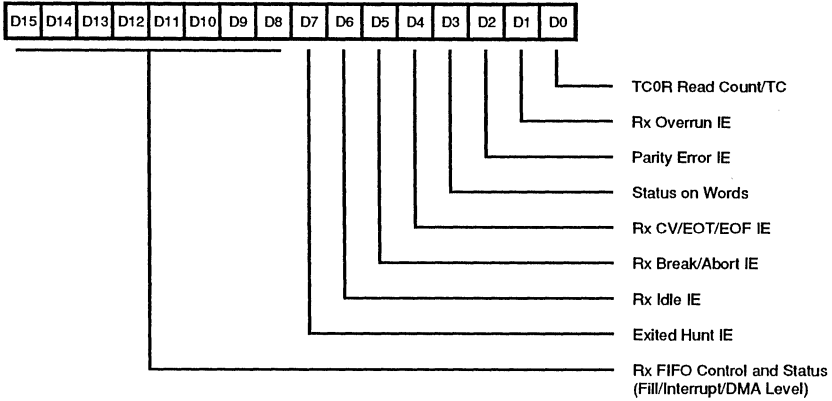


Figure 38. Receive Interrupt Control Register

Address: 10100

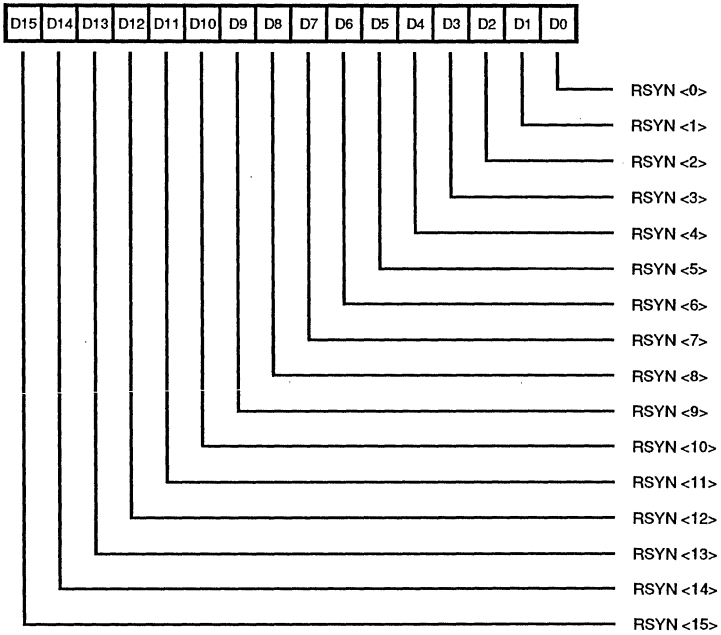


Figure 39. Receive Sync Register

Address: 10101

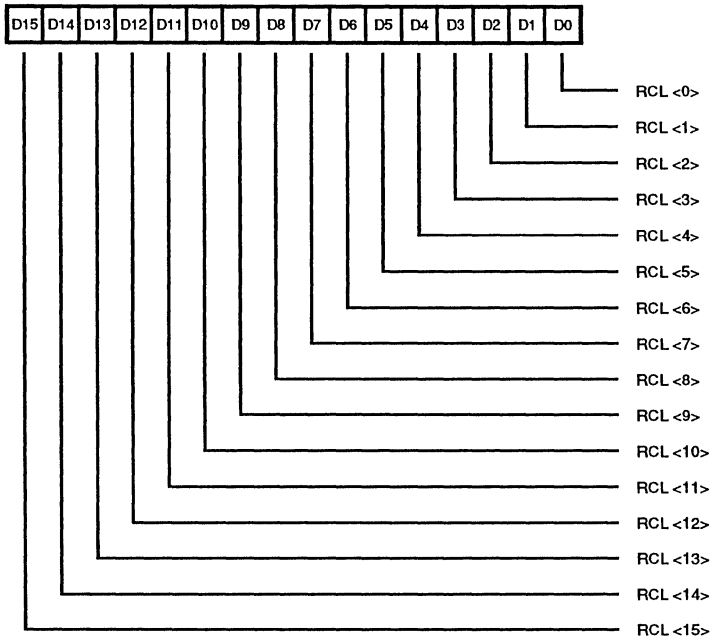


Figure 40. Receive Count Limit Register

Address: 10110

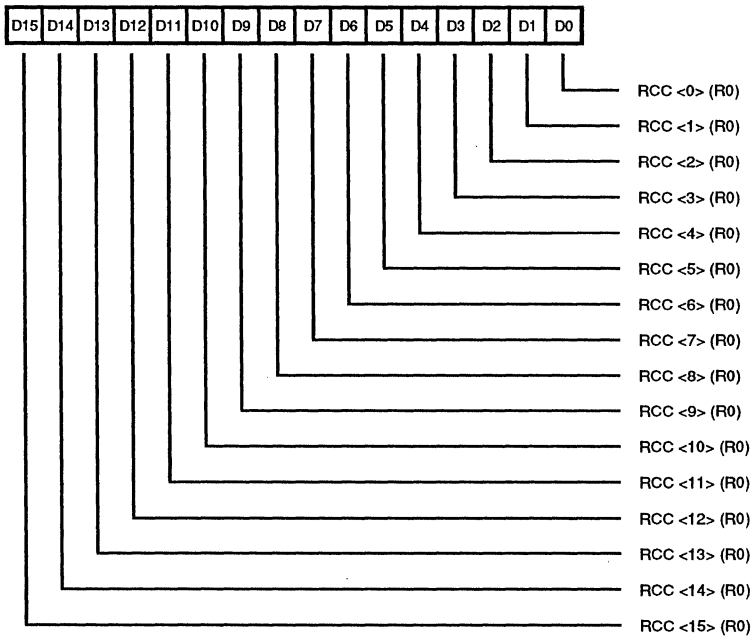


Figure 41. Receive Character Count Register

Address: 10111

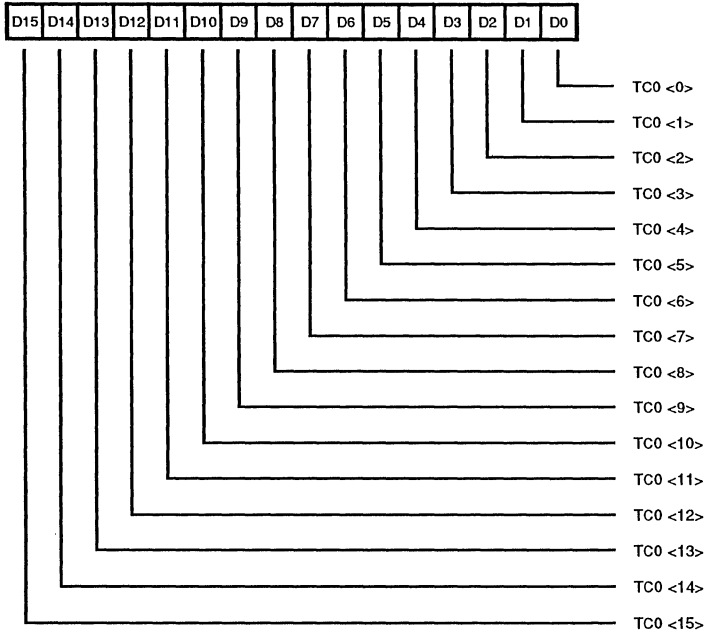


Figure 42. Time Constant 0 Register



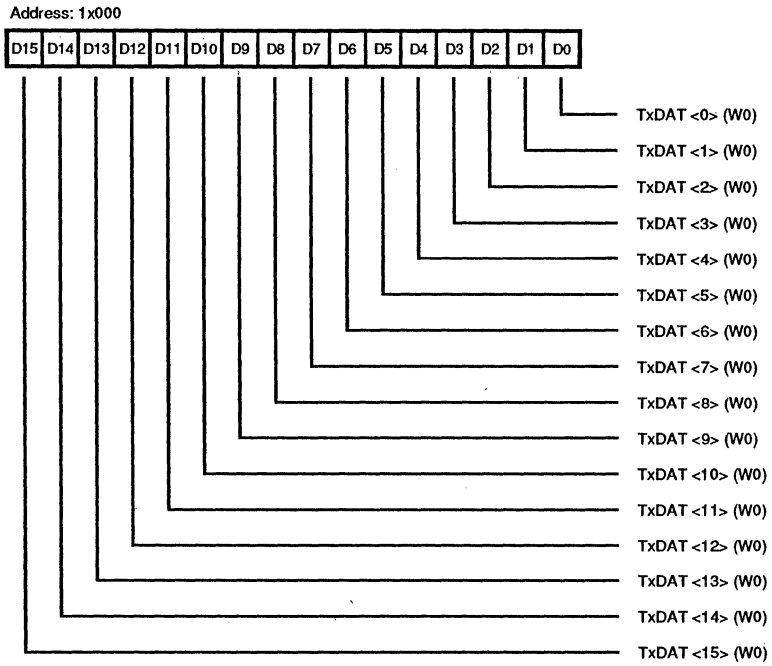


Figure 43. Transmit Data Register

Address: 11001

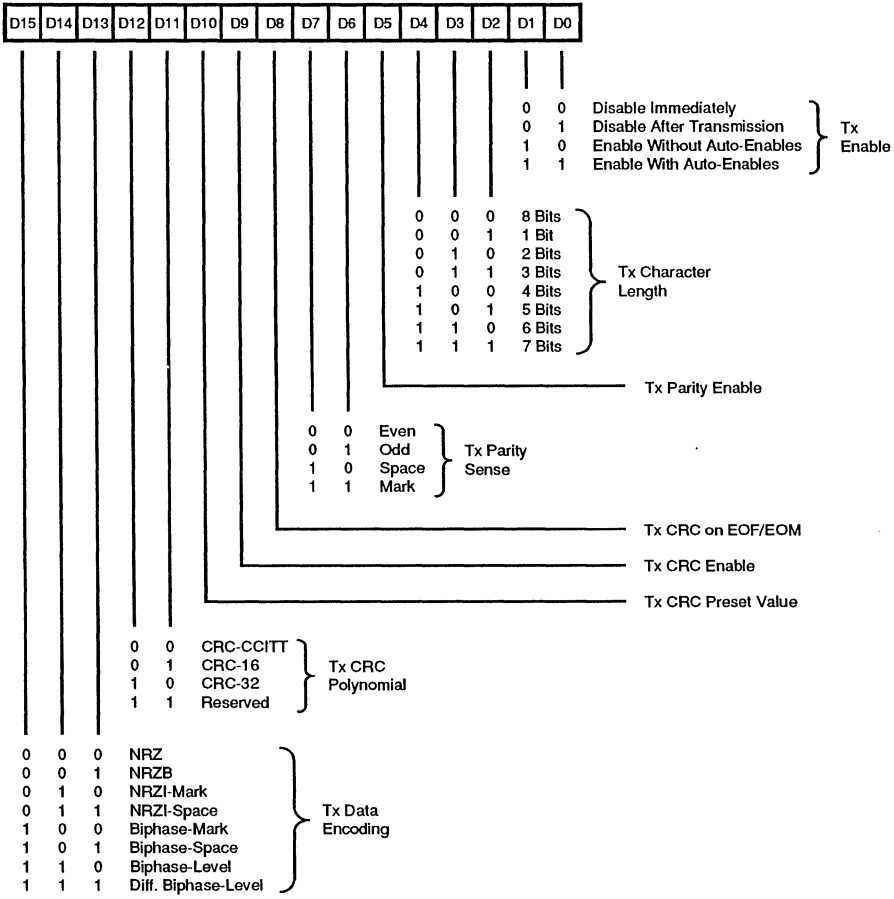


Figure 44. Transmit Mode Register



Address: 11011

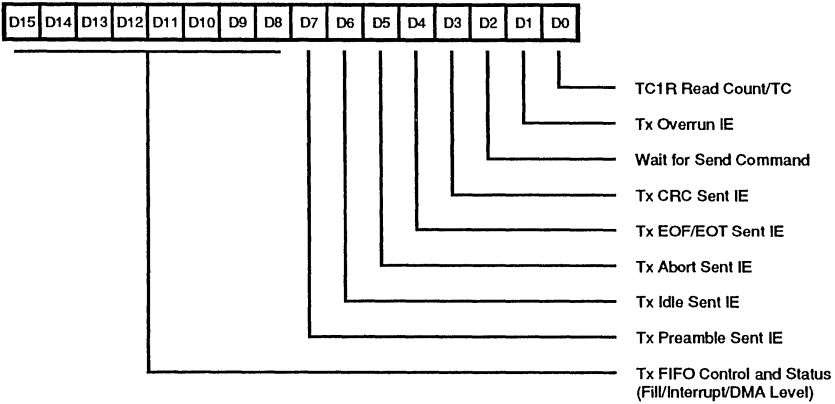


Figure 46. Transmit Interrupt Control Register

Address: 11100

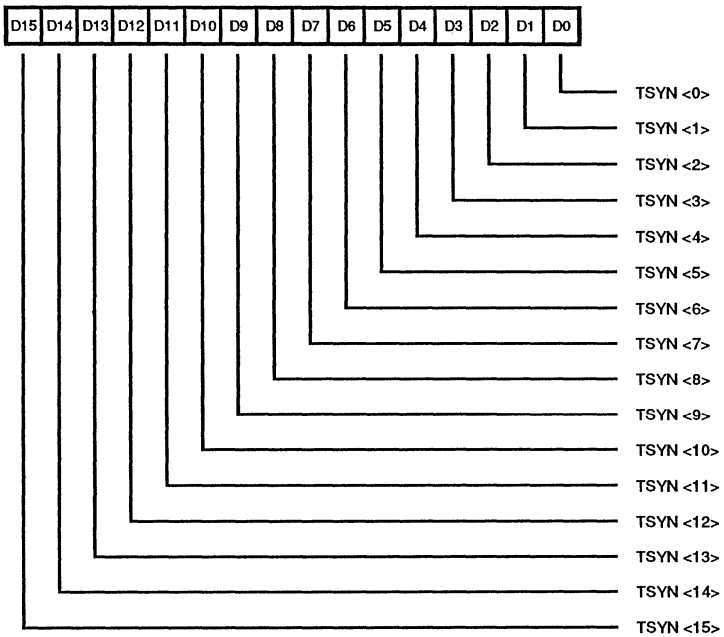


Figure 47. Transmit Sync Register

Address: 11101

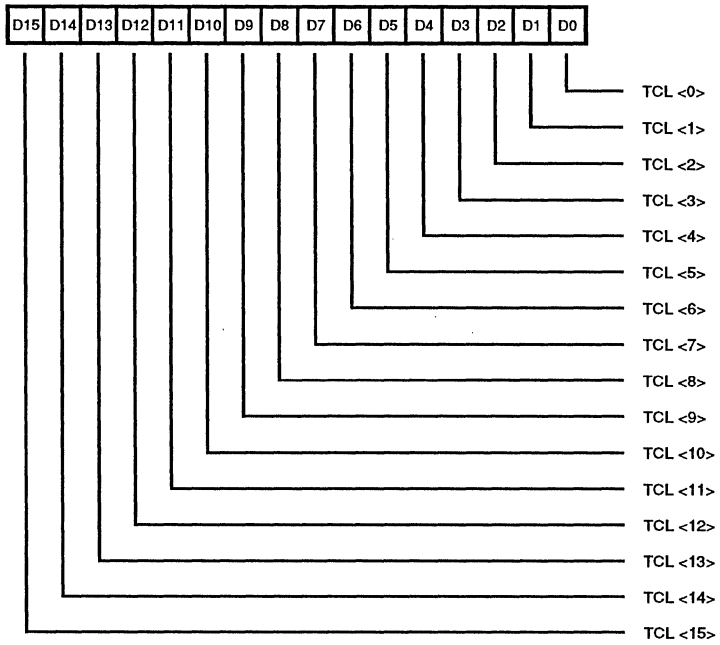


Figure 48. Transmit Count Limit Register

Address: 11110

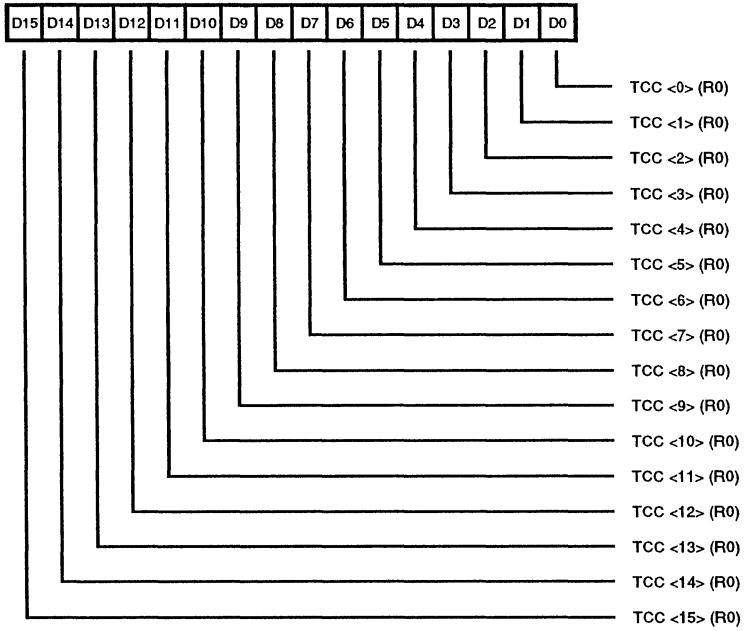


Figure 49. Transmit Character Count Register

Address: 11111

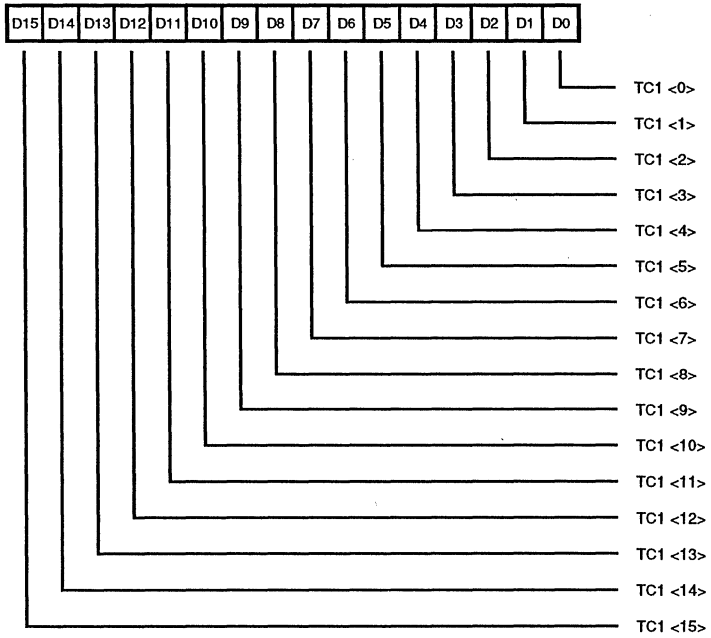
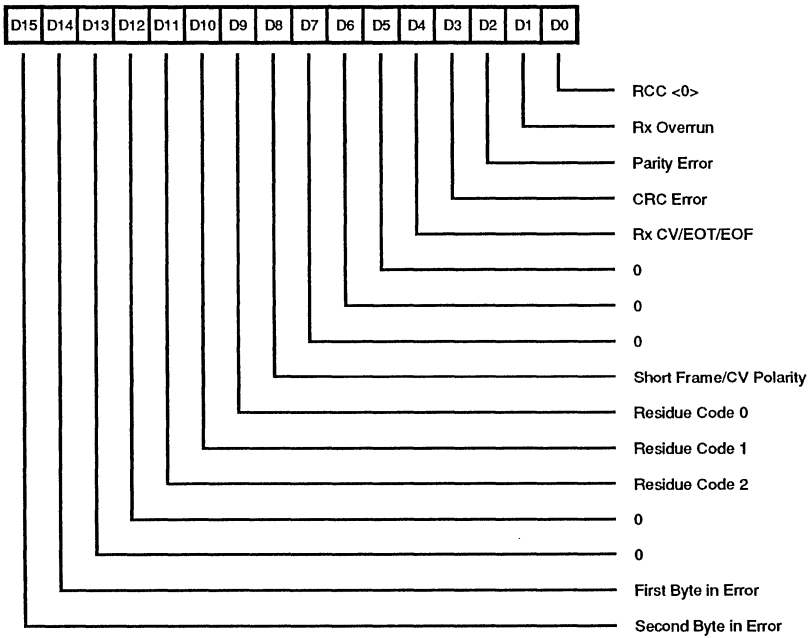


Figure 50. Time Constant 1 Register

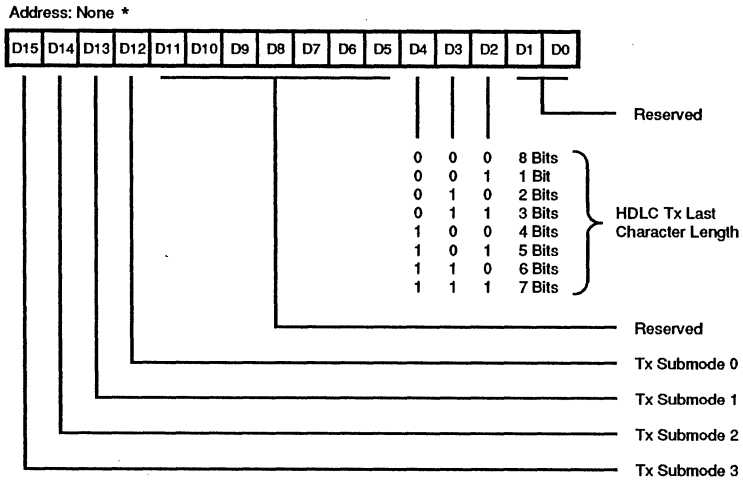
Address: None \*



\* Refer to Figure 22 (Channel Control Register)  
Bits 6-7 for Access Method

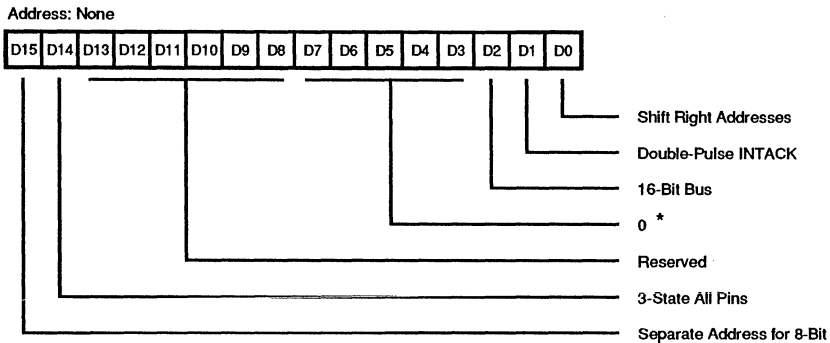
Figure 51. Receive Status Block Register





\* Refer to Figure 22 (Channel Control Register) Bits15-14 for Access Method

Figure 52. Transmit Status Block Register



\* Must be programmed as zero.

Figure 53. Bus Configuration Register

## USC TIMING

The USC interface timing is similar to that found on a static RAM, except that it is much more flexible. Up to eight separate timing strobe signals may be present on the interface: /DS, /RD, /WR, /PITACK, /RxACKA, /RxACKB, /TxACKA and /TxACKB. Only one of these timing strobes may be active at any time. Should the external logic

activate more than one of these strobes the USC will enter a pre-reset state that i hardware reset. Do not allow overlap of tir timing diagrams, beginning on the next p different bus transactions possible, wil setup, hold and delay times.

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins  
with respect to V<sub>SS</sub> ..... -0.3 V to +7.0 V  
Voltages on all inputs  
with respect to V<sub>SS</sub> ..... -0.3V to V<sub>CC</sub> +0.3V  
Operating Ambient  
Temperature ..... See Ordering Information  
Storage Temperature ..... -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

+4.5 V < V<sub>CC</sub> < +5.5 V  
GND = 0 V  
T<sub>A</sub> as specified in Ordering Information

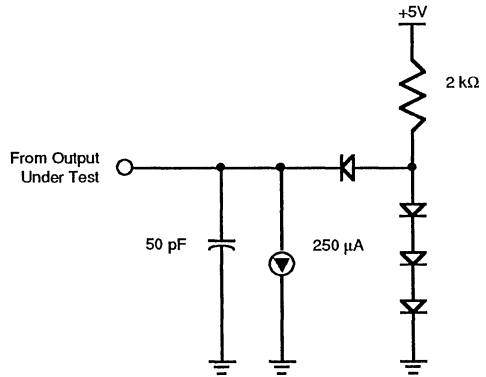


Figure 54. Standard Test Load

## TANCE

Parameter	Min	Max	Unit	Condition
Input Capacitance		10	pf	Unmeasured pins returned to Ground.
Output Capacitance		15	pf	
Bidirectional Capacitance		20	pf	

ified temperature range.

**EOUS** Transistor Count - 174,000

## DC CHARACTERISTICS

Z16C30

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC}+0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6\text{mA}$
$V_{OH2}$	Output High Voltage	$V_{CC}-0.8$			V	$I_{OH} = -250\ \mu\text{A}$
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = +2.0\text{mA}$
$I_{IL}$	Input Leakage			$\pm 10.00$	$\mu\text{A}$	$0.4 < V_{IN} < +2.4\text{V}$
$I_{OL}$	Output Leakage			$\pm 10.00$	$\mu\text{A}$	$0.4 < V_{OUT} < +2.4\text{V}$
$I_{CC1}$	$V_{CC}$ Supply Current		7	50	mA	$V_{CC}=5\text{V } V_{IH}= 4.8\text{V } V_{IL}= 0.2\text{V}$

### Note:

$V_{CC} = 5\text{V} \pm 10\%$  unless otherwise specified, over specified temperature range.

## AC CHARACTERISTICS

Z16C30

No	Symbol	Parameter	[6] 10 Mbps		[7] 20 Mbps		Units	No'
			Min	Max	Min	Max		
1	Tcyc	Bus Cycle Time	160		110		ns	
2	TwASl	/AS Low Width	40		30		ns	
3	TwASh	/AS High Width	90		60		ns	
4	TwDSL	/DS Low Width	70		60		ns	
5	TwDSh	/DS High Width	60		50		ns	
6	TdAS(DS)	/AS Rise to /DS Fall Delay Time	5		5		ns	
7	TdDS(AS)	/DS Rise to /AS Fall Delay Time	5		5		ns	
8	TdDS(DRa)	/DS Fall to Data Active Delay	0		0		ns	
9	TdDS(DRv)	/DS Fall to Data Valid Delay		85		60	n	
10	TdDS(DRn)	/DS Rise to Data Not Valid Delay	0		0		r	
11	TdDS(DRz)	/DS Rise to Data Float Delay		20		20	r	
12	TsCS(AS)	/CS to /AS Rise Setup Time	15		15		r	
13	ThCS(AS)	/CS to /AS Rise Hold Time	0		0		t	
14	TsADD(AS)	Direct Address to /AS Rise Setup Time	15		15			
15	ThADD(AS)	Direct Address to /AS Rise Hold Time	5		5			
16	TsSIA(AS)	/SITACK to /AS Rise Setup Time	15		15			
17	ThSIA(AS)	/SITACK to /AS Rise Hold Time	5		5			
18	TsAD(AS)	Address to /AS Rise Setup Time	15		15			
19	ThAD(AS)	Address to /AS Rise Hold Time	5		5			
20	TsRW(DS)	R/W to /DS Fall Setup Time	0		0			

**AC CHARACTERISTICS**  
Z16C30

No	Symbol	Parameter	[6] 10 Mbps		[7] 20 Mbps		Units	Note
			Min	Max	Min	Max		
21	ThRW(DS)	R//W to /DS Fall Hold Time	25		25		ns	
22	TsDSi(RRQ)	/DS Fall to /RxREQ Inactive Delay		60		60	ns	[4]
23	TdDSr(RRQ)	/DS Rise to /RxREQ Active Delay	0		0		ns	
24	TsDW(DS)	Write Data to /DS Rise Setup Time	30		30		ns	
25	ThDW(DS)	Write Data to /DS Rise Hold Time	0		0		ns	
26	TdDSi(TRQ)	/DS Fall to /TxREQ Inactive Delay		65		65	ns	[5]
27	TdDSr(TRQ)	/DS Rise to /TxREQ Active Delay	0		0		ns	
28	TwRDI	/RD Low Width	70		60		ns	
29	TwRDh	/RD High Width	60		50		ns	
30	TdAS(RD)	/AS Rise to /RD Fall Delay Time	5		5		ns	
31	TdRD(AS)	/RD Rise to /AS Fall Delay Time	5		5		ns	
32	TdRD(DRa)	/RD Fall to Data Active Delay	0		0		ns	
33	TdRD(DRv)	/RD Fall to Data Valid Delay		85		60	ns	
34	TdRD(DRn)	/RD Rise to Data Not Valid Delay	0		0		ns	
35	TdRD(DRz)	/RD Rise to Data Float Delay		20		20	ns	
36	TdRDi(RRQ)	/RD Fall to /RxREQ Inactive Delay		60		60	ns	[4]
37	TdRDr(RRQ)	/RD Rise to /RxREQ Active Delay	0		0		ns	
38	TwWRI	/WR Low Width	70		60		ns	
39	TwWRh	/WR High Width	60		50		ns	
40	TdAS(WR)	/AS Rise to /WR Fall Delay Time	5		5		ns	
1	TdWR(AS)	/WR Rise to /AS Fall Delay Time	5		5		ns	
	TsDW(WR)	Write Data to /WR Rise Setup Time	30		30		ns	
	ThDW(WR)	Write Data to /WR Rise Hold Time	0		0		ns	
	TdWRi(TRQ)	/WR Fall to /TxREQ Inactive Delay		65		65	ns	[5]
	TdWRr(TRQ)	/WR Rise to /TxREQ Active Delay	0		0		ns	
	tCS(DS)	/CS to /DS Fall Setup Time	0		0		ns	[2]
	tCS(DS)	/CS to /DS Fall Hold Time	25		25		ns	[2]
	tADD(DS)	Direct Address to /DS Fall Setup Time	5		5		ns	[1,2]
	tADD(DS)	Direct Address to /DS Fall Hold Time	25		25		ns	[1,2]
	tIA(DS)	/SITACK to /DS Fall Setup Time	5		5		ns	[2]
	tIA(DS)	/SITACK to /DS Fall Hold Time	25		25		ns	[2]
	tIR(D)	/CS to /RD Fall Setup Time	0		0		ns	[2]
	tIR(D)	/CS to /RD Fall Hold Time	25		25		ns	[2]
	tID(RD)	Direct Address to /RD Fall Setup Time	5		5		ns	[1,2]
	tID(RD)	Direct Address to /RD Fall Hold Time	25		25		ns	[1,2]
	tIA(RD)	/SITACK to /RD Fall Setup Time	5		5		ns	[2]
	tIA(RD)	/SITACK to /RD Fall Hold Time	25		25		ns	[2]
	tIR(WR)	/CS to /WR Fall Setup Time	0		0		ns	[2]
	tIR(WR)	/CS to /WR Fall Hold Time	25		25		ns	[2]
	tADD(WR)	Direct Address to /WR Fall Setup Time	5		5		ns	[1,2]

## AC CHARACTERISTICS

### Z16C30

No	Symbol	Parameter	[6] 10 Mbps		[7] 20 Mbps		Units	Note
			Min	Max	Min	Max		
61	ThADD(WR)	Direct Address to /WR Fall Hold Time	25		25		ns	[1,2]
62	TsSIA(WR)	/SITACK to /WR Fall Setup Time	5		5		ns	[2]
63	ThSIA(WR)	/SITACK to /WR Fall Hold Time	25		25		ns	[2]
64	TwRAKI	/RxACK Low Width	70		60		ns	
65	TwRAKh	/RxACK High Width	60		50		ns	
66	TdRAK(DRa)	/RxACK Fall to Data Active Delay	0		0		ns	
67	TdRAK(DRv)	/RxACK Fall to Data Valid Delay		85		60	ns	
68	TdRAK(DRn)	/RxACK Rise to Data Not Valid Delay	0		0		ns	
69	TdRAK(DRz)	/RxACK Rise to Data Float Delay		20		20	ns	
70	TdRAKf(RRQ)	/RxACK Fall to /RxREQ Inactive Delay		60		60	ns	[4]
71	TdRAKr(RRQ)	/RxACK Rise to /RxREQ Active Delay	0		0		ns	
72	TwTAKI	/TxACK Low Width	70		60		ns	
73	TwTAKh	/TxACK High Width	60		50		ns	
74	TsDW(TAK)	Write Data to /TxACK Rise Setup Time	30		30		ns	
75	ThDW(TAK)	Write Data to /TxACK Rise Hold Time	0		0		ns	
76	TdTAKf(TRQ)	/TxACK Fall to /TxREQ Inactive Delay		65		65	ns	[5]
77	TdTAKr(TRQ)	/TxACK Rise to /TxREQ Active Delay	0		0		ns	
78	TdDSI(RDY)	/DS Fall (INTACK) to /RDY Fall Delay		200		200	ns	
79	TdRDY(DRv)	/RDY Fall to Data Valid Delay		40		40	ns	
80	TdDSr(RDY)	/DS Rise to /RDY Rise Delay		40		40	ns	
81	TsIEI(DSI)	IEI to /DS Fall (INTACK) Setup Time	60		10		ns	
82	ThIEI(DSI)	IEI to /DS Rise (INTACK) Hold Time	0		0		ns	
83	TdIEI(IEO)	IEI to IEO Delay		60		30	ns	
84	TdAS(IEO)	/AS Rise (Intack) to IEO Delay		60		60	ns	
85	TdDSI(INT)	/DS Fall (INTACK) to /INT Inactive Delay		200		200	ns	
86	TdDSI(Wf)	/DS Fall (INTACK) to /WAIT Fall Delay		40		40	ns	
87	TdDSI(Wr)	/DS Fall (INTACK) to /WAIT Rise Delay		200		200	ns	
88	TdW(DRv)	/WAIT Rise to Data Valid Delay		40		40	ns	
89	TdRDI(RDY)	/RD Fall (INTACK) to /RDY Fall Delay		200		200	ns	
90	TdRDf(RDY)	/RD Rise to /RDY Rise Delay		40		40	ns	
91	TsIEI(RDI)	IEI to /RD Fall (INTACK) Setup Time	60		10		ns	
92	ThIEI(RDI)	IEI to /RD Rise (INTACK) Hold Time	0		0		ns	
93	TdRDI(INT)	/RD Fall (INTACK) to /INT Inactive Delay		200		200	ns	
94	TdRDI(Wf)	/RD Fall (INTACK) to /WAIT Fall Delay		40		40	ns	
95	TdRDI(Wr)	/RD Fall (INTACK) to /WAIT Rise Delay		200		200	ns	
96	TwPIAI	/PITACK Low Width	70		60		ns	
97	TwPIAh	/PITACK High Width	60		50		ns	
98	TdAS(PIA)	/AS Rise to /PITACK Fall Delay Time	5		5		ns	
99	TdPIA(AS)	/PITACK Rise to /AS Fall Delay Time	5		5		ns	
100	TdPIA(DRa)	/PITACK Fall to Data Active Delay	0		0		ns	

## AC CHARACTERISTICS

Z16C30

No	Symbol	Parameter	[6] 10 Mbps		[7] 20 Mbps		Units	Note
			Min	Max	Min	Max		
101	TdPIA(DRn)	/PITACK Rise to Data Not Valid Delay	0		0		ns	
102	TdPIA(DRz)	/PITACK Rise to Data Float Delay		20		20	ns	
103	TsIEI(PIA)	IEI to /PITACK Fall Setup Time	60		10		ns	
104	ThIEI(PIA)	IEI to /PITACK Rise Hold Time	0		0		ns	
105	TdPIA(IEO)	/PITACK Fall to IEO Delay		60		60	ns	
106	TdPIA(INT)	/PITACK Fall to /INT Inactive Delay		200		200	ns	
107	TdPIAf(RDY)	/PITACK Fall to /RDY Fall Delay		200		200	ns	
108	TdPIAr(RDY)	/PITACK Rise to /RDY Rise Delay		40		40	ns	
109	TdPIA(Wf)	/PITACK Fall to /WAIT Fall Delay		40		40	ns	
110	TdPIA(Wr)	/PITACK Fall to /WAIT Rise Delay		200		200	ns	
111	TdSIA(INT)	/SITACK Fall to IEO Inactive Delay		200		200	ns	[2]
112	TwSTBh	/Strobe High Width	60		50		ns	[3]
113	TwRESL	/RESET Low Width	170		170		ns	
114	TwRESH	/RESET High Width	60		60		ns	
115	Tdres(STB)	/RESET Rise to /STB Fall	60		60		ns	[3]
116	TdDSf(RDY)	/DS Fall to /RDY Fall Delay		50		50	ns	
117	TdWRf(RDY)	/WR Fall to /RDY Fall Delay		50		50	ns	
118	TdWRr(RDY)	/WR Rise to /RDY Rise Delay		40		40	ns	
119	TdRDf(RDY)	/RD Fall to /RDY Fall Delay		50		50	ns	
120	TdRAKf(RDY)	/RxACK Fall to /RDY Fall Delay		50		50	ns	
121	TdRAKr(RDY)	/RxACK Rise to /RDY Rise Delay		40		40	ns	
122	TdTAKf(RDY)	/TxACK Fall to /RDY Fall Delay		50		50	ns	
123	TdTAKr(RDY)	/TxACK Rise to /RDY Rise Delay		40		40	ns	

**Notes:**

- 1 Direct address is any of A//B, D//C or AD15-AD8 used as an address bus.
- 2 This parameter applies only when /AS is not present.
- 3 Strobe (/STB) is any of /DS, /RD, /WR, /PITACK, /RxACK or /TxACK.
- 4 This parameter applies only if read empties the receive FIFO.
- 5 This parameter applies only if write fills the transmit FIFO.
- 6  $V_{DD} = 5V \pm 10\%$
- 7  $V_{DD} = 5V \pm 5\%$ . AC timings for 20 Mbps are preliminary data.

## TIMING DIAGRAMS

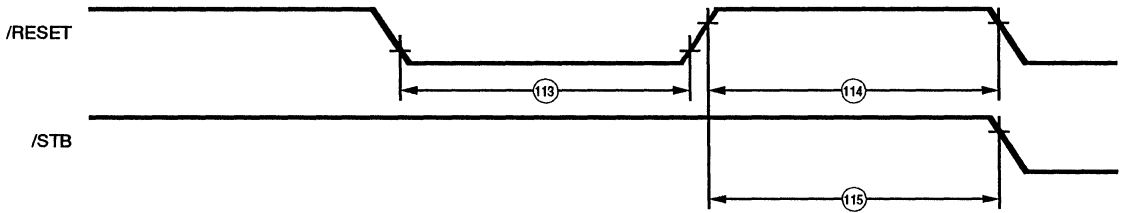


Figure 55. Reset Timing

**Note:**  
**/STB** is any of **/DS**, **/RD**, **/WR**, **/PITACK**, **/RxACK**, or **/TxACK**.

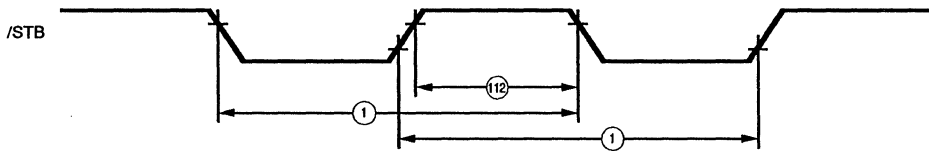


Figure 56. Bus Cycle Timing

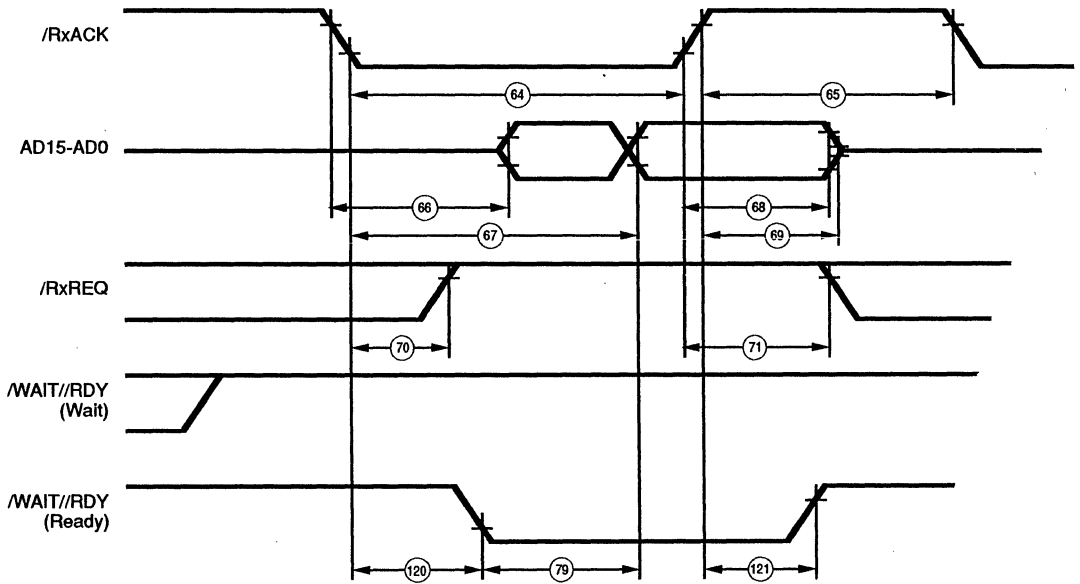


Figure 57. DMA Read Cycle

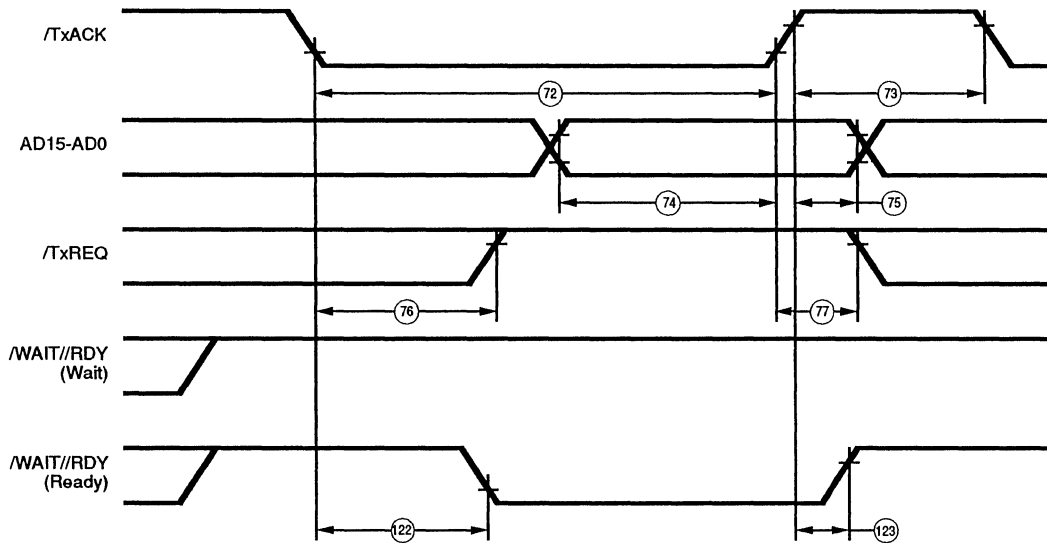
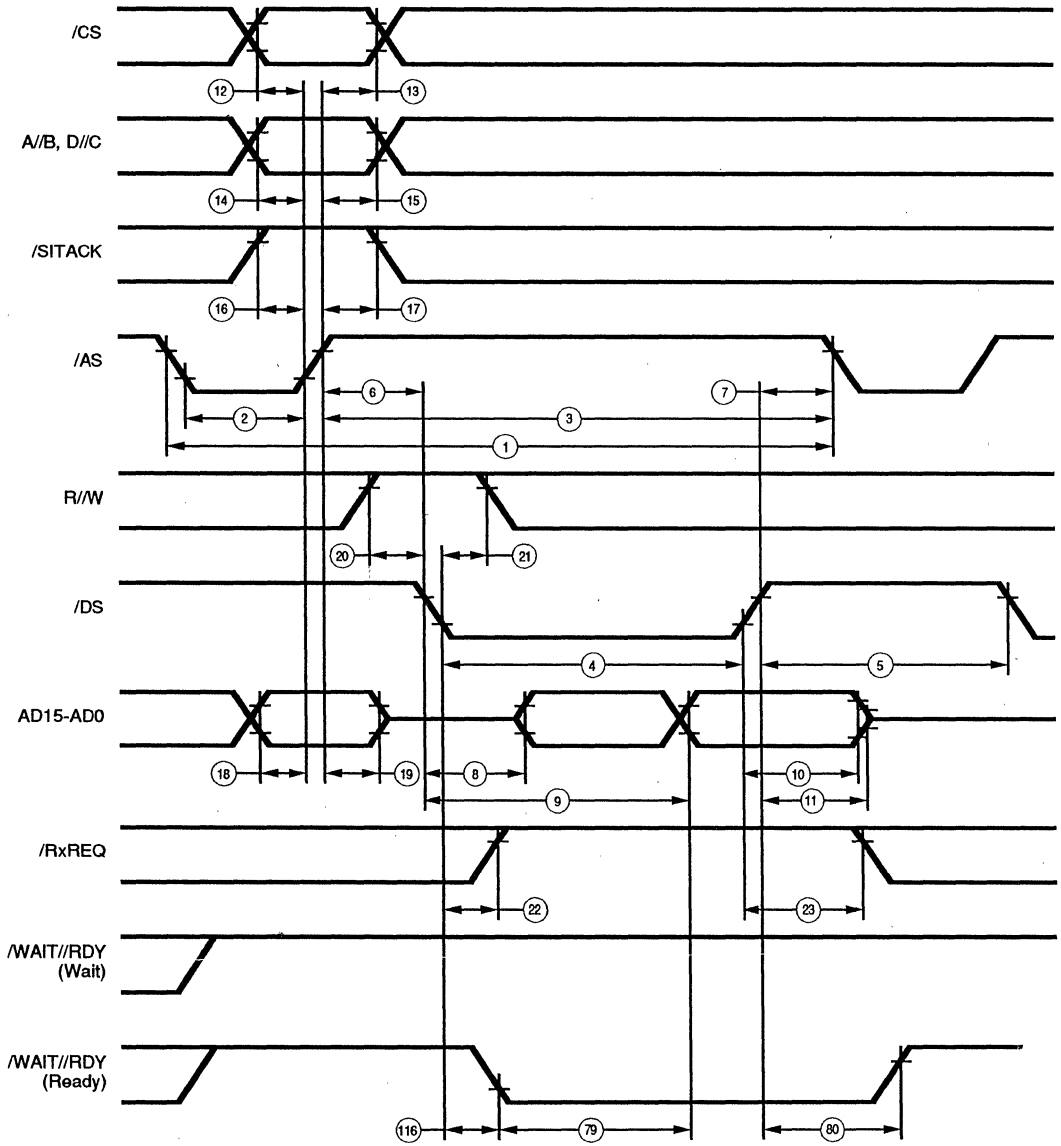


Figure 58. DMA Write Cycle



**TIMING DIAGRAMS** (Continued)



**Figure 59. Multiplexed /DS Read Cycle**

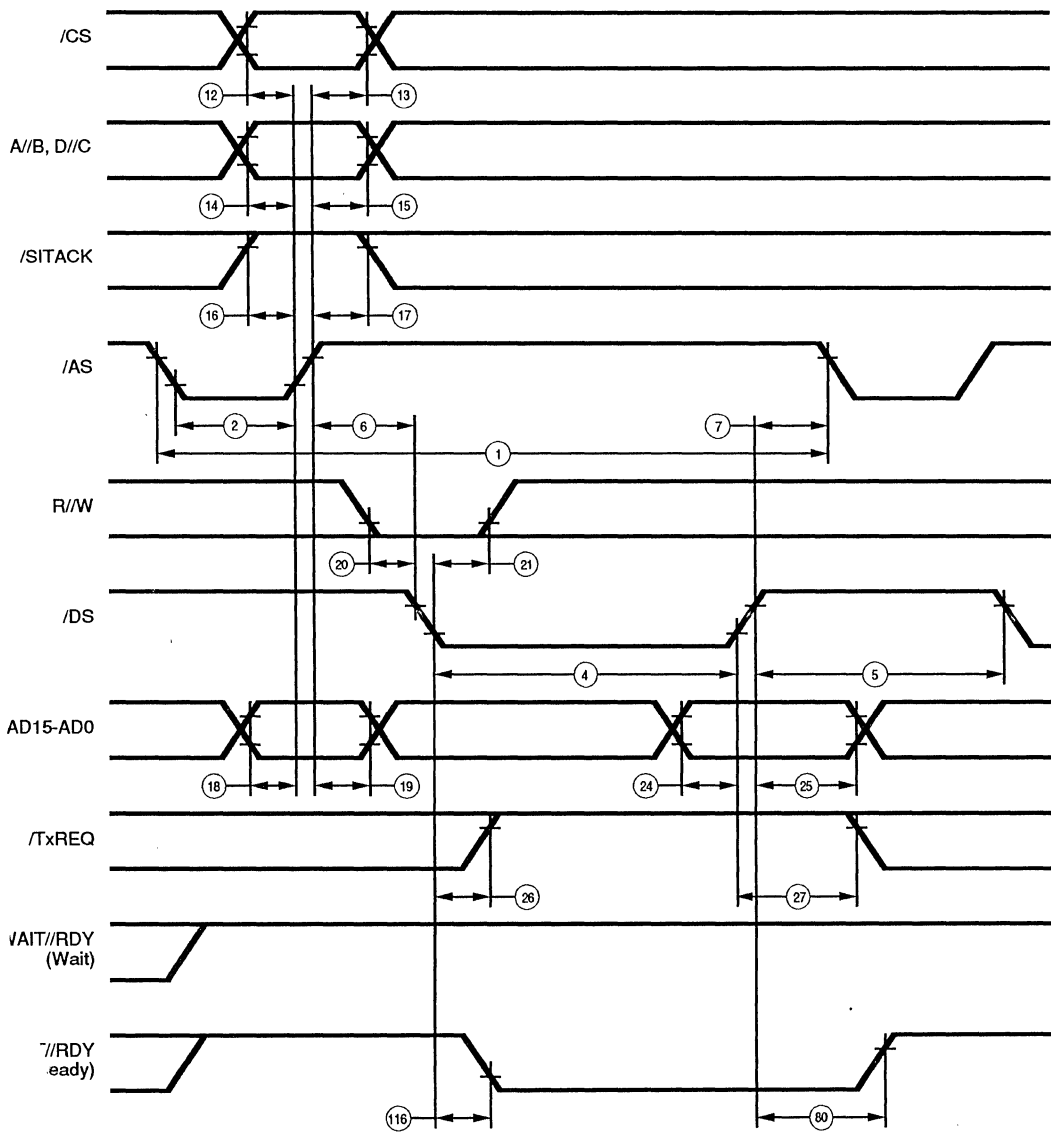
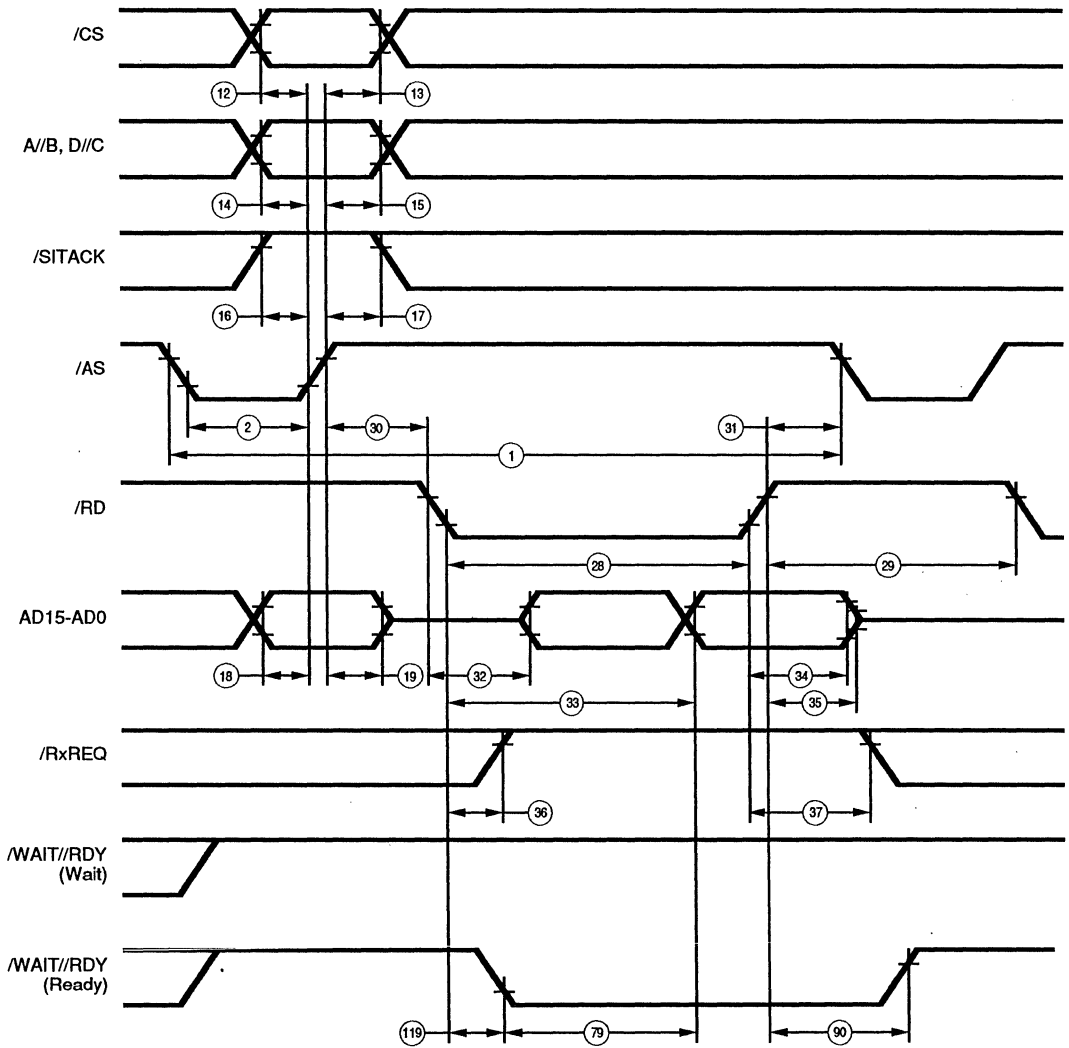


Figure 60. Multiplexed /DS Write Cycle

**TIMING DIAGRAMS (Continued)**



**Figure 61. Multiplexed /RD Read Cycle**

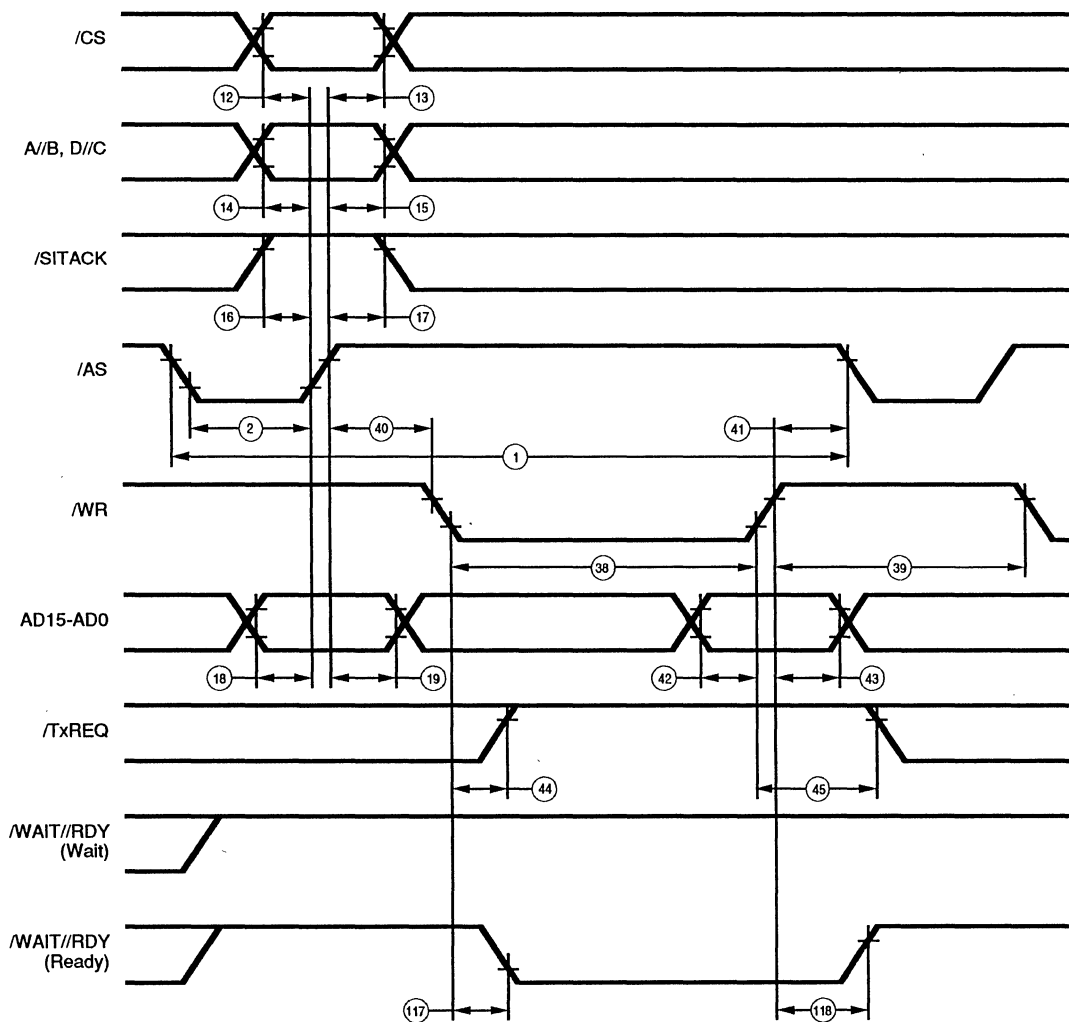
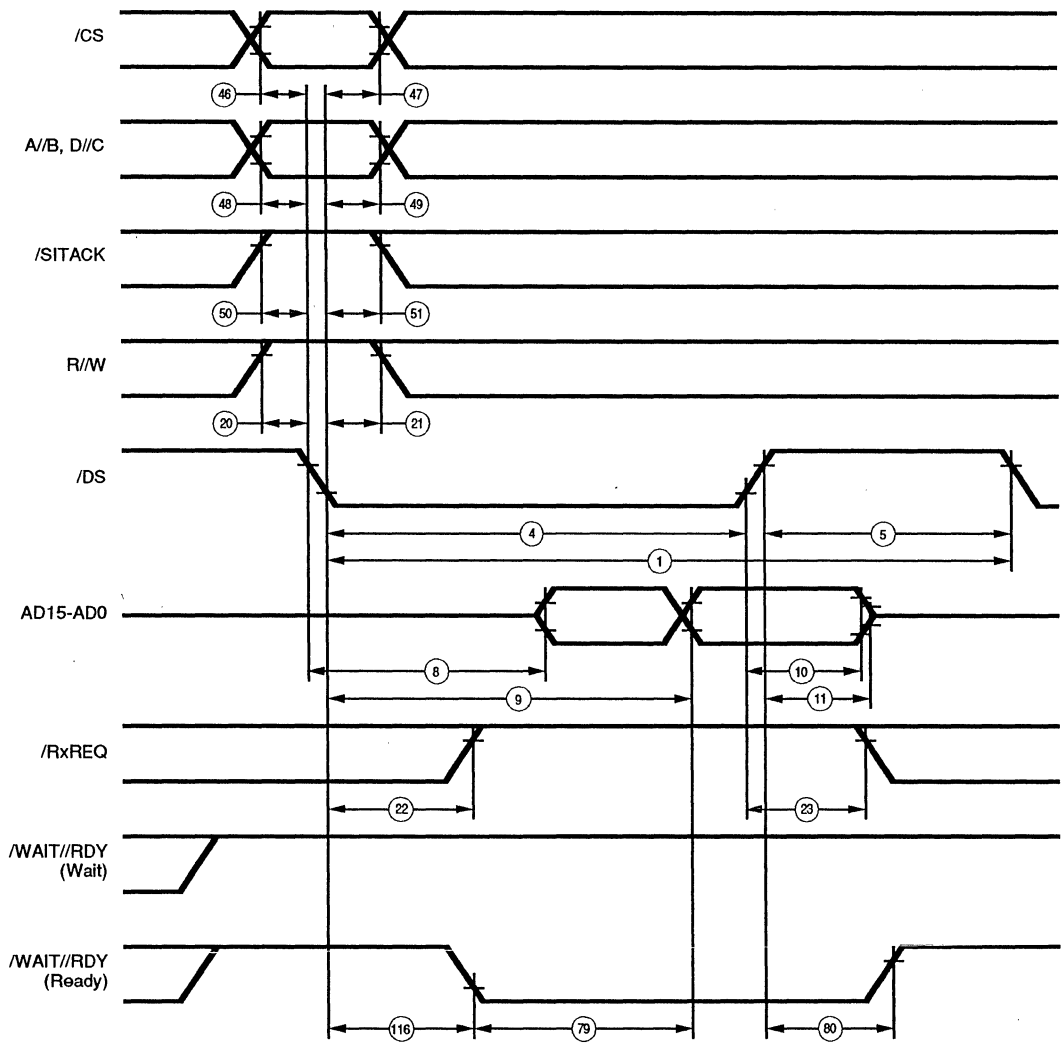


Figure 62. Multiplexed /WR Write Cycle

**TIMING DIAGRAMS (Continued)**



**Figure 63. Non-Multiplexed /DS Read Cycle**

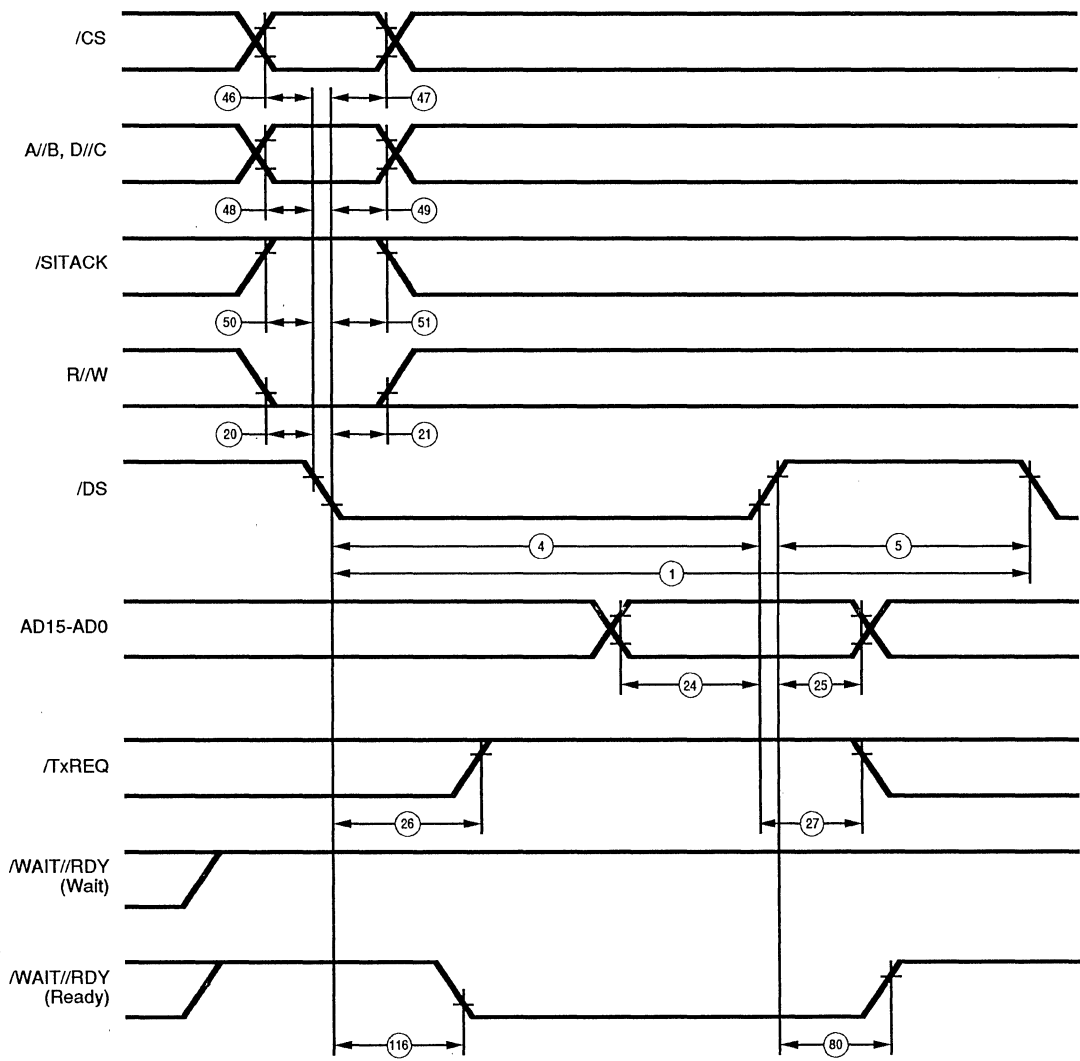
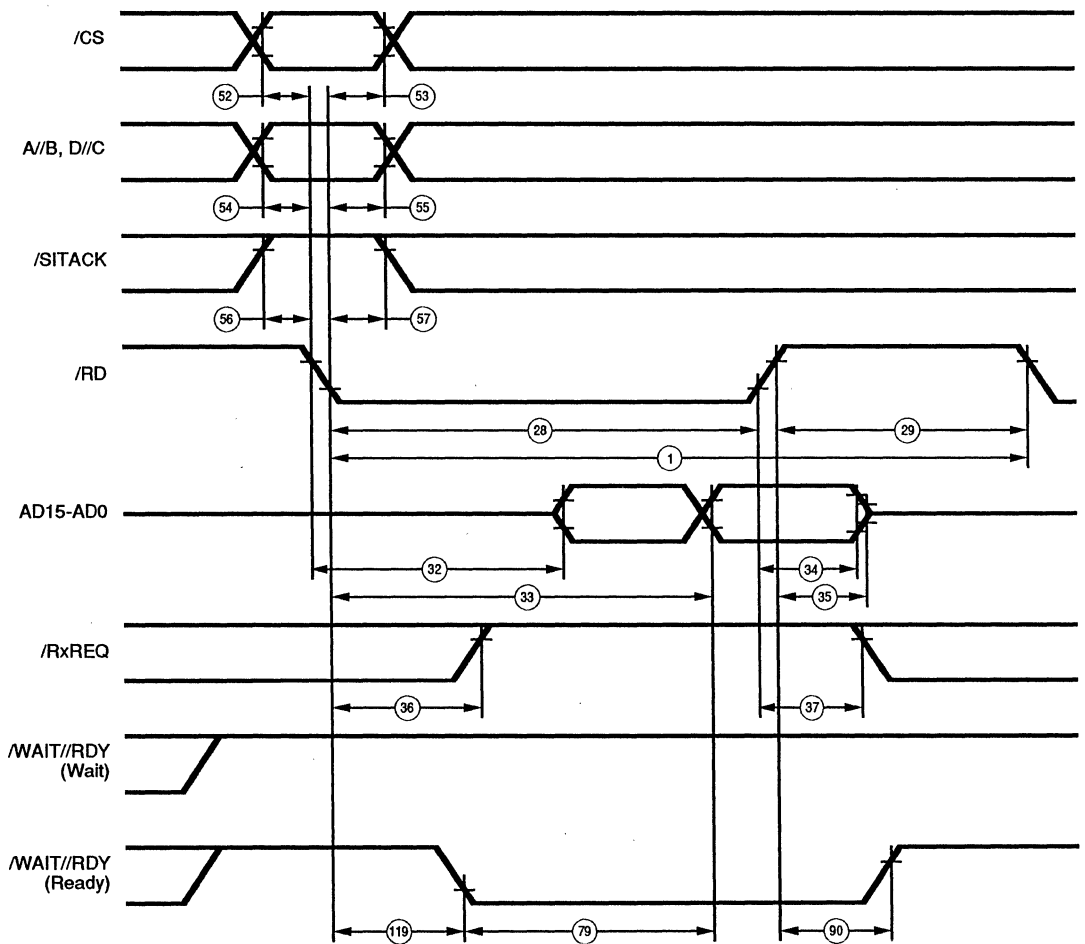


Figure 64. Non-Multiplexed /DS Write Cycle

**TIMING DIAGRAMS** (Continued)



**Figure 65. Non-Multiplexed /RD Read Cycle**

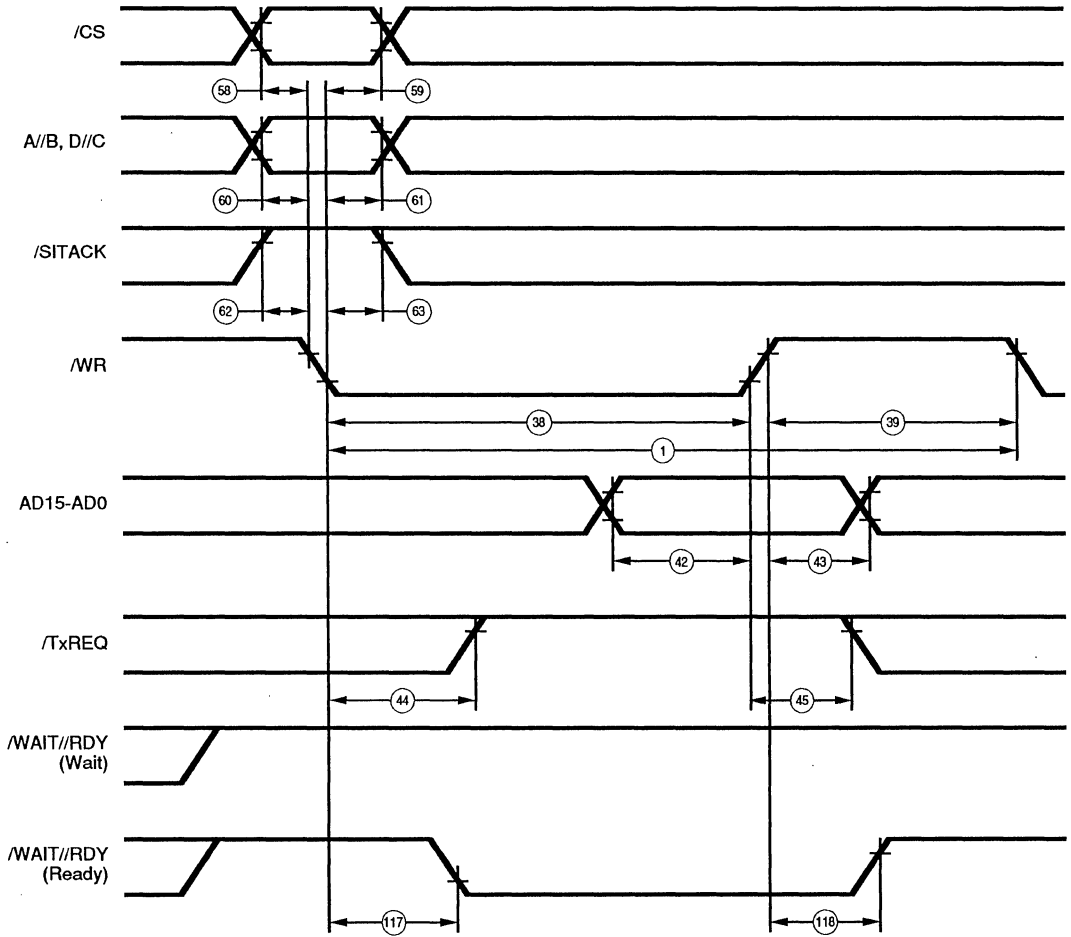
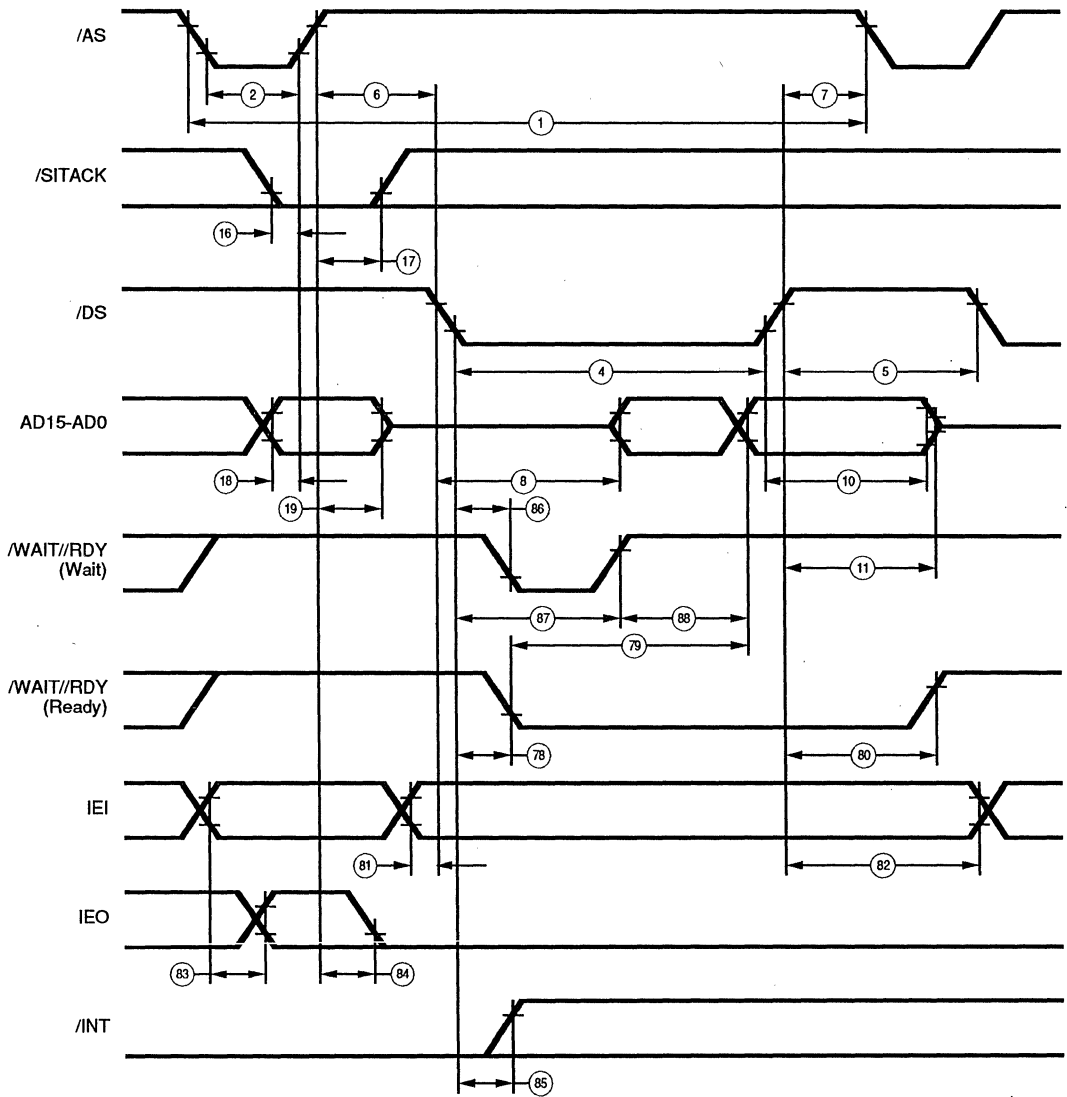


Figure 66. Non-Multiplexed /WR Write Cycle



**TIMING DIAGRAMS (Continued)**



**Figure 67. Multiplexed /DS Interrupt Acknowledged Cycle**

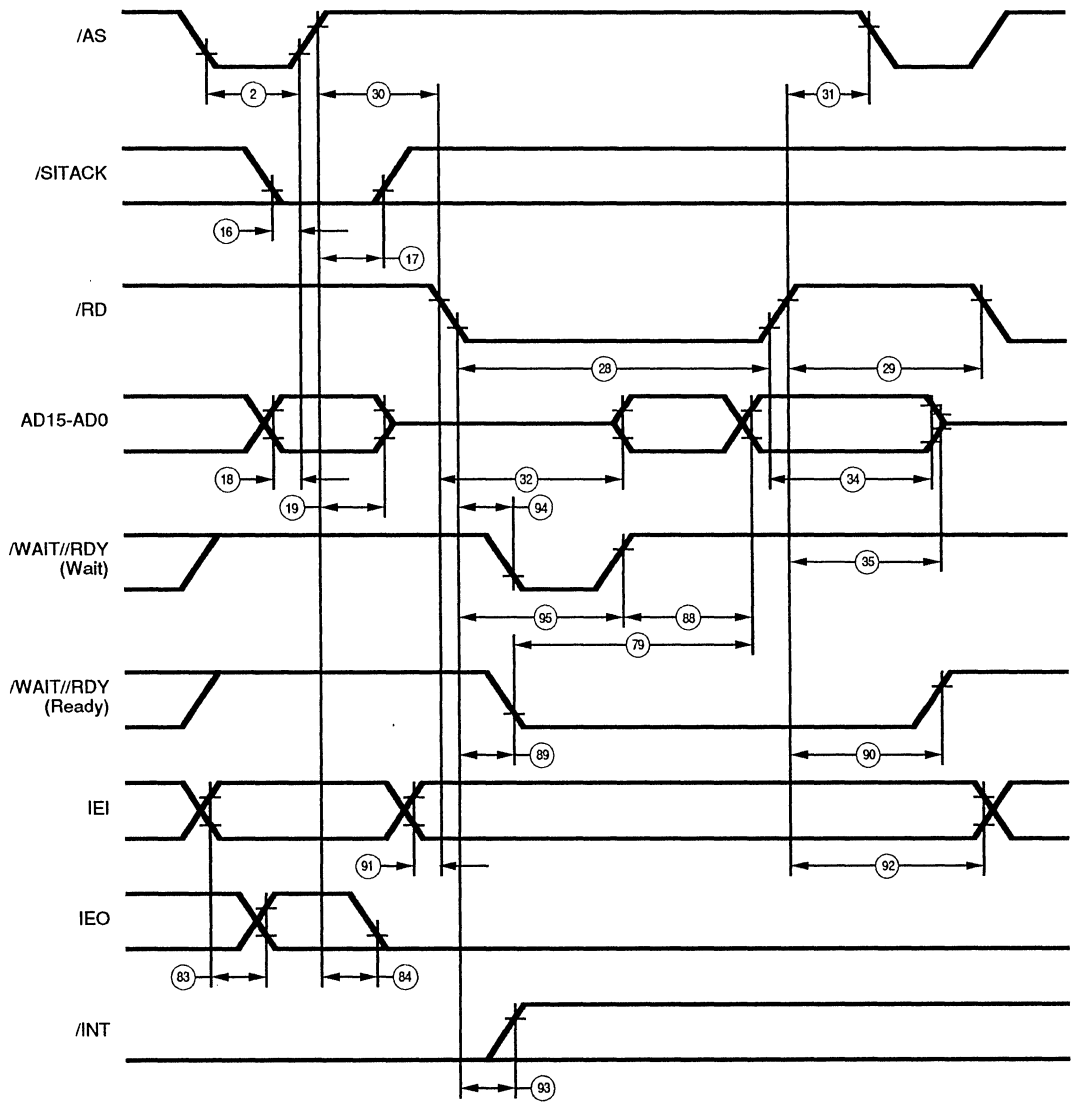
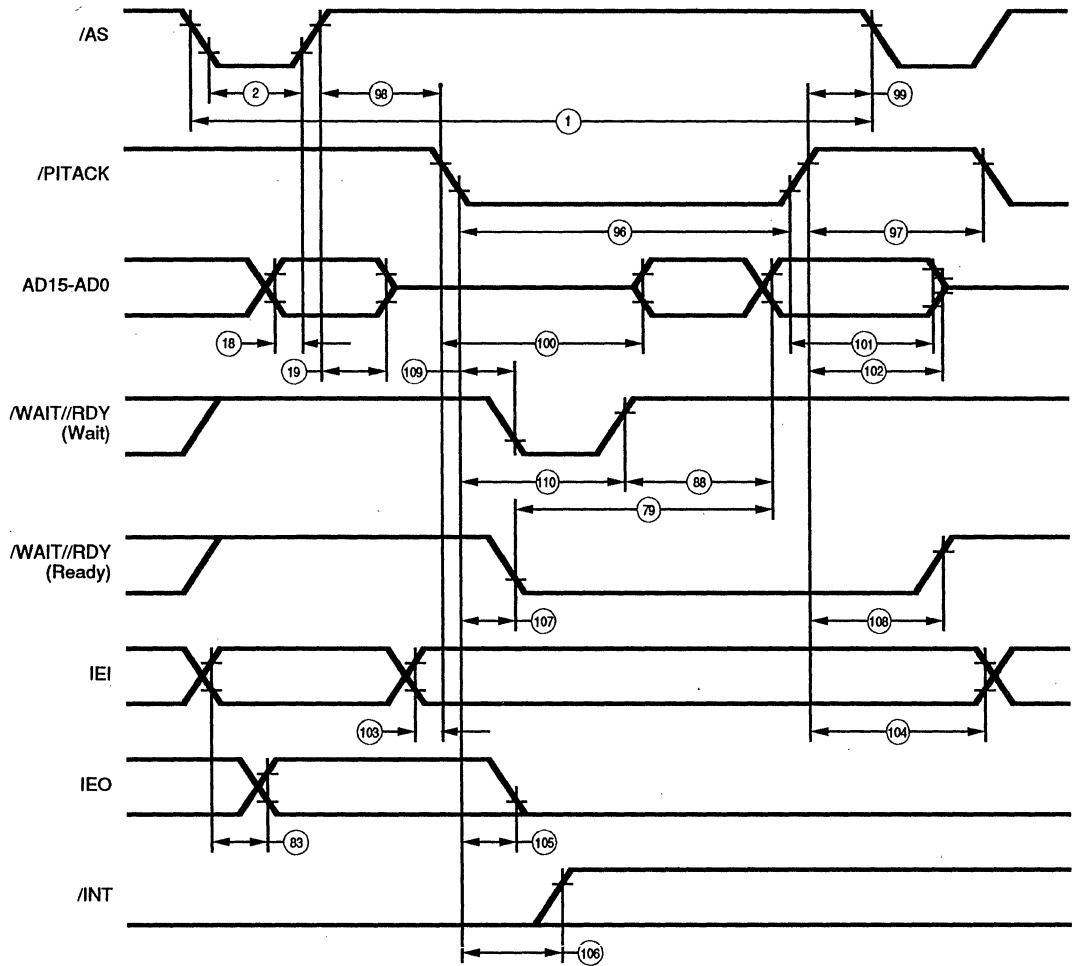


Figure 68. Multiplexed /RD Interrupt Acknowledge Cycle

**TIMING DIAGRAMS** (Continued)



**Figure 69. Multiplexed Pulsed Interrupt Acknowledge Cycle**

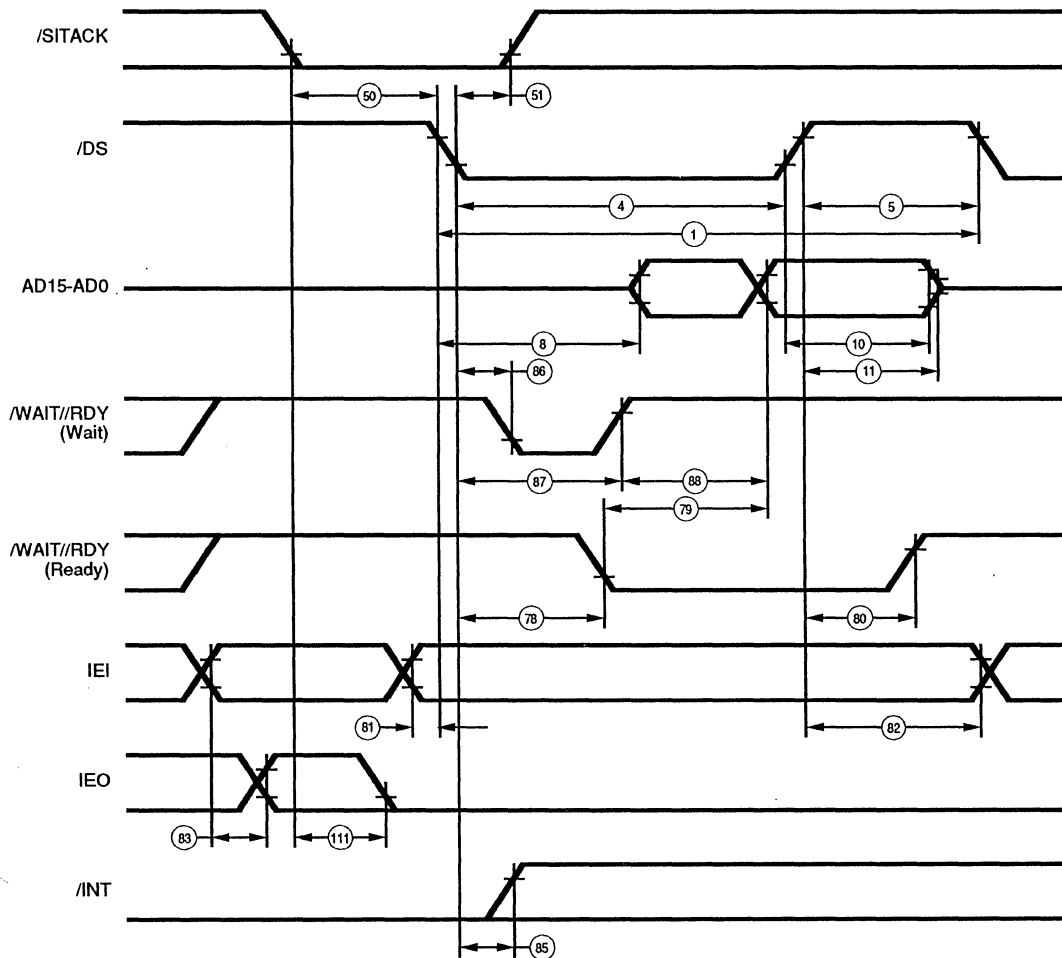
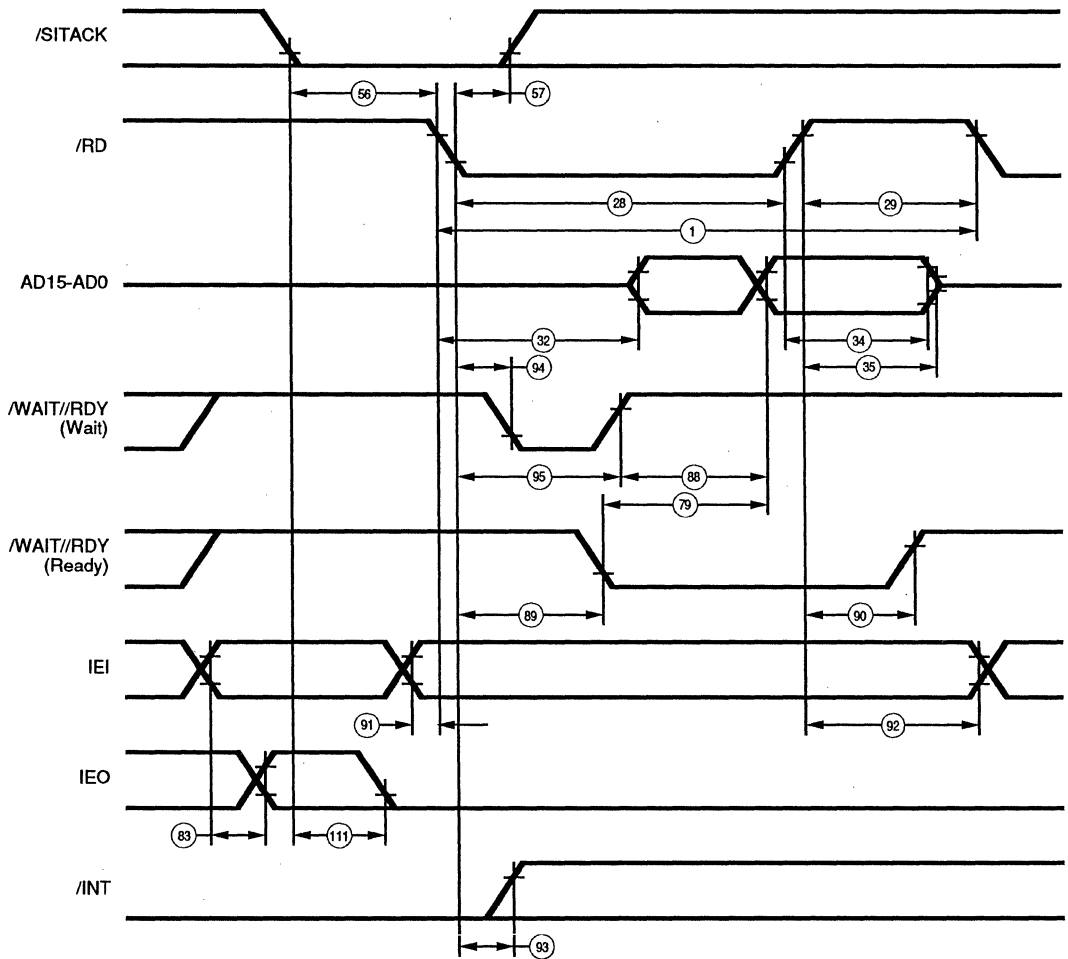


Figure 70. Non-Multiplexed /DS Interrupt Acknowledge Cycle

**TIMING DIAGRAMS** (Continued)



**Figure 71. Non-Multiplexed /RD Interrupt Acknowledge Cycle**

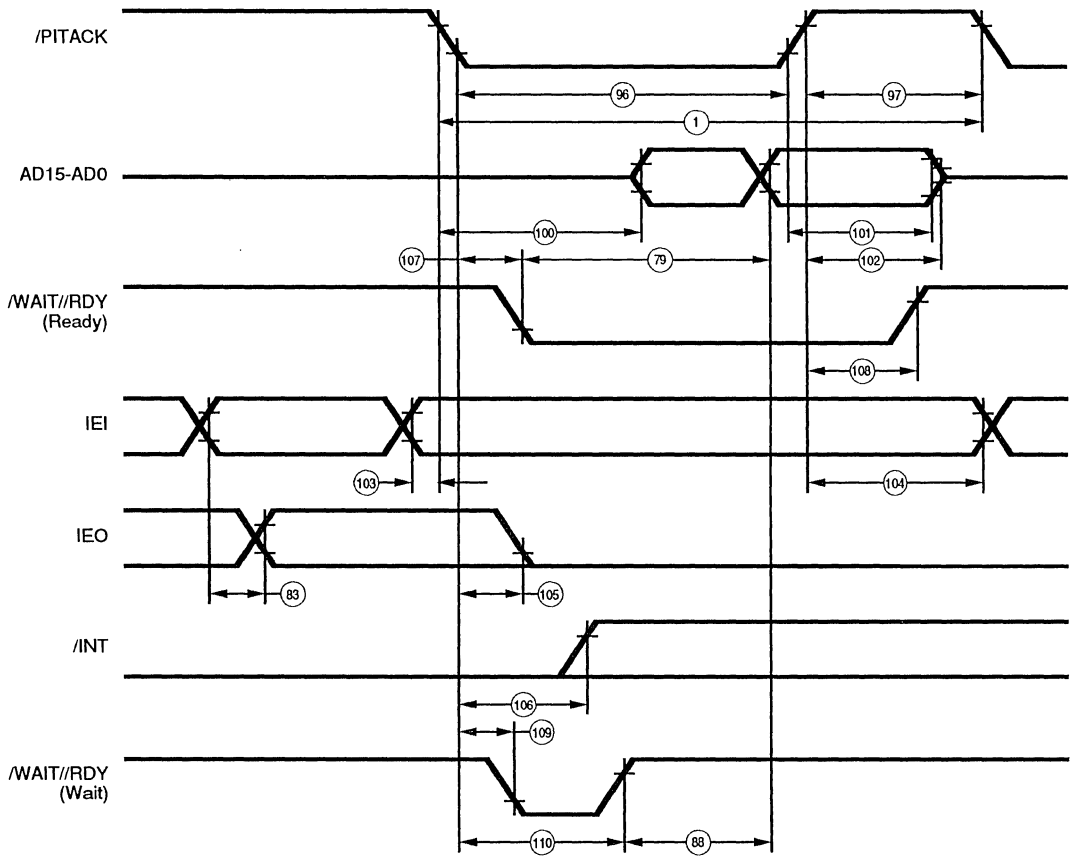
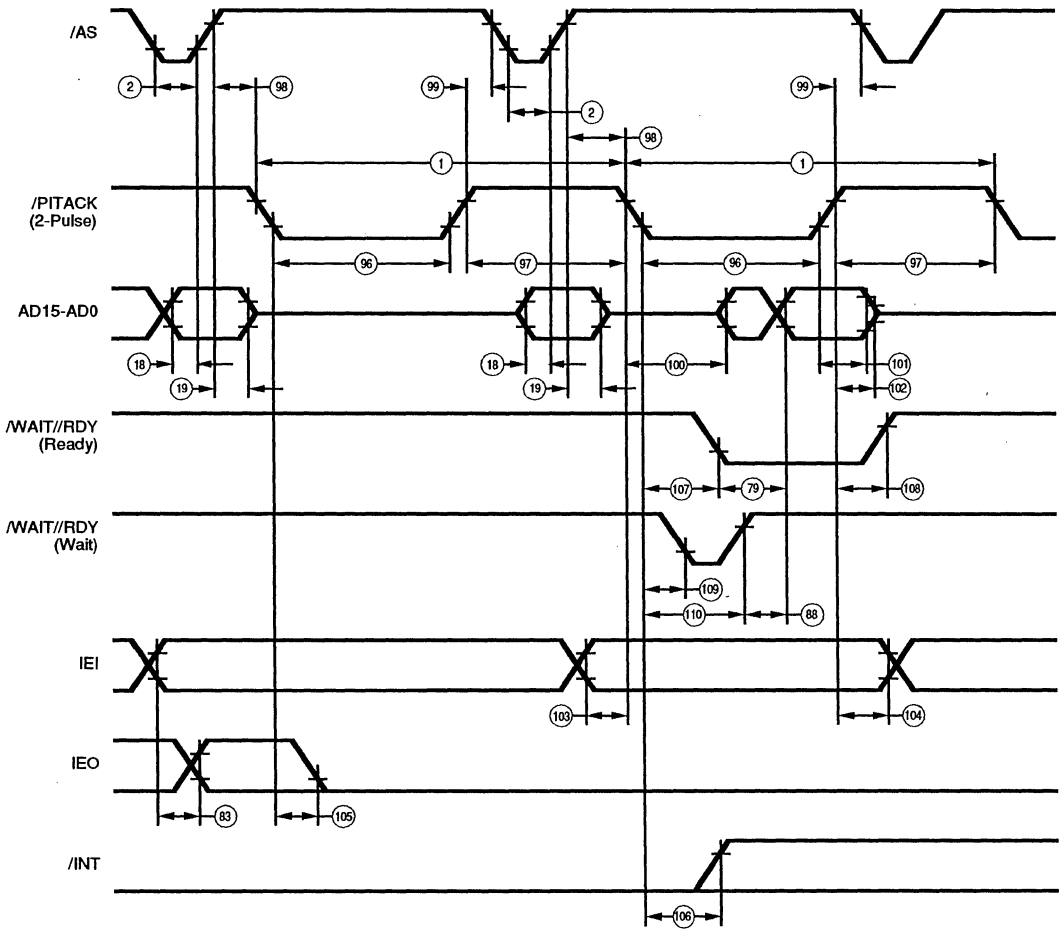


Figure 72. Non-Multiplexed Pulsed Interrupt Acknowledge Cycle

**TIMING DIAGRAMS** (Continued)



**Figure 73. Multiplexed Double-Pulse Intack Cycle**

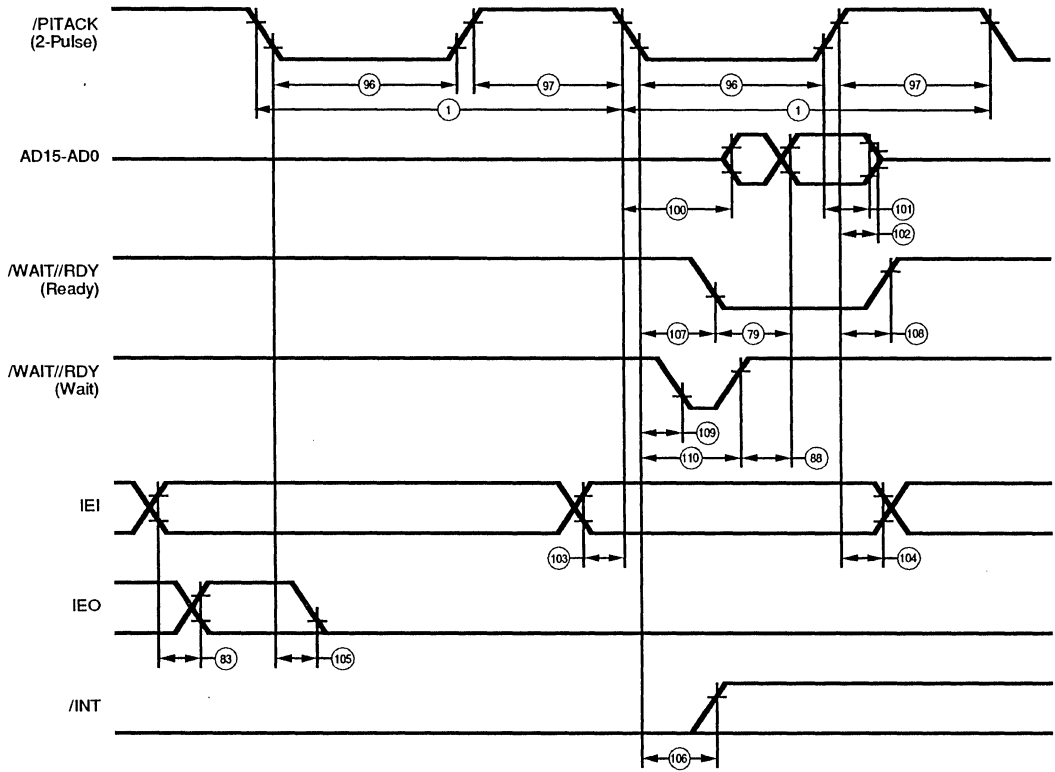
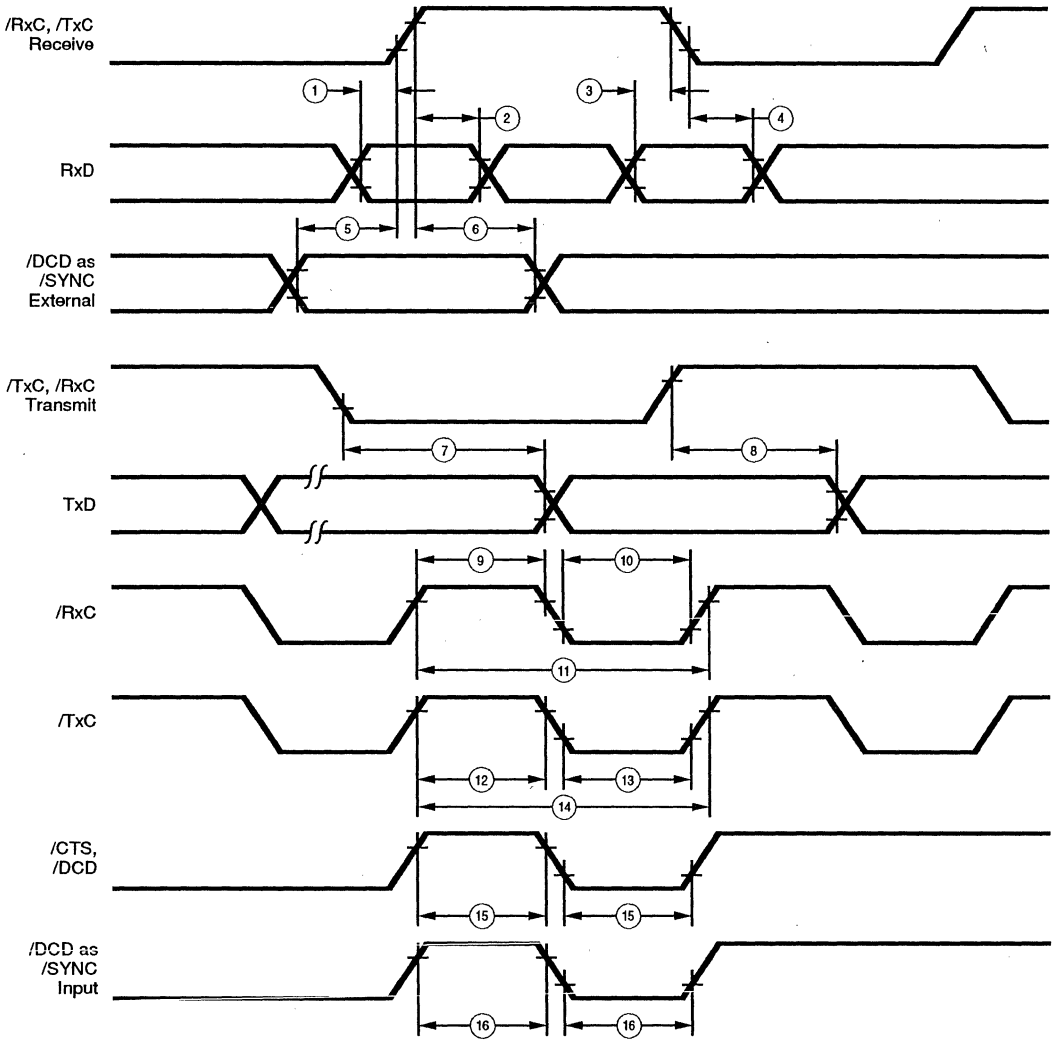


Figure 74. Non-Multiplexed Double-Pulse Intack Cycle



**TIMING DIAGRAMS (Continued)**



**Figure 75. Z16C30 General Timing**

## AC CHARACTERISTICS

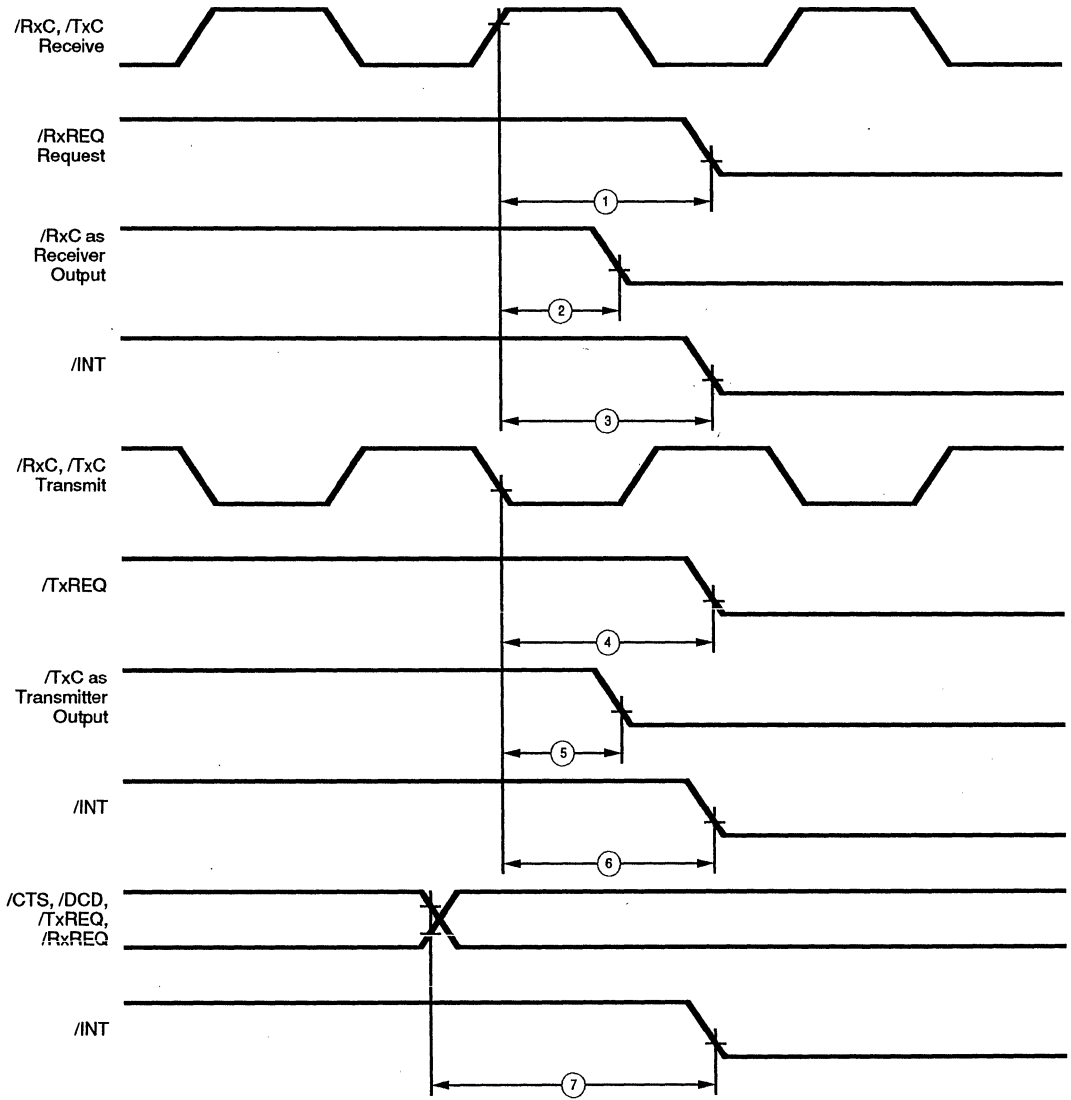
### Z16C30 General Timing

No	Symbol	Parameter	[4] 20 Mbps		[5] 10 Mbps		Units	Note
			Min	Max	Min	Max		
1	TsRxD(RxCr)	RxD to /RxC Rise Setup Time (x1 Mode)	0		0		ns	[1]
2	ThRxD(RxCr)	RxD to /RxC Rise Hold Time (x1 Mode)	20		40		ns	[1]
3	TsRxD(RxCf)	RxD to /RxC Fall Setup Time (x1 Mode)	0		0		ns	[1,3]
4	ThRxD(RxCf)	RxD to /RxC Fall Hold Time (x1 Mode)	20		40		ns	[1,3]
5	TsSy(RxC)	/DCD as /SYNC to /RxC Rise Setup Time	0		0		ns	[1]
6	ThSy(RxC)	/DCD as /SYNC to /RxC Rise Hold Time (x1 Mode)	20		40		ns	[1]
7	TdTxCr(TxD)	/TxC Fall to TxD Delay		35		50	ns	[2]
8	TdTxCr(TxD)	/TxC Rise to TxD Delay		35		50	ns	[2,3]
9	TwRxCh	/RxC High Width	20		40		ns	
10	TwRxCl	/RxC Low Width	20		40		ns	
11	TcRxC	/RxC Cycle Time	50		100		ns	
12	TwTxCh	/TxC High Width	20		40		ns	
13	TwTxCl	/TxC Low Width	20		40		ns	
14	TcTxC	/TxC Cycle Time	50		100		ns	
15	TwExT	/DCD or /CTS Pulse Width	35		70		ns	
16	TWSY	/DCD as /SYNC Input Pulse Width	35		70		ns	

#### Notes:

- [1] /RxC is /RxC or /TxC, whichever is supplying the receive clock.  
 [2] /TxC is /TxC or /RxC, whichever is supplying the transmit clock.  
 [3] Parameter applies only to FM encoding/decoding.  
 [4]  $V_{cc} = 5V \pm 5\%$ . AC Timings for 20 Mbps are preliminary.  
 [5]  $V_{cc} = 5V \pm 10\%$

**TIMING DIAGRAMS (Continued)**



**Figure 76. Z16C30 System Timing**

## AC CHARACTERISTICS

### Z16C30 System Timing

No	Symbol	Parameter	[3] 20 Mbps		[4] 10 Mbps		Units	Note
			Min	Max	Min	Max		
1	TdRxC(REQ)	/RxC Rise to /RxREQ Valid Delay		50		100	ns	[2]
2	TdRxC(RxC)	/TxC Rise to /RxC as Receiver Output Valid Delay		50		100	ns	[2]
3	TdRxC(INT)	/RxC Rise to /INT Valid Delay		50		100	ns	[2]
4	TdTxC(REQ)	/TxC Fall to /TxREQ Valid Delay		50		100	ns	[2]
5	TdTxC(TxC)	/RxC Fall to /TxC as transmitter Output Valid Delay		50		100	ns	
6	TdTxC(INT)	/TxC Fall to /INT Valid Delay		50		100	ns	[2]
7	TdEXT(INT)	/CTS, /DCD, /TxREQ, /RxREQ transition to /INT Valid Delay		50		100	ns	

#### Notes:

[1] /RxC is /RxC or /TxC, whichever is supplying the receive clock.

[2] /TxC is /TxC or /RxC, whichever is supplying the transmit clock.

[3]  $V_{cc} = 5V \pm 5\%$ . AC Timings for 20 Mbps are preliminary.

[4]  $V_{cc} = 5V \pm 10\%$





## Z16C32

### IUSC™ INTEGRATED UNIVERSAL SERIAL CONTROLLER

#### FEATURES

- Two full-capacity 20 MHz DMA Channels each with 32-bit addressing and 16-bit data transfers.
- DMA Modes include single buffer, pipelined, array-chained and linked-array chained.
- Ring Buffer feature supports circular queue of buffers in memory.
- Linked Frame Status Transfer feature writes status information for received frames and reads control information for transmit frames to the DMA channel's array or linked list. This significantly simplifies processing frame status and control information.
- Programmable throttling of DMA bus occupancy in burst mode with Bus Occupancy Time limitation.
- 0 to 20 Mbit/sec, full-duplex channel, with two baud rate generators and a digital phase-locked loop for clock recovery.
- 32-byte data FIFOs for receiver and transmitter
- Up to 12.5 MByte/sec (16-bit) data bus bandwidth
- Multiprotocol operation under program control with independent mode selection for receiver and transmitter.
- Async mode with one-to-eight bits/character, 1/16 to two stop bits/character in 1/16 bit increments; 16x, 32x, or 64x oversampling; break detect and generation; odd, even, mark, space or no parity and framing error detection. Supports Nine-bit and MIL-STD-1553B protocols.
- HDLC/SDLC mode with 8-bit address compare; extended address field option; 16- or 32-bit CRC; programmable idle line condition; optional preamble transmission and loop mode. Selectable number of flags between back-to-back frames.
- Byte oriented synchronous mode with one-to-eight bits/character; programmable sync and idle line conditions; optional receive sync stripping; optional preamble transmission; 16- or 32-bit CRC; transmit-to-receive slaving (for X.21).
- External character sync mode for receive
- Transparent Bisync mode with EBCDIC or ASCII character code; automatic CRC handling; programmable idle line condition; optional preamble transmission; automatic recognition of DLE, SYN, SOH, ITX, ETX, ETB, EOT, ENQ and ITB.
- Flexible bus interface for direct connection to most microprocessors; user programmable for 8 or 16 bits wide. Directly supports 680X0 family or 8X86 family bus interfaces.
- Receive and Transmit Time Slot Assigners for ISDN, T1 and E1 (CEPT) applications.
- 8-bit general-purpose port with transition detection
- Low power CMOS
- 68-pin PLCC package
- Electronic Programmer's Manual support tool and software drivers are available.

#### GENERAL DESCRIPTION

The IUSC (Integrated Universal Serial Controller) is a multiprotocol datacommunications device with on-chip dual-channel DMA. The integration of a high-speed serial

communications channel with high-performance DMA facilitates higher data throughput than is likely to be achieved with discrete serial/DMA chip combinations. The

---

## GENERAL DESCRIPTION (Continued)

IUSC's value is more than just reduced chip count and saving board space. The DMA and serial channel inter-communication with each other provides real application benefits. For example, events like the reception of the end of a HDLC frame is internally communicated from the serial controller to the DMA so that each frame can be written into a separate memory buffer. The buffer chaining capabilities, ring buffer support, automated frame status/control blocks, and buffer termination at the end of the frame combine to significantly reduce CPU overhead (Figure 1).

The IUSC is software configurable to satisfy a wide variety of serial communication applications. The 20 Mbit/second data rate and multiple protocol support make it ideal for applications in today's dynamic environment of changing specifications and increasing speed. The many programmable features allow the user to tune the device response to meet system requirements and adapt to future requirements. The IUSC contains a variety of sophisticated internal functions including two baud rate generators, a digital phase-locked loop, character counters, and 32-byte FIFOs for both the receiver and the transmitter.

The on-chip DMA channels allow high speed data transfers for both the receiver and the transmitter. The IUSC supports automatic status and control transfer via DMA and allows initialization of the serial controller under DMA control. Each DMA channel can do a 16-bit transfer in as little as three 50 ns clock cycles and can generate addresses compatible with 32-, 24- or 16-bit memory ranges. The DMA channels operate in any of four modes: single buffer, pipelined, array-chained, or linked-list. The array-chained and linked-list modes provide scatter-read and gather-write capabilities with minimal software intervention. To prevent the DMA from holding bus mastership too long, mastership time may be limited by counting the absolute number of clock cycles, the number of bus transactions, or both.

The CPU bus interface is designed for use with any conventional multiplexed or non-multiplexed bus from manufacturers of CISC and RISC processors including Intel, Motorola, and Zilog. The bus interface is configurable

for 16-bit data, 8-bit data with separate address or 8-bit data without separate address to support multiplexed or non-multiplexed busses.

The IUSC handles asynchronous formats, synchronous bit-oriented formats such as HDLC and synchronous byte-oriented formats (e.g., BISYNC & DDCMP). This device supports virtually any serial data transfer application.

The IUSC can generate and check CRC in any synchronous mode. Complete access to the CRC value allows system software to resend or manipulate the CRC as needed in various applications. The IUSC also provides facilities for modem control signals. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

Interrupts are supported by a daisy-chain hierarchy within the serial channel and between the serial channel and the DMA. Separate interrupt vectors for each type of interrupt within the serial controller and the DMA facilitate fast discrimination of the interrupt source. The IUSC supports Pulsed, Double Pulsed, and Status Interrupt Acknowledge cycles.

Support tools are available to aid the designer in efficiently programming the IUSC. The Technical Manual describes in detail all the features and gives programming sequence hints. The Electronic Programmer's Manual is an MS-DOS, disk-based programming initialization tool that can generate custom sequences. Also, Zilog offers assorted application notes and development boards to assist the designer in hardware and software development.

### Notes:

All Signals with a preceding front slash, '/', are active Low, e.g.:  $\overline{B}/\overline{W}$  (WORD is active Low);  $\overline{B}/\overline{W}$  (BYTE is active Low, only).

Power connections follow conventional descriptions below:

Connection	Circuit	Device
Power	$V_{CC}$	$V_{DD}$
Ground	GND	$V_{SS}$

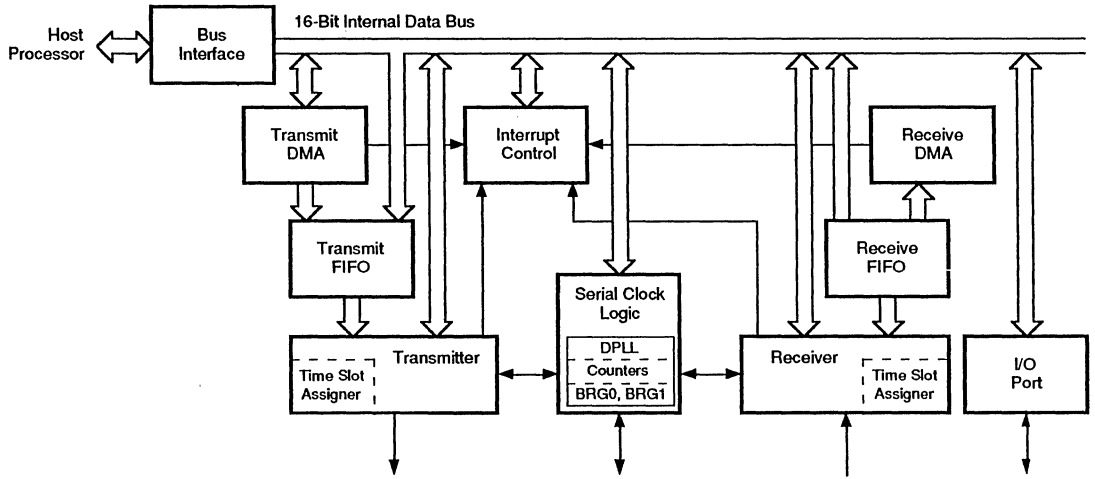


Figure 1. IUSC Block Diagram



GENERAL DESCRIPTION (Continued)

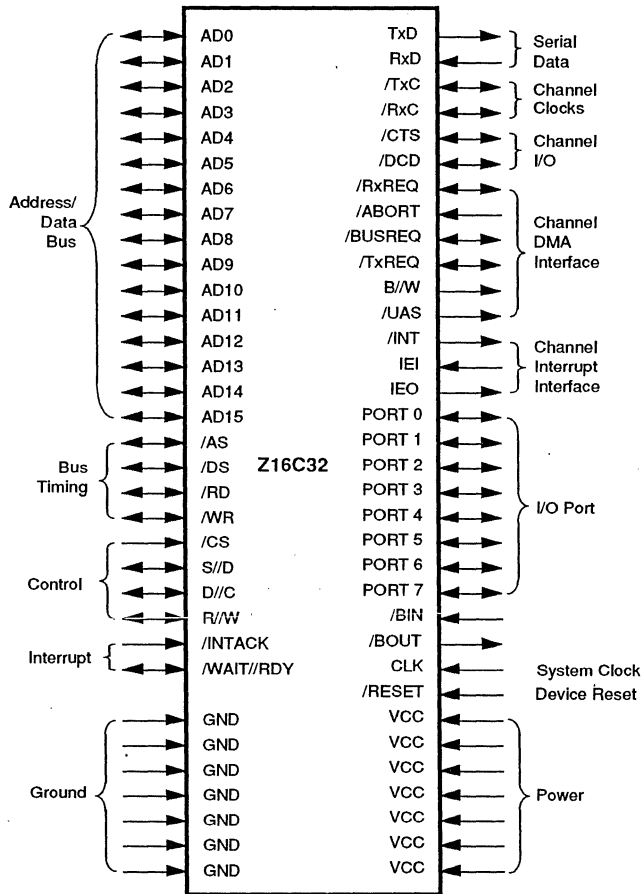


Figure 2. Pin Functions

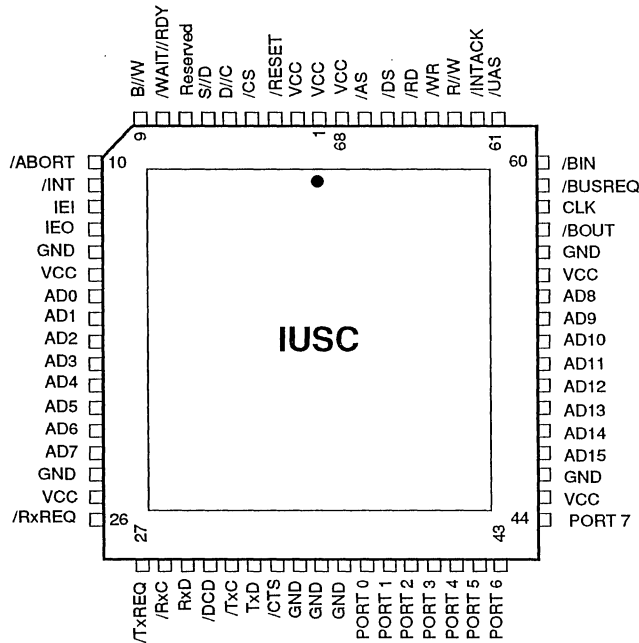


Figure 3. Pin Assignments

## PIN DESCRIPTION

Figure 2 shows the logical pin groupings of the IUSC's pins, and Figure 3 shows the physical pin assignments.

Only one strobe pin (*/DS*, */RD*, */WR* or Pulsed *INTACK*) should ever be active at one time. Any unused input pin (if an input when the IUSC is bus master or slave) must be pulled up to its inactive state.

***/RESET***. *Reset* (input, active Low). A Low on this line places the IUSC in a known, inactive state, and conditions it so that the data, from the next write operation that asserts the */CS* pin, goes into the Bus Configuration Register (BCR) regardless of register addressing. */RESET* should be driven Low as soon as possible during power-up, and as needed when restarting the overall system or the communications subsystem.

**CLK**. *System Clock* (input). This signal is the timing reference for the DMA and bus interface logic. (The serial controller section is clocked by the selected sources of receive and transmit clocking.)

**AD15-0**. *Address/Data Bus* (inputs/tri-state outputs). After Reset, these lines carry data between the controlling microprocessor and the IUSC, and may also carry multiplexed addresses of registers within the IUSC. Such operation, between the host processor and the IUSC, is often called slave mode. Once the software has set up the device and placed it into operation, these lines also carry multiplexed addresses and data between the IUSC and system memory; such operation is called master mode. AD15-0 can be used in a variety of ways based on whether the IUSC senses activity on */AS* after Reset, and on the data written to the Bus Configuration Register (BCR).

***/CS***. *Chip Select* (input, active Low). A Low on this line indicates that the controlling microprocessor's current bus cycle refers to a register in the IUSC. The IUSC ignores */CS* when a Low on */INTACK* indicates that the current bus operation is an interrupt acknowledge cycle. On a multiplexed bus the IUSC latches the state of this pin at rising edges on */AS*; on a non-multiplexed bus, it latches */CS* at leading/falling edges on */DS*, */RD*, or */WR*.

---

## PIN DESCRIPTION (Continued)

**S//D.** *Serial/DMA* (input/tri-state output, input High indicates serial). Cycles with /CS Low, and /INTACK and this pin both High, access registers in the serial controller section. Cycles with /INTACK High, and /CS and this pin both Low, access registers in the DMA controller section. The state of this line when the Bus Configuration Register is written determines wait vs acknowledge operation, as described in the text. On a multiplexed bus, the IUSC latches the state of this pin at rising edges on /AS; on a non-multiplexed bus, it latches the state at leading/falling edges on /DS, /RD, or /WR.

Software can program the IUSC so that when it is acting as a bus master, it drives this line High to indicate a DMA cycle for serial data and Low to indicate an "array" or "list" access. (Array/list accesses read the address and length of the next memory buffer.)

**D//C.** *Data/Control* (input/tri-state output, input High indicates Data). A slave read cycle with /CS Low, and all three of /INTACK, S//D, and this pin High, fetches data from the serial controller's receive FIFO via the Receive Data Register (RDR). A slave write cycle with the same conditions writes data into the transmit FIFO via its Transmit Data Register (TDR). Slave cycles with /INTACK and S//D High, and /CS and this pin Low, read or write registers in the serial controller. On a multiplexed bus, the IUSC determines which register to access from the low-order AD lines at the rising edge of /AS; on a non-multiplexed bus it typically selects the register based on the LSBs of the serial controller's Channel Command/Address Register. On a multiplexed bus, the IUSC latches the state of this pin at rising edges on /AS; on a non-multiplexed bus it latches the state at leading/ falling edges on /DS, /RD, or /WR.

For slave cycles on a multiplexed bus, with /INTACK High and both /CS and S//D Low, the state of this line at the rising edge of /AS selects between the registers of the transmit DMA channel (Low) and those of the receive DMA channel (High). On a non-multiplexed bus, with /INTACK High and /CS and S//D both Low, the IUSC can take the DMA channel selection from this line or from the DMA Command/Address Register.

Software can program the IUSC so that when it is acting as a bus master, it drives this line High to indicate a DMA cycle for the receiver and Low to indicate a cycle for the transmitter.

**/AS.** *Address Strobe* (input/tri-state output, active Low). After a reset, the IUSC's bus interface logic monitors this signal to see if the host bus multiplexes addresses and data on AD15-0. If the logic sees activity on /AS before (or

as) software writes the Bus Configuration Register, then in subsequent slave cycles directed to the IUSC, it captures register selection from the AD lines, S//D, and C//D on rising edges of /AS.

When the IUSC takes control of the bus and operates as a master, it always uses the bus in a multiplexed fashion, driving /AS Low when it places the least significant 16 bits of an address on the AD15-0 lines. External devices can be used to de-multiplex the address and data, if this is necessary to match the characteristics of the host processor or host bus.

For a non-multiplexed bus, this pin should be pulled up to +5V using a resistor of about 10 kOhms. If a processor uses a non-multiplexed bus, yet has an output called Address Strobe (e.g., 680x0 devices), this pin should not be tied to the output.

**/UAS.** *Upper Address Strobe* (tri-state output, active Low). When the IUSC takes control of the bus and operates as a master, it drives /UAS Low when it places the more significant 16 bits of an address on AD15-0. External memory and other slave devices (or de-multiplexing latches) should capture the MS address at each rising edge on this line.

**R//W.** *Read/Write control* (input/tri-state output, Low signifies "write"). R//W and /DS indicate read and write cycles on the bus, for host processors/buses having this kind of signalling. When the IUSC has taken control of the bus and is operating in master mode, this pin is an output that remains valid throughout the Low time of /DS. In slave cycles, the IUSC samples R//W at each leading/falling edge on /DS.

**/DS.** *Data Strobe* (input/tri-state output, active Low). R//W and /DS indicate read and write cycles on the bus, for host processors/buses having this kind of signalling. It is an output when the IUSC has taken control of the bus and is operating in master mode, otherwise, it is an input that is qualified by /CS Low or /INTACK Low. In master mode, the R//W line remains valid throughout the Low time of this line. In slave mode, the IUSC samples R//W at each leading/falling edge on this line. For slave write cycles and master read cycles, the IUSC captures data at the rising (trailing) edge on this line. For slave read cycles the IUSC provides valid data on the AD lines within the specified access time after this line goes Low, and keeps the data valid until after the master releases this line to High. For master write cycles, the IUSC places valid data on the AD lines before it drives this signal to Low, and keeps the data valid until after it drives this line back to High.

---

**/RD. Read Strobe** (input/tri-state output, active Low ). This line indicates a read cycle on the bus, for host processors/buses having this kind of signalling. It is an output when the IUSC has taken control of the bus and is operating in master mode, otherwise, it is an input that is qualified by /CS Low or /INTACK Low. For master read cycles, the IUSC captures data at the rising (trailing) edge of this line. For slave read cycles the IUSC provides valid data on the AD lines within the specified access time after this line goes Low, and keeps the data valid until after the master releases this line to High.

**/WR. Write Strobe** (input/tri-state output, active Low ). This line indicates write cycles on the bus, for host processors/buses having this kind of signalling. It is an output when the IUSC has taken control of the bus and is operating in master mode, otherwise it is an input that is qualified by /CS Low. For slave write cycles, the IUSC captures write data at the rising (trailing) edge of this line. For master write cycles, the IUSC places valid data on the AD lines before it drives this signal to Low, and keeps the data valid until after it drives this line back to High.

**B/W. Byte / Word Select** (tri-state output, High indicates 8-bit transfer). When the IUSC takes control of the bus and operates as a master, a High on this line indicates that a byte is to be transferred, and a Low indicates that 16 bits are to be transferred. The IUSC ignores this signal during slave cycles: it takes the byte/word distinction from an AD line at the rising edge of /AS, or from a bit in the serial or DMA Command/Address Register.

**/WAIT/RDY. Wait, Ready, or Acknowledge handshaking** (input/tri-state output, active Low). This line is an input when the IUSC has taken control of the bus and is operating in master mode. For slave cycles, the IUSC activates this line as an output. In both directions, the line can carry wait or acknowledge signalling depending on the state of the S//D input during the initial BCR write. If S//D is High when the BCR is written, this line operates as a Ready/Wait line for Zilog and most Intel processors. In this mode, the IUSC will not complete a master cycle while this line is Low, and it asserts this line Low until it's ready to complete an interrupt acknowledge cycle; it never asserts this line when the host accesses one of the IUSC registers.

If S//D is Low when the BCR is written, this line operates thereafter as an Acknowledge line for Motorola and some Intel processors. In this mode, the IUSC will not complete a master cycle until this line is Low. It asserts this line Low for register read and write cycles, and when it is ready to complete an interrupt acknowledge cycle.

For slave cycles, this is a full time (totem pole) output. The board designer can combine this signal with similar signals from other slaves, by means of an external logic gate or a tri-state or open-collector driver.

**/INT. Interrupt Request** (output, active Low ). The IUSC drives this line Low when (1) its IEI pin is High, (2) one or more of its interrupt condition(s) is (are) enabled and pending, and (3) the Under Service flag is not set for its highest priority enabled/pending condition, nor for any higher-priority internal condition. Software can program whether the bus interface drives this pin in a totem-pole or an open-drain fashion.

**/INTACK. Interrupt Acknowledge** (input, active Low ). A Low on this line indicates that the host processor is performing an interrupt acknowledge cycle. In some systems, a Low on this line may further indicate that external logic has selected this IUSC as the device to be acknowledged, or as a potential device to be acknowledged. A field in the Bus Configuration Register selects whether this line carries a level-sensitive "status" signal that the IUSC should sample at the leading edge of /AS or /DS, or a single-pulse or double-pulse protocol. The IUSC responds to an interrupt acknowledge cycle in a variety of ways depending on this programming and the state of the /INT and IEI lines, as described in the text.

**IEI. Interrupt Enable In** (input, active High). This signal and the IEO pin can be part of an interrupt-acknowledge daisy chain with other devices that may request interrupts. If IEI is High outside of an interrupt acknowledge cycle, one or more IUSC interrupt condition(s) is(are) enabled and pending, and the Under Service flag isn't set for the highest priority condition nor for any higher-priority one, then the IUSC requests an interrupt by driving its /INT pin Low. If the IEI pin is High during an interrupt acknowledge cycle, one or more IUSC interrupt condition(s) is(are) enabled and pending, and the Under Service flag isn't set for the highest priority condition nor for any higher-priority, then the IUSC keeps IEO Low and responds to the cycle.

**IEO. Interrupt Enable Out** (output, active High). This signal and/or IEI can be part of an interrupt acknowledge daisy chain with other devices that may request interrupts. The IUSC drives its IEO pin Low whenever its IEI pin is Low, and/or if the Under Service flag is set for any condition. This IUSC drives this signal slightly differently during an interrupt acknowledge cycle, in that it also forces IEO Low if it is (has been) requesting an interrupt.

---

## PIN DESCRIPTION (Continued)

**/BUSREQ.** *Bus Request* (output, active Low ). The DMA controller section drives this line Low to request control of the host bus. /BUSREQ can be an open-drain or totem-pole output depending on a bit in the Bus Configuration Register. In open-drain mode the IUSC samples the pin as an input and only drives it Low after sampling it high.

**/BIN.** *Bus Acknowledge In* (input, active Low ). When the IUSC receives a falling edge on this input, it samples whether it has been driving (or has just begun to drive) /BUSREQ. If so, it keeps /BOUT High and takes control of the host bus. If not, it passes the bus grant by driving /BOUT Low. This signal can be used with /BOUT to form a bus-grant daisy chain for arbitration of bus control. Alternatively, it can be connected to a direct, positive grant from an external arbiter, and the /BOUT pin can be left unconnected.

**/BOUT.** *Bus Acknowledge Out* (output, active Low ). As noted above, this signal can be used with /BIN to form a bus-grant daisy chain for arbitration of bus control.

**/ABORT.** *Abort Master Cycle* (input, active Low ). A Low on this line during a master cycle makes the currently active DMA channel terminate its activity and enter a disabled state. Note that /ABORT is only effective during a DMA cycle, so that the IUSC knows which channel should be aborted. Also note that external logic must set /WAIT//RDY to the right state for the cycle to complete, before /ABORT becomes effective.

**RxD.** *Received Data* (input, positive logic). The serial input.

**TxD.** *Transmit Data* (output, positive logic). The serial output.

**/RxC.** *Receive Clock* (input or output). This signal can be used as a clock input for any of the functional blocks in the serial controller. Or, software can program the IUSC so that this pin is an output carrying any of several receiver or internal clock signals, a general-purpose input or output, or an interrupt input.

**/TxC.** *Transmit Clock* (input or output). This signal can be used as a clock input for any of the functional blocks in the serial controller. Or, software can program the IUSC so that this pin is an output carrying any of several transmitter or internal clock signals, a general purpose input or output, or an interrupt input.

**/RxREQ.** *Receive DMA Request* (input or output). In device testing or in applications not using the serial controller and DMA controller sections together in the usual way, this pin can carry a Low-active DMA Request from the receive FIFO. On the IUSC this request is internally routed to the on-chip Receive DMA channel; it is more typical to use the RxREQ pin as a general-purpose output or as an interrupt input.

**/TxREQ.** *Transmit DMA Request* (input or output). In device testing or in applications not using the serial controller and DMA controller sections together in the usual way, this pin can carry a Low-active DMA Request from the transmit FIFO. On the IUSC this request is internally routed to the on-chip Transmit DMA channel, and it's more typical to use the RxREQ pin as a general-purpose output or as an interrupt input.

**/DCD.** *Data Carrier Detect* (input or output, active Low). Software can program the IUSC so that this signal enables/disables the receiver. In addition, software can program the device to request interrupts in response to transitions on this line. The pin can also be used as a simple input or output.

**/CTS.** *Clear to Send* (input or output, active Low). Software can program the IUSC so that this signal enables/disables the transmitter. In addition, software can program the device to request interrupts in response to transitions on this line. The pin can also be used as a simple input or output.

---

**PORT7/TxCOMPLT.** *General-Purpose I/O or Transmit Complete* (input or output). Software can program the IUSC so that this pin is a general purpose input or output, or so that it carries a Transmit Complete signal from the Transmitter, that can control an external driver. The IUSC captures transitions on this pin in internal latches.

**PORT6/FSYNC.** *General-Purpose I/O or Frame Sync* (input or output). Software can program the IUSC so that this pin is a general-purpose input or output, or a Frame Sync input for the IUSC's Time Slot Assigner circuits. The IUSC captures transitions on this pin in internal latches.

**PORT5/RxSYNC.** *General-Purpose I/O or Receive Sync* (input or output). Software can program the IUSC so that this pin is a general-purpose input or output, or so that it carries a Receive Sync output from the Receiver. The IUSC captures transitions on this pin in internal latches.

**PORT4/TxTSA.** *General-Purpose I/O or Transmit Time Slot Assigner Gate* (input or output). Software can program the IUSC so that this pin is a general-purpose input or output, or so that it carries the Gate output of the Transmit Time Slot Assigner, that can enable an external TxD driver in time-slotted ISDN or Fractional T1 applications. The IUSC captures transitions on this pin in internal latches, as described in the text.

**PORT3/RxTSA.** *General Purpose I/O or Receive Time Slot Assigner Gate* (input or output). Software can program the IUSC so that this pin is a general purpose input or output, or so that it carries the Gate output of the Receive Time Slot Assigner. The IUSC captures transitions on this pin in internal latches.

**PORT2.** *General Purpose I/O* (input or output). Software can program the IUSC so that this pin is a general purpose input or output. The IUSC captures transitions on this pin in internal latches.

**PORT1-0/CLK1-0.** *General Purpose I/Os or Reference Clocks* (inputs or outputs). Software can program the IUSC so that either of these pins is a general purpose input or output, or a reference clock that can be divided down to derive clocking for the Receiver and/or Transmitter. When one of these pins is a general-purpose I/O, the IUSC captures transitions on it in internal latches.

**V<sub>cc</sub>, V<sub>ss</sub>.** *Power and Ground.* The inclusion of seven pins for each power rail insures good signal integrity, prevents transients on outputs, and improves noise margins on inputs. The IUSC's internal power distribution network requires that all these pins be connected appropriately.

---

## ARCHITECTURE

The IUSC integrates a fast and efficient dual-channel DMA with a highly versatile serial communications controller. The functional capabilities of the IUSC are described from two different points of view; as a datacommunications device, it transmits and receives data in a wide variety of datacommunications protocols; as a microprocessor peripheral with two DMA channels that offer such features

as four DMA transfer types, a flexible bus interface, and vectored interrupts. The architecture is described in three sections, DMA and Bus Interface Capabilities, Communication between the DMA and Serial Channel, and Serial Communication Capabilities. The structure of the IUSC is shown in Figure 1.

---

## DMA AND BUS INTERFACE CAPABILITIES

The IUSC's two versatile DMA channels combined with a flexible bus interface gives it the ability to meet a wide variety of application requirements. The time required to move data into and out of the transmitter and receiver is minimized by the IUSC's speed (20 MHz clock, three clock cycles per word, typical); two buffer-chaining modes with linked-frame status transfer; early buffer termination to keep received frames in separate memory buffers; and vectored interrupts. Some of the many features are briefly described below. See the IUSC Technical Manual for details.

### DMA Modes

The IUSC contains two DMA channels, one for the transmitter and one for the receiver. Each channel supports a 32-bit address and a 16-bit byte count. The channels operate in one of four modes. In normal mode, the processor must reload the address and length at the end of each buffer. In Pipelined mode, the processor can load the address and length of the next buffer at any time during the DMA transfer to the first buffer. In Array-Chained mode the processor creates a table of address/length pairs in memory for automatic transfer by the channel. In Linked List mode the processor creates a linked list of address and length pairs in memory to be automatically transferred by the channel.

Single Buffer Mode is the most basic of the four data transfer types. The starting address of each memory buffer and the maximum number of characters to be transferred to or from memory are programmed into IUSC registers. When the DMA is enabled, it transfers all data between system memory and the transmit and receive FIFOs.

Pipelined mode is similar to single buffer mode with the addition of an extra set of registers into which the processor can load to reload the DMA with the address and count of the next memory buffer. Therefore, when a buffer is complete, the IUSC is pre-programmed with the address and count of the next buffer so the DMA need not stop between each buffer as long as software stays one step ahead of memory buffer usage.

In array mode, one of the two chaining modes, software sets up a table of memory buffer information. The length of the array is only limited by the amount of system memory available for buffers. The IUSC is programmed with the location of the array of buffer addresses and sizes. This mode has the advantage that a burst of short frames is less likely to overrun the systems ability to keep up. The use of

receive status block and transmit control block along with the early buffer termination feature simplifies the segmentation and re-assembly of serial messages in memory buffers. When a DMA channel fetches a buffer count of zero, it stops and can create an End-Of-Array interrupt.

Linked List mode is the most versatile of DMA modes. It has array mode's ability to switch buffers rapidly without the requirement for the buffer information to be in a continuous table. Each link entry contains: The starting address to write or read the data; the size of the buffer; optional status or control information; and a pointer to the next link. Memory buffers can easily be added and removed from the list by changing the links in list entries.

### DMA Features

In linked list mode, the IUSC has a programmable feature to facilitate the use of buffers in a ring. When this feature is enabled, the DMA writes a zero back to the buffer length field of each array or list entry after it is read. Therefore, if a linked list wraps around on itself, a DMA channel will not reuse a buffer until software has processed the buffer, and indicated that its eligible for reuse by writing a non-zero value in the count field (fetching a count value of zero deactivates the DMA channel). This feature can also be used in array mode to track buffer use.

In both bus slave and master modes, the IUSC can read and write data words in either byte order. It supports the Little Endian convention used by many Intel microprocessors and the Big Endian convention used by many Motorola microprocessors. When the IUSC is bus master, it can be programmed to only generate the upper 16-bit address when required and, consequently, save a clock cycle on each transfer (three clocks per transfer instead of four). When using the IUSC on a 16-bit bus and the starting address of the message is on an odd address, the IUSC automatically re-orientes itself onto even word boundaries by first fetching a byte. This is especially valuable when retransmitting a frame with a different size header than was received. Two pins are available as status signals of the type of transfer in progress.

There are a variety of command and status registers to control and monitor the DMA channels. A DMA channel can be aborted with either the /ABORT pin or by software command. A pause command is also available to temporarily suspend transfers.

---

## Bus Interface & Utilization

The bus interface module stands between the external bus pins and an on-chip 16-bit data bus that interconnects the other functional modules. It includes several flexible bus interfacing options that are controlled by the contents of the Bus Configuration Register (BCR). The BCR is automatically the destination of the first write to the IUSC by the host processor after a reset.

The IUSC is compatible with both multiplexed and non-multiplexed bus interfaces and can transfer either 8 or 16 bits. It supports data transfers with /RD & /WR or R/W & /DS strobe pins and either format of byte ordering. The IUSC generates the Wait or Ready acknowledge handshaking used by Intel or Motorola microprocessors. Also, three styles of interrupt acknowledge signals are supported for automated return of an interrupt vector to any common microprocessor.

There are several options that control how the IUSC uses the bus. The /BIN and /BOUT pins are available to form a bus-grant daisy chain. The IUSC has several options on how it arbitrates requests for bus mastership between channels and how long it stays off the bus between requests. The priority of the two DMA channels is programmable and can alternate between requests to allow both channels equal access to the bus. Once one of the channels has mastership of the bus, control can be passed to the other channel if it is requesting or the IUSC can be forced off the bus. A programmable preempt feature selects whether the higher priority channel can take over control of the bus if it starts requesting control while the lower priority channel is using the bus.

The IUSC maximizes the use of its 32-byte FIFOs by holding /BUSREQ active until the transmit FIFO is full, the receive FIFO is empty, or both. The programmable dwell timers can be used to limit how long the IUSC holds bus mastership by counting either bus transfers, clock cycles or both. Therefore, the combination of programmable FIFO request levels, channel arbitration options, and programmable dwell timer features provide application software the flexibility to optimize the IUSC's bus occupancy to meet system throughput and bus response requirements.

### Interrupts

The interrupt subsystem of the IUSC derives from Zilog's experience in providing the most advanced interrupt capabilities in the microprocessor field. These capabilities are at their best when used with a Zilog microprocessor,

but it is easy to interface the IUSC to work well with other microprocessors as well. Four pins are dedicated to create an interrupt daisy-chain hierarchy within the Serial Channel and between the Serial Channel and the DMA.

When an IUSC responds to an interrupt acknowledge from the CPU, it places an interrupt vector on the data bus. To speed interrupt response time, the IUSC modifies three bits in the vector to indicate which type of interrupt is being requested. Separate vectors are provided for the serial channel and DMA to easily discriminate the interrupt source.

The DMA has four interrupt sources each for the receive and transmit channels. Each interrupt source is independently enabled and there is a master enable for all DMA interrupts. The four interrupt sources are End Of Array/ End of Link, End Of Buffer, Hardware Abort, and Software Abort.

Each of the six types of interrupts in the serial portion IUSC (Receive Status, Receive Data, Transmit Status, Transmit Data, I/O Status and Device Status) has three bits associated with it: Interrupt Pending (IP), Interrupt-Under-Service (IUS) and Interrupt Enable (IE). If the IE bit for a given source is set, then that bit can source request interrupts. Note that individual sources within the six types also have their own interrupt arm bits. Finally, there is a Master Interrupt Enable (MIE) bit which globally enables or disables all interrupts from the serial channel.

The Interrupt (/INT), Interrupt Acknowledge (/INTACK), Interrupt Enable In (IEI) and Interrupt Enable Out (IEO) pins are provided to create an automated mechanism to place the vector on the bus among the highest priority pending interrupts from multiple devices. The device with the highest pending interrupt (/INT Low, IEI High) places a vector on the bus in response to an interrupt acknowledge cycle.

In the IUSC, the IP bit signals that an interrupt is pending. If an IUS bit is set, this interrupt is being serviced and all interrupt sources of lower priority are prevented from requesting interrupts. An IUS bit is set during an interrupt acknowledge cycle if there are no higher priority devices requesting interrupts.



---

## DMA AND BUS INTERFACE CAPABILITIES (Continued)

There are six sources of Receive Status interrupt. Each one is individually armed: Receiver exited hunt, received idle line, received break/abort, received code violation/end-of-transmission/end-of-message, parity error/abort and overrun error. The Receive Data interrupt is generated whenever the receive FIFO fills with data beyond the level programmed in the Receive Interrupt Control Register (RICR). There are six sources of Transmit Status interrupt. Each one is individually armed: Preamble sent, idle line sent, abort sent, end-of-frame/end-of-message sent, CRC sent and underrun error. The Transmit Data interrupt is generated whenever the transmit FIFO empties below the level programmed in the Transmit Interrupt Control Register (TICR).

The I/O Status interrupt serves to report transitions on any of six pins. Interrupts are generated on either or both edges with individual edge selection and arming for each pin. The pins that can be programmed to generate I/O Status interrupts are /RxC, /TxC, /RxREQ, /TxREQ, /DCD and /CTS. These interrupts are independent of the programmed function of the pins.

The Device Status interrupt has four individually enabled sources: Receive character counter underflow, DPPLL sync acquired, BRG1 zero count and BRGO zero count. Refer to Chapter 6 of the Technical Manual for more detail.

---

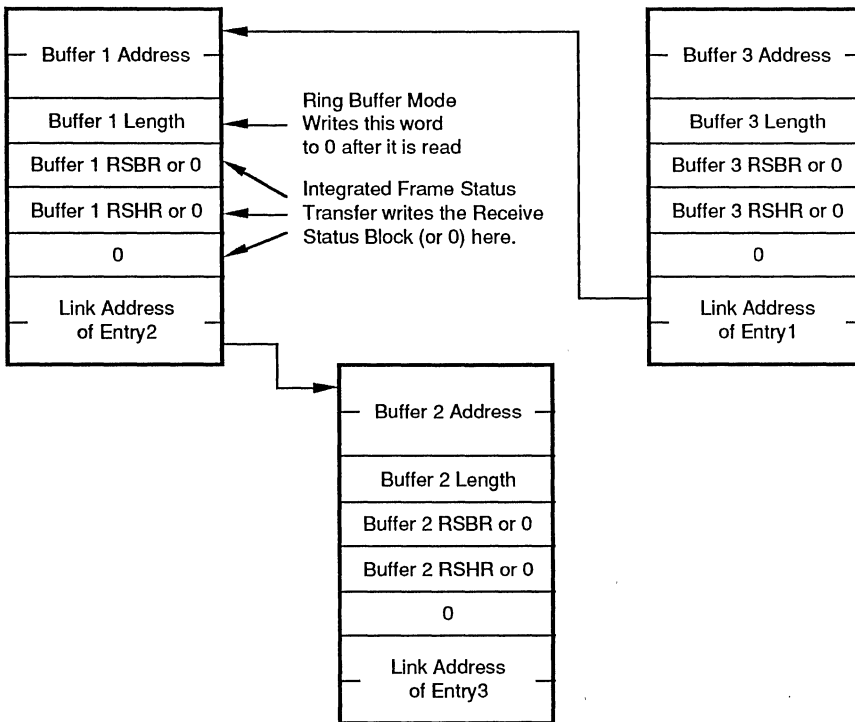
## COMMUNICATION BETWEEN THE DMA AND SERIAL CHANNELS

The IUSC's intra-chip communication between the DMA and serial communications controller give it the power to achieve higher efficiency than is possible with a separate DMA controller. The Linked Frame Status Transfer feature writes the status and byte count of each received frame to memory as part of an array or linked list. This provides a simple and easy to use mechanism for storing the results of a received message without arbitrary restrictions on how quickly the host software must examine the results. Similarly, control information for transmit frames can be automatically read by the DMA from the array or link and transferred into registers in the serial logic.

In all modes, the DMA can accept a signal from the serial channel for early buffer termination. When the end of a message is received, the data is transferred to the buffer and the status is written to memory. The status is written after the data in single buffer and pipelined modes or to the array/link in array and linked-list modes if Linked-Frame Status Transfer is enabled. This early buffer termination is

treated identically to the terminal count condition in the DMA. Therefore, the receipt of the end of a message is a seamless transition from one memory buffer to the next.

An example of using these inter-communication features using linked-list mode is shown in Figure 4. This example shows the format of a ring of memory buffers with the linked frame status transfer and ring buffer features enabled. Any protocol that sets the "RxBound" bit (RCSR4=1), like HDLC or 802.3, is appropriate to this example. The linked list is shown in Figure 4 with three links for simplicity and may be as large as memory allows. The sixth word in each list entry is reserved and should not be used (it keeps the list entries on 32-bit boundaries). If the end of the buffer is reached and it is not the end of the frame, the IUSC writes zeros as the status and count. Also, if the transmit channel needs to start a new memory buffer other than at the beginning of a frame, the DMA ignores the transmit control block.



**Figure 4. Linked-List Mode with Linked-Frame Status Transfer & Ring Buffer Features**

Another method by which the DMA and serial channel work together is using the Transmit Character Counter to break a large block of data into a number of fixed length frames. For example, it is desired to transmit a large file which is located in several memory buffers as fixed length smaller frames. With the IUSC, the serial channel is programmed to send the end-of-frame sequence each time the set number of bytes is transmitted. Therefore, DMA transfers are not interrupted, nor is system response required to break the large file into frames.

The IUSC provides higher throughput than discrete serial and DMA chip solutions because discrete chips do not directly communicate with each other and, therefore, the status of one device must be read by the CPU and communicated to the other. This typically requires interrupts and the suspension of activity until status/control information is updated. This uses precious time and bus bandwidth, which can limit total throughput.

---

## DATA COMMUNICATIONS CAPABILITIES

The IUSC provides a full-duplex channel programmable for use in any common data communication protocol. The receiver and transmitter are completely independent and each is supported by a 32-byte deep FIFO and a 16-bit frame length counter. All modes allow optional even, odd, mark or space parity. Synchronous modes allow the choice of either of two 16-bit or a 32-bit CRC polynomials. Character length of up to 8 bits can be programmed for the receiver and transmitter independently. Error and status conditions are carried with the data in the receive FIFO to greatly reduce the CPU overhead required to send or receive a message, while key control parameters accompany transmit characters through the Tx FIFO. Interrupts can be individually armed to signal such conditions as overrun, parity error, framing error, end-of-frame, idle line received, sync acquired, transmit underrun, CRC sent, closing sync/flag sent, abort sent, idle line sent and preamble sent. In addition, several useful internal signals like receive character boundary, received sync, transmit character boundary and transmission complete can be sent to pins for use by external circuitry.

### Protocols

**Asynchronous Mode.** The receiver and transmitter handle data at a rate of 1/16, 1/32, or 1/64 the clock rate. The receiver rejects start bits less than one-half a bit time and includes recovery logic following a framing error. The transmitter is capable of sending one, two, or anywhere in the range of 9/16th to two stop bits per character in 1/16 bit increments.

**Nine-Bit Mode.** This mode is identical to async except that the receiver checks for the status of an additional address/data bit between the parity bit and the stop bit. The value of this bit is FIFO'ed along with the data. In the transmitter, this bit is automatically inserted with the value that is FIFO'ed from the transmit data.

**Isochronous Mode.** Both transmitter and receiver operate on start-stop (async) data using a 1x clock. The transmitter sends one or two stop bits.

**Asynchronous With Code Violations.** This is similar to Isochronous mode except that the start bit is replaced by a three bit-time code violation pattern as in MIL-STD-1553B. The transmitter sends zero, one or two stop bits.

**HDLC Mode.** In this mode, the receiver recognizes flags, performs optional address matching, accommodates extended address fields, and performs zero deletion and CRC checking. The receiver is capable of receiving shared-zero flags, recognizes abort sequences and can receive

arbitrary length frames. The transmitter automatically sends opening and closing flags, performs zero insertion and can be programmed to send an abort, an extended abort, a flag or CRC and a flag on transmit underrun. The transmitter automatically sends a closing flag with optional CRC at the end of a programmed message length. Shared-zero flags are selected in the transmitter and a separate character length is programmed for the last character in the frame.

Frames terminated with an ABORT can be marked with a status bit on the preceding character in addition to the status interrupt that can be enabled. Abort is only detected in-frame and, therefore, eliminates false detection due to an idle line. The IUSC provides four choices (flag, all 1s, all 0s, or alternating 1s & 0s) of line preamble to condition the line before beginning data transmission. This feature is valuable to get the receiver DPLL in sync and as a flow control mechanism to slow down frame transmission without slowing down the clock or disabling the transmitter.

**HDLC Loop Mode.** This mode is available only in the transmitter and allows the IUSC to be used in an HDLC Loop configuration. In this mode, the receiver is programmed to operate in HDLC mode to allow the transmitter to echo received messages. Upon receipt of a particular bit pattern (actually a sequence of seven consecutive ones) the transmitter stops repeating data and inserts its own frame(s).

**802.3 Mode.** This mode implements the data format of IEEE 802.3 with a 16-bit address compare. In this mode, /DCD and /CTS are used to implement the carrier sense and collision detect interactions with the receiver and transmitter. Back-off timing must be provided externally.

**Monosync Mode.** In this mode, a single character is used for synchronization. The sync character can be either eight bits long or the same length as the data characters. The receiver can automatically strip sync characters from the received data stream. The transmitter is programmed to automatically send CRC on either an underrun or at the end of a programmed message length.

**Slaved Monosync Mode.** This mode is available only in the transmitter and allows the transmitter (operating just as though it were in monosync mode) to send data with its byte boundaries synchronized to those of the received data.

**Bisync Mode.** This mode is identical to monosync mode except that character synchronization requires two successive characters. The two characters need not be identical.

**Transparent Bisync Mode.** In this mode, the synchronization pattern is DLE-SYN, programmable selected from either ASCII or EBCDIC encoding. The receiver recognizes control character sequences and automatically handles CRC calculations without CPU intervention. The transmitter is programmed to send either SYN, DLE-SYN, CRC-SYN, or CRC-DLE-SYN upon underrun and automatically sends the closing DLE-SYN with optional CRC at the end of a programmed message length.

**External Sync Mode.** The receiver is synchronized to the receive data by an externally-supplied signal on a pin for custom protocol applications.

## Data Encoding

The IUSC is programmed to encode and decode the serial data in any of eight different ways (Figure 5). The transmitter encoding method is selected independently of the receiver decoding method.

**NRZ.** In NRZ, a 1 is represented by a High level for the duration of the bit cell and a 0 is represented by a Low level for the duration of the bit cell.

**NRZB.** NRZB is inverted from NRZ.

**NRZI-Mark.** In NRZI-Mark, a 1 is represented by a transition at the beginning of a bit cell, i.e., the level present in the preceding bit cell is reversed. A 0 is represented by the absence of a transition at the beginning of the bit cell.

**NRZI-Space.** In NRZI-Space, a 1 is represented by the absence of a transition at the beginning of a bit cell, i.e., the level present in the preceding bit cell is maintained. A 0 is represented by a transition at the beginning of the bit cell.

**Biphase-Mark.** In Biphase-Mark, a 1 is represented by a transition at the beginning of the bit cell and another transition at the center of the bit cell. A 0 is represented by a transition at the beginning of the bit cell only.

**Biphase-Space.** In Biphase-Space, a 1 is represented by a transition at the beginning of the bit cell only. A 0 is represented by a transition at the beginning of the bit cell and another transition at the center of the bit cell.

**Biphase-Level.** In Biphase-Level, a 1 is represented by a High during the first half of the bit cell and a Low during the second half of the bit cell. A 0 is represented by a Low during the first half of the bit cell and a High during the second half of the bit cell.

**Differential Biphase-Level.** In Differential Biphase-Level, a 1 is represented by a transition at the center of the bit cell, with the opposite polarity from the transition at the center of the preceding bit cell. A 0 is represented by a transition at the center of the bit cell with the same polarity as the transition at the center of the preceding bit cell. In both cases, there are transitions at the beginning of the bit cell to set up the level required to make the correct center transition.

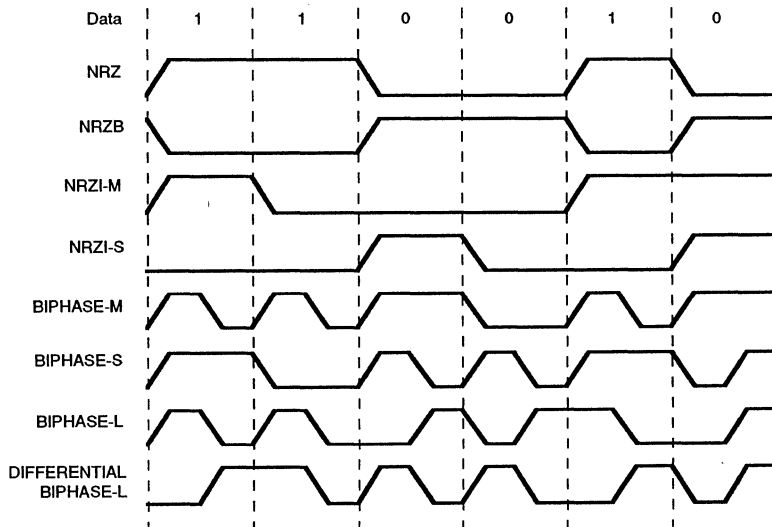


Figure 5. Data Encoding

---

## DATA COMMUNICATIONS CAPABILITIES (Continued)

### Character Counters

The IUSC contains separate 16-bit character counters for the receiver and transmitter. The receive character counter is set to a programmable starting value or automatically at the beginning of each received frame and can be reloaded under software control during a frame. The counter decrements with each receive character. At the end of the receive message the current value in the counter is automatically loaded into a four-deep FIFO. With the Receive Status Block (RSB) feature enabled, the counter value and the status (RCSR) can be automatically transferred to memory following the data. In array and lined list modes, the RSB can be transferred to the array or list entry for easy software access. This allows DMA transfer of data to proceed without CPU intervention at the end of a received frame, as the values in the FIFO allow the CPU to determine the status and length of each frame.

Similarly, the transmit character counter is loaded automatically at the beginning of each transmit frame and can be reloaded under software control during a frame. The counter is with each write to the transmit FIFO. When the counter reaches zero, and that byte is sent, the transmitter automatically terminates the message in the appropriate fashion (usually by sending the CRC and the closing flag or sync character) without requiring CPU intervention. In linked list and array modes, the transmit character count and frame control word can be fetched from the linked list or array.

### Baud Rate Generators

The IUSC contains two baud rate generators. Each generator consists of a 16-bit time constant register and a 16-bit down counter. In operation, the counter decrements with each cycle of its selected input clock, and the time constant can be automatically reloaded when the count reaches zero. The output of the baud rate generator toggles when the counter reaches a count of one-half of the time constant and again when the counter reaches zero. A new time constant can be written at any time but the new value does not take effect until the next load of the counter. The outputs of both baud rate generators are sent to the clock multiplexer for use internally or externally. The input

to the baud rate generator can be the /TxC pin, the /RxC pin, a PORT pin, or the output of either counter. The Baud Rate Generator output frequency is related to the baud rate generator input clock frequency by the following formula:

$$\text{Output frequency} = \text{Input frequency} / (\text{time constant} + 1).$$

**Note:** This allows an output frequency in the range of 1 to 1/65536 of the input frequency, inclusive.

The output of either BRG (Baud Rate Generator) can be used as the transmit or receive clock, the reference clock input to the DPLL Circuit, and/or can be output on the /RxC or /TxC pin.

### Digital Phase-Locked Loop

The IUSC contains a DPLL (Digital Phase-Locked Loop) to recover clock information from a data stream with NRZI or Biphase encoding. The DPLL is driven by a clock that is nominally 8, 16 or 32 times the receive data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock can be routed to the receiver, transmitter, or both, or to a pin for use externally. In all modes, the DPLL counts the input clock to create nominal bit times. While counting, the DPLL watches the incoming data stream for transitions. When a transition is detected, the DPLL may make a count adjustment (during the next counting cycle) to produce an output clock which tracks the incoming bit cells. The DPLL provides properly phased transmit and receive clocks to the clock multiplexer.

### Counters

The IUSC contains two 5-bit counters, which are programmed to divide an input clock by 4, 8, 16 or 32. The outputs of these two counters are sent to the clock multiplexer. The counters can be used as prescalers for the baud rate generators. They also provide a stable transmit clock from a common source when the DPLL is providing the receive clock. The PORT0 and PORT1 pins can be used as inputs to the counters.

---

## Clock Multiplexers

The clock multiplexer logic selects the receive and transmit clocks and optional outputs on the /RxC and/or /TxC pin(s). In the Z16C32, the PORT0 and PORT1 pins can be used directly as receive and transmit clocks, as well as being used as inputs to the counters.

## Time Slot Assigner

The IUSC is equipped with two Time Slot Assigners to support ISDN and Fractional T1 communications. There is one assigner for the receiver. Each time slot assigner selects one or more time slots within a frame, however, the selected time slots must be contiguous. The first selected time slot is programmable from slot 0 (the first slot) to slot 127 of the frame. The number of concatenated slots is programmable from 1 to 15 (total slots). The time of the first slot can be offset an integral number of clocks. This offset is a delay and is programmable from 0 (no offset) to 7 clocks in increments of one clock (one bit cell). This offset can be used to compensate for delays in frame sync detection logic.

## Test Modes

The IUSC can be programmed for local loopback or auto echo operation. In local loopback, the output of the transmitter is internally routed to the input of the receiver. This allows testing of the IUSC data paths without any external logic. Auto echo connects the RxD pin directly to the TxD pin. This is useful for testing serial links external to the IUSC.

## I/O Port

The Port pins are general-purpose I/O pins. They are used as additional modem control lines or other I/O functions. Each port bit is individually programmable as general-purpose input, as an output, or for a dedicated input or output function. This programming is done in the Port Control Register. Whether used as inputs or outputs, the port pins can be read at any time.

The dedicated functions of the port pins include Time Slot Assigner gate outputs, transmit complete output, clock inputs, receive sync output, or frame sync input.

The port pins capture edge transitions. Programming for the capture is done using the Port Latched/Unlatch command bits in the Port Status Register. Each port bit is individually controlled. The Latched/Unlatch bit is used as a status signal to indicate that a transition has occurred on the port pin and as a command to open the latches that capture this transition. Both rising edge and falling edge are detected. When a transition is detected, the latch closes holding the post transition state of the input.

The Latched/Unlatch bit is held at 0 if no transitions occur on the port pin; this bit is set to a 1 when a rising edge or falling edge transition is detected, or immediately after the latch is opened if one or more transitions occurred while the latch was closed. Writing a 0 to the Latched/Unlatch bit has no effect on the latch. Writing a 1 to this bit resets the status bit and opens the latch. To use the port as an input without edge detection, a 1 would be written to the Latched/Unlatch bit to open the latch and then the Port Status Register would be read to obtain the current pin input status.

---

## PROGRAMMING

An Electronic Programmer's Manual (MS DOS based) and a Technical Manual are available to provide details about programming the IUSC. Also included are explanations and features of all registers in the IUSC.

The registers in the IUSC are programmed by the system to configure the channel. Before this can occur, the system must set up the bus interface by writing to the Bus Configuration Register (BCR). The BCR has no specific address and is only accessible after a hardware reset of the device. The first write to the IUSC, after a hardware reset, programs the BCR. From that time on other channel registers can be accessed. No specific address need be presented to the IUSC for the BCR write; the IUSC knows that the first write after a hardware reset is destined for the BCR.

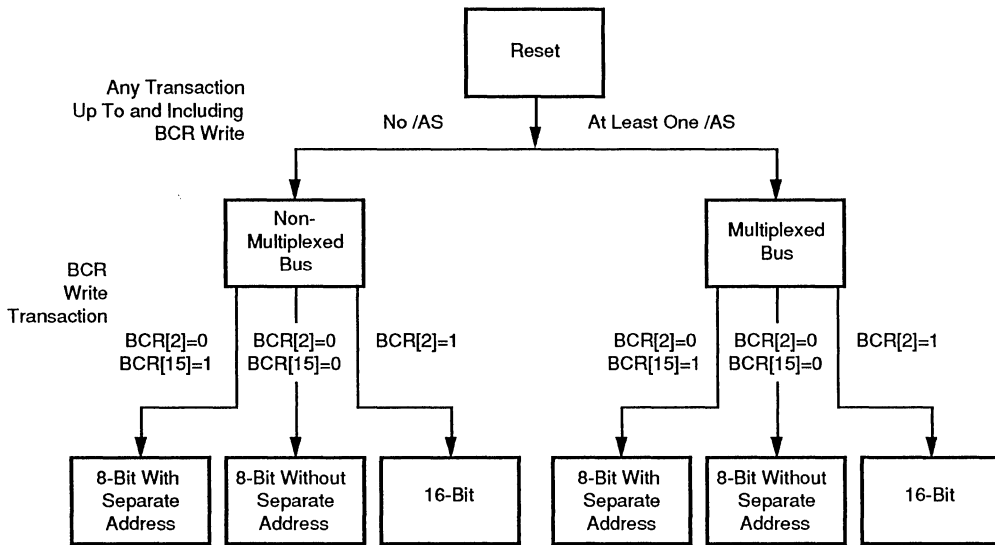
In the multiplexed bus case, all registers are directly addressable via the address latched by /AS at the beginning of each bus cycle. The D//C pin is still used to directly access the receive and send data registers (RDR and TDR) with a multiplexed bus; if D//C is High, the address latched by /AS is ignored and an access of RDR or TDR is performed.

In the non-multiplexed bus case, the channel registers are accessed indirectly using the address pointer in the Channel Command/Address Register (CCAR). The address of the desired register is first written to the CCAR and then the selected register is accessed; the pointer in the CCAR is automatically cleared after this access.

Two more points about the IUSC should be noted here. Channel Reset bit in the CCAR places the channel in the reset state. To exit this reset state either a word of all zeros is written to the CCAR (16-bit bus) or a byte of all zeros is written to the lower byte of the CCAR (8-bit bus). Secondly, after reset, the transmit and receive clocks are disabled. The first thing that should be done in any initialization sequence is a write to the Clock Mode Control Register (CMCR) to select a clock source for the receiver and transmitter.

The Serial/DMA (S//D) pin is used to differentiate between the serial channel and the DMA registers. The DMA registers fall into three logic groupings; common registers that apply to both transmit and receive, transmit registers, and receive registers. The registers for DMA transmit functions and receive functions are symmetric and therefore, a single diagram is shown for each in the following pages. When addressing the DMA registers, the Data/Control (D//C) pin selects between the transmit and receive registers. For example, there is a DMA byte count register for transmit and receive (TBCR and RBCR) at address 10101 with S//D pin Low. The TBCR is selected with the D//C pin Low, and the RBCR is selected with the D//C pin High. The format of these two registers is shown in Figure 20.

The register addressing is shown in Table 2 and the table assumes that the BCR register bit 0 is set to 1. The A5-1 column in the Table reflects the state of AD5-1, AD13-9, CCAR5-1 or DCAR5-1 as applicable. The bit assignments of the registers are shown in Figures 7 through 80. See the IUSC Technical Manual for details. The register addressing is shown in Table 2 and the bit assignments for the registers are shown in Figure 6.



Note:  
The presence of one transaction with an /AS active, between reset up to and including the BCR write, chooses a multiplexed type of bus.

Figure 6. BCR Reset Sequence and Bit Assignments



**PROGRAMMING** (Continued)

**Table 1. Register Address List**

S/D	D/C	Address A5-A1		
1	0	00000	CCAR	Channel Command/Address Register
1	0	00001	CMR	Channel Mode Register
1	0	00010	CCSR	Channel Command/Status Register
1	0	00011	CCR	Channel Control Register
1	0	00100	PSR	Port Status Register
1	0	00101	PCR	Port Control Register
1	0	00110	TMDR	Test Mode Data Register
1	0	00111	TMCR	Test Mode Control Register
1	0	01000	CMCR	Clock Mode Control Register
1	0	01001	HCR	Hardware Configuration Register
1	0	01010	IVR	Interrupt Vector Register
1	0	01011	IOCR	I/O Control Register
1	0	01100	ICR	Interrupt Control Register
1	0	01101	DCCR	Daisy-Chain Control Register
1	0	01110	MISR	Misc Interrupt Status Register
1	0	01111	SICR	Status Interrupt Control Register
1	1	XXXXX	RDR	Receive Data Register (Read Only)
1	0	1X000	RDR	Receive Data Register (Read Only)
1	0	10001	RMR	Receive Mode Register
1	0	10010	RCSR	Receive Command/Status Register
1	0	10011	RICR	Receive Interrupt Control Register
1	0	10100	RSR	Receive Sync Register
1	0	10101	RCLR	Receive Count Limit Register
1	0	10110	RCCR	Receive Character Count Register
1	0	10111	TC0R	Time Constant 0 Register
1	1	XXXXX	TDR	Transmit Data Register (Write Only)
1	0	1X000	TDR	Transmit Data Register (Write Only)
1	0	11001	TMR	Transmit Mode Register
1	0	11010	TCSR	Transmit Command/Status Register
1	0	11011	TICR	Transmit Interrupt Control Register
1	0	11100	TSR	Transmit Sync Register
1	0	11101	TCLR	Transmit Count Limit Register
1	0	11110	TCCR	Transmit Character Count Register
1	0	11111	TC1R	Time Constant 1 Register

**Table 1. Register Address List (Continued)**

S//D	D//C	Address A5-A1		
X	0	XXXXX	BCR	Bus Configuration Register
0	X	00000	DCAR	DMA Command/Address Register
0	0	00001	TDCMR	Transmit DMA Channel Mode Register
0	X	00011	DCR	DMA Control Register
0	X	00100	DACR	DMA Array Count Register
0	X	01001	BDCR	Burst Dwell Control Register
0	X	01010	DIVR	DMA Interrupt Vector Register
0	X	01100	DICR	DMA Interrupt Control Register
0	X	01101	CDIR	Clear DMA Interrupt Register
0	X	01110	SDIR	Set DMA Interrupt Register
0	0	01111	TDIAR	Transmit DMA Interrupt Arm
0	0	10101	TBCR	Transmit Byte Count Register
0	0	10110	TARL	Transmit Address Register (Lower)
0	0	10111	TARU	Transmit Address Register (Upper)
0	0	11101	NTBCR	Next Transmit Byte Count Register
0	0	11110	NTARL	Next Transmit Address Register (Lower)
0	0	11111	NTARU	Next Transmit Address Register (Upper)
0	1	00001	RDMMR	Receive DMA Mode Register
0	1	01111	RDIAR	Receive DMA Interrupt Arm
0	1	10101	RBCR	Receive DMA Byte Count Register
0	1	10110	RARL	Receive Address Register (Lower)
0	1	10111	RARU	Receive Address Register (Upper)
0	1	11101	NRBCR	Next Receive Byte Count Register
0	1	11110	NRARL	Next Receive Address Register (Lower)
0	1	11111	NRARU	Next Receive Address Register (Upper)

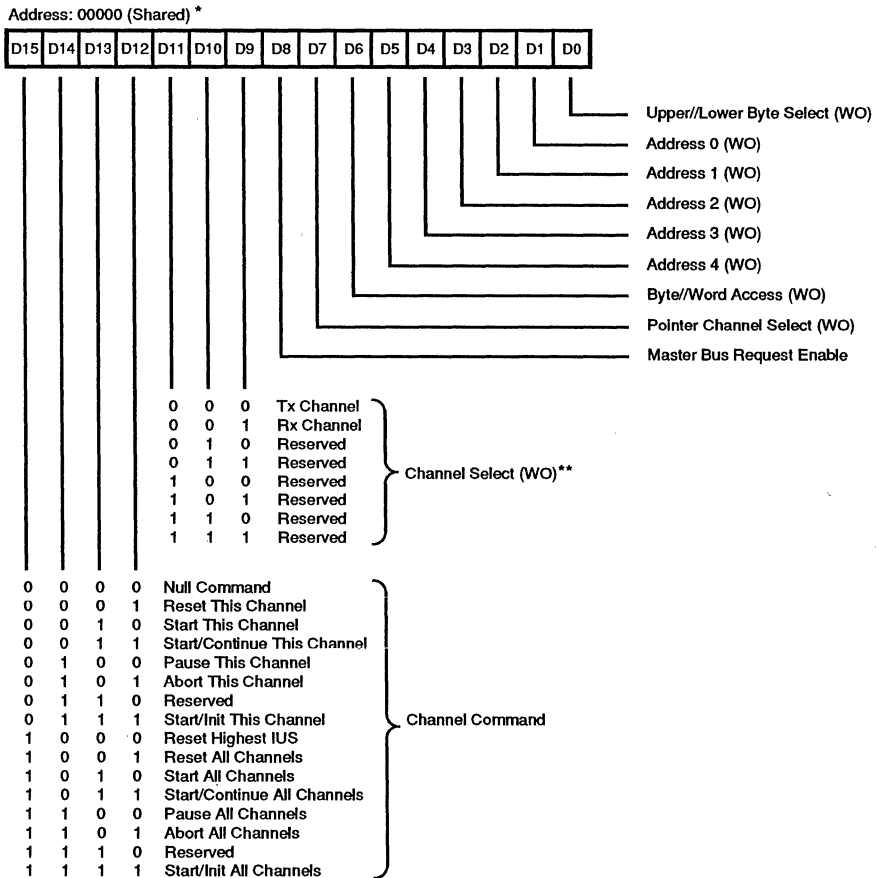
## REGISTER DESCRIPTION

This section describes the function of the various bits in the registers of the device. Throughout this section the following conventions are discussed:

Control bits are written and read by the CPU and are not modified by the device. Command bits are written by the CPU to initiate an action in the device and are read as zeros. Status bits are controlled by the device and are read to check device status. Any writes to status bits are ignored by the device. Command/Status bits are controlled by both the device and the CPU. They may be written and read by the CPU and may also be modified by the device.

Reserved bits are not used in this implementation of the device and may or may not be physically present in the device. Any reserved bits that are physically present are readable and writable, but reserved bits that are not present are always read as zeros. To ensure compatibility with future versions of the device, reserved bits should always be written with zeros. Reserved commands should not be used for the same reason.

First, the DMA registers unique to the IUSC are described in the following pages (Figures 7-16) and then the serial channel registers are described (Figures 17-80).

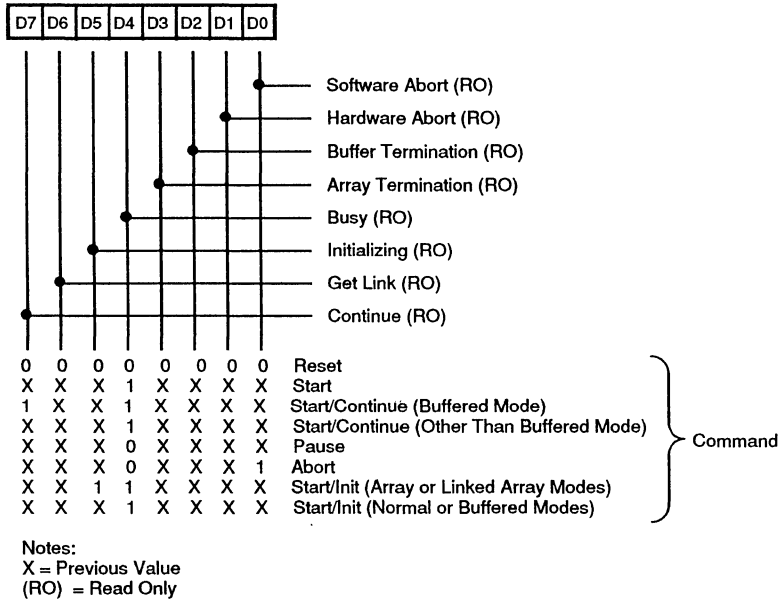


Notes:

\* (Shared) means, shared between DMA Channels

\*\* (WO) means Write Only

Figure 7. DMA Command/Address Register



**Figure 8. Affect of Commands on Status Bits**

## CONTROL REGISTERS (Continued)

Address: 00001

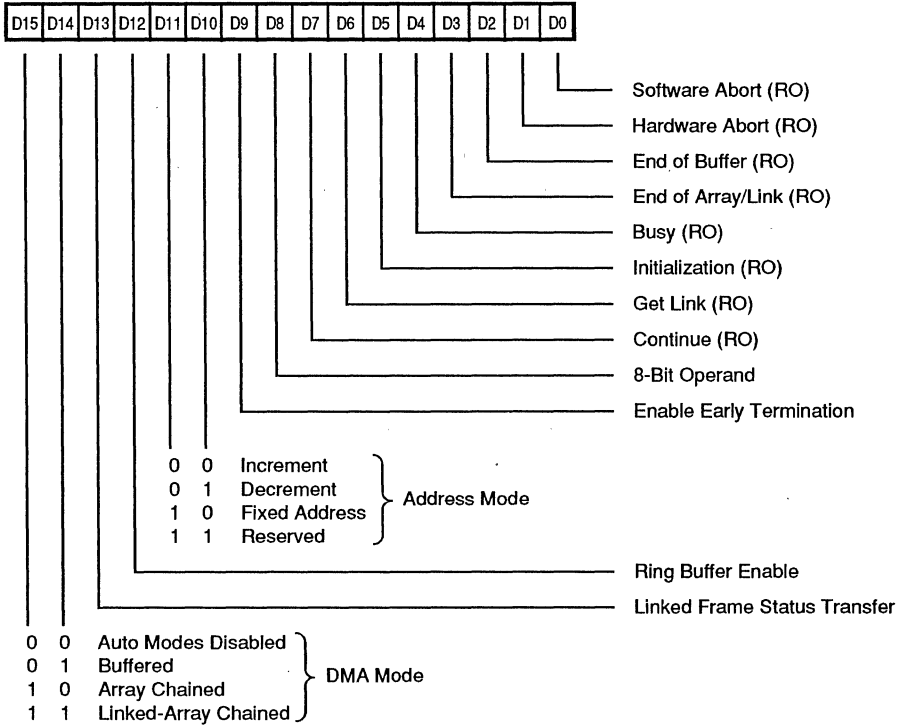


Figure 9. Tx/Rx DMA Mode Register (TDMR) (RDMR)

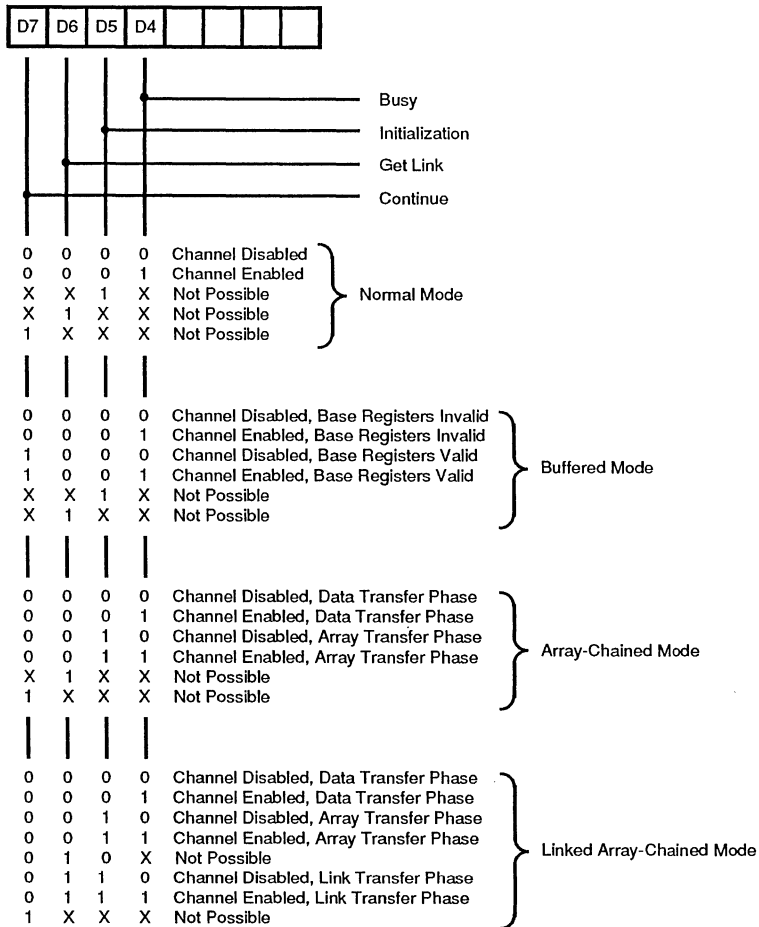


Figure 10. Status Bit Combinations

# CONTROL REGISTERS (Continued)

Address: 00011 (Shared)

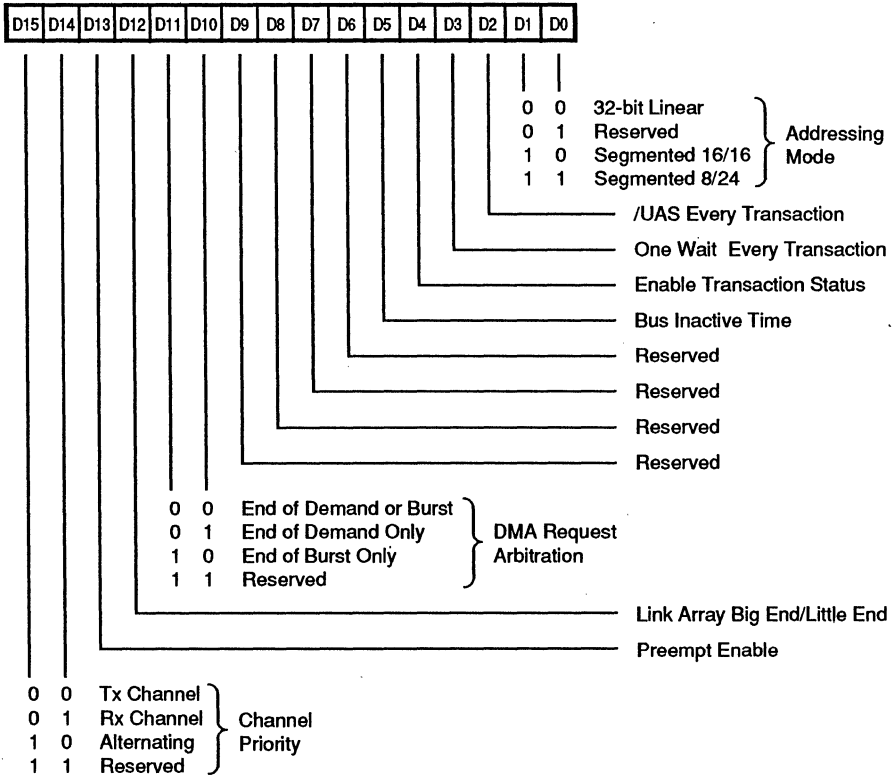


Figure 11. DMA Control Register (DCR)

**Big End Array  
(16-Bit bus)**

	AD15														AD0	
Address n	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Address n+2	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

**Little End Array  
(16-Bit bus)**

	AD15														AD0	
Address n	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Address n+2	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

**Big End Array  
(8-Bit bus)**

	AD7							AD0
Address n	31	30	29	28	27	26	25	24
Address n+1	23	22	21	20	19	18	17	16
Address n+2	15	14	13	12	11	10	09	08
Address n+3	07	06	05	04	03	02	01	00

**Little End Array  
(8-Bit bus)**

	AD7							AD0
Address n	07	06	05	04	03	02	01	00
Address n+1	15	14	13	12	11	10	09	08
Address n+2	23	22	21	20	19	18	17	16
Address n+3	31	30	29	28	27	26	25	24

**Note:**

Bit 12 in DCR is used to control the byte ordering of addresses and counts stored in memory in the array and linked array modes. The above figure shows the two cases for both bus bandwidths.

**Figure 12. Array-Chained Bit Ordering**



## CONTROL REGISTERS (Continued)

Address: 00100 (Shared)

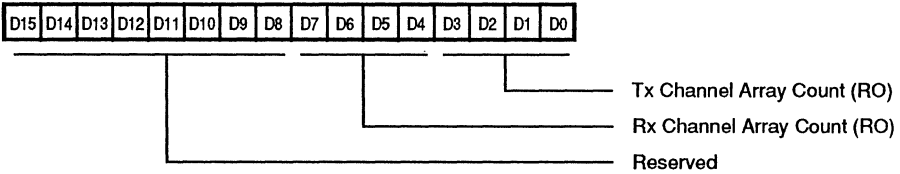


Figure 13a. DMA Array Count Register (DACR)

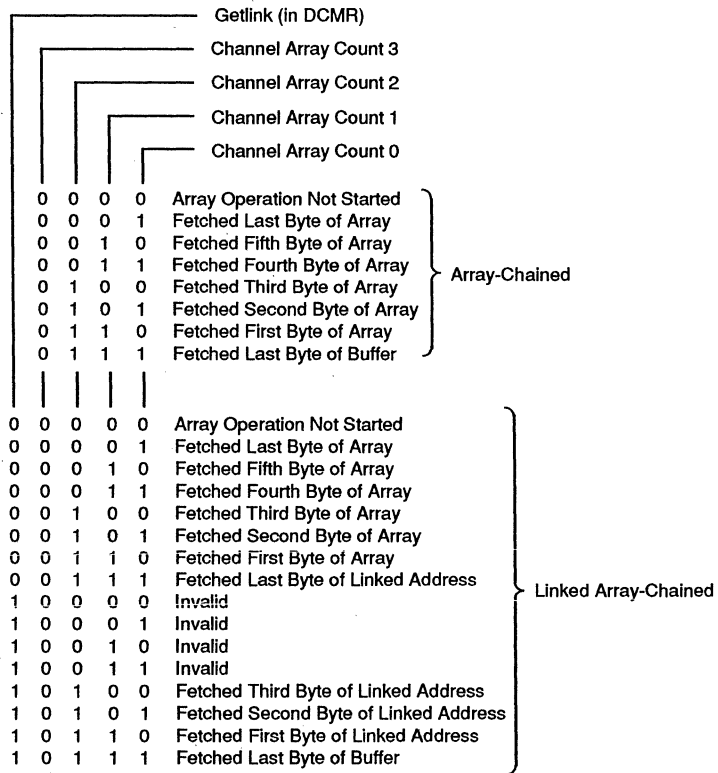
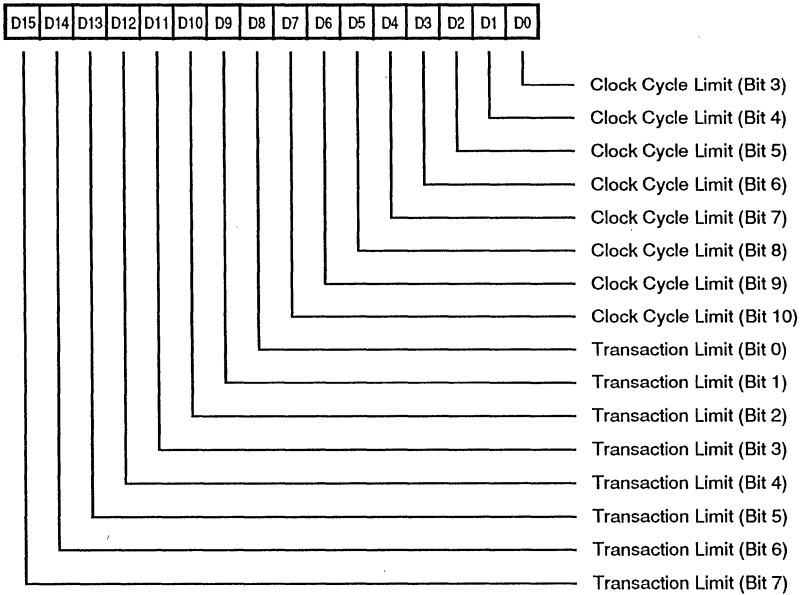


Figure 13b. Channel Array Count Bit Combinations

**Note:** See the Z16C32 Technical Manual for the appropriate table with Linked Status Transfer feature enabled.

Address: 01001 (Shared)



**Notes:**

BDCR Controls the amount of time that DMA may remain bus master.

Bits 15 through 8 are used to select a limit for the number of DMA transfers on the Bus while the DMA is bus master. This limit is a binary number, a value of zero disables the transaction limit function.

Bits 7 through 0 are used to select a limit for the number of clock cycles that the DMA may remain on the bus as bus master.

Bus transaction will always complete, even if the clock cycle limit is exceeded during the bus cycle, and even if the cycle is extended by external hardware signalling through /WAIT//RDY.

**Figure 14. Burst Dwell Control Register (BDCR)**

## CONTROL REGISTERS (Continued)

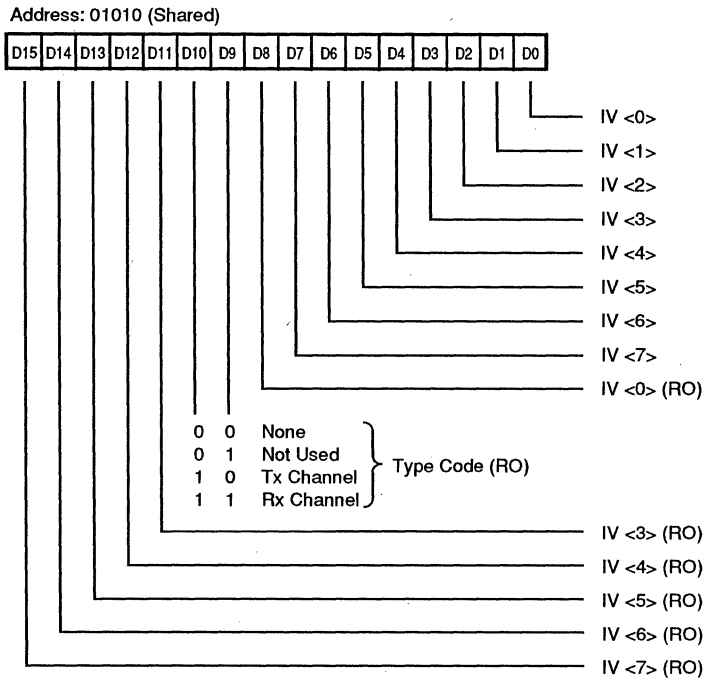


Figure 15. DMA Interrupt Vector Register (DIVR)

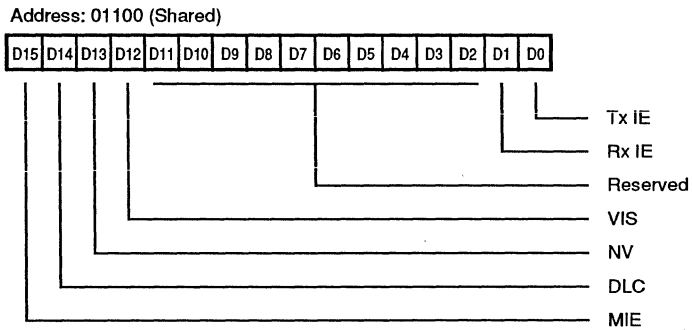


Figure 16. DMA Interrupt Control Register (DICR)

Address: 01101 (Shared)

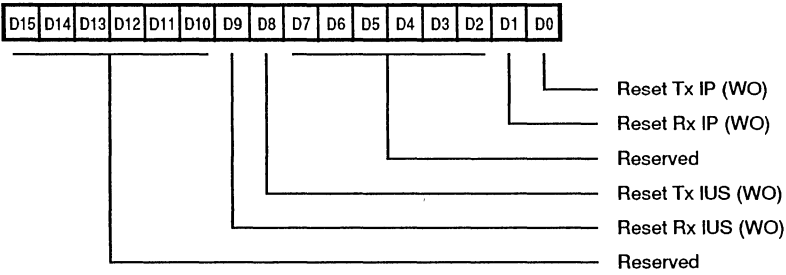


Figure 17. Clear DMA Interrupt Register (CDIR)

Address: 01110 (Shared)

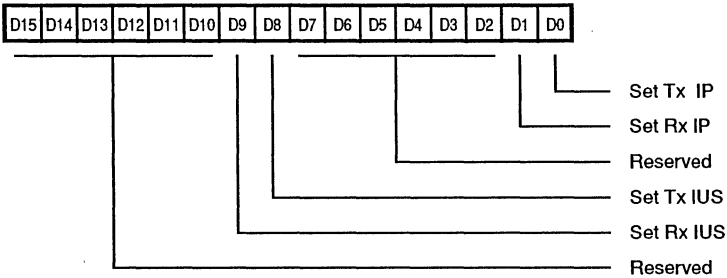


Figure 18. Set DMA Interrupt Register (SDIR)

Address: 01111 \*

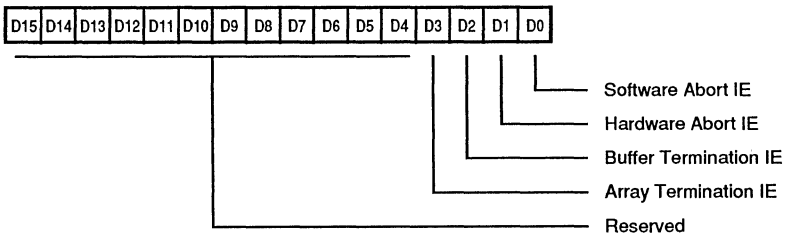


Figure 19. Tx/Rx DMA Interrupt Arm (TDIAR)/(RDIAR)

\* The format of this register is the same for the receiver and transmitter. The transmit register is accessed by addressing it with the D//C pin Low (0). The receive register is accessed by addressing it with the D//C pin High (1). This applies to Figures 19 through 25.

## CONTROL REGISTERS (Continued)

Address: 10101 \*

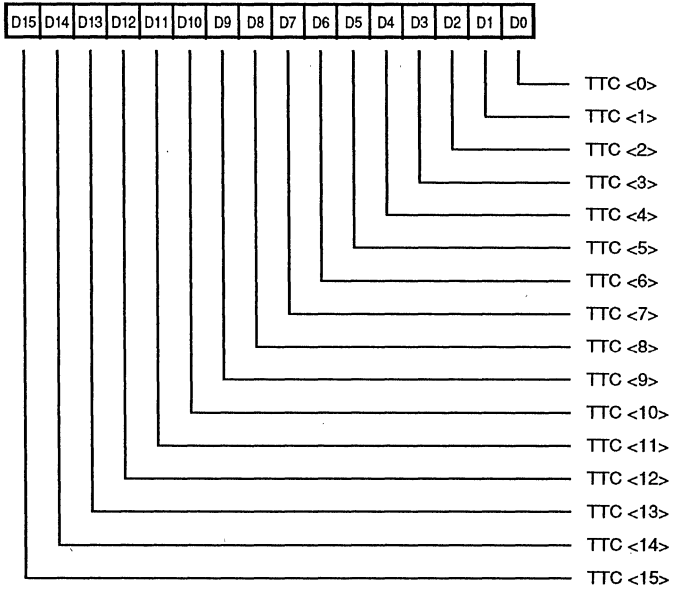


Figure 20. Tx/Rx Byte Count Register (TBCR)/(RBCR)

Address: 10110 \*

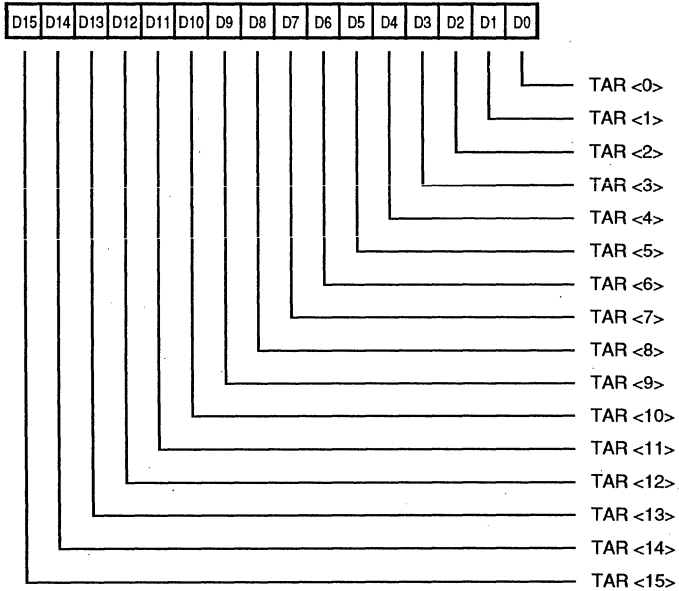


Figure 21. Tx/Rx Address Register (lower) (TARL)/(RARL)

Address: 10111 \*

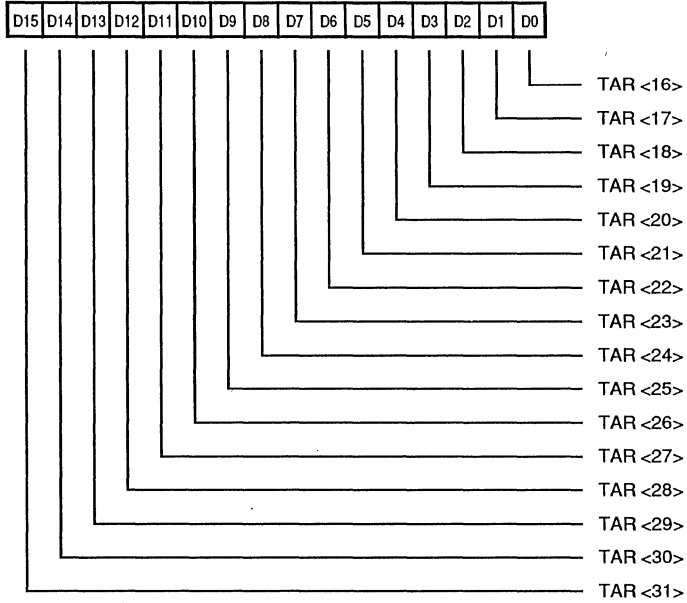


Figure 22. Tx/Rx Address Register (Upper) (TARU)/(RARU)

Address: 11101 \*

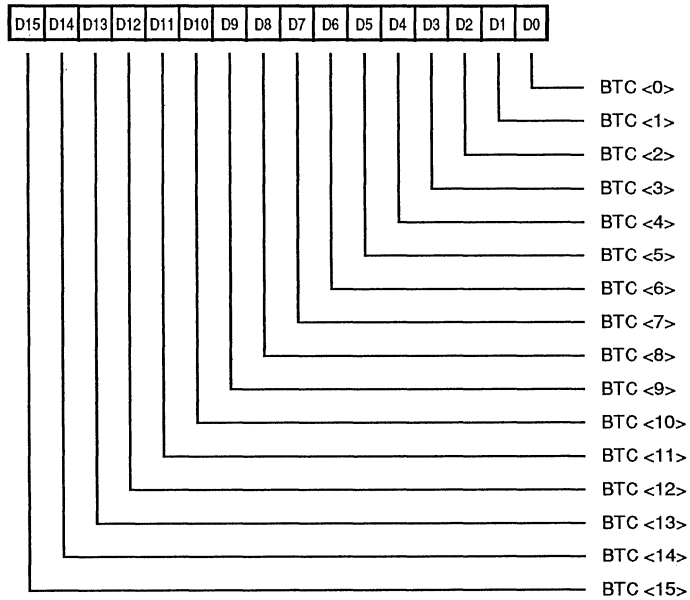


Figure 23. Next Tx/Rx Byte Counter Register (NTBCR)/(RTBCR)

## CONTROL REGISTERS (Continued)

Address: 11110\*

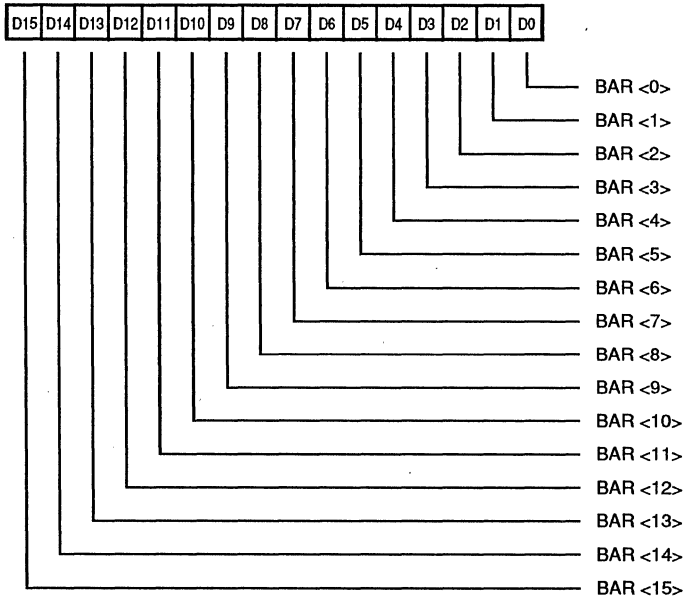


Figure 24. Next Tx/Rx Address Register (Lower) (NTARL)/(RTARL)

Address: 11111\*

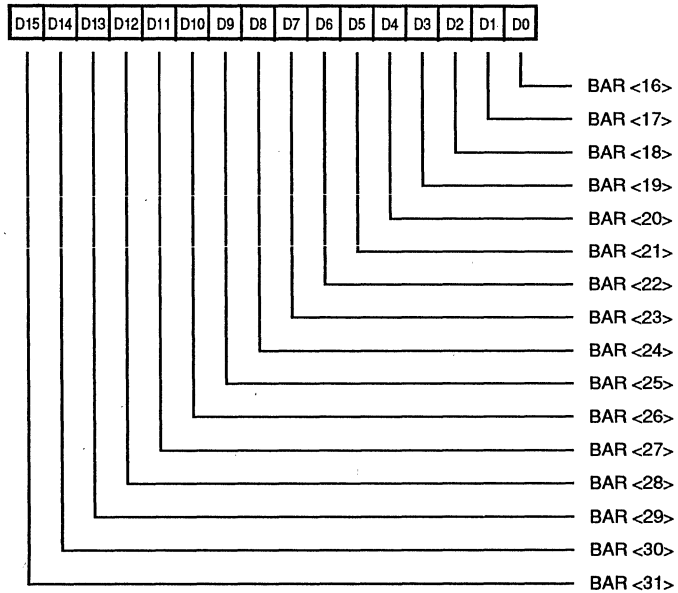


Figure 25. Next Tx/Rx Address Register (Upper) (NTARU)/(RTARU)

		AD15	AD0
Base Address	Buffer #1	AD<31-24>	AD<23-16>
Base Address + 2		AD<15-8>	AD<7-0>
Base Address + 4		CNT<15-8>	CNT<7-0>
Base Address + 6	Buffer #2	AD<31-24>	AD<23-16>
Base Address + 8		AD<15-8>	AD<7-0>
Base Address + 10		CNT<15-8>	CNT<7-0>
Base Address + 12	Buffer #3	AD<31-24>	AD<23-16>
Base Address + 14		AD<15-8>	AD<7-0>
Base Address + 16		CNT<15-8>	CNT<7-0>
Last Base Address	Dummy	Ignored	Ignored
Last Base Address + 2		Ignored	Ignored
Last Base Address + 4		0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
Base Address Register After Termination			

**Figure 26a. Array-Chained, 16-Bit Bus, Big End Array**

**Note:**

The addition of frame status/control information in the array with Linked Frame Status Transfer Enabled is similar for Big and Little End Array. See Figure 26b.



## CONTROL REGISTERS (Continued)

		AD15	AD0
Base Address	Buffer #1	AD<31-24>	AD<23-16>
Base Address + 2		AD<15-8>	AD<7-0>
Base Address + 4		CNT <15-8>	CNT <7-0>
Base Address + 6		RSB/TCB <15-8>	RSB/TCB <7-0>
Base Address + 8		RCHR/TCLR <15-8>	RCHR/TCLR <7-0>
Base Address + 10		0	0
Base Address + 12	Buffer #2	AD<31-24>	AD<23-16>
Base Address + 14		AD<15-8>	AD<7-0>
Base Address + 16		CNT <15-8>	CNT <7-0>
Base Address + 18		RSB/TCB <15-8>	RSB/TCB <7-0>
Base Address + 20		RCHR/TCLR <15-8>	RCHR/TCLR <7-0>
Base Address + 22		0	0
Base Address + 24	Buffer #3	AD<31-24>	AD<23-16>
Base Address + 26		AD<15-8>	AD<7-0>
Base Address + 28		CNT <15-8>	CNT <7-0>
Base Address + 30		RSB/TCB <15-8>	RSB/TCB <7-0>
Base Address + 32		RCHR/TCLR <15-8>	RCHR/TCLR <7-0>
Base Address + 34		0	0
Last Base Address	Dummy	Ignored	Ignored
Last Base Address + 2		Ignored	Ignored
Last Base Address + 4		0 0 0 0 0 0 0	0 0 0 0 0 0 0
Base Address Register After Termination			

Figure 26b. Array-Chained, 16-Bit Bus, Big End Array  
Linked Frame Status Transfer Enabled

	AD15	AD0	
Base Address	Buffer #1	AD<15-8>	AD<7-0>
Base Address + 2		AD<31-24>	AD<23-16>
Base Address + 4		CNT<15-8>	CNT<7-0>
Base Address + 6	Buffer #2	AD<15-8>	AD<7-0>
Base Address + 8		AD<31-24>	AD<23-16>
Base Address + 10		CNT<15-8>	CNT<7-0>
Base Address + 12	Buffer #3	AD<15-8>	AD<7-0>
Base Address + 14		AD<31-24>	AD<23-16>
Base Address + 16		CNT<15-8>	CNT<7-0>
Last Base Address	Dummy	Ignored	Ignored
Last Base Address + 2		Ignored	Ignored
Last Base Address + 4		0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
Base Address Register After Termination			

Figure 27. Array-Chained, 16-Bit Bus, Little End Array

## CONTROL REGISTERS (Continued)

		AD7	AD0
Base Address	Buffer #1	AD<31-24>	
Base Address + 1		AD<23-16>	
Base Address + 2		AD<15-8>	
Base Address + 3		AD<7-0>	
Base Address + 4		CNT<15-8>	
Base Address + 5	CNT<7-0>		
Base Address + 6	Buffer #2	AD<31-24>	
Base Address + 7		AD<23-16>	
Base Address + 8		AD<15-8>	
Base Address + 9		AD<7-0>	
Base Address + 10		CNT<15-8>	
Base Address + 11		CNT<7-0>	
Last Base Address	Dummy	Ignored	
Last Base Address + 1		Ignored	
Last Base Address + 2		Ignored	
Last Base Address + 3		Ignored	
Last Base Address + 4		0 0 0 0 0 0 0	
Last Base Address + 5		0 0 0 0 0 0 0	
Base Address Register After Termination			

**Figure 28a. Array-Chained, 8-Bit Bus, Big End Array**

**Note:**

The addition of frame status/control information in the array with Linked Frame Status Transfer Enabled is similar for Big and Little End Array. See Figure 28b.

		AD7	AD0
Base Address	Buffer #1	AD<31-24>	
Base Address + 1		AD<23-16>	
Base Address + 2		AD<15-8>	
Base Address + 3		AD<7-0>	
Base Address + 4		CNT<15-8>	
Base Address + 5		CNT<7-0>	
Base Address + 6		RSB/TCB <15-8>	
Base Address + 7		RSB/TCB <7-0>	
Base Address + 8		RCHR/TCLR <15-8>	
Base Address + 9		RSHR/TCLR <7-0>	
Base Address + 10		0	
Base Address + 11	0		
Base Address + 12	Buffer #2	AD<31-24>	
Base Address + 13		AD<23-16>	
Base Address + 14		AD<15-8>	
Base Address + 15		AD<7-0>	
Base Address + 16		CNT<15-8>	
Base Address + 17		CNT<7-0>	
Base Address + 18		RSB/TCB <15-8>	
Base Address + 19		RSB/TCB <7-0>	
Base Address + 20		RCHR/TCLR <15-8>	
Base Address + 21		RSHR/TCLR <7-0>	
Base Address + 22		0	
Base Address + 23		0	
Last Base Address	Dummy	Ignored	
Last Base Address + 1		Ignored	
Last Base Address + 2		Ignored	
Last Base Address + 3		Ignored	
Last Base Address + 4		0 0 0 0 0 0 0	
Last Base Address + 5		0 0 0 0 0 0 0	
Base Address Register After Termination			

**Figure 28b. Array-Chained, 8-Bit Bus, Big End Array, Linked Frame Status Transfer Enabled**

## CONTROL REGISTERS (Continued)

	AD7	AD0
Base Address	Buffer #1	AD<7-0>
Base Address + 1		AD<15-8>
Base Address + 2		AD<23-16>
Base Address + 3		AD<31-24>
Base Address + 4		CNT<7-0>
Base Address + 5		CNT<15-8>
Base Address + 6	Buffer #2	AD<7-0>
Base Address + 7		AD<15-8>
Base Address + 8		AD<23-16>
Base Address + 9		AD<31-24>
Base Address + 10		CNT<7-0>
Base Address + 11		CNT<15-8>
Last Base Address	Dummy	Ignored
Last Base Address + 1		Ignored
Last Base Address + 2		Ignored
Last Base Address + 3		Ignored
Last Base Address + 4		0 0 0 0 0 0 0
Last Base Address + 5		0 0 0 0 0 0 0
Base Address Register After Termination		

Figure 29. Array-Chained, 8-Bit Bus, Little End Array

		AD15	AD0
Base Address	Buffer #1	AD<31-24>	AD<23-16>
Base Address + 2		AD<15-8>	AD<7-0>
Base Address + 4		CNT<15-8>	CNT<7-0>
Base Address + 6	Base #2	AD<31-24>	AD<23-16>
Base Address + 8	Buffer #2	AD<15-8>	AD<7-0>
#2 Base Address		AD<31-24>	AD<23-16>
#2 Base Address + 2		AD<15-8>	AD<7-0>
#2 Base Address + 4		CNT<15-8>	CNT<7-0>
#2 Base Address + 6	Base #3	AD<31-24>	AD<23-16>
#2 Base Address + 8	Buffer #3	AD<15-8>	AD<7-0>
#3 Base Address		AD<31-24>	AD<23-16>
#3 Base Address + 2		AD<15-8>	AD<7-0>
#3 Base Address + 4		CNT<15-8>	CNT<7-0>
#n - 1 Base Address + 6	Base #n	AD<31-24>	AD<23-16>
#n - 1 Base Address + 8	Buffer #n	AD<15-8>	AD<7-0>
#n Base Address		Ignored	Ignored
#n Base Address + 2		Ignored	Ignored
#n Base Address + 4		0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0

**Figure 30a. Linked Array-Chained, 16-Bit Bus, Big End Array**

**Note:**

The addition of frame status/control information in the array with Linked Frame Status Transfer Enabled is similar for Big and Little End Array. See Figure 30b.

## CONTROL REGISTERS (Continued)

		AD15	AD0
Base Address	Buffer #1	AD<31-24>	AD<23-16>
Base Address + 2		AD <15-8>	AD<7-0>
Base Address + 4		CNT<15-8>	CNT<7-0>
Base Address + 6		RSB/TCB <15-8>	RSB/TCB <7-0>
Base Address + 8		RCHR/TCLR <15-8>	RCHR/TCLR <7-0>
Base Address + 10		0	0
Base Address + 12	Base #2	AD<31-24>	AD<23-16>
Base Address + 14	Buffer #2	AD <15-8>	AD<7-0>
#2 Base Address		AD<31-24>	AD<23-16>
#2 Base Address + 2		AD <15-8>	AD<7-0>
#2 Base Address + 4		CNT<15-8>	CNT<7-0>
#2 Base Address + 6		RSB/TCB <15-8>	RSB/TCB <7-0>
#2 Base Address + 8		RCHR/TCLR <15-8>	RCHR/TCLR <7-0>
#2 Base Address + 10	0	0	
#2 Base Address + 12	Base #3	AD<31-24>	AD<23-16>
#2 Base Address + 14	Buffer #3	AD <15-8>	AD<7-0>
#3 Base Address		AD<31-24>	AD<23-16>
#3 Base Address + 2		AD <15-8>	AD<7-0>
#3 Base Address + 4		CNT<15-8>	CNT<7-0>
#3 Base Address + 6		RSB/TCB <15-8>	RSB/TCB <7-0>
#3 Base Address + 8		RCHR/TCLR <15-8>	RCHR/TCLR <7-0>
#3 Base Address + 10	0	0	

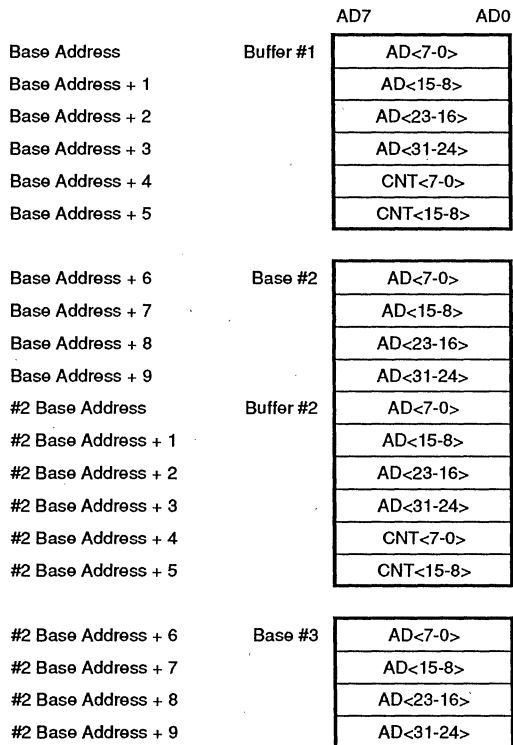
Figure 30b. Linked Array-Chained, 16-Bit Bus, Big End Array

		AD15	AD0
Base Address	Buffer #1	AD<15-8>	AD<7-0>
Base Address + 2		AD<31-24>	AD<23-16>
Base Address + 4		CNT<15-8>	CNT<7-0>
Base Address + 6	Base #2	AD<15-8>	AD<7-0>
Base Address + 8	Buffer #2	AD<31-24>	AD<23-16>
#2 Base Address		AD<15-8>	AD<7-0>
#2 Base Address + 2		AD<31-24>	AD<23-16>
#2 Base Address + 4		CNT<15-8>	CNT<7-0>
#2 Base Address + 6	Base #3	AD<15-8>	AD<7-0>
#2 Base Address + 8	Buffer #3	AD<31-24>	AD<23-16>
#3 Base Address		AD<15-8>	AD<7-0>
#3 Base Address + 2		AD<31-24>	AD<23-16>
#3 Base Address + 4		CNT<15-8>	CNT<7-0>
#n - 1 Base Address + 6	Base #n	AD<15-8>	AD<7-0>
#n - 1 Base Address + 8	Buffer #n	AD<31-24>	AD<23-16>
#n Base Address		Ignored	Ignored
#n Base Address + 2		Ignored	Ignored
#n Base Address + 4		0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0

Figure 31. Linked Array-Chained, 16-Bit Bus, Little End Array



## CONTROL REGISTERS (Continued)



**Figure 32a. Linked Array-Chained, 8-Bit Bus, Big End Array**

**Note:**

The addition of frame status/control information in the array with Linked Frame Status Transfer Enabled is similar to Big End Array. See Figure 32b.

		AD7	AD0
Base Address	Buffer #1	AD<31-24>	
Base Address + 1		AD<23-16>	
Base Address + 2		AD<15-8>	
Base Address + 3		AD<7-0>	
Base Address + 4		CNT<15-8>	
Base Address + 5		CNT<7-0>	
Base Address + 6		RSB/TCB <15-8>	
Base Address + 7		RSB/TCB <7-0>	
Base Address + 8		RCHR/TCLR <15-8>	
Base Address + 9		RCHR/TCLR <7-0>	
Base Address + 10		0	
Base Address + 11	0		
Base Address + 12	Base #2	AD<31-24>	
Base Address + 13		AD<23-16>	
Base Address + 14		AD<15-8>	
Base Address + 15		AD<7-0>	
#2 Base Address	Buffer #2	AD<31-24>	
#2 Base Address + 1		AD<23-16>	
#2 Base Address + 2		AD<15-8>	
#2 Base Address + 3		AD<7-0>	
#2 Base Address + 4		CNT<15-8>	
#2 Base Address + 5		CNT<7-0>	
#2 Base Address + 6		RSB/TCB <15-8>	
#2 Base Address + 7		RSB/TCB <7-0>	
#2 Base Address + 8		RCHR/TCLR <15-8>	
#2 Base Address + 9		RCHR/TCLR <7-0>	
#2 Base Address + 10		0	
#2 Base Address + 11	0		

Figure 32b. Linked Frame Status Transfer Enables

## CONTROL REGISTERS (Continued)

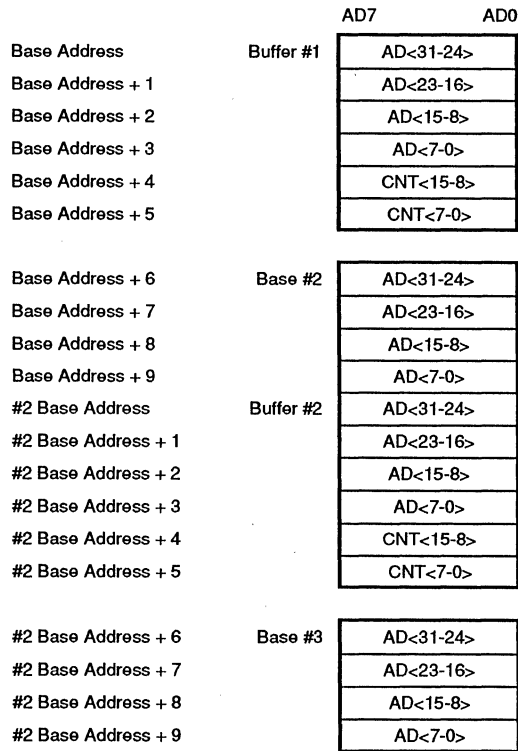


Figure 33. Linked Array-Chained 8-Bit Bus, Little End Array



# CONTROL REGISTERS (Continued)

Address: 00001

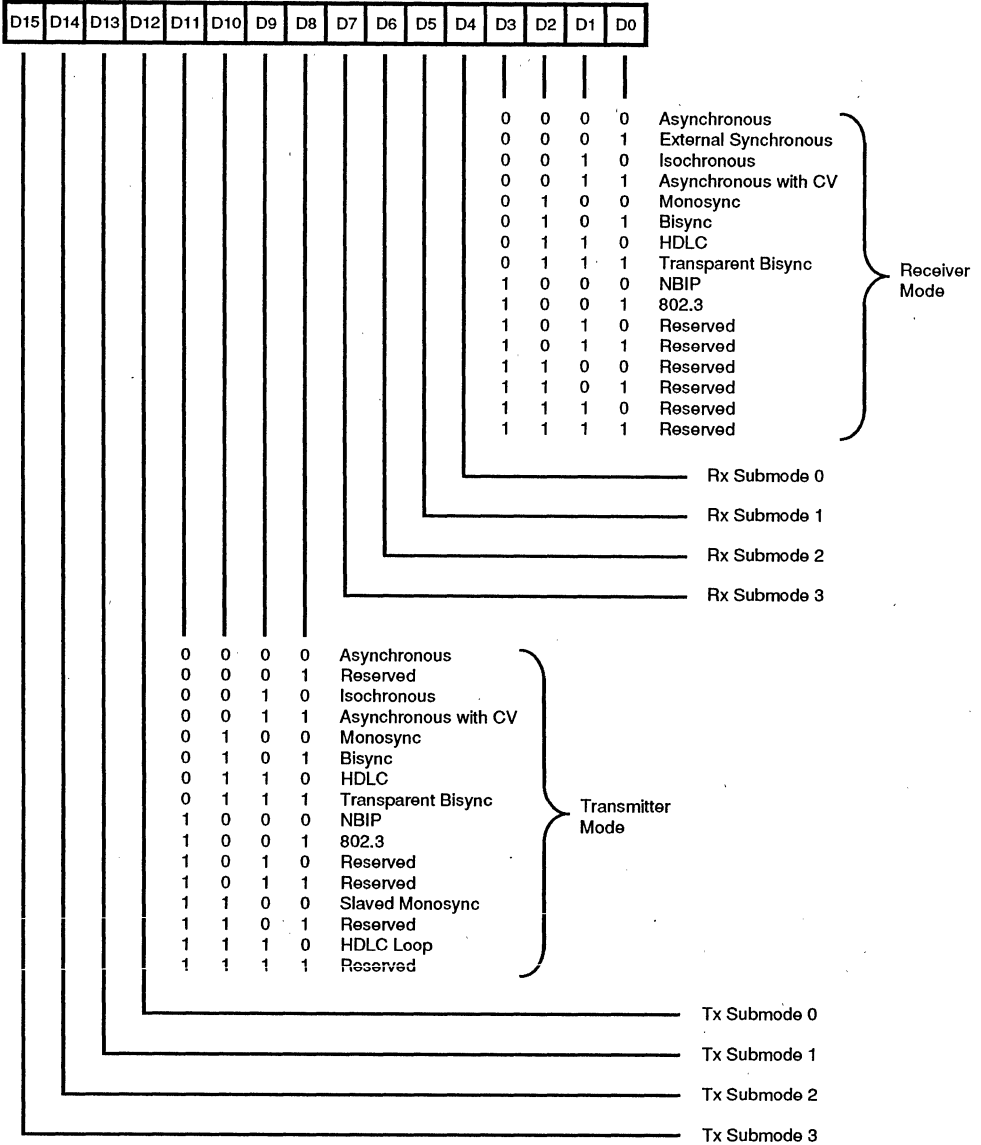


Figure 35. Channel Mode Register (CMR)

Address: 00001

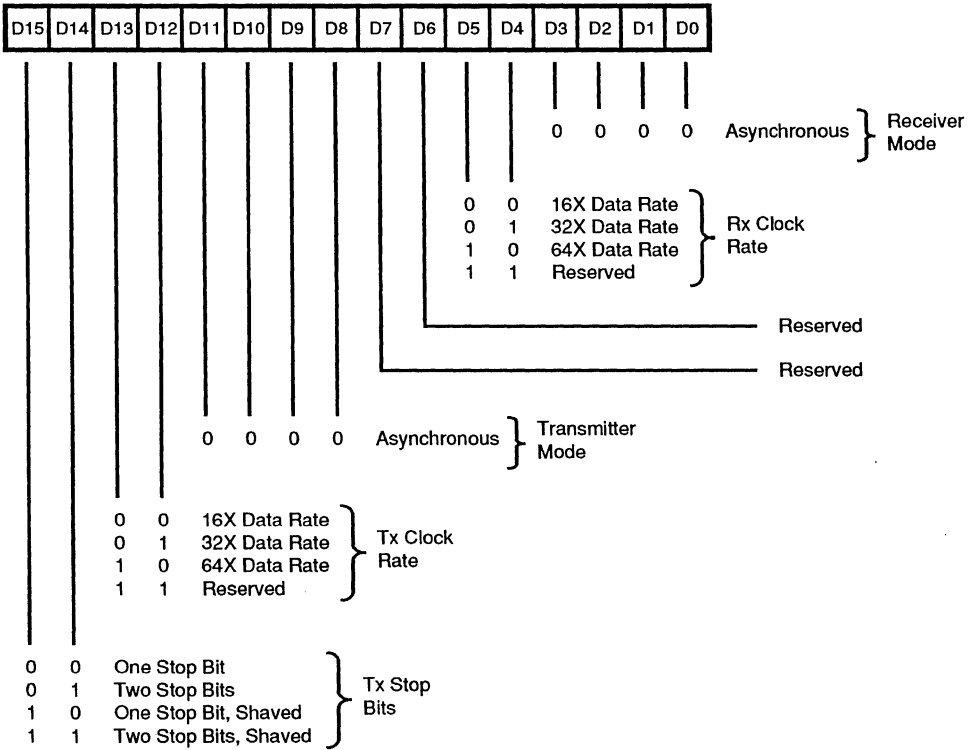


Figure 36. Channel Mode Register, Asynchronous Mode

Address: 00001

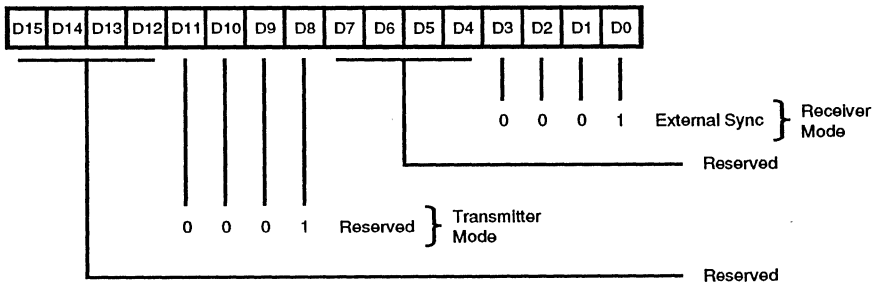


Figure 37. Channel Mode Register, External Sync Mode

## CONTROL REGISTERS (Continued)

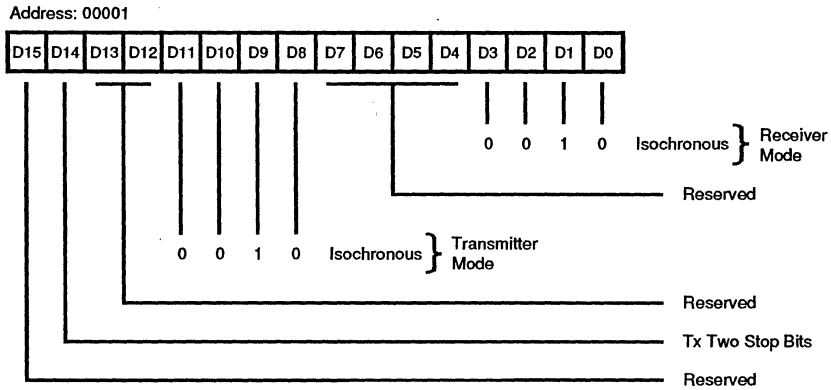


Figure 38. Channel Mode Register, Isochronous Mode

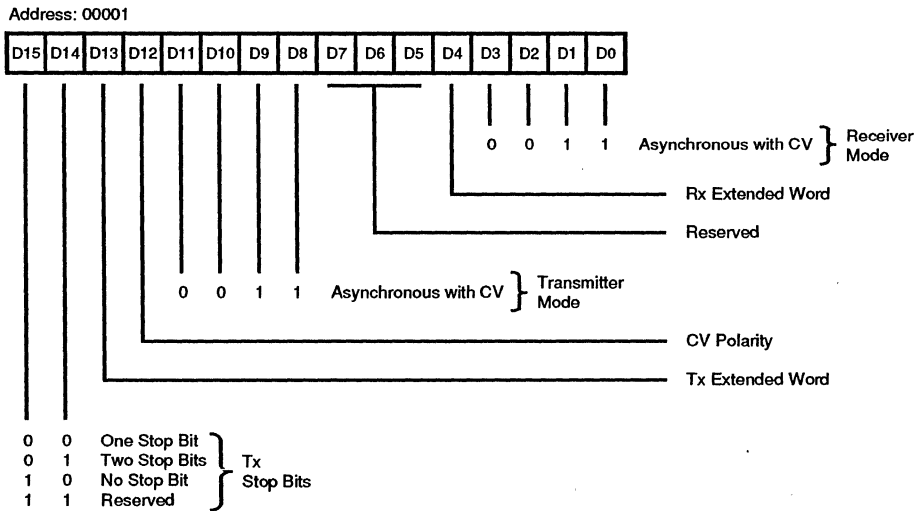


Figure 39. Channel Mode Register, Asynchronous Mode with Code Violation (MIL STD 1553)

Address: 00001

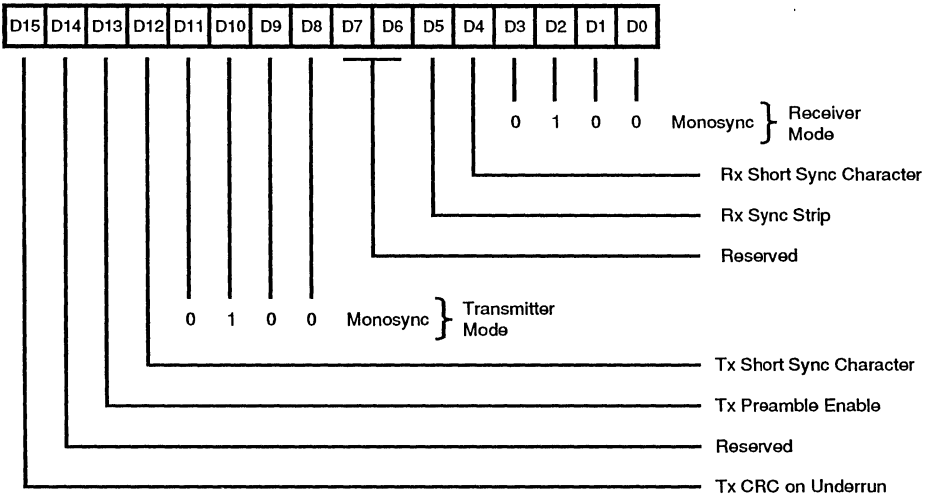


Figure 40. Channel Mode Register, Monosync Mode

Address: 00001

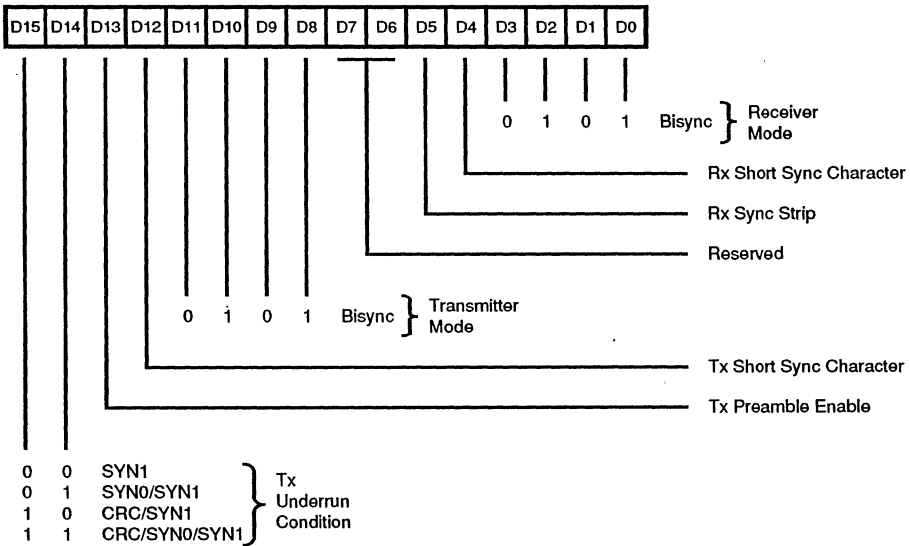


Figure 41. Channel Mode Register, Bisync Mode



# CONTROL REGISTERS (Continued)

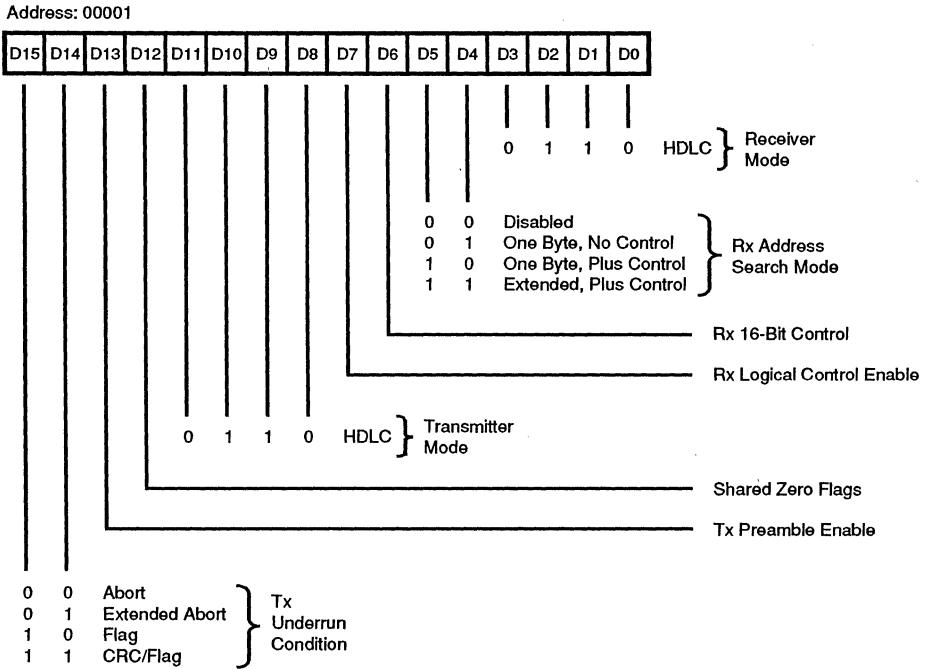


Figure 42. Channel Mode Register, HDLC Mode

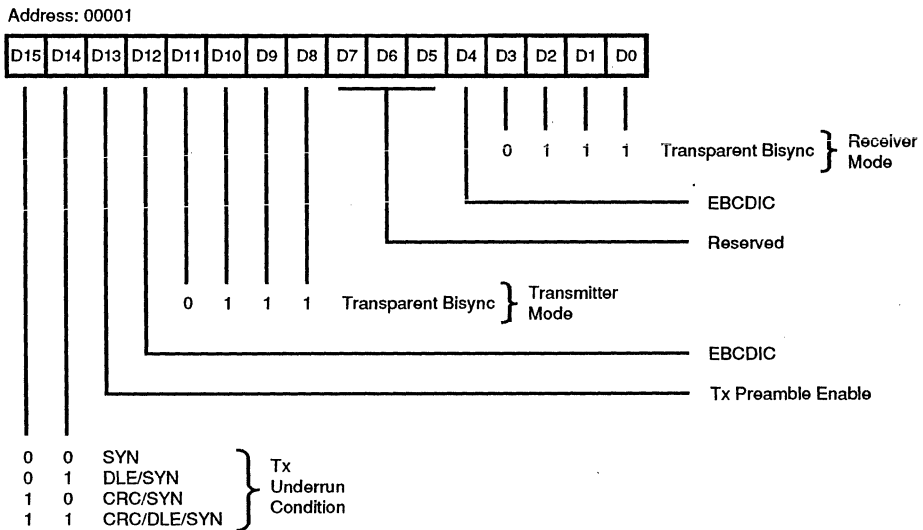


Figure 43. Channel Mode Register, Transparent Bisync Mode

Address: 00001

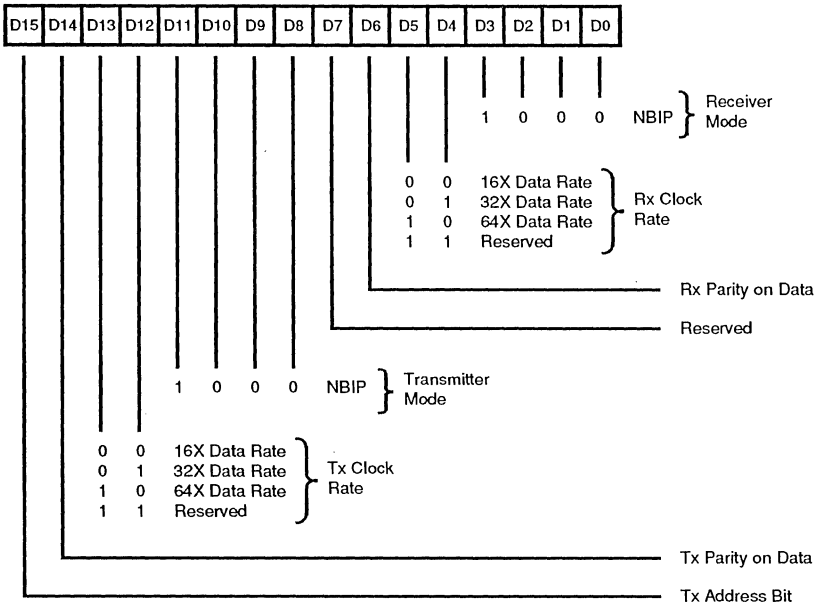


Figure 44. Channel Mode Register, NBIP Mode

Address: 00001

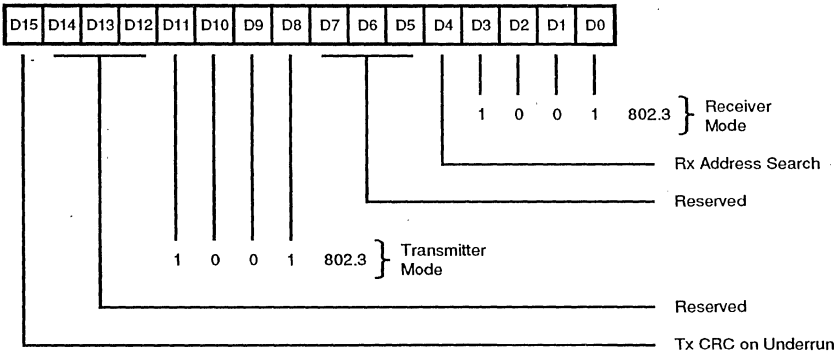


Figure 45. Channel Mode Register, 802.3 Mode

## CONTROL REGISTERS (Continued)

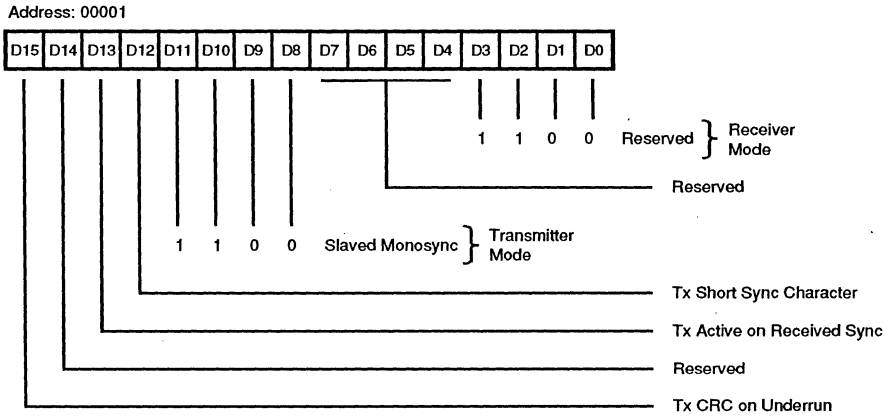


Figure 46. Channel Mode Register, Slaved Monosync Mode

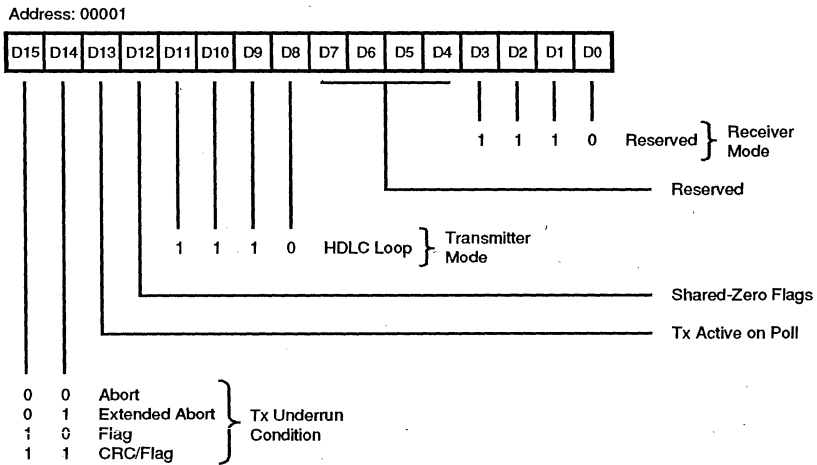


Figure 47. Channel Mode Register, HDLC Loop Mode

Address: 00010

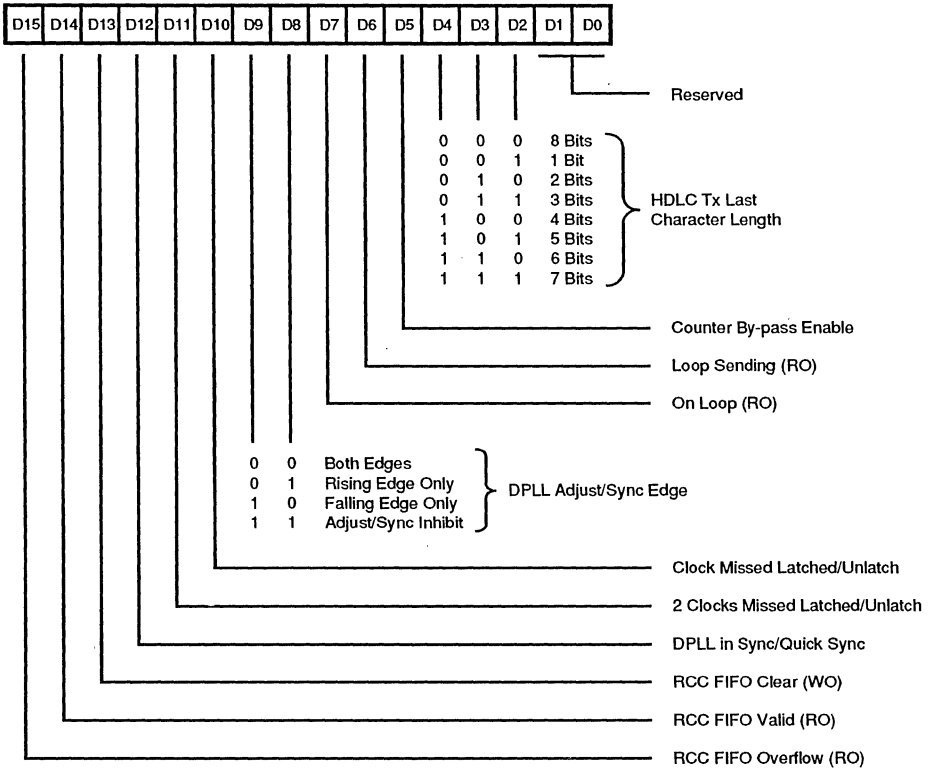


Figure 48. Channel Command/Status Register (CCSR)

# CONTROL REGISTERS (Continued)

Address: 00011

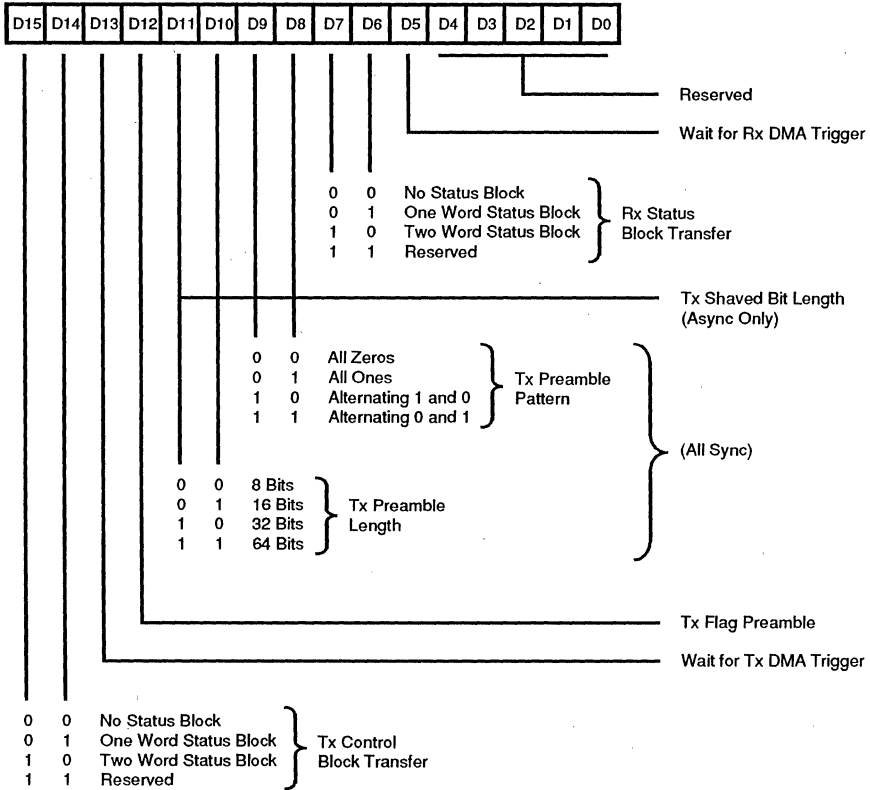


Figure 49. Channel Control Register (CCR)

Address: 00100

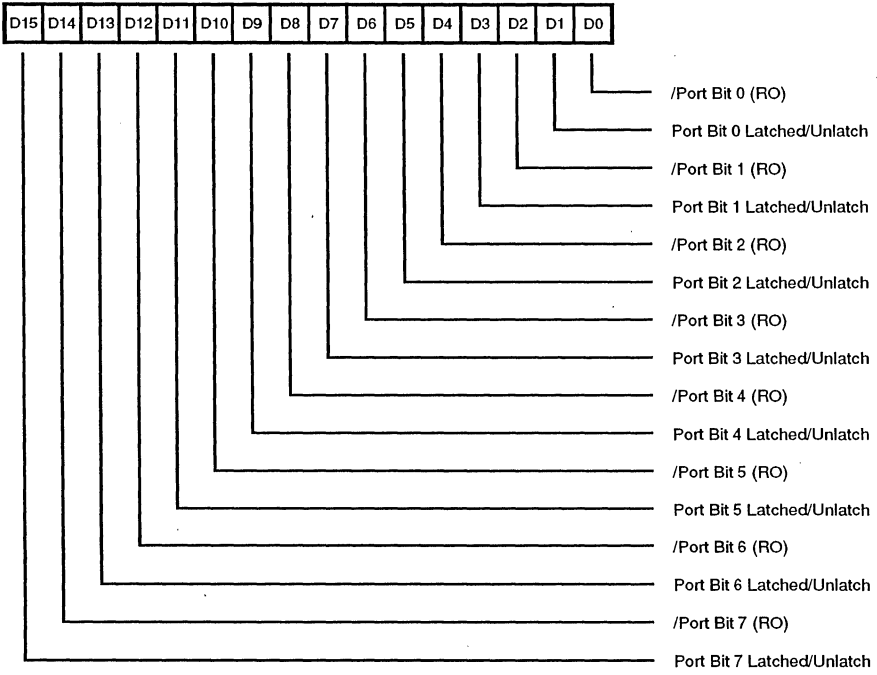


Figure 50. Port Status Register (PSR)

# CONTROL REGISTERS (Continued)

Address: 00101

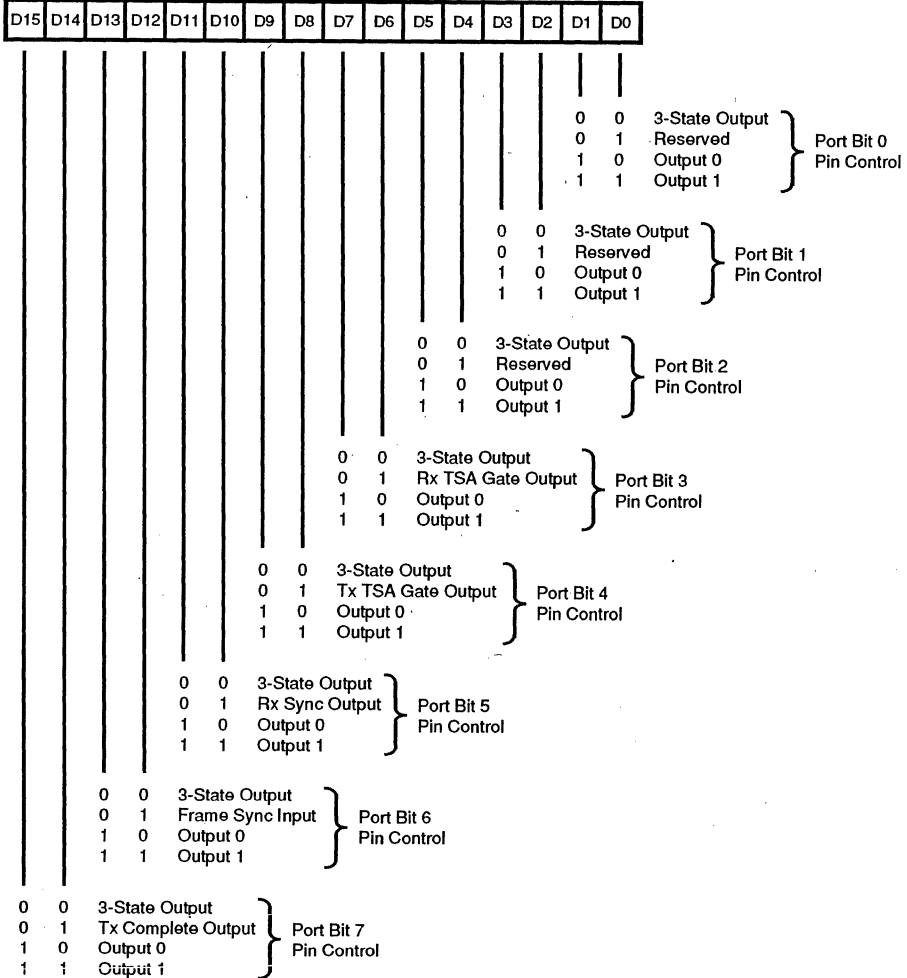


Figure 51. Port Control Register (PCR)

Address: 00110

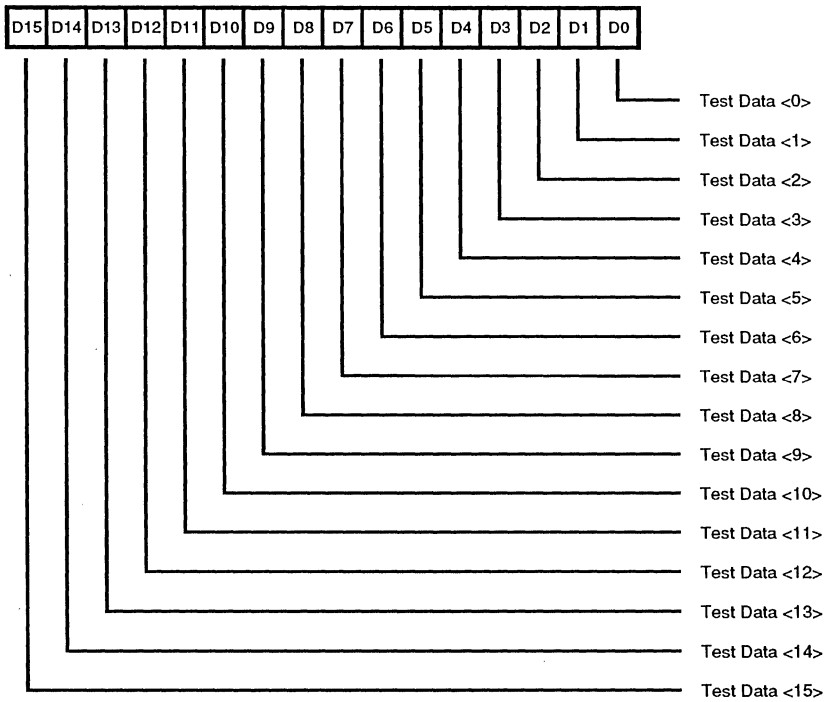


Figure 52. Test Mode Data Register (TMDR)



# CONTROL REGISTERS (Continued)

Address: 00111

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
											0	0	0	0	0	Null Address
											0	0	0	0	1	High Byte of Shifters
											0	0	0	1	0	CRC Byte 0
											0	0	0	1	1	CRC Byte 1
											0	0	1	0	0	Rx FIFO (Write)
											0	0	1	0	1	Clock Multiplexer Outputs
											0	0	1	1	0	CTR0 and CTR1 Counters
											0	0	1	1	1	Clock Multiplexer Inputs
											0	1	0	0	0	DPLL State
											0	1	0	0	1	Low Byte of Shifters
											0	1	0	1	0	CRC Byte 2
											0	1	0	1	1	CRC Byte 3
											0	1	1	0	0	Tx FIFO (Read)
											0	1	1	0	1	Reserved
											0	1	1	1	0	I/O and Device Status Latches
											0	1	1	1	1	Internal Daisy Chain
											1	0	0	0	0	Reserved
											1	0	0	0	1	Reserved
											1	0	0	1	0	Reserved
											1	0	0	1	1	Reserved
											1	0	1	0	0	Reserved
											1	0	1	0	1	Reserved
											1	0	1	1	0	Rx Count Holding Register
											1	0	1	1	1	Reserved
											1	1	0	0	0	Reserved
											1	1	0	0	1	Reserved
											1	1	0	1	0	Reserved
											1	1	0	1	1	Reserved
											1	1	1	0	0	Reserved
											1	1	1	1	0	Reserved
											1	1	1	1	1	Reserved

Figure 53. Test Mode Control Register (TMCR)

Address: 01000

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0		
													0	0	0	Disabled	
													0	0	1	/RxC Pin	
													0	1	0	/TxC Pin	
													0	1	1	DPLL Output	
													1	0	0	BRG0 Output	
													1	0	1	BRG1 Output	
													1	1	0	CTR0 Output	
													1	1	1	CTR1 Output	
																} Receive Clock Source	
													0	0	0		Disabled
													0	0	1		/RxC Pin
													0	1	0		/TxC Pin
													0	1	1		DPLL Output
													1	0	0	BRG0 Output	
													1	0	1	BRG1 Output	
													1	1	0	CTR0 Output	
													1	1	1	CTR1 Output	
																} Transmit Clock Source	
													0	0	0		Disabled
													0	0	1		/RxC Pin
													0	1	0		/TxC Pin
													0	1	1		DPLL Output
													1	0	0	BRG0 Output	
													1	0	1	BRG1 Output	
													1	1	0	CTR0 Output	
													1	1	1	CTR1 Output	
																} DPLL Clock Source	
													0	0	0		BRG0 Output
													0	1	0		BRG1 Output
													1	0	0		/RxC Pin
													1	1	0	/TxC Pin	
																} BRG0 Clock Source	
													0	0	0		CTR0 Output
													0	1	0		CTR1 Output
													1	0	0		/RxC Pin
													1	1	0	/TxC Pin	
																} BRG1 Clock Source	
													0	0	0		CTR0 Output
													0	1	0		CTR1 Output
													1	0	0		/RxC Pin
													1	1	0	/TxC Pin	
																} CTR0 Clock Source	
													0	0	0		Disabled
													0	1	0		Port0 Pin
													1	0	0		/RxC Pin
													1	1	0	/TxC Pin	
																} CTR1 Clock Source	
													0	0	0		Disabled
													0	1	0		Port1 Pin
													1	0	0		/RxC Pin
													1	1	0	/TxC Pin	

Figure 54. Clock Mode Control Register (CMCR)

## CONTROL REGISTERS (Continued)

Address: 01001

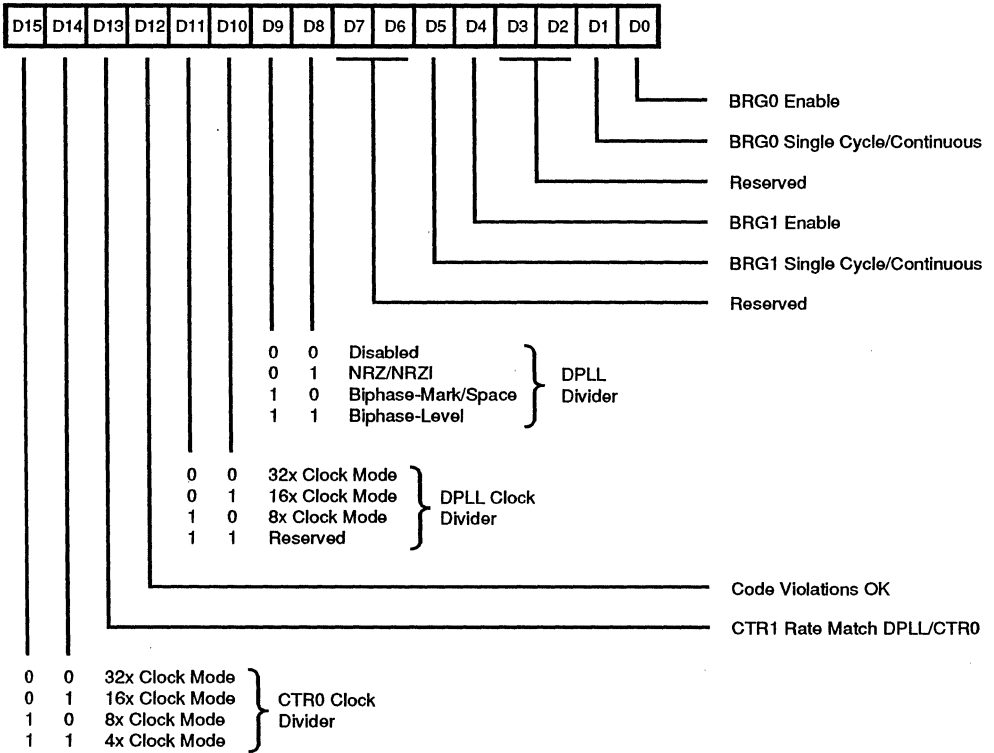


Figure 55. Hardware Configuration Register (HCR)

Address: 01010

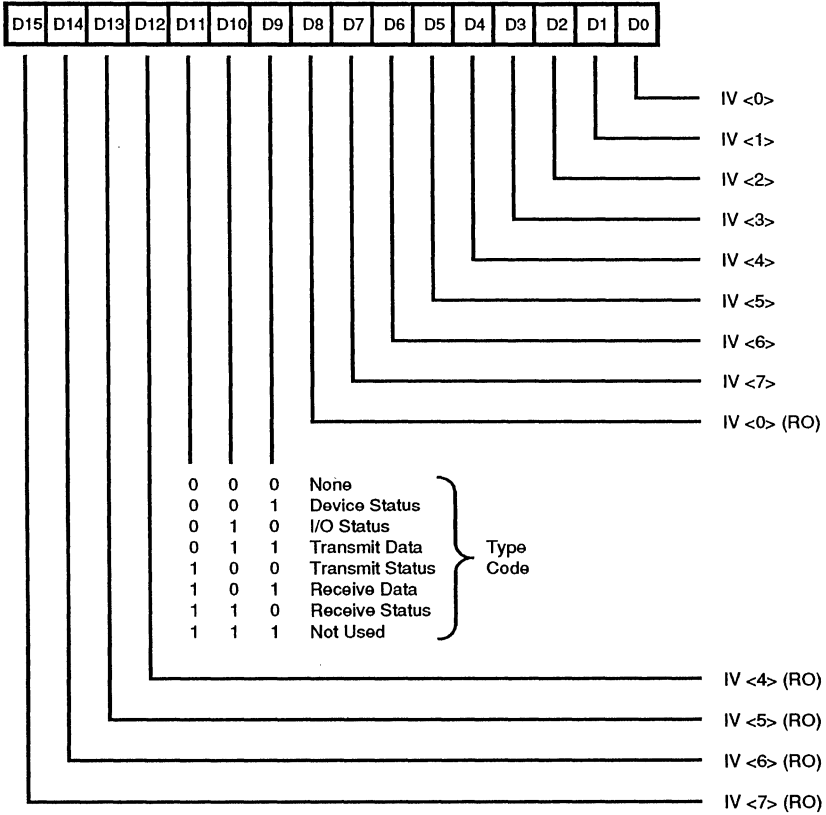


Figure 56. Interrupt Vector Register (IVR)

# CONTROL REGISTERS (Continued)

Address: 01011

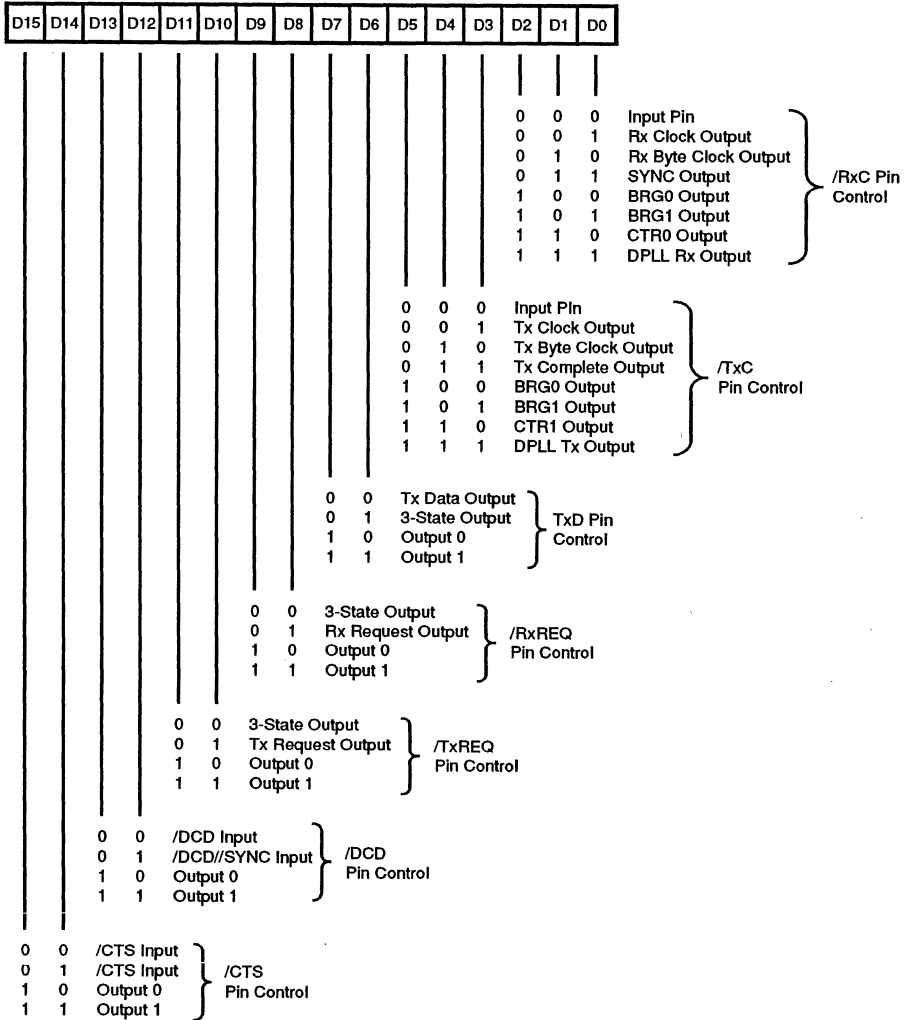


Figure 57. I/O Control Register (IOCR)

Address: 01100

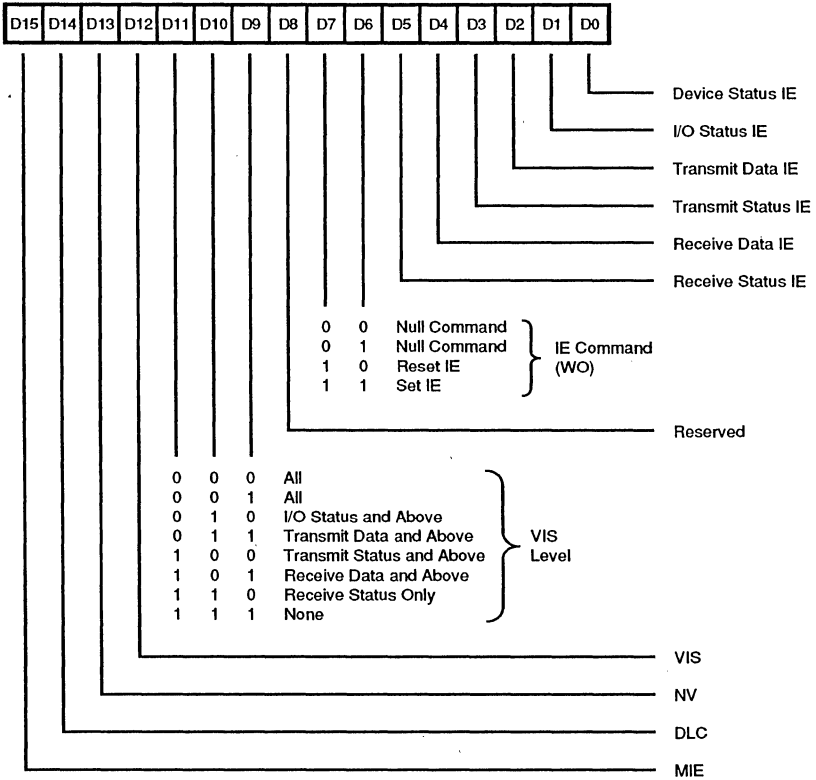


Figure 58. Interrupt Control Register (ICR)

## CONTROL REGISTERS (Continued)

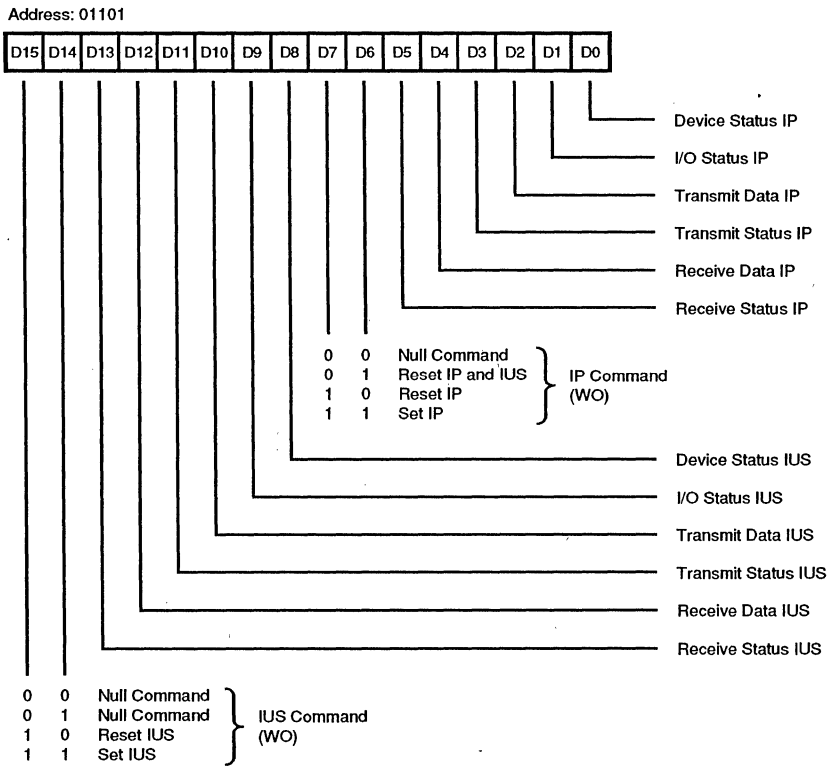


Figure 59. Daisy-Chain Control Register (DCCR)

Address: 01110

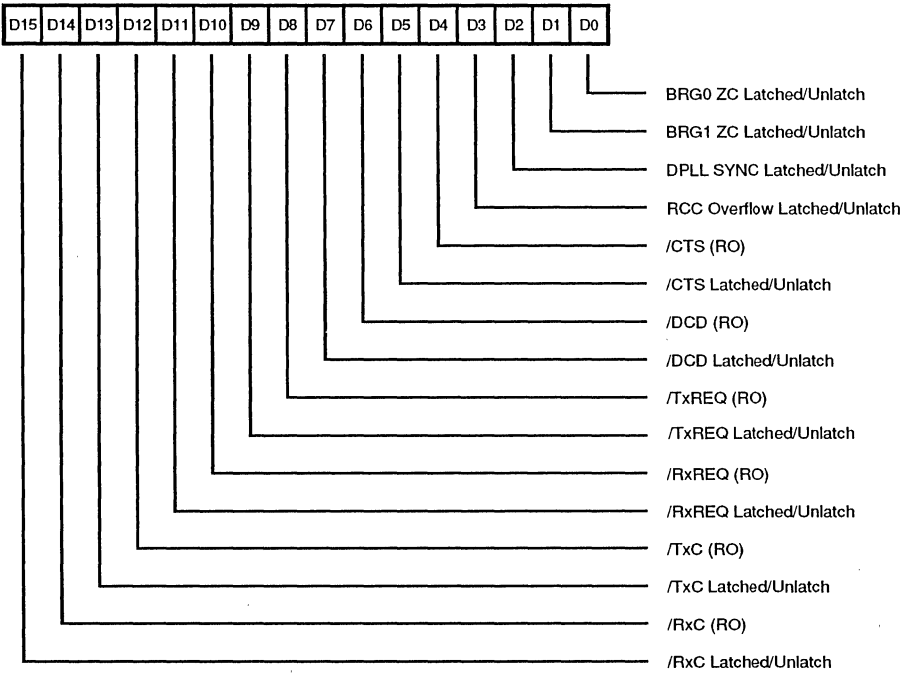


Figure 60. Miscellaneous Interrupt Status Register (MISR)





Address: 1x000

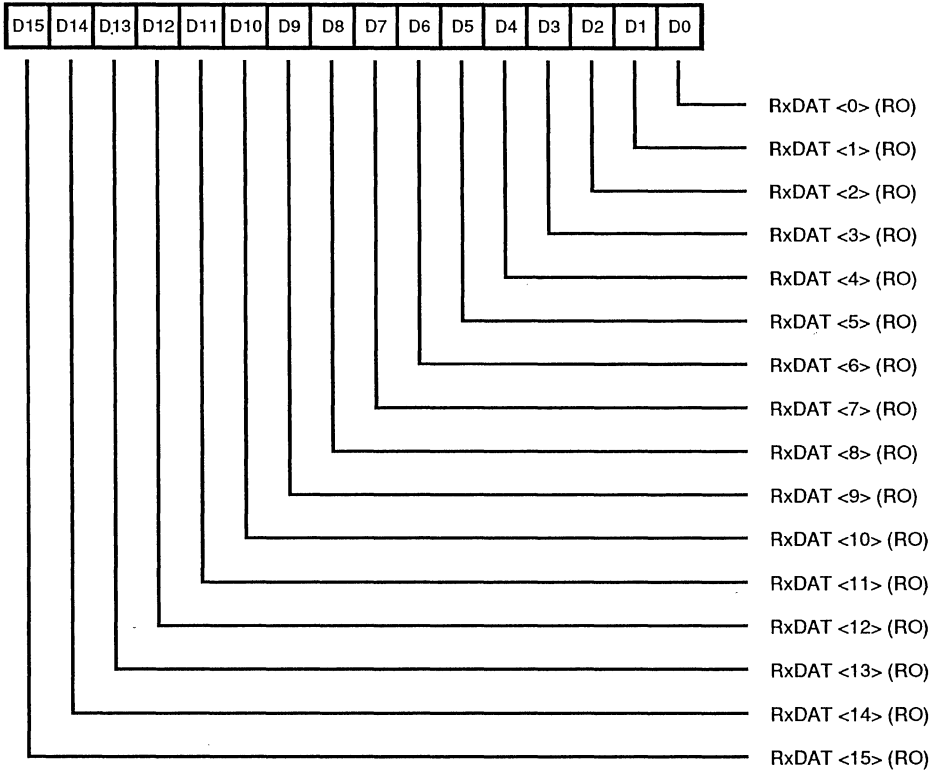


Figure 62. Receive Data Register (RDR)



Address: 10010

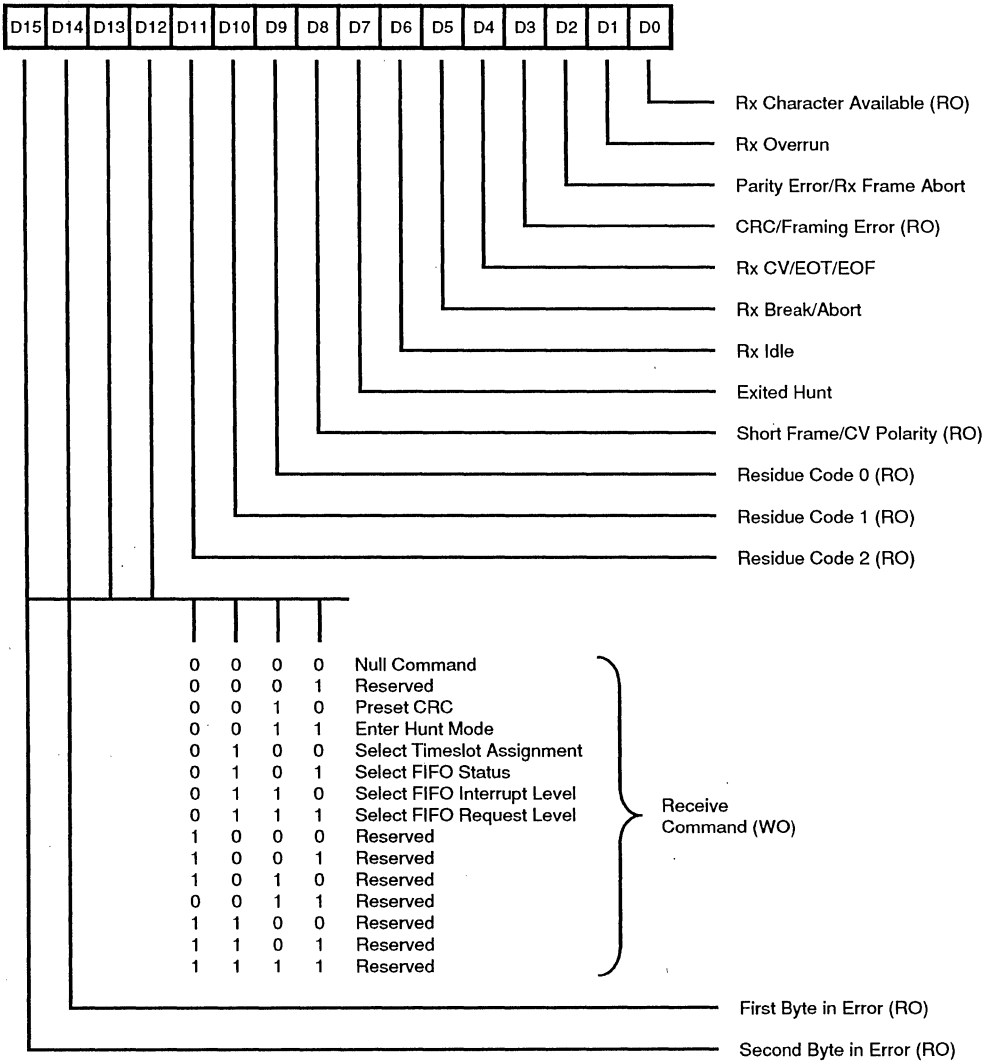


Figure 64. Receive Command Status Register (RCSR)

## CONTROL REGISTERS (Continued)

Address: 10011

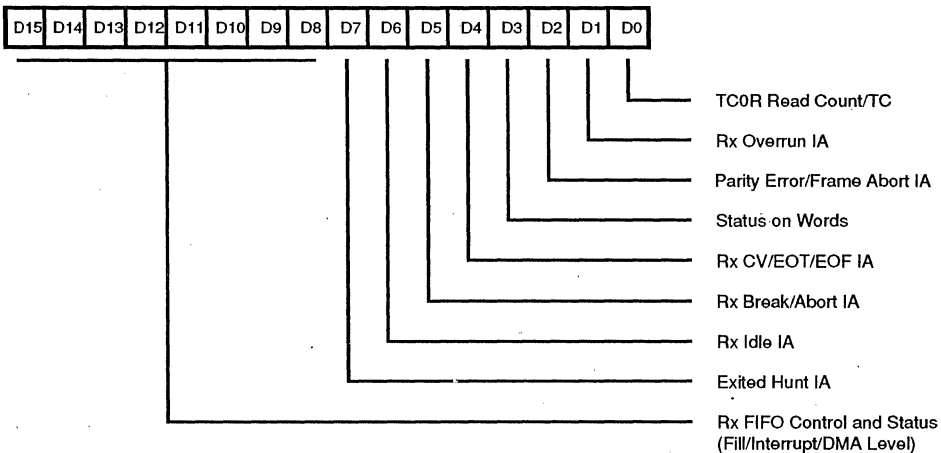


Figure 65a. Receive Interrupt Control Register (RICR)

Address: 10011

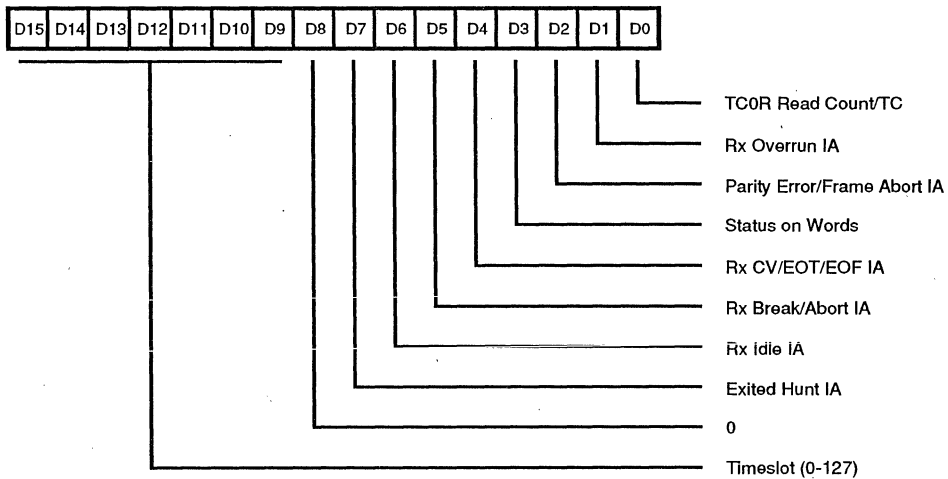


Figure 65b. Receive Interrupt Control Register (RICR)

Address: 10011

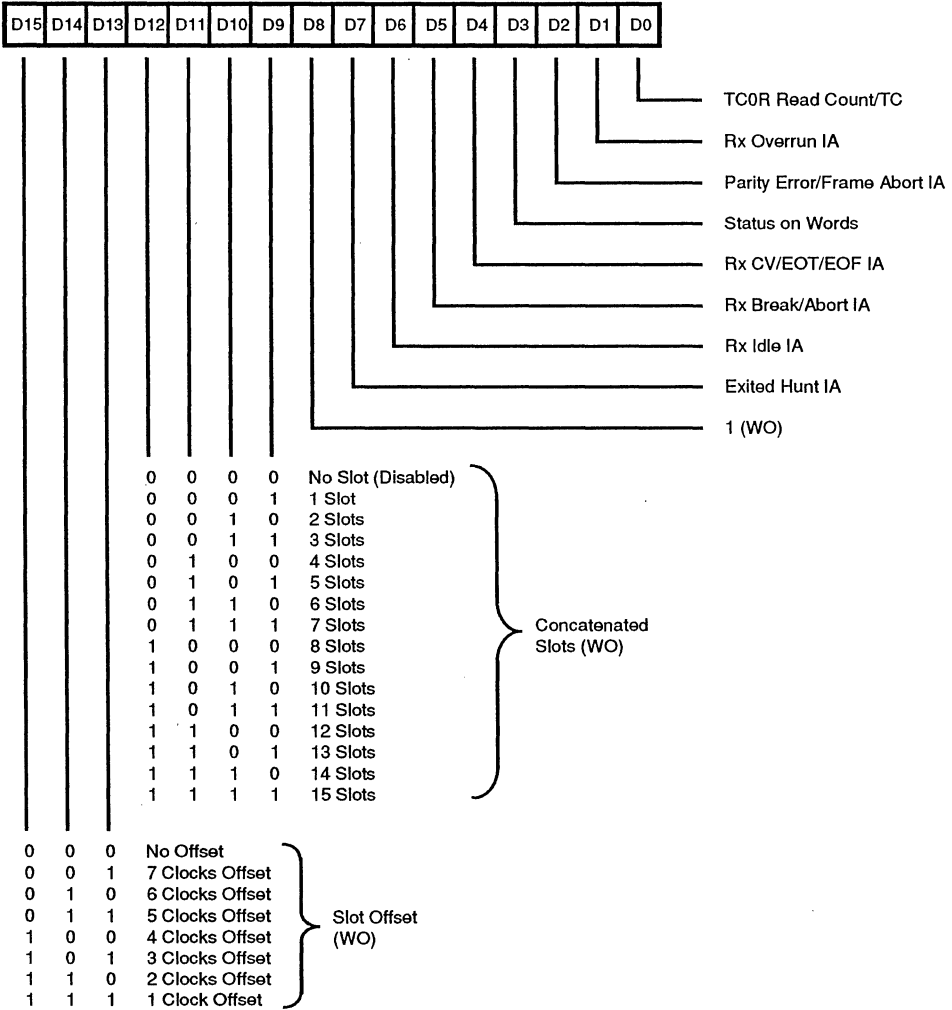


Figure 65c. Receive Interrupt Control Register (RICR)

## CONTROL REGISTERS (Continued)

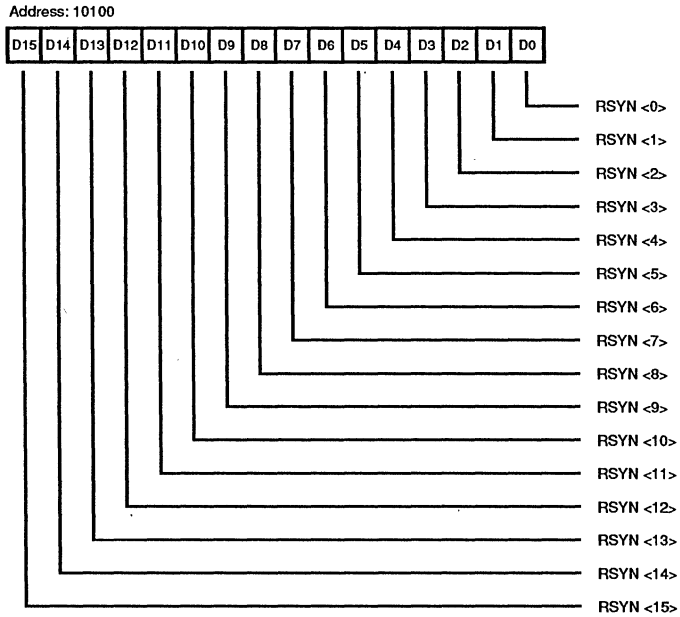


Figure 66. Receive Sync Register (RSR)

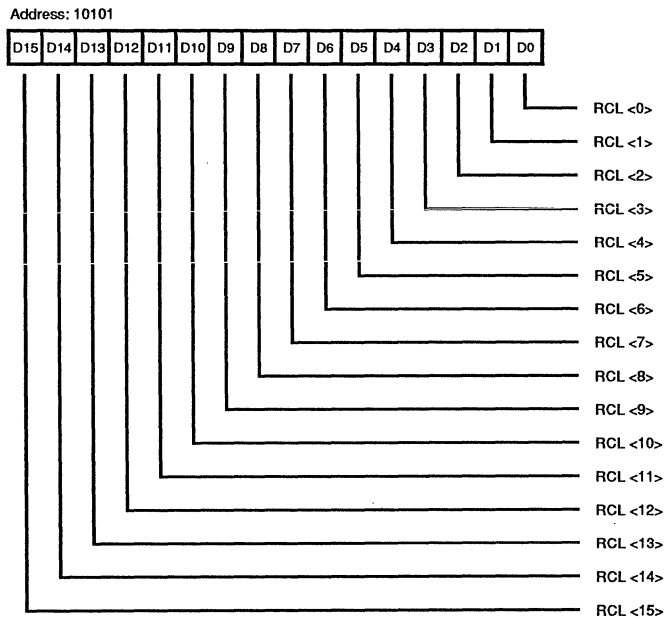


Figure 67. Receive Count Limit Register (RCLR)

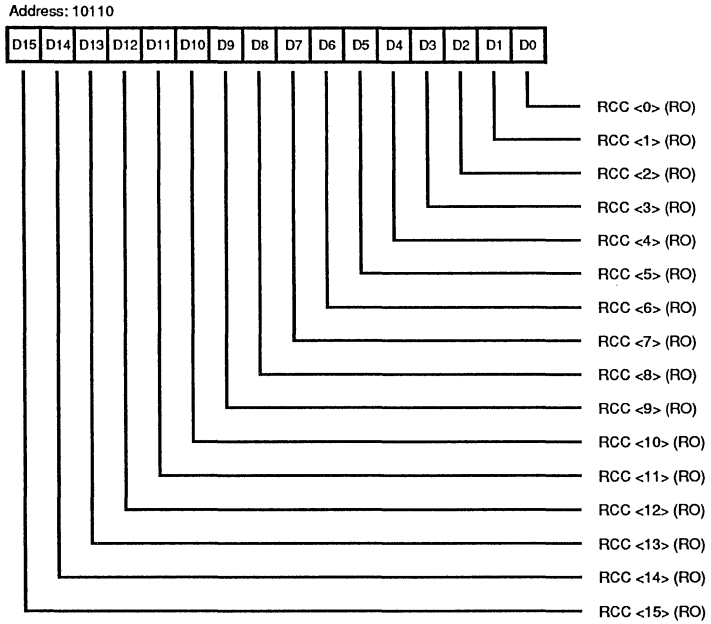


Figure 68. Receive Character Count Register (RCCR)



## CONTROL REGISTERS (Continued)

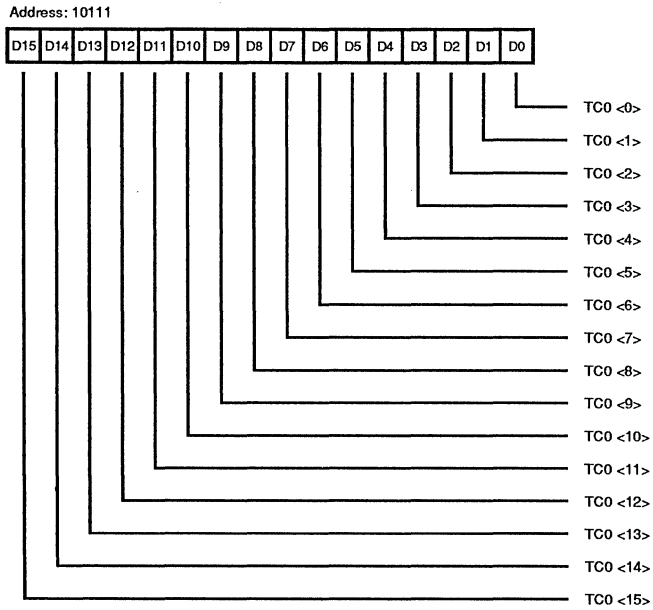


Figure 69. Time Constant 0 Register (TC0R)

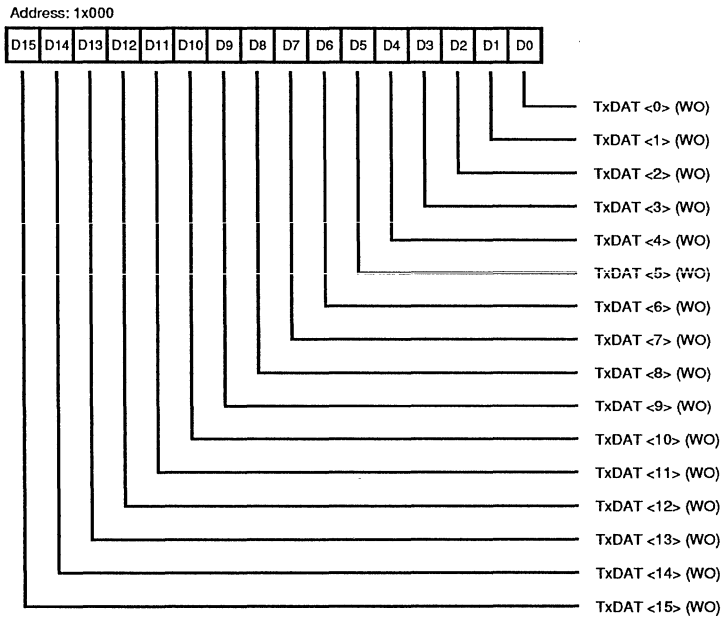


Figure 70. Transmit Data Register (TDR)



# CONTROL REGISTERS (Continued)

Address: 11010

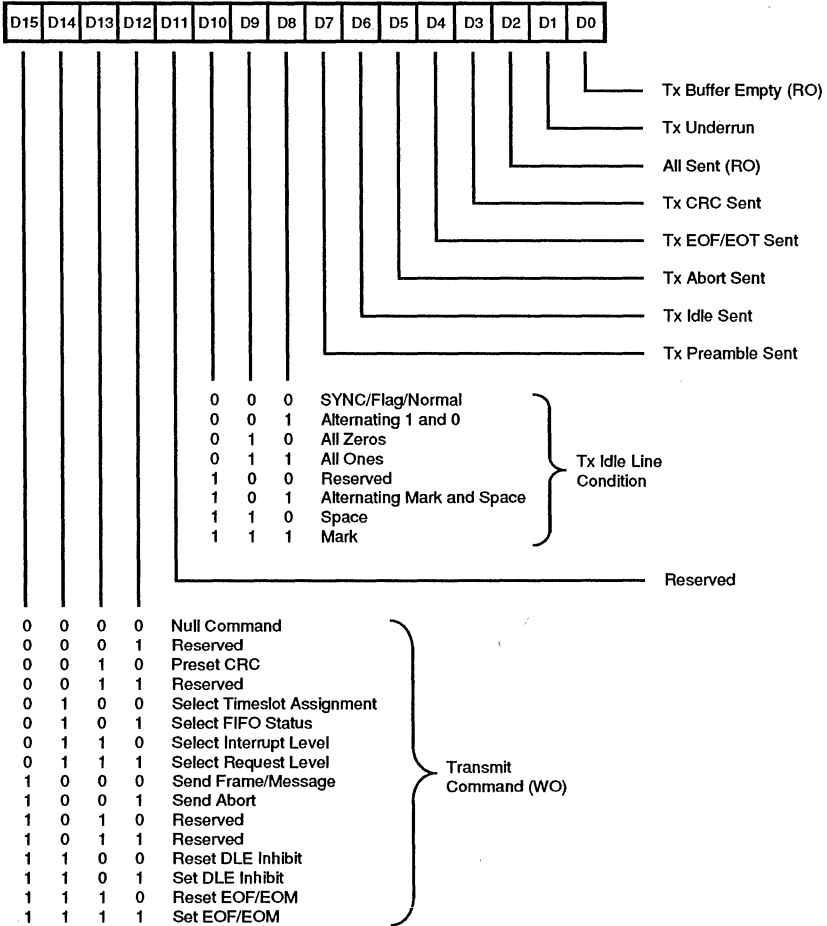


Figure 72. Transmit Command/Status Register (TCSR)

Address: 11011

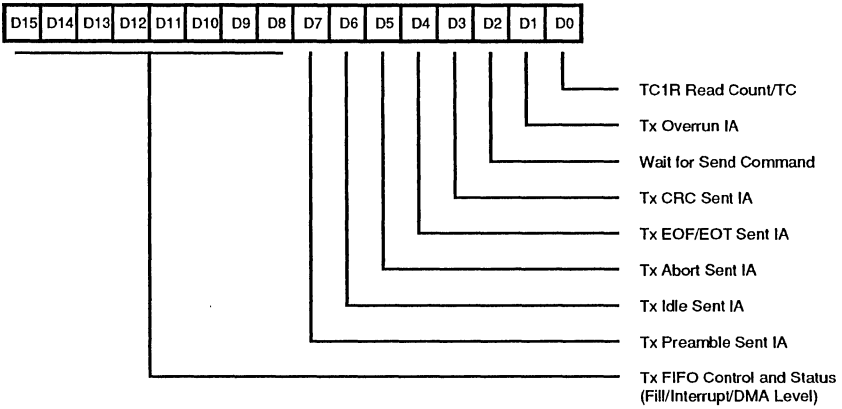


Figure 73a. Transmit Interrupt Control Register (TICR)

Address: 11011

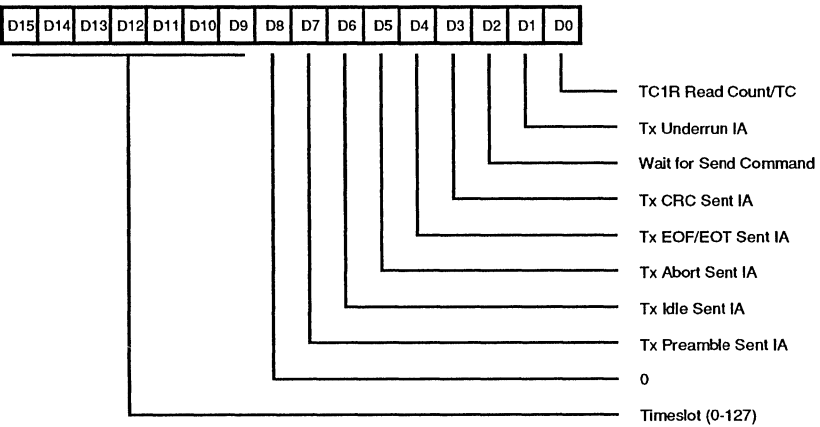


Figure 73b. Transmit Interrupt Control Register (TICR)



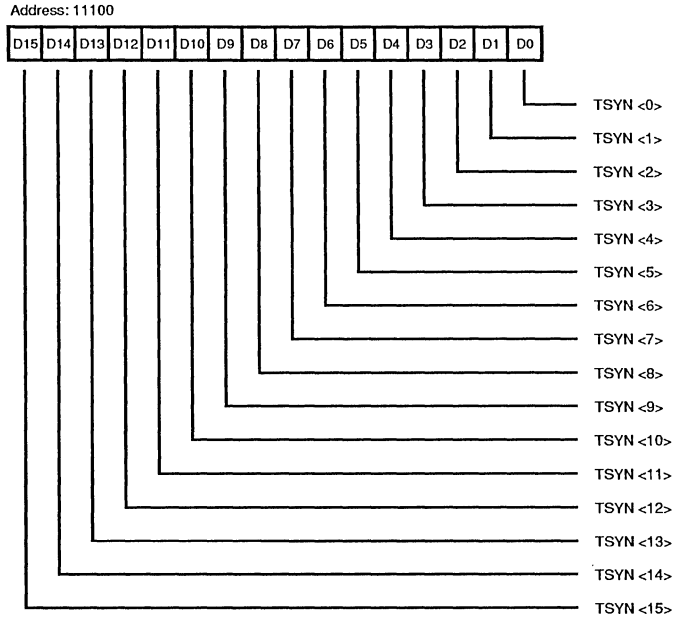


Figure 74. Transmit Sync Register (TSR)

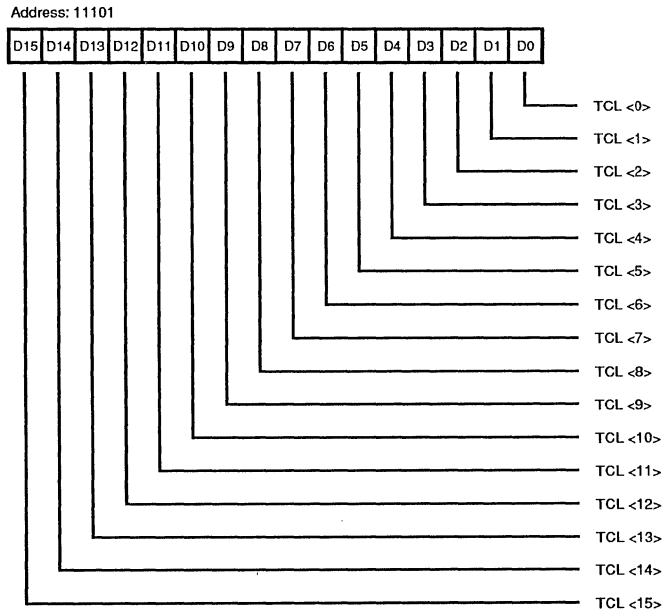


Figure 75. Transmit Count Limit Register (TCLR)

## CONTROL REGISTERS (Continued)

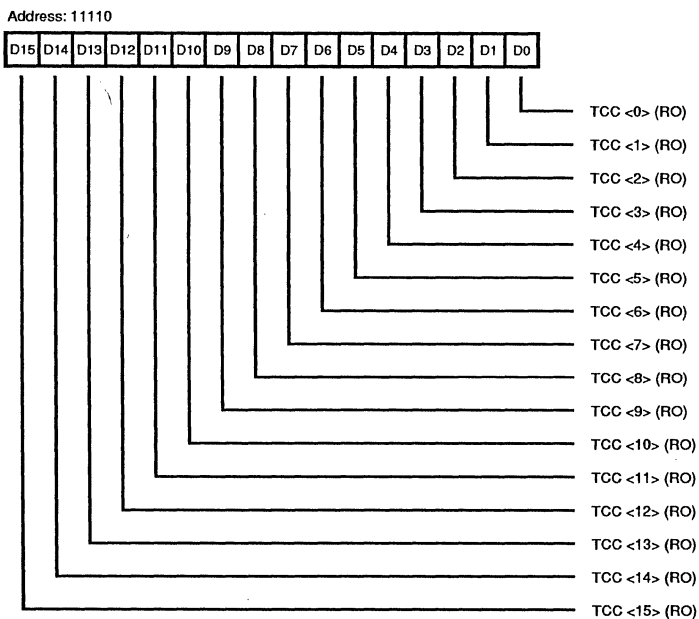


Figure 76. Transmit Character Count Register (TCCR)

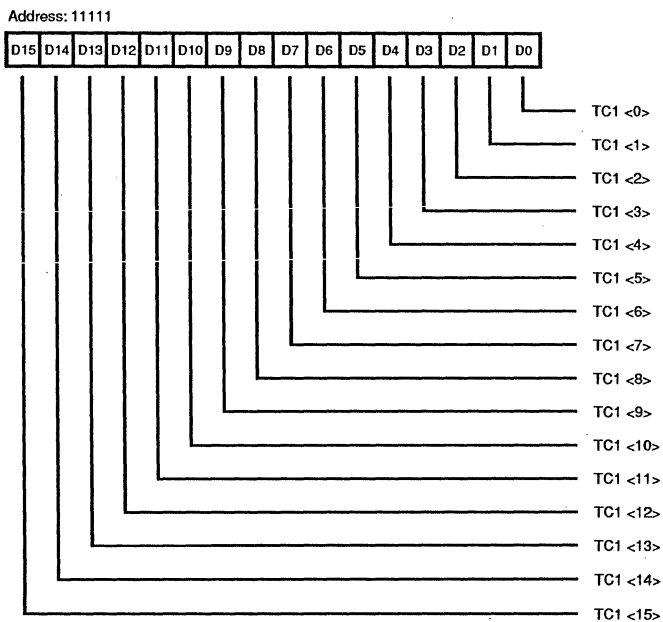
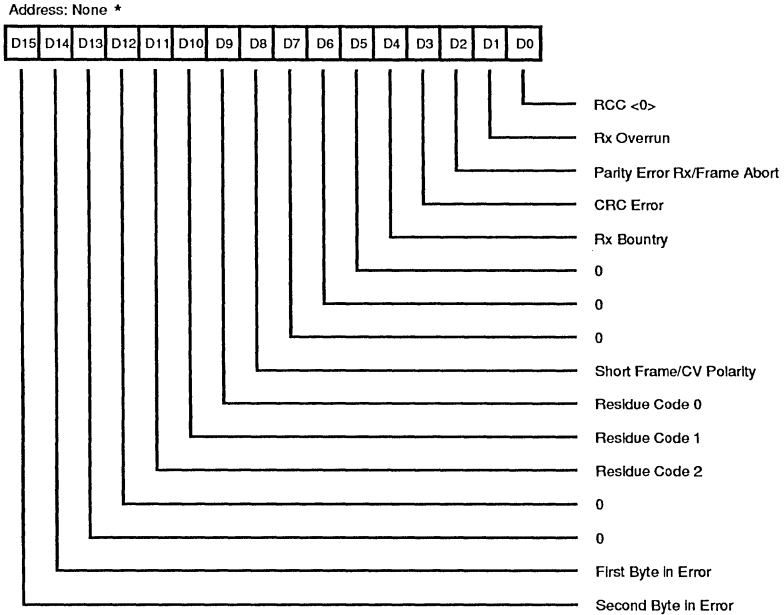
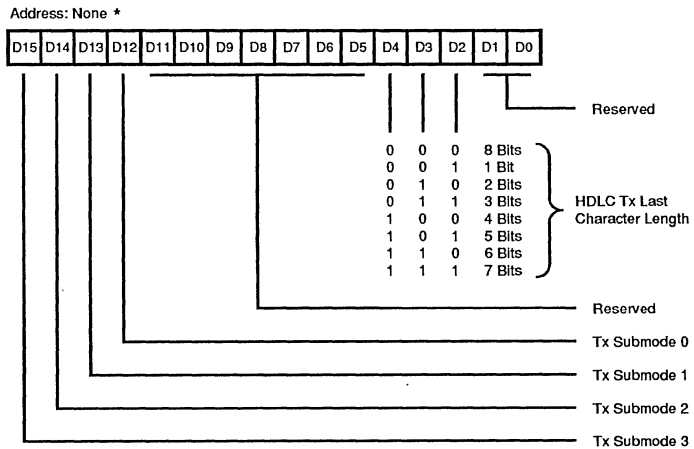


Figure 77. Time Constant 1 Register (TC1R)



\* Refer to Figure 22 (Channel Control Register)  
Bits 6-7 for Access Method

Figure 78. Receive Status Block Register (RSBR)

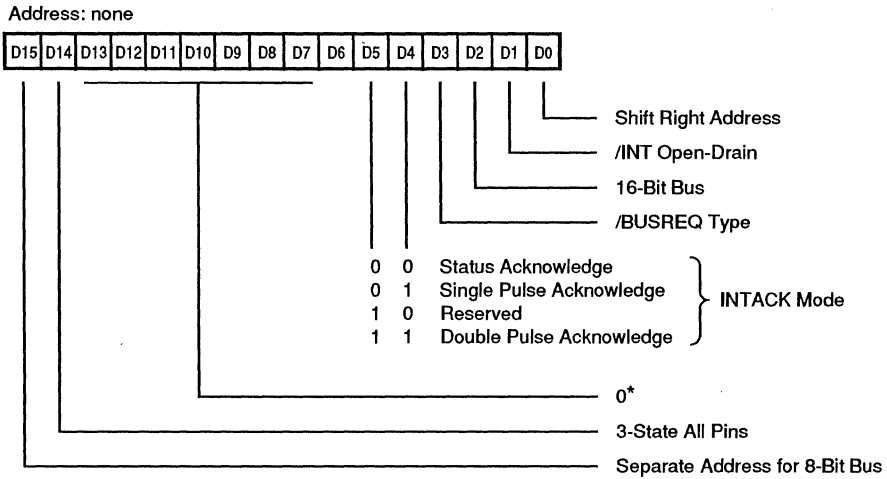


\* Refer to Figure 22 (Channel Control Register)  
Bits 15-14 for Access Method

Figure 79. Transmit Status Block Register (TSBR)



## CONTROL REGISTERS (Continued)



\* Must be programmed as 0

**Figure 80. Bus Configuration Register (BCR)**

## IUSC TIMING

The IUSC interface timing is similar to that found on a static RAM, except that it is much more flexible. Up to four separate timing strobe signals are present on the interface: /DS, /RD, /WR and /INTACK. Only one of these timing strobes is active at any time. Should the external logic activate more than one of these strobes at the same time,

the IUSC will enter a pre-reset state. This state is only exited by a hardware reset. Do not allow overlap of timing strobes. The timing diagrams, beginning on the next page, illustrate the different bus transactions possible with the necessary setup, hold, and delay times. IUSC Timing diagrams are shown from Figure 82 through Figure 106.

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins with respect to GND .....	-0.3V to +7.0 V
Voltages on all inputs with respect to GND .....	-0.3V to $V_{CC}+0.3V$
Operating Ambient Temperature .....	See Ordering Information
Storage Temperature .....	-65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance Section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to Ground. Positive current flows into the referenced pin. Standard conditions are as follows:

- $+4.5\text{ V} < V_{CC} < +5.5\text{ V}$
- $GND = 0\text{ V}$
- $T_A$  as specified in Ordering Information

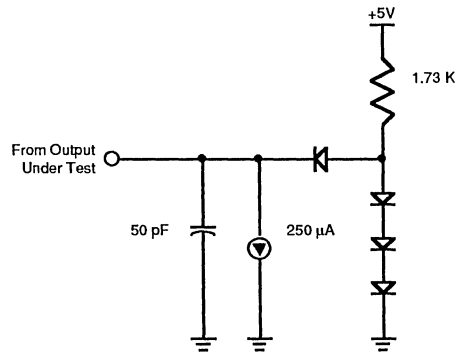


Figure 81. Standard Test Load

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit	Condition
$C_{IN}$	Input Capacitance		10	pF	*
$C_{OUT}$	Output Capacitance		15	pF	*
$C_{VO}$	Bidirectional Capacitance		20	pF	

### Notes:

F = 1MHz, over specified temperature range.

\* Unmeasured pins returned to Ground.

## MISCELLANEOUS

Transistor Count - 100,000

## DC CHARACTERISTICS

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC} + 0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6mA$
$V_{OH2}$	Output High Voltage	$V_{CC} - 0.8$			V	$I_{OH} = -250 \mu A$
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = +2.0 mA$
$I_{IL}$	Input Leakage			+10.00	$\mu A$	$0.4 < V_{IN} < +2.4V$
$I_{OL}$	Output Leakage			+10.00	$\mu A$	$0.4 < V_{OUT} < +2.4V$
$I_{CC1}$	$V_{CC}$ Supply Current		7	50	mA	$V_{CC} = 5V$ $V_{IH} = 4.8V$ $V_{IL} = 0.2V$

**Note:**

$V_{CC} = 5V \pm 10\%$  unless otherwise specified, over specified temperature range.

## AC CHARACTERISTICS

Timing Diagrams (Figures 82-104)

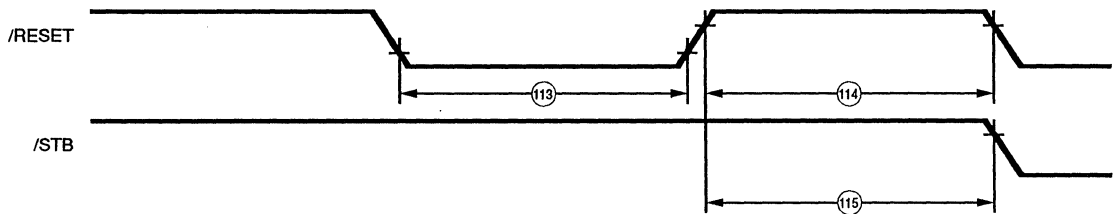
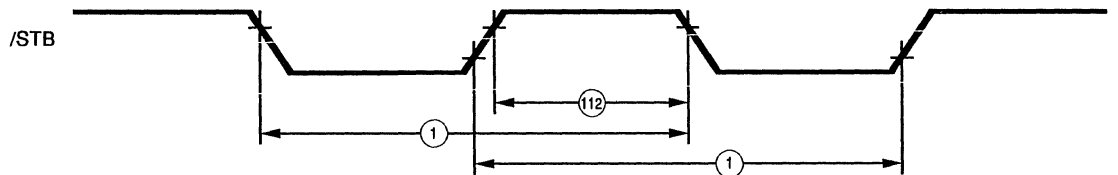


Figure 82. Reset Timing



**Note:**

$/STB$  is any of the following:  $/DS$ ,  $/RD$ ,  $/WR$  or Pulsed  $/INTACK$ .

Figure 83. Bus Cycle Timing

# AC CHARACTERISTICS

## Timing Diagrams (Continued)

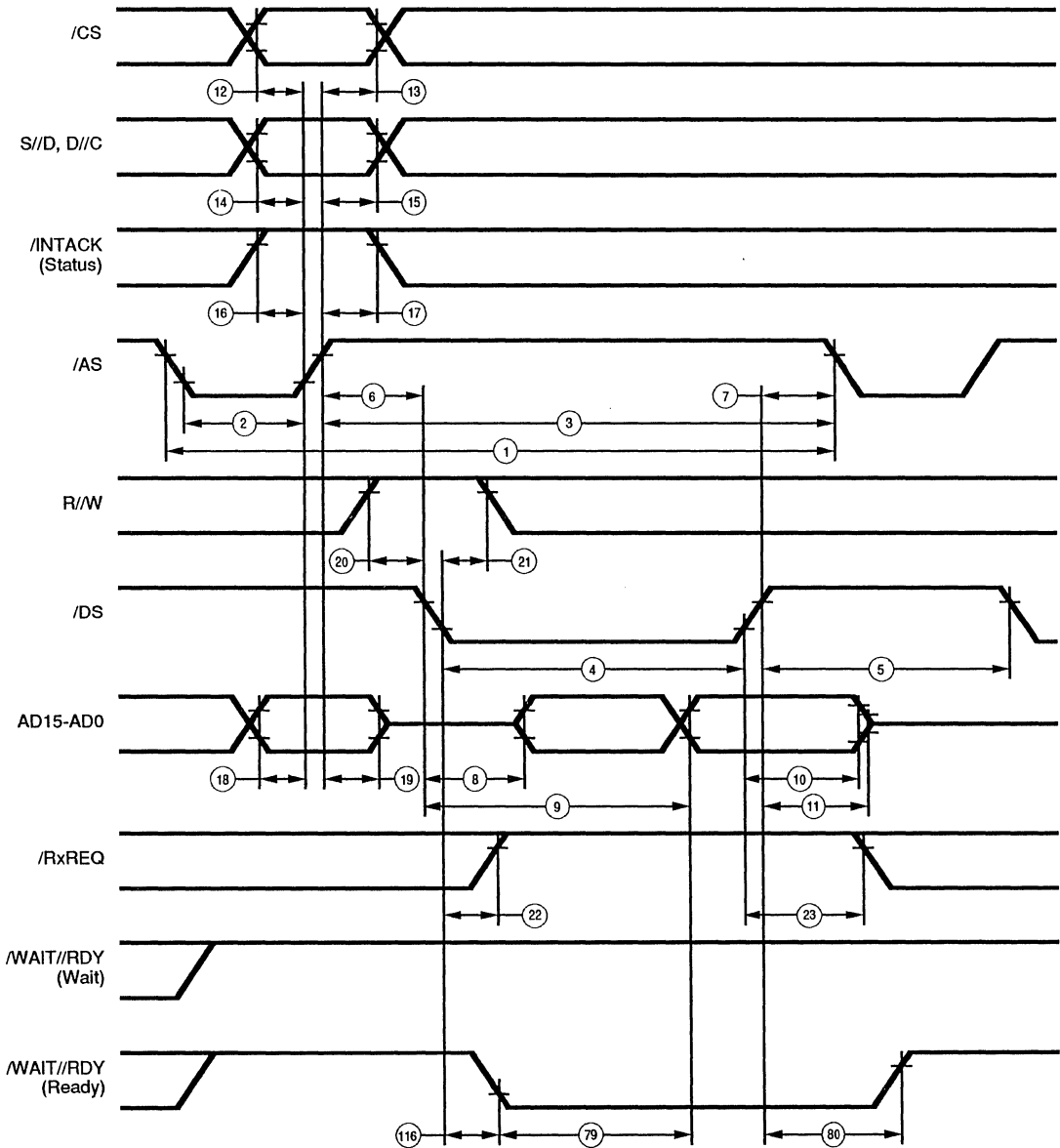


Figure 84. Multiplexed /DS Read Cycle

**AC CHARACTERISTICS**  
Timing Diagrams (Continued)

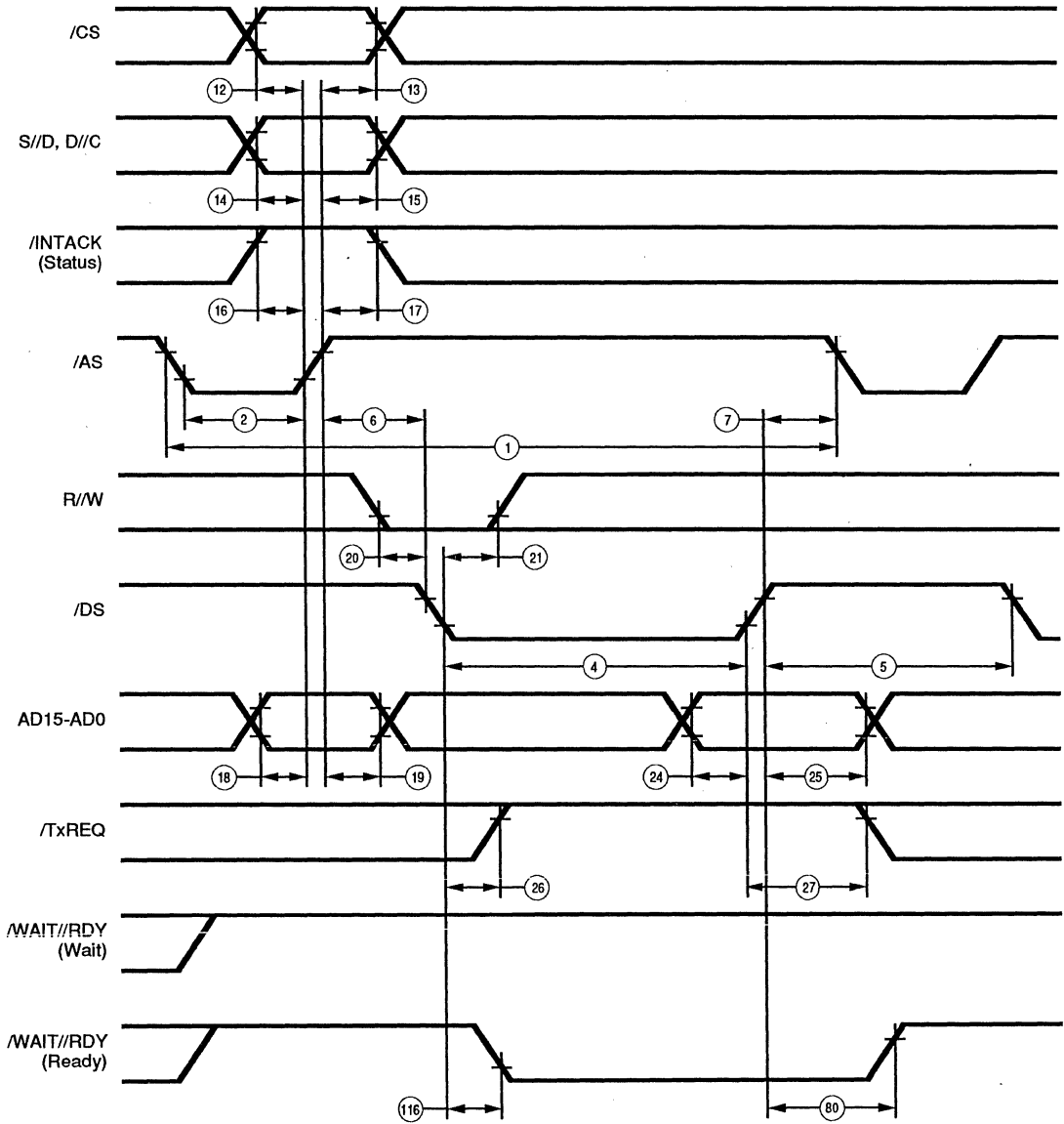


Figure 85. Multiplexed /DS Write Cycle

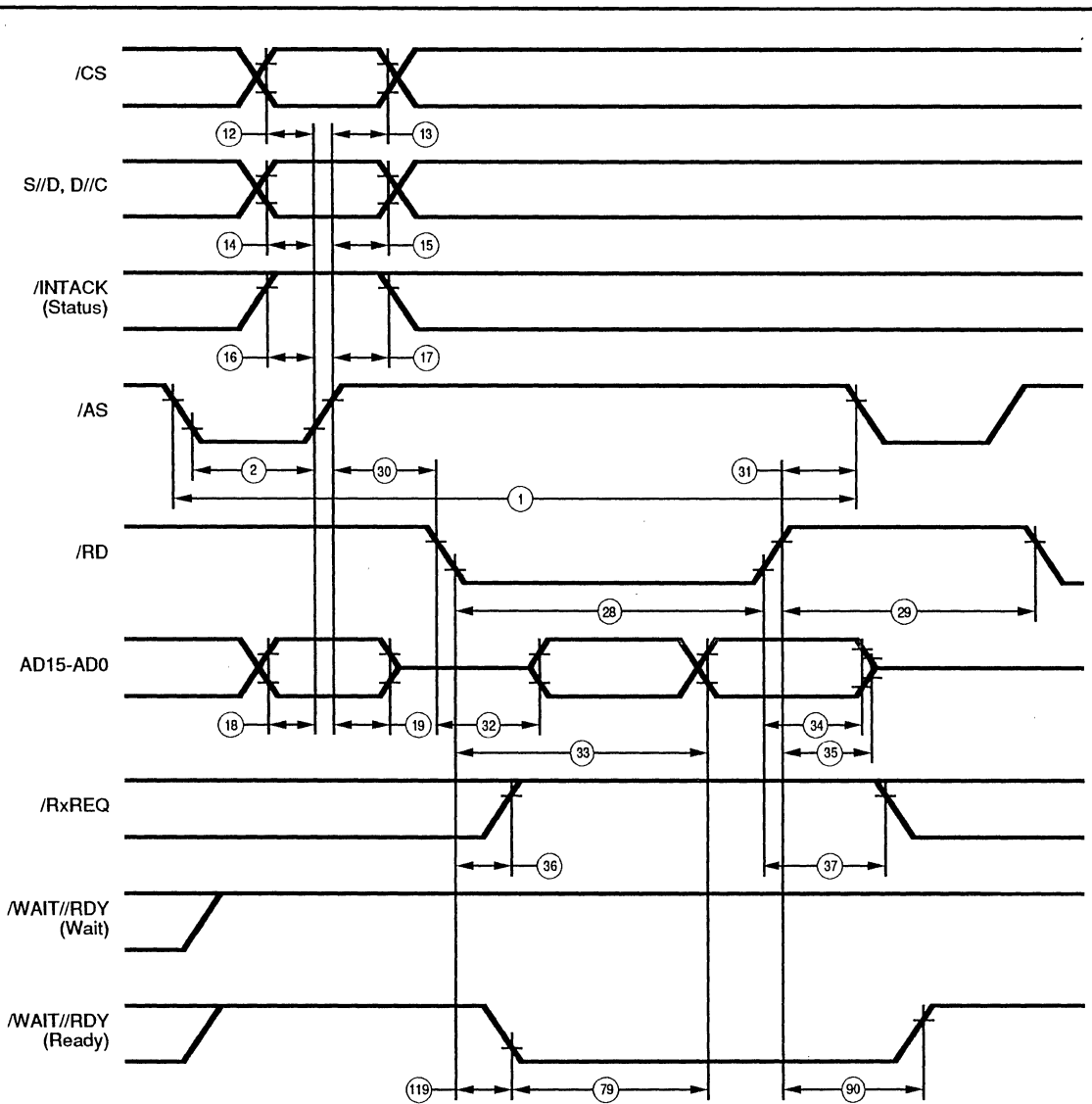


Figure 86. Multiplexed /RD Read Cycle

**AC CHARACTERISTICS**  
Timing Diagrams (Continued)

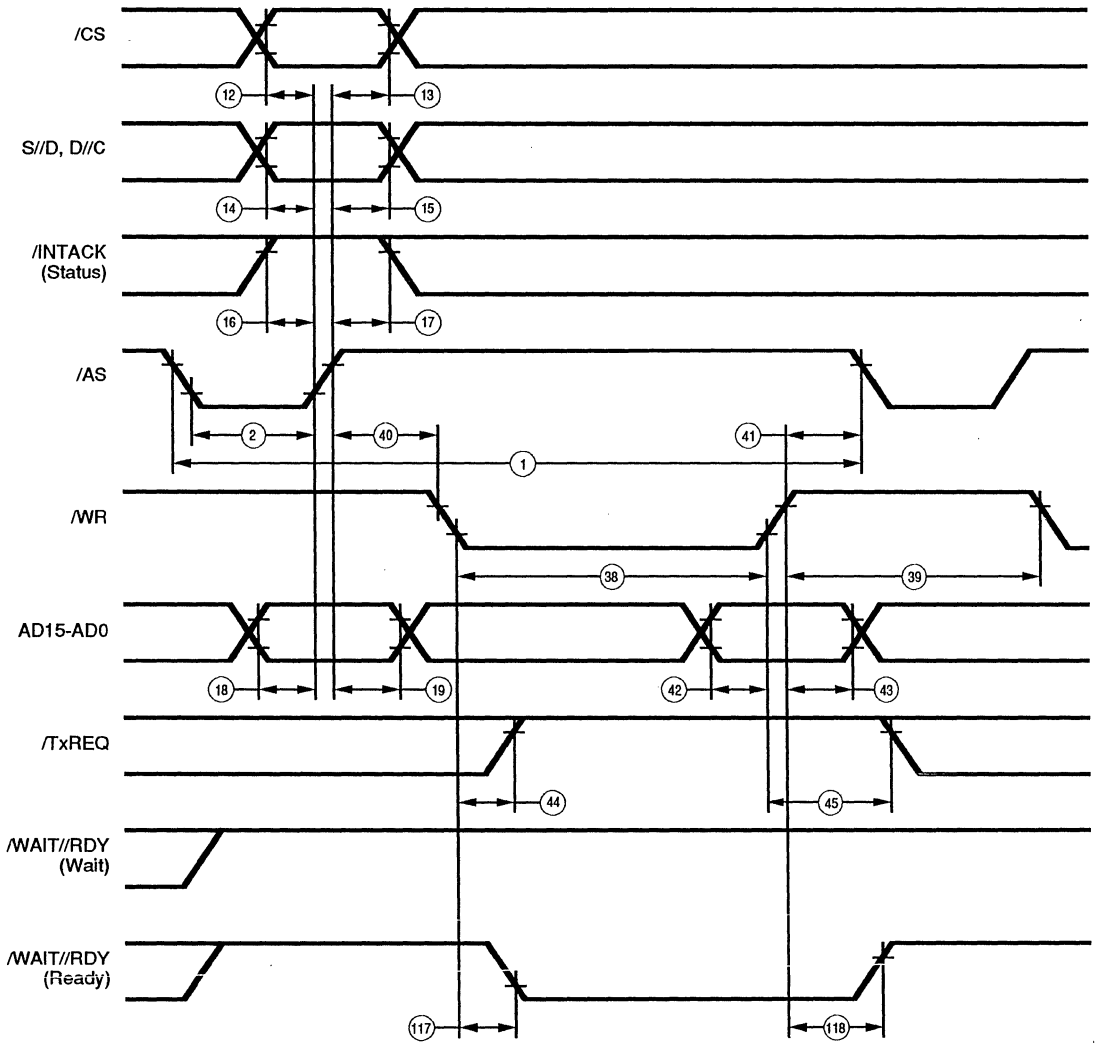


Figure 87. Multiplexed /WR Write Cycle

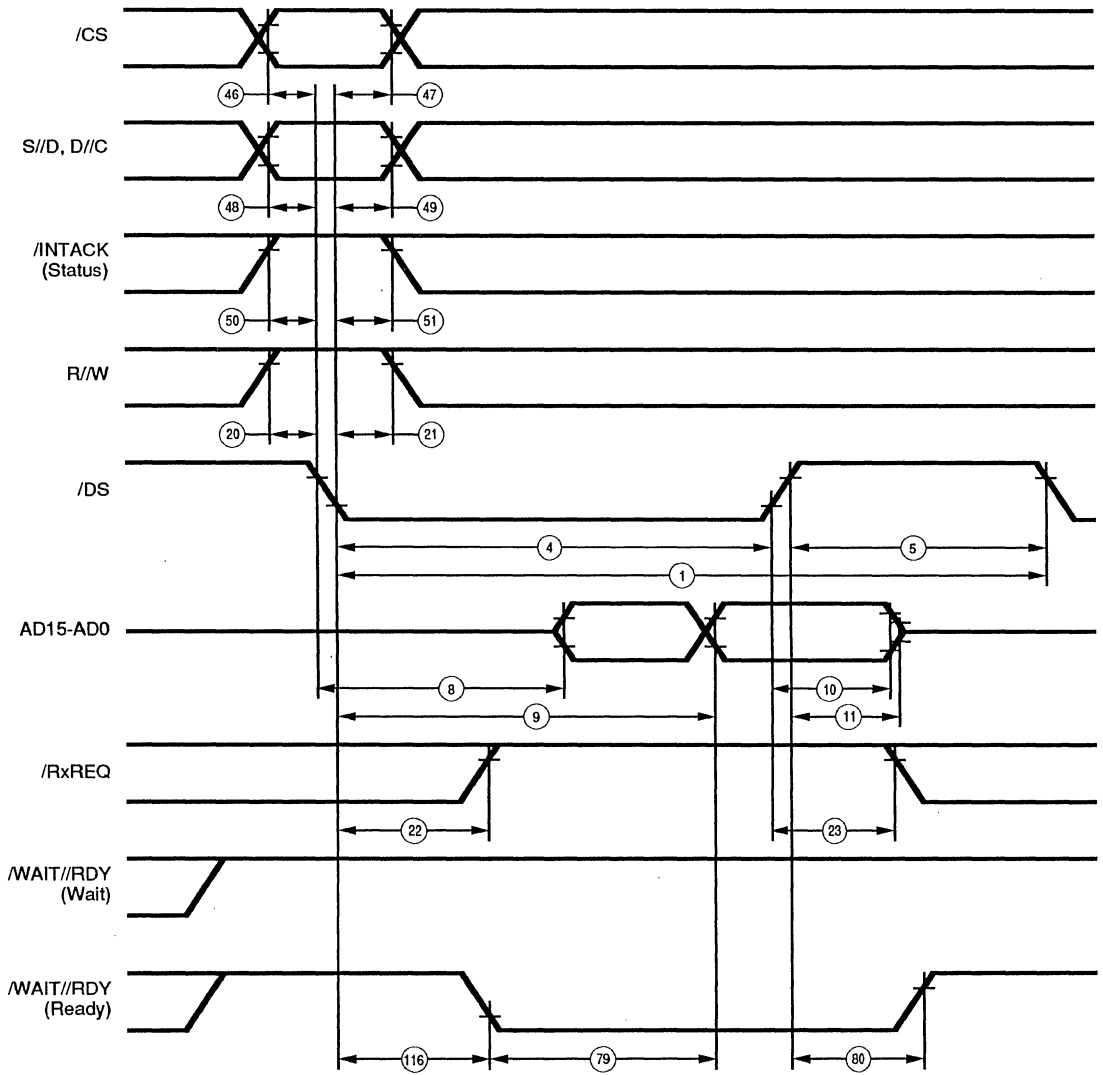


Figure 88. Non-Multiplexed /DS Read Cycle



# AC CHARACTERISTICS

## Timing Diagrams (Continued)

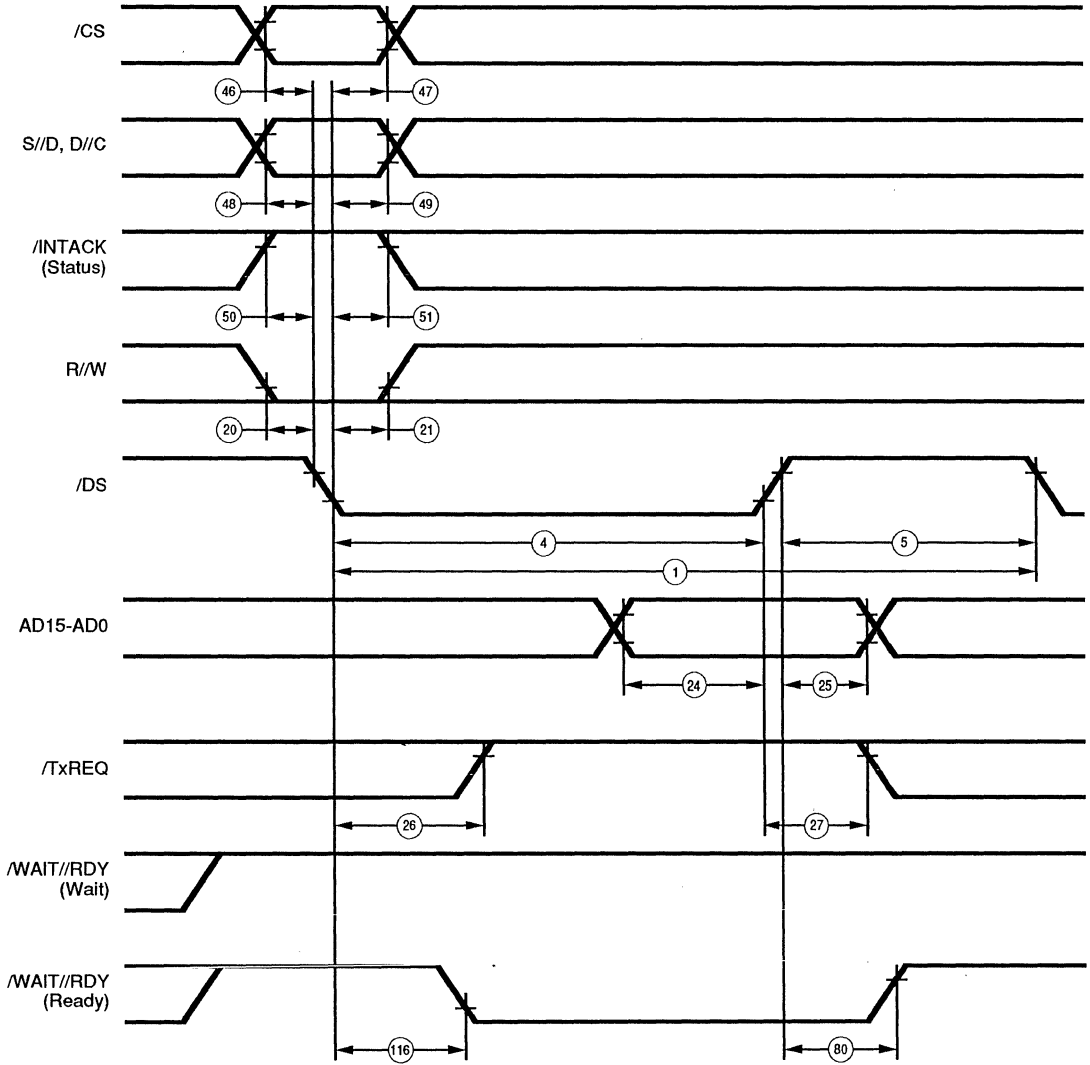


Figure 89. Non-Multiplexed /DS Write Cycle

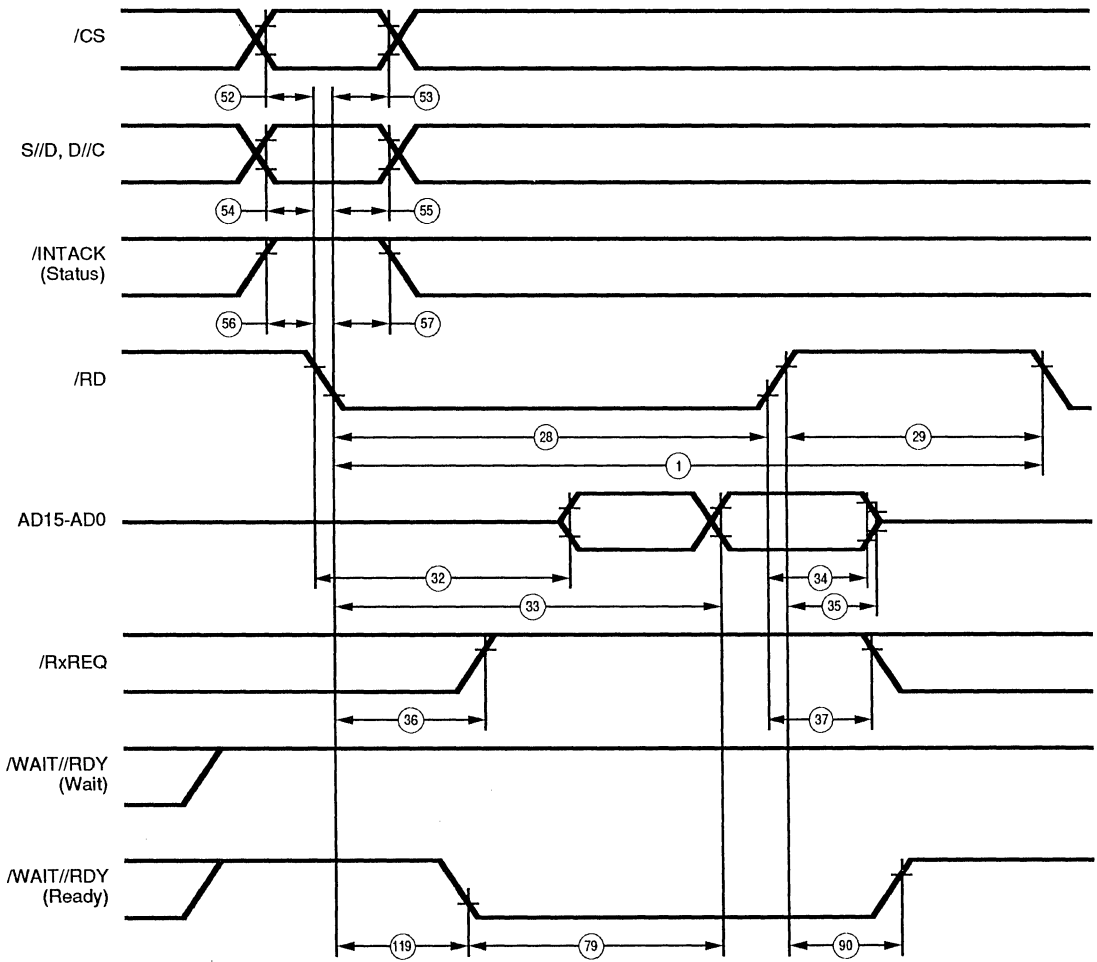


Figure 90. Non-Multiplexed /RD Read Cycle

# AC CHARACTERISTICS

## Timing Diagrams (Continued)

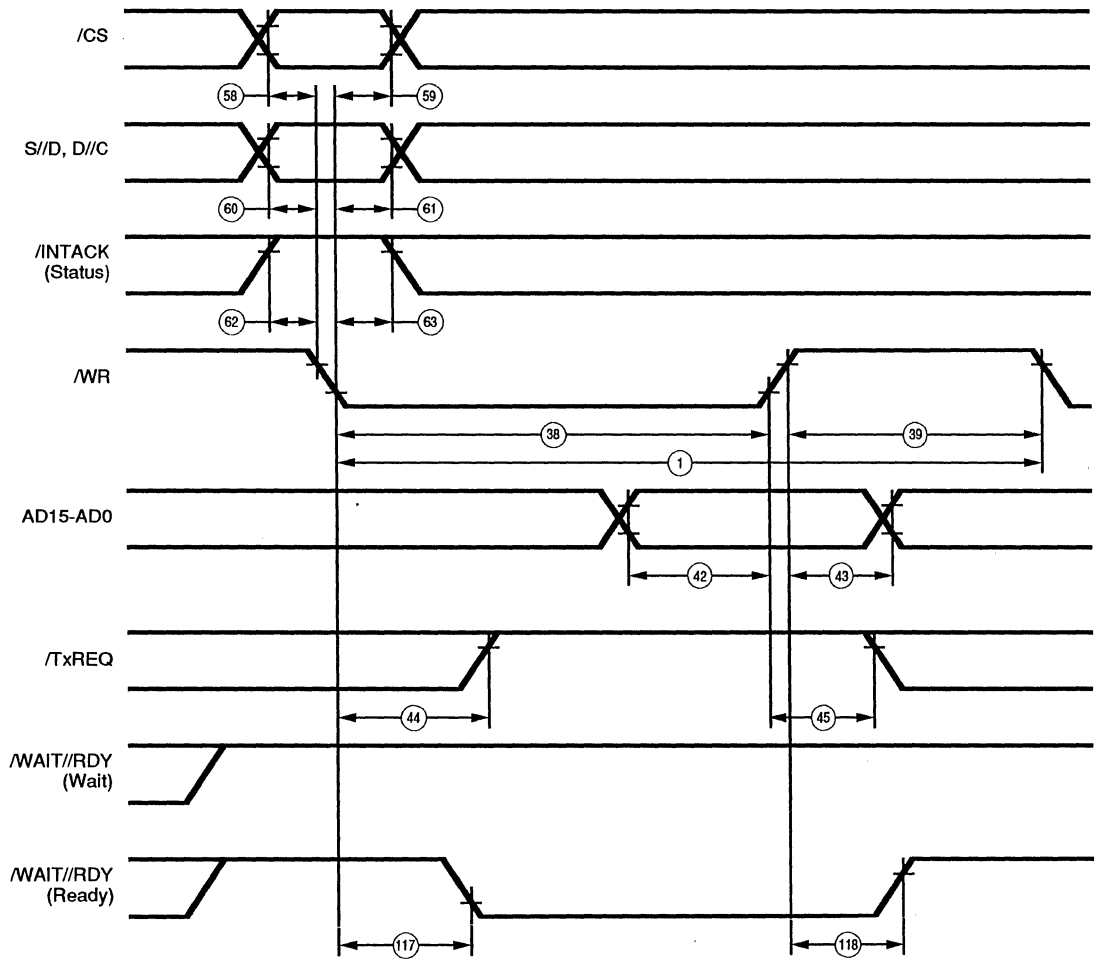


Figure 91. Non-Multiplexed /WR Write Cycle

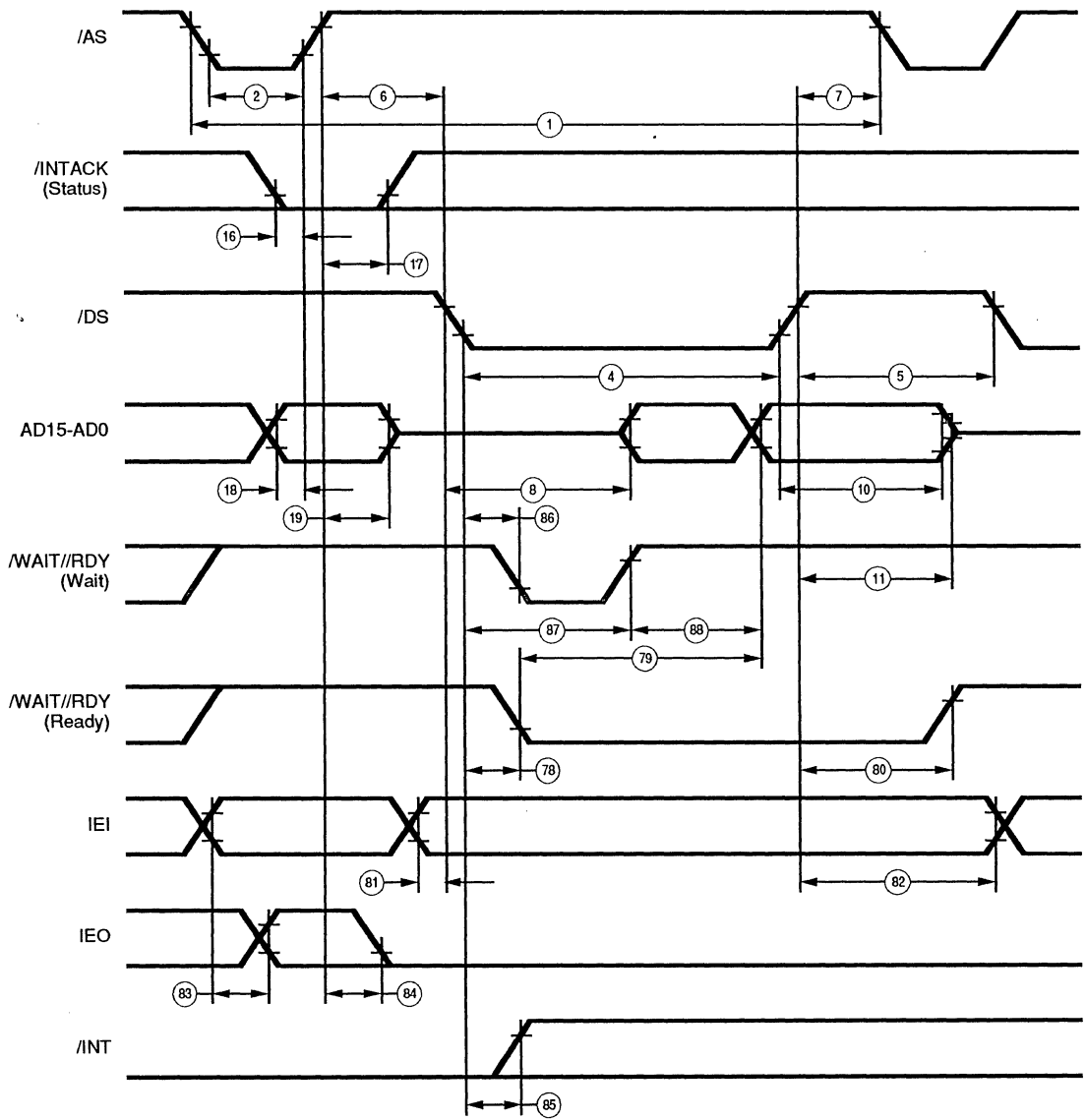


Figure 92. Multiplexed /DS Interrupt Acknowledge Cycle

# AC CHARACTERISTICS

## Timing Diagrams (Continued)

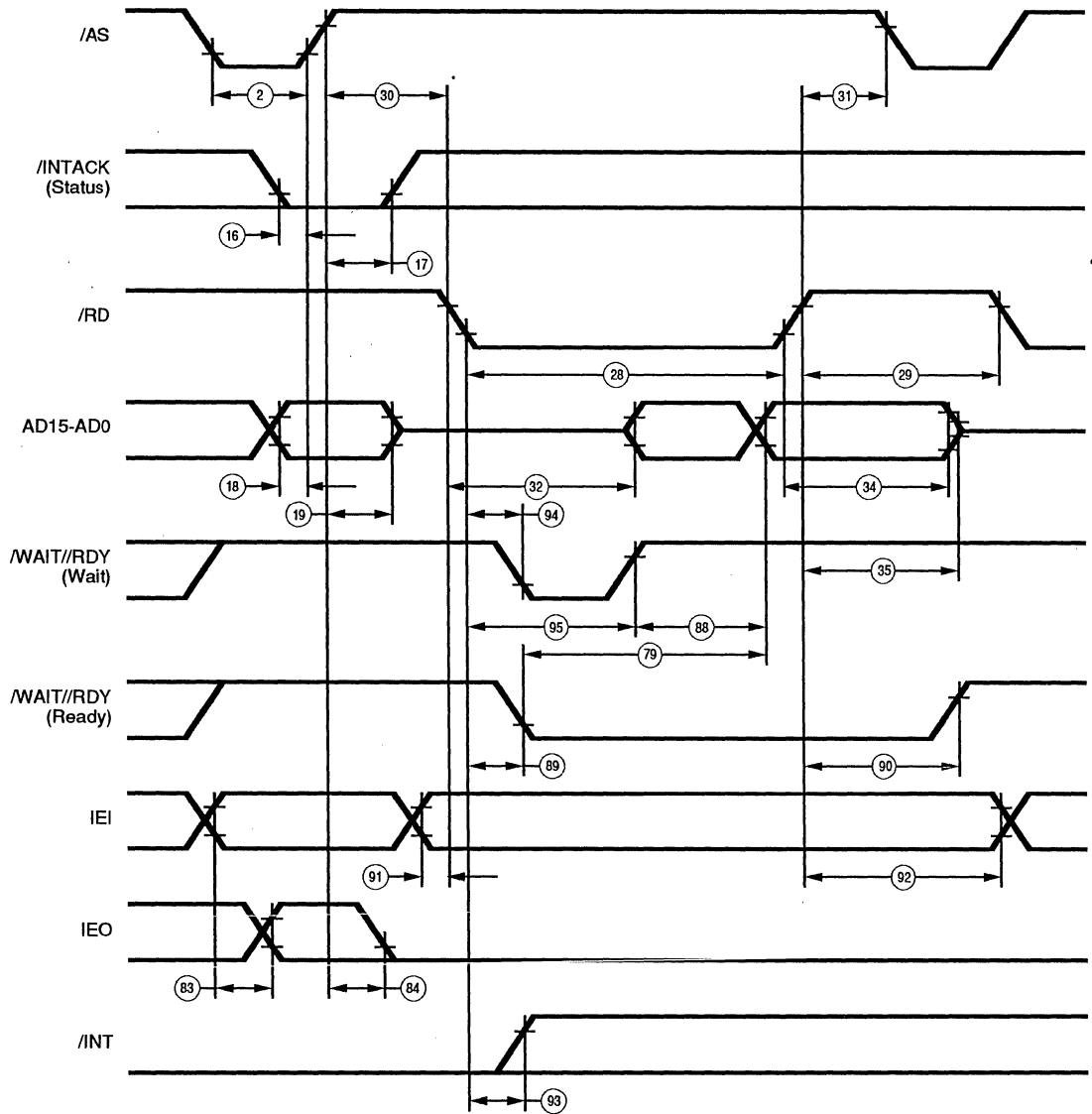


Figure 93. Multiplexed /RD Interrupt Acknowledge Cycle

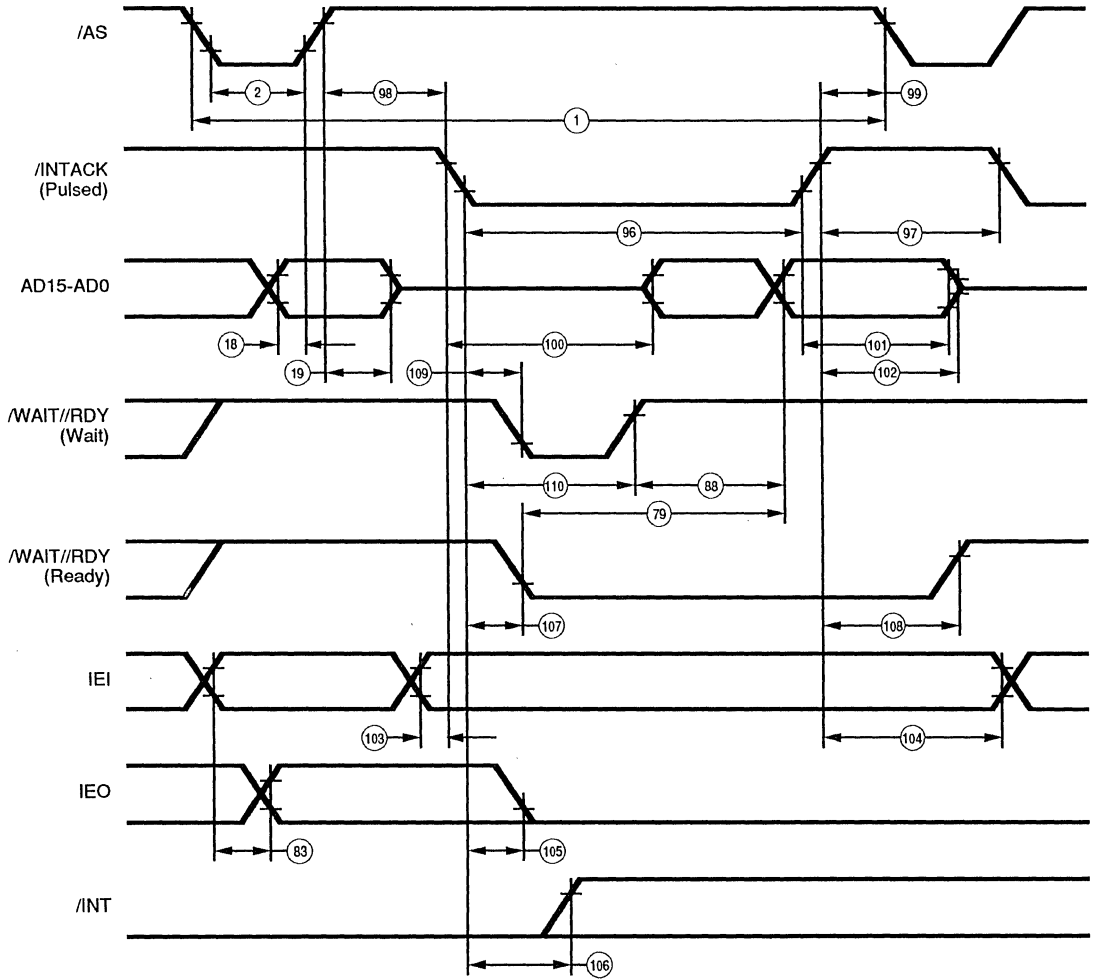


Figure 94. Multiplexed Pulsed Interrupt Acknowledge Cycle

**AC CHARACTERISTICS**  
Timing Diagrams (Continued)

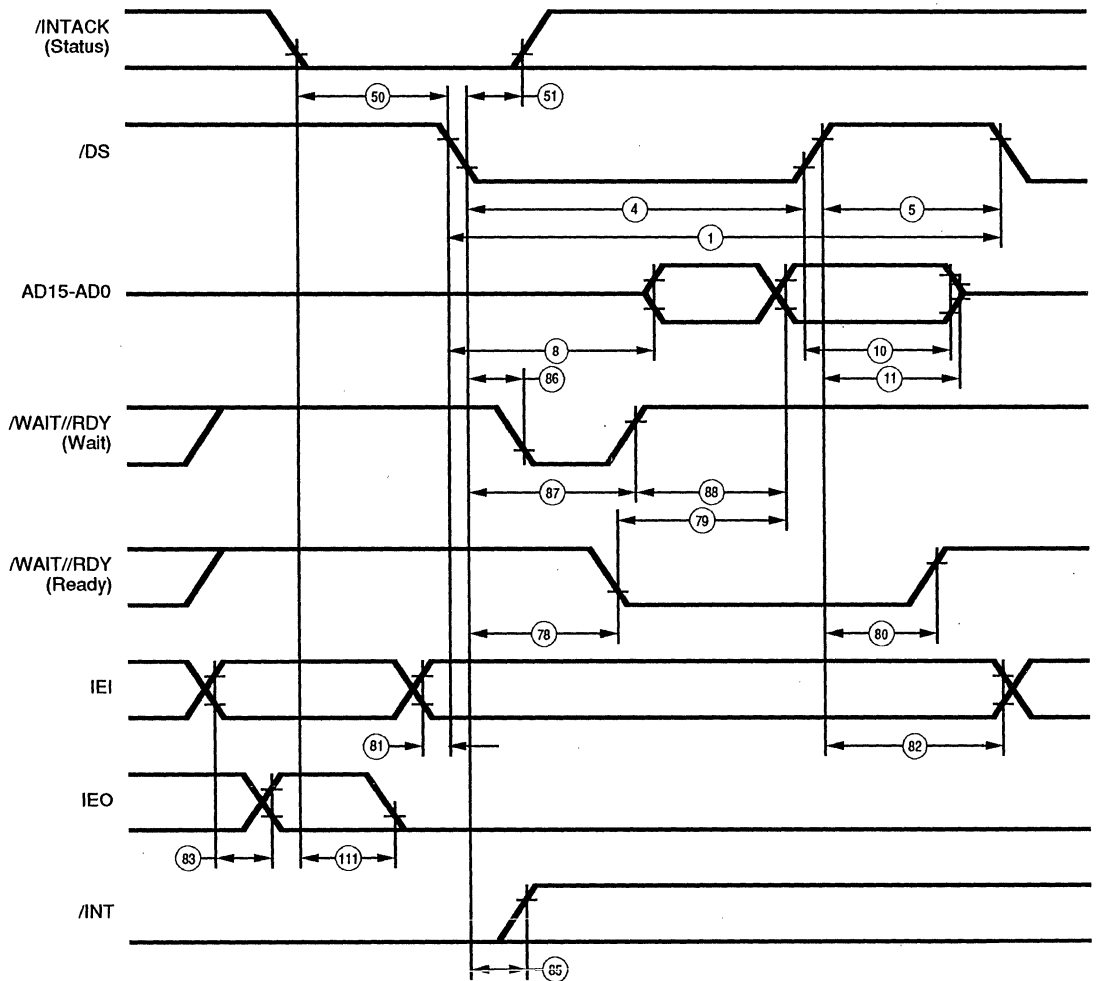


Figure 95. Non-Multiplexed /DS Interrupt Acknowledge Cycle

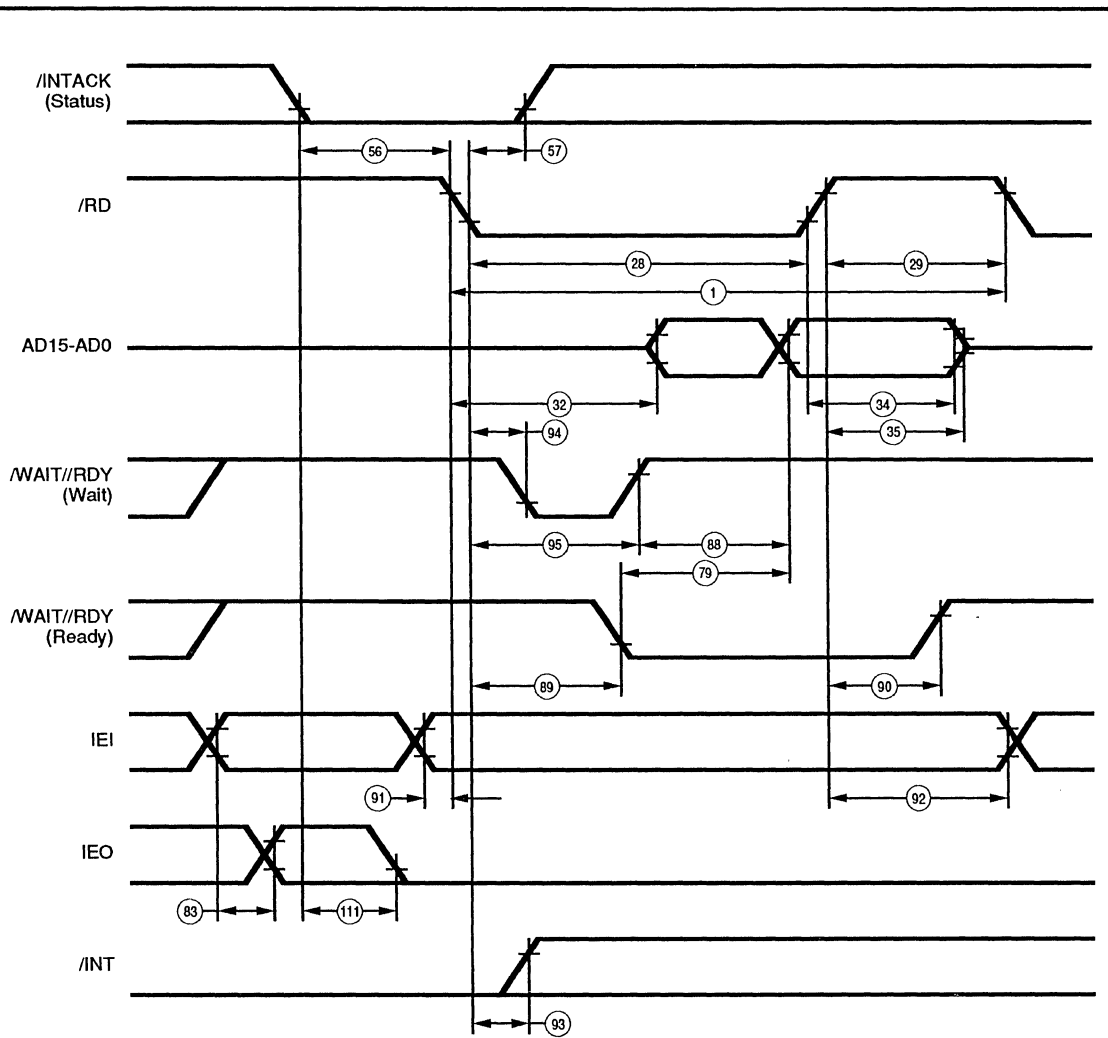


Figure 96. Non-Multiplexed /RD Pulsed Interrupt Acknowledge Cycle



# AC CHARACTERISTICS

## Timing Diagrams (Continued)

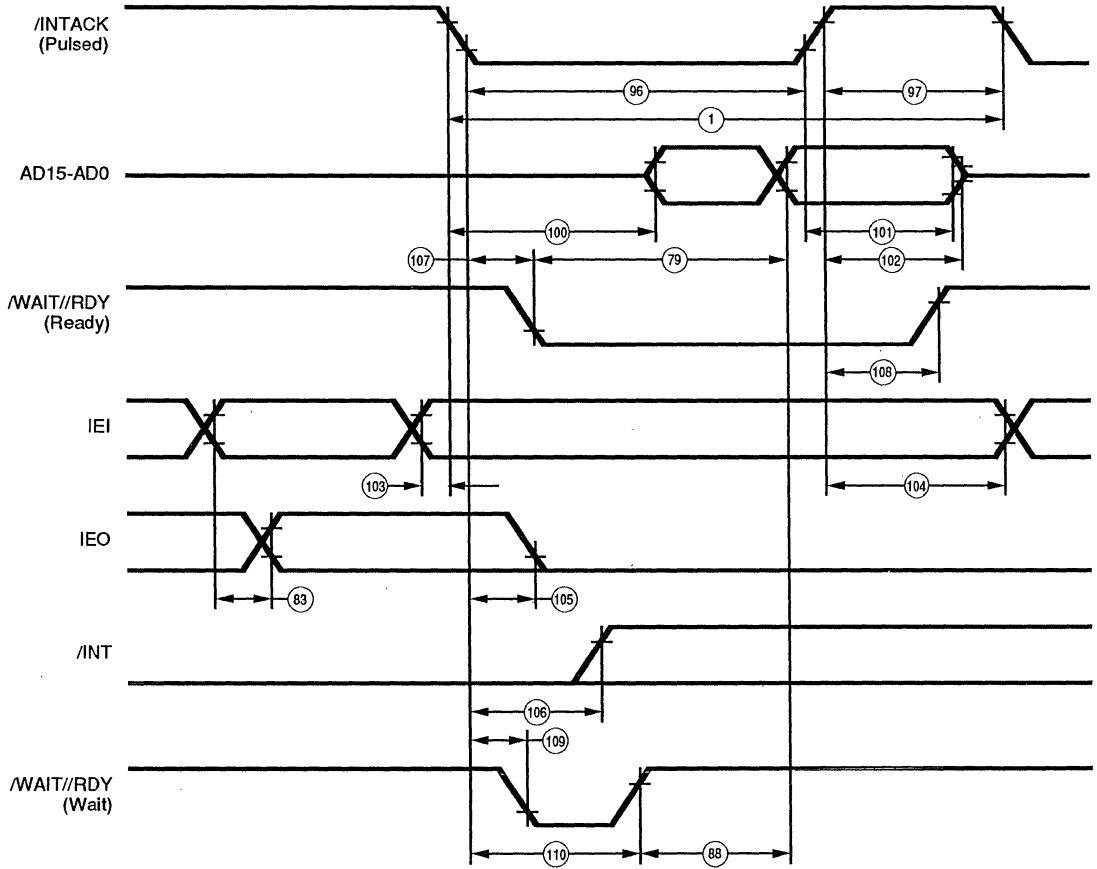


Figure 97. Non-Multiplexed Pulsed Interrupt Acknowledge Cycle

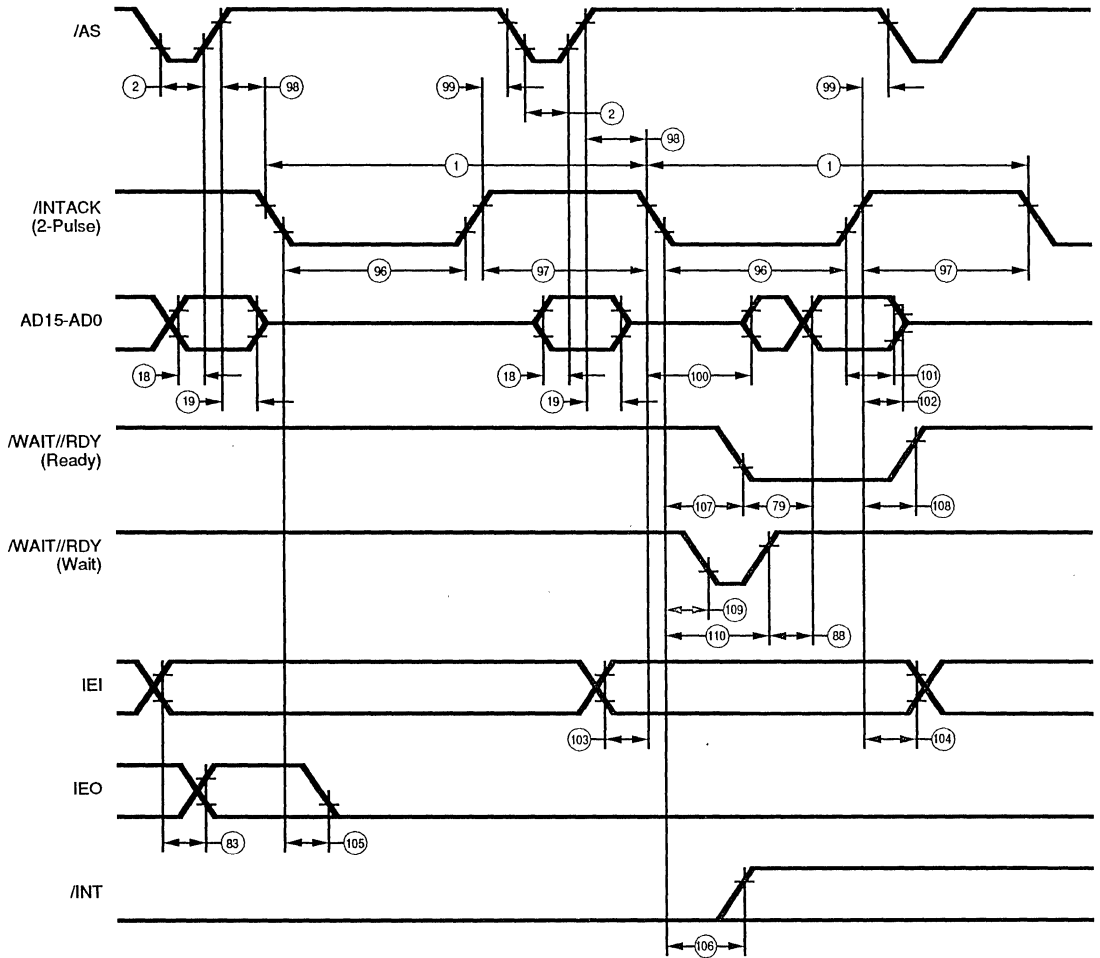


Figure 98. Multiplexed Double-Pulse Intack Cycle

**AC CHARACTERISTICS**  
Timing Diagrams (Continued)

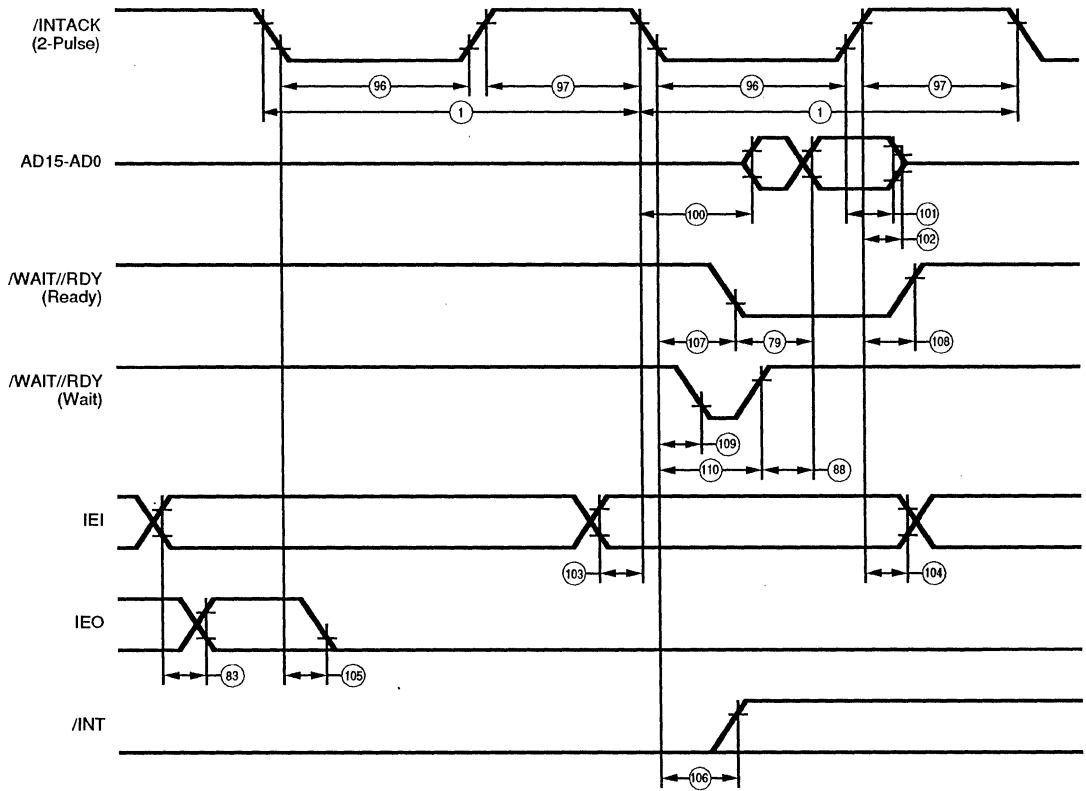


Figure 99. Non-Multiplexed Double-Pulse Intack Cycle

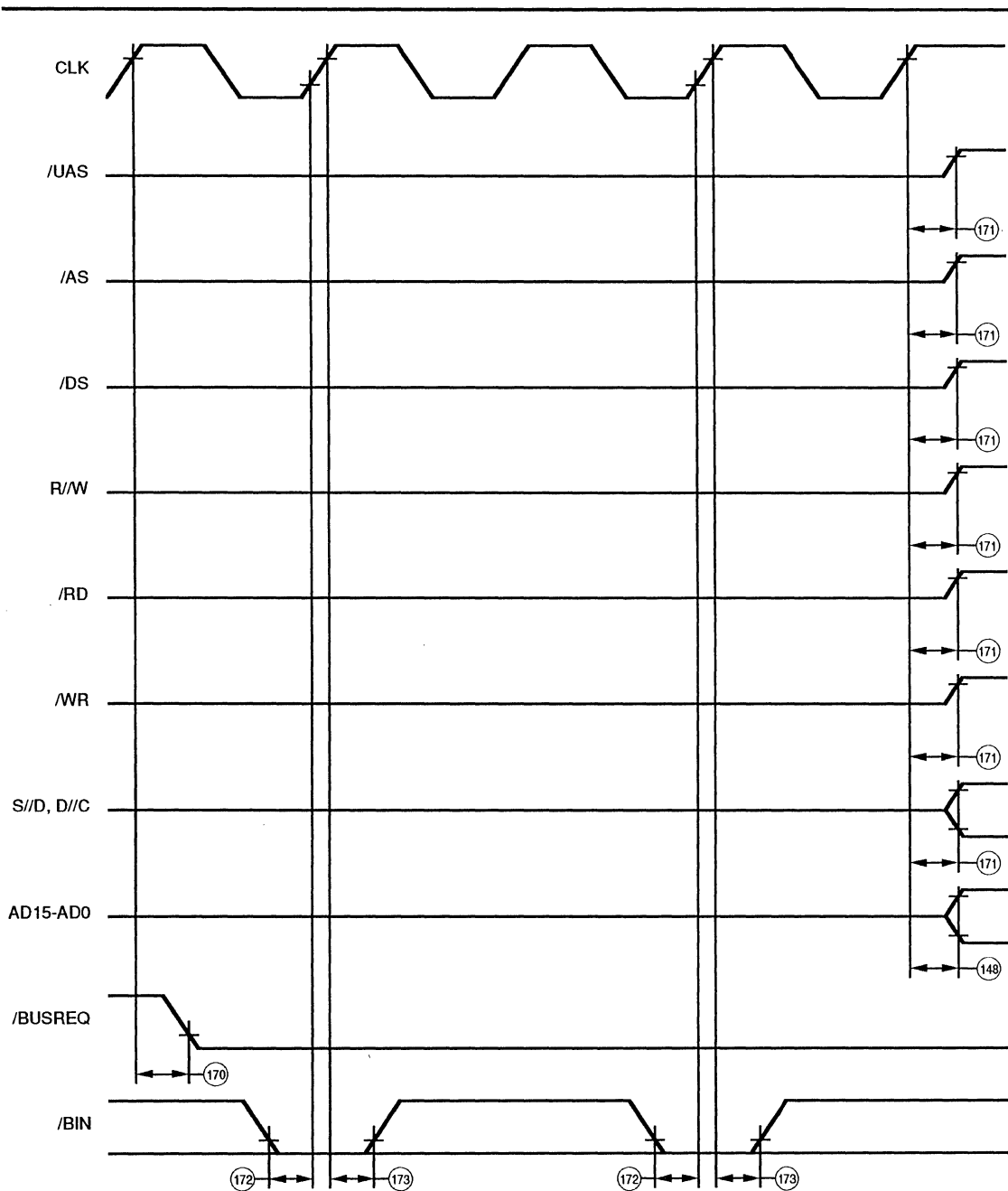


Figure 100. DMA Start-up

**AC CHARACTERISTICS**  
 Timing Diagrams (Continued)

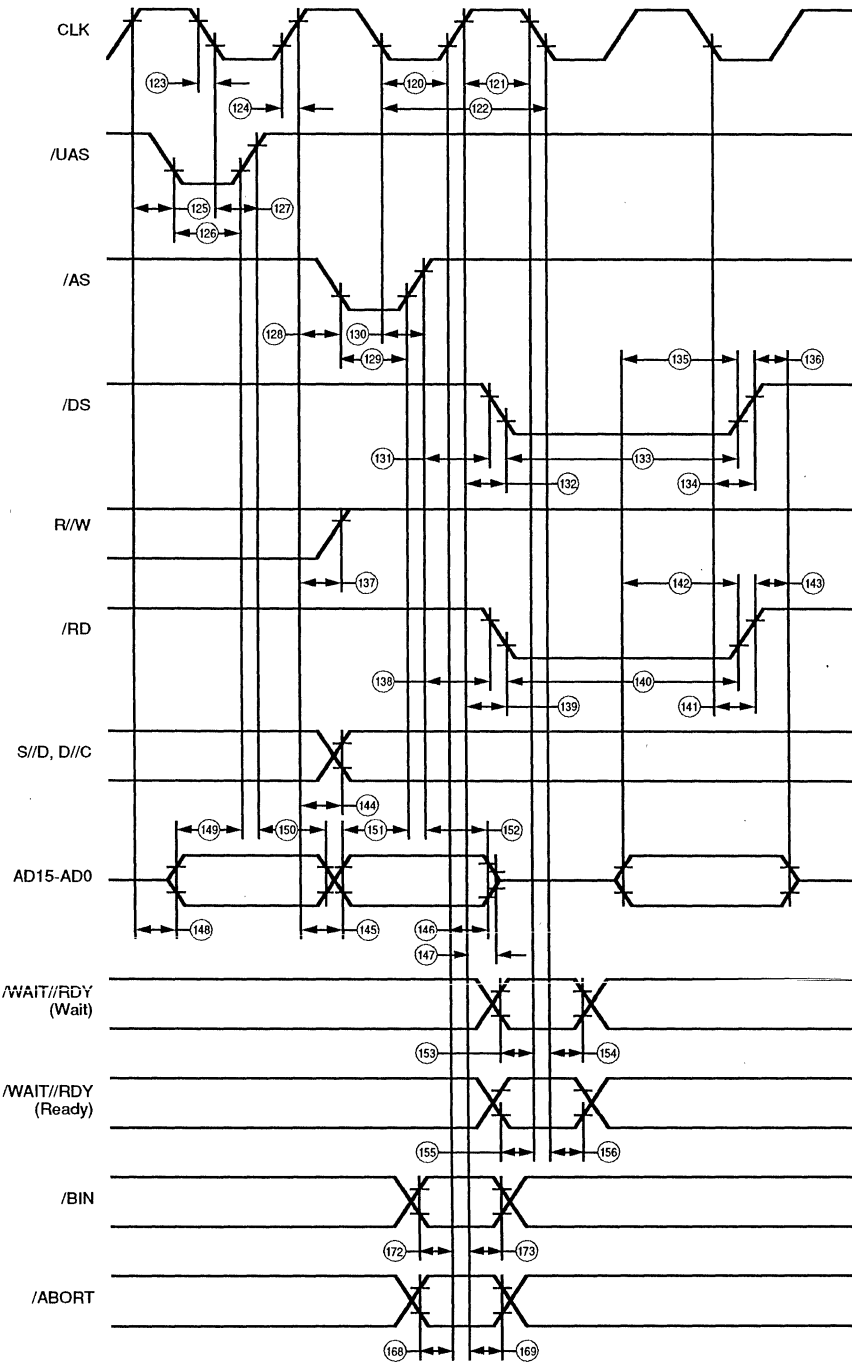


Figure 101. Memory Read

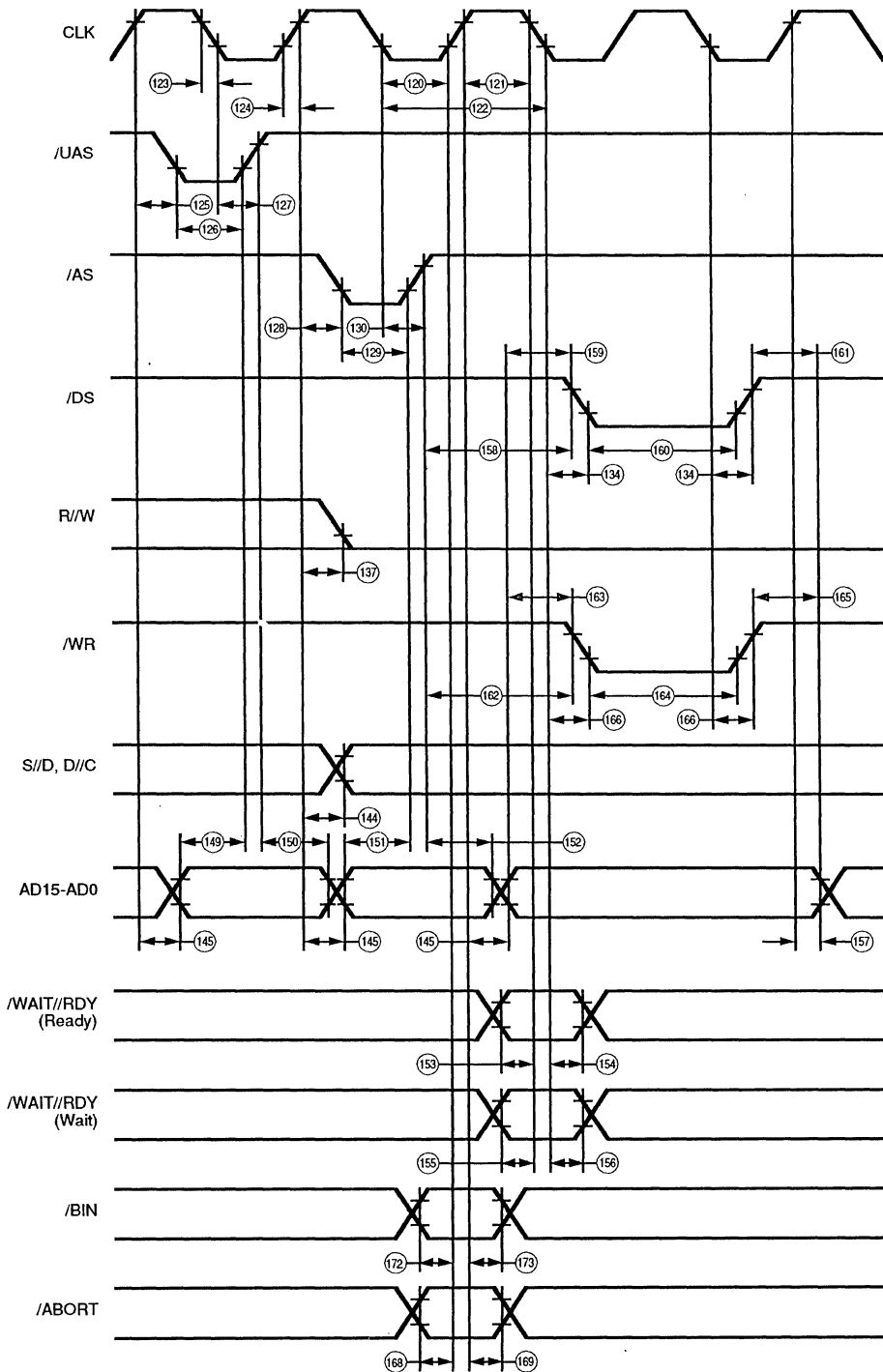


Figure 102. Memory Write

**AC CHARACTERISTICS**  
Timing Diagrams (Continued)

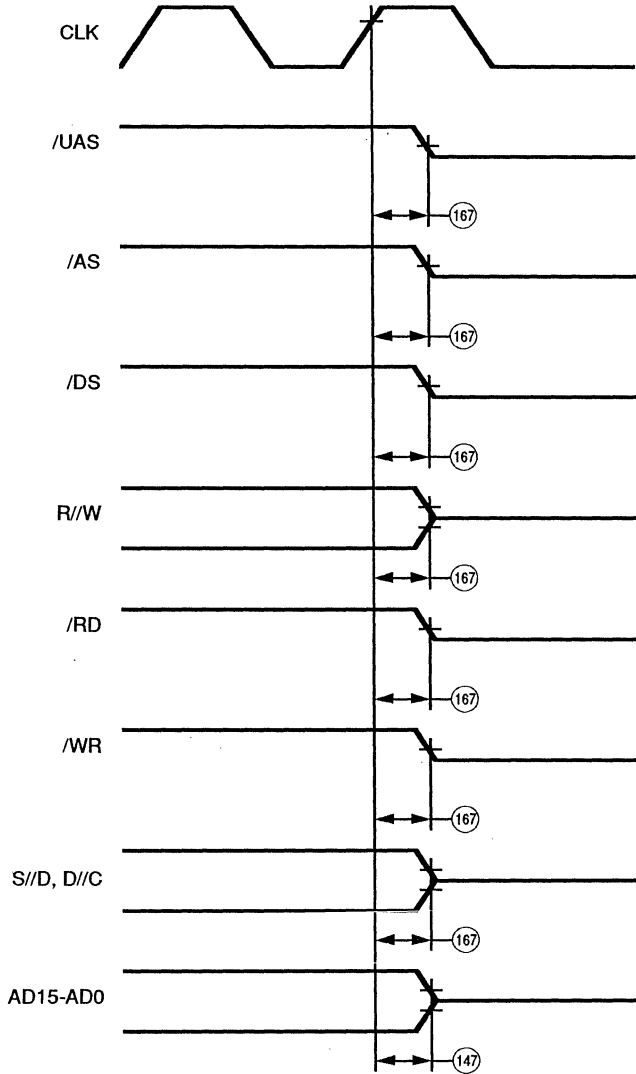


Figure 103. Bus Release

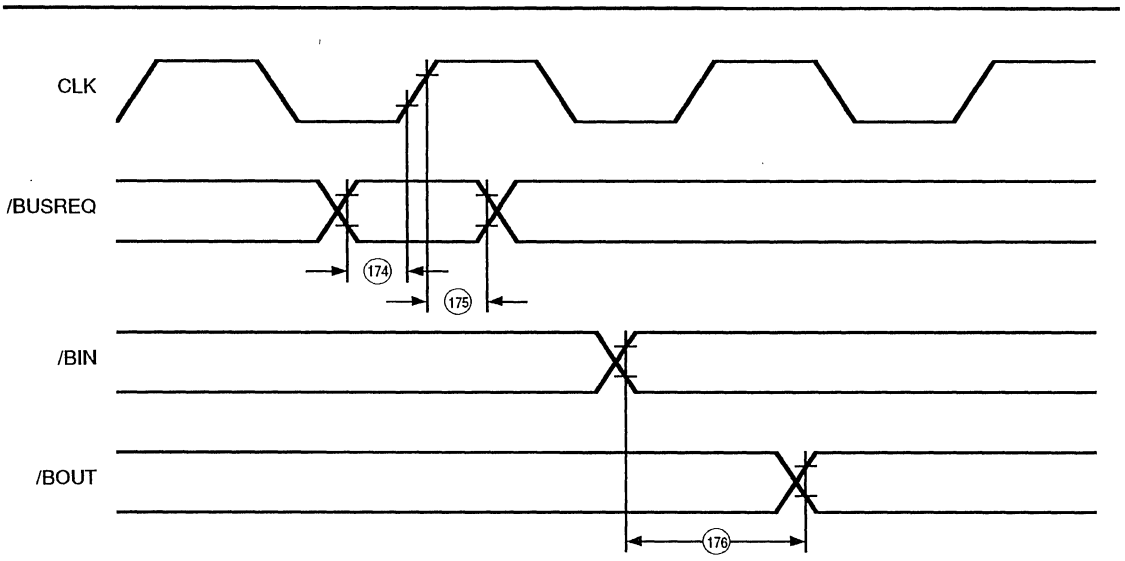


Figure 104. Request Timing



## AC CHARACTERISTICS

### Timing Table

No	Symbol	Parameter	Min	Max	Units	Note
1	Tcyc	Bus Cycle Time	110		ns	
2	TwASI	/AS Low Width	30		ns	
3	TwASh	/AS High Width	60		ns	
4	TwDSI	/DS Low Width	60		ns	
5	TwDSh	/DS High Width	50		ns	
6	TdAS(DS)	/AS Rise to /DS Fall Delay Time	5		ns	
7	TdDS(AS)	/DS Rise to /AS Fall Delay Time	5		ns	
8	TdDS(DRa)	/DS Fall to Data Active Delay	0		ns	
9	TdDS(DRv)	/DS Fall to Data Valid Delay		60	ns	
10	TdDS(DRn)	/DS Rise to Data Not Valid Delay	0		ns	
11	TdDS(DRz)	/DS Rise to Data Float Delay		20	ns	
12	TsCS(AS)	/CS to /AS Rise Setup Time	15		ns	
13	ThCS(AS)	/CS to /AS Rise Hold Time	0		ns	
14	TsADD(AS)	Direct Address to /AS Rise Setup Time	15		ns	[1]
15	ThADD(AS)	Direct Address to /AS Rise Hold Time	5		ns	[1]
16	TsSIA(AS)	Status /INTACK to /AS Rise Setup Time	15		ns	
17	ThSIA(AS)	Status /INTACK to /AS Rise Hold Time	5		ns	
18	TsAD(AS)	Address to /AS Rise Setup Time	15		ns	
19	ThAD(AS)	Address to /AS Rise Hold Time	5		ns	
20	TsRW(DS)	R/W to /DS Fall Setup Time	0		ns	
21	ThRW(DS)	R/W to /DS Fall Hold Time	25		ns	
22	TsDSi(RRQ)	/DS Fall to /RxREQ Inactive Delay		60	ns	[4]
23	TdDSr(RRQ)	/DS Rise to /RxREQ Active Delay	0		ns	
24	TsDW(DS)	Write Data to /DS Rise Setup Time	30		ns	
25	ThDW(DS)	Write Data to DS Rise Hold Time	0		ns	
26	TdDSi(TRQ)	/DS Fall to /TxREQ Inactive Delay		60	ns	[5]
27	TdDSr(TRQ)	/DS Rise to /TxREQ Active Delay	0		ns	
28	TwRDI	/RD Low Width	60		ns	
29	TwRDh	/RD High Width	50		ns	
30	TdAS(RD)	/AS Rise to /RD Fall Delay Time	5		ns	
31	TdRD(AS)	/RD Rise to /AS Fall Delay Time	5		ns	
32	TdRD(DRa)	/RD Fall to Data Active Delay	0		ns	
33	TdRD(DRv)	/RD Fall to Data Valid Delay		60	ns	
34	TdRD(DRn)	/RD Rise to Data Not Valid Delay	0		ns	
35	TdRD(DRz)	/RD Rise to Data Float Delay		20	ns	
36	TdRDf(RRQ)	/RD Fall to /RxREQ Inactive Delay		60	ns	[4]
37	TdRDr(RRQ)	/RD Rise to /RxREQ Active Delay	0		ns	
38	TwWRI	/WR Low Width	60		ns	
39	TwWRh	/WR High Width	50		ns	
40	TdAS(WR)	/AS Rise to /WR Fall Delay Time	5		ns	
41	TdWR(AS)	/WR Rise to /AS Fall Delay Time	5		ns	
42	TsDW(WR)	Write Data to /WR Rise Setup Time	30		ns	
43	ThDW(WR)	Write Data to /WR Rise Hold Time	0		ns	
44	TdWRI(TRQ)	/WR Fall to /TxREQ Inactive Delay		60	ns	[5]

No	Symbol	Parameter	Min	Max	Units	Note
45	TdWRr(TRQ)	/WR Rise to /TxREQ Active Delay	0		ns	
46	TsCS(DS)	/CS to /DS Fall Setup Time	0		ns	[2]
47	ThCS(DS)	/CS to /DS Fall Hold Time	25		ns	[2]
48	TsADD(DS)	Direct Address to /DS Fall Setup Time	5		ns	[1,2]
49	ThADD(DS)	Direct Address to /DS Fall Hold Time	25		ns	[1,2]
50	TsSIA(DS)	Status /INTACK to /DS Fall Setup time	5		ns	[2]
51	ThSIA(DS)	Status /INTACK to /DS Fall Hold Time	25		ns	[2]
52	TsCS(RD)	/CS to /RD Fall Setup Time	0		ns	[2]
53	ThCS(RD)	/CS to /RD Fall Hold Time	25		ns	[2]
54	TsADD(RD)	Direct Address to /RD Fall Setup Time	5		ns	[1,2]
55	ThADD(RD)	Direct Address to /RD Fall Hold Time	25		ns	[1,2]
56	TsSIA(RD)	Status /INTACK to /RD Fall Setup Time	5		ns	[2]
57	ThSIA(RD)	Status /INTACK to /RD Fall Hold Time	25		ns	[2]
58	TsCS(WR)	/CS to /WR Fall Setup Time	0		ns	[2]
59	ThCS(WR)	/CS to /WR Fall Hold Time	25		ns	[2]
60	TsADD(WR)	Direct Address to /WR Fall Setup Time	5		ns	[1,2]
61	ThADD(WR)	Direct Address to /WR Fall Hold Time	25		ns	[1,2]
62	TsSIA(WR)	Status /INTACK to /WR Fall Setup Time	5		ns	[2]
63	ThSIA(WR)	Status /INTACK to /WR Fall Hold Time	25		ns	[2]
78	TdDSI(RDY)	/DS Fall (Intack) to /RDY Fall Delay		200	ns	
79	TdRDY(DRv)	/RDY Fall to Data Valid Delay		40	ns	
80	TdDSr(RDY)	/DS Rise to /RDY Rise Delay		40	ns	
81	TsIEI(DSI)	IEI to /DS Fall (Intack) Setup Time	10		ns	
82	ThIEI(DSI)	IEI to /DS Rise (Intack) Hold Time	0		ns	
83	TdIEI(IEO)	IEI to IEO Delay		30	ns	
84	TdAS(IEO)	/AS Rise (Intack) to IEO Delay		60	ns	
85	TdDSI(INT)	/DS Fall to /INT Inactive Delay		200	ns	
86	TdDSI(Wf)	/DS Fall (Intack) to /WAIT Fall Delay		40	ns	
87	TdDSI(Wr)	/DS Fall (Intack) to /WAIT Rise Delay		200	ns	
88	TdW(DRv)	/WAIT Rise to Data Valid Delay		40	ns	
89	TdRDf(RDY)	/RD Fall (Intack) to /RDY Fall Delay		200	ns	
90	TdRDr(RDY)	/RD Rise to /RDY Rise Delay		40	ns	
91	TsIEI(RDI)	IEI to /RD Fall (Intack) Setup Time	10		ns	
92	ThIEI(RDI)	IEI to /RD Rise (Intack) Hold Time	0		ns	
93	TdRDI(INT)	/RD Fall (Intack) to /INT Inactive Delay		200	ns	
94	TdRDI(Wf)	/RD Fall (Intack) to /WAIT Fall Delay		40	ns	
95	TdRDI(Wr)	/RD Fall (Intack) to /WAIT Rise Delay		200	ns	
96	TwPIAl	Pulsed /INTACK Low Width	60		ns	
97	TwPIAh	Pulsed /INTACK High Width	50		ns	
98	TdAS(PIA)	/AS Rise to Pulsed /INTACK Fall Delay Time	5		ns	
99	TdPIA(AS)	Pulsed /INTACK Rise to /AS Fall Delay Time	5		ns	
100	TdPIA(DRa)	Pulsed /INTACK Fall to Data Active Delay	0		ns	
101	TdPIA(DRn)	Pulsed /INTACK Rise to Data Not Valid Delay	0		ns	
102	TdPIA(DRz)	Pulsed /INTACK Rise to Data Float Delay		20	ns	

## AC CHARACTERISTICS (Continued)

Timing Table

No	Symbol	Parameter	Min	Max	Units	Note
103	TsIEI(PIA)	IEI to Pulsed /INTACK Fall Setup Time	10		ns	
104	ThIEI(PIA)	IEI to Pulsed /INTACK Rise Hold Time	0		ns	
105	TdPIA(IEO)	Pulsed /INTACK Fall to IEO Delay		60	ns	
106	TdPIA(INT)	Pulsed /INTACK Fall to /INT Inactive Delay		200	ns	
107	TdPIAf(RDY)	Pulsed /INTACK Fall to /RDY Fall Delay		200	ns	
108	TdPIAr(RDY)	Pulsed /INTACK Rise to /RDY Rise Delay		40	ns	
109	TdPIA(Wf)	Pulsed /INTACK Fall to /WAIT Fall Delay		40	ns	
110	TdPIA(Wr)	Pulsed /INTACK Fall to /WAIT Rise Delay		200	ns	
111	TdSIA(INT)	Status /INTACK Fall to IEO Inactive Delay		200	ns	[2]
112	TwSTBh	/Strobe High Width	50		ns	[3]
113	TwRESl	/RESET Low Width	170		ns	
114	TwRESH	/RESET High Width	60		ns	
115	TdRES(STB)	/RESET Rise to /STB Fall	60		ns	[3]
116	TdDSf(RDY)	/DS Fall to /RDY Fall Delay		50	ns	
117	TdWRf(RDY)	/WR Fall to /RDY Fall Delay		50	ns	
118	TdWRr(RDY)	/WR Rise to /RDY Rise Delay		40	ns	
119	TdRDf(RDY)	/RD Fall to /RDY Fall Delay		50	ns	
120	TwCLKl	CLK Low Width	20		ns	
121a	TwCLKh	CLK High Width	20		ns	
121b	TwCLKh	CLK High Width (Linked List Mode)	28		ns	[12]
122a	TcCLK	CLK Cycle Time	50		ns	
122b	TcCLK	CLK Cycle Time (Linked List Mode)	60		ns	[12]
123	TfCLK	CLK Fall Time		5	ns	
124	TrCLK	CLK Rise Time		5	ns	
125	TdCLKr(UAS)	CLK Rise to /UAS Fall Delay		25	ns	[6]
126	TwUASl	/UAS Low Width	20		ns	[6,7]
127	TdCLKf(UAS)	CLK Fall to /UAS Rise Delay		25	ns	[6]
128	TdCLKr(AS)	CLK Rise to /AS Fall Delay		25	ns	[6]
129	TwASl	/AS Low Width	20		ns	[6,7]
130	TdCLKf(AS)	CLK Fall to /AS Rise Delay		25	ns	[6]
131	TdAS(DSr)	/AS Rise to /DS Fall (Read) Delay	20		ns	[6,8]
132	TdCLKr(DS)	CLK Rise to /DS Delay		25	ns	[6]
133	TwDSl	/DS (Read) Low Width	70		ns	[6,9]
134	TdCLKf(DS)	CLK Fall to /DS Delay		25	ns	[6]
135	TsDR(DS)	Read Data to /DS Rise Setup Time	20		ns	[6]
136	ThDR(DS)	Read Data to /DS Rise Hold Time	0		ns	[6]
137	TdCLK(RW)	CLK Rise to R/W Delay		25	ns	[6]
138	TdAS(RD)	/AS Rise to /RD Fall Delay	20		ns	[6,8]
139	TdCLKr(RD)	CLK Rise to /RD Delay		25	ns	[6]
140	TwRDI	/RD Low Width	70		ns	[6,9]
141	TdCLKf(RD)	CLK Fall to /RD Delay		25	ns	[6]
142	TsDR(RD)	Read Data to /RD Rise Setup Time	20		ns	[6]
143	ThDR(RD)	Read Data to /RD Rise Hold Time	0		ns	[6]
144	TdCLK(ADD)	CLK Rise to Direct Address Delay		25	ns	[1,6]
145	TdCLK(AD)	CLK Rise to Address Delay	TdCLKf(DS)	30	ns	[6]
146	ThAD(PC)	Address to CLK Rise Hold Time	0		ns	[6]

No	Symbol	Parameter	Min	Max	Units	Note
147	TdCLK(ADz)	CLK Rise to Address Float Delay		25	ns	[6]
148	TdCLK(ADa)	CLK Rise to Address Active Delay		25	ns	[6]
149	TsAD(UAS)	Address to /UAS Rise Setup Time	10		ns	[6]
150	ThAD(UAS)	Address to /UAS Rise Hold Time	10		ns	[6]
151	TsAD(AS)	Address to /AS Rise Setup Time	10		ns	[6]
152	ThAD(AS)	Address to /AS Rise Hold Time	10		ns	[6]
153	TsW(CLK)	/WAIT to CLK Fall Setup Time	10		ns	[6]
154	ThW(CLK)	/WAIT to CLK Fall Hold Time	15		ns	[6]
155	TsRDY(CLK)	/READY to CLK Fall Setup Time	10		ns	[6]
156	ThRDY(CLK)	/READY to CLK Fall Hold Time	15		ns	[6]
157	ThDW(CLK)	Write Data to CLK Rise Hold Time	0		ns	[6]
158	TdAS(DSw)	/AS Rise to /DS Fall (Write) Delay	40		ns	[6,10]
159	TsDW(DS)	Write Data to /DS Fall Setup Time	20		ns	[6,7]
160	TwDSlw	/DS (Write) Low Width	45		ns	[6,11]
161	ThDW(DS)	Write Data to /DS Rise Hold Time	20		ns	[6,8]
162	TdAS(WR)	/AS Rise to /WR Fall Delay	40		ns	[6,10]
163	TsDW(WR)	Write Data to /WR Fall Setup Time	20		ns	[6,7]
164	TwWRI	/WR Low Width	45		ns	[6,11]
165	ThDW(WR)	Write Data to /WR Rise Hold Time	20		ns	[6,8]
166	TdCLK(WR)	CLK Fall to /WR Delay		25	ns	[6]
167	TdCLK(BUSz)	CLK Rise to Bus Float Delay		25	ns	[6]
168	TsABT(CLK)	/ABORT to CLK Rise Setup Time	10		ns	[6]
169	ThABT(CLK)	/ABORT to CLK Rise Hold Time	15		ns	[6]
170	TdCLK(BRQ)	CLK Rise to /BUSREQ Delay		25	ns	[6]
171	TdCLK(BUSa)	CLK Rise to Bus Active Delay		25	ns	[6]
172	TsBIN(CLK)	/BIN to CLK Rise Setup Time	10		ns	[6]
173	ThBIN(CLK)	/BIN to CLK Rise Hold Time	15		ns	[6]
174	TsBRQ(CLK)	/BUSREQ to CLK Rise Setup Time	25		ns	[6]
175	ThBRQ(CLK)	/BUSREQ to CLK Rise Hold Time	0		ns	[6]
176	TdBIN(BOT)	/BIN to /BOUT Delay		60	ns	

#### Notes:

##### AC Test Conditions:

$V_{CC} = 5V \pm 5\%$  unless otherwise specified,  
over specified temperature range.

$V_{IH} = 2.0V$   $V_{OH} = 2.0V$

$V_{IL} = 0.8V$   $V_{OL} = 0.8V$

Float = +0.5V

##### Link List Mode Timing Parameters

In Linked List Mode, the system clock cycle is extended to 60 ns, and the system clock High pulse width is extended to 30 ns. This is due to the internal timing paths unique to the Linked List Mode. This equates to a system clock frequency of 16.667 MHz. The transmit and receive bit rates are not affected.

- [1] Direct Address is any of S//D, D//C or AD15-AD8 used as an address bus.
- [2] The parameter applies only when /AS is not present.
- [3] Strobe is any of /DS, /RD, /WR or Pulsed /INTACK.
- [4] Parameter applies only if read empties the receive FIFO.
- [5] Parameter applies only if write fills the transmit FIFO.
- [6] Parameter applies only while the IUSC is bus master.
- [7] Parameter is clock-cycle dependent,  $TwCLKh + TtCLK - 5$ .
- [8] Parameter is clock-cycle dependent,  $TwCLKl + TrCLK - 5$ .
- [9] Parameter is clock-cycle dependent,  
 $TcCLK + TwCLKh + TtCLK - 5$ .
- [10] Parameter is clock-cycle dependent,  $TcCLK - 10$ .
- [11] Parameter is clock-cycle dependent,  $TcCLK - 5$ .
- [12] Clock cycle parameters  $TwCLKh$  and  $TcCLK$  have unique values for Linked List Mode.

# AC CHARACTERISTICS

## General Timing Diagram

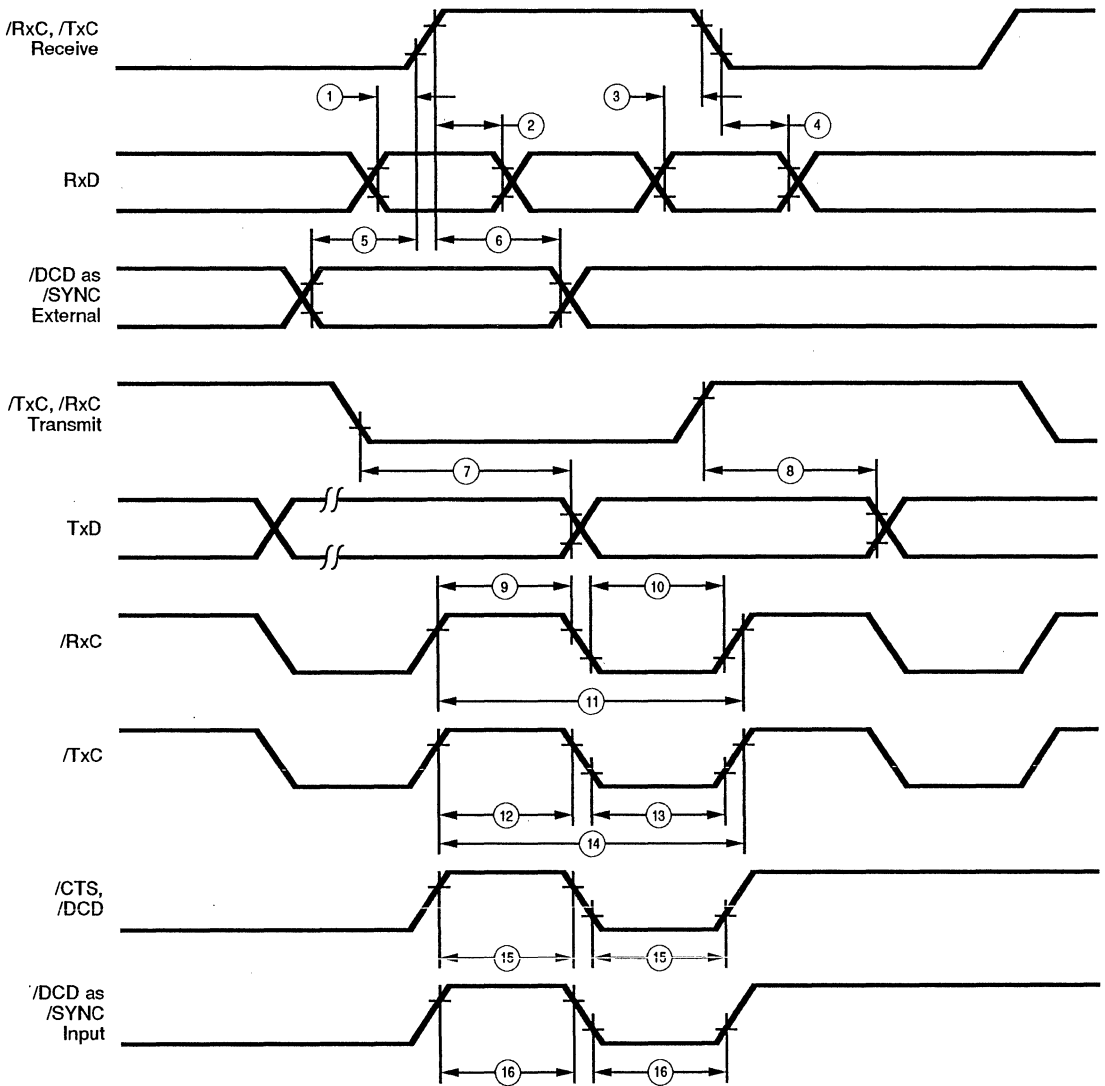


Figure 105. General Timing

## AC CHARACTERISTICS

### General Timing Table

No	Symbol	Parameter	Min	Max	Units	Note
1	TsRxD(RxCr)	RxD to /RxC Rise Setup Time (x1 Mode)	0		ns	[1]
2	ThRxD(RxCr)	RxD to /RxC Rise Hold Time (x1 Mode)	20		ns	[1]
3	TsRxd(RxCf)	RxD to /RxC Fall Setup Time (x1 Mode)	0		ns	[1,3]
4	ThRxD(RxCf)	RxD to /RxC Fall Hold Time (x1 Mode)	20		ns	[1,3]
5	TsSy(RxC)	/DCD as /SYNC to /RxC Rise Setup Time	0		ns	[1]
6	ThSy(RxC)	/DCD as /SYNC to /RxC Rise Hold Time (x1 Mode)	20		ns	[1]
7	TdTxCl(TxD)	/TxCl Fall to TxD Delay		35	ns	[2]
8	TdTxCr(TxD)	/TxCr Rise to TxD Delay		35	ns	[2,3]
9	TwRxCh	/RxC High Width	20		ns	
10	TwRxCl	/RxC Low Width	20		ns	
11	TcRxC	/RxC Cycle Time	50		ns	
12	TwTxCh	/TxCl High Width	20		ns	
13	TwTxCl	/TxCl Low Width	20		ns	
14	TcTxCl	/TxCl Cycle Time	50		ns	
15	TwExT	/DCD or /CTS Pulse Width	35		ns	
16	TWSY	/DCD as /SYNC Input Pulse Width	35		ns	

**AC CHARACTERISTICS**  
System Timing Diagram

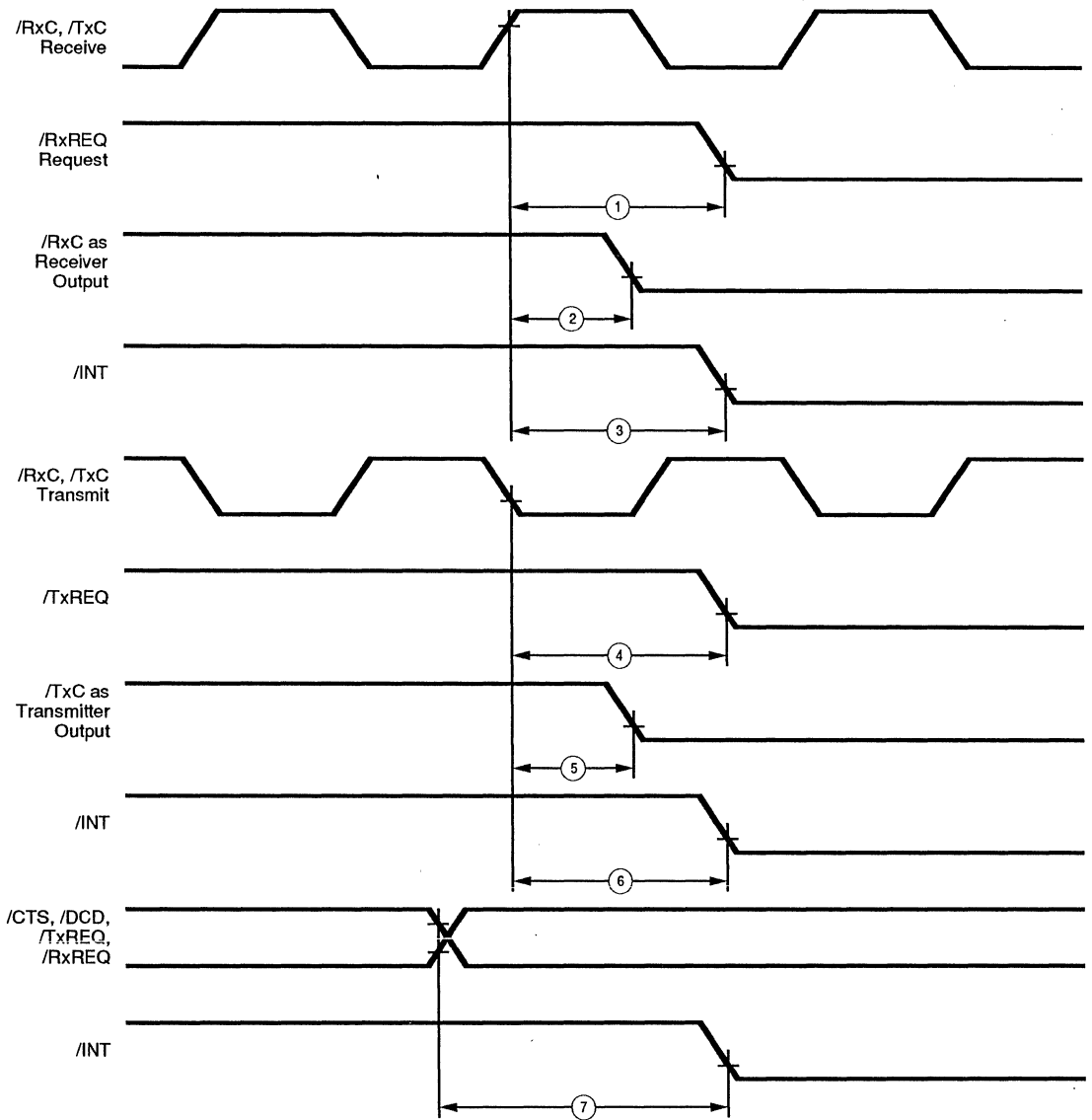


Figure 106. Z16C32 System Timing

## AC CHARACTERISTICS

### System Timing Table

No	Symbol	Parameter	Min	Max	Units	Note
1	TdRxC(REQ)	/RxC Rise to /RxREQ Valid Delay		50	ns	[2]
2	TdRxC(RxC)	/TxC Rise to /RxC as Receiver Output Valid Delay		50	ns	[2]
3	TdRxC(INT)	/RxC Rise to /INT Valid Delay		50	ns	[2]
4	TdTxC(REQ)	/TxC Fall to /TxREQ Valid Delay		50	ns	[2]
5	TdTxC(TxC)	/RxC Fall to /TxC as transmitter Output Valid Delay		50	ns	
6	TdTxC(INT)	/TxC Fall to /INT Valid Delay		50	ns	[2]
7	TdEXT(INT)	/CTS, /DCD, /TxREQ, /RxREQ transition to /INT Valid Delay		50	ns	

**Notes:**

[1] /RxC is /RxC or /TxC, whichever is supplying the receive clock.

[2] /TxC is /TxC or /RxC, whichever is supplying the transmit clock.

[3] Parameter applies only to FM encoding/decoding.



---



# Z16C33

## CMOS MUSC

### MONO-UNIVERSAL SERIAL CONTROLLER

#### FEATURES

- 0 to 10 Mbit/sec, full-duplex channel, with two baud rate generators and a digital phase-locked loop for clock recovery.
- 32-byte data FIFO's for receiver and transmitter
- 12.5 MByte/sec (16 bit) data bus bandwidth
- Multi-protocol operation under program control with independent mode selection for receiver and transmitter.
- Async mode with one to eight bits/character, 1/16 to 2 stop bits/character in 1/16 bit increments; programmable clock factor; break detect and generation; odd, even, mark, space or no parity and framing error detection. Supports one Address/Data bit and MIL STD 1553B protocols.
- Byte oriented synchronous mode with one to eight bits/character; programmable idle line condition; optional receive sync stripping; optional preamble transmission; 16- or 32-bit CRC and transmit-to-receive slaving (for X.21).
- Bisync mode with 2- to 16-bit programmable sync character; programmable idle line condition; optional receive sync stripping; optional preamble transmission; 16- or 32 bit CRC.
- External character sync mode for receive
- Transparent Bisync mode with EBCDIC or ASCII character code; automatic CRC handling; programmable idle line condition; optional preamble transmission; automatic recognition of DLE, SYN, SOH, ITX, ETX, ETB, EOT, ENQ and ITB.
- HDLC/SDLC mode with eight bit address compare; extended address field option; 16- or 32-bit CRC; programmable idle line condition; optional preamble transmission and loop mode.
- DMA interface with separate request and acknowledge for the receiver and transmitter
- Channel load command for DMA controlled initialization
- Flexible bus interface for direct connection to most microprocessors; user programmable for 8 or 16 bits wide. Directly supports 680X0 family or 8X86 family bus interfaces.
- ISDN time slot assigner
- 8-bit general purpose port with transition detection
- Low power CMOS
- 68-pin PLCC package

#### GENERAL DESCRIPTION

The MUSC (Mono - Universal Serial Controller) is a single-channel multi-protocol data communications peripheral designed for use with any conventional multiplexed or non-multiplexed bus. The MUSC functions as a serial-to-parallel, parallel-to-serial converter/controller and is software configured to satisfy a wide variety of serial communications applications. The device contains a variety of new, sophisticated internal functions including two baud rate

generators, a digital phase-locked loop, character counters for both receive and transmit, and 32-byte data FIFO's for both the receiver and transmitter.

The MUSC handles asynchronous formats, synchronous byte-oriented formats (e.g. BISYNC), and synchronous bit-oriented formats like HDLC. This device supports virtually any serial data transfer application.

## GENERAL DESCRIPTION (Continued)

The device can generate and check CRC in any synchronous mode and is programmed to check data integrity in various modes. The MUSC also has facilities for modem controls. In applications where these controls are not needed, the modem controls are used for general-purpose I/O. The same is true for most of the other pins.

Interrupts are supported by a daisy-chain hierarchy with the serial channel. There are no interrupts associated with the 8-bit Port.

High-speed data transfers via DMA are supported by a Request/Acknowledge signal pair for both receiver and transmitter. The device supports automatic status transfer via DMA and allows device initialization under DMA control.

Support tools are available to aid the designer in efficiently programming the MUSC. The Technical Manual describes in detail all features presented in this Product Specification and gives programming sequence hints. The Programmer's Assistant is an MS-DOS, disk-based programming initialization tool, used in conjunction with the Technical Manual. Also, there are assorted application notes and development boards to assist the designer in hardware/software development.

**Note:** All Signals with a preceding front slash, "/", are active Low, e.g.: B/W (WORD is active Low); /B/W (BYTE is active Low, only); /N/S (NORMAL and SYSTEM are both active Low).

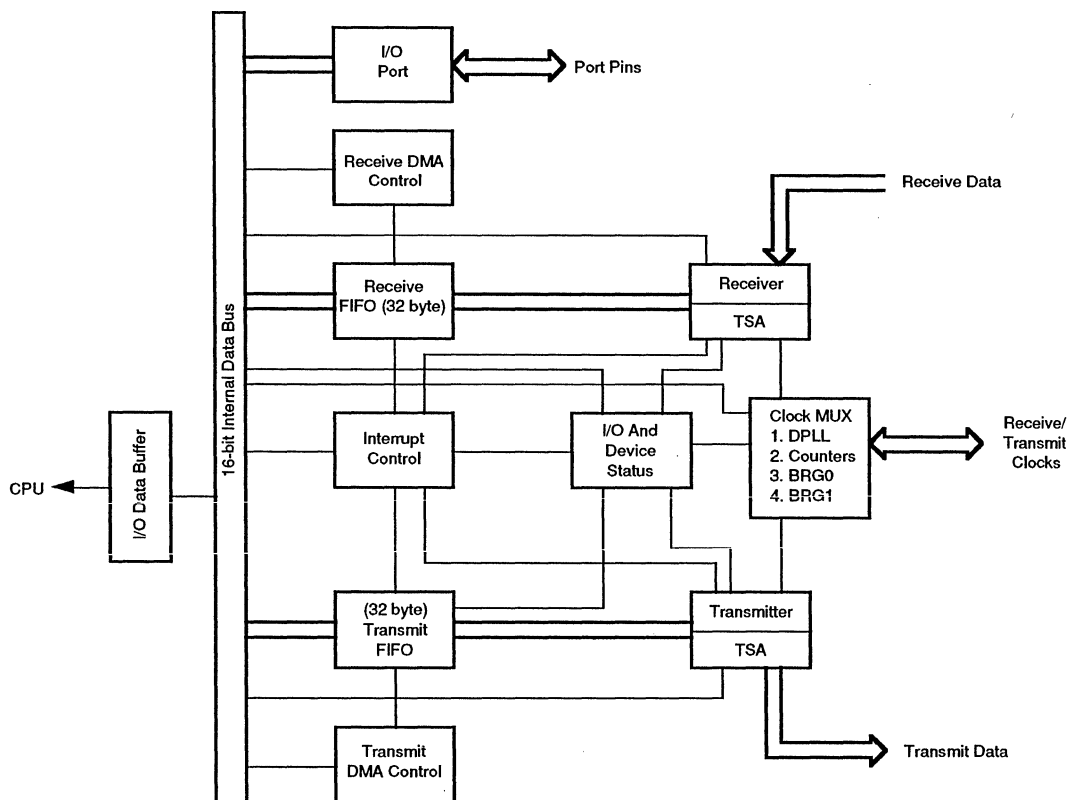


Figure 1. MUSC Block Diagram

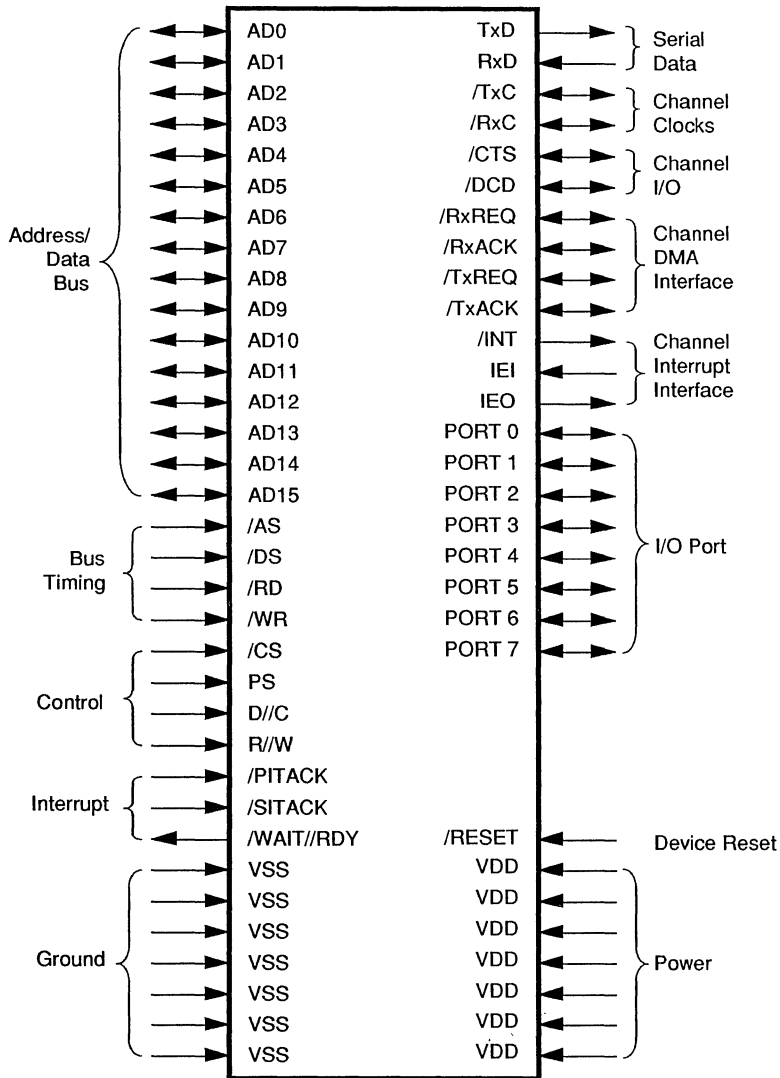
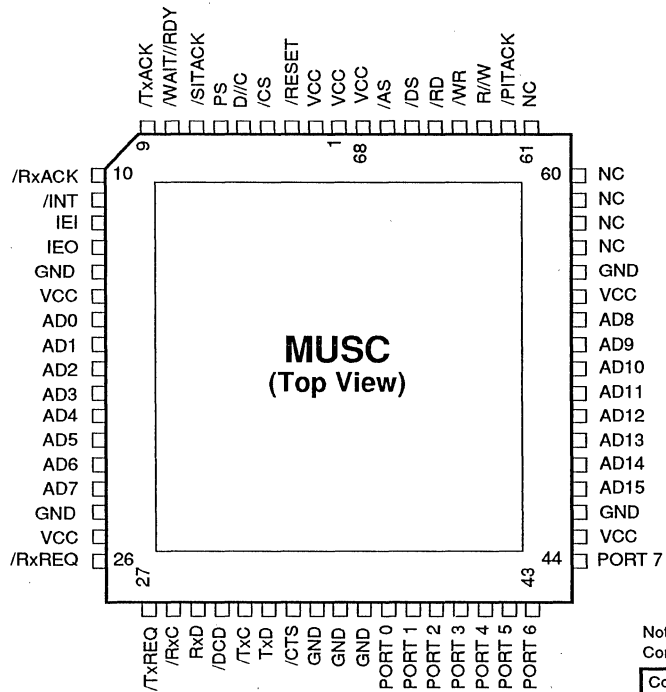


Figure 2. Pin Functions



Note: Power connections follow conventional descriptions below

Connection	Circuit	Device
Power	V <sub>CC</sub>	V <sub>DD</sub>
Ground	GND	V <sub>SS</sub>

Figure 3. Pin Assignments

## PIN DESCRIPTION

The device contains 13 pins for channel I/O, 16 pins for address and data, 12 pins for CPU handshake, 8 pins for the I/O Port, and 14 pins for power and ground.

Three separate bus interface types are available for the device: The Bus Configuration Register (BCR), the external connection of the Protocol Select (PS) pin, and external connections to the AD bus control selection of the bus type.

A 16-bit bus is selected by setting BCR bit 2 to a 1.

The 8-bit bus is selected by setting BCR bit 2 to zero and tying AD15 - AD8 to VSS.

The 8-bit bus with separate address is selected by setting BCR bit 2 to zero and, during the BCR write, forcing AD15 to a 1 and forcing AD14-AD8 to zero.

The multiplexed bus is selected for the MUSC if there is an Address Strobe prior to, or during, the transaction which writes the BCR.

If no Address Strobe is present prior to or during the transaction which writes the BCR, a non-multiplexed bus is selected (See Figure 6).

---

## PIN ASSIGNMENTS

**/RESET.** *Reset* (input, active Low). This signal resets the device to a known state. The first write to the MUSC after a reset accesses the BCR to select additional bus options for the device.

**/AS.** *Address Strobe* (input, active Low). This signal is used in the multiplexed bus modes to latch the address on the AD lines. The /AS signal is not used in the non-multiplexed bus modes and is tied to VDD in these cases.

**/DS.** *Data Strobe* (input, active Low). This signal strobes data out of the device during a read and strobes an interrupt vector out of the device during an interrupt acknowledge cycle. /DS also strobes data into the device depending on the state of R//W.

**/RD.** *Read Strobe* (input, active Low). This signal strobes data out of the device during a read and strobes an interrupt vector out of the device during an interrupt acknowledge cycle.

**/WR.** *Write Strobe* (input, active Low). This signal strobes data into the device during a write.

**R//W.** *Read/Write* (input). This signal determines the direction of data transfer for a read or write cycle in conjunction with /DS.

**/CS.** *Chip Select* (input, active Low). This signal selects the device for access and is asserted for read and write cycles, but is ignored during interrupt acknowledge and fly-by DMA transfers. In the case of a multiplexed bus interface, /CS is latched by the rising edge of /AS.

**PS.** *Protocol Select* (input, active High). This input is sampled and stored during the BCR (Bus Configuration Register) write. It selects the sense of the /WAIT//RDY signal appropriate for different bus interfaces. With PS High, /WAIT//RDY functions as a /WAIT signal and with PS Low, /WAIT//RDY functions as a /READY signal. This selection applies to all bus transactions.

**D//C.** *Data/Control Select* (input). This signal, when High, provides for direct access to the RDR and TDR. In the case of a multiplexed bus interface, D//C High overrides the address provided to the device.

**/SITACK.** *Status Interrupt Acknowledge* (input, active Low). This signal is a status signal indicating that an interrupt acknowledge cycle is in progress. The device is capable of returning an interrupt vector that is encoded with the type of interrupt pending during this acknowledge cycle.

**/PITACK.** *Pulsed Interrupt Acknowledge* (input, active Low). This is a strobe signal indicating that an interrupt acknowledge cycle is in progress. The device is capable of returning an interrupt vector that is encoded with the type of interrupt pending during this acknowledge cycle. /PITACK is programmed to accept either single pulse or double pulse acknowledges. This programming is performed in the BCR. The double pulse acknowledge is compatible with 8X86 family microprocessors.

**/WAIT//RDY.** *Wait/Data Ready* (output, active Low). This signal serves to indicate when the data is available during a read cycle, when the device is ready to receive data during a write cycle, and when a valid vector is available during an interrupt acknowledge cycle. It is programmed to function either as a Wait signal or a Ready signal using the state of the PS pin during the BCR write. When PS is High during the BCR write, this signal functions as a wait output and supports the READY function of 8X86 family microprocessors. When PS is Low during the BCR write, it functions as a ready output and supports the DTACK function of 680X0 family microprocessors.

**AD15-AD0.** *Address/Data Bus* (bidirectional, active High, 3-state). The AD signals carry addresses to, and data to and from, the device. When the 16 bit non-multiplexed bus is selected, AD15-0 carries data to and from the device. Addresses are provided using a pointer within the device that is loaded with the desired register address. When the 8-bit non-multiplexed bus, without separate address, is selected only AD7-0 is used to transfer data. The pointer is used for addressing, and AD15-8 is unused. When the 8-bit non-multiplexed bus with separate address is selected AD7-0 is used to transfer data, while AD15-8 is used as an address bus. When the 16 bit multiplexed bus is selected, addresses are latched from AD7-0 and data transfers are sixteen bits wide. When the 8-bit multiplexed bus without separate address is selected only AD7-0 is used to transfer addresses and data, and AD15-8 is unused. When the 8-bit multiplexed bus with separate address is selected only AD7-0 is used to transfer data, while AD15-8 is used as an address bus.

The non-multiplexed, 16-bit bus interface mode directly supports 680X0 family microprocessors and the multiplexed, 16-bit bus interface mode directly supports the 8X86 family microprocessors. The multiplexed, 8-bit bus interface mode without separate address supports the 8088 family microprocessors.

---

## PIN ASSIGNMENTS (Continued)

**/INT.** *Interrupt Request* (output, active Low). Indicates that the channel has an interrupt condition pending and is requesting service. This output is NOT open-drain.

**IEI.** *Interrupt Enable In* (input, active High). The IEI signal is used with the accompanying IEO signal to form an interrupt daisy chain. An active IEI indicates that no device having higher priority is requesting or servicing an interrupt.

**IEO.** *Interrupt Enable Out* (output, active High). The IEO signal is used with the accompanying IEI signal to form an interrupt daisy chain. IEO is Low if IEI is Low, an interrupt is under service in the channel, or an interrupt is pending during an interrupt acknowledge cycle.

**/TxACK.** *Transmit Acknowledge* (input or output, active Low). The primary function of this signal is to perform fly-by DMA transfers to the transmit FIFO. It also is used as bit input or output.

**/RxACK.** *Receive Acknowledge* (input or output, active Low). The primary function of this signal is to perform fly-by DMA transfers from the receive FIFO. It also is used as bit input or output.

**TxD.** *Transmit Data* (output, active High, 3-state). TxD carries the serial transmit data for the channel.

**RxD.** *Receive Data* (input, active High). RxD carries the serial receive data for the channel.

**/TxC.** *Transmit Clock* (input or output, active Low). This signal may be used as a clock input for any of the functional

blocks within the device. It also is used as an output for various transmitter signals or internal clock signals.

**/RxC.** *Receive Clock* (input or output, active Low). This signal is used as a clock input for any of the functional blocks within the device. It also is used as an output for various receiver signals or internal clock signals.

**/TxREQ.** *Transmit Request* (input or output, active Low). The primary function of this signal is to request DMA transfers to the transmit FIFO. It also is used as a simple input or output.

**/RxREQ.** *Receive Request* (input or output, active Low). The primary function of this signal is to request DMA transfers from the receive FIFO. It also is used as a simple input or output.

**/CTS.** *Clear To Send* (input or output, active Low). /CTS is used as an enable for the transmitter. It also is programmed to generate interrupt on either transition or used as a simple input or output.

**/DCD.** *Data Carrier Detect* (input or output, active Low). This signal is used as an enable for the receiver. It also is programmed to generate an interrupt on either transition or used as a simple input or output.

**PORT7 - PORT0.** *Port Signals* (inputs or outputs, active High). These pins are general purpose I/O pins. They are used as additional MODEM control lines or for other I/O functions. When used as inputs, the ports capture transitions on these pins.

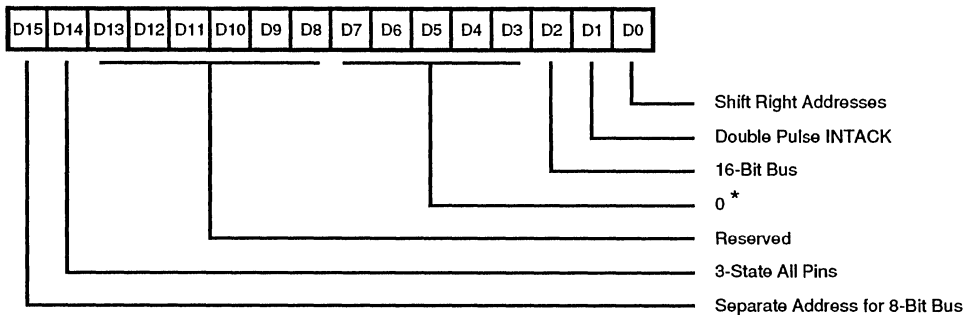
---

## ARCHITECTURE

The MUSC internal structure includes a full-duplex serial channel with two baud rate generators, a digital phase-locked loop for clock recovery, transmit and receive character counters and a full-duplex DMA interface. The bus interface is designed to provide easy interface to most microprocessors, whether they employ a multiplexed,

non-multiplexed, 8-bit or 16-bit bus structure. The channel is controlled by a set of thirty-two 16-bit registers, almost all of which are readable and writable. There is one additional 16-bit register in the bus interface used to configure the nature of the bus interface. The BCR functions are shown in Figure 4.

Address: None



\* Must be programmed as 0.

Figure 4. Bus Configuration Register

## DATA PATH

Both the transmitter and the receiver in the channel are actually microcoded serial processors. As the data shifts through the transmit or receive shift register, the microcode watches for specific bit patterns, counts bits, and at the

appropriate time transfers data to or from the FIFOs. The microcode checks status and generates status interrupts as appropriate.

## FUNCTIONAL DESCRIPTION

The functional capabilities of the MUSC are described from two different points of view: as a data communications device, it transmits and receives data in a wide variety of data communications protocols; as a microprocessor peripheral, the MUSC offers such features as read/write registers, a flexible bus interface, DMA interface support, vectored interrupts, and an eight-bit I/O port.

signal such conditions as: overrun, parity error, framing error, end-of-frame, idle line received, sync acquired, transmit underrun, CRC sent, closing sync/flag sent, abort sent, idle line sent and preamble sent. In addition, several useful internal signals like receive FIFO load, received sync, transmit FIFO read and transmission complete are sent to pins for use by external circuitry.

### Data Communications Capabilities

The MUSC provides a full-duplex channel programmable for use in any common data communication protocol. The receiver and transmitter modes are completely independent. The receiver and transmitter are each supported by a 32-byte deep FIFO and a 16-bit message length counter. All modes allow optional even, odd, mark or space parity. Synchronous modes allow the choice of two 16-bit or one 32-bit CRC polynomial. Selection of from one to eight bits per character is available in both receiver and transmitter independently. Error and status conditions are carried with the data in the receive and transmit FIFOs to greatly reduce the CPU overhead required to send or receive a message. Specific, appropriately timed interrupts are available to

**Asynchronous Mode.** The receiver and transmitter handle data at a rate of 1/16, 1/32, or 1/64 the clock rate. The receiver rejects start bits less than one-half a bit time and will not erroneously assemble characters following a framing error. The transmitter is capable of sending one, two, or anywhere in the range of 1/16th to two stop bits per character in 1/16 bit increments.

**External Sync Mode.** The receiver is synchronized to the receive data stream by an externally-supplied signal on a pin for custom protocol applications.

**Isochronous Mode.** Both transmitter and receiver may operate on start-stop (async) data using a 1x clock. The transmitter sends one or two stop bits.



---

## FUNCTIONAL DESCRIPTION (Continued)

**Asynchronous With Code Violations.** This is similar to Isochronous mode except that the start bit is replaced by a three bit-time code violation pattern as in MIL-STD 1553B. The transmitter sends zero, one or two stop bits.

**Monosync Mode.** In this mode, a single character is used for synchronization. The sync character can be either eight bits long with an arbitrary data character length or programmed to match the data character length. The receiver is capable of automatically stripping sync characters from the received data stream. The transmitter is programmed to automatically send CRC on either an underrun or at the end of a programmed message length.

**Bisync Mode.** This mode is identical to monosync mode except that character synchronization requires two successive characters for synchronization. The two characters need not be identical.

**HDLC Mode.** In this mode, the receiver recognizes flags, performs optional address matching, accommodates extended address fields, 8- or 16-bit control fields and logical control fields, performs zero deletion and CRC checking. The receiver is capable of receiving shared-zero flags, recognizes the abort sequence and can receive arbitrary length messages. The transmitter automatically sends opening and closing flags, performs zero insertion and is programmed to send an abort, an extended abort, a flag or CRC and a flag on transmit underrun. The transmitter automatically sends the closing flag with optional CRC at the end of a programmed message length. Shared-zero flags are selected in the transmitter and a separate character length is programmed for the last character in the frame.

**Bisync Transparent Mode.** In this mode, the synchronization pattern is DLE-SYN, programmable selected from either ASCII or EBCDIC encoding. The receiver recognizes control character sequences and automatically handles CRC calculations without CPU intervention. The transmitter is programmed to send either SYN, DLE-SYN, CRC-SYN, or CRC-DLE-SYN upon underrun and automatically sends the closing DLE-SYN with optional CRC at the end of a programmed message length.

**NBIP Mode.** This mode is identical to async except that the receiver checks for the status of an additional address/data bit between the parity bit and the stop bit. The value of this bit is FIFO'ed along with the data. In the transmitter, this bit is automatically inserted with the value that is FIFO'ed from the transmit data.

**802.3 Mode.** This mode implements the data format of IEEE 802.3 with 16-bit address compare. In this mode,

/DCD and /CTS are used to implement the carrier sense and collision detect interactions with the receiver and transmitter.

**Slaved Monosync Mode.** This mode is available only in the transmitter and allows the transmitter (operating just as though it were in monosync mode) to send data that is byte-synchronous to the data being received by the receiver.

**HDLC Loop Mode.** This mode is available only in the transmitter and allows the MUSC to be used in an HDLC loop configuration. In this mode, the receiver is programmed to operate in HDLC mode to allow the transmitter to echo received messages. Upon receipt of a particular bit pattern (actually a sequence of seven consecutive ones) the transmitter breaks the loop and inserts its own frame(s).

### Data Encoding

The MUSC is programmed to encode and decode the serial data in any of eight different ways (Figure 5). The transmitter encoding method is selected independently of the receiver decoding method.

**NRZ.** In NRZ, a 1 is represented by a High level for the duration of the bit cell and a 0 is represented by a Low level for the duration of the bit cell.

**NRZB.** NRZB is inverted from NRZ.

**NRZI-Mark.** In NRZI-Mark, a 1 is represented by a transition at the beginning of a bit cell, i.e., the level present in the preceding bit cell is reversed. A 0 is represented by the absence of a transition at the beginning of the bit cell.

**NRZI-Space.** In NRZI-Space, a 1 is represented by the absence of a transition at the beginning of a bit cell; i.e., the level present in the preceding bit cell is maintained. A 0 is represented by a transition at the beginning of the bit cell.

**Biphase-Mark.** In Biphase-Mark, a 1 is represented by a transition at the beginning of the bit cell and another transition at the center of the bit cell. A 0 is represented by a transition at the beginning of the bit cell only.

**Biphase-Space.** In Biphase-Space, a 1 is represented by a transition at the beginning of the bit cell only. A 0 is represented by a transition at the beginning of the bit cell and another transition at the center of the bit cell.

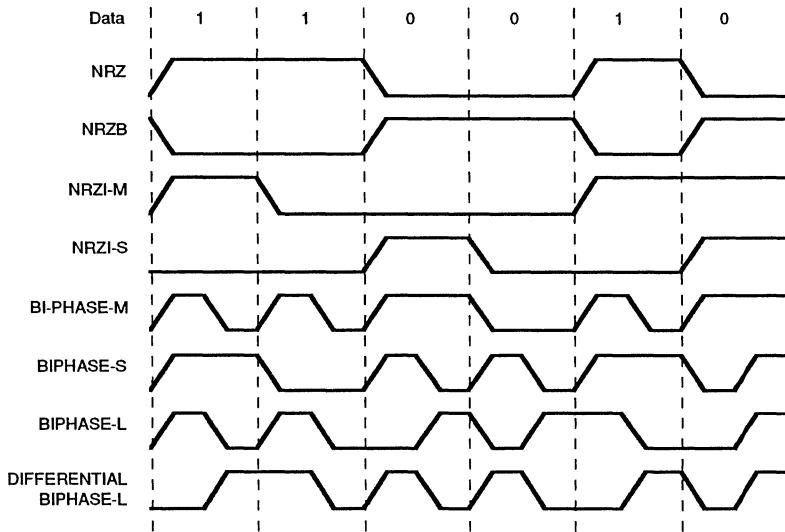


Figure 5. Data Encoding

**Biphase-Level.** In Biphase-Level, a 1 is represented by a High during the first half of the bit cell and a Low during the second half of the bit cell. A 0 is represented by a Low during the first half of the bit cell and a High during the second half of the bit cell.

**Differential Biphase-Level.** In Differential Biphase-Level, a 1 is represented by a transition at the center of the bit cell, with the opposite polarity from the transition at the center of the preceding bit cell. A 0 is represented by a transition at the center of the bit cell with the same polarity as the transition at the center of the preceding bit cell. In both cases, there are transitions at the beginning of the bit cell to set up the level required to make the correct center transition.

### Character Counters

The MUSC contains a 16-bit character counter for both the receiver and transmitter. The receive character counter may be preset either under software control or automatically at the beginning of a receive message. The counter decrements with each receive character and at the end of the receive message the current value in the counter is automatically loaded into a four-deep FIFO. This allows DMA transfer of data to proceed without CPU intervention at the end of a received message, as the values in the FIFO allow the CPU to determine message boundaries in memory. Similarly, the transmit character counter is loaded either

under software control or automatically at the beginning of a transmit message. The counter is decremented with each write to the transmit FIFO. When the counter has decremented to zero, and that byte is sent, the transmitter automatically terminates the message in the appropriate fashion (usually CRC and the closing flag or sync character) without requiring CPU intervention.

### Baud Rate Generators

The MUSC contains two baud rate generators. Each generator consists of a 16-bit time constant register and a 16-bit down counter. In operation, the counter decrements with each baud rate generator clock, and the time constant is automatically reloaded when the count reaches zero. The output of the baud rate generator toggles when the counter reaches a count of one-half of the time constant and again when the counter reaches zero. A new time constant is written at any time but the new value does not take effect until the next load of the counter. The outputs of both baud rate generators are sent to the clock multiplexer for use internally or externally. The baud rate generator output frequency is related to the baud rate generator input clock frequency by the following formula:

$$\text{Output frequency} = \text{Input frequency} / (\text{time constant} + 1).$$

**Note:** This allows an output frequency in the range of 1 to 1/65536 of the input frequency, inclusive.

---

## Digital Phase-Locked Loop

The MUSC contains a digital phase-locked loop (DPLL) to recover clock information from a data stream with NRZI or Biphase encoding. The DPLL is driven by a clock that is nominally 8, 16 or 32 times the receive data rate. The DPLL uses this clock, along with the data stream, to construct a clock for the data. This clock is routed to the receiver, transmitter, or both, or to a pin for use externally. In all modes, the DPLL counts the input clock to create nominal bit times. As the clock is counted, the DPLL watches the incoming data stream for transitions. Whenever a transition is detected, the DPLL makes a count adjustment (during the next counting cycle), to produce an output clock which tracks the incoming bit cells. The DPLL provides properly phased transmit and receive clocks to the clock multiplexer.

### Counters

The MUSC contains two 5-bit counters, which may be programmed to divide an input clock by 4, 8, 16 or 32. The

inputs of these two counters are sent to the clock multiplexer. The counters are used as prescalers for the baud rate generators. They also provide a stable transmit clock from a common source when the DPLL is providing the receive clock.

### Clock Multiplexer

The clock multiplexer selects the clock source for the various blocks in the channel, as well as selecting an internal clock signal to potentially be sent to either the /RxC or /TxC pin.

### Test Modes

The MUSC is programmed for local loopback or auto echo operation. In local loopback, the output of the transmitter is internally routed to the input of the receiver. This allows testing of the MUSC data paths without any external logic. Auto echo connects the RxD pin directly to the TxD pin. This is useful for testing serial links external to the MUSC.

---

## I/O INTERFACE CAPABILITIES

The MUSC offers the choice of polling, interrupt (vectored or non-vectored) and block transfer modes to transfer data, status and control information to and from the CPU.

### Polling

All interrupts are disabled. The registers in the MUSC are automatically updated to reflect current status. The CPU polls the Daisy Chain Control Register (DCCR) to determine status changes and then reads the appropriate status register to find and respond to the change in status. MUSC status bits are grouped according to function to simplify this software action.

### Interrupt

When a MUSC responds to an interrupt acknowledge from the CPU, an interrupt vector may be placed on the data bus. This vector is held in the Interrupt Vector Register (IVR). To speed interrupt response time, the MUSC modifies three bits in this vector to indicate which type of interrupt is being requested.

Each of the six sources of interrupts in the MUSC (Receive Status, Receive Data, Transmit Status, Transmit Data, I/O Status and Device Status) has three bits associated with the interrupt source: Interrupt Pending (IP), Interrupt-Under-Service (IUS) and Interrupt Enable (IE). If the IE bit for a given source is set, then that source requests interrupts. Note that individual sources within the six groups also have

interrupt enable bits which are separately enabled. There is a Master Interrupt Enable (MIE) bit which globally enables or disables interrupts within the serial channel.

The other two bits are related to the interrupt priority chain. The MUSC requests an interrupt only when no higher priority interrupt source is requesting one, e.g., when IE1 is High for the channel. In this case the channel activates the /INT signal. The CPU then responds with an interrupt acknowledge cycle, and the device places a vector on the data bus.

In the MUSC, the IP bit signals that an interrupt request is being serviced. If an IUS is set, all interrupt sources of lower priority within the channel and external to the channel are prevented from requesting interrupts. The internal interrupt sources are inhibited by the state of the internal daisy chain, while lower priority devices are inhibited by the IEO output of the channel being pulled Low and propagated to subsequent peripherals. An IUS bit is set during an interrupt acknowledge cycle if there are no higher priority devices requesting interrupts.

There are six sources of interrupt in the following priority: Receive Status, Receive Data, Transmit Status, Transmit Data, I/O Status and Device Status. There are six sources of Receive Status interrupt. Each one is individually enabled: receiver exited hunt, received idle line, received break/abort, received code violation/end-of-transmission/end-of-frame, parity error and overrun error. The Receive Data

---

interrupt is generated whenever the receive FIFO fills with data beyond the level programmed in the Receive Interrupt Control Register (RICR). There are six sources of Transmit Status interrupt. Each one is individually enabled: preamble sent, idle line sent, abort sent, end-of-frame/end-of-transmission sent, CRC sent and underrun error. The Transmit Data interrupt is generated whenever the transmit FIFO empties below the level programmed in the Transmit Interrupt Control Register (TICR). The I/O Status interrupt serves to report transitions on any of six pins. Interrupts are generated on either or both edges with individual selection and enables for each pin. The pins that are programmed to generate I/O Status interrupts are /RxC, /TxC, /RxREQ, /TxREQ, /DCD and /CTS. These interrupts are independent of the programmed function of the pins. The Device Status interrupt has four individually enabled sources: receive character count FIFO overflow, DPLL sync acquired, BRG1 zero count and BRG0 zero count.

### Block Transfer Mode

The MUSC accommodates block transfers via DMA through the /RxREQ, /TxREQ, /RxACK and /TxACK pins. The /RxREQ signal is activated when the fill level of the receive FIFO exceeds the value programmed in the RICR. The DMA responds with either a normal bus transaction or by activating the /RxACK pin to read the data directly (fly-by transfer). The /TxREQ signal is activated when the empty level of the transmit FIFO falls below the value programmed in the TICR. The DMA responds either with a normal bus transaction or by activating the /TxACK pin to write the data directly (fly-by transfer). The /RxACK and /TxACK pin functions for this mode are controlled by the Hardware Configuration Register (HCR). When using the /RxACK and /TxACK pins to transfer data, no chip select is necessary; these are dedicated strobes for the appropriate FIFO.

### Time Slot Assigner

The MUSC is equipped with two time slot assigners to support ISDN communications. There is one assigner for the receiver and one assigner for the transmitter and the assigners function independently. The time slot assigner

selects one or more time slots within a frame, however, all selected time slots must be contiguous. The first selected time slot is programmable from slot 0 (the first slot) to slot 127 of the frame. The total number of concatenated slots is programmable from 1 to 15 (total slots).

The time of the slot is offset an integral number of clocks. This offset is a delay and is programmable from 0 (no offset) to 7 clocks in increments of one clock (one bit cell). This offset is used to compensate for delays in frame sync detection logic.

### I/O Port

The Port pins are general purpose I/O pins. They are used as additional MODEM control lines or other I/O functions. Each port bit is individually programmable for the three-state mode, output a logic 0, or output a logic 1. This programming is done in the Port Control Register. When programmed to be three-stated, the ports are used as inputs. Whether used as inputs or outputs, the port pins can be read at any time.

The port pins capture edge transition's input to the port. This programming for the capture is done using the Port Latched/Unlatch command bits in the Port Status Register. Each port bit is individually controlled. The Latched/Unlatch bit is used as a status signal to indicate that a transition has occurred on the port pin and as a command to open the latches that capture this transition. Both rising edge and falling edge transitions are detected. When a transition is detected, the latch closes holding the post transition state of the input.

The Latched/Unlatch bit is held at 0 if no transitions occur on the port pin; this bit is set to a 1 when a rising edge or falling edge transition is detected, or immediately after the latch is opened if one or more transitions occurred while the latch was closed. Writing a 0 to the Latched/Unlatch bit has no effect on the latch. Writing a 1 to this bit resets the status bit and opens the latch. To use the port as an input without edge detection, a 1 would be written to the Latched/Unlatch bit to open the latch and then the Port Status Register would be read to obtain the current pin input status.

---

## PROGRAMMING

The Programmer's Assistant (MS DOS based) and Technical Manual are available to provide details about programming the MUSC. Also included are explanations and features of all registers in the MUSC.

The registers in the MUSC must be programmed by the system to configure the channel. Before this can occur, the system must program the bus interface by writing to the Bus Configuration Register (BCR). The BCR has no specific address and is only accessible immediately after a hardware reset of the device. The first write to the MUSC, after a hardware reset, programs the BCR. From that time on the normal channel registers may be accessed. No specific address need be presented to the MUSC for the BCR write; the MUSC knows that the first write after a hardware reset is destined for the BCR.

In the multiplexed bus case, all registers are directly addressable via the address latched by /AS at the beginning of a bus transaction. The address may be decoded from either AD6-AD0 or AD7-AD1. This is controlled by the Shift Right/Shift Left bit in the BCR. The address maps for these two cases is shown in Table 1. The D//C pin is still used to directly access the receive and send data registers (RDR and TDR) in the multiplexed bus; if D//C is High, the address latched by /AS is ignored and an access of RDR or TDR is performed.

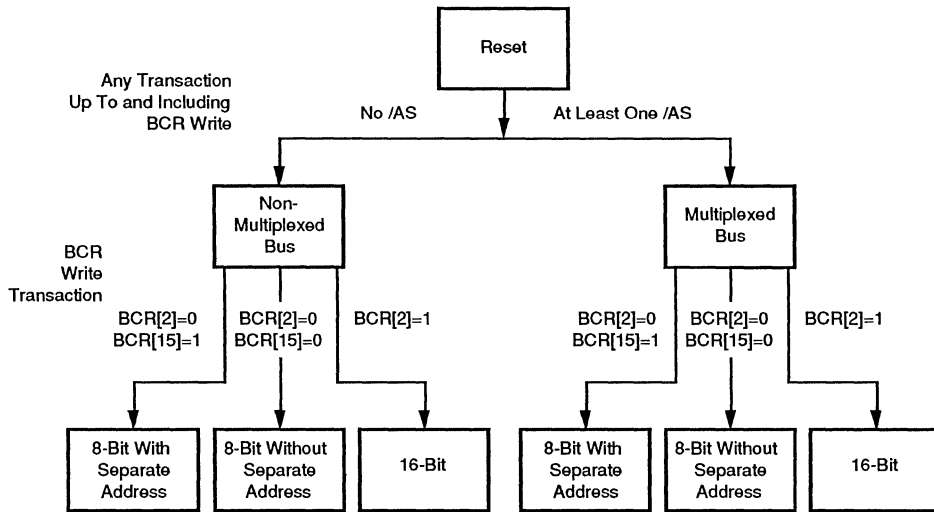
**Table 1. Multiplexed Bus Address Assignments**

Address Signal	Shift Left	Shift Right
Byte/Word Access	AD7	AD6
Address 4	AD6	AD5
Address 3	AD5	AD4
Address 2	AD4	AD3
Address 1	AD3	AD2
Address 0	AD2	AD1
Upper//Lower Byte Select	AD1	AD0

In the non-multiplexed bus case, the channel registers are accessed indirectly using the address pointer in the Channel Command/Address Register (CCAR). The address of the desired register is first written to the CCAR and then the selected register is accessed; the pointer in the CCAR is automatically cleared after this access. The RDR and TDR are still accessed directly using the D//C pin, without disturbing the contents of the pointer in the CCAR.

There are two important things to note about the MUSC. First, the Channel Reset bit in the CCAR places the channel in the reset state. To exit this reset state either a word of all zeros is written to the CCAR (16-bit bus) or a byte of all zeros is written to the lower byte of the CCAR (8-bit bus). The second thing to note is that after reset, the transmit and receive clocks are not connected. The first thing that should be done in any initialization sequence is a write to the Clock Mode Control Register (CMCR) to select a clock source for the receiver and transmitter.

The register addressing is shown in Table 2 and the bit assignments for the registers are shown in Figure 6.



Note:  
The presence of one transaction with an /AS active, between reset up to and including the BCR write, chooses a multiplexed type of bus.

Figure 6. BCR Reset Sequence and Bit Assignments

Table 2. Register Address List

Address A4-A0			Address A4-A0		
0000	CCAR	Channel Command/Address Register	10001	RMR	Receive Mode Register
0001	CMR	Channel Mode Register	10010	RCSR	Receive Command/Status Register
00010	CCSR	Channel Command/Status Register	10011	RICR	Receive Interrupt Control Register
00011	CCR	Channel Control Register	10100	RSR	Receive Sync Register
00100	PSR	Port Status Register	10101	RCLR	Receive Count Limit Register
00101	PCR	Port Control Register	10110	RCCR	Receive Character Count Register
00110	TMDR	Test Mode Data Register	10111	TC0R	Time Constant 0 Register
00111	TMCR	Test Mode Control Register	1X000	TDR	Transmit Data Register (Write Only)
01000	CMCR	Clock Mode Control Register	11001	TMR	Transmit Mode Register
01001	HCR	Hardware Configuration Register	11010	TCSR	Transmit Command/Status Register
01010	IVR	Interrupt Vector Register	11011	TICR	Transmit Interrupt Control Register
01011	IOCR	I/O Control Register	11100	TSR	Transmit Sync Register
01100	ICR	Interrupt Control Register	11101	TCLR	Transmit Count Limit Register
01101	DCCR	Daisy-Chain Control Register	11110	TCCR	Transmit Character Count Register
01110	MISR	Misc Interrupt Status Register	11111	TC1R	Time Constant 1 Register
01111	SICR	Status Interrupt Control Register	XXXXX	BCR	Bus Configuration Register
1X000	RDR	Receive Data Register (Read Only)			

Address: 00000

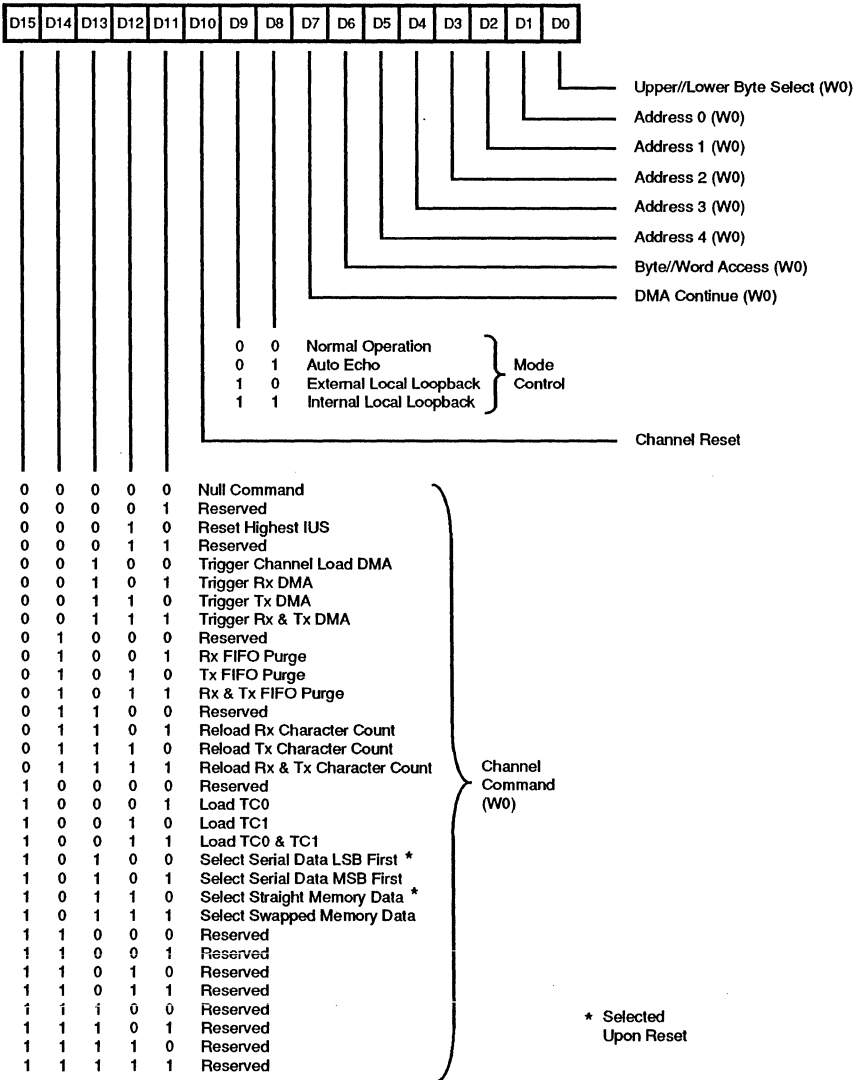


Figure 7. Channel Command/Address Register

Address: 00001

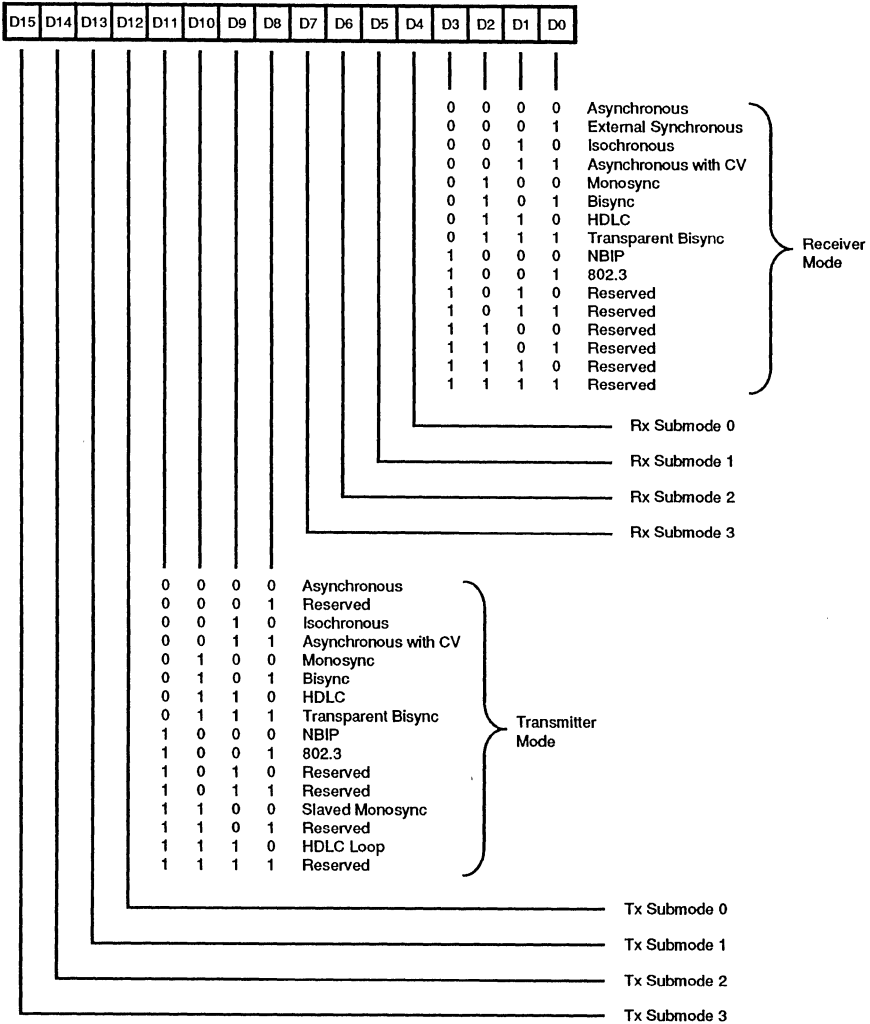


Figure 8. Channel Mode Register





Address: 00001

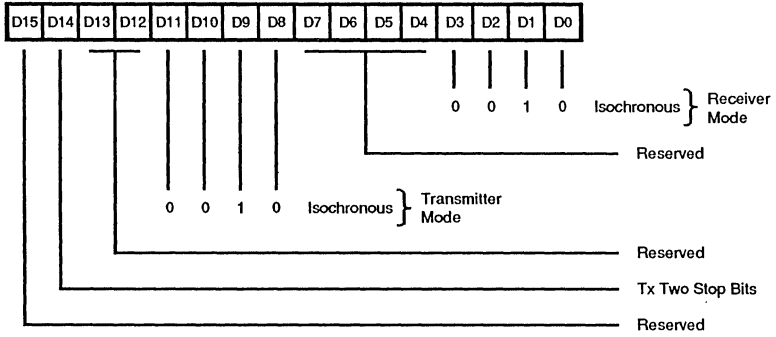


Figure 11. Channel Mode Register, Isochronous Mode

Address: 00001

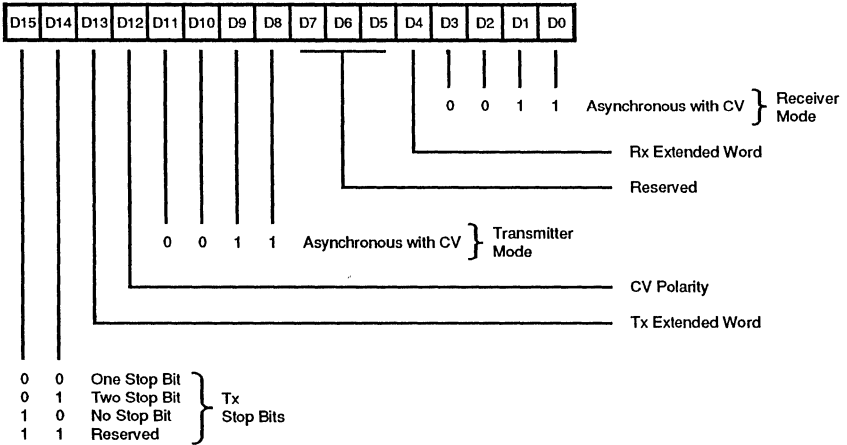


Figure 12. Channel Mode Register, Asynchronous Mode with Code Violation (MIL STD 1553)

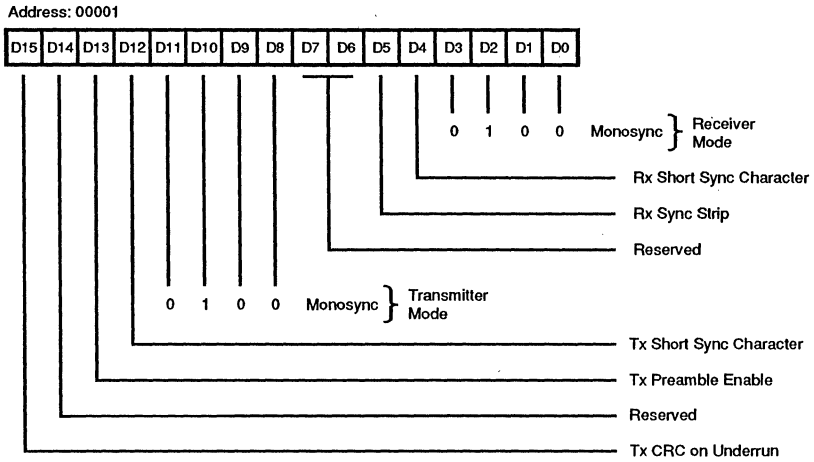


Figure 13. Channel Mode Register, Monosync Mode

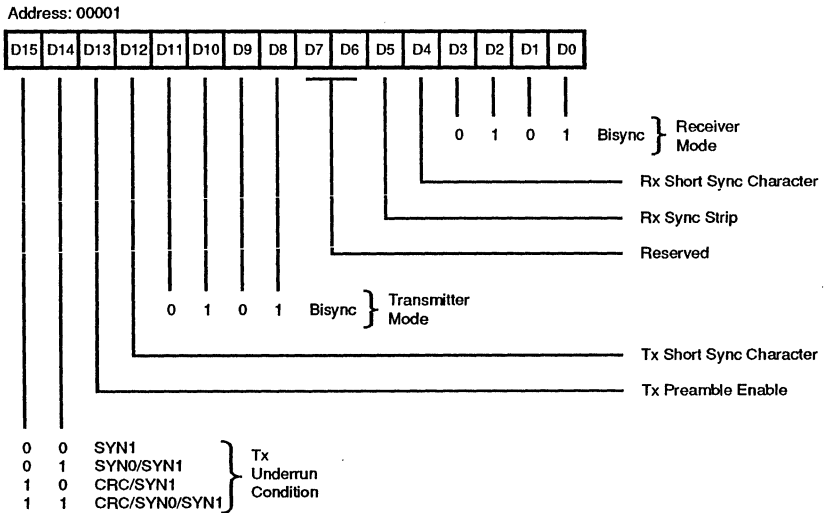


Figure 14. Channel Mode Register, Bisync Mode

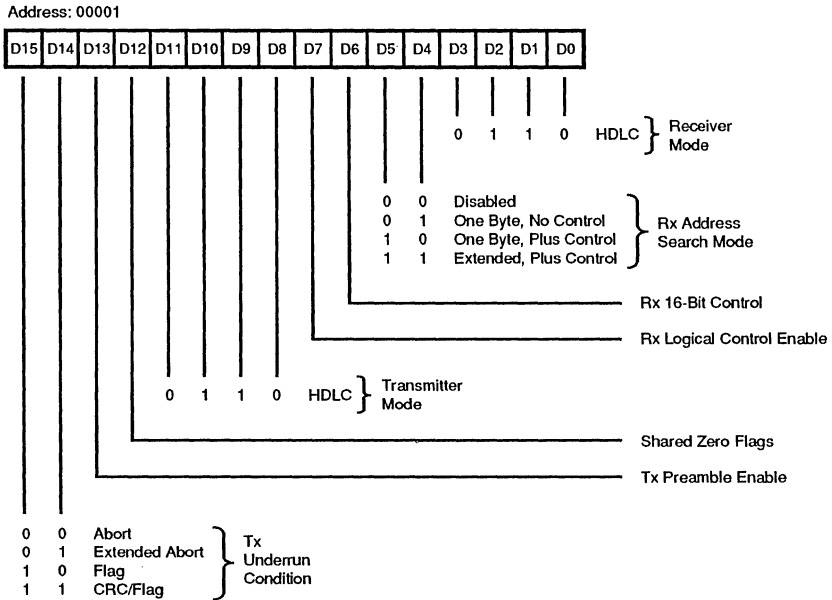


Figure 15. Channel Mode Register, HDLC Mode

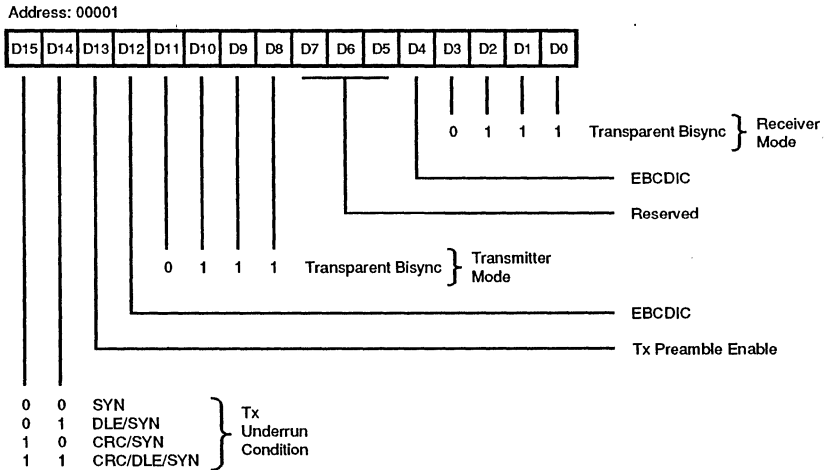


Figure 16. Channel Mode Register, Transparent Bisync Mode

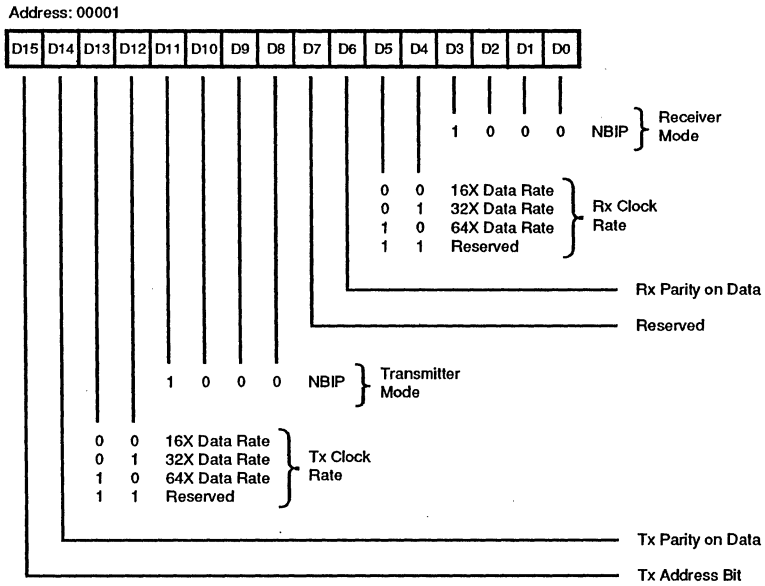


Figure 17. Channel Mode Register, NBIP Mode

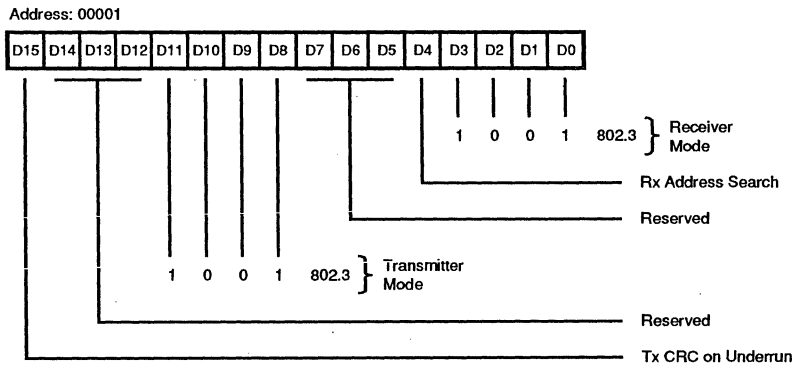


Figure 18. Channel Mode Register, 802.3 Mode

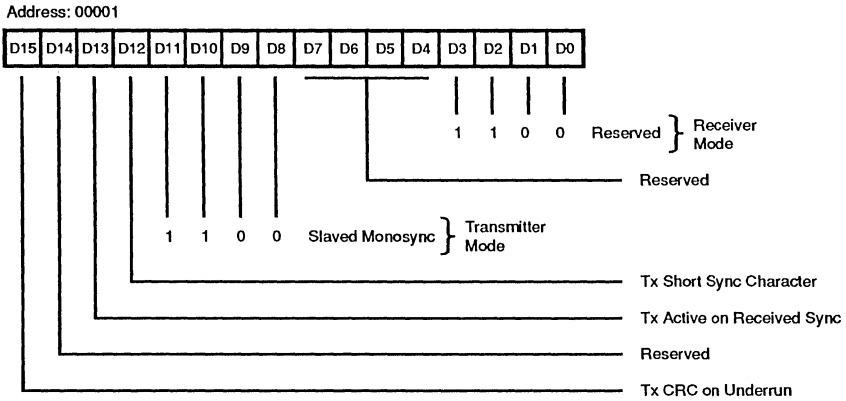


Figure 19. Channel Mode Register, Slaved Monosync Mode

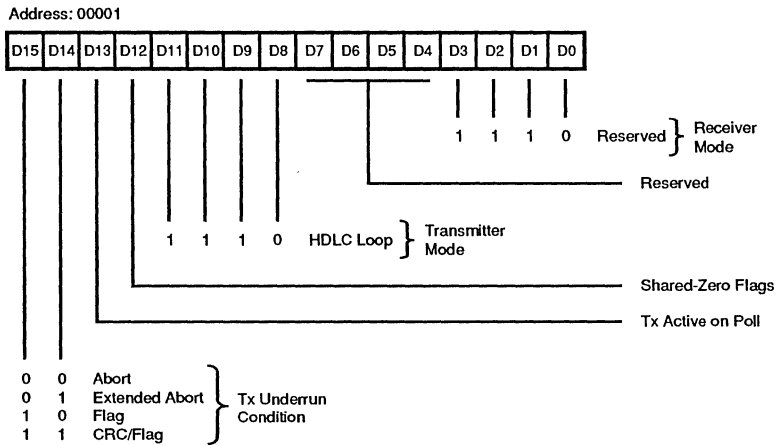


Figure 20. Channel Mode Register, HDLC Loop Mode

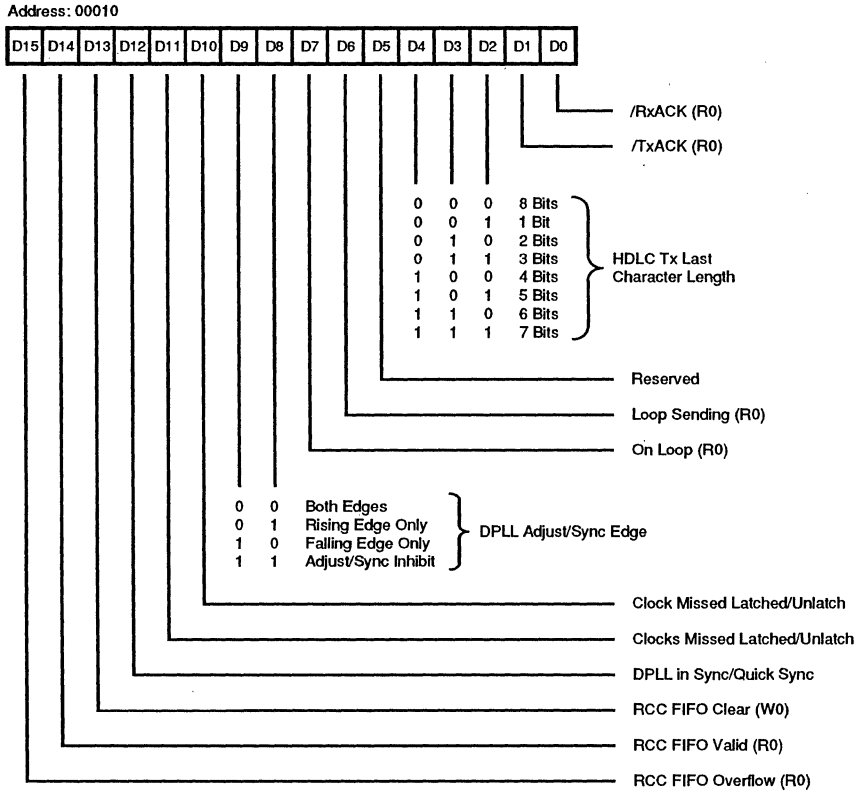


Figure 21. Channel Command/Status Register

Address: 00011

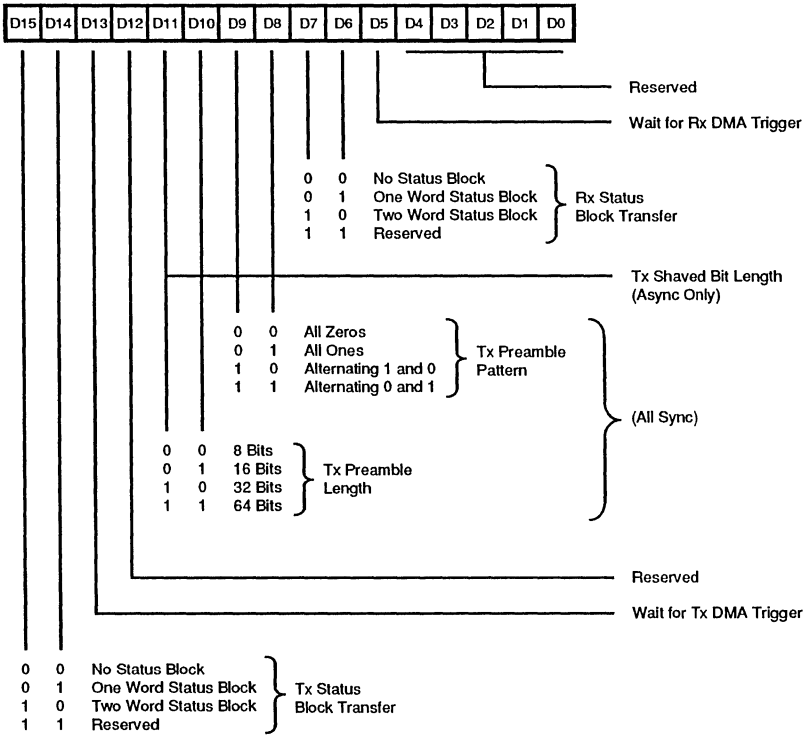


Figure 22. Channel Control Register



Address: 00100

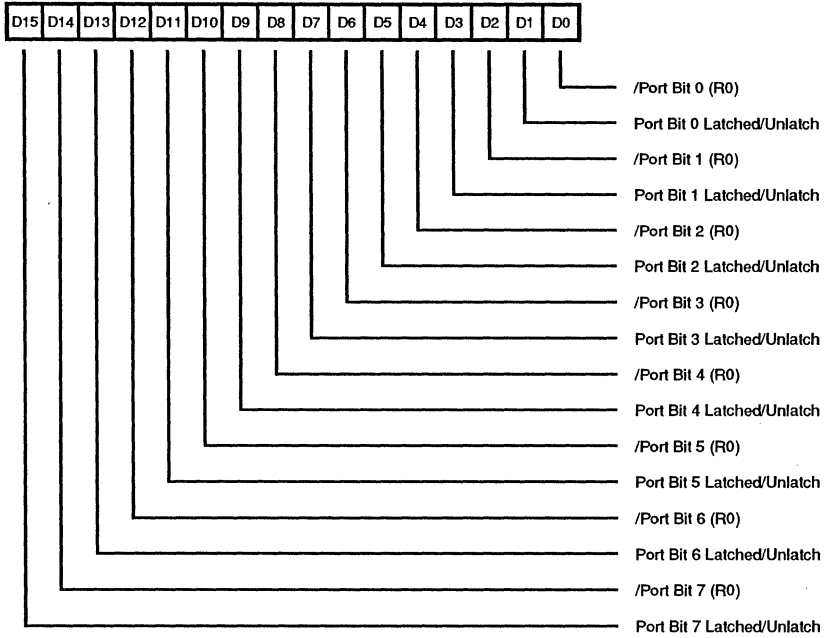


Figure 23. Port Status Register

Address: 00101

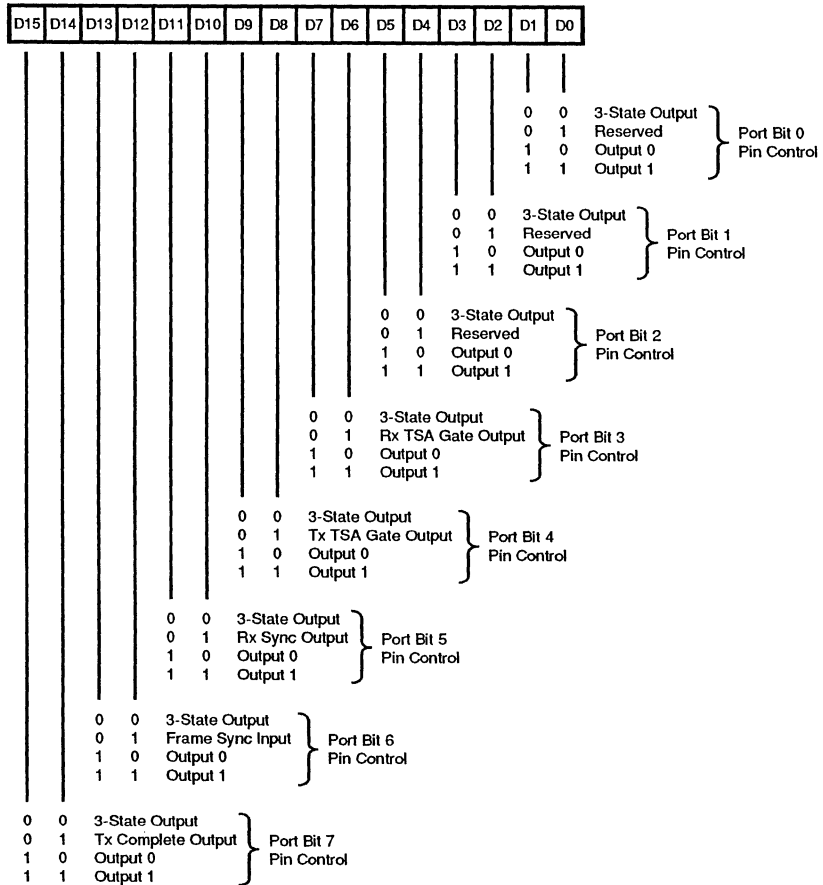


Figure 24. Port Control Register

Address: 00110

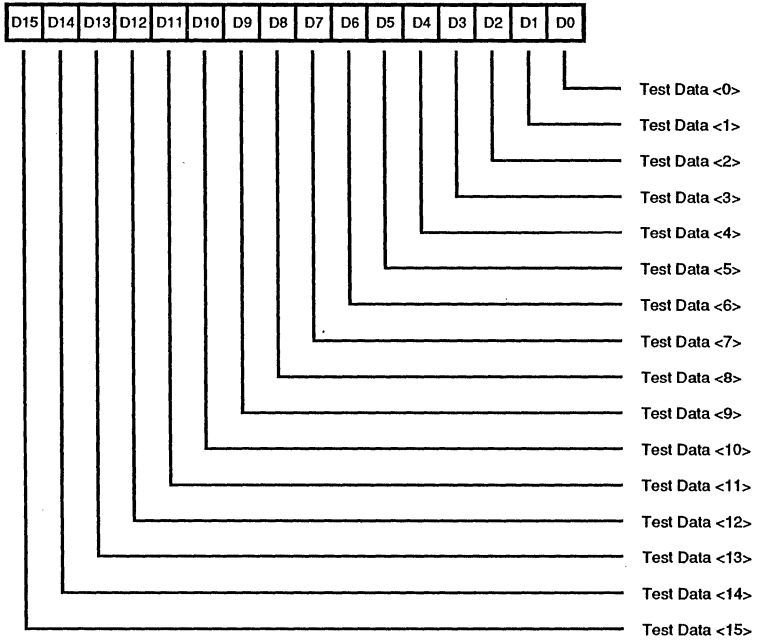


Figure 25. Test Mode Data Register

Address: 00111

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
											0	0	0	0	0	Null Address
											0	0	0	0	1	High Byte of Shifters
											0	0	0	1	0	CRC Byte 0
											0	0	0	1	1	CRC Byte 1
											0	0	1	0	0	Rx FIFO (Write)
											0	0	1	0	1	Clock Multiplexer Outputs
											0	0	1	1	0	CTR0 and CTR1 Counters
											0	0	1	1	1	Clock Multiplexer Inputs
											0	1	0	0	0	DPLL State
											0	1	0	0	1	Low Byte of Shifters
											0	1	0	1	0	CRC Byte 2
											0	1	0	1	1	CRC Byte 3
											0	1	1	0	0	Tx FIFO (Read)
											0	1	1	0	1	Reserved
											0	1	1	1	0	I/O and Device Status Latches
											0	1	1	1	1	Internal Daisy Chain
											1	0	0	0	0	Reserved
											1	0	0	0	1	Reserved
											1	0	0	1	0	Reserved
											1	0	0	1	1	Reserved
											1	0	1	0	0	Reserved
											1	0	1	0	1	Reserved
											1	0	1	1	0	Rx Count Holding Register
											1	0	1	1	1	Reserved
											1	1	0	0	0	Reserved
											1	1	0	0	1	Reserved
											1	1	0	1	0	Reserved
											1	1	0	1	1	Reserved
											1	1	1	0	0	Reserved
											1	1	1	0	1	Reserved
											1	1	1	1	0	Reserved
											1	1	1	1	1	Reserved
											Reserved					

Figure 26. Test Mode Control Register

Address: 01000

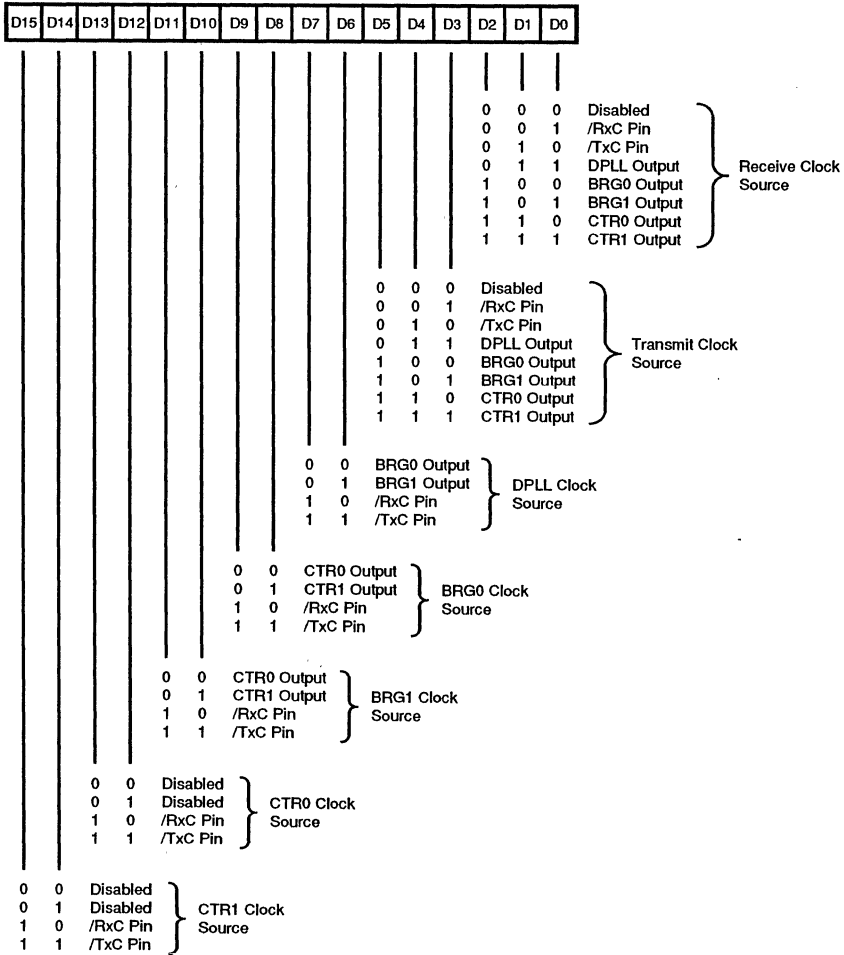


Figure 27. Clock Mode Control Register

Address: 01001

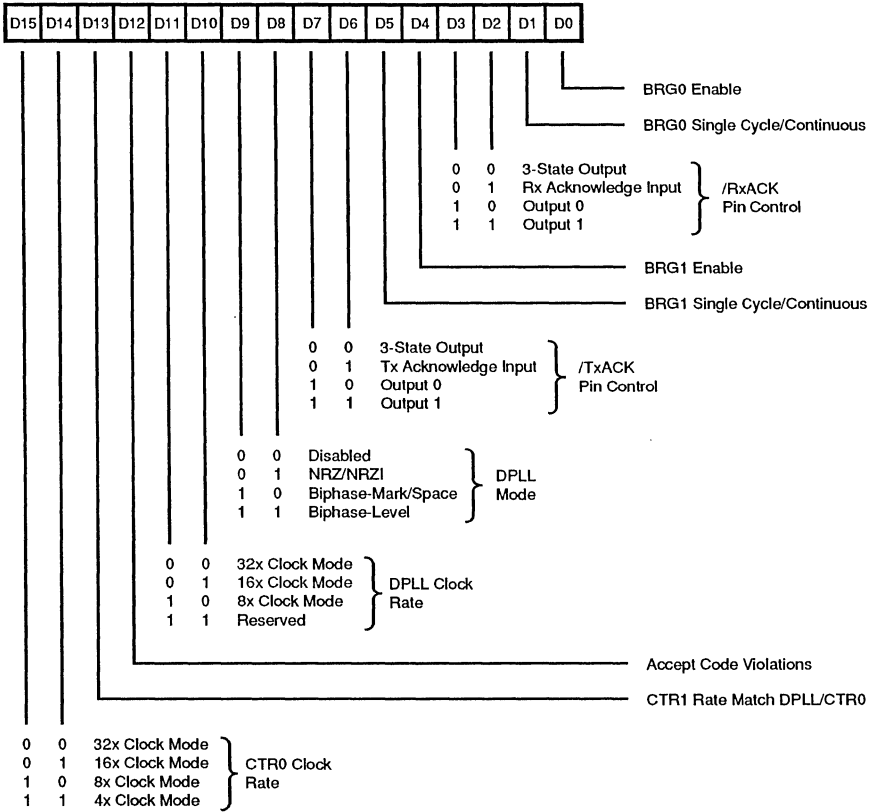


Figure 28. Hardware Configuration Register

Address: 01010

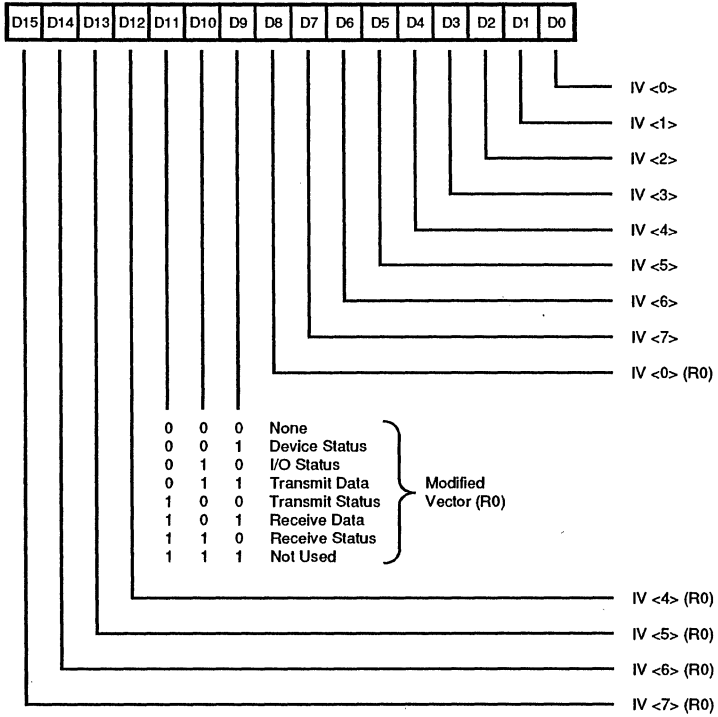


Figure 29. Interrupt Vector Register

Address: 01011

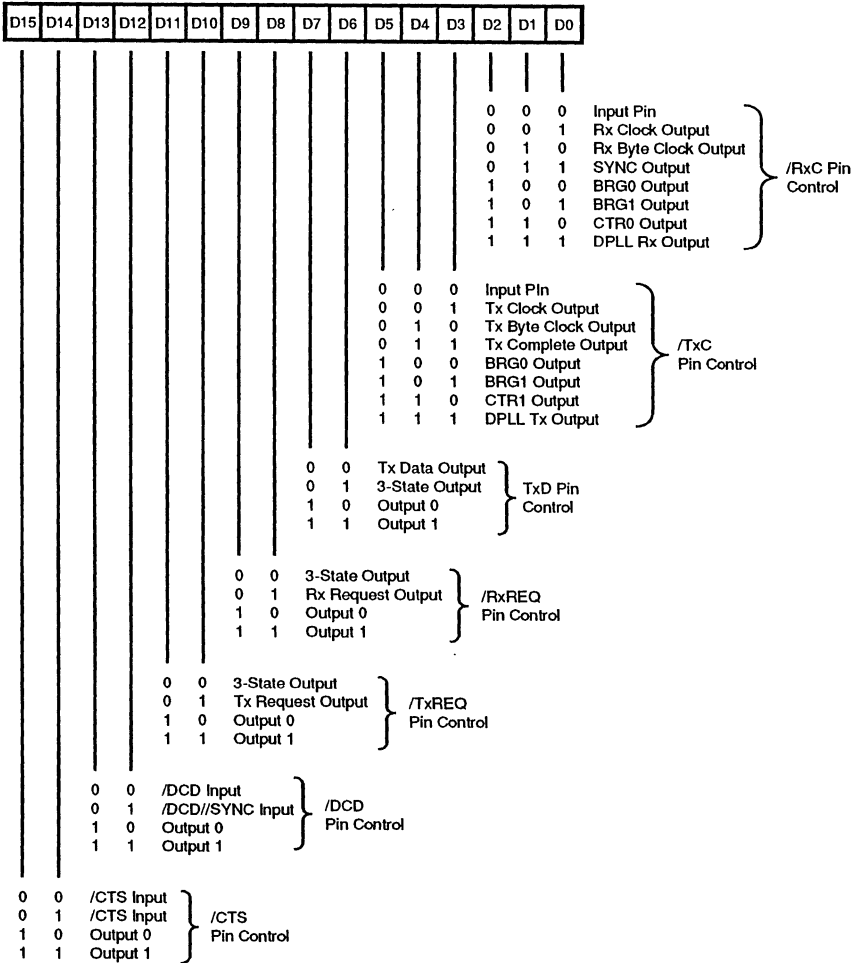


Figure 30. I/O Control Register



Address: 01100

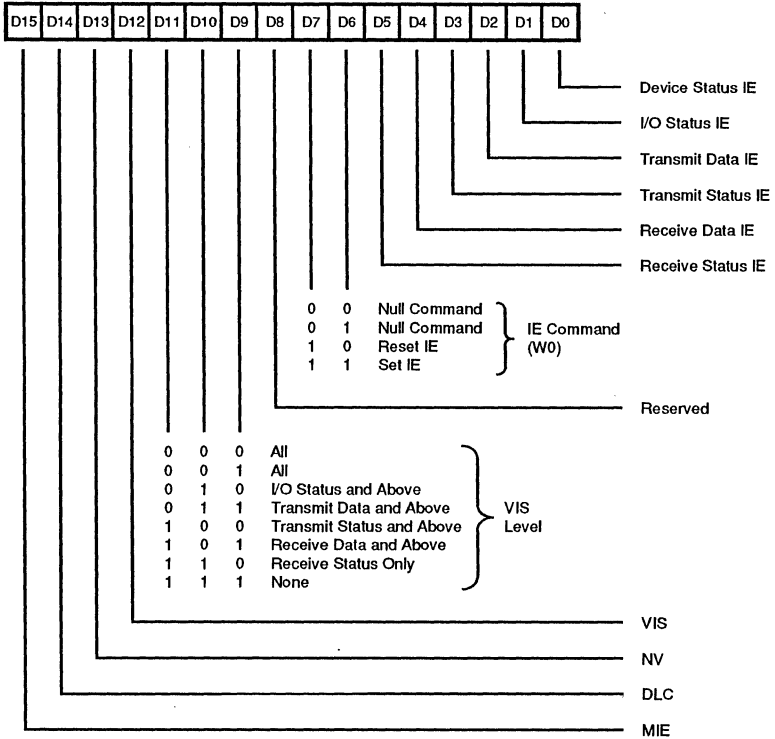


Figure 31. Interrupt Control Register

Address: 01101

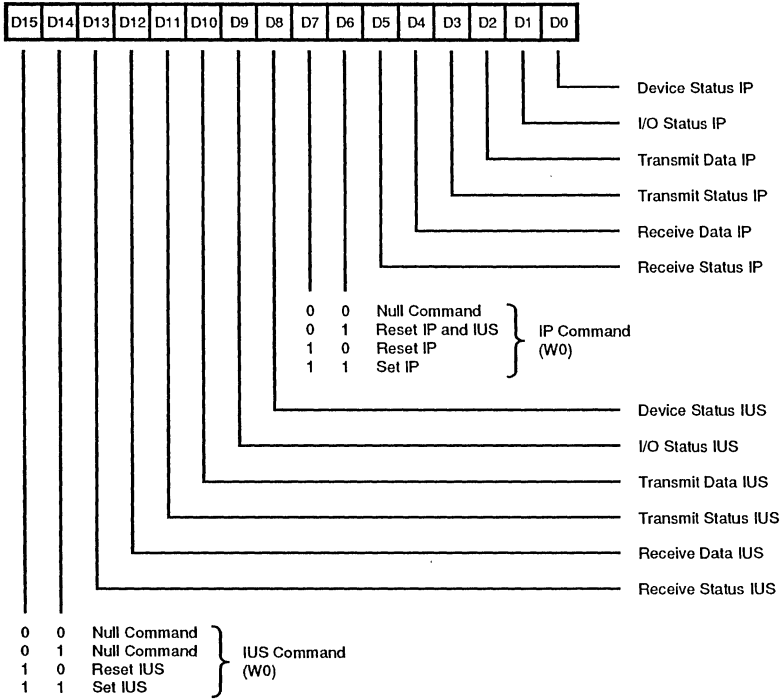


Figure 32. Daisy-Chain Control Register

Address: 01110

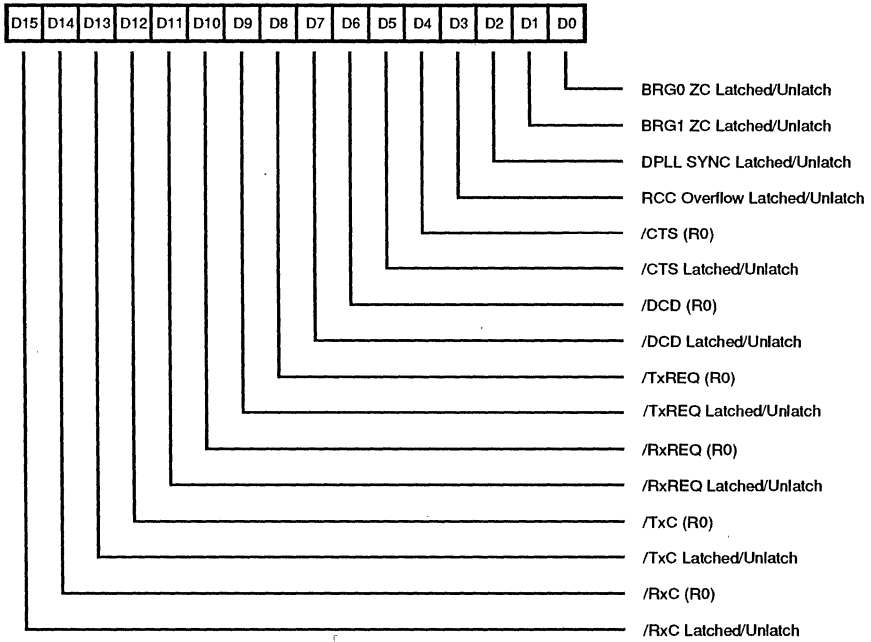


Figure 33. Miscellaneous Interrupt Status Register

Address: 01111

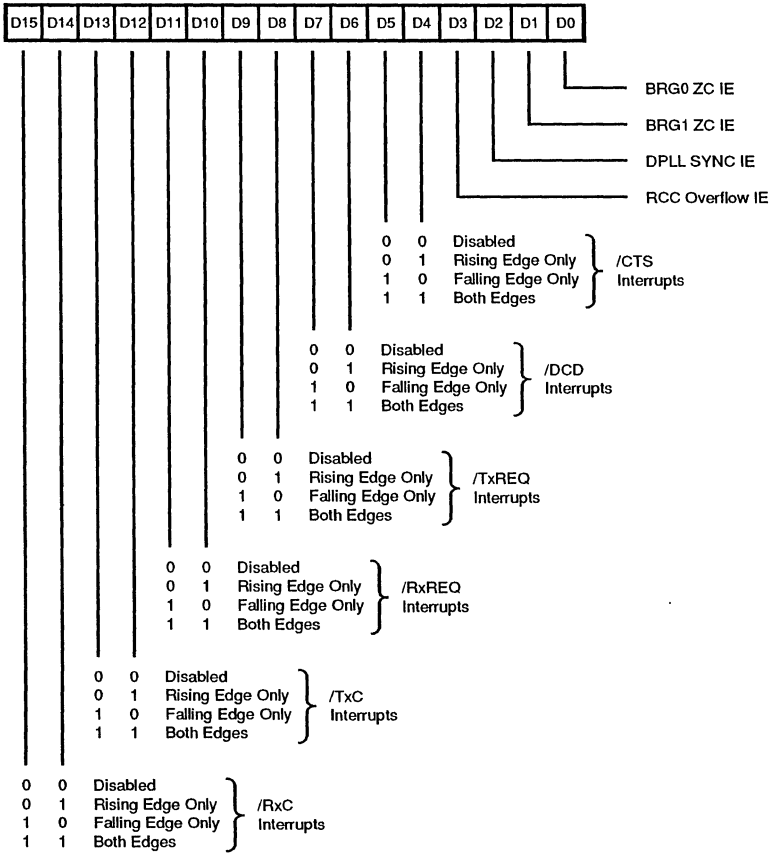


Figure 34. Status Interrupt Control Register

Address: 1x000

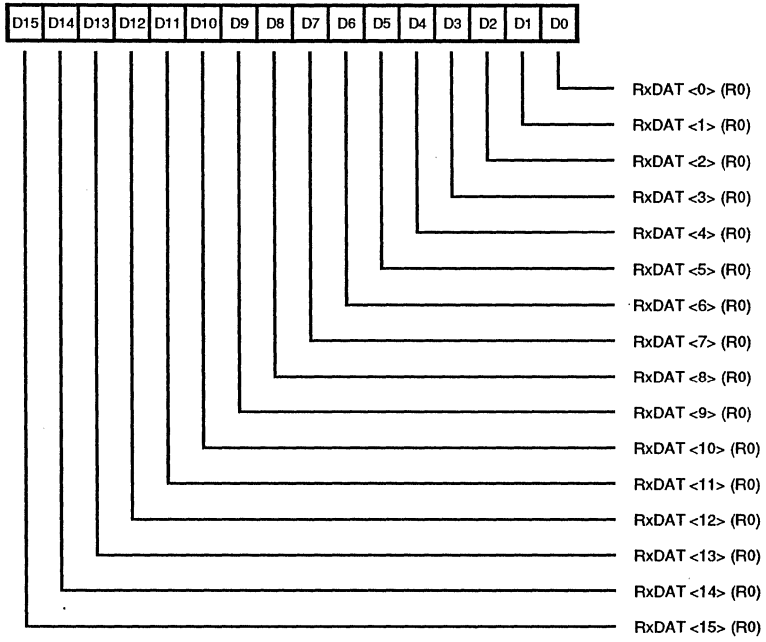


Figure 35. Receive Data Register

Address: 10001

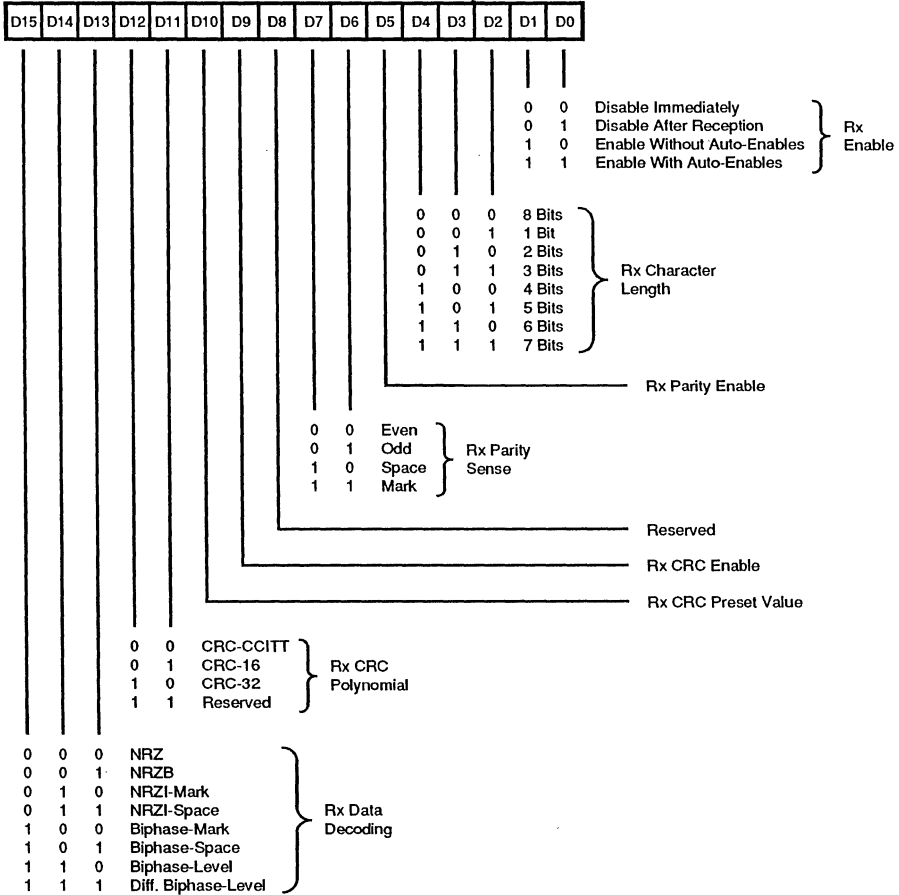


Figure 36. Receive Mode Register

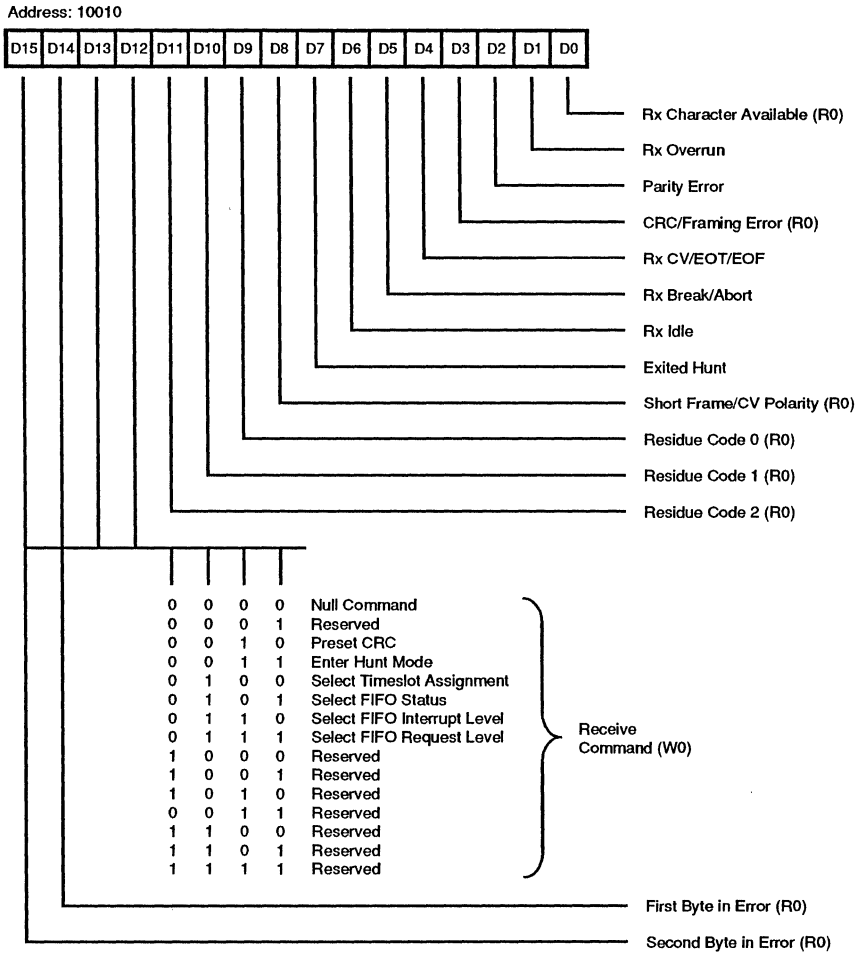


Figure 37. Receive Command Status Register

Address: 10011

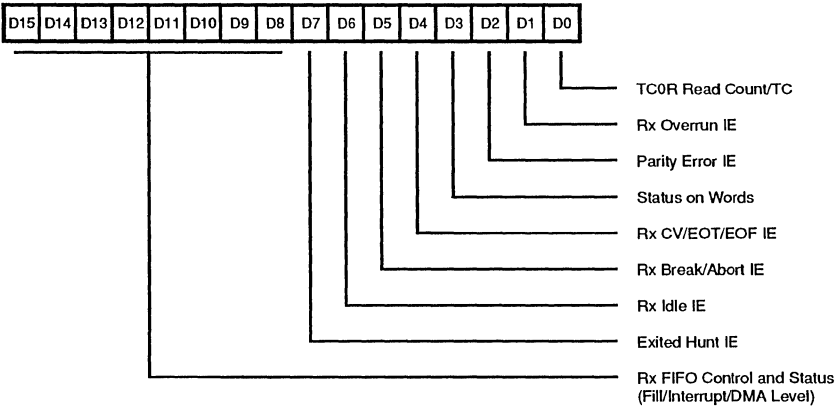


Figure 38a. Receive Interrupt Control Register

Address: 10011

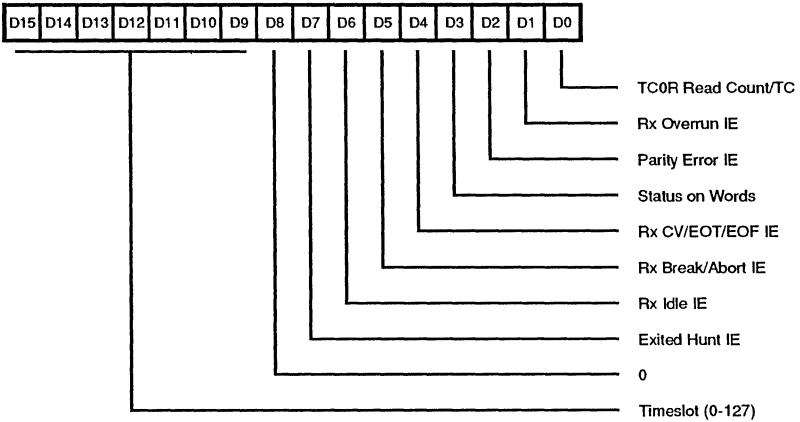


Figure 38b. Receive Interrupt Control Register



Address: 10011

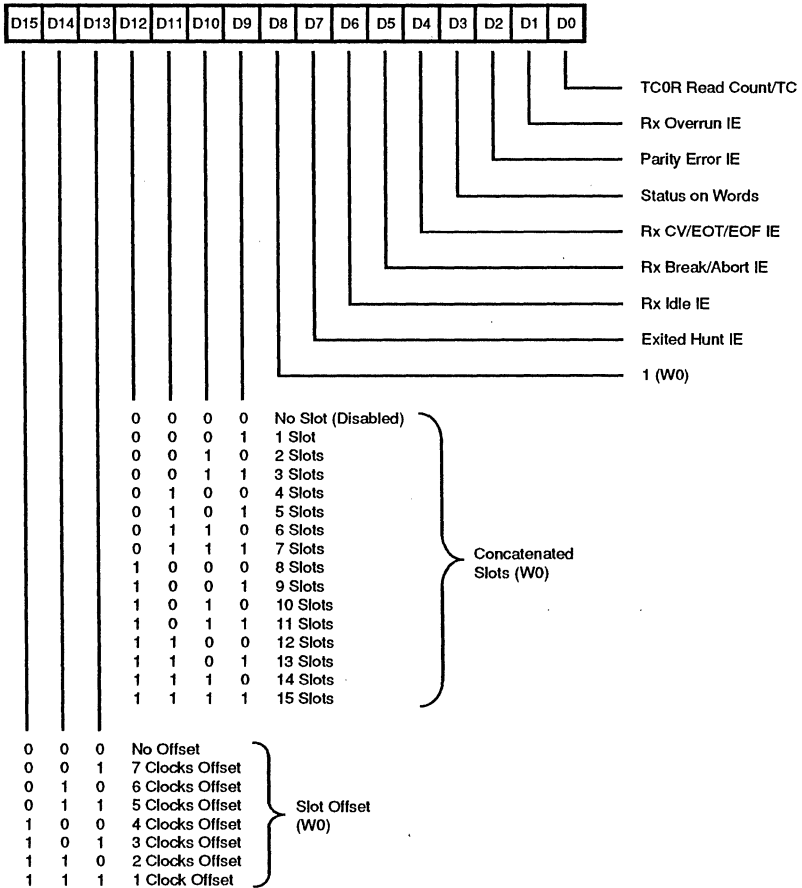


Figure 38c. Receive Interrupt Control Register

Address: 10100

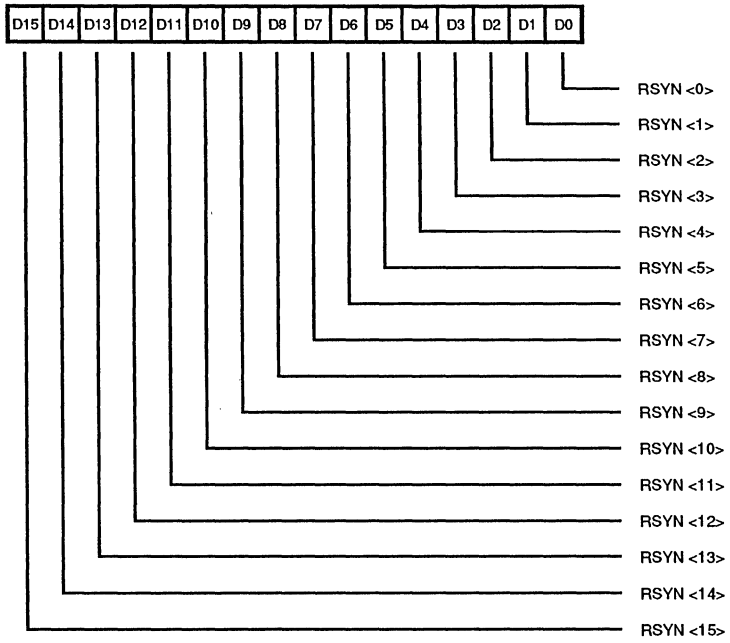


Figure 39. Receive Sync Register

Address: 10101

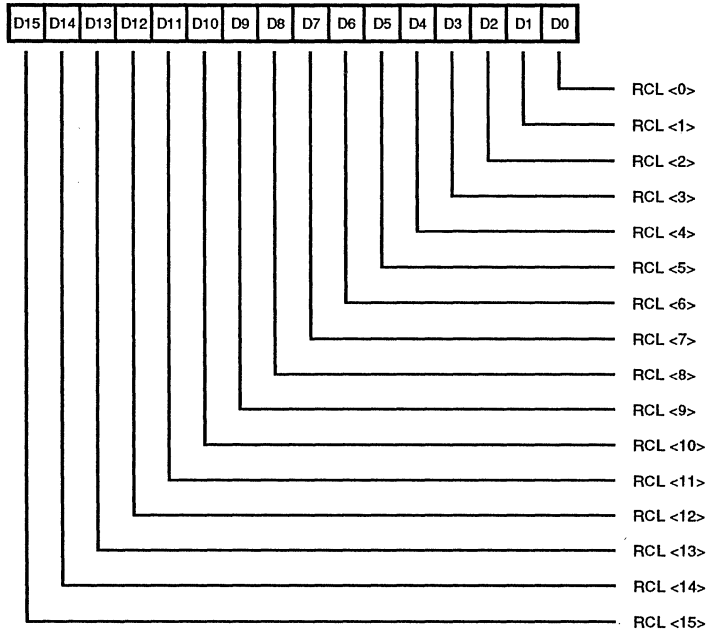


Figure 40. Receive Count Limit Register

Address: 10110

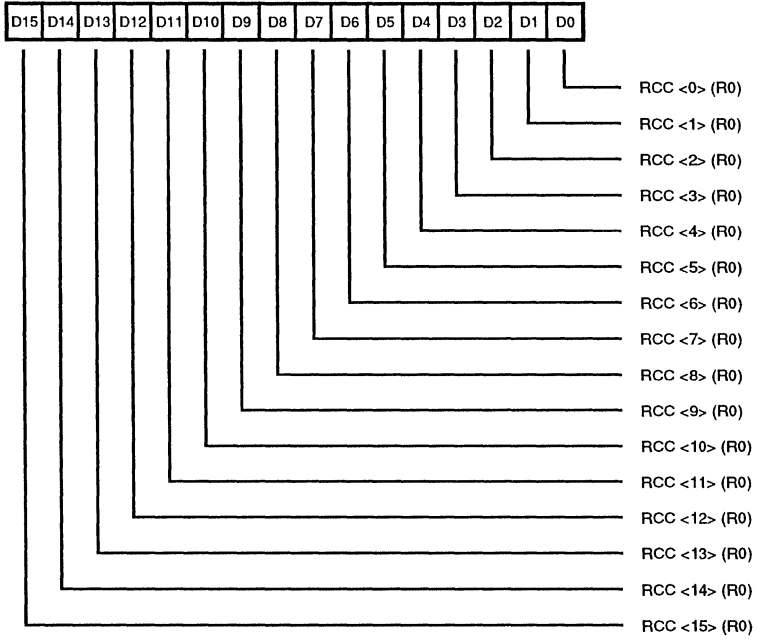


Figure 41. Receive Character Count Register

Address: 10111

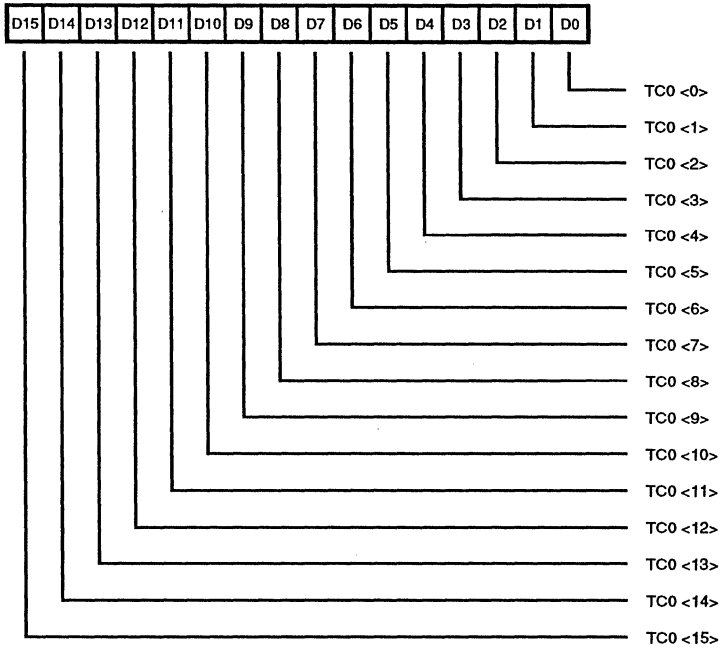


Figure 42. Time Constant 0 Register

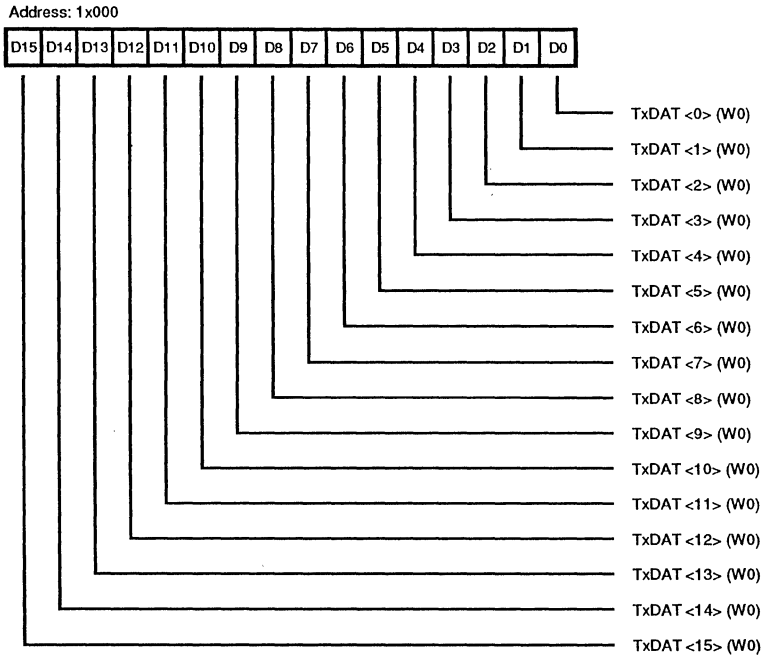


Figure 43. Transmit Data Register







Address: 11011

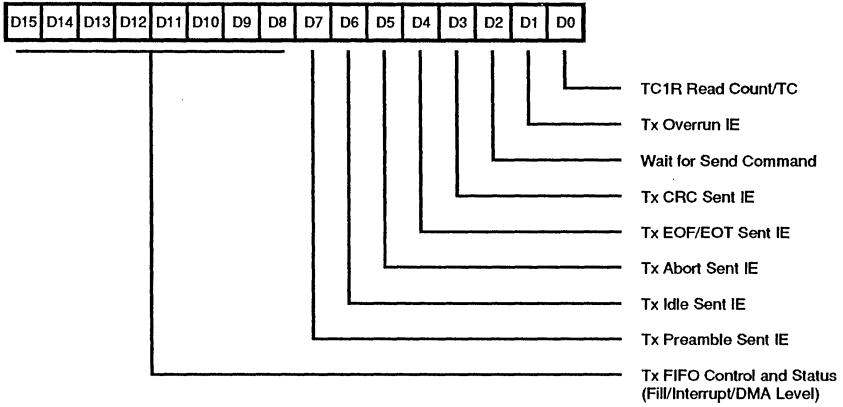


Figure 46a. Transmit Interrupt Control Register

Address: 11011

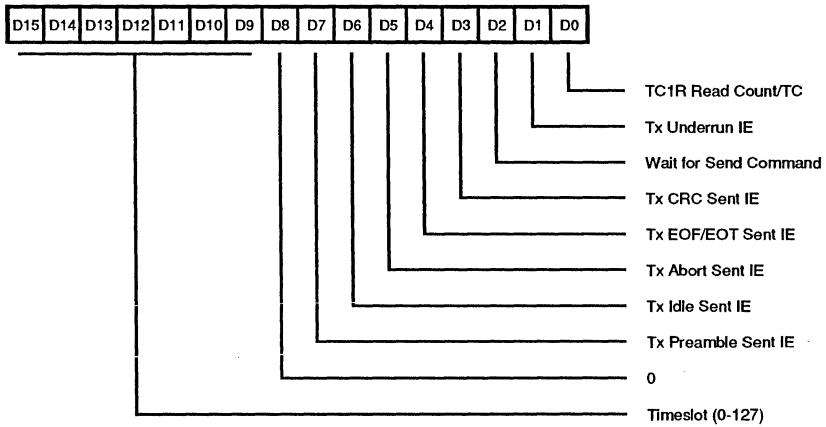


Figure 46b. Transmit Interrupt Control Register

Address: 11011

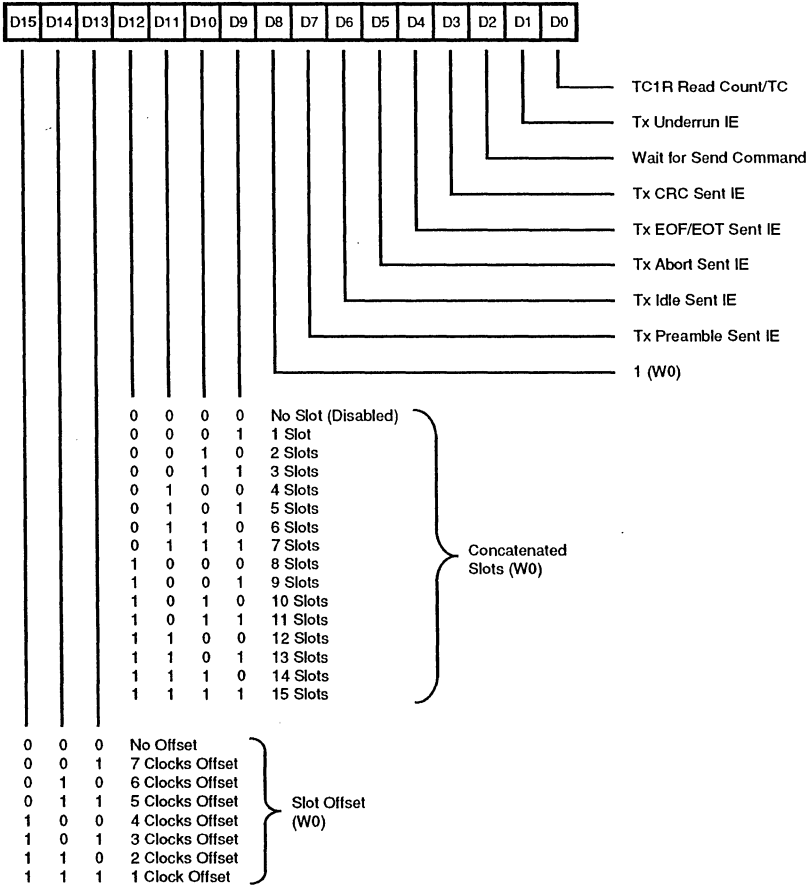


Figure 46c. Transmit Interrupt Control Register

Address: 11100

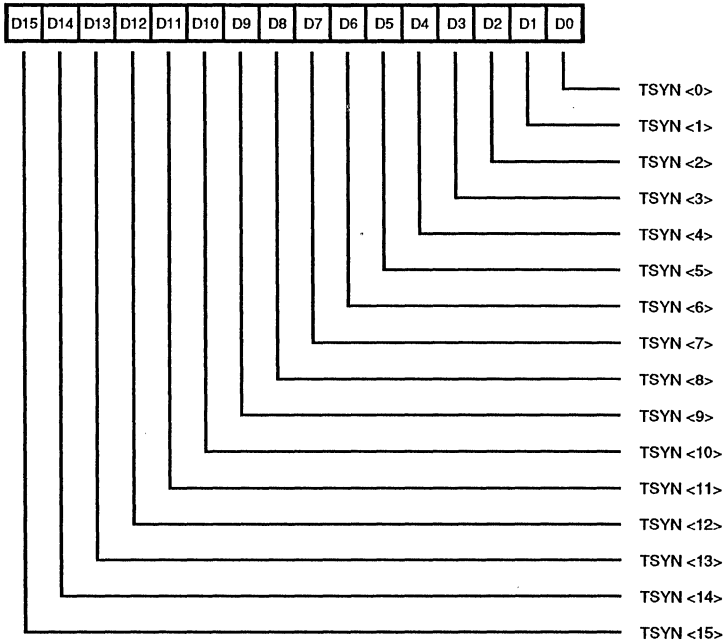


Figure 47. Transmit Sync Register

Address: 11101

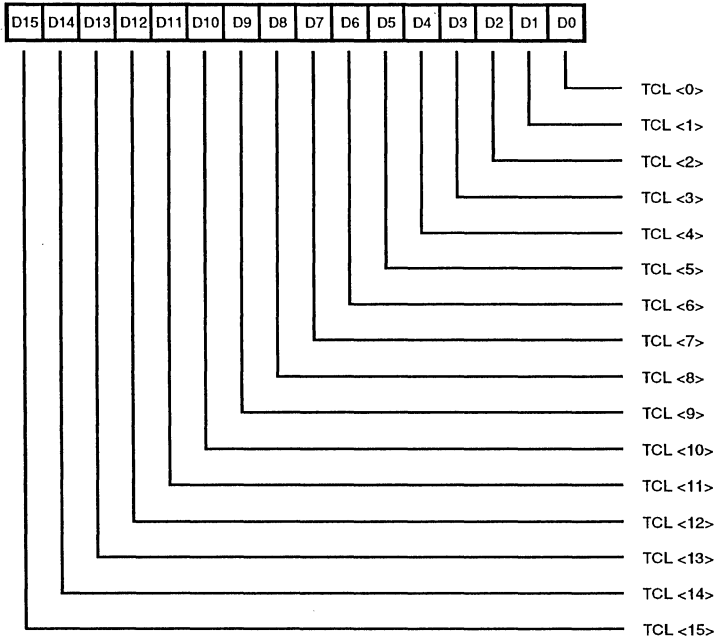


Figure 48. Transmit Count Limit Register

Address: 11110

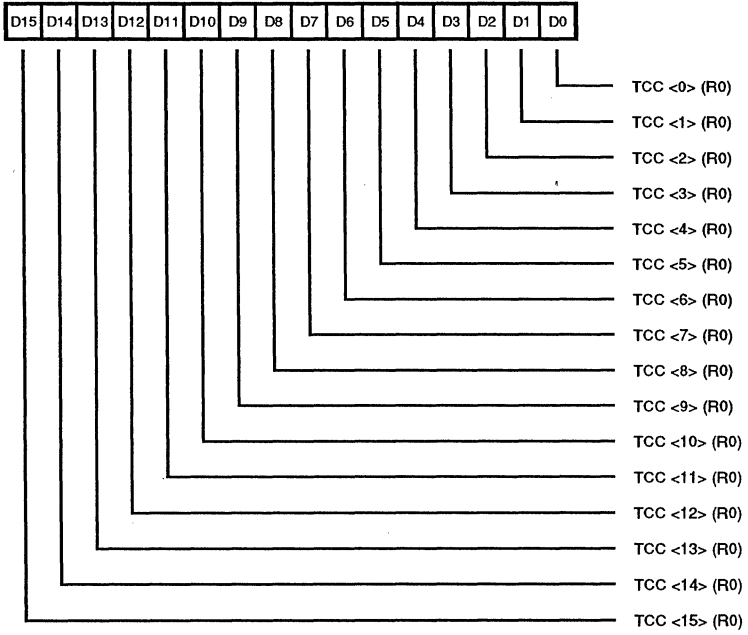


Figure 49. Transmit Character Count Register

Address: 11111

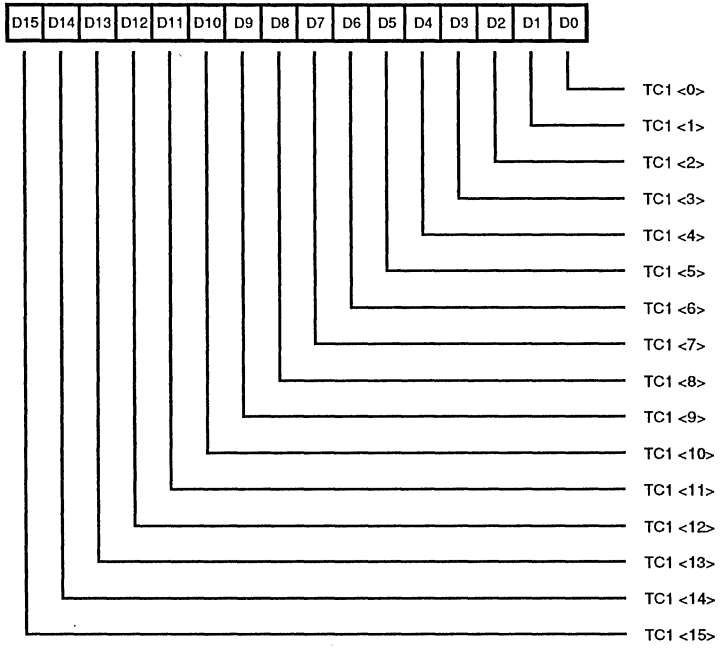
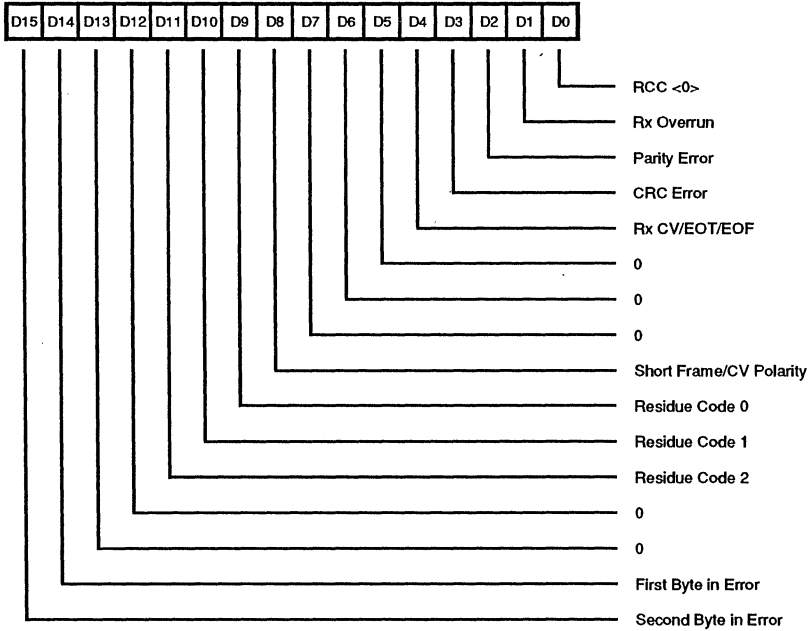


Figure 50. Time Constant 1 Register

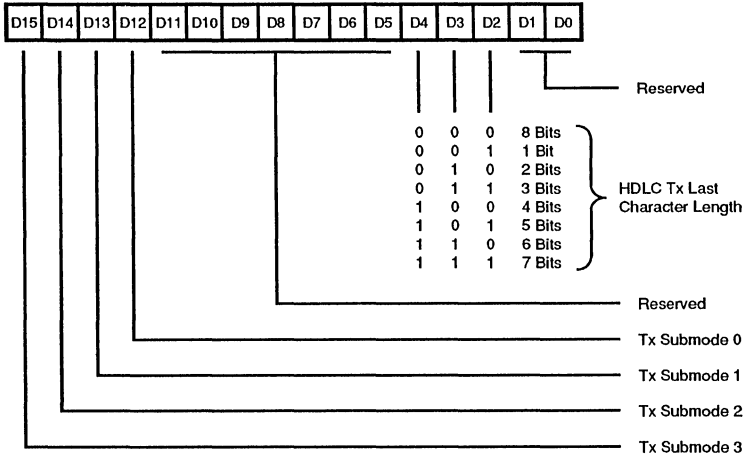
Address: None \*



\* Refer to Figure 22 (Channel Control Register)  
Bits 6-7 for Access Method

Figure 51. Receive Status Block Register

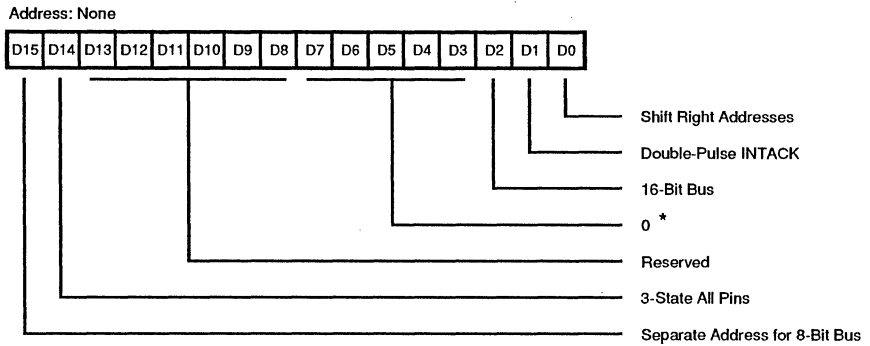
Address: None \*



\* Refer to Figure 22 (Channel Control Register) Bits15-14 for Access Method

Figure 52. Transmit Status Block Register





\* Must be programmed as zero.

**Figure 53. Bus Configuration Register**

## MUSC TIMING

The MUSC interface timing is similar to that found on a static RAM, except that it is much more flexible. Up to eight separate timing strobe signals are present on the interface: /DS, /RD, /WR, /PITACK, /RxACKA, /RxACKB, /TxACKA and /TxACKB. Only one of these timing strobes may be active at any time. Should the external logic

activate more than one of these strobes at the same time the MUSC will enter a pre-reset state. This state is only exited by a hardware reset. Do not allow overlap of timing strobes. The timing diagrams, beginning on the next page, illustrate the different bus transactions possible, with the necessary setup, hold and delay times.

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins  
 with respect to Vss ..... -0.3 V to +7.0 V  
 Voltages on all inputs  
 with respect to Vss ..... -0.3V to Vcc +0.3V  
 Operating Ambient  
 Temperature ..... See Ordering Information  
 Storage Temperature ..... -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance section below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

$$+4.5\text{ V} < V_{CC} < +5.5\text{ V}$$

$$\text{GND} = 0\text{ V}$$

$T_A$  as specified in Ordering Information

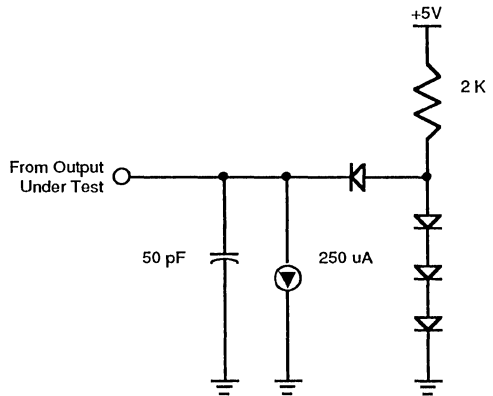


Figure 54. Standard Test Load

## CAPACITANCE

Symbol	Parameter	Min	Max	Unit	Condition
$C_{IN}$	Input Capacitance		10	pF	Unmeasured Pins
$C_{OUT}$	Output Capacitance		15	pF	Returned to Ground
$C_{VO}$	Bidirectional Capacitance		20	pF	

**Note:**

$f = 1\text{ MHz}$ , over specified temperature range. Unmeasured pins returned to ground.

## MISCELLANEOUS Transistor Count - 100,000

## DC CHARACTERISTICS

Z16C33

Symbol	Parameter	Min	Typ	Max	Unit	Condition
$V_{IH}$	Input High Voltage	2.2		$V_{CC} + 0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3		0.8	V	
$V_{OH1}$	Output High Voltage	2.4			V	$I_{OH} = -1.6\text{ mA}$
$V_{OH2}$	Output High Voltage	$V_{CC} - 0.8$			V	$I_{OH} = -250\text{ }\mu\text{A}$
$V_{OL}$	Output Low Voltage			0.4	V	$I_{OL} = +2.0\text{ mA}$
$I_{IL}$	Input Leakage			+10.00	$\mu\text{A}$	$0.4 < V_{IN} < +2.4\text{ V}$
$I_{OL}$	Output Leakage			+10.00	$\mu\text{A}$	$0.4 < V_{OUT} < +2.4\text{ V}$
$I_{CC1}$	$V_{CC}$ Supply Current	7		50	mA	$V_{CC} = 5\text{ V}$ $V_{IH} = 4.8\text{ V}$ $V_{IL} = 0.2\text{ V}$

**Note:**

$V_{CC} = 5\text{ V} \pm 10\%$  unless otherwise specified, over specified temperature range.

## AC CHARACTERISTICS

Z16C33

No	Symbol	Parameter	Min	Max	Units	Note
1	Tcyc	Bus Cycle Time	160		ns	
2	TwASl	/AS Low Width	40		ns	
3	TwASh	/AS High Width	90		ns	
4	TwDSl	/DS Low Width	70		ns	
5	TwDSH	/DS High Width	60		ns	
6	TdAS(DS)	/AS Rise to /DS Fall Delay Time	5		ns	
7	TdDS(AS)	/DS Rise to /AS Fall Delay Time	5		ns	
8	TdDS(DRa)	/DS Fall to Data Active Delay	0		ns	
9	TdDS(DRv)	/DS Fall to Data Valid Delay		85	ns	
10	TdDS(DRn)	/DS Rise to Data Not Valid Delay	0		ns	
11	TdDS(DRz)	/DS Rise to Data Float Delay		20	ns	
12	TsCS(AS)	/CS to /AS Rise Setup Time	15		ns	
13	ThCS(AS)	/CS to /AS Rise Hold Time	0		ns	
14	TsADD(AS)	Direct Address to /AS Rise Setup Time	15		ns	[1]
15	ThADD(AS)	Direct Address to /AS Rise Hold Time	5		ns	[1]
16	TsSIA(AS)	/SITACK to /AS Rise Setup Time	15		ns	
17	ThSIA(AS)	/SITACK to /AS Rise Hold Time	5		ns	
18	TsAD(AS)	Address to /AS Rise Setup Time	15		ns	
19	ThAD(AS)	Address to /AS Rise Hold Time	5		ns	
20	TsRW(DS)	R//W to /DS Fall Setup Time	0		ns	
21	ThRW(DS)	R//W to /DS Fall Hold Time	25		ns	
22	TsDSf(RRQ)	/DS Fall to /RxREQ Inactive Delay		60	ns	[4]
23	TdDSr(RRQ)	/DS Rise to /RxREQ Active Delay	0		ns	
24	TsDW(DS)	Write Data to /DS Rise Setup Time	30		ns	
25	ThDW(DS)	Write Data to DS Rise Hold Time	0		ns	
26	TdDSf(TRQ)	/DS Fall to /TxREQ Inactive Delay		65	ns	[5]
27	TdDSr(TRQ)	/DS Rise to /TxREQ Active Delay	0		ns	
28	TwRDI	/RD Low Width	70		ns	
29	TwRDh	/RD High Width	60		ns	
30	TdAS(RD)	/AS Rise to /RD Fall Delay Time	5		ns	
31	TdRD(AS)	/RD Rise to /AS Fall Delay Time	5		ns	
32	TdRD(DRa)	/RD Fall to Data Active Delay	0		ns	
33	TdRD(DRv)	/RD Fall to Data Valid Delay		85	ns	
34	TdRD(DRn)	/RD Rise to Data Not Valid Delay	0		ns	
35	TdRD(DRz)	/RD Rise to Data Float Delay		20	ns	
36	TdRDf(RRQ)	/RD Fall to /RxREQ Inactive Delay		60	ns	[4]
37	TdRDr(RRQ)	/RD Rise to /RxREQ Active Delay	0		ns	
38	TwWRI	/WR Low Width	70		ns	
39	TwWRh	/WR High Width	60		ns	
40	TdAS(WR)	/AS Rise to /WR Fall Delay Time	5		ns	
41	TdWR(AS)	/WR Rise to /AS Fall Delay Time	5		ns	
42	TsDW(WR)	Write Data to /WR Rise Setup Time	30		ns	
43	ThDW(WR)	Write Data to /WR Rise Hold Time	0		ns	
44	TdWRf(TRQ)	/WR Fall to /TxREQ Inactive Delay		65	ns	[5]

## AC CHARACTERISTICS

Z16C33

No	Symbol	Parameter	Min	Max	Units	Note
45	TdWRr(TRQ)	/WR Rise to /TxREQ Active Delay	0		ns	
46	TsCS(DS)	/CS to /DS Fall Setup Time	0		ns	[2]
47	ThCS(DS)	/CS to /DS Fall Hold Time	25		ns	[2]
48	TsADD(DS)	Direct Address to /DS Fall Setup Time	5		ns	[1,2]
49	ThADD(DS)	Direct Address to /DS Fall Hold Time	25		ns	[1,2]
50	TsSIA(DS)	/SITACK to /DS Fall Setup time	5		ns	[2]
51	ThSIA(DS)	/SITACK to /DS Fall Hold Time	25		ns	[2]
52	TsCS(RD)	/CS to /RD Fall Setup Time	0		ns	[2]
53	ThCS(RD)	/CS to /RD Fall Hold Time	25		ns	[2]
54	TsADD(RD)	Direct Address to /RD Fall Setup Time	5		ns	[1,2]
55	ThADD(RD)	Direct Address to /RD Fall Hold Time	25		ns	[1,2]
56	TsSIA(RD)	/SITACK to /RD Fall Setup Time	5		ns	[2]
57	ThSIA(RD)	/SITACK to /RD Fall Hold Time	25		ns	[2]
58	TsCS(WR)	/CS to /WR Fall Setup Time	0		ns	[2]
59	ThCS(WR)	/CS to /WR Fall Hold Time	25		ns	[2]
60	TsADD(WR)	Direct Address to /WR Fall Setup Time	5		ns	[1,2]
61	ThADD(WR)	Direct Address to /WR Fall Hold Time	25		ns	[1,2]
62	TsSIA(WR)	/SITACK to /WR Fall Setup Time	5		ns	[2]
63	ThSIA(WR)	/SITACK to /WR Fall Hold Time	25		ns	[2]
64	TwRAKI	/RxACK Low Width	70		ns	
65	TwRAKh	/RxACK High Width	60		ns	
66	TdRAK(DRa)	/RxACK Fall to Data Active Delay	0		ns	
67	TdRAK(DRv)	/RxACK Fall to Data Valid Delay		85	ns	
68	TdRAK(DRn)	/RxACK Rise to Data Not Valid Delay	0		ns	
69	TdRAK(DRz)	/RxACK Rise to Data Float Delay		20	ns	
70	TdRAKI(RRQ)	/RxACK Fall to /RxREQ Inactive Delay		60	ns	[4]
71	TdRAKr(RRQ)	/RxACK Rise to /RxREQ Active Delay	0		ns	
72	TwTAKI	/TxACK Low Width	70		ns	
73	TwTAKh	/TxACK High Width	60		ns	
74	TsDW(TAK)	Write Data to /TxACK Rise Setup Time	30		ns	
75	ThDW(TAK)	Write Data to /TxACK Rise Hold Time	0		ns	
76	TdTAKf(TRQ)	/TxACK Fall to /TxREQ Inactive Delay		65	ns	[5]
77	TdTAKr(TRQ)	/TxACK Rise to /TxREQ Active Delay	0		ns	
78	TdDSf(RDY)	/DS Fall (Intack) to /RDY Fall Delay		200	ns	
79	TdRDY(DRv)	/RDY Fall to Data Valid Delay		40	ns	
80	TdDSr(RDY)	/DS Rise to /RDY Rise Delay		40	ns	
81	TsIEI(DSI)	IEI to /DS Fall (Intack) Setup Time	60		ns	
82	ThIEI(DSI)	IEI to /DS Rise (Intack) Hold Time	0		ns	
83	TdIEI(IEO)	IEI to IEO Delay		60	ns	
84	TdAS(IEO)	/AS Rise (Intack) to IEO Delay		60	ns	
85	TdDSf(INT)	/DS Fall to /INT Inactive Delay		200	ns	
86	TdDSf(Wf)	/DS Fall (Intack) to /WAIT Fall Delay		40	ns	
87	TdDSf(Wr)	/DS Fall (Intack) to /WAIT Rise Delay		200	ns	
88	TdW(DRv)	/WAIT Rise to Data Valid Delay		40	ns	

## AC CHARACTERISTICS

Z16C33

No	Symbol	Parameter	Min	Max	Units	Note
89	TdRDI(RDY)	/RD Fall (Intack) to /RDY Fall Delay		200	ns	
90	TdRDf(RDY)	/RD Rise to /RDY Rise Delay		40	ns	
91	TsIEI(RDI)	IEI to /RD Fall (Intack) Setup Time	60		ns	
92	ThIEI(RDI)	IEI to /RD Rise (Intack) Hold Time	0		ns	
93	TdRDI(INT)	/RD Fall (Intack) to /INT Inactive Delay		200	ns	
94	TdRDI(Wf)	/RD Fall (Intack) to /WAIT Fall Delay		40	ns	
95	TdRDI(Wr)	/RD Fall (Intack) to /WAIT Rise Delay		200	ns	
96	TwPIAf	/PITACK Low Width	70		ns	
97	TwPIAh	/PITACK High Width	60		ns	
98	TdAS(PIA)	/AS Rise to /PITACK Fall Delay Time	5		ns	
99	TdPIA(AS)	/PITACK Rise to /AS Fall Delay Time	5		ns	
100	TdPIA(DRa)	/PITACK Fall to Data Active Delay	0		ns	
101	TdPIA(DRn)	/PITACK Rise to Data Not Valid Delay	0		ns	
102	TdPIA(DRz)	/PITACK Rise to Data Float Delay		20	ns	
103	TsIEI(PIA)	IEI to /PITACK Fall Setup Time	60		ns	
104	ThIEI(PIA)	IEI to /PITACK Rise Hold Time	0		ns	
105	TdPIA(IEO)	/PITACK Fall to IEO Delay		60	ns	
106	TdPIA(INT)	/PITACK Fall to /INT Inactive Delay		200	ns	
107	TdPIAf(RDY)	/PITACK Fall to /RDY Fall Delay		200	ns	
108	TdPIAr(RDY)	/PITACK Rise to /RDY Rise Delay		40	ns	
109	TdPIA(Wf)	/PITACK Fall to /WAIT Fall Delay		40	ns	
110	TdPIA(Wr)	/PITACK Fall to /WAIT Rise Delay		200	ns	
111	TdSIA(INT)	/SITACK Fall to IEO Inactive Delay		200	ns	[2]
112	TwSTBh	/Strobe High Width	60		ns	[3]
113	TwRESl	/RESET Low Width	170		ns	
114	TwRESH	/RESET High Width	60		ns	
115	TdRES(STB)	/RESET Rise to /STB Fall	60		ns	[3]
116	TdDSf(RDY)	/DS Fall to /RDY Fall Delay		50	ns	
117	TdWRf(RDY)	/WR Fall to /RDY Fall Delay		50	ns	
118	TdWRr(RDY)	/WR Rise to /RDY Rise Delay		40	ns	
119	TdRDI(RDY)	/RD Fall to /RDY Fall Delay		50	ns	
120	TdRAKf(RDY)	/RxAck Fall to /RDY Fall Delay		50	ns	
121	TdRAKr(RDY)	/RxAck Rise to /RDY Rise Delay		40	ns	
122	TdTAKf(RDY)	/TxACK Fall to /RDY Fall Delay		50	ns	
123	TdTAKr(RDY)	/TxACK Rise to /RDY Rise Delay		40	ns	

### Notes:

- [1] Direct address is any of PS, D//C or AD15-AD8 used as an address bus.
- [2] The parameter applies only when /AS is not present.
- [3] Strobe (/STB) is any of /DS, /RD, /WR, /PITACK, /RxAck or /TxACK.
- [4] Parameter applies only if read empties the receive FIFO.
- [5] Parameter applies only if write fills the transmit FIFO.

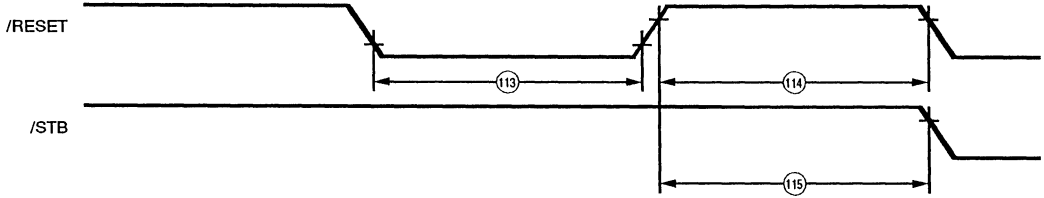


Figure 55. Reset Timing

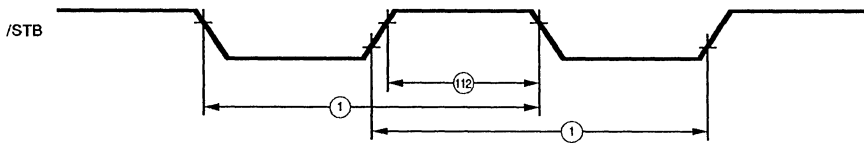


Figure 56. Bus Cycle Timing

**Note:**

/STB is any of the following: /DS, /RD, /WR, /PITACK, /RxACK, or /TxACK.

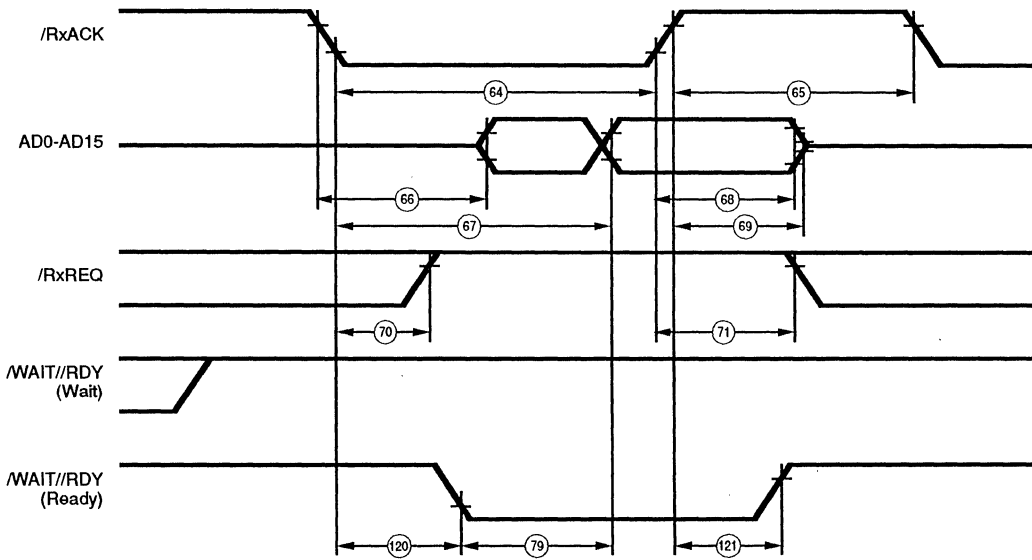


Figure 57. DMA Read Cycle

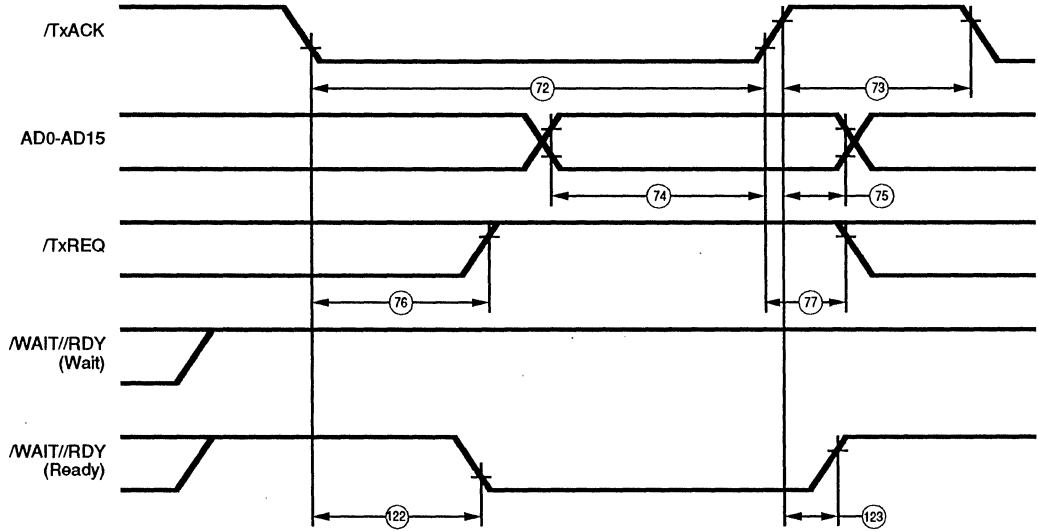


Figure 58. DMA Write Cycle

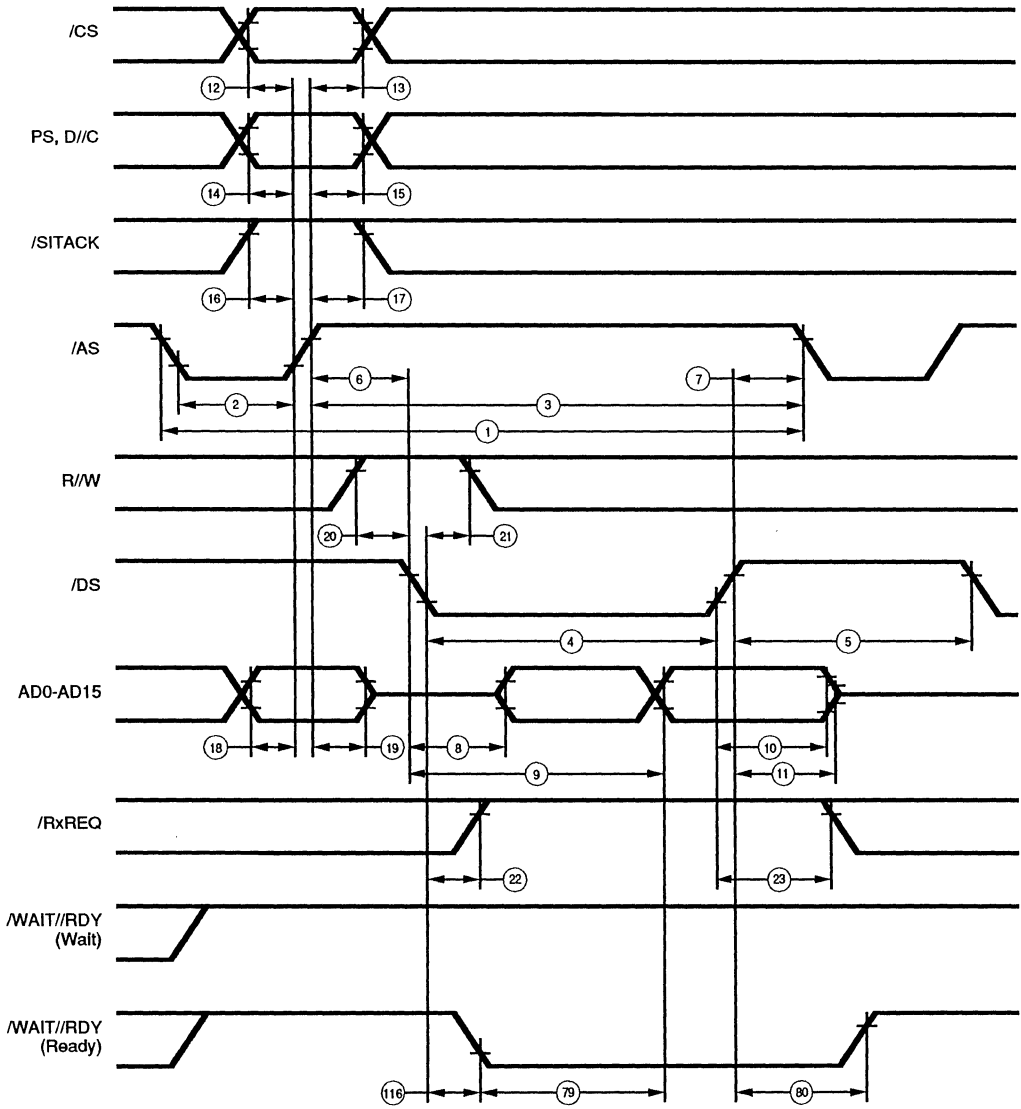


Figure 59. Multiplexed /DS Read Cycle



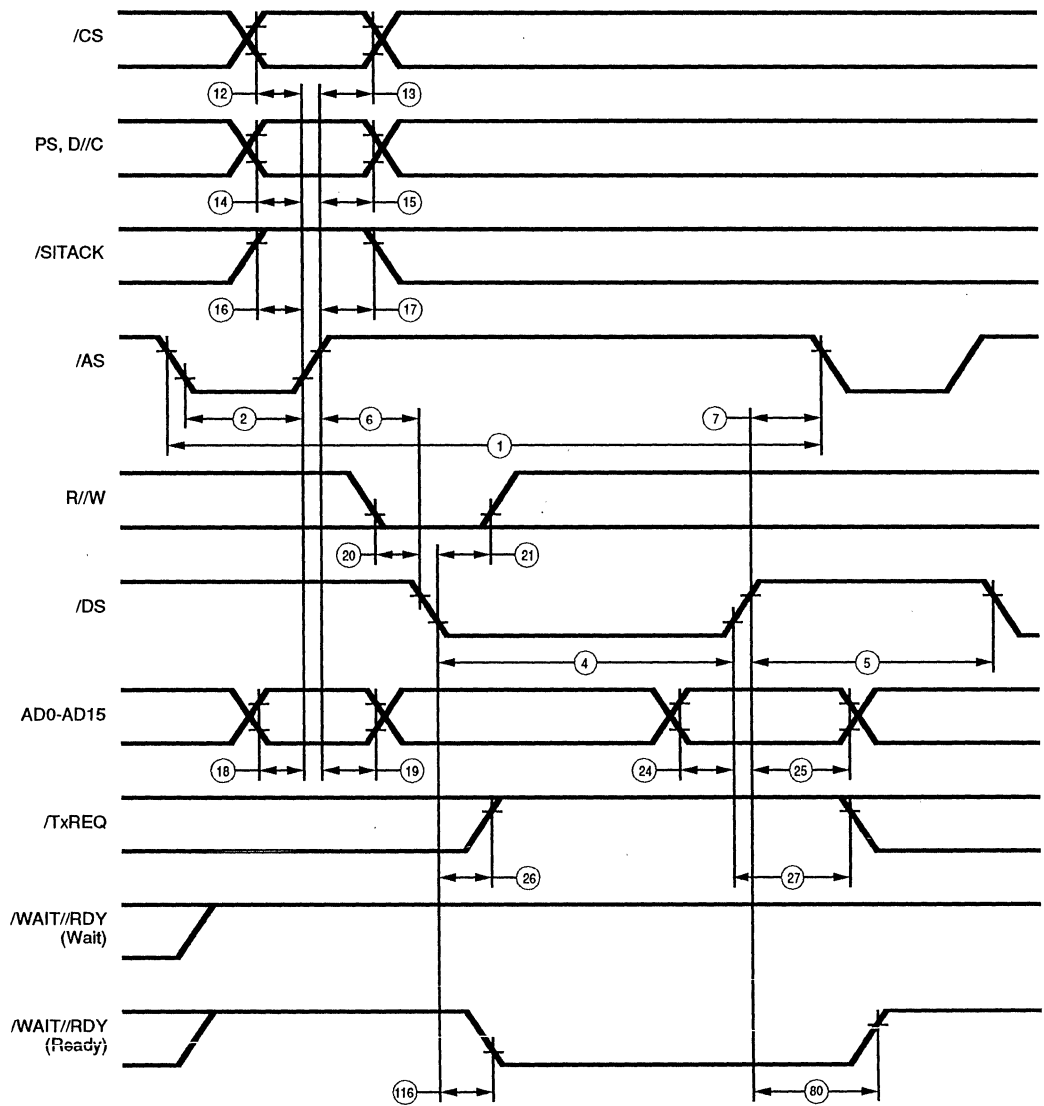


Figure 60. Multiplexed /DS Write Cycle

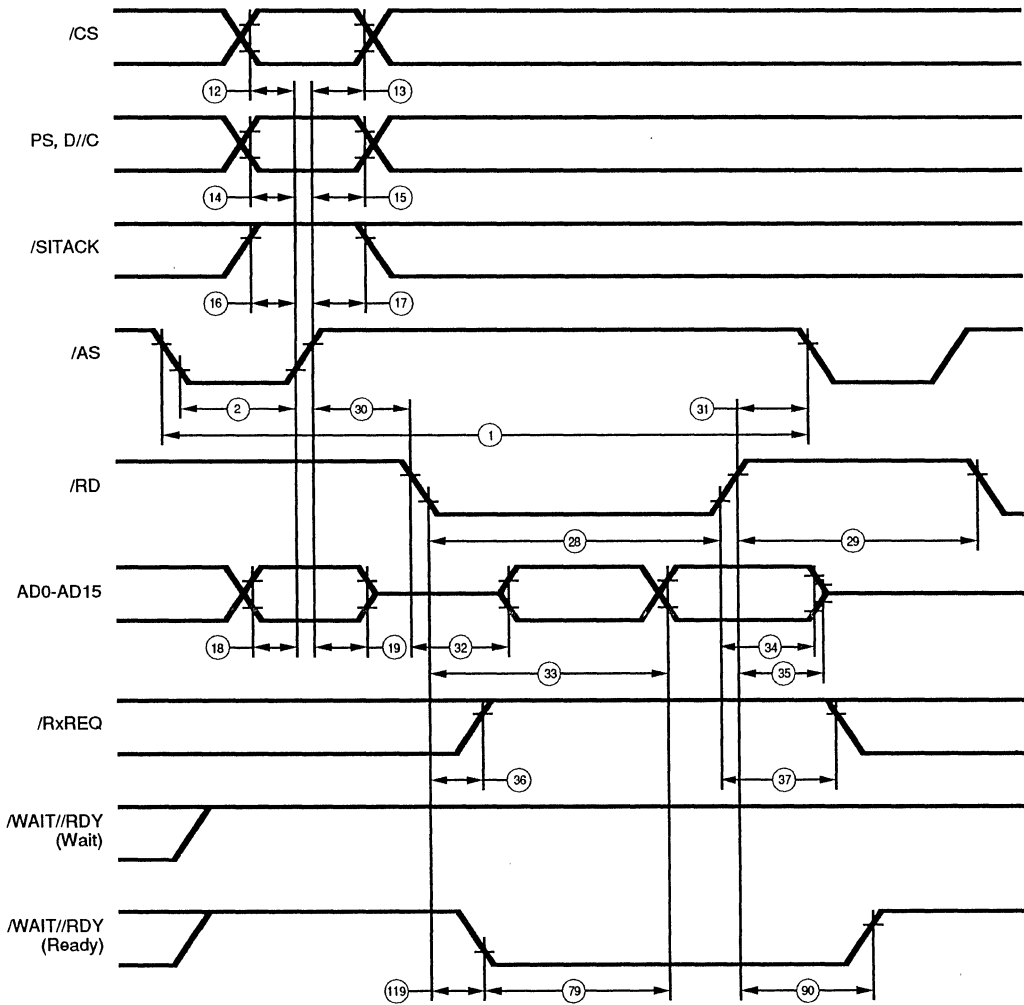


Figure 61. Multiplexed /RD Read Cycle

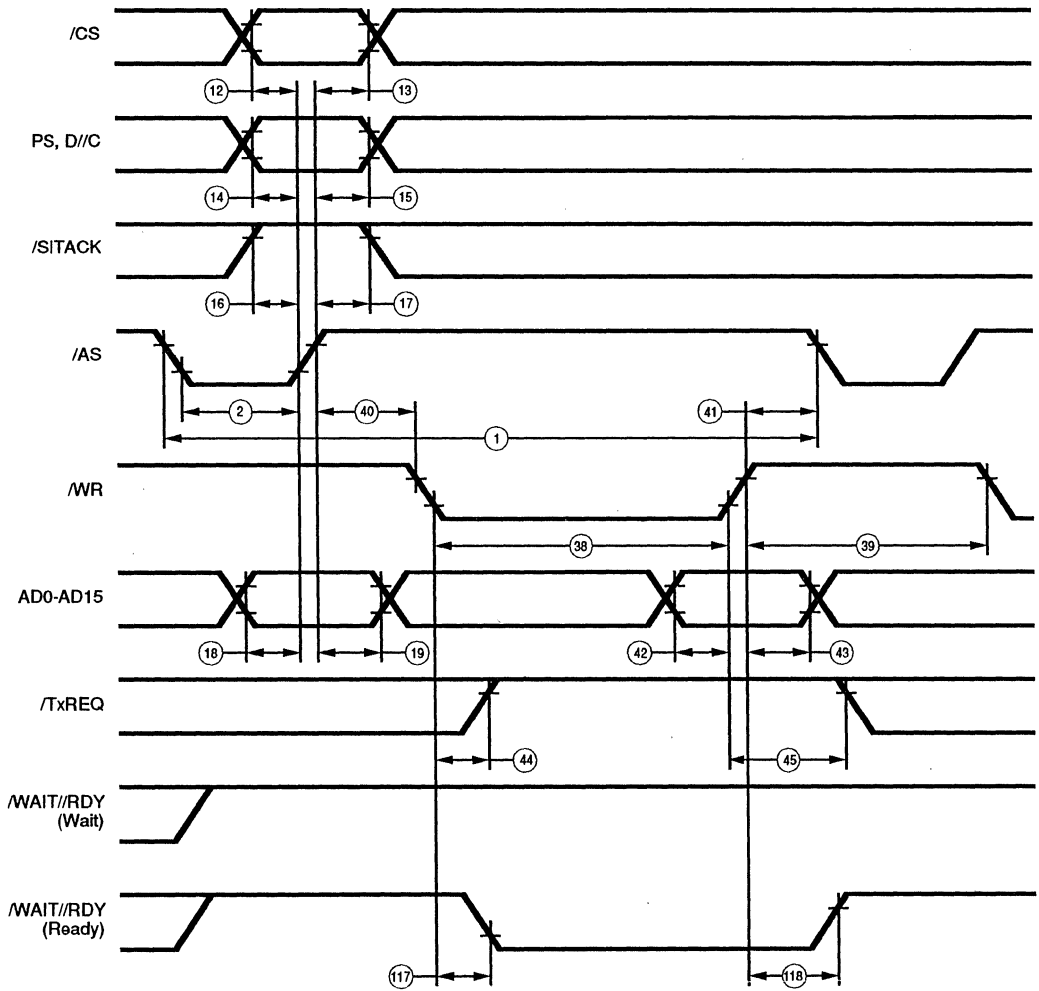


Figure 62. Multiplexed /WR Write Cycle

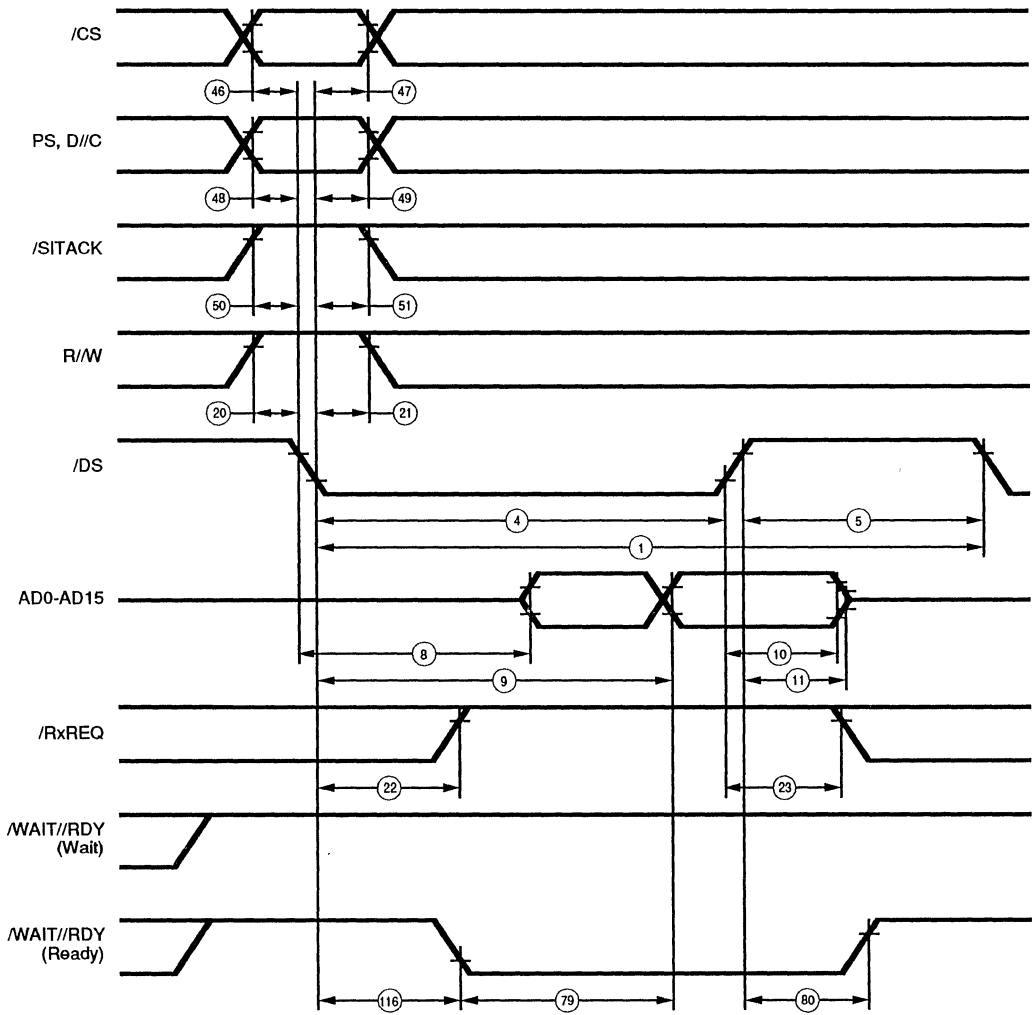


Figure 63. Non-Multiplexed /DS Read Cycle

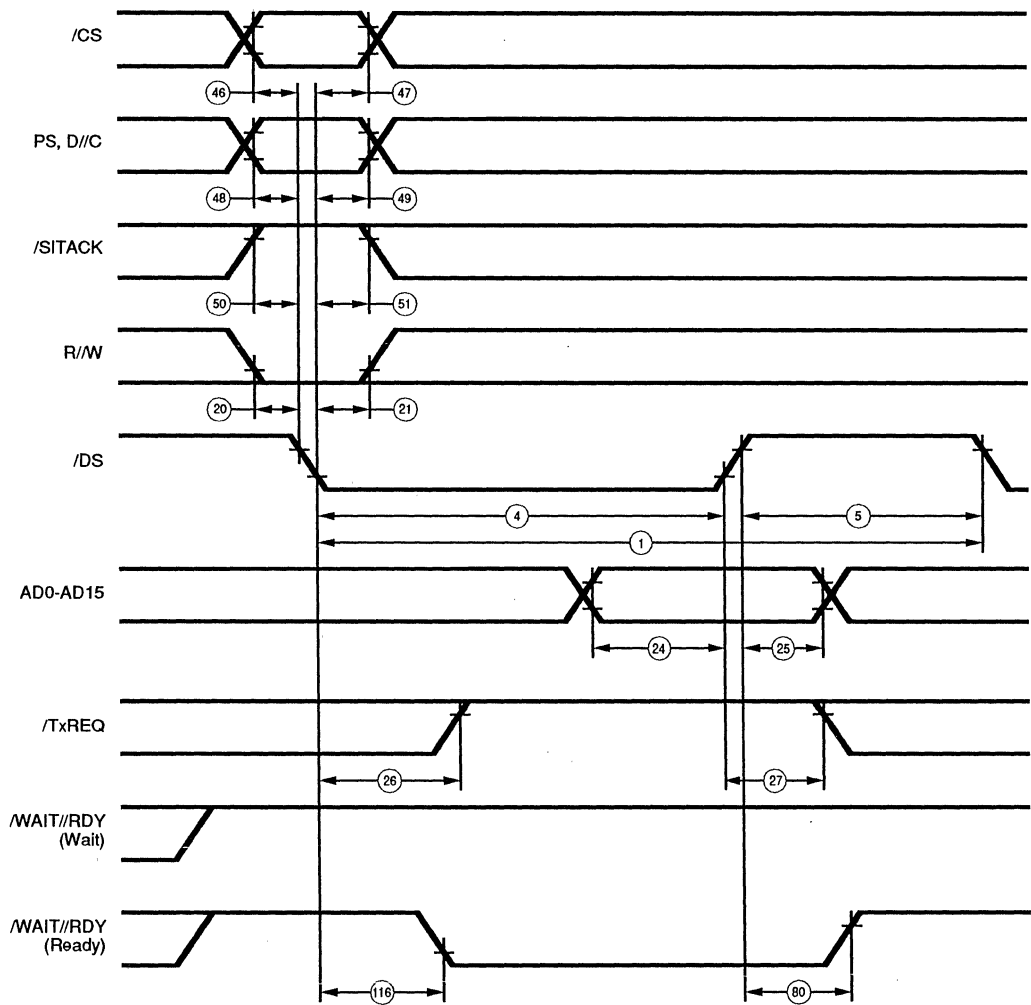


Figure 64. Non-Multiplexed /DS Write Cycle

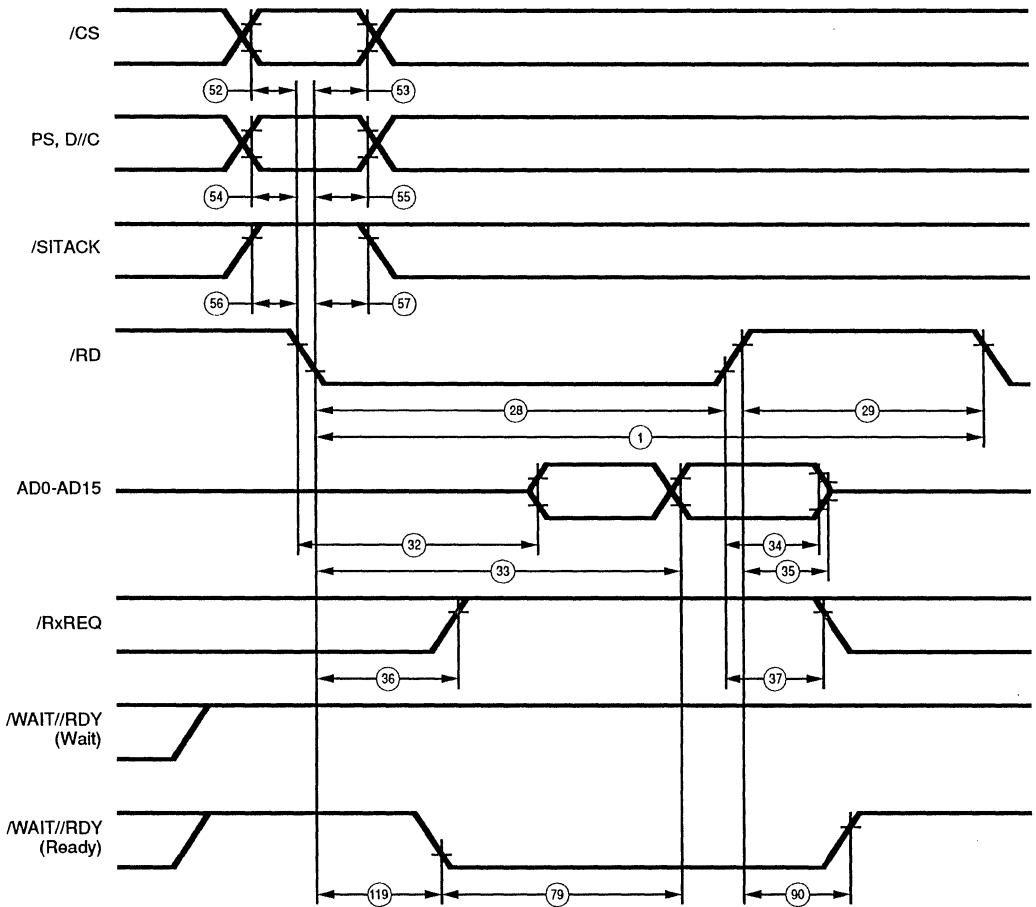


Figure 65. Non-Multiplexed /RD Read Cycle

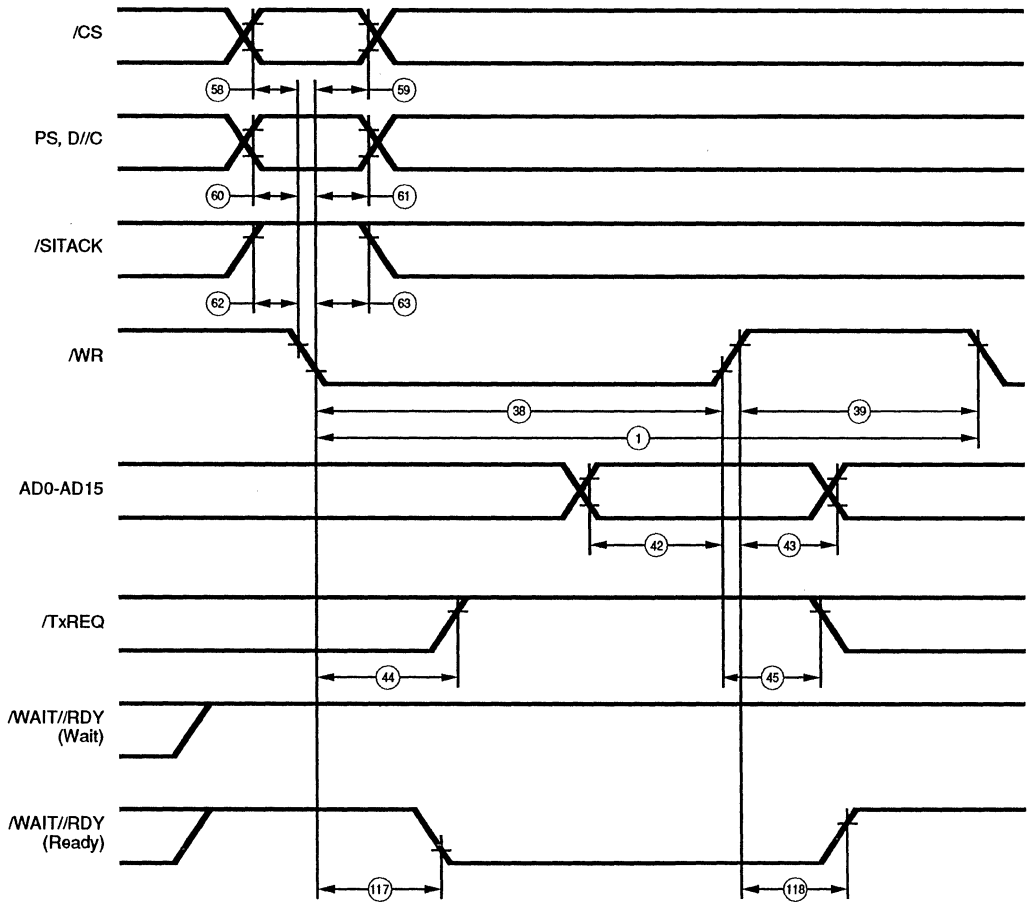


Figure 66. Non-Multiplexed /WR Write Cycle

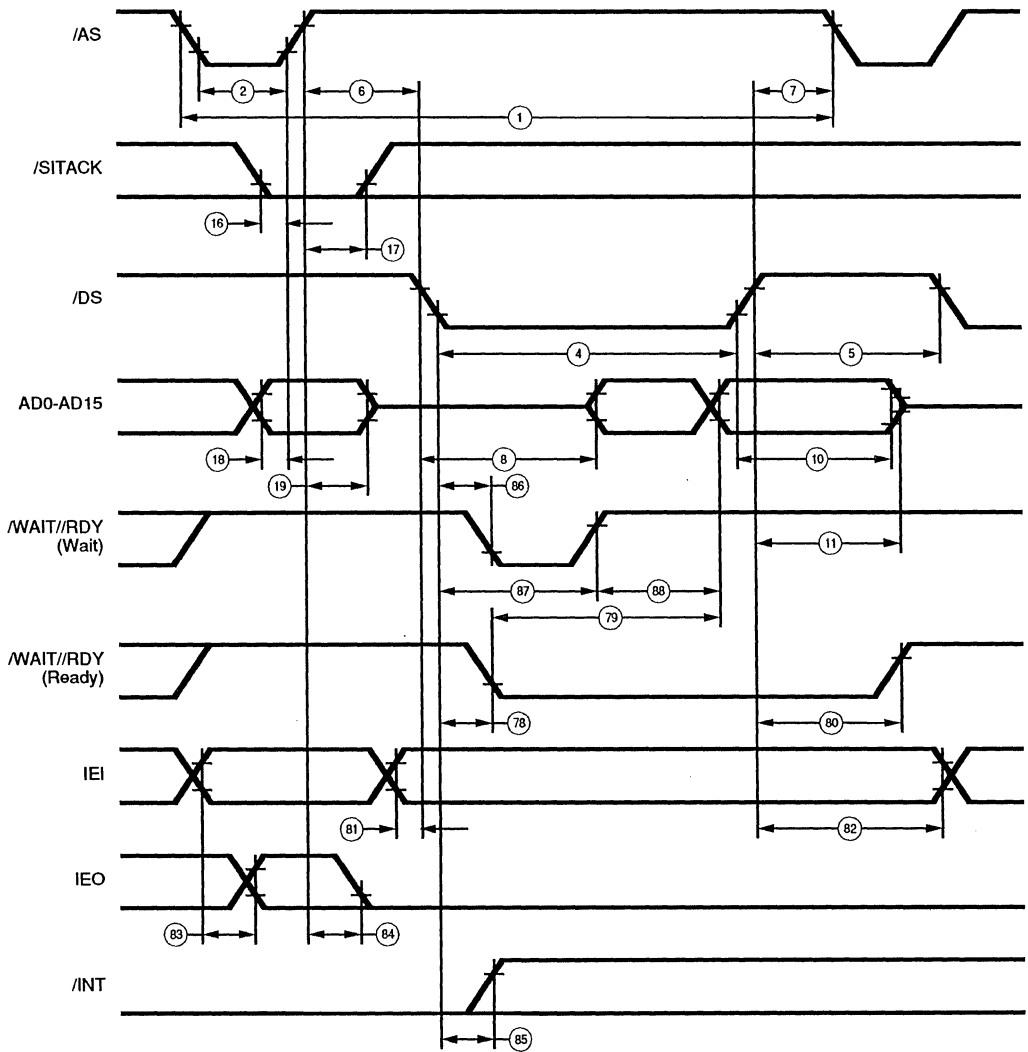


Figure 67. Multiplexed /DS Interrupt Acknowledge Cycle



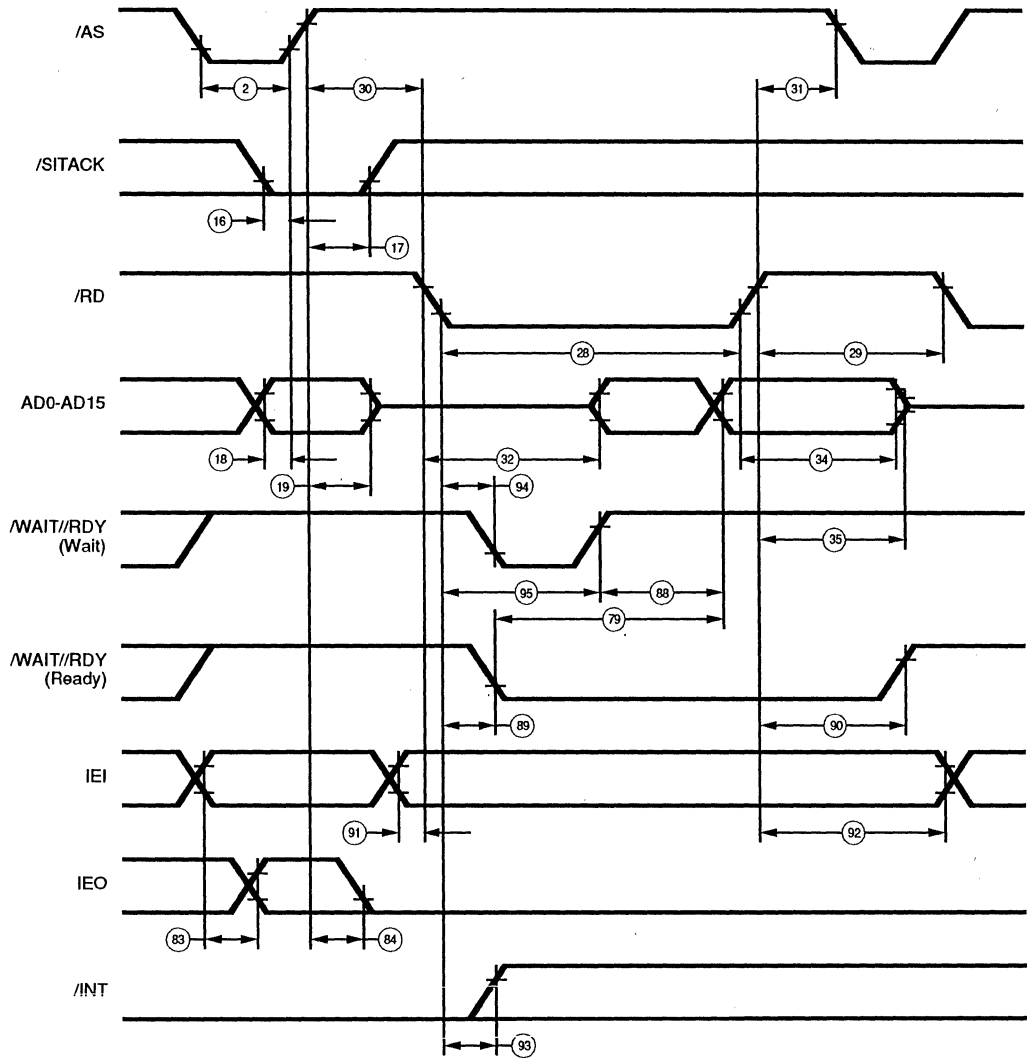


Figure 68. Multiplexed /RD Interrupt Acknowledge Cycle

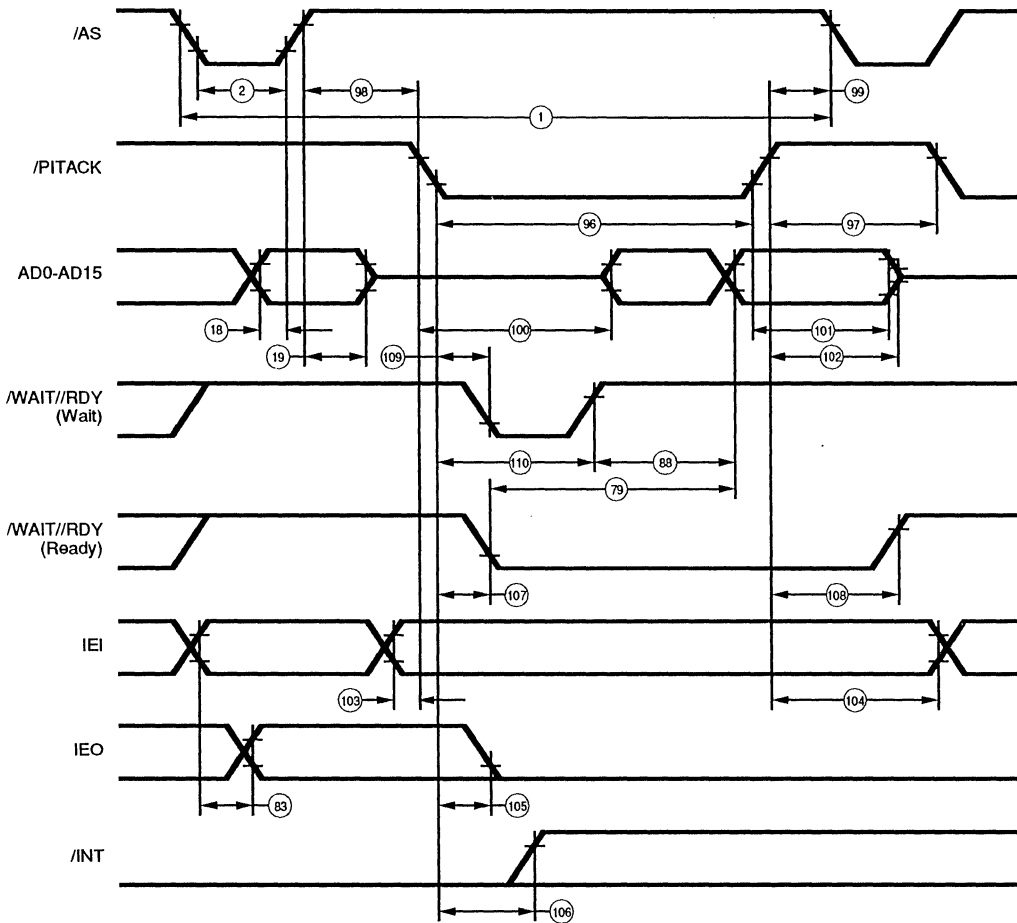


Figure 69. Multiplexed Pulsed Interrupt Acknowledge Cycle

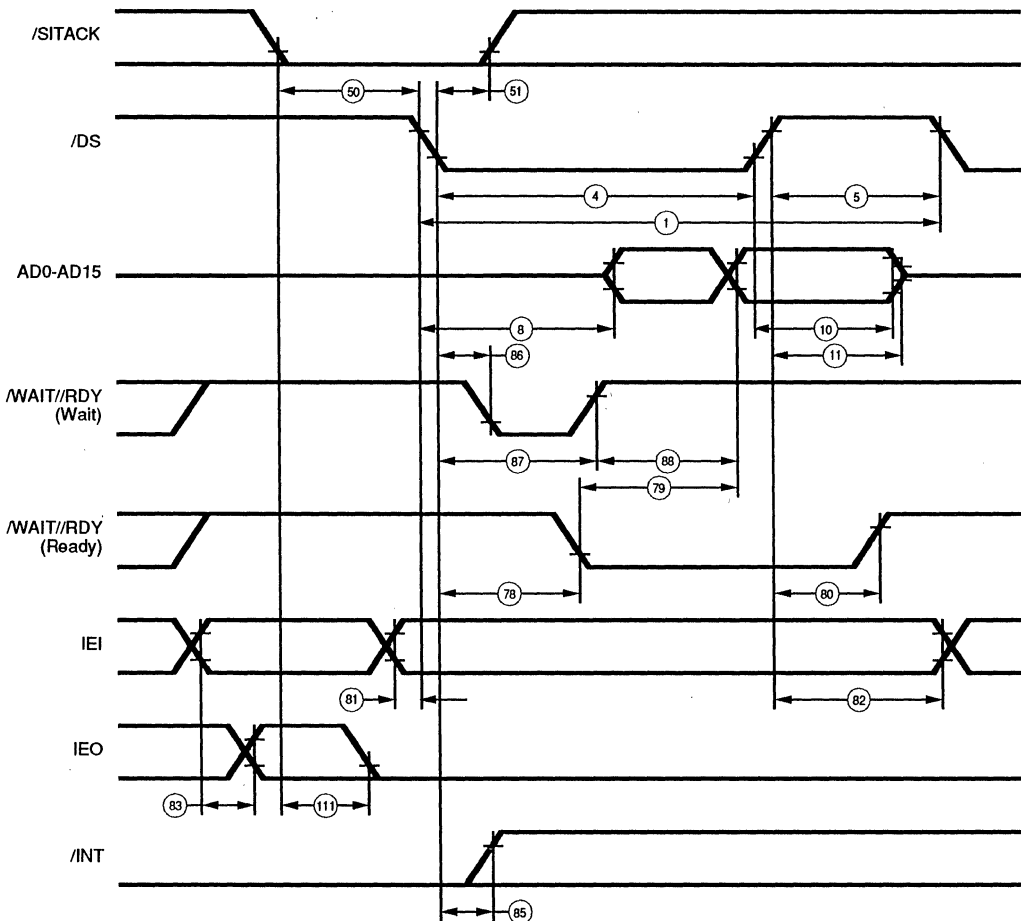


Figure 70. Non-Multiplexed /DS Interrupt Acknowledge Cycle

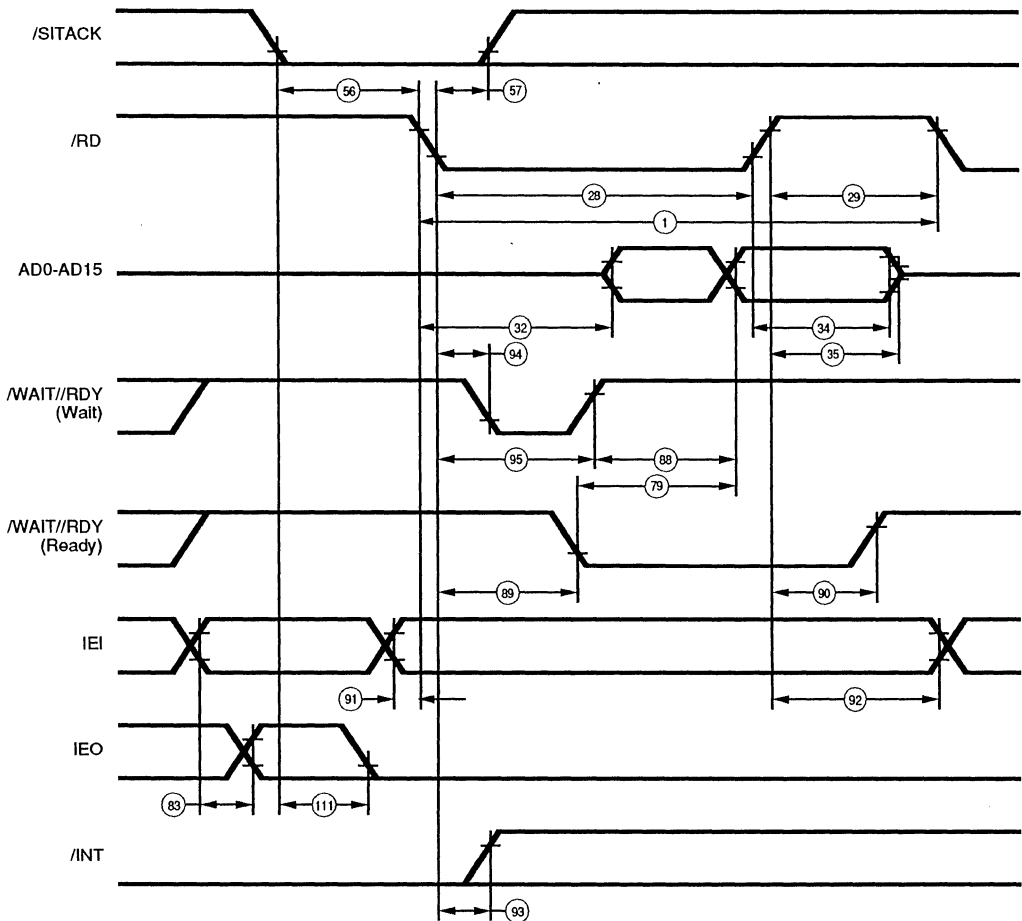


Figure 71. Non-Multiplexed /RD Pulsed Interrupt Acknowledge Cycle

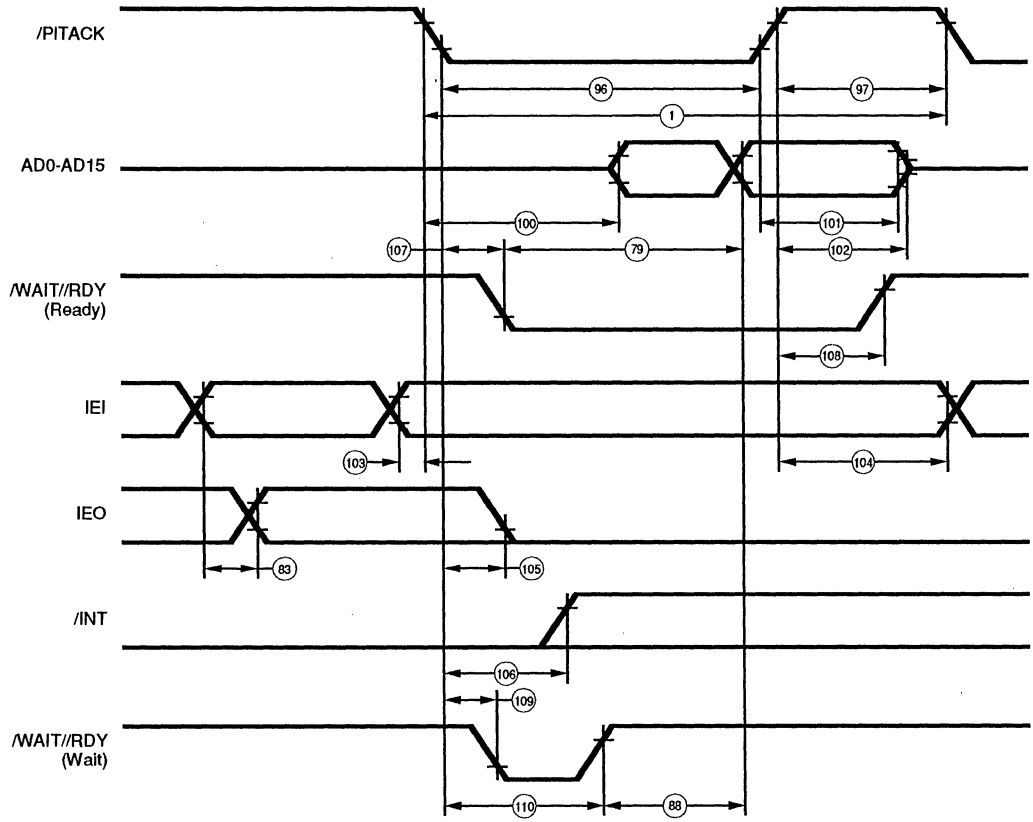


Figure 72. Non-Multiplexed Pulsed Interrupt Acknowledge Cycle

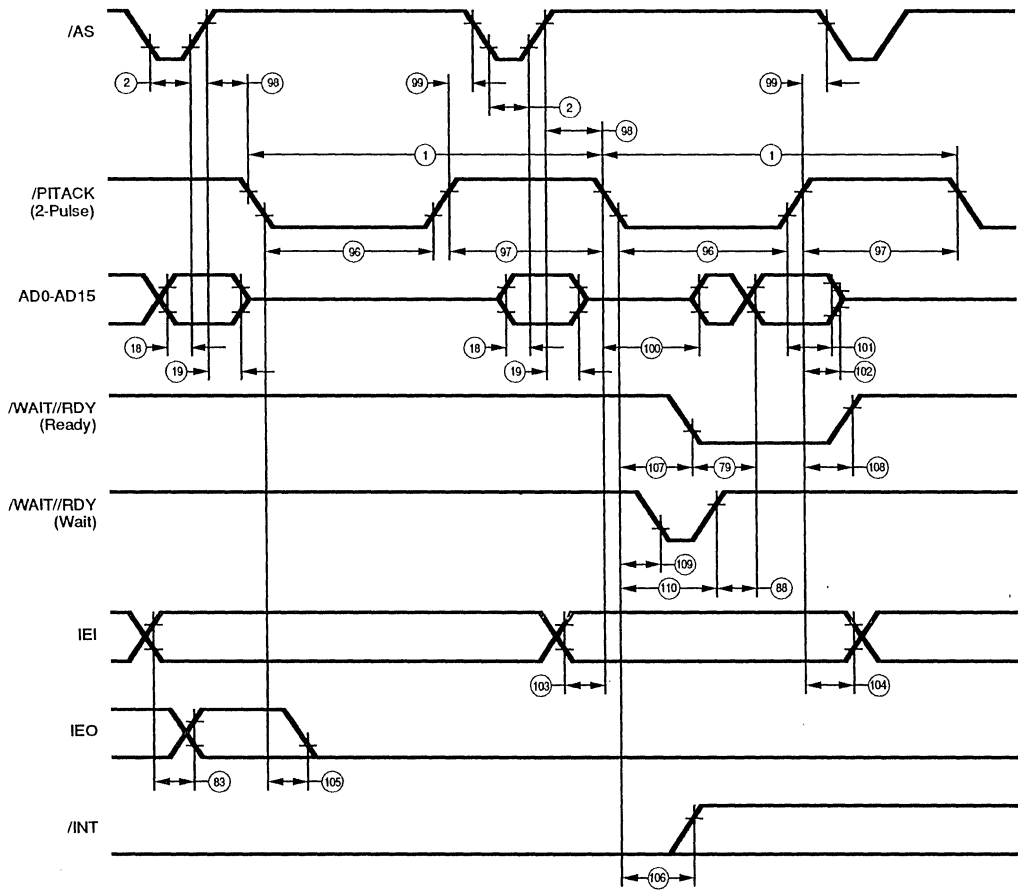


Figure 73. Multiplexed Double-Pulse Intack Cycle

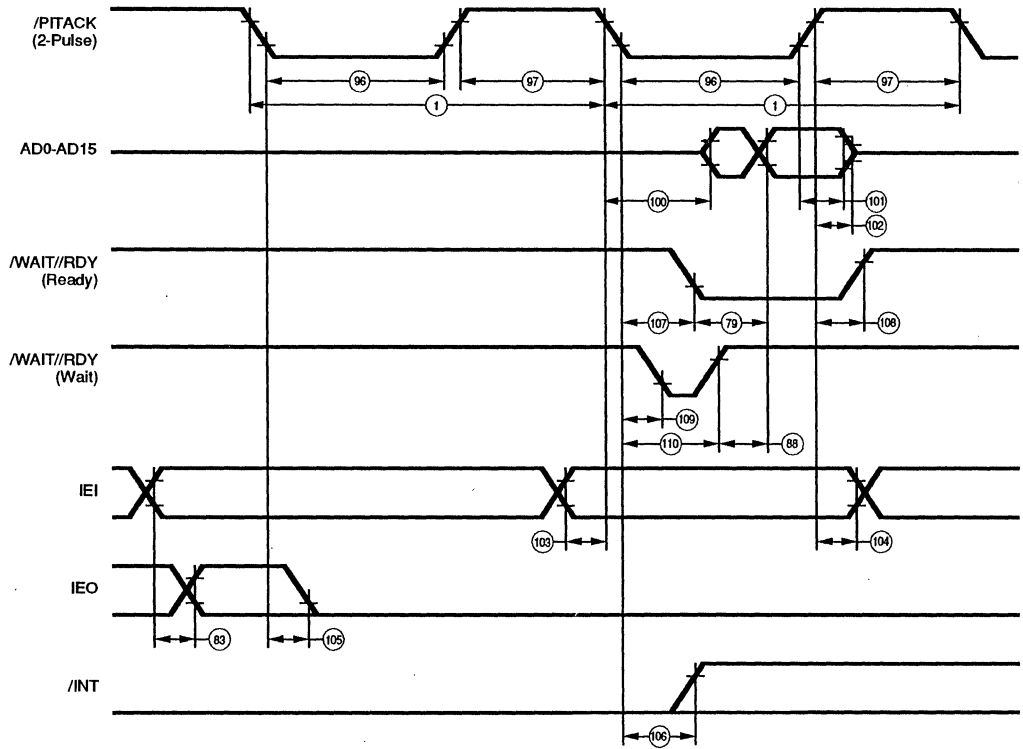


Figure 74. Non-Multiplexed Double-Pulse Intack Cycle

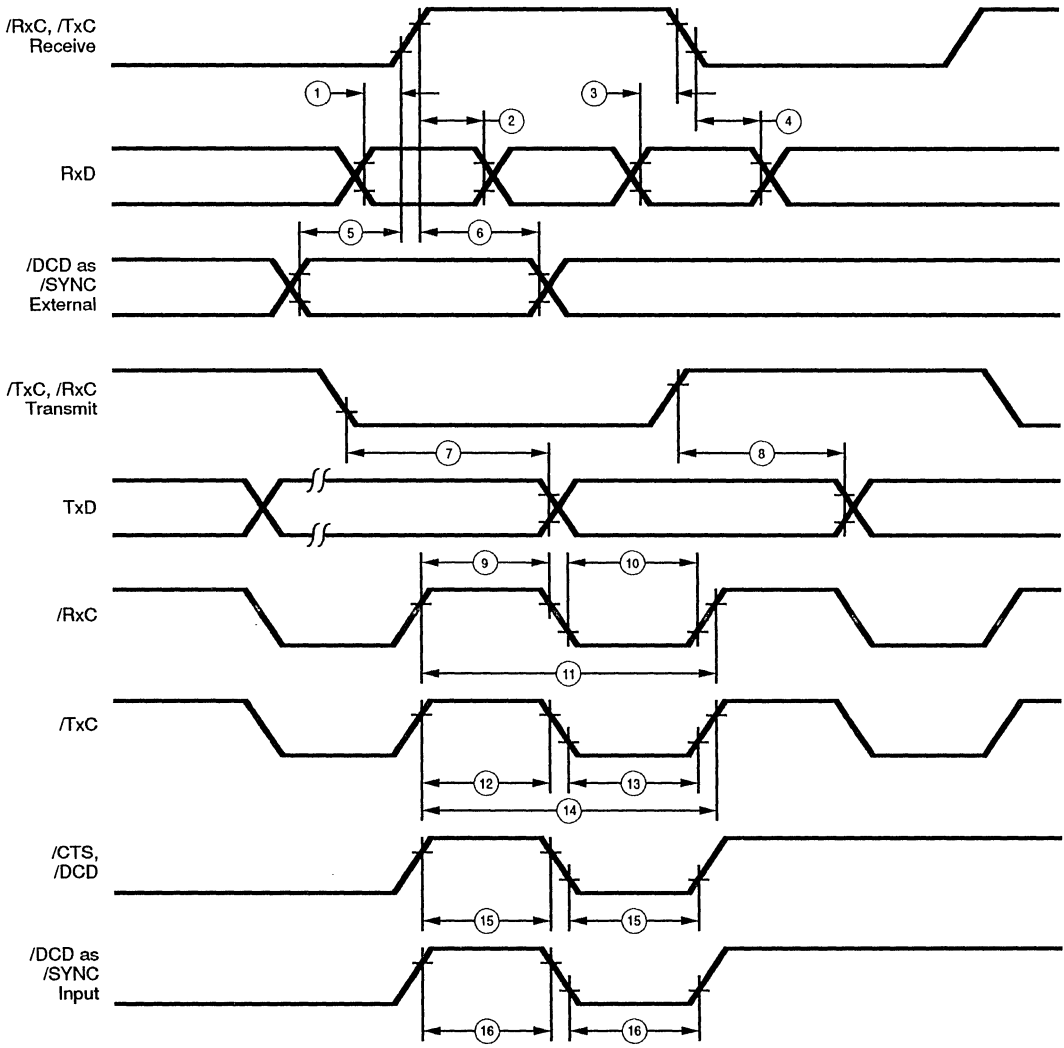


Figure 75. Z16C33 General Timing



## AC CHARACTERISTICS

### Z16C33 General Timing

No	Symbol	Parameter	Min	Max	Units	Note
1	TsRxD(RxCr)	RxD to /RxC Rise Setup Time (x1 Mode)	0		ns	[1]
2	ThRxD(RxCr)	RxD to /RxC Rise Hold Time (x1 Mode)	40		ns	[1]
3	TsRxD(RxCf)	RxD to /RxC Fall Setup Time (x1 Mode)	0		ns	[1,3]
4	ThRxD(RxCf)	RxD to /RxC Fall Hold Time (x1 Mode)	40		ns	[1,3]
5	TsSy(RxC)	/DCD as /SYNC to /RxC Rise Setup Time	0		ns	[1]
6	ThSy(RxC)	/DCD as /SYNC to /RxC Rise Hold Time (x1 Mode)	40		ns	[1]
7	TdTxCl(TxD)	/TxCl Fall to TxCl Delay		50	ns	[2]
8	TdTxCr(TxD)	/TxCl Rise to TxCl Delay		50	ns	[2,3]
9	TwRxCh	/RxC High Width	40		ns	
10	TwRxCl	/RxC Low Width	40		ns	
11	TcRxC	/RxC Cycle Time	100		ns	
12	TwTxCh	/TxCl High Width	40		ns	
13	TwTxCl	/TxCl Low Width	40		ns	
14	TcTxCl	/TxCl Cycle Time	100		ns	
15	TwExT	/DCD or /CTS Pulse Width	70		ns	
16	TWSY	/DCD as /SYNC Input Pulse Width	70		ns	

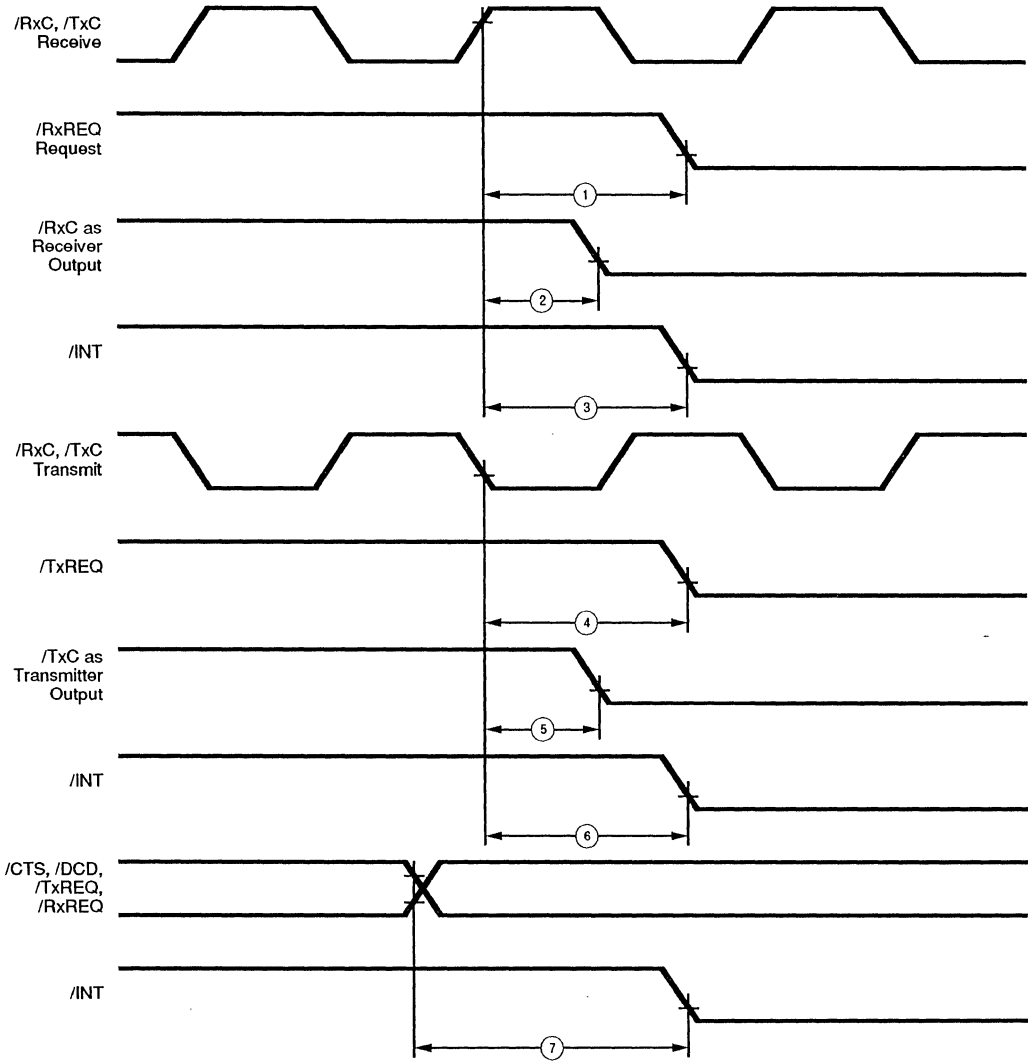


Figure 76. Z16C33 System Timing

---

## AC CHARACTERISTICS

### Z16C33 System Timing

No	Symbol	Parameter	Min	Max	Units	Note
1	TdRxC(REQ)	/RxC Rise to /RxREQ Valid Delay		100	ns	[2]
2	TdRxC(RxC)	/TxC Rise to /RxC as Receiver Output Valid Delay		100	ns	[2]
3	TdRxC(INT)	/RxC Rise to /INT Valid Delay		100	ns	[2]
4	TdTxC(REQ)	/TxC Fall to /TxREQ Valid Delay		100	ns	[2]
5	TdTxC(TxC)	/RxC Fall to /TxC as transmitter Output Valid Delay		100	ns	
6	TdTxC(INT)	/TxC Fall to /INT Valid Delay		100	ns	[2]
7	TdEXT(INT)	/CTS, /DCD, /TxREQ, /RxREQ transition to /INT Valid Delay		100	ns	

---

**Notes:**

[1] /RxC is /RxC or /TxC, whichever is supplying the receive clock.

[2] /TxC is /TxC or /RxC, whichever is supplying the transmit clock.

[3] Parameter applies only to FM encoding/decoding.

# Z16C50

## DDPLL DUAL DIGITAL PHASE LOCKED LOOP MICROCONTROLLER

### FEATURES

- Two independent Digital Phase Locked Loops in one package.
- 10 MHz and 20 MHz Clock operation
- Selectable clock rate, clock sampling edge, and data decoding.
- Synchronous status output
- Accept Code Violation input
- Implemented in 1.6μ CMOS technology
- 28-pin DIP package

### GENERAL DESCRIPTION

The 16C50 DDPLL is a fully static CMOS device that packs two independent Digital Phase Locked Loops, with separate controls for selecting the decoding mode, clock rate, and synchronization edge, in one integrated package (Figure 1). The only common input between the two phase locked loops is /RESET ( / denotes active low signal).

The DDPLL is used in many communication applications requiring detection and extraction of clock from data. It can be used together with Serial Communication Controllers to allow operation at higher data rates. The data rate is programmable at 1/8, 1/16, or 1/32 clock rate. The DDPLL is offered in two speed grades: 10MHz and 20MHz maximum clock speed, which translates to a maximum data rate of 1.25 Mbps and 2.5 Mbps, respectively.

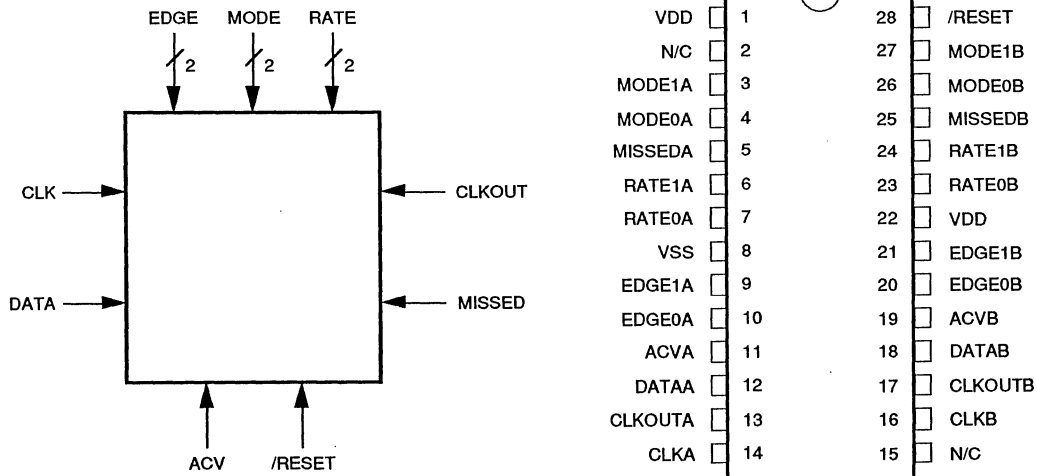


Figure 1. DDPLL Block and Pin Diagrams

---

## PIN DESCRIPTION

The following is a list of DDPLL pins and their descriptions. "A" and "B" at the end of pin names designate the signal connecting to the A- or B-channel of the DDPLL.

**ACVA, ACVB.** *Accept Code Violation* (input, active HIGH). The ACV signal is used to control the response of the DDPLL to code violations present in the received data stream.

**CLKA, CLKB** *Clock*. *Input* (input, active HIGH). The Clock runs at 8-, 16-, or 32-times the received data rate and is used by the DDPLL to generate the CLKOUT signal.

**CLKOUTA, CLKOUTB.** *Clock Output* (output, active HIGH). CLKOUT is the recovered clock for the receive data stream. The receiver should use this clock to sample and decode the received data.

**DATAA, DATAB.** *Receive Data* (input, active HIGH). The DATA signal is the received data stream that the DDPLL is attempting to synchronize with. The DDPLL will provide the CLKOUT signal for use by a receiver attempting to decode this data stream.

**EDGE0A, EDGE0B, EDGE1A, EDGE1B.** *Adjust/Synchronize Edge Controls* (input, active HIGH). These signals

select which edge is used by the DDPLL to achieve and maintain synchronization.

**MISSEDA, MISSEDB.** *Clock Missed* (output, active HIGH). MISSED signal is activated when the DDPLL detects missing edge(s) in the data stream.

**MODE0A, MODE0B, MODE1A, MODE1B.** *DDPLL Mode Controls* (input, active HIGH). MODE0 and MODE1 are used to control the mode of operation of the DDPLL with respect to the encoded format of the incoming data.

**RATE0A, RATE0B, RATE1A, RATE1B.** *Clock Rate Selects* (input, active HIGH). RATE inputs are used to select the data rate divisor to generate the DDPLL clock.

**/RESET.** *Reset* (input, active LOW). This input resets the DDPLL to a known state and must be active for at least two cycles of the slowest CLK signal. This is the only common input to the two Digital Phase Locked Loops.

**VDD.** +5V supply.

**VSS.** 0V (GND) supply.

---

## FUNCTIONAL DESCRIPTION

Prior to device operation, the control inputs of the DDPLL must be set to known states corresponding to the desired mode of operation.

Data decoding format is programmed via MODE0 and MODE1 inputs. Table 1 demonstrates the truth table for these inputs. NRZ, NRZI, FM1 (biphase mark), FM0 (biphase space) and Manchester (biphase level) formats are supported. MODE1-MODE0 of LOW-LOW disables the DDPLL, setting the CLKOUT output LOW. In NRZ format, a "1" is represented by a HIGH level and a "0" is represented by a LOW level. In NRZI format, a "1" is represented by no change in level and a "0" is represented by a change in level. A MODE1-MODE0 of LOW-HIGH selects the NRZ or NRZI decoding modes.

In both of these modes, transitions on the input may only occur on bit cell boundaries and the DDPLL provides CLKOUT to match these bit cell boundaries. In FM1 (biphase mark) and FM0 (biphase space) formats, a transition occurs at the beginning of every bit cell. In addition to this, in FM1, a "1" is represented by an additional transition at the center of the bit cell and a "0" is represented by the absence of such transition. In contrast, in FM0, a "0" is represented by an additional transition at the center of the bit cell and a "1" is represented by the absence of such transition. MODE1-MODE0 of HIGH-LOW selects the biphase-mark (FM1) or biphase-space (FM0) modes.

**Table 1. Mode Selection Truth Table**

MODE1	MODE0	Selected Mode
0	0	Disable/Sync
0	1	NRZ/NRZI
1	0	Biphase-Mark/Space
1	1	Biphase-Level

In Manchester (biphase level) mode, a transition occurs at the center of every bit cell. If the bit is "1", the transition is HIGH to LOW, and if the bit is "0", the transition is LOW to HIGH. Additionally, a LOW to HIGH transition occurs at the boundary of a "1" bit. A HIGH-HIGH selects the Manch-

ester (biphase level) mode. Figure 2 demonstrates an example of a serial data stream with its corresponding encoded waveforms in NRZ, NRZI, FM1, FM0, and Manchester modes.

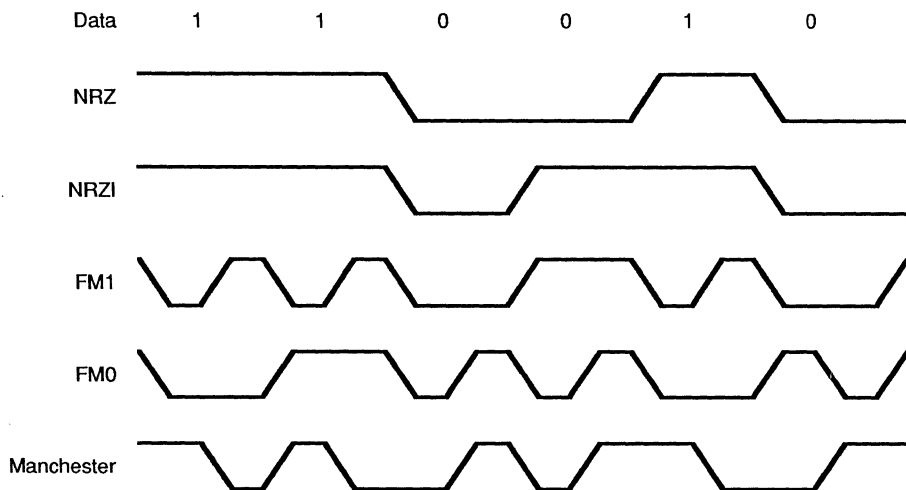


Figure 2. Data Decoding Formats

Clock rate is programmed through RATE1-RATE0 inputs. Clock rate can be set for 8-, 16-, or 32-times the data rate. With maximum clock operation of 20MHz in 8X mode, data rates of 2.5Mbps is achieved. Table 2 illustrates clock rate divisor's truth table. Note that RATE1-RATE0 of HIGH-HIGH is illegal. All DDPLL inputs (with the exception of /RESET) are sampled by the rising edge of CLK and all outputs change state in response to the rising edge of the CLK signal. The two DPLLs are completely independent except for the /RESET input.

Table 2. Data Rate Divisor Truth Table

RATE1	RATE0	Data Rate Divisor
0	0	32X Clock Mode
0	1	16X Clock Mode
1	0	8X Clock Mode
1	1	Not Allowed

EDGE1 and EDGE0 select the edge(s) in the receive data stream used by the DDPLL to achieve and maintain synchronization. Table 3 shows how the rising edge, the falling edge, or both edges of the receive data stream can be

used for synchronization. A HIGH on both EDGE inputs inhibits the DDPLL from using either edge for synchronization. As far as the DDPLL is concerned, edges that are not used to achieve or maintain synchronization are not present. They are reported as missing edges when they occur where an edge is expected.

Table 3. Clock Edge Selection Truth Table

EDGE1	EDGE0	Selected Edge
0	0	Both Edges
0	1	Rising Edge
1	0	Falling Edge
1	1	Adjust/Sync Inhibit

The response of the DDPLL to code violations present in the received data stream can be controlled using the ACV input. This signal is ignored in the NRZ/NRZI mode, where code violations are not possible. In all other modes, however, a HIGH on the ACV allows the DDPLL to recognize an isolated code violation without losing synchronization. Code violations are then used by the receiver for synchronization.

When the DDPLL detects missing edge(s) in the data stream, it activates the MISSED output. If the DDPLL is configured to accept code violations, two consecutive code violations will activate the MISSED output. If the DDPLL is configured not to accept code violations, this output is activated on any missing clock. MISSED will never be activated in the NRZ/NRZI modes of operation, as code violations are not possible in these modes. The DDPLL re-enters the sync-up phase when the MISSED output is activated.

The CLKOUT is the recovered Clock for the receive data stream. The receiver uses this clock to sample and decode the received data.

The only common input between the two phase locked loops in the DDPLL is the /RESET input. This signal must remain active for at least two cycles of the slowest CLK signal and resets the device to a known state: MISSED=LOW and CLKOUT=LOW. Synchronization attempt begins once the /RESET signal is deactivated.

## ABSOLUTE MAXIMUM RATINGS

Voltages on all pins with respect to GND ..... -0.3V to +7.0V  
 Operating Ambient Temperature ..... See Ordering Information  
 Storage Temperature ..... -65°C to +150°C

Stresses greater than those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; operation of the device at any condition above those indicated in the operational sections of these specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## STANDARD TEST CONDITIONS

The DC Characteristics and Capacitance section below apply for the following standard test conditions, unless otherwise noted (Figure 3). All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

+4.5 V <  $V_{CC}$  < +5.5 V  
 GND = 0 V  
 $T_A$  as specified in Ordering Information

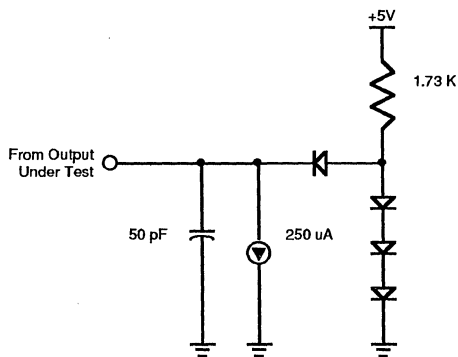


Figure 3. Standard Test Load

## DC CHARACTERISTICS

Symbol	Parameter	Min	Max	Units	Conditions
$V_{IH}$	Input High Voltage	2.0	$V_{DD}+0.3$	V	
$V_{IL}$	Input Low Voltage	-0.3	0.8	V	
$V_{OL}$	Output High Voltage	2.4		V	10H = -1.6mA
$I_{IL}$	Input Leakage Current		±10	µA	0.4V ≤ $V_{IN}$ ≤ 2.4V
$I_{CC}$	Supply Current		40	mA	$V_{DD}=5V, V_{IH}=4.8V, V_{IL}=0.2V$
$C_{IN}$	Input Capacitance		10	pf	Unmeasured pins returned to GND
$C_{OUT}$	Output Capacitance		15	pf	Unmeasured pins returned to GND

Notes:

- $V_{DD}=5V \pm 10\%$  unless otherwise specified, over specified temperature range.
- Capacitance values specified at f=1MHz.

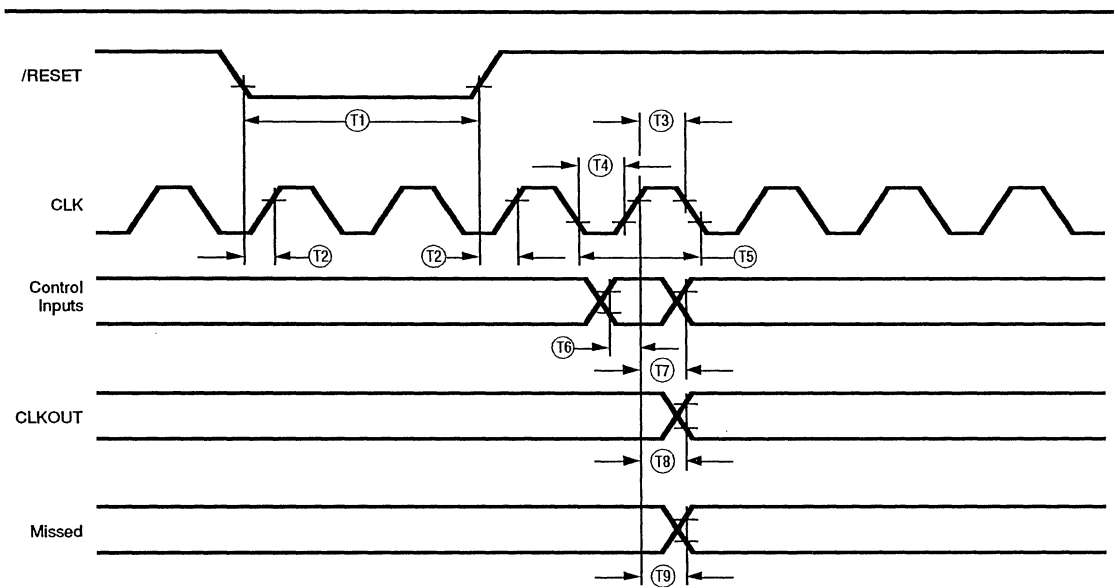


Figure 4. DDPLL Timing Diagram

Figure 2 illustrates the DDPLL timings. "Control Inputs" in Figure 4 refer to MODE1-MODE0, RATE1-RATE0, EDGE1-EDGE0, and ACV inputs.

## AC CHARACTERISTICS

Timing	Symbol	Parameter	Z16C5010		Z16C5020		Units	Notes
			Min	Max	Min	Max		
T1	TwRESI	/RESET LOW Width	2TcC		2TcC			
T2	TsRES(CLK)	/RESET to CLK Setup Time	15		15		ns	
T3	TwCLKh	CLK HIGH Time	40		20		ns	
T4	TwCLKl	CLK Low Time	40		20		ns	
T5	TcC	CLK Cycle Time	100		50		ns	
T6	TsIN(CLK)	Input Valid to CLK Setup Time	15		15		ns	1,2,3
T7	ThIN(CLK)	Input Valid to CLK Hold Time	10		10		ns	1,2,3
T8	TdCLK(OUT)	CLK to CLKOUT Delay Time		30		30	ns	
T9	TdCLK(MIS)	CLK to MISSED Delay Time		30		30	ns	





# TECHNICAL ARTICLES

---



## Z80 Family Questions & Answers

This application note contains the most commonly asked questions about the Zilog Z80 Family. They are divided into following sections:

- Z80 CPU
- Z80 DMA
- Z80 PIO
- Z80 CTC
- Z80 SIO, Z80 DART

Obviously, not every questions on Z80 Family components are answered. However, this application note should give you a good feel for the Z80 Family devices. Along with the technical Manual, Product Specification and some other application notes, it should help make your Z80 design family a little easier. Also, this Application Note is applicable to the Z80 KIO and other Z80 family based Super Integration Devices.

### Z80 CPU

Q: Are the Z80 CPU 6 and 8MHz clocks sensitive like their predecessors?

A: Yes, specifications for rise and fall times and clock voltage levels must be met.

Q: Can the rising edge on the CLK input affect the operation of the CPU?

A: Very much so. For NMOS devices, a negative voltage spike on any pin without back bias will forward-bias the diode that exists between the N+ material connected to the pad and p-type substrate. This action causes the injection of many electrons into the substrate. Once in the substrate, they are free to drift into any region of higher potential, which is the N+ region at Vcc of storage nodes storing a "1". Since storage holds don't store much charge (in order to minimize capacitance), these electrons in the substrate can be swept across the junction and destroy the "1" stored there. This reaction obviously affects the operation of the part.

Also, on CMOS devices, positive spikes on any pin exceeding Vcc voltage could cause "Latch-up"!

Q: What is the clock input impedance (load)?

A: Capacitive load only (35pF max).

Q: Will Non-maskable interrupts continue occurring and executing if the NMI line pulses prior to the finish of the service routine?

A: Yes. Non-Maskable interrupts can not be disabled by user. Even though Non-Maskable Interrupts are negative edge triggered, if the input to the CPU pulses before termination of the service routine, then the service routine will begin again.

Q: How does the Non-Maskable Interrupt acknowledge cycle and RETN instruction actually work?

A: When a Non-Maskable Interrupt is acknowledged, interrupt flip-flop #1 (IFF1) is actually cleared to inhibit the acknowledgement of maskable interrupts. The state of interrupt flip-flop #2 (IFF2) is not altered. This is the only time that the contents of IFF1 and IFF2 can disagree. When the RETN instruction executes, the state of IFF2 is copied back into IFF1. This allows the state of maskable interrupts, before a Non-Maskable Interrupt, to be restored after service routine execution.

Q: How are subtraction operations performed?

A: Although the actual operation is probably a 2's complement addition, the flags are affected as if it were a logical subtraction operation.

Q: What is the setup time to recognize an NMI?

A: Through characterization of the CPU, Zilog has found that a setup time of 120nS (@ 4MHz) is required in order to assure that NMI is recognized before INT is recognized.

Q: What do the EI and DI instruction actually do?

A: Only the interrupt control flip-flops (IFF1 and IFF2) are affected by those instructions. The DI instruction will clear both IFF1 and IFF2 and prevent any further maskable interrupt from being recognized from that point on. The EI instruction will set both IFF1 and IFF2, but maskable interrupts will not be recognized till the completion of the next instruction.

---

Q: What is the status of the output drivers when the CPU is in a power-down situation?

A: When the CPU is without power, the output drivers appear to be in a high impedance state.

Q: How can I use the on chip refresh mechanism of the Z80 CPU to handle refreshing of 64K D-RAMs?

A: Here are some suggestions (assuming 256 cycle refresh):

1. Use an external counter to count 128 M1 cycles and toggle refresh address line A7.
2. Use external hardware to generate an NMI every 2mS and change the state of bit D7 in the R Register via software.
3. Use refresh address bit A6 to toggle the state of refresh address bit A7.

Q: Is there a method for testing hardware without removing the Z80 CPU from the socket?

A: Two methods are available:

1. Use BUSREQ to tri-state all control signals and then use external hardware to simulate the logic;
2. Remove power and ground from the CPU, all signals should go to a high-impedance.

Q: Does the CPU tristate M1 during reset?

A: No.

Q: Is Zilog going to add a 3.15 Volt current drive spec for designers using 74HCxx series of components?

A: No. There is a choice of either using 74HCTxx logic or using our CMOS Z80 CPU.

Q: If NMI is activated DURING reset, will the processor execute the NMI or address 0000h after reset goes high?

A: Since NMI input is "edge-triggered" input, if the CPU has active NMI "during" reset, CPU won't detect NMI and will execute the instruction at 0000h. If NMI goes low after RESET goes inactive, then CPU will process NMI.

Q: I've heard the CPU is a static device. Can I use the clock to single step it?

A: It's different for NMOS and CMOS.

NMOS: No, it violates the clock specs.

CMOS: Yes. You can do that.

Q: I don't seem to get the correct state of the interrupts when using the LD A,I and LD A,R instructions to read the state of IFF2. Why is this? How can I get around this?

A: On CMOS Z80 CPU, we've fixed this problem. On NMOS Z80 CPU, in certain narrowly defined circum-

stances, the Z80 CPU interrupt enable latch, IFF2, does not necessarily reflect the true interrupt status. The two instructions LD A,R and LD A,I copy the state of interrupt enable latch (IFF2) into the parity flag and modifies the accumulator contents (See table 7.0.1 in the Z80 CPU technical manual for details). Thus, it is possible to determine whether interrupts are enabled or disabled at the time that the instruction is executed. This facility is necessary to save the complete state of the machine. However, if an interrupt is accepted by the CPU during the execution of the instruction -- implying that the interrupts must be enabled -- the P/V flag is cleared. This incorrectly asserts that interrupts were disabled at the time the instruction was executed.

This paradox can be traced to the internal timing of the CPU. The problem is that the interrupt flip-flop (IFF2) is cleared before it is actually transferred to the P/V flag. The state of the interrupt enable latch is not copied into the parity flag until after the interrupt time, occurring during the execution of the instruction, has been accepted. Since the acceptance of the interrupt automatically clears the interrupt enable latch, the parity flag is also cleared, despite the fact that interrupts were enabled when the instruction started executing.

A neat solution to this anomaly relies on the fact that at least one item -- the old PC value -- is saved on the stack when an interrupt is accepted. The "next entry" position on the stack (the word below the address currently held in the stack pointer) may be cleared before execution of LD A,I (or LD A,R). If that zero value has changed by the time that the next instruction in the routine is executed, then an interrupt must have been accepted. This implies that interrupts were enabled, even if the state of the parity flag suggests that they were not. Of course, if the parity flag is found to be set after LD A,R (LD A,I) has been executed, there is no need to check the stack top. Interrupts are definitely enabled if the parity flag is in this state.

Two routines are listed here. Both return carry clear if interrupts are enabled, set otherwise. Both corrupt the A register; it does not contain the value in the I (or R) register on exit. The status of all flags except the carry flag are undefined on exit.

The first routine may be loaded anywhere in memory except "page zero" -- 0000h to 00FFh. This small restriction comes about because the routine checks only the most significant byte of the "next" stack entry. This byte will be non-zero after an interrupt has occurred if and only if the routine itself is not on page zero. The second routine tests both bytes of the "next" entry and, therefore, overcomes this restriction.

Caution, these routines presume that the service routine for any acceptable interrupt will re-enable interrupts before it terminates. This is almost always the case. They may not return the correct result if an interrupt service routine, which does not re-enable interrupts, is entered after the execution of LD A,I (or LD A,R).

Listing 1: This routine may not be loaded in page zero (0000h to 00FFh).

```
GETIFF:
XOR  A      ;C flag, acc. := 0
PUSH AF    ;stack bottom := 00xxh
POP  AF     ;Restore SP
LD   A,I   ;P flag := IFF2
RET  PE    ;Exit if enabled
DEC  SP    ;May be disabled.
DEC  SP    ;Has stack bottom been
POP  AF    ;overwritten ?
AND  A     ;If not 00xxh, INTs were
RET  NZ    ;actually enabled.
SCF      ;Otherwise, they really are
RET      ;disabled.
END
```

Listing 2: This routine may be loaded anywhere in memory.

```
GETIFF:
PUSH HL    ;Save HL contents
XOR  A     ;C flag, acc. := 0
LD   H,A  ;HL := 0000h
LD   L,A
PUSH HL    ;Stack bottom := 0000h
POP  HL    ;Restore SP
LD   A,I  ;P flag := IFF2
JP   PE,  ;Exit if isn't enabled
POP HL    ;May be disabled.
DEC  SP    ;Let's see if stack bottom
DEC  SP    ;is still 0000h.
POP  HL    ;Are any bits set in H
LD   A,H  ;or in L ?
OR   L
POP  HL    ;Restore old contents.
RET  NZ    ;HL < 0 : isn't enabled.
SCF      ;Otherwise, they really are
RET      ;disabled.

POP HL:
POP  HL    ;Exit when P flag is
RET      ;set by LD A,I
END
```

asynchronous with respect to the system clock (i.e. no setup time required). Zilog's official position on this topic is as follows.

All asynchronous inputs to the Z80 family CPUs should be externally synchronized with the CPU clock. The required synchronization is specified by the setup and hold times for asynchronous inputs to the CPU. The synchronization is automatically provided for by the Z80 Family peripherals that are capable of driving the asynchronous inputs to the CPU.

In the Z80 CPU Technical Manual (Pages 70 and 72, footnote B), it is stated that "All control signals are internally synchronized so that they are totally asynchronous with respect to the clock." This statement should be amended to say "When interfacing the Z80 CPU to the Z80 family peripherals, the interface control signals are internally synchronized with the system clock by the peripherals themselves. When interfacing to the Z80 CPU with other devices, these control signals should be synchronized with respect to the system clock." Note that the former statement has been removed from the data book and the CPU product specification, but has not been removed from the technical manual yet.

The basis for the synchronization of the input control signals is the potential for the occurrence of a phenomenon called a "meta-stable state". The details of the meta-stable state are complex, but the concept is fairly simple. A meta-stable state occurs in bi-stable logic devices at the interface between an asynchronous and synchronous environment. All two-state logic devices spend some finite amount of time in the "linear region" (between the logic state of one and zero). The length of time spent in the linear region depends upon the switching speed of the device. If a synchronous system samples asynchronous inputs at the precise point in time that it passes through the linear region, the output of the sampling logic may spend time in an undefined logic state (the meta-stable state). The settling time to a valid logic state is proportional to the inverse exponential of the speed of the switching devices. More importantly, if the device in the meta-stable state is connected to several other bi-state devices in the system, the possibility exists for each of these bi-state devices to interpret the non-binary (or meta-stable) input differently. The final result can be an undefined or unpredictable state for a sequential state machine such as the Z80 CPU.

There are several points that should be remembered concerning these asynchronous inputs:

Q: Are all of the Z80 control lines internally synchronized?  
A: The inputs in question are INT, NMI, BUSREQ, WAIT, and RESET. In the past, it seems that some of our customers have assumed that those inputs are totally



---

Q: When using the variable timing modes, are there any constraints in setting up the two ports that the user should be aware of?

A: Yes. When using the early cycle end timing feature of the DMA, it is strongly recommended that both ports be initialized with the same timing constraints.

Q: Is there any way to reset the DMA besides the RESET command and power-down?

A: With the CMOS DMA: On 44-pin PLCC package, there is a newly added "hardware reset pin" on pin 12 (This pin is left open on NMOS PLCC). Also, we've added special functions to the M1 signal line that allows you to reset C-MOS DMA. During an active M1 signal, without an active RD or IORQ, the DMA is reset. This feature is the same as that with Z80 PIO.

With NMOS, the only way to reset the DMA is by reset command. Actually, the RESET command can only reset the DMA if the CPU has control of the bus. If the DMA has control of the bus there is no way to reset it other than powering down the system (or the DMA).

Q: How long does power need to be removed from the DMA for an internal reset to occur?

A: Zilog tests the power-on reset circuit at 10mS. If the user is going to remove power from the DMA, Zilog recommends that it be done with the CLK input high.

Q: What limitations are not specified in the data book?

A: For NMOS DMA, when using the DMA in BURST mode with 2 cycle timing, an extra transaction is generated at the end of the burst.

For CMOS DMA, we've fixed all limitations.

Q: How can I use the DMA to transfer a page of information but do it one line at a time and wait between lines? (Printer application)

A: Operate the DMA in the Burst mode and use the printer I/O READY line to control DMA. Program the DMA for auto-restart mode to transfer the same "page" area continuously. When the printer is unable to accept a "line", the DMA will allow the CPU to control the bus.

---

## Z80 PIO

Q: When using a port of the PIO in bit mode (mode 3), can any of the bits, programmed as outputs, affect the interrupt conditions set for recognizing inputs?

A: While it is undocumented, it is possible that the state of the bits programmed as outputs could be used as satisfying conditions for the mode 3 interrupt equation. It is recommended that all bits not needed for the interrupts be masked off.

Q: Can the PIO be programmed to provide a 16-bit input port and an 8-bit bidirectional port at the same time?

A: Yes, but there are some major concerns in doing it. Remember that when Port A is programmed into the bidirectional mode (mode 2), the handshake lines from Port B are used as input handshake lines for Port A. Some confusion occurs within the PIO if Port B is also programmed into the input mode (mode 1) and tries to use the handshake lines. A combination of software and hardware can be used to insure that data will not change until both ports can be read.

Q: Is the PIO port protected against hysteresis?

A: No.

Q: Do you have to strobe data into the port for proper mode 1 operation?

A: Yes, if you want to generate interrupts for mode 1 operation. If you only want to read the port data, then the STB input can be held low to make the input data latches transparent.

Q: How can I get Port B interrupt in Mode 3 and Port A interrupt in Mode 2?

A: You can get them, but it can cause severe interrupt conflicts if you choose that option. Port B interrupts are used by Port A in bidirectional mode for receive data interrupts. To prevent interrupt conflicts, Port B interrupts should be enabled, but all bits of Port B should be masked from affecting the interrupts. In the PIO Technical Manual it states that, "the same interrupt vector will be returned for a Mode 3 interrupt on Port B and an input interrupt during Mode 2 operation of Port A" (Section 5.3).

Q: Can the PIO control register be written while the PIO IUS bit is set?

A: Yes. But it is a safer programming practice to program the device after the RETI command.

Q: The on-chip power-on reset does not always work properly. How can I get around this?

A: Use the external hardware reset condition. Activate M1 for a minimum of two clock cycles without activating either RD or IORQ.

Q: When using the PIO in Mode 2, a 55h is written to the port. On the port side, an 0AAh is strobed into the port via BSTB. When the processor reads the data port, the 55h is read back instead of the 0AAh. Why and How?

A: The only way that the system can read the same data that it wrote into the PIO was if the ASTB signal was active (place the 55h onto the port bus) and the BSTB went active to strobe it into the data register. Suggest that system logic inhibit both strobe signals from becoming active during the same time.



Q: On which clock edge is the PIO reset (with M1 active and IORQ and RD inactive)?

A: The actual reset function will take place when the M1 signal goes low active (must have been active a minimum of 2 clock cycles).

Q: Can the PIO catch pending interrupts while interrupts are disabled?

A: Yes. Enabling the interrupts allow the interrupt daisy chain to function and the interrupt under service flip-flops to be set.

Q: A question came in concerning how the Z80 PIO handled its interrupts. Is the PIO capable of storing pending interrupts or must an interrupt be serviced and cleared (via either RESET or RETI) before another interrupt can be accepted?

A: It seems that the Z80 PIO interrupt structure is designed so that pending interrupts can be stored. There are caveats to watch for in this however. The only way to store a pending interrupt is while another one is under service, and only one pending interrupt can be stored. Be aware that if you are operating in Modes 0, 1 or 2, the transition of the STB signal can cause new data to be latched into the input data register and generate a pending interrupt. Be sure that any previous data can be read from the PIO before any new data is strobed in.

The storage for pending interrupts is only one deep. This means that a second interrupt condition cannot be stored if the first one has not been acknowledged.

Q: Does an interrupt mask word have to follow the interrupt control word (assuming bit 4 was set) if the PIO is not programmed for Mode 3 operation?

A: Yes. Follow the interrupt control word with a dummy write to reset the PIO's write control logic.

Q: How can you get two PIOs to talk with each other in Mode 2 operation?

A: Suggest using ARDY1 and BRDY2 to generate a strobe pulse for ASTB1 and BSTB2. Same setup could be used for ARDY2 and BRDY1 and for ASTB2 and BSTB1. The logic basically consists of a 74LS123 (one shot) and a 74LS08 (AND gate). The ARDY1 and BRDY2 signals are ANDed together and supplied as B-TRG for generating ASTB1 and BSTB2. The ARDY2 and BRDY2 signals are ANDed together and supplied as the A-TRG for generating BSTB1 and ASTB2. The A-TRG for generating ASTB1 and BSTB2 is always low (grounded). In this manner, the port control signals are used to set the priority for strobe signal generation.

Please refer to Figure 2 and Table 1.

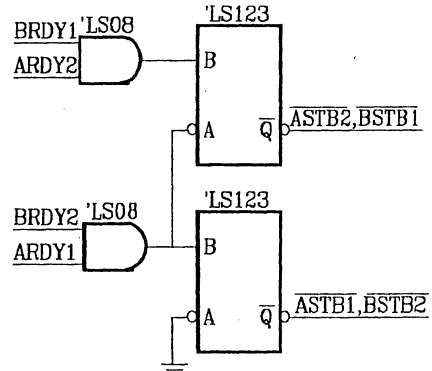


Figure 2. I/F circuit example

ARDY1	BRDY2	BRDY1	ARDY2	ASTB1 - BSTB1	ASTB2 - BSTB1	Direction
0	0	0	0	1	1	
0	0	0	1	1	1	
0	0	1	1	1	1	
0	0	1	1	1	0	2→1
0	1	0	0	1	1	
0	1	0	1	1	1	
0	1	1	0	1	1	
0	1	1	1	1	0	2→1
1	0	0	0	1	1	
1	0	0	1	1	1	
1	0	1	0	1	1	
1	0	1	1	1	0	2→1
1	1	0	0	0	1	1→2
1	1	0	1	0	1	1→2
1	1	1	0	0	1	1→2
1	1	1	1	0	1	1→2

Table 1. Truth Table for strobe signal generation

Q: How can the PIO be reprogrammed without having pending interrupts locking the system?

A: Try the following procedure.

1. Disable CPU interrupts;
2. Disable interrupt in the PIO;
3. Clear any pending interrupts within the PIO by using the interrupt control word with bit D4 set;
4. Reprogram the PIO as desired; and
5. Re-enable CPU interrupts.

Q: The PIO generates false interrupts during the programming sequence. What can cause this?

A: This symptom is almost always the result of a programming error. Depending upon the details of the problem, there are several solutions.

1. The interrupts should be enabled last in the initialization sequence. The Interrupt Control Word should be written with interrupts disabled so that the logical interrupt equation should be set (Mode 3). Finish the initialization with the Interrupt Enable Control Word (83H) to enable the interrupts.

---

2. The STB and RDY signals should be in a defined state. A transition on the STB input could cause a pending interrupt to be stored and executed as soon as interrupts are enabled.

3. A change in bit pattern (while in Mode 3) may cause an interrupt. A defined state for external inputs is recommended for power-up sequences.

Q: How can I get around the fact that only one "bit set" can be detected at a time in the OR bit mode?

A: One possibility would be to "mask" that bit during the interrupt service routine. Another possibility is to use external hardware to "mask" the bit.

Q: How can I get the PIO to give me interrupts on both transition of an input signal (Mode 3) ?

A: One method to use would require the PIO to be reprogrammed with a different logic equation during the interrupt service routine for the first transition. When the second transition occurs, then a new interrupt can be generated and the logic equation could be set back to its original state. Another method would require the use of external hardware to change the state of interrupting bit. Some possible logic could be to use an output port along with an exclusive-or (XOR) gate to control the state of the input bits.

---

## Z80 CTC

Q: How does the software reset command to the CTC affect the rest of the CTC's operation?

A: The data book and the technical manual differ in the information that is presented on this subject. A software reset command stop the counter from counting any further. In order to start the counter again, a new time constant must be loaded into the time constant register. All bits in a mode control word will cause the operation of the CTC to be affected.

Q: What is the maximum frequency of the counter?

A: If external input is synchronized to the system clock, it's half that of the CLK (system clock) input. If it's not, 1/3 of the system clock.

Q: Are there any other uses for the CTC besides counting and timing?

A: Yes, the CTC makes a very nice interrupt controller for the Z80 bus. By programming the counter for a terminal count of one and defining the transition of the trigger, you can interface non-vectored interrupting devices onto the Z80 bus.

Q: How can I have control over an individual counter so that it cannot be started, stopped, and started again ?

A: Use an external gate to qualify the clock input to the counter.

Q: When does the time constant (from the time constant register) get loaded into the down-counter?

A: On the first down count ---- ? verify.

Q: The CTC product specification states that no additional wait states (other than the automatic wait state inserted by the CPU) are allowed in the I/O cycles. Why?

A: It is not that the the wait states aren't allowed, it is just that they don't accomplish anything. The data will arrive at a particular time for the read cycles, and the internal write strobe is generated as a result of the clock edges that will be available. During the read cycle, it is possible that an improper value of the down-counter could be released onto the bus if additional wait states were added (the counter could change in the middle of the read operation).

---

## Z80 SIO

This section contains the most commonly asked questions about the Zilog SIO. They are divided into following groups:

- Features
- Registers
- Interrupt
- Modem control signals
- Enable & Disable Tx & Rx, Auto enable mode
- Questions around DMA
- Internal timings
- External interface
- Asynchronous mode of operation
- Synchronous mode
- Questions about SDLC mode

---

## Features

Q: What is the maximum data rate of the SIO?

A: 1/5 of the system clock rate. So it is 1.6Mb/s max for 8MHz version.

Q: What are the differences between Z80 SIO/0, /1, /2 and /4?

A: The differences between those four devices is "a combination of Channel B Modem signals". In fact, the SIO die itself has 41 pins internally. But a 40 pin DIP package has only "40 PINs", so we made three kinds of SIO's:

1. Z80 SIO/0: Have all channel B modem signals, except TxCB and RxCB, bonded together internally.

2. Z80 SIO/1: Lacks "DTRB".
3. Z80 SIO/2: Lacks "SYNCB".

For PLCC packages we are only offering "SIO/4", which covers all, since PLCC has 44 pins to bond out all signals.

**Q:** What are the differences between the SIO and Z80 DART?

**A:** The Z80 DART (Dual Asynchronous Receiver/Transmitter) is the device which only supports asynchronous mode of operation. The functionality, internal architecture and AC/DC characteristics are identical to the SIO in asynchronous mode. Also, pin assignment of it is identical to Z80 SIO/0 with the exception of one signal name. The "SYNC" pin on SIO/0 is "RI" (Ring Indicator) on SIO/0, but the functionality is exactly the same as the SIO/0 in asynchronous mode.

---

## Registers

**Q:** How do you read the status registers?

**A:** Reads from RR0 (Read Register 0) are accomplished by simply doing a read from the SIO. Reads from RR1 or RR2 are accomplished by writing a register pointer to the SIO (WR0) and then doing a read operation.

**Q:** What happens when you read an empty FIFO?

**A:** You will read the last character in the buffer.

**Q:** How do you avoid an overrun in the receiver FIFO?

**A:** The receive buffer must be read before the recently received data character on the serial input is shifted into the receive data FIFO. This FIFO is three bytes deep. Thus, if the buffer is not read, the fifth character that just arrived caused an overrun condition. There is no set or reset bit to disable the buffering.

**Q:** When the FIFO gets locked due to an error condition, can it still receive?

**A:** The SIO continues to receive until an overrun error occurs.

**Q:** When does the FIFO buffer lock on an error condition?

**A:** The receive data FIFO gets locked when the following receiver interrupt modes are selected:

- Receive interrupt on Special condition only.
- Receive interrupt on First character or Special condition.

In both of these modes the special condition interrupt occurs after the character with the special condition has been read. The error status has to be valid when read

in the service routine. The special condition locks the FIFO and guarantee that the DMA will not transfer any character until the special condition has been serviced.

**Q:** When a special condition occurs due to parity error, will a receive interrupt for that byte still be generated?

**A:** No. In the case of Receive interrupt on Special condition only mode, the interrupt will not occur until after the character with the special condition is read. In the case of Receive interrupt on First character or Special condition mode, the interrupt is generated on every character whether or not it has a special condition.

**Q:** What is the function of the Error FIFO?

**A:** The Error FIFO buffers the error conditions status bits for each of the received characters.

**Q:** When should the status in RR1 be checked?

**A:** Always read RR1 before reading the data.

**Q:** What information is contained in the Error FIFO?

**A:** End of frame, CRC/Framing error, Receive overrun error and Parity error. These are all contained in RR1 as well. The other status offered in RR1 is not part of an Error FIFO.

The Overrun and Parity error bits are held in the FIFO until they are reset by issuing the Error Reset Command. They will not be overwritten by new error information.

**Q:** How many register pointers does the SIO have?

**A:** The SIO has one for each channel. So it's possible to set the pointers for each channel first, then accessing each channel's register afterward. But it's not recommended, since program readability gets worse.

---

## Interrupt

**Q:** What are the various Interrupting conditions?

**A:** The SIO can generate interrupts from the receiver, Transmitter and External/status for each channel (6 sources). This is a list of all conditions that could possibly generate an interrupt (one channel only listed):

Transmitter:	Transmit Buffer Empty
Receiver:	Receiver Character Available, Parity Error, Framing Error, Receive Overrun Error
External/Status: (Transition on DCD)	CTS, Sync/Hunt, Transmit Underrun/EOM, Break/Abort Detection

---

Q: Can the IP bits be set while the SIO is servicing other interrupts?

A: Yes. If the interrupting condition has a higher priority than the interrupt currently being serviced it will cause another interrupt, thus nesting the interrupt service.

Q: How many levels of pending interrupts are there and how does the internal daisy chain operate?

A: Each possible source of an interrupt (6 possible) has one level of pending interrupts. The internal daisy chain operates in the same manner as would an external daisy chain.

Q: Does the RETI Instruction reset any status register?

A: No.

Q: If the CPU does not have the Return From Interrupt sequence (RETI instruction on the Z80 CPU), how may the SIO be informed of the completion of interrupt handling?

A: This may be done by writing the Return From Interrupt command (38h) to WR0 in Channel A of the SIO.

Q: Can the IUS bits be accessed?

A: No.

Q: When do IUS bits get set?

A: The IUS bits will be set during an interrupt acknowledge cycle on the falling edge of RD.

Q: When responding to an Interrupt, can you have the following sequence:

Int Ack, Disable INT, RETI, Clear interrupt condition?

A: No. The correct sequence is : Int Ack, Disable INT., Reset.

Q: Will enabling Interrupt after a transition on the Sync/Hunt bit cause Interrupt to occur?

A: No. External/Status Interrupt should be enabled before the transition occurs.

Note: It is advisable to execute the Reset Ext/Status Interrupt command in advance, so that the status of RR0, bit D4 reflects the current condition.

Q: Why is the Reset/Status Interrupt command recommended to be used several times in SIO setup?

A: Because many of the status bits that reflect interrupting conditions are latched bits and need to be reset to reflect current status rather than what may have occurred due to earlier Interrupts (changes in state).

Q: Will the SIO continue to request interrupt if the condition has not been satisfied?

A: Yes. There are several methods that can be used to clear the interrupt conditions. If it is a transmitter interrupt, then the transmitter must either be loaded with data or the Reset Transmit Interrupt Pending command must be issued. If the interrupt is for External/Status, then the Reset External/Status Interrupt command must be issued. If the interrupt is for a receive character being available, then the receive character must be read. If the interrupt is for an error condition, then the Error Reset command must be given.

Q: What conditions cause the transmit IP to be set?

A: Either the buffer empty or the flag after CRC is being loaded.

Q: How do the external/status bits affect the interrupts?

A: The external/status interrupt structure is affected by bits D7-D3 of RR0. These bits can be "reset" by either a hardware reset, a channel reset, or by the Reset External/Status command. The first status change on any one of the five bits after the reset will cause an interrupt to be issued and also will cause all five status bits to be latched. The latching effect is caused whether or not External/Status interrupts are enabled. If the current status at the time of reset is different than the latched status, then another Interrupt request is generated immediately. To clear the interrupt structure, two resets are necessary. The configuration of the SIO can change the definition of some of these signals. If the state of the bit changes across definition boundaries, an interrupt can be generated. Issue the Reset External/Status Interrupts command after definition. To process an external/Status interrupt, the Reset External/Status Interrupts command must be issued after reading these status bits and before the RETI.

Q: Can you use the SIO without an interrupt acknowledge cycle sequence (Z80 CPU)?

A: Reset the responsible interrupt pending bit (IP). The INT line will follow the IP bit.

Q: If the CPU can be interrupted but cannot be used with vectored interrupts, how should processing be done?

A: Immediately after being interrupted, proceed in a manner similar to polling the SIO for both receive and transmit. Alternatively, the Status affects vector bit (Bit D2 in WR1) may be set and a 0 byte placed into the interrupt vector register (WR2 in channel B). Then, the contents of the interrupt vector register can be used to determine the cause of the interrupt and the channel on which the interrupt occurred. This is queried by reading register RR1 of channel B. Also, IE1 is tied high and M1 is tied high. No equivalent to an interrupt acknowledge is issued.

---

Q: When interfacing the SIO to the CPU other than the Z80 CPU, is it possible to assert M1 and IORQ at the same time as the Interrupt acknowledge cycle to simulate Z80 timing?

A: The SIO requires "Internal daisy chain settle time" even if you don't have devices other than the SIO on the interrupt daisy chain. The period for that purpose is "M1 is active but IORQ is inactive", and is at least 100nS (for 4MHz clock ; Parameter # 16, IEI-IEO delay time).

---

## Modem control signals

Q: What is the state of the transmitter output when data is no longer available in the following modes?

- a) Asynchronous?
- b) Synchronous
- c) SDLC?

A: a) In asynchronous modes, the transmitter goes into a marking state whenever all data has been sent.

b) In the synchronous mode, the SIO will send out 16 bits of CRC (2 bytes; if programmed and the Transmitter underrun/EOM Latch has been reset) followed by the appropriate number of Sync character. The line will then continue to idle sync characters.

c) In the SDLC mode, the SIO will send out 16 bits of CRC (2 bytes ; if programmed and the Transmitter underrun/EOM Latch has been reset) followed by the SDLC flag character (7Eh). The line will then continue to idle SDLC flag characters.

Q: What is the delay time for RTS/ to TxD?

A: Two Tx clocks for asynchronous and synchronous, 7 Tx clocks for SDLC.

Q: What is the delay time for the transmit buffer empty to RTS/?

A: Two Tx clocks for asynchronous gate delays for synchronous and SDLC.

Q: Does the frequency of the CTS or DCD signals have any adverse affects on the External/Status Interrupt? (even if auto enable is not programmed)?

A: Since every transition locks the External/Status latches, you could get constant interrupts (if External/Status Interrupt are enabled) or constant status latches.

Q: Is it possible to deactivate the DTR output without reprogramming WR5?

A: Only by resetting the channel or chip.

Q: Can you gate data by stretching the receive clock?

A: You can hold the clock until you have valid data. There are

no maximum specs on the RxC period, and the edges are used to sample the data. If there are no edges, no data is sampled.

---

## Enable&Disable Tx&Rx, Auto enable mode

Q: What happens to the character being assembled if the receiver becomes disabled?

A: Assembly of a character stops immediately and the character is lost.

Q: What happens to the characters already in the receive FIFO if the receiver becomes disabled?

A: They will remain in the FIFO until they are either read by the CPU or DMA, or until the channel is reset.

Q: When Auto enable bit is set, will DCD & CTS going true cause an Interrupt?

A: Interrupt will occur only on transition of DCD & CTS since both are edge triggered if WR1,D0 is set for Ext. Int enable.

However, since these are latched conditions in Status Register RR0 (D3 & D5), current status must only read after issuing Reset Ext/Status Interrupt command.

Q: In the auto enable mode, what happens when CTS goes inactive (High) in the middle of transferring a byte?

A: If the Auto Enable mode is selected, the CTS pin is an enable for transmitter (Ideally, Transmitter enable bit is ANDed with the status of CTS). So when CTS is inactive, transmit stops immediately. (The data being shifted out will be sending out completely, however).

---

## Questions around DMA

Q: Can the SIO operate with a DMA in full duplex on each channel?

A: No. The SIO has only one ready line per channel and can only operate in half duplex mode.

If full duplex operation is required under DMA control, both channels A & B need to be used ; One for transmit and one for receive.

Q: Can both channels make simultaneous DMA requests?

A: Yes.

Q: What happens when you program the SIO to interrupt on Buffer Empty and the DMA to act on Buffer Empty?

A: This would not be a wise thing to do. However the Interrupt occurs, the DMA will take over the bus before Interrupt has acknowledged. The buffer will be filled by the DMA and the Interrupt Request will go away due to a Buffer Full condition and the Interrupt Acknowledge will

---

occur causing bus confusion. The same thing occurs on Receive buffer empty interrupt and DMA on Receive character.

Q: How can the SIO/DMA combination be used for synchronous communications and ensure that the CRC characters are also transmitted?

A: Try the following procedure:

1. Initialize the SIO for use of the READY function with a DMA controller and then poll (or interrupt on) external/status (not transmit buffer empty).
2. Initialize DMA controller for data transfer and bus release at end-of-block. DO NOT ENABLE DMA YET!
3. Send first byte of data to SIO for transmission followed by a Reset Transmit Underrun/EOM Latch command.
4. Enable the DMA controller now (it should take control of the bus).
5. When the end-of-block is reached, the DMA controller should release the bus back to the CPU.

Q: When does the SIO terminate the READY signal?

A: The rising edge of the system clock that samples IORQ low causes READY to go inactive. The delay is specified by parameter 19 in the data sheet.

Q: When does the READY signal become active after an access to the SIO?

A: The READY signal will be inactive for a minimum of 5 clock cycles and will become active again 700 nS after CE goes inactive.

---

## Internal timing

Q: When the transmitter is disabled, when does the TxD line go to a marking state?

A: One bit time after the last bit of the data leaves the transmit shift register.

Q: When the transmitter is empty, does status register RR0, bit D2 indicate that the buffer is now empty or that the last data in the buffer is in the process of being shifted out?

A: It indicates the buffer is now empty. The status register has nothing to do with the transmit shift register.

Q: Does the Transmit interrupt occur when Transmit Buffer is empty or Transmitter itself is empty?

A: Interrupt occurs when the Transmit Buffer is empty.

Q: How many bit times from external clock is the Transmit Buffer Empty Interrupt delayed?

A: The interrupt occurs a maximum of 9 clock periods from the Txc clock edge that causes the buffer to become empty. The exact time is highly dependent upon the mode of operation and is transparent to the user.

Q: When is the data available at the top of the FIFO?

A: Data is available after a maximum of 13 clock periods from the rising edge of RxC.

Q: What is the delay time between transmit shift register to the TxD pin?

A: Two Tx clocks for asynchronous and synchronous, Seven Tx clocks (five for zero inserter, two for internal delay) for SDLCL.

Q: Does an Interrupt occur on RxC for last data bit assembled or does it occur relative to the RxC, but delayed?

A: Interrupt occurs when data is moved from the receive shift register to FIFO. The relationship of this event is relative to an external clock edge. This relationship however, is of no concern to the user. There is, however a specific delay from the external clock edge to the interrupt, caused by internal SIO logic.

---

## External interface

Q: Can a sloppy system clock cause problems in SIO operation?

A: Yes. The specs on this system clock are very tight and must be met to prevent SIO malfunction. The specifications are:

---

Symbol	Description	Min	Max	Unit
VIHC	Clock "H"	Vcc-0.6	5.5	Volt
VIHL	Clock "L"	-0.3	0.45	Volt

---

Clock rise/Fall time = 30nS each edge  
(For N-MOS, 4MHz device).

---

Should there be any ringing or undershoot/overshoot on the clock input, the SIO could behave in any number of indeterminable ways.

Q: Must the system clock, fed to the SIO, have a 50% duty cycle?

A: The duty cycle doesn't have to be 50% as long as the minimum specification is met.

Q: Are input control lines to the SIO synchronized to system clocks so that garbage may exist on the buses anytime before setup requirements are satisfied?

A: Yes.

---

Q: Since setup time for CE and IORQ may be satisfied during T2, is time T1 required?

A: If the Z80 CPU is being used, then T1 timing state is required in order to utilize the interrupt structure. (interrupt request, acknowledge, and RETI).

No, if not using the Z80.

Q: Do wait states have to be added to provide I/O response to the SIO in non-Z80 based systems? (The Z80 adds wait states automatically)

A: No. As long as setup times as specified for the SIO are met. The SIO does not know about wait states inserted by the Z80. The Z80 puts in wait states in order to match the Z80 SIO setup times.

Q: What pins are noise sensitive and should be strapped to avoid strange interrupts?

A: The Ext Sync pin, and any Ext status pin that is not used. Also, all inputs are sensitive to signal ringing and undershoot problems.

Q: Is M1 required if no interrupts are used in the SIO?

A: No. M1 should then be tied high.

Q: Can you use the Ready output for an Interrupt request?

A: Yes, for byte move action in or out and Respond to Interrupt. However, it is not recommended to use Ready for Interrupt with the CPU.

Q: How long must RD and the other control signals remain active?

A: Although RD and IORQ are latched internally, they must remain active for a minimum of two system clock periods.

Q: Are there any timing specifications for "Access recovery time"?

A: No.

---

## Asynchronous Mode

Q: Why are there different Clock factors?

A: These clock factors enable the SIO to sample the center of the data cell. In the X16 mode, the SIO divides the bit cell into 16 counts and samples on count 8.

Q: For asynchronous mode of operation, must the clock rates selected be the same for Receiver and Transmitter?

A: No. However, the multiplier for both RxC & TxC must be the same because of internal logic.

Q: When running in the Async mode, is it necessary to use the X16 clock scalar?

A: No, X1 synchronization can be selected but the user must maintain data synchronization with the clock. The start bit detection logic does not work in X1 mode and the 1.5 stop bit cannot be used in X1. In other words, X1 Mode for Async mode is NOT asynchronous mode, it's a "clocked serial channel".

Q: What does the SIO recognize as the Break character?

A: A character of all zeros including stop bits (indicating a framing error).

Q: When attempting to detect a break condition by sensing the break/abort status bit, is it necessary to enable External/Status interrupts?

A: No. The External/Status latches work regardless of whether or not the external/status interrupts are enabled. This can be confusing because once the latches are strobed, the status in RR0 is frozen until a Reset External/Status Interrupt command is issued. If you desire the true current status, issue this command before reading RR0.

Q: Can a break sequence be sent for a fixed number of character periods?

A: Yes. Break is continuously transmitted as logic 1 by setting bit 4 of WR5. You can then send characters to the transmitter as long as the break level persists. A Break signal rather than the characters sent is transmitted, but each bit of each character sent will be clocked as if it were transmitted. The All sent bit, bit 0 of RR0, is set to 1 when the last bit of a character is clocked for transmission. This may be used to determine when to reset bit 4 of WR5 and stop the Break signal.

Q: If a Break sequence is initiated by setting bit 4 of WR5, will any character in the process of being transmitted, be completed?

A: No. Break is effective immediately when bit 4 of WR5 is set. The "all sent" bit in RR1 should be monitored to determine when it is safe to initiate a Break sequence.

Q: When using the SIO only in Asynchronous mode, can the SYNC pin have any use?

A: It may be used as a general purpose input. For example, by connecting it to a modem ring indicator, the status of that ring indicator can be monitored by the CPU.

Q: How can the SIO be used to transmit characters containing fewer than 5 bits?

A: First, set bit 6 and 5 in WR5 to indicate that five or fewer bits per character will be transmitted. The SIO then determines the number of bits to actually transmit from the data byte itself. The data byte should consist of zero or more 1s, three zeros, and the data to be transmitted. Thus, beginning the data byte with 1111001 will cause only the last bit to be transmitted.

---

Contents of data bytes(D=arbitrary value)

	D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	0	0	0	d
2	1	1	1	0	0	0	d	d
3	1	1	0	0	0	d	d	d
4	1	0	0	0	d	d	d	d
5	0	0	0	d	d	d	d	d

---

## Synchronous Mode

Q: Can you cause interrupts on CRC error bit (RR1;D6) changes?

A: No. The CRC error status is not one of the special receive conditions. Perhaps, explanation of cyclic redundancy block checking and how the SIO operates for CRC is relevant.

---

## CRC

Cyclic Redundancy Checking is a method of checking for errors in serial data transmission. It is also known as the polynomial error code check. The polynomial is an algebraic function used to create a constant from the message bit pattern. This constant, generated and accumulated in both the Transmitter and Receiver, is used to divide the binary numeric value of the character. The quotient is discarded and the remainder added to the next character, which again is divided. This continues until the last character, when the remainder is transmitted to the receiver for comparison with the Receiver's remainder. An equal comparison indicated no errors, while an unequal comparison indicates an error in transmission.

---

## SIO-CRC

The SIO contains CRC generation and checking in the Transmitter and Receiver.

It allows for either of two polynomials to be used.

a)  $X^{16}+X^{15}+X^2+1$ : Called CRC-16, generally used in synchronous communication.

b)  $X^{16}+X^{12}+X^5+1$ : Called CRC-CCITT, generally used in SDLC communication and also recommended by the CCITT.

---

## CRC Error Check

Status register RR1 which contains error conditions, allocates bit D6 for CRC error status. Since CRC checking is a continuous process and takes place character

by character and intermediate results are shifted into the Receive Error FIFO continuously, bit D6 of RR1 is continuously updated because it is not latched.

However, checking the status of this bit at any intermediate point in time in the middle of a transmission is meaningless. It must be remembered that the result of a CRC check is valid only on completion of a message. Also, in most cases, bit D6 will usually be a "1" in the middle of a message since most serial bit combinations result in a non-zero CRC. Unless it is at completion of a message transmission.

The SIO does not generate an Interrupt for CRC Error Status.

Q: Suggest a hardware way to count or determine when the 16 or 20 bit times have passed before the CRC check is valid in BiSync mode?

A: Allow two "buffer full" interrupts to occur to determine that 16 bit times have elapsed, or have an external clock count 20 bit times.

Q: How do you read the CRC error status bit when receiving data in the bisync mode?

A: This is one possible method.

After two CRC bytes have been received and read, wait for the next receive character interrupted and stop CRC accumulation, then read the next received character. After the next character is interrupted, disable the receiver and read the status byte.

Q: In switched carrier Bisync application, the clock may go away before CRC calculate is complete since only one pad will be received. How can valid CRC be ensured?

A: SIO spec requires at least two pads for a valid CRC check in Bisync mode.

Q: Is CRC enabled automatically after first data in a non-SDLC node?

A: Only if it is programmed to be so.

Q: Are Sync patterns (or flags) included in CRC?

A: SDLC - No.

Yes for Bisync - CRC must be turned on/off as required or Sync will be included in CRC.

Q: In synchronous mode, does CRC get stripped from data?

A: Not normally, but it is possible if the CRC byte happens to match the contents of WR6 and the sync character load inhibit feature is enabled.

Otherwise, SIO won't delete the CRC bytes from the data stream.



---

Q: What is the proper sequence for a valid reading of the CRC error status bit?

A: To check the CRC error status and read the CRC bytes. The following sequence is recommended because of delays in Receive logic and the time at which EOM Interrupt occurs.

1. Interrupt, read and discard - 1st CRC byte.
2. Interrupt, read and discard - 2nd CRC byte.
3. Interrupt, read 1st pad character and discard.
4. Disable CRC
5. Interrupt, read CRC status then read 2nd pad and discard.

Q: In Monosync, is the Sync Comparison done in the Receive shift register or the Sync register?

A: Sync comparison is done in the Sync Register against the contents of WR7.

Q: For Monosync, which register contains the Sync character for comparison?

A: Write Register 7. Comparison is done in the Receive Sync Register.

Q: How does the SIO avoid losing a single sync character in the case of back-to-back Bisync messages that are separated by a single sync pad?

A: It does not. The SIO loses sync characters because the Bisync spec requires a minimum of two pad characters.

Q: Do Sync patterns (or flags) in data get stripped and still cause Interrupts?

A: All leading sync patterns (and all flags) are stripped automatically. In SDLC, sync characters (flags) will cause Interrupts if programmed to. Sync characters may or may not be stripped in Bisync depending on the state of the Sync Character Load Inhibit Bit (WR3,D1). Any data stripped from the data stream cannot cause a receive character available interrupt but may cause other interrupts (such as External/status for Sync/Hunt and special receive condition for EOM). In SDLC, programming Sync Character Load Inhibit will cause stripping of the address field and not cause Interrupts.

Q: Do interrupts occur after each Sync pattern?

A: Yes, if programmed to do so (External/Status interrupts).

Q: Do sync patterns automatically get transmitted in Bisync mode when Transmit Buffer becomes empty?

A: Yes, but the CRC bytes may be allowed to precede those sync characters.

Q: How does the SIO handle synchronous protocols which use less than 8-bit sync characters?

A: The sync character match logic within the SIO only makes comparison on 8-bit boundaries (8-bits for

monosync and 16-bits for bisync). In order to match on patterns that are not integer multiples of 8-bits, the sync character must overlay the pattern stored in the sync register, or use "External Sync mode".

Q: Are sync characters subject to parity?

A: No.

Q: Assuming that there are characters available in the FIFO, what happens to them if the receiver goes into the hunt mode?

A: They will remain in the FIFO until they are either read by the CPU or DMA, or until the channel is reset.

Q: Is sync character transmission suppressed when the SIO is programmed for external sync operation?

A: Yes.

Q: How is it possible for the SIO to achieve synchronization on erroneous sync patterns (in monosync and bisync modes)?

A: The design of the SIO is such that the sync register serves as the CRC delay register after synchronization has been achieved. If the SIO goes out of synchronization or is placed into the hunt mode, the CRC delay register again becomes the sync register but it's contents are not cleared. Any data in it can be used by the comparison logic for synchronization. The best solution is to disable the receiver each time you place it in hunt mode and then re-enable it. This sequence will reset the contents of the sync register.

---

## SDLC mode

Q: How does the SIO send CRC?

A: The SIO can be programmed to automatically send the CRC. First, write the first byte of the message to be sent. This guarantees the transmitter is full. Then, reset the Transmit Underrun/EOM latch (WR0;10h). Write the rest of the data frame. When the transmit buffer under-runs, the CRC will be sent. The following table describes the action taken by the SIO for the bit oriented protocols.

---

Tx Underrun	Action	Comment
EOM Latch Bit	Upon Tx Underrun	
0	Send CRC+Flags	Valid Frame
1	Send Flags	Software CRC

---

Q: In SDLC mode, when do you get the End of Message (EOM) interrupt?

A: The EOF interrupt occurs after the 1st CRC is loaded to the transmit buffer and 2 bit times before the 2nd CRC is loaded to the buffer.

---

Q: How can you make sure that a flag is transmitted after CRC?

A: Use the external status End of Message (EOM) interrupt to start the CRC transmission, then enable the transmit buffer empty interrupt. When you get the interrupt, it means that the buffer is empty, a flag is loaded in the shift register, and you can send the next packet of information.

Q: When using the SIO in the SDLC mode of operation, the transmitter loses two data bits (gets shifted by two bit positions) when the last character before the closing flag is transmitted. Transmit data is looped to the receiver and CRC is not enabled?

A: The transmitter is working. The receiver actually causes the shift of two bits upon recognition of the closing flag.

Q: Why is the second CRC byte in the receiver FIFO not the full CRC byte?

A: The 2nd CRC byte read from the FIFO is not the full 8 bit byte. The transmitted byte is made up of the last two bits of the 1st CRC byte and the first 6 bits of the 2nd CRC byte. This is because of the delay in the Receive path and the point in time when the EOM Interrupt occurs causing transfer of contents of the Receive Shift Register into the FIFO.

However, since the above 2 bytes are to be discarded by the user, it does not matter.

Except in cases where users may want to include two bytes of data in place of CRC, it is important to note that the last byte will be off by two bits.

Q: In SDLC, when do you reset the CRC generator and checker?

A: The reset Tx CRC generator command should be issued when transmitter is enabled and idling (WR0). This needs to be done only once at initialization time for SDLC mode.

Q: If the SIO is idling flags and a byte of data is loaded into the transmit buffer, what will be transmitted?

A: Data takes priority over flags, and will be loaded into the shift register and transmitted.

Q: What does the SEND ABORT command do to the SDLC transmit sequence?

A: The transmission of the current character is aborted and a sequence of 8-one's are inserted into the data stream. This means that the user may see between 8 and 13 one's in the data stream because of the zero inserter. If there is data in the transmit buffer, it is destroyed and a Transmit Buffer Empty interrupt is pending.

Q: Can the SIO detect multiple aborts?

A: The SIO searches for seven consecutive 1's on the receive data line for the abort detection. This condition may be allowed to cause an external status interrupt. After these seven 1's are received, the receiver automatically enters Hunt mode, where it looks for flags. So even if more than seven 1's are received in case of multiple aborts, only the first sequence of 1's is significant.

Q: In the SDLC mode of operation, what is the relationship between the TxD output and transmitter interrupts?

A: Transmitter interrupts occur when the data from the transmit buffer is loaded into the transmit shift register. The output to the TxD pin is delayed by 6 bit times (five for the zero inserter and one for the pin delay) from the last bit leaving the shift register.

Q: Is it possible to monitor all received characters, including flags, in the SDLC mode?

A: No. if you want to monitor everything, then use the SIO in another mode of operation where the sync pattern has no special meaning.

Q: Can interrupts be generated on the idle line flag in SDLC?

A: Upon receipt of seven continuous ones, the break/abort bit will be set to indicate the abort condition. This bit will remain set until the SIO receives a zero.

Q: Does Hunt in SDLC continue until the Address Field has been recognized?

A: It does if the address search mode feature has been programmed.

Q: Does IBM SDLC specify parity?

A: No.

Q: Can the SIO include parity in SDLC mode?

A: Yes. It is appended at the end of the character.





# Z180™ Family

## QUESTIONS AND ANSWERS

September 1989

*This application note contains the most commonly asked questions about the Z180. Obviously, not every possible question on the Z180 is answered. However, this application note should give you a good feel for how the Z180 works. Along with the technical manual and product specification, it should facilitate your Z180 design.*

### FEATURES

**Q: What are the differences between the Z180, Hitachi's HD64180R0/R1 and Z versions?**

A: Our Z180 is identical to Hitachi's HD64180Z version except some of the signal names are different in order to match Z80 signal names (Z180 was jointly developed with Hitachi).

The HD64180R0 version is the original version, and has some bugs in it. The R1 version is the version which corrected the bugs in the R0 version. The 68-pin PLCC and 80-pin QFP versions have 1M bytes of physical memory address space. Also, There is a "test" pin (output; not open to user) assigned to the pin which is not used on the R0 version.

The Z version corrected the R0 and R1 problems (the problems involve the Z80 peripheral interface.)

**Q: In Z mode of operation (M1E of OMCR cleared to 0), there is no interrupt from the Z80 PIO after enabling the Z80's PIO interrupt. Why?**

A: Write zero to the /M1TE bit of OMCR after enabling the Z80's PIO interrupt. Because the Z80's PIO interrupt control logic requires /M1 to activate its interrupt logic while /M1 only occurs during Interrupt Acknowledge and RETI cycles with /M1E cleared.

### CPU CLOCK

**Q: Can I stop the clock in order to minimize the power consumption?**

A: No. However, one possible way is save the registers into battery backed up RAM, and then remove power from the CPU.

**Q: What is the relationship between EXTAL and PHI clock output when an external clock is input through EXTAL?**

A: PHI output changes its status on the falling edge of

EXTAL input. And the delay from the falling edge of EXTAL to PHI is about 30nS (Reference only; not guaranteed value)

**Q: In our system, sometimes the PHI frequency is the same as XTAL frequency, not XTAL divided by two. Why?**

A: Please check the following points:

- Reset is held low at least 6 clock cycles.
- The status of ST line during reset. ST should not be tied low (ST line is the OUTPUT signal!)

### INSTRUCTIONS

**Q: Which instruction, RET or RETI, is used at the end of an interrupt service routine?**

A: If you don't have any Z80 peripherals (do not use "interrupt daisy chain"), then you can use either of these instructions at the end of interrupt service routine (for interrupt from on-chip peripherals). If you have Z80 peripherals on board, then you should use RETI for the interrupts of Z80 peripherals, and RET for the interrupts of on-chip peripherals.

The reason, from CPU stand point, is both instructions are the same (POP PC value from stack and return). But Z80 peripherals are looking for RETI sequence on the bus to correct its interrupt daisy chain status upon receiving that sequence. If you have Z80 peripherals and are using RETI for the interrupt for on-chip peripherals, Z80 peripherals are confused and thus set the wrong daisy chain status.

**Q: Is the instruction set of the Z180 fully identical to the Z80 CPU's except for new instructions?**

A: There are three instructions which are not the same. They are: DAA and RRD/RLD.

For DAA (Decimal adjust), if you execute this instruction

---

## INSTRUCTIONS (Continued)

after DEC instruction (especially DEC instruction on 00h, then execute DAA), Z180 results in F9H while Z80 results in 99H. It is because the Z80 CPU refers "Internal Carry flag" while the Z180 doesn't.

For RLD/RRD (Rotate Left/Right Digit), Z180's flag will reflect the contents of the memory location pointed by HL register, while Z80 reflects the contents of the Accumulator.

But, there are very few applications which use DAA instructions after DEC and use flag after RLD/RRD instructions.

---

## REFRESH

**Q: Is the functionality of R register the same as Z80 CPU's?**

A: No. Z180's R register is counting M1 cycles, and there is no relationship with current refresh address.

**Q: Is the refresh mechanism of the Z180 different from the Z80 CPU's?**

A: Yes. Z180's refresh mechanism is "periodic refresh" and the refresh period can be programmed and disabled. Refresh address is 8 bit. On the Z80 CPU, refresh cycle is inserted after every M1 cycle and can not be disabled.

**Q: Can the refresh cycle occur during an on-chip DMA cycle?**

A: Yes. Because Z180's refresh mechanism can not distinguish whether current activity is by CPU or on-chip DMA. Refresh cycle will be inserted after the end of DMA or CPU machine cycle.

---

## I/O ADDRESSING SPACE

**Q: Z180's technical manual states that I/O addressing space is 64K Bytes while Z80 CPU's I/O address space is only 256 Bytes. How do you access "expanded I/O address space"?**

A: In fact, Z180's I/O addressing space is the same as Z80 CPU's. You can specify lower half (A7-A0) of I/O address directly, as on Z80 CPU, but the upper half depends on the instruction which you are using for access. There are four groups of I/O instructions on Z180 (On Z80 CPU, group D does not exist)

A) 8080 type instruction  
IN A,(n) OUT (n),A  
A15-A8 ←Acc

B) C register indirect  
IN A,(C) OUT (C),A  
A15-A8 ←B register

C) C register indirect with auto increment/decrement  
IND INDR INI INIR OUTD OTDR OUTI OTIR  
A15-A8 ←B register (B register is loop counter)

D) Z180 original instructions which force A15-A8 to 0  
IN0 OUT0 OTIM OTIMR OTDM OTDMR TSTIO  
A15-A8 ←0

To utilize "64K Bytes" of I/O address, you can use group A) or B) instructions with care, or use group D) instructions to access the "Page zero" I/O address space.

**Q: To access on-chip peripherals (and system control registers), Should A15-A8 be zero?**

A: Yes. It is a good idea to use Z180's new I/O instructions for that purpose (These instructions force A15-A8 to 0).

**Q: What happens if off-chip peripheral's address is assigned to the internal I/O devices (overlapped)?**

A: I/O read: data from addressed internal peripheral is read, and the data on the bus at that time is just ignored. I/O write: output the data to the data bus as well as to on-chip peripherals. Also, this transaction could write the data to off-chip peripherals.

---

## BUS TIMING

**Q: For the Interrupt Acknowledge cycle timing chart, there are no timing specifications for PHI to M1 and PHI to IORQ. Do you have these numbers?**

A: Yes. These parameters are as follows:

PHI rising edge to /M1 falling edge (Interrupt Acknowledge cycle): Same as parameter #10 ( $t_{M1D1}$ )

PHI falling edge (of first  $T_{wa}$  state) to /IORQ falling edge (Interrupt Acknowledge cycle): Same as parameter #28 ( $t_{IOD1}$ , Case IOC=1).

PHI rising edge to /M1 rising edge (Interrupt Acknowledge cycle): Same as parameter #14 ( $t_{M1D2}$ ).

PHI falling edge to /IORQ rising edge (Interrupt Acknowledge cycle): Same as parameter #29 ( $t_{IOD2}$ ).

**Q: What about the bus status during the access to the on-chip peripherals?**

A: During I/O access to the on-chip peripherals,

I/O read: Data bus - Hi-Z  
Address bus: holds the I/O address

I/O write: Data bus - Data to be written  
Address bus: Holds the I/O address

**Q: During sleep or bus release mode, is it possible to extend the E signal pulse width by inserting wait states?**

A: No. during sleep or bus release mode, the CPU won't sample the status of /WAIT input (cannot extend the cycle).

**Q: What is the timing of E output during -DMA, -Refresh, -I/O cycle?**

A: For refresh cycle, E output will be held low. For DMA and I/O cycle, E timing is identical to the CPU cycle. Thus:

E is active during...  
Memory R/W: T2 rising edge to T3 falling edge.  
I/O read: 1st Tw rising edge to T3 falling edge.  
I/O write: 1st Tw rising edge to T3 falling edge.

**Q: Does the Z180 sample the data at the different points during memory read and op-code fetch cycles, as the Z80 CPU does?**

A: Yes. The data is sampled on the rising edge of T3 during op-code fetch cycles, falling edge of T3 for memory read cycles.

## WAIT STATES

**Q: When using automatic wait state generator, does the Z180 sample external /WAIT state before automatic wait state insertion, or after?**

A: External /WAIT status will be sampled after the automatic wait state insertion.

**Q: Is it possible to insert wait state(s) into a refresh cycle by /WAIT input?**

A: No. WAIT is not sampled during refresh cycle. However, Z180 has a capability to insert "software" wait state into refresh cycle by setting REFW (Bit D6) of RCR register to 1.

**Q: Is the automatic wait state during I/O cycle removable?**

A: No.

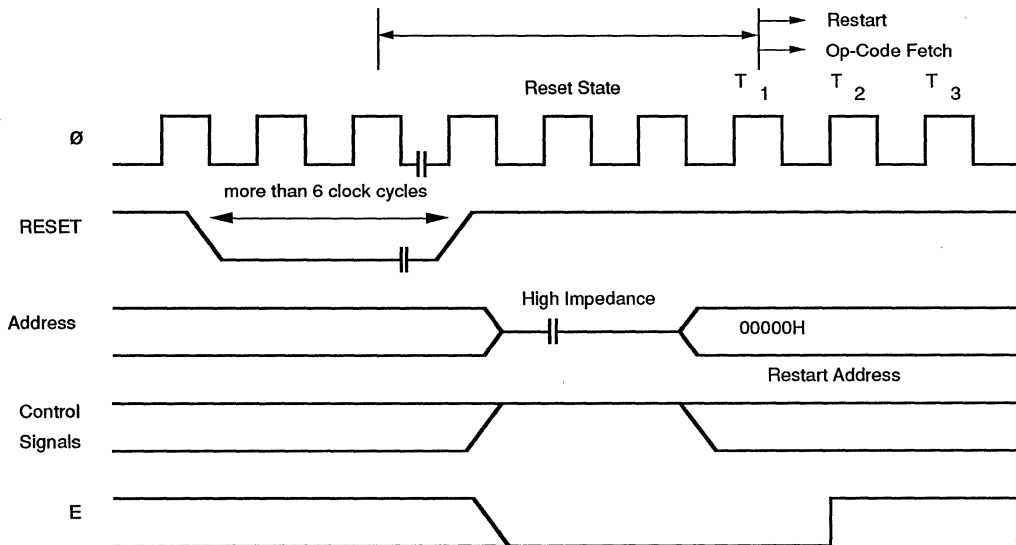
**Q: When accessing on-chip peripherals, it seems that required access time varies case by case. Why?**

A: When accessing on-chip peripherals (ASCI, CSI/O, PRT data registers), zero to four wait states are automatically inserted depending on the status of CPU and peripherals. In those cases, the value set in DAM/WAIT control register is ignored.

## RESET

**Q: How is the power-on reset sequence performed?**

A: The power-on reset sequence is as follows.



Note that /RESET pin should be asserted low for at least 6 clock cycles to perform power-on reset correctly.

## POWER SAVE MODES

**Q: How to exit from SYSTEM STOP mode, and what is the CPU's response after exiting this mode?**

A: /NMI, /INT (external) or RESET is necessary to exit from SYSTEM STOP mode. If receiving RESET, normal RESET sequence takes place. In case of /NMI, /NMI sequence takes place. In case of external /INT:

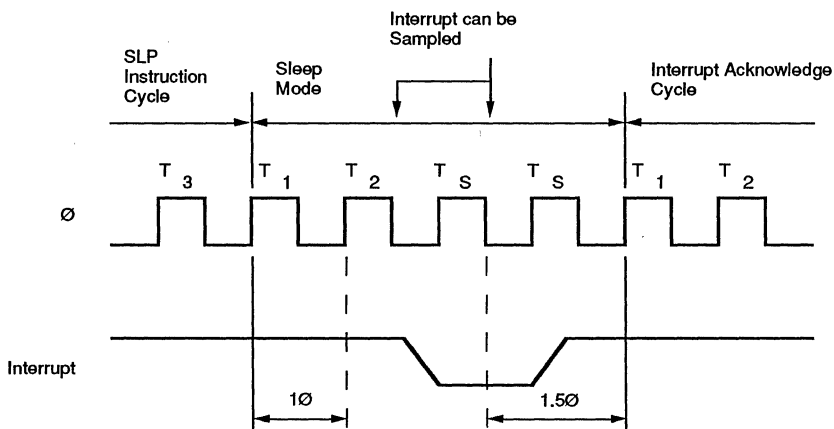
If interrupt is globally disabled (IEF1=0):  
CPU will execute the instruction following the SLEEP instruction.

If interrupt is globally enabled (IEF1=1):  
Appropriate normal interrupt response sequence will be performed.

Except for RESET, I/O STOP mode is maintained until the I/O stop bit is cleared to 0 after exiting from SYSTEM STOP MODE.

**Q: During SLEEP mode, when is the CPU sampling the status of INT line?**

A: The CPU starts to sample the status of INT line on the falling edge of the 1.5 clock cycles after entering the SLEEP mode. When CPU samples INT as active, 1.5 clock cycles later CPU will wake up from SLEEP mode.

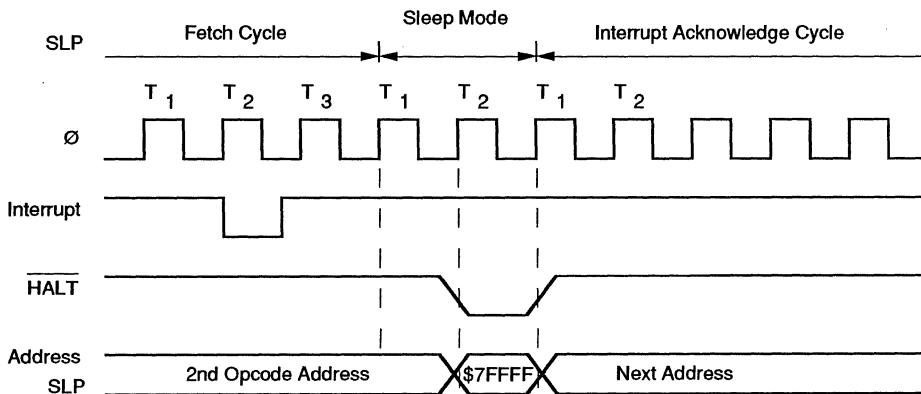


**Q: What is the status of the bus during SLEEP mode?**

A: A0-A19 are all one, /HALT stays low, and /MREQ, /M1, /RD stay high.

**Q: What happens if an interrupt is received during the execution of the Sleep (SLP) instruction?**

A: If an interrupt is received during a SLP instruction, /HALT is asserted low for one clock cycle and the address bus is set to all ones, and is then followed by an interrupt acknowledge cycle, as shown below.



## TRAP

**Q: What happens if another trap condition (detecting undefined op-code) occurs before clearing the TRAP bit of the ITC register?**

A: You will have another TRAP. If the trap handling routine causes a TRAP it will probably get into an infinity loop, and crash. For this case, the TRAP bit remains 1 and the UFO bit shows the status for the previous TRAP, since the status of the UFO bit cannot be changed while TRAP=1.

**Q: What is the purpose of the UFO bit?**

A: UFO bit shows whether TRAP occurred during 2nd op-code fetch cycle or 3rd. If it is zero, TRAP has occurred in the 2nd op-code fetch cycle and the PC value pushed onto the stack is, 1st op-code address + 1. And if it is one, TRAP has occurred in the 3rd op-code fetch cycle and the PC value pushed onto the stack is, 1st op-code address + 2. All hex numbers for first op-code on Z180 are allocated for instructions, so TRAP won't happen on 1st op-code.

**Q: How do you determine the op-code address which caused TRAP using the UFO bit?**

A: Upon TRAP, the PC value pushed onto the stack for the address of the instruction which caused TRAP is:

Instruction start address =  
(value on the stack) - 1 when UFO=0  
Instruction start address =  
(value on the stack) - 2 when UFO=1

So that you can determine the starting address for the instruction which caused the TRAP.

## NMI AND INT

**Q: Are the addresses of interrupt vectors treated as physical addresses or logical addresses?**

A: Z180 always treats those addresses as Logical addresses. So if you enabled the MMU, care must be taken.

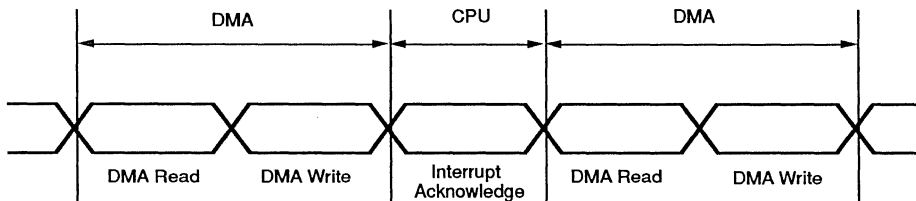
Here is one suggestion: If you can make the vector table in Common Area 0, the physical address is the same as the Logical address, which helps.

**Q: What happens if an interrupt is received during an on-chip DMA operation?**

A: If it is an /NMI, DMA operations will stop and an /NMI acknowledge cycle will be initiated. If it is /INT or interrupt from on-chip peripherals:

During Mem-Mem burst mode transfer, the interrupt is just ignored.

During Mem-Mem cycle steal mode of operation, interrupt acknowledge cycle is initiated between DMA cycles (See following figure).



**Q: For interrupt mode 2, shall I set bit D0 of interrupt vector to 0, as Z80 requires?**

A: Yes. However, the Z180 works correctly even if bit D0 of the interrupt vector is 1.

**Q: Is /NMI acknowledged during an interrupt acknowledge cycle?**

A: Yes. one instruction (except EI and DI instruction) is executed after /INT acknowledge cycle, then the /NMI acknowledge cycle starts. It is also the same for /NMI received during /NMI acknowledge cycle.

**Q: Are /NMI or /INT acknowledged immediately after RESET?**

A: No. /NMI and /INT are disabled for three clock cycles immediately after RESET (power on reset). After those three cycles, the instruction is executed (from 0000h) and then interrupt is enabled. Note that /NMI status is latched internally right after the power on reset cycle.

**Q: Is an interrupt (/NMI or /INT) acknowledged during a refresh cycle?**

A: /NMI status will be latched internally if /NMI occurs during the refresh cycle. When current instruction execution is completed, the following cycle will be an /NMI acknowledge cycle. /INT status is not sampled during a refresh cycle, and it will be sampled during the following instruction execution cycle.



## NMI AND INT (Continued)

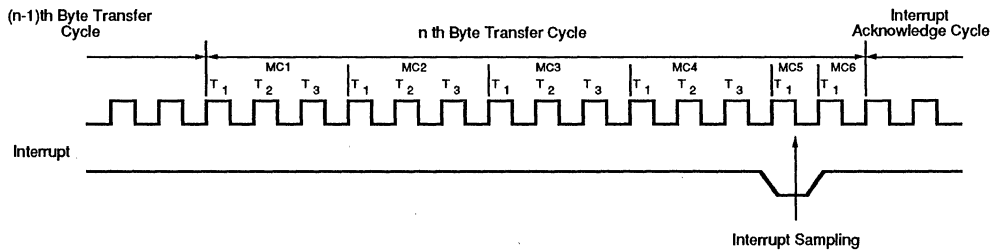
**Q: I have an /NMI input with a pulse width of a couple of hundred nS. Can it be accepted?**

A: The /NMI input will be accepted as long as you can satisfy the /NMI pulse width specification (120nS min), since /NMI is edge triggered input and its status is latched internally.

**Q: During the execution of EI instruction, can an interrupt be acknowledged?**

A: No. During execution of an EI instruction, interrupt is not sampled. So if interrupt (Maskable interrupt) goes active during the instruction just before an EI instruction, or during EI instruction, it will be acknowledged during the following instruction.

**Q: When does the Z180 sample an interrupt status?**



**Q: During interrupt Mode 0, if a call instruction is put onto the bus during the interrupt acknowledge cycle, software won't return the correct address. Why?**

A: The PC value pushed onto the stack during the interrupt acknowledge cycle is the PC value for the next instruction if the instruction put onto the bus is the RST (restart) instruction (one byte instruction).

However, call instruction is a three byte instruction and the return address pushed onto the stack by that Call instruction will be the PC value plus two (PC value stays the same during interrupt acknowledge cycle, and incremented by two during operand fetch for jump address). So at the end of the interrupt service routine, you need to decrement the return address pushed onto stack by two.

**Q: What happens if there is /NMI going active and at the same time (or just before) /DREQn is also going active when using DMA with /DREQn input?**

A: The input on /DREQn is ignored. To restart DMA, please refer to previous Q&A.

**Q: When interrupts from on-chip peripherals are sampled, is it the same as the external interrupts?**

A: Yes. Interrupt status is sampled on the falling edge of T2 or T1 cycle of the last machine cycle of the instruction.

A: The Z180 samples an interrupt status at the falling edge of PHI one cycle prior to the last machine cycle of each instruction (excluding EI instruction).

Also, the status of the internal /NMI latch is sampled at the same time.

**Q: During execution of block transfer instructions, doesn't the Z180 sample an interrupt status until the end of block transfer?**

A: No. The Z180 checks the status of interrupt at the end of every sequence, as shown below. Note that if you accept interrupt during block transfer, your interrupt service routine will not destroy the contents of registers in order to resume from the interrupt correctly (save the registers using EX/EXX instructions at the beginning of interrupt service routine).

**Q: Is there any way to connect 8259 (interrupt controller) under mode 0 interrupt?**

A: When using Mode 0 interrupt, which is "8080 compatible interrupt mode", there is one important point, which is "concern about the INTA pulse".

Mode 0 is the mode which maintains the "software compatibility" with the 8080. That means in this mode, during the INTACK cycle, the Z180 fetches the data on the bus as an "instruction" and executes it, like the 8080.

However, from the hardware stand point, it is not true. Because the 8080 generates three INTA pulses during the interrupt acknowledge cycle, while the Z180 generates only one INTACK signal (which can be decoded from /M1 and /RD).

This system works fine if you are not using the 8259 and put "RST" (restart) instruction onto the bus during the Interrupt acknowledge cycle which is a one byte instruction. However, if you want to use the 8259 with the Z180, you'll have a problem, which is:

The 8259 requires three INTA pulses but the Z180 generates only one INTACK cycle.

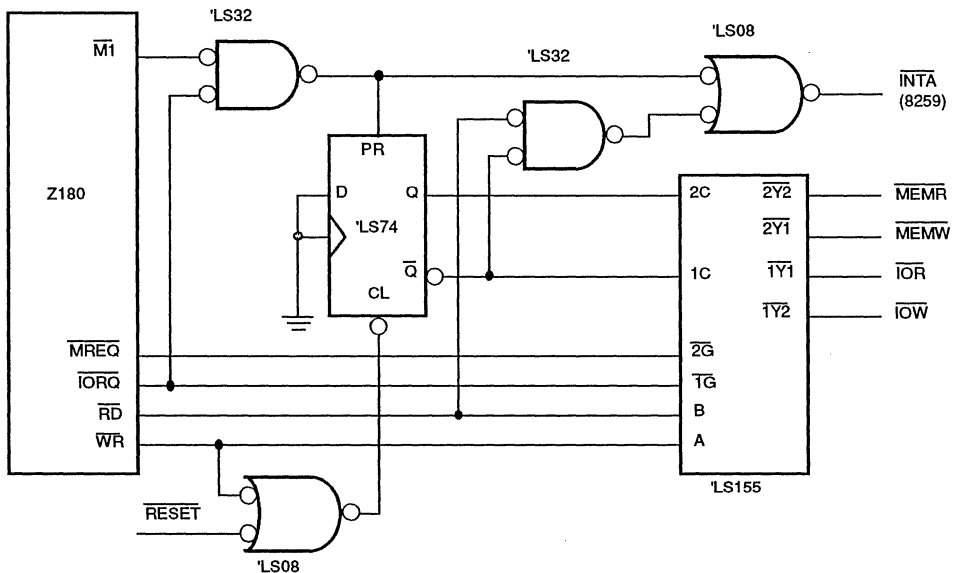
The best way to solve the problem is to "simulate an 8080 interrupt acknowledge cycle" - generating a total of three "INTA" pulses for 8259 upon the Z180's interrupt acknowledge cycle by external logic.

The following figure is one example of the implementation. This circuit works as follows (Assume that the instruction sent by the 8259 was a "CALL" instruction). On interrupt acknowledge cycle, a decoded INTA signal is sent out as an INTA pulse for the 8259 and at the same time, sets the LS74 to indicate that the interrupt acknowledge cycle is started.

On the following memory read cycle of the jump address for the call instruction, this circuit generates two additional INTA pulses for the 8259 and also masks off the read signal for the memory to avoid bus contention problems.

On the following write cycle, /WR signal resets the LS74 to indicate that the interrupt acknowledge cycle is completed.

By using this circuit, you can use the 8259 with the Z180.



## MMU

**Q:** When does the effect take place after changing the contents of MMU related registers?

**A:** From the instruction following the I/O write instruction to the register.

**Q:** What happens if the MMU base register is programmed to exceed 512K (64-pin DIP) or 1M Bytes (68-pin PLCC/80-pin QFP) of physical addressing space?

**A:** 64-pin DIP version -

When "calculated address" is above 7FFFFH, carry bit and MSB are just ignored (A19 and carry from A19). That means the address wraps around to 00000H.

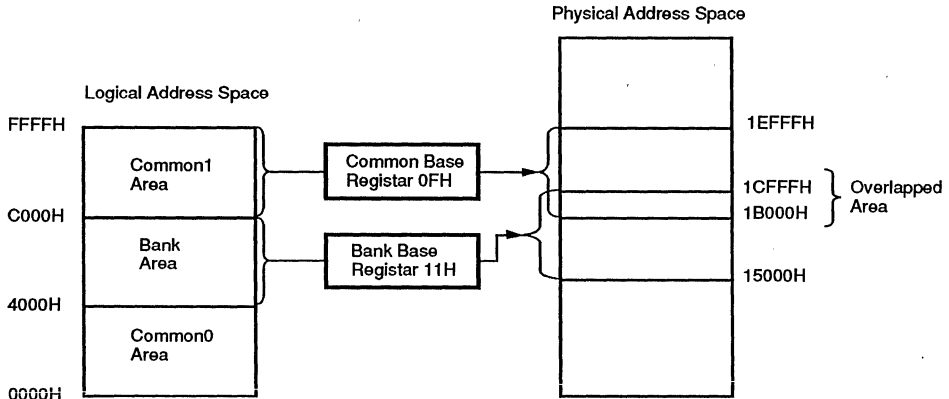
68-pin PLCC/80-pin QFP version -

When "calculated address" is above FFFFFH, carry bit is just ignored (carry from A19). That means the address wraps around to 00000H.

**Q:** Can I have a Common Area 1 and Bank Area (or, Common Area 0 and Bank Area) overlapped by programming associated MMU registers?

**A:** Yes. You can have overlapped areas by programming MMU Common Base/Bank Base registers (See figure on next page).

## MMU (Continued)

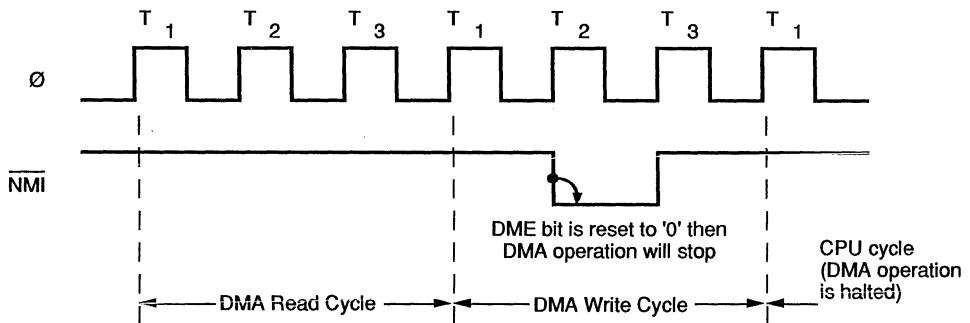


## DMA CHANNELS

**Q: What happens if an NMI occurs during a DMA cycle?**

A: When the DMA cycle is completed, the CPU gains

control. The DME (DMA enable) bit of that DMA channel is reset to Zero.



To restart DMA after the /NMI processing, DE (DE0 or DE1) bit should be set to 1 along with DME bit to 1.

Here is an example to restart DMA0.

```
LD    A, 01100011B ;DE0=1 with DME=1, /DWE=0
OUT0 (30H),A      ;Write out to DSTAT reg.
```

**Q: What is the purpose of the /DWE bit in the DSTAT register?**

A: To set/reset the DE0 or DE1 bit in DSTAT register, you also have to set the corresponding /DWE0 or /DWE1 bit

to 1. This scheme allows changing the status of the DE0 or DE1 bit without affecting the operation of the other DMA channel.

**Q: When DMA0 is used for data transfer from/to on-chip ASCI, does it accept the request from DREQ0 input?**

A: When DMA0 is used for data transfer for an internal peripheral (ASCI Tx or Rx data), external input on /DREQ0 is ignored. When using /DREQ0 input for DMA request from an off-chip peripheral, both bits of SAR17/16 or DAR17/16 should be cleared to zero.

---

**Q: When using DMA0 for ASCII data transfer, what value shall I program into SAR0 (DMA0 Source Address Register) or DAR0 (DMA0 Destination Address Register)?**

A: You have to program SAR0 or DAR0 as follows:

For Receive data transfer:

Set SAR0L (I/O address 20H) to:  
08H for ASCII0 receive data  
09H for ASCII1 receive data

Set SAR0H (I/O address 21H) to 00H

Set SAR0B (I/O address 22H) to:  
01H for ASCII0 receive data  
02H for ASCII1 receive data

For Transmit data transfer:

Set DAR0L (I/O address 23H) to:  
06H for ASCII0 transmit data  
07H for ASCII1 transmit data

Set DAR0H (I/O address 24H) to 00H

Set DAR0B (I/O address 25H) to:  
01H for ASCII0 transmit data  
02H for ASCII1 transmit data

**Q: For the DMA transfer from/to ASCII channels, does it require A15-A8 of SAR or DAR equal to 00H?**

A: Yes. Those bits have to be zero for ASCII data transfer. If the value is not zero, the access by DMA is not going to ASCII registers but will try to access off-chip I/O devices. The result is ASCII will continue to request DMA service (Request status would not be cleared).

**Q: I'm trying to transfer ASCII receive data using on-chip DMA. But after enabling the DMA channel, it won't transfer the data. Why?**

A: Please check whether ASCII has already received data when enabling the DMA channel. If ASCII already has

data (RDRF flag=1), DMA won't start the data transfer. For that case, before enabling DMA channel, read TSR register to reset the RDRF flag.

Remember that you cannot use "Level sense mode" for ASCII data transfer.

**Q: Can I transfer the memory data using DMA and using logical address?**

A: No, you can't. You have to know the physical address for the target memory as well as the Common0/Bank/Common1 address assignments. You may need to calculate the "physical address" using the value in MMU related registers - BBR, CBR and CBAR.

Another solution might be using Z180's "Block transfer instructions". For this case, you don't have to know the physical address for the target memory.

**Q: I want to transfer the data from a memory mapped I/O device (Fixed address) to one of the peripherals assigned to an I/O address. Can I do that?**

A: No. The following source/destination combinations are NOT allowed.

Memory (Fixed address) to Memory  
(Fixed address).

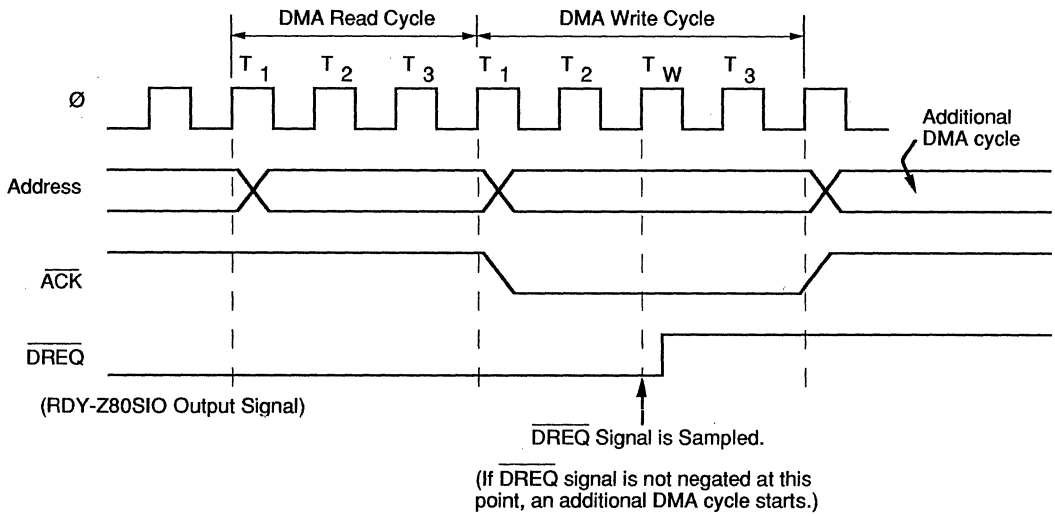
Memory (Fixed address) to I/O (Fixed address).

I/O(Fixed address) to I/O(Fixed address).

With the memory mapped I/O device (Fixed address), you can transfer data from/to Memory (variable address).

**Q: When using on-chip DMA for Z80 SIO Tx data transfers with level sense mode, sometimes Tx data is missed (It has "extra DMA cycle"). Why, and are there any work-arounds for this situation?**

A: The reason for "extra DMA cycles" is the timing when DMA samples the DREQ status and the timing when SIO negates the DREQ signal. (See next page)



Here are possible workarounds.

- 1) Program DMA to edge sense mode.
- 2) Put in one wait state for the I/O write cycle.
- 3) /DREQ signal masked by the Chip enable signal of the Z80 SIO.

**Q: Which DMA channel has higher priority?**

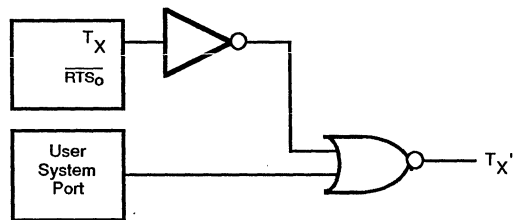
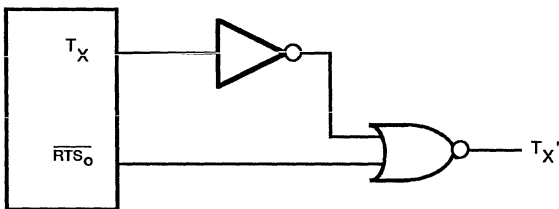
A: DMA Channel 0 has higher priority.

## ASCII CHANNELS

**Q: Are there any ways to send BREAK via software command?**

A: No. ASCII doesn't have any software commands to send

a break. One possible way to send a break is to utilize the unused modem control signal of the Z180, or using an output port bit and connect as follows:



**Q: How to calculate baud rate?**

A: Use the following formula to calculate.

$$\text{Baud rate} = \frac{\text{System Clock Speed}}{(\text{Clock factor}) \times (\text{PS bit}) \times (\text{Divide ratio})}$$

Where :

- Clock factor - 16 or 64
- PS bit - 10 or 30
- SS0-SS2 - 1, 2, 4, 8, 16, 32 or 64

**Q: I could not send data since TDRE always reads zero. Why?**

A: Please check the /CTS line status. TDRE is qualified by the status of the /CTS line. If /CTS stays high, the TDRE bit is zero. Transmit interrupt is generated by the TDRE bit, so with /CTS high, you won't receive transmit interrupt if you enabled the interrupt.

**Q: What happens if /CTS goes inactive at the middle of the data transmission ?**

A: The character being sent out will be sent completely.

**Q: When reading the status of /DCD0 found in STAT0, the status is still one but the real status of the /DCD pin has already changed to zero. How and why is this?**

A: The /DCD0 bit in the status register is reflecting the status of the /DCD0 pin. The low to high change on this pin updates the status of the /DCD0 bit right away. But if the change on the /DCD0 pin is high to low then the bit will remain the same until this bit is read (to change its status). The reason for this scheme is to serve the interrupt for /DCD correctly. If it changes its status at both transitions, and if there is an interrupt from ASCIO before the service for /DCD, the status of /DCD would be lost.

**Q: What is the maximum baud rate for ASCII channels?**

A: When using the internal clock for baud rate generation, it is:

$$BRG_{max} = PHI/(1X10X16)$$

Where: 1 is divider value, 10 is prescaler value, and 16 is sampling rate.

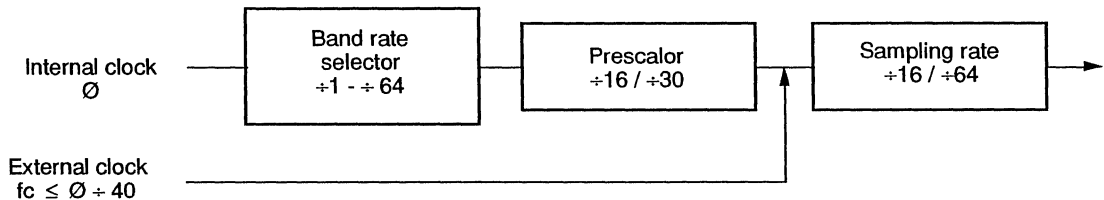
When using the external clock, clock frequency on the external clock input is limited up to  $PHI/40$ , so:

$$BRG_{max} = PHI/(40X16)$$

Where: 16 is sampling rate.

**Q: When using the external clock for baud rate generation, can I use the on-chip clock divider/prescaler?**

A: No. When using external clock input, the on-chip clock divider/prescaler is skipped.



## TIMER

**Q: Does timer (PRT) Channel 0 have a timer output ?**

A: No. If you need to have timer output, you have to use PRT channel 1 and use TOUT.

Note that TOUT pin is multiplexed with A18, so if you want to have TOUT, you need to consider physical memory assignments (For this case, physical memory space is half size).

**Q: Can I use PRT channel(s) to count external events (as counter)?**

A: No. PRT's input is always PHI divided by 20 (Works as timer only).

**Q: I'm using timer for generating PWM pulses. Sometime I cannot get the desired pulse width for a cycle. Why?**

A: When you modify RLDR (Timer Reload Register), make sure the write transactions (write 2 bytes; RLDRH then followed by RLDRH) start just after TDR (Timer Data Register) reaches 0000H, and complete before reaching 0000H again. If reload condition occurs in the middle of these two writes, the Timer loads "Old" RLDRH and "New" RLDRH values.

**Q: What happens to the Timer after resuming IOSTOP mode if IOSTOP mode is entered during operation?**

A: The timer will hold current settings and values, and resume operation from the point when IOSTOP mode was entered.





# ZILOG SCC

## Z8030/Z8530

### QUESTIONS AND ANSWERS

March 1992

*This document contains the most commonly asked questions about the Zilog SCC. They are divided into five sections:*

- Hardware Considerations
- Interrupts and Polling
- Asynchronous Mode
- Synchronous Mode
- Miscellaneous Questions

## HARDWARE CONSIDERATIONS

This section includes questions and answers on the hardware interface, the clocks, the FIFO, special modes (Local Loopback, DPLL, Manchester), and internal timing consideration.

### Hardware (Includes DMA Interface)

**Q. What is the SCC transistor count?**

A. Approximately 6000 gates, or 18,000 transistors.

**Q. What is the difference between the Z8030 and the Z8530?**

A. The Z8030 and Z8530 are packaged from the same die. The multiplexed bus (Z8030) or non-multiplexed bus (Z8530) version of the chip is selected at packaging time by an internal bonding option.

**Q. Can /AS be active only when the Z8030 is being accessed and High all other times?**

A. Since the interrupt pending bits (IPs) are updated on address strobes, interrupts will not occur unless /AS is continuous.

**Q. How do /WR and /CE interact on the Z8530?**

A. /WR and /CE are ANDed to enable a transparent latch. Data is latched on the falling edge when both /CE and /WR go Low.

**Q. How many register pointers does the Z8530 have?**

A. The SCC has only one register pointer for both channels. The SIO (Z844X) has two, one for each channel.

**Q. Do you have to write to the pointer with the Z8530 to access WR0 or RR0?**

A. No. Both registers are accessed automatically without first writing to the pointer.

**Q. Does /CE (/CS) have to be High during an interrupt acknowledge cycle?**

A. No.

**Q. Does the SCC support full duplex DMA?**

A. The SCC allows full duplex DMA transfers by using the DTR/REQ and W/REQ as two separate DMA control lines for transmit request and receive request on each channel.

**Q. When using full duplex DMA, how do you program W/REQ?**

A. W/REQ should be programmed for receive and DTR/REQ pin should be programmed for transmit.

**Q. Can both channels make simultaneous DMA requests?**

A. Yes.

**Q. Do you have to reset the SCC in hardware?**

A. No. A software reset is the same as a hardware reset, (WR9 CO). It also does not matter whether the Z8030 is in shift right or shift left mode because the address is the same in either.



---

## Hardware (Includes DMA Interface)

**Q. Do you need to clear the reset bit in WR0 after a software reset?**

A. The reset is clocked with PCLK; so it must be active during reset.

**Q. How long after a hardware reset should you wait before programming the SCC.**

A. Four PCLKs.

**Q. Why does the SCC initialization require that the External Status Interrupts be reset twice?**

A. Because of the possibility of noise causing an interrupt pending bit (IP) to be set. The second reset guarantees that the latch is clear. If the latch is closed high and the external signal is low, the first reset will open the latch at the high-to-low transition causing an interrupt.

---

## Clocks

**Q. Does PCLK have to have a 50% duty cycle?**

A. The duty cycle doesn't have to be 50% as long as the minimum specification is met.

**Q. Can the SCC PCLK be stretched?**

A. Yes, as long as the pertinent specification is met. However, this could cause a problem if PCLK is used to generate the bit rate.

**Q. The bit rate generator is driven from what sources?**

A. It may be driven from the RTxC pin or PCLK, or from a crystal.

**Q. How do you connect a bit rate crystal to the SCC?**

A. A crystal can be connected between RTxC and SYNC to supply the clock if the SCC is programmed for WR11 D7-1.

**Q. What is the crystal specification?**

A. It is a fundamental, parallel resonant crystal. For further details see the "Design Considerations Using Quartz Crystals with Zilog's Components" Application Note.

**Q. Can RTxC on both channels be driven from the same crystal.**

A. No. A separate crystal should be used for each channel. The crystal should be connected between /SYNC and RTxC of the respective channels. The alternate solution may be to use crystal on one channel and reflect the clock out of the TRxC output and feed it into another channel.

**Q. How do you select a crystal frequency?**

A. Time constant: (Clock Frequency/2 x Bit rate x clock factor) - 2 (The SCC Technical Manual assumed a clock factor of one in the formula). Two examples are given below:

For PCLK = 3.6864 MHz			For PCLK = 3.9936 MHz		
Bit Rate	TC	Error	Bit Rate	TC	Error
38400	46	-	19200	102	-
19200	94	-	9600	206	-
9600	190	-	7200	275	12%
7200	254	-	4800	414	-
4800	382	-	3600	553	.06%
3600	510	-	2400	830	-
2400	766	-	2000	996	.04%
1200	1534	-	1800	1107	.03%
			1200	1662	-
			600	3326	-
			300	6654	-
			150	13310	-
			134.5	14844	.0007%
			110	18151	.0015%
			75	26622	-
			50	39934	-

---

---

**Q. Why are there different Clock factors?**

A. These clock factors enable the SCC to sample the center of the data cell. In the 16x mode, the SCC divides the bit cell into 16 counts and samples on count 8. Clock factors are generally only used with Asynchronous modes.

**Q. How is the error in the receive/transmit clock reduced?**

A. The ideal way to reduce this error is by adjusting the crystal frequency such that only an integer value of TC is yielded when the equation is used.

**Q. What are the maximum transfer rates?**

A. The following table shows the PCLK rates (in bps).

---

	4 MHz	6 MHz	8 MHz	10 MHz	16 MHz	20 MHz
<b>Asynchronous mode:</b>						
External clock						
6x mode (no BRG)	250K	375K	500K	635K	1M	1.25M
BRG						
16x mode (TX + 0)	62.5K	93.75K	125K	156.5K	250K	312.5K
<b>Synchronous mode:</b>						
Using external clock	1M	1.5M	2M	2.5M	4M	5M
Using DPLL, FM encoding	250K	375K	500K	625K	1M	1.25M
Using DPLL, MRZ/NRZI encoding	125K	187.5K	250K	312.5K	500K	625K
Using DPLL, FM, BRG	62.5K	93.75K	125K	156.25K	250K	312.5K
Using DPLL, NRZ/NRZI, BRG	32.25K	46.88K	62.5K	78.125K	125K	156.25K

---

**Q. Can the maximum transfer rate using an external clock be achieved?**

A. Yes, but it is not trivial. In order to achieve the maximum rate on transmit, the SCC should have a dedicated processor or DMA. For example, at a 1 MHz rate, a byte must be loaded into the SCC every 8

microseconds. To achieve the maximum rate on receive, requires that the receive clock and the SCC PCLK be synchronized. (RTxC to PCLK setup time at maximum rate in the Product Specification.) It is probably easier to use a slightly faster PCLK SCC, or back off slightly from the maximum rate.

---

**FIFO****Q. How do you avoid an overrun in the received FIFO?**

A. The receive buffer must be read before the recently received data character on the serial input is shifted into the receive data FIFO. This FIFO is three bytes deep. Thus, if the buffer is not read, the fifth character just arrived causes an overrun condition. There is no bit that can be set or reset to disable the buffering.

**Q. What happens when you read an empty FIFO?**

A. You read the last character in the buffer.

**Q. When the FIFO gets locked due to an error condition, can it still receive?**

A. The SCC continues to receive until an overrun occurs.

**Q. Assuming that there are characters available in the FIFO, what happens to them if the receiver goes into the hunt mode?**

A. They will remain in the FIFO until they are either read by the CPU or DMA, or until the channel is reset.

---

## Special Modes (Local, Loopback, DPLL, Manchester)

**Q. How are the Local, Loopback, and Auto Echo modes implemented?**

- A. The TxD and RxD pins are connected through drivers. If both modes are simultaneously enabled, then Auto Echo overrides.

**Q. Can the SCC transmit when the Auto Echo mode is enabled?**

- A. No, the transmitter is logically disconnected from the TxD pin.

**Q. Can the Digital Phase Lock Loop (DPLL) be used with NRZ?**

- A. The DPLL simply generates the receive clock which is the same for both NRZ and NRZI.

**Q. Do you have to use the DPLL with NRZI and FM encoding?**

- A. If the DPLL is not used, a properly phased external clock must be supplied.

**Q. What is the error tolerance for the DPLL?**

- A. The DPLL can only tolerate a + or - 1/32 deviation in frequency, or about 3%.

**Q. Can you receive and transmit between two channels on the same SCC using the DPLL to generate both the transmit and receive clocks?**

- A. To transmit and receive using the same clock, you need to divide the transmit clock by 16 or 32 to be the same rate for transmitting and receiving, because the DPLL requires a divide-by-16 or -32 on the receiver, depending on the encoding. An external divide-by-16 or -32 is required, and can be connected by outpouring the bit rate generator on the /TRxC pin, through the external divide circuit, and back in the /RTxC pin as an input to the transmitter.

**Q. How fast will Manchester be decoded?**

- A. The SCC can decode Manchester data by using the DPLL in the FM mode and programming the receiver for NRZ data. Hence, the 125K bit/s is the maximum rate for decoding at 8MHz SCC. A circuit for encoding Manchester is available from Zilog.

**Q. When will the Time Constant be loaded into the BRG counter?**

- A. After a S/W reset or a Zero Count is reached.

**Q. How to run NRZ data using the DPLL?**

- A. Use NRZI for DPLL (WR14) but set to NRZ (WR10).

---

## INTERNAL TIMING

**Q. When does data transfer from the transmit buffer to the shift register?**

- A. About 3 PCLK's after the last bit is shifted out.

**Q. How long does it take for a write operation to get to the transmit buffer?**

- A. It takes about 5 PCLK's for the data to get to the buffer.

**Q. What is Valid Access Recovery Time?**

- A. Since WR/ and RD/ (AS/ and DS/ on the Z8030) have no phase relationship with PCLK, the circuitry generating these internal control signals must provide time for metastable conditions to disappear. This gives rise to a recovery time related to PCLK.

**Q. How long is Valid Access Recovery Time?**

- A. On the current device from Zilog introduced in 1986, the recovery time is 4 PCLK's. Earlier SCC's required 6 PCLK's plus some additional nanoseconds.

**Q. Why does the Z8030 require that the PCLK be "at least 90% of the CPU clock frequency for Z8000?"**

- A. If the clocks are within 90%, then the setup and hold times will be met. Otherwise, the setup and hold times must be met by the user.

**Q. Does Valid Access Recovery Time apply to all successive accesses to the SCC?**

- A. Any access to the SCC requires that the recovery time be observed before a new access. This includes reading several bytes from the receive FIFO, accessing separate bytes on two different channels, etc. When using DMA or block transfer methods, the recovery time must be considered.

---

**Q. Do the DMA request and wait lines on the SCC take the Valid Access Recovery time into account before they make a request?**

A. No, they are not that intelligent. The user must take this into account, and program the DMA accordingly. For example, be inserting wait states during the memory access between SCC accesses, which will lengthen the time in between SCC accesses, or by requiring the DMA to release the bus between accesses to the SCC, to prevent simultaneous data requests from two channels from violating the recovery time.

**Q. What happens if Valid Access Recovery Time is violated?**

A. Invalid data can result.

**Q. Does Valid Access Recovery Time affect the interrupt acknowledge cycle?**

A. No. The interrupt vector is put on the bus by the SCC during the interrupt acknowledge cycle, but does not require any recovery time.

**Q. Why can some systems violate the recovery time by 1 or 2 PCLK's without affecting the data to the SCC?**

A. This violation may or may not matter to the SCC. This phase relationship between PCLK, /RD, /WR, (/AS, /DS for Z8030) can be ASYNC. The SCC requires some time internally to synchronize these signals. The electrical specs for the SCC indicate a recovery time, which is the worst case maximum.

---

## INTERRUPT CONSIDERATIONS

**Q. What conditions must exist for the SCC to generate an interrupt request?**

A. Interrupts must be enabled (MIE = 1 and IE = 1). The Interrupt Enable Input (IEI) must be high. The interrupt pending bit (IP) must be set and its interrupt under service bit (IUS) must be reset. No interrupt acknowledge cycle may be active.

**Q. How can the /INTACK signal be synchronized with PCLK?**

A. /INTACK needs to be synchronized with PCLK. This can be accomplished by changing /INTACK only on the falling edge of PCLK by using a D flip-flop that is clocked with the inverted PCLK.

**Q. Is /CE required during an Interrupt Acknowledge cycle?**

A. No.

**Q. How long does /INT stay active low when requesting an interrupt?**

A. If the SCC is operated in a polled mode, the /INT will remain active until the IP bit is reset. For an interrupt acknowledge cycle, the /INT will go inactive shortly after the falling edge of /RD or /DS when the IUS bit is set.

**Q. Can you use the SCC without a hardware interrupt acknowledge?**

A. Yes. If you are not using the hardware daisy chain, you don't need to give an interrupt acknowledge. Tie the intack pin high, enable interrupts, and on responding to an interrupt, check RR3 for the cause, and special receive conditions if you are in receive mode. The internal daisy-chain settling time must still be met. (IEI to IEO delay time specification.)

**Q. How do you acknowledge an interrupt without a hardware interrupt acknowledge?**

A. Reset the responsible interrupt pending bit (IP). The /INT line follows the IP bit.

**Q. When are the IP bits cleared?**

A. A transmitter empty IP is cleared by writing to the data register. A receive character available IP is cleared by reading the data register. The external/status interrupt IP is cleared by the command Reset Ext/Status Interrupts.

**Q. Can the IP bits be set while the SCC is servicing other interrupts?**

A. Yes. If the interrupting condition has a higher priority than the interrupt currently being serviced, it causes another interrupt, thus nesting the interrupt services.

**Q. Can the IUS bits be accessed?**

A. No. They are not accessible.

**Q. When do IUS bits get set?**

A. The IUS bits are set during an interrupt acknowledge cycle on the falling edge of /RD or /DS.

**Q. How do you reset interrupts on the SCC?**

A. The interrupt under service bit (IUS) can be reset by the command "Reset Highest IUS" or 38 Hex to WR0. Reset Highest IUS should be the last command issued in the interrupt service routine.

**Q. Why is the interrupt daisy chain settle time required?**

A. This mechanism allows the peripheral with the highest priority interrupt pending in the hardware interrupt daisy chain to have its interrupt serviced.

---

## INTERRUPT CONSIDERATIONS (Continued)

**Q. Is there still a settle time if the peripherals are not chained?**

- A. Even if only one SCC is used, there still is a minimum daisy-chain settle time due to the internal chain.

**Q. How should the vectors be read when utilizing the /INTACK?**

- A. /INTACK should be tied to 5 volts through a register. Erroneous reads can result from a floating INTACK. The interrupt vectors can be read after an interrupt from RR2.

**Q. How is the vector register different from the other registers?**

- A. The vector register is shared between both channels. The Write register can be accessed from either channel. Reading "Read Register 2" on Channel A (RR2A) returns the unmodified vector, and RR2B returns the modified vector that includes status. The vector includes the status bit (VIS, WR9) and determines which vector register is put out on the bus during an interrupt cycle.

**Q. How do you poll the external/status IP?**

- A. Set the IE bits in WR15 so the conditions are latched and set ext/status master interrupt enable bit in WR1. To guarantee the current status, the processor should issue a Reset External/Status interrupts command in WR0 to open the latches before reading the register. For further details see the SCC Technical Manual, section 3.4.7.

**Q. When should the status in RR1 be checked?**

- A. Always read RR1 before reading the data.

**Q. What conditions cause the transmit IP to be set?**

- A. Either the buffer is empty, or the flag after CRC is being loaded.

**Q. How do you tell if you have a Zero Count (ZC) interrupt?**

- A. This bit is not latched like the other external IP bits. If an external interrupt occurs and none of the other IP bits have changed since the last ext/status interrupt, then the ZC condition caused it. A ZC interrupt will not be generated if there are other ext/status (IP) pending. The ZC stays active for each time only when the count reached zero, approximately two PCLK time periods.

**Q. How do you poll the bits in RR3A?**

- A. Enable interrupts in WR1 and disable MIE before polling.

**Q. What happens when the SCC is programmed to interrupt on transmit buffer empty and also to request DMA activity on transmit buffer empty?**

- A. This would not be a wise thing to do. The interrupt would occur but the DMA could gain control of the bus and remove the interrupting condition before the interrupt acknowledge could take place. When the CPU recovers control of the bus and starts the interrupt acknowledge cycle, bus confusion results because the peripheral no longer has a reason to interrupt.

**Q. Will IP bit (s) for external status be cleared by the Reset Ext/Status interrupt?**

- A. Yes.

---

## ASYNCHRONOUS MODE

**Q. Can the Sync Character Load Inhibit function strip characters in Asynchronous mode if not disabled?**

A. Yes. If not disabled it will strip any characters which match the value in the sync character register. Always disable this function in asynchronous mode (WR3, bit D1).

**Q. What controls the DTR/WREQ pin?**

A. The DTR pin follows the D7 bit in WR5 (inverse) as a Data Terminal Ready pin, or it is a DMA request line (WREQ). The bit can be set or reset by writing to WR5.

**Q. How is the Asynchronous mode selected?**

A. The Asyn mode is selected by programming the number of stop bits in write register 4.

**Q. How are receiver breaks handled?**

A. The SCC should monitor the break condition and wait for it to terminate. When the break condition stops, the single NULL character in the receive buffer should be read and discarded.

**Q. Where can you get the DTR input if the DTR/REQ pin is being used for DMA?**

A. The SYNC can be used as an input if operating in the Async mode. It will cause an interrupt on both transitions.

**Q. When a special condition occurs due to a parity error, will a receive interrupt for that byte still be generated?**

A. No. In the case of Receive interrupt on Special Condition Only mode, the interrupt will not occur until after the character with the special condition is read. In the case of Receive Interrupt on All Characters or Special Condition Only mode, the interrupt is generated on every character whether or not it has a special condition.

**Q. In the Auto Enable mode, what happens when CTS/ goes inactive (high) in the middle of transferring a byte?**

A. If the Auto Enable mode is selected, the CTS/ pin is an enable for the transmitter. So, when CTS/ is inactive, transmit stops immediately.

**Q. Can X1 clock mode really be used for the Async operation?**

A. X1 mode cannot be used unless the receive and transmit clocks are synchronized. Using a synchronous modem is one way of satisfying this requirement.

**Q. When does the FIFO buffer lock on an error condition?**

A. The receive data FIFO gets locked only in cases where the following receiver interrupt modes are selected:

- Receive Interrupt on Special Condition only
- Receive Interrupt on First Character or Special Condition

In both of these modes, the Special Condition interrupt occurs after the character with the special condition has been read. The error status has to be valid when read in the service routine. The Special Condition locks the FIFO and guarantees that the DMA will not transfer any characters until the Special Condition has been serviced.

## SYNCHRONOUS MODES

(SDLC, HDLC, Bysync, And Monosync Modes Included)

**Q. For what are the cyclical redundancy check (CRC) residue codes used?**

- A. The residue codes provide a secondary method to check the reception of the message.

**Q. Why is the second byte of the CRC incorrect when read from the receiving SCC?**

- A. The second byte of the CRC actually consists of the last two bits of the first byte or CRC, and the first six bits of the second byte of CRC. For more details there is an Application Note available from Zilog called "SDLC Residue Codes for the Z80 SIO" which applies to the SCC as well.

**Q. How does the SCC send CRC?**

- A. The SCC can be programmed to automatically send the CRC. First, write the first byte of the message to be sent. This guarantees the transmitter is full. Then reset the Transmit Underrun/EOM latch (WRO 10). Write the rest of the data frame. When the transmit buffer underruns, the CRC is sent. The following table describes the action taken by the SCC for the bit-oriented protocols:

Tx Underrun EOM Latch Bit	Abort/Flag Bit	Action Upon Tx Underrun	Comment
0	0	Sends CRC + Flags	Valid Frame
0	0	Sends Abort + Flags	Aborted Frame
1	X	Sends Flags	Software CRC

The SCC sets the Tx Underrun/EOM latch when the CRC or Abort is loaded into the shift register for transmission. This event causes an interrupt (if enabled).

**Q. In SDLC, when do you reset the CRC generator and checker?**

- A. The Reset TxCRC Generator command should be issued when the transmitter is enabled and idling (WRO). This needs to be done only once at initialization time for SDLC mode.

**Q. How can you make sure that a flag is transmitted after CRC?**

- A. Use the external status end of message (EOM) interrupt to start the CRC transmission, then enable the transmit buffer empty interrupt. When you get the interrupt, it means that the buffer is empty, a flag is loaded in the shift register, and you can send the next packet of information.

**Q. If the SCC is idling flags, and a byte of data is loaded into the transmit buffer, what will be transmitted?**

- A. Data takes priority over flags and will be loaded in the shift register and transmitted.

**Q. Since data is preferred, can this cause a problem?**

- A. This allows you to append on the end of a message, but it can cause problems with DMA. A character could be transmitted without an opening flag. To make sure that a flag has been transmitted, watch for the W/REQ line to toggle when the flag is loaded into the shift register.

**Q. Can you gate data by stretching the receive clock?**

- A. You can hold the clock until you have valid data. There are no maximum specs on the RxC period, and the edges are used to sample the data. If there are no edges, no data is sampled.

**Q. How do you synchronize the DPLL in SDLC mode?**

- A. There are two methods to synchronize the DPLL. Supply at least 16 transitions at the beginning of each message so the DPLL has time to make adjustments, or use the DPLL search mode in WR14 to cause the SCC to synchronize on first transition. The first edge must be guaranteed to be a cell boundary.

**Q. In SDLC, is the flag and address stripped-off?**

- A. No, only the flag is stripped. The address will be the 1st character received.

**Q. Does IBM® SDLC specify parity?**

- A. No.

**Q. Can the SCC include parity in SDLC mode?**

- A. Yes. It is appended at the end of the character.

**Q. How does the SCC operate in transparent mode?**

- A. The transparentness, as defined by IBM SNA, should be provided by the software. The SCC does not perform any automatic insertion and deletion of link control nor does it automatically exclude the characters from the CRC calculation. This also applies to other high level protocols.

**Q. When does the Abort function take effect?**

- A. The abort takes place immediately by inserting eight consecutive 1's.

---

**Q. Can the SCC detect multiple aborts?**

A. The SCC searched for seven consecutive 1's on the receive data line for the abort detection. This condition may be allowed to cause an external status interrupt. After these seven 1's are received, the receiver automatically enters Hunt mode, where it looks for flags. So, even if more than seven 1's are received in case of multiple aborts, only the first sequence of 1's is significant.

**Q. How do you send an end of poll (EOP) flag in SDLC loop mode?**

A. To send the EOP message, simply toggle the bit which idles flags or ones to mark flags, then mark ones. This produces a zero and more than seven 1's; an EOP condition.

**Q. When the SCC is programmed for 6 bit sync, how are bits sent?**

A. Six bits are sent. The 12-bit sync character sends 12 bits.

**Q. Do sync patterns (or flags) in data transmissions get stripped and still cause interrupts?**

A. All leading sync patterns (and all flags) are automatically stripped if the Sync Character Load Inhibit feature is programmed. Any data stripped from the transmission stream cannot cause a receive character available interrupt but may cause other interrupts (such as External/Status for Sync/Hunt and special receive condition for EOM).

**Q. How are the sync characters sent at the beginning of a Bisync frame?**

A. Load the transmit buffer with the first byte and the sync characters are automatically sent out.

**Q. How can you determine when the flag has been completely sent?**

A. There are several ways to determine if the flag has been completely sent. This allows the transmitter to be shut off, or in half duplex the line can be turned around. This requires a little work by the user because the SCC does not know when the last flag bit has been shifted out. The following are some suggestions:

- Once the flag is loaded into the transmit shift register, start an external clock. Use the baud rate generator as the counter.

- Tie the transmit line into DCD or an available input pin, and watch for a zero, or end of flag. If you are running half-duplex, use the local loopback mode and watch for the flag to end.

- Allow an abort, although this destroys the last character. Be sure to send a dummy character - then idle flags after the abort latch is set.

**Q. How do the DMA W/REQ lines operate?**

A. DMA request lines follow the state of the transmit buffer.

**Q. How does the SCC handle messages less than four bytes in length?**

A. A 4-byte message consists of an address, control word, no data, and 2 bytes of CRC. SDLC defines messages of less than 4-bytes as an error. It is not defined how the SCC will react, however, as tested by a SCC user, 4-, 3-, and 2-byte messages cause an interrupt on end of frame, but a 1 byte message does not cause an interrupt.



---

## MISCELLANEOUS QUESTIONS

**Q. Can the SCC support MARK and SPACE parity in async?**

- A. The SCC can transmit-end the equivalent of MARK parity by setting WR4 to select two STOP bits. The receiver always checks for only one STOP bit; therefore, the receiver does not verify the MARK parity bit.

The SCC (and products using the SCC cell) does not support SPAC parity for transmitting or receiving. The Zilog USC Family of serial datacom controllers do support odd, even, mark, & space parity types.

**Q. Since both D7 and D1 bits in RRO are not latched, it is possible that the receiver detected an Abort condition, set D7 to 1, initiated an external/status interrupt and before the processor entered the service routine, termination of the abort was detected, which reset the Break Abort bit. Currently in the TM (page 7-20), the description for Bit1: Zero Count states if the interrupt service routine does not see any changes in the External/Status conditions, it should assume that a zero count transition occurred when in fact, an Abort condition occurred and was missed. What could be done to correct this and not miss the fact that an Abort occurred?**

- A. Very few people actually use the Zero Count interrupt. This interrupt is generated TWICE during each bit time and is usually used to count a specific number of bits that are sent or received. If this interrupt is not used by your customer, then what is said in the TM about the Zero Count is true for the Abort Condition. If no other changes occurred in the external/status conditions and the Zero Count is not used, then the source of the interrupt was the Abort condition.

**Q. Can the SCC resynchronize independent clocks (at the same frequency, but could be out of phase), one for Rx data and one for Tx data?**

- A. No, the two clocks are independent of each other. However, the SCC provides a special transmitter-to-receiver synchronization function that may be used to guarantee that the character boundaries for the received and transmitted data are the same. This function can be found in the SCC Technical Manual (Q3/90 Section 5.2.3).

**Q. When is EOM and EOF asserted?**

- A. EOM is asserted when it detects depletion of data in the Tx buffer; EOF is asserted when it detects a closing flag.

**Q. After powering up the SCC, are the reset values in the write and read registers guaranteed?**

- A. No. You must perform a hardware or software reset. A list of the hardware and software reset values are listed in the SCC Technical Manual (Q3/90) on page 3-9.

**Q. Can you read the status of a write register, such as the MIE bit in WR9?**

- A. No, in order to retain the status of a write register, you must keep its status in a separate memory for later use. However, the only exception is that WR15 is a mirror image of RR15. Also, the ESCC has a new feature to allow the user to read some of the write registers (see the ESCC Product Specification or Technical Manual for more details).

**Q. Is there a signal to indicate that a closing SDLC flag is completely shifted out of the Tx pin? This is needed to indicate that the frame is completely free of the output to allow carrier cut off without disrupting the CRC or closing flag.**

- A. No, the only way to find this timing is to count the number of clocks from Tx Underrun Interrupt to the closing flag. The ESCC contains the feature by deasserting the /RTS pin after the closing flag. Upgrade to the ESCC!

**Q. Does the SCC detect a loss of the receive clock signal?**

- A. No, if the clock stops, the SCC senses that the bit time is very long. Use a watch-dog timer to detect a loss in the receive clock signal.

**Q. Is there any harm in grounding the "NO CONNECT" (NC) pins in the PLCC package (pin #17, 18, 28, 36)?**

- A. These NC pins are not physically connected inside the die. Therefore, it is safe to tie them to ground.

**Q. Can the SCC be used as a shift register in one of the synchronous modes with only data sent to the Tx register with no CRC and no sync characters?**

- A. CRC is optional in Mono-, Bi-, and External Sync Modes only. The sync characters can be stripped out via software.



---

## ZILOG ESCC™ CONTROLLER

### Questions and Answers

---

March 1992

#### PRODUCT DESCRIPTION

**Q. Which of the following is the major factor in differentiating the ESCC from the USC Family?**

- (a) The ESCC has less communications channels than the USC
- (b) The protocols supported by ESCC and USC are different
- (c) The ESCC is limited in operation to less than 5 Mbps, but the USC Family can operate up to 10 Mbps

(d) The USC supports the T1 data rate, not the ESCC

- A. (c) Most ESCC and USC Family members have two channels and protocols. Support by the SCC is a subset of ESCC. Both ESCC and USC can support T1 data rates so (a), (b), (d) are not correct.

---

**Q. Which of the following is not an improvement from the SCC to the ESCC?**

- (a) The ESCC has deeper FIFOs
  - (b) The ESCC has new SDLC enhancements
  - (c) The ESCC has added new READ Registers
  - (d) The ESCC has added new WRITE Registers
- A. (c) No new READ register addressing is added in the ESCC although we allowed some Write Registers to become readable through the existing READ Register.

The ESCC has 4 bytes of Tx FIFO and 8 bytes of Rx FIFO, while the SCC has 1 byte for the Tx and 3 bytes for the Rx.

The ESCC has many new SDLC enhancements, such as automatic EOM reset, automatic opening flag generation, etc.

The ESCC has added WR7' as a new WRITE Register to configure the new options, therefore, (a), (b), (d) are all differences between the SCC and ESCC.

---

## APPLICATIONS

**Q. Which of the following is a benefit from deeper FIFOs offered by the ESCC?**

- (a) More CPU bandwidths available for other system tasks
- (b) Can support faster data rates on each channel
- (c) Can support more channels for the same CPU
- (d) All of the above

A. (d) (a), (b) and (c) are consequences of reduction in interrupt frequency that allows more horsepower to be delivered from the CPU.

**Q. Which of the following CRC polynomials is supported in ESCC?**

- (a) CRC-16
- (b) CRC-32
- (c) CRC-CCITT
- (d) (a) and (c)
- (e) (b) and (c)

A. (d) CRC-32 is not supported in ESCC.

**Q. How long does it usually take for the customer to migrate from SCC to ESCC in order to take the advantage of the FIFO?**

- (a) Less than 3 month
- (b) About 6 month
- (c) About a year

A. (a) Since the ESCC is a drop-in replacement to the SCC and using the deeper FIFO only requires minimal efforts.

**Q. Which of the following is an applications support the tool for ESCC:**

- (a) Sealevel Board
- (b) Electronic Programmers Manual
- (c) Application Note "Boost Your System Performance Using the Zilog ESCC"
- (d) All of the above

A. (d)

**Q. Which of the following is a target application for the ESCC?**

- (a) AppleTalk-LocalTalk Peripherals
- (b) X.25 Packet Switches
- (c) SNA connectivity products
- (d) All of the above

A. (d) ESCC could support the data rate and protocol required in the above applications.



## ZILOG ISCC™ CONTROLLER Questions and Answers

March 1992

### ISCC QUESTIONS AND ANSWERS

- Q. Is the interrupt vector present on both the lower 8 bits and the upper 8 bits in an interrupt cycle (See Figure 40 of the Z16C35 CMOS ISCC Product Spec)?**
- A. Both halves of the AD bus are driven during an interrupt acknowledge cycle by the ISCC. In fact, both halves of the data bus are never driven individually.
- Q. In DMA mode, must the /WAIT//READY and /BUSACK signals be externally synchronized to PCLK (See Figure 46 of the Z16C35 CMOS ISCC Product Spec)?**
- A. No, not exactly. The documented timing shows when the ISCC samples these coming back from memory.
- Q. Can the address and data bus be outputted before /BUSACK is received (See Figure 46 of Z16C35 CMOS ISCC Product Spec)?**
- A. No.
- Q. What causes the Terminal Count to be Reset?**
- A. Refer to P.5-26 TM, Sec. 5.6.2, "the status in this register is automatically cleared after a Read." In other words the bits are Reset when you Read the contents of the register.
- Q. Which Rev of the SCC is in the ISCC?**
- A. It is the D Rev (but without the oscillator fix).
- Q. Does the ISCC allow software interrupt acknowledge (WR9 bit D5)?**
- A. Yes, it does. It is not required to use the /INTACK signal of the ISCC to process interrupts. The source of the interrupt can be determined by reading the interrupt vector just like a normal interrupt is determined by reading the interrupt vector (like a normal register Read). The SCC RR2B is modified to reflect the source. RR2A is not modified. Also, the other status registers could be used to figure out who interrupted. SCC interrupts can be Reset by reading RR2B if software interrupt acknowledge is enabled (WR9 D5=1).
- Q. Does the software interrupt acknowledge support DMA operation?**
- A. No. Unlike the SCC core, the DMA core does not support this feature. The DMA has two sources of the interrupt, i.e., IP and IUS bits.
- Q. When the ISCC is used on a multiplexed bus, the ISCC does not interrupt when the SCC source interrupts occur until after another Write to the ISCC. Why?**
- A. When programmed for multiplexed bus operation, similar to the Z8030/Z80C30, the /AS signal is used to update the interrupt status of the SCC. Consequently, if no /AS is present, the interrupt status is not updated until an /AS occurs. If /AS of the ISCC is tied to the /AS of the processor, sufficient /AS signals will occur to keep the ISCC interrupts up to date. However, if /AS is only generated to the ISCC when it is being accessed, any pending SCC interrupts will not assert the /INT pin until after the /AS of an access to the part. This typically occurs when a PAL is used to generate the access signals to the ISCC and /AS is only generated to the ISCC when it is being accessed.
- Q. Can the Upper Address Strobe be defeated (to shorten the transfer cycle time)?**
- A. No. But this is possible in the IUSC!
- Q. How many clock cycles does it take to do a DMA transfer, after BUSACK is granted?**
- A. By looking at Figure 45 of Z16C35 CMOS ISCC Product Spec, it takes TS0, TS1, T0, T1, T2, T3, T4, T5, about 8 cycles total.

---

**Q: Is there any reason why the ISCC couldn't use *pclk* twice as fast as the processor, in order to cut access recovery times?**

A: No, as long as the required timings are met!

**Q: What's the recovery time required for the ISCC?**

A: A recovery time may apply to ANY access of the ISCC. Thus, a bus transaction before or after an access of the ISCC looks like it requires that the recovery time be met for those accesses. The timing for /Strobe signals, i.e. /DS, /WR, /RD or Pulsed /INTACK relative to CLK is three clocks if /Strobe, synched to the /INTACK relative to CLK, is three clocks if /Strobe is synched to the rising edge of CLK; or four clocks otherwise. The Recovery time is independent of /CS. Please note, if in any design application with the ISCC the reads and writes are unreliable, this recovery timing should be checked very carefully and as this could be a bug with the ISCC.

**Q: Is the SDLC FIFO available in ISCC?**

A: Yes, the SDLC FIFO is available in the SCC cell of the ISCC. There is a mistake in our ISCC Technical Manual, P.5-20, on Register Description. The statement 'Bit 2 is not used and must be programmed "0"' is wrong. Bit 2 of WR15 is used for enabling the SDLC FIFO.

**Q: Will DMA be enabled by writing the Enable Command in the Channel Command/Address Register?**

A: Yes, DMA operation is triggered by the command, 'Enable DMA' on Channel Command/Address Register. This is another mistake in our ISCC Technical Manual, P.5-25, on Register Description. The statement 'DMA operation is not triggered by this command' is wrong, e.g., Writing "100" to bits 7 through 5 enables and triggers TxB DMA operation.

**Q: Will DMA operation be triggered by the DMA enable command in the DMA Enable Register?**

A: Yes, DMA operation will also be triggered by setting corresponding DMA Enable bits in the DMA Enable Register (P.5-29 sec 5.6.7, DMA Enable Register in ISCC Technical Manual). Note that this is a read/write register. Read-modify-write instructions should be used in writing this register to avoid the register value to be overwritten and cause accidental enabling/disabling of the DMA operations.

## **ADDITIONAL INFORMATION**

---

### Using Traditional 8- and 16-Bit Architectures in Embedded Designs

Jim Magill

*Director, Intelligent Peripheral Controllers*

---

#### Building on the Z80

Regardless of age, the Z80, with its powerful instruction set, fast execution time and huge installed software base has remained viable even in today's 32-bit world. It has also been said "the only place 8-bit architectures are dead is in the press." But this is certainly not true in the customer base. The Z80 is by far the most popular microprocessor of all time being used in everything from toys to military products.

The Z80 and 8080/85 (which runs the Z80 instruction set) currently comprise 62% of all 8-bit microprocessor sales. The Z80 is object code compatible with the 8080/85 but not mnemonic compatible. Zilog added enhancements to Intel's 8080 architecture which made the Z80 become accepted by the industry as the standard against which all other 8-bit microprocessors are judged. This is due to its high performance, clean general purpose architecture, ease of programming, and simple hardware design.

Other semiconductor suppliers tried to gain back some of the market share with products like the 8085 from Intel and NS800 from National but to no avail. In order to support the power and market acceptance of the Z80 architecture Zilog developed (along with second sources) a variety of high performance peripherals to allow the designer to implement complex functions in a minimum number of chips in a short time. Figure 1 outlines the original Z80 family of peripheral devices.

In 1986/7 the Z80 family was converted to CMOS, which brought all the traditional advantages of CMOS like lower power consumption, higher noise immunity, and faster operation. The Z80 CPU was originally introduced at 2MHz, today the complete family is available in 10MHz while the CPU is released at 20MHz; a 10 time performance increase using the same code. A 20MHz Z80 CPU can be equated to a 5MIPS peak machine. More importantly, Zilog has converted them into megacells which allows them to be easily combined using Zilog's Superintegration technology. Some of the advantages of Superintegration are outlined in Figure 2. Zilog has introduced a wide range of Superintegration™ products, all of which are code compatible with the original family of devices. This is very important, since over 70% of project development dollars spent are on software as opposed to hardware.

There is a major trend today away from central processing, where a main CPU handles all of the control and computation for a system, and towards distributed processing where the intelligence is placed in the peripheral function. This is often called "Intelligent Peripheral Control" or "Embedded Control". This has the advantage of off loading the central CPU of the control tasks and increasing the overall system performance. When used as an embedded controller in a peripheral function, cost is always an issue because generally there are more peripherals in a system than central processors. Further, the CPU in the intelligent peripheral does not need to be as powerful as the central processor in terms of number crunching capability, because other aspects become more important like interrupt handling and interrupt response time. For these and many other reasons the Z80 has become the CPU of choice in embedded controllers. Other trends include the addition of a memory management unit (MMU) to increase the address space, and other common peripheral functions like DMAs, UARTs, Timers, etc.

#### The Upgrade Path

The nature of today's dynamic marketplace means products undergo continuous redesign during their life-cycle. New features are often added to respond to competitive pressures. Although it is impossible to design a product that can be expanded ad infinitum, the design should be open ended without starting over from scratch.

Software is almost always the most expensive and hence the most valuable part of the project's design. For this reason the engineer must ensure that the processor is not operating at peak capacity. Being forced to change the CPU in mid-development due to software bottlenecks will drive engineering costs out of sight. The use of high level languages like C somewhat reduces the cost of switching CPUs. However, in real time embedded systems, where a lot of the software is programmed in assembly language (processor dependent code), the cost of making a change is significant. Only three major processor families give the designer a wide range of performance with upgrade compatible instructions. Motorola's 68000/68020/68030 series



spans the range from 16- to 32-bits. Intels 8088/80286/80386 family is similar although its use in the PC line gives it a much larger software base. Finally, the Z80/Z180/Z280 family gives the user a choice of compatible CPUs with 8- and 16-bit bus versions.

Apart from the selection of the processor, other issues are important when considering the total cost of the design. These are peripheral availability, time to market, risk, power consumption and the cost to do the development, etc. Other questions are what is the cost of the development system and will the development system be so awkward that the engineers spend all their time chasing simple problems?

---

<b>Z8400</b>	CPU	Z80 CPU
<b>Z8410</b>	DMA	1 Channel DMA Controller
<b>Z8420</b>	PIO	2x8-Bit Ports
<b>Z8430</b>	CTC	4x8-Bit Counter Timers
<b>Z844X</b>	SIO	2 Sync/Async Channels

---

**Figure 1. Z80 Family of Discrete Devices**

---

*Reduced Real Estate*  
*Reduced Power Consumption*  
*Higher Speed*  
*Migrate Software from Discrete Devices*  
*Higher Reliability (reduced pin count/components)*  
*Short Design-In Cycle*  
*Reduced Assembly Costs*  
*Reduced Inventory Carrying Costs*  
*Reduced Weight*  
*Higher Noise Immunity*

---

**Figure 2. Advantages of Superintegration**

### 8-, 16- or 32-Bits?

Given a sufficiently high clock rate, the perfect processor is 8-bits. An 8-bit processor exactly matches the bus bandwidth of the vast majority of memories and peripherals. An 8-bit bus uses memory efficiently by avoiding the wasteful handling of byte data in 16-bit words. Simple circuitry, with fewer components, are

used. However, clock rates are limited by a number of factors, so, 16- and 32-bit buses are used to move data in parallel. A wider instruction word, handled in a single machine cycle, means that more complex operations are accomplished in less time. Larger address spaces are used efficiently giving the machine access to more memory.

Unfortunately, today, 8-bit processors are considered old and uninteresting technology. Although 16- and 32-bit machines are glamorous, the 8-bit machines are still the work horses of process control and embedded systems. It seems management often has a 32-bit frenzy; they want a product's data sheet to proudly claim "includes a 32-bit CPU". It is axiomatic that using a wider bus will cost more than a narrower one and use more PC board real estate and power.

If performance demands a 32-bit computer, by all means use one. But one must be prepared to pay the price in the total system cost. A 16-bit machine usually offers more performance than 8-bits. One of the biggest advantages of a 16-bit processor over an 8-bit machine is its ability to address a larger address space. Most 8-bit machines are limited to a 64K memory space, where a 16-bit machine generally supports 1 to 16Mbytes. The memory may be cumbersome to access as in the 8088 series, but it is available. A large memory space is important for big data structures and programs.

Huge programs are becoming more prevalent due to dramatic increases in software costs. Ten years ago virtually all microcomputer programs were coded in assembly language. Today more and more companies are switching to High Level Languages (HLL) but, at a price. In typical embedded control applications it is very common to find that the majority of code is still written in assembly language, especially for tight control loops. Whereas, often non-critical macros are often written in high level languages. Embedded controllers predominantly use 8-bit CPU cores as opposed to 16/32-bit, due to their price performance ratio, especially when the "total system cost" is evaluated.

---

## Z80 DEVELOPMENT SUPPORT

---

### Z80 Family of Embedded Controllers

Figure 3 outlines the Z80 family of embedded controllers. They are discussed briefly to demonstrate the products available to suit the wide range of embedded control applications.

#### Z84C90 (KIO)

The Z84C90, or KIO, was Zilog's first Superintegration™ product which combines all of the Z80 peripherals SIO, CTC, and PIO into a single piece of silicon. Eight additional parallel I/O lines have been added. Housed in an 84-pin PLCC or 80 pin QFP package, the Z84C90 is used in existing Z80 applications as well as new applications requiring less real estate, power consumption, etc. For example, the Z84C90 is utilized in cellular telephones, where serial channels are required for display purposes.

#### Z84C50 (RAM80)

The next product is the Z84C50 which combines Z80 CPU with 2K bytes of Static RAM. The RAM can be accessed with no wait states thereby allowing the overall system performance to be increased. The Z84C50 is pin compatible with the Z80 discrete device, thus reducing design time. The CPU can be tri-stated allowing use of existing development systems. This product development approach is used on all products containing a CPU. Typical applications for the Z84C50 are where larger memory space is needed and a section is required to be executed on quickly. This can be put into the 2K SRAM cache memory and accessed with no wait states. An example would be a disk controller where a particular segment is loaded into the on-board memory. Other application examples are products utilizing relational databases like a phonetic intelligent dictionary where a large database is scanned, but a small section is pulled into memory for scrolling.

#### Z80280

The Z80280 offers the highest level of integration of any Z80 based product and includes a 16-Bit Z80 code compatible CPU that runs extra instructions. The CPU has a three stage pipeline that allows overlapping of fetch, decode and execution of instructions. An on-board MMU allows opening of the address space up to 16 MByte (as opposed to one MByte of the Z80180 and 64K of the Z8400) plus a 256 byte instruction and data cache is included for commonly used parameters. Like the Z80180, the Z80280 has on board many peripherals

including four DMAs, three 16-bit Counter timers, UART, etc. The Z80280 is often found in data communications applications where the four DMA channels are utilized with a dual channel data communications device like the SIO or Zilog's SCC.

#### Z84C15 (IPC)

The Z84C15 or Intelligent Peripheral Controller combines all of the original Z80 family members into one piece of silicon, i.e., CPU, PIO, CTC and SIO. This product is used not only in traditional applications that have used discrete components in the past, but is currently taking advantage of Superintegration with new application areas becoming feasible. One example is that of error correcting modems that utilize MNP software (Microcom Networking Protocol), the architecture of which is written around the Z84C15.

#### Z84C11 PIC

The Z84C11 was developed for those applications using parallel I/O lines (keyboards, displays, etc.). This product contains the Z84C00 CPU combined with the Z84C20 (CTC) and 40 parallel I/O lines.

### SUMMARY

Z80, the world's most popular 8-bit architecture, is an accepted standard in embedded control applications. Engineers and project leaders are learning that in an embedded control or intelligent peripheral control function, frequently, 8-bit CPU's offer the required price performance point. From a semiconductor manufacturer's point of view, die size is directly proportional to cost. For this reason 8-bit and some 16-bit architectures will remain dominate in the embedded control application areas.

The Z80 offers a very competitive, price performance advantage over other architectures. Further, it is well known and accepted. More lines of code have been written for the Z80 than any other microprocessor. Specifically, the Z80 architecture readily lends itself to embedded control applications. Apart from additional performance, the CPU exhibits a fast interrupt response time, which is critical.

With the technological advances via Superintegration, CPU's are easily integrated with peripherals and Z80 based architectures, making them ideally suited to embedded control applications.



# Superintegration™ Products Guide









## Embedded Controllers

Z80® Family	84C00 CPU		84C01		SIO		CTC		SIO		CGC		40 I/O		Z80 CPU		Z80/Z-Bus Interface		CTC	
	Power Down	OSC	2K Bytes SRAM	WSG	OSC	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA	PIA
<b>Block Diagram</b>	84C00 CPU Power Down		84C01 2K Bytes SRAM WSG		SIO OSC PIA		CTC PIA		SIO OSC PIA		CGC WDT CTC		40 I/O CTC WDT		Z80 CPU		Z80/Z-Bus Interface MMU CACHE		CTC 16 I/O Z180	
<b>Part #</b>	Z84C01	Z84C50	Z84C90	Z84013/C13	Z84015/C15	Z84011/C11	Z80180/Z8S180	Z80280	Z80181											
<b>Acronym</b>		RAM80™ IC	KIO™ IC	IPC/EIPC™	IPC/EIPC Controller	IPC/EIPC Controller	Z180™ MPU	Z280™ MPU	Z181™ MPU											
<b>Description</b>	Z80 CPU with CGC	Z80/84C01 with 2K SRAM	Killer I/O (3 Z80 peripherals)	Intelligent Peripheral Controller	Intelligent Peripheral Controller	Parallel I/O Controller	High-performance Z80 CPU with peripherals	16-bit Z80 code compatible CPU with peripherals	Zilog I/O Controller											
<b>Features</b>	Z80 CPU On-board oscillator with reduced components Four power-down modes	Z80 CPU 2 Kbytes SRAM Zero wait-state WSG Sync. memory Oscillator Pin compatible with Z84C00 DIP & PLCC EV mode¹	SIO, PIO, CTC plus 8 I/O lines	Z80 CPU, SIO, CTC WDT, CGC, WSG, Power-On Reset Two chip selects EV mode¹	Z80 CPU, SIO, CTC WDT, CGC The Z80 Family in one device Power-On Reset Two chip selects 32-bit CRC WSG EV mode¹	Z80 CPU, CTC, WDT 40 I/O lines bit programmable Power-On Reset EV mode¹	Enhanced Z80 CPU MMU 1 Mbyte 2 DMAs 2 UARTs with BRGs C/Serial I/O Port Oscillator Z8S180 includes; Pwr down, Prgmble EMI, divide-by-one clock option	16-bit code compatible Z80 CPU Three stage pipe MMU 16 Mbyte CACHE 256 byte Inst. & Data Peripherals 4 DMAs, UART, 3 16-bit C/T, WSG Z80/Z-BUS® interface	Complete Z180 plus SCC/2 CTC 16 I/O lines Emulation Adapter¹											
<b>Speed MHz</b>	6, 10	10	8, 10, 12.5	6, 10, 16	6, 10, 16	6, 10	6, 8, 10, 16*, 20* *Z8S180 only	10, 12	10, 12.5											
<b>Package</b>	44-pin PLCC 40-pin QFP	40-pin DIP 44-pin PLCC 44-pin QFP	84-pin PLCC	84-pin PLCC	100-pin QFP 100-pin VQFP	100-pin QFP	64-pin DIP 68-pin PLCC 80-pin QFP Z8S180 in PLCC and QFP only	68-pin PLCC	100-pin QFP											
<b>Application</b>	All Z80 uses	All Z80 consumer products	All Z80 uses	Intelligent datacom controllers	Intelligent peripheral controllers Modems	Intelligent parallel-I/O controllers Industrial display terminals	Embedded Control	Embedded Control Terminals Printers	Intelligent peripheral controllers Printers, Faxes, Modems, Terminals											

¹ Allows use of existing development systems.



**Z8000® Family**  
**Zilog Superintegration™ Products Guide**  
**Communication Products**

Datacom Block Diagram								
<b>Part #</b>	<b>Z8030/Z80C30</b> <b>Z8530/Z85C30</b>	<b>Z16C35/Z85C35</b>	<b>Z85230</b> <b>Z80230</b>	<b>Z85C80</b>	<b>Z16C50</b>	<b>Z16C30</b>	<b>Z16C33</b>	<b>Z16C32</b>
<b>Acronym</b>	<b>SCC</b> Controller	<b>ISCC™</b> Controller	<b>ESCC™</b> Controller	<b>SCSCI™</b> Interface	<b>DDPLL™</b> IC	<b>USC™</b> Controller	<b>MUSC™</b> Controller	<b>IUSC™</b> Controller ✓
<b>Description</b>	Serial Com. Controller	Integrated Serial Com. Controller	Enhanced Serial Com. Controller	Serial Com. and Small Computer Interface	Dual Digital Phase-Locked Loop	Universal Serial Controller	Mono-channel Universal Serial Controller	Integrated Universal Serial Controller
<b>Process/Speed/Clock Data Rate</b>	NMOS: 4,6,8 MHz CMOS: 8,10, 16 MHz 2, 2.5, 4 Mb/s	CMOS: 10, 16 MHz 2.5, 4.0 Mb/s	CMOS: 8, 10, 16 20 MHz 2.0, 2.5, 4.0, 5.0 Mb/s	CMOS: SCC - 10, 16 MHz SCSI - 3.0 MB/s	CMOS: 10 MHz 2.5 Mb/s	CMOS: 20 MHz CPU Bus 10 Mb/s 20 Mb/s	CMOS: 20 MHz CPU Bus 10 Mb/s	CMOS: 20 MHz CPU Bus 20 Mb/s
<b>Features</b>	Two independent full-duplex channels Enhanced DMA support: 10x19 status FIFO 14-bit byte counter NRZ/NRZI/FM	Full dual-channel SCC plus 4 DMA controllers and a bus interface unit	Full dual-channel SCC plus deeper FIFOs: 4 bytes on Tx 8 bytes on Rx DPLL counter per channel Software compatible to SCC	Full dual-channel SCC plus SCSI sharing data bus and read/write functions	Two independent DPLLs Selectable clock rate Sampling edge and decoding status output.	Two dual-channel 32-byte receive & transmit FIFOs 16-bit bus B/W: 18 Mb/s 2 BRGs per channel Flexible 8/16-bit bus interface	Single-channel (half of USC) plus a Time Slot Assigner for ISDN	Single-channel (half of USC) plus two DMA controllers. Array chained and linked-list modes with ring buffer support.
<b>Package</b>	40-pin DIP 44-pin CERDIP 44-pin PLCC	68-pin PLCC	40-pin DIP 44-pin PLCC	68-pin PLCC	28-pin DIP	68-pin PLCC	68-pin PLCC	68-pin PLCC
<b>Application</b>	General-Purpose datacom.	High performance datacom. SCC upgrades	General-Purpose datacom. High performance SCC software compatible upgrade	AppleTalk® networking SCSI disk drives	High-speed datacom. Increases data rate in SCC designs	General-Purpose high-end datacom. Ethernet HDLC X.25	General-Purpose high-end datacom. Ethernet HDLC X.25	General-Purpose high-end datacom. Ethernet HDLC X.25



## Z8000® Family Superintegration™ Products Guide

Z8000® Family	Z16C01/2/3	Z16C04	Z8036 Z8536	Z8038	Z16C20	Z80210	Z32H00	Z5380 Z53C80
<b>Description</b>	Central Processing Unit (CPU)	Central Processing Unit (CPU)	Counter/Timer & parallel I/O Unit (CIO)	FIFO Input/Output Interface Unit	General Logic Unit (GLU)	Dual-Memory Management Unit (DMMU)	Hyperstone Enhanced Fast Instruction Set Computer (EFISC) Embedded (RISC) Processor	Small Computer System Interface (SCSI)
<b>Process/Speed</b>	CMOS 10 MHz	CMOS 16 MHz	NMOS 4,6 MHz	NMOS 4,6 MHz	CMOS 10, 16 MHz	CMOS 16 MHz	CMOS 25 MHz	CMOS 10 MHz, Z5380: 1.5 MB/s Z53C80: 3.0 MB/s
<b>Features</b>	General-Purpose register architecture. Part number differs depending on number of segments supported, i.e., 1 seg. = 64 Kbytes; 01 = 128 segments 02 = 1, 03 = 32	8 Mbyte segmented addressable memory (128 seg. x 64 Kbytes)	Three 16-bit Counter/Timers, Three I/O ports with bit catching, pattern matching interrupts and handshake I/O	128-byte FIFO buffer with CPU interface on both sides.	8 Mbyte memory control interface DMA Clock Generator bus interface Watch-Dog Timer Wait-State Generator	Two Z8010 type MMUs in one package.	32-bit MPU 4 Gbytes address space 19 global and 64 local registers of 32 bits each 128 bytes instruction cache 1.2µ CMOS 42 mm² die	Direct SCSI bus interface On-board 48 mA drivers DMA Bus interface, target and initiator Glitch Eater (Z53C80 only)
<b>Package</b>	48-pin PDIP/CERDIP (Z16C01) 40-pin DIP/CERDIP 44-pin PLCC (Z16C02) 44-pin PLCC (Z16C03)	68-pin PLCC	40-pin PDIP 44-pin PLCC	40-pin DIP 44-pin PLCC	84-pin PLCC	68-pin PLCC	144-pin PGA	Z5380: 40-pin DIP 44-pin PLCC Z53C80: 48-pin DIP 44-pin PLCC
<b>Application</b>	Embedded controllers High-performance General-Purpose CPU	Embedded controllers High-performance General-Purpose CPU	General-Purpose Counter/Timers and I/O system designs	Multiple CPU systems I/O subsystems	Integrated embedded controllers FAX equip., tape drives, copiers	High-speed General-Purpose CPU memory management	Embedded high-performance industrial controller Workstations	Bus host adapters, formatters, host ports

\*Software and hardware compatible with discrete devices.



High Performance Micros For Application Specific Products

Z8® Family Block Diagram	1K ROM	2K ROM	2K/4K ROM	4K ROM	4K ROM	8K ROM   UART	16K ROM   UART
	SPI   CPU WDT   124 RAM P2   P3	CPU WDT   124 RAM P2   P3   P0	CPU WDT   124 RAM P2   P3	CPU WDT   236 RAM P2   P3   P0	CPU WDT   236 RAM   P1 P2   P3   P0	CPU 236 RAM P0   P1   P2   P3	CPU   236 RAM P0   P1   P2 P3   P4   P5   P6
Part #	Z86C06/Z86E09	Z86C08/Z86E08	Z86C09/C19/E09	Z86C30/E30	Z86C40/E40	Z86C21/E21/E22	Z86C61/62/96
Description	Z8® Consumer Controller Processor (CCP™) with 1K ROM, SPI, 2 Ports, 18-pin package E09: OTP version	Z8® MCU with 2K ROM, 3 Ports, 18-pin package E08: OTP version	Z8® Consumer Controller Processor (CCP™) with 2K/4K ROM, 3 Ports, 18-pin package E09: OTP version (Q4/92)	Z8® Consumer Controller Processor (CCP™) with 4K ROM, 3 Ports, 28-pin package E30: OTP version	Z8® Consumer Controller Processor (CCP™) with 4K ROM, 4 Ports, 40-, 44-pin package E40: OTP version	Z86C21; Z8® MCU with 8K ROM Z86E21; OTP version of Z86C21 Z86E22; Low Noise version of Z86E21	Z8® MCU with 16K ROM Z86C61 40-, 44-pin Z86C62 68-pin Z86C96 68-pin ROMless
Process/Speed	CMOS 8, 12 MHz	CMOS 8, 12 MHz	CMOS 8, 12 MHz	CMOS 8, 12 MHz	CMOS 8, 12 MHz	CMOS 12, 16 MHz	CMOS 8, 12, 16 MHz
Features	Watch-Dog Timer (WDT) 2 Analog Comparators 2 Standby Modes (STOP and HALT) Serial Peripheral Interface RC OSC option 2 Counter/Timers Auto Power-On Reset Low EMI mode Low Noise option 3 volt operation (Z86C06)	Watch-Dog Timer (WDT) 2 Analog Comparators 2 Standby Modes (STOP and HALT) ROM Protect option 2 Counter/Timers Auto Power-On Reset 3 volt operation (Z86C08) Low EMI option	Watch-Dog Timer (WDT) 2 Analog Comparators 2 Standby Modes (STOP and HALT) 2 Counter/Timers Z86C09/E09 = 2K ROM Z86C19 = 4K ROM RC OSC option ROM Protect option Auto Power-On Reset 3 volt operation (Z86C09/C19)	Watch-Dog Timer (WDT) 2 Analog Comparators 2 Standby Modes (STOP and HALT) 2 Counter/Timers RC OSC option ROM Protect option Auto Power-On Reset RAM Protect option 236 byte RAM 3 volt operation (Z86C30) Low EMI mode	Watch-Dog Timer (WDT) 2 Analog Comparators 2 Standby Modes (STOP and HALT) 2 Counter/Timers RC OSC option ROM Protect option Auto Power-On Reset RAM Protect option 236 byte RAM 3 volt operation (Z86C40) Low EMI mode	8K ROM for 86C21 and 8K OTP for 86E21 Full-duplex UART 2 Standby Modes (STOP and HALT) 2 Counter/Timers 2 Counter/Timers (STOP and HALT) 2 Counter/Timers ROM Protect option 236 Byte RAM RAM Protect option Low EMI option	16K ROM Full duplex UART 2 Standby Modes (STOP and HALT) 2 Counter/Timers ROM Protect option RAM Protect option Pin compatible to Z86C21 7 Ports (Z86C62/96)
Package	18-pin DIP 18-pin SOIC	18-pin DIP 18-pin SOIC	18-pin DIP 18-pin SOIC	28-pin DIP	40-pin DIP 44-pin PLCC 44-pin QFP	40-pin DIP 44-pin PLCC 44-pin QFP	40-pin DIP 44-pin PLCC 68-pin PLCC
Application	Automotive Toys & Games Security	Automotive Toys & Games Security	Automotive Appliances Sports Equipment Remote Controls Toys	Automotive Answering Machines Coin Changers Card Readers	Automotive Weight Scales Coin Changers Robotics Toys	Software Debug Z8 prototypes and small Z8 production runs Card Reader	Cable Television Remote Control Security


**Peripherals  
Business Unit**

	Mass Storage			Modem/FAX			Mouse			Keyboard																																																																											
<b>Z8® Family Block Diagram</b>	<table border="1"> <tr><td>MULT</td><td>DIV</td><td>UART</td></tr> <tr><td>CPU</td><td colspan="2">OSC</td></tr> <tr><td>236 RAM</td><td colspan="2">CLOCK</td></tr> <tr><td>P0</td><td>P1</td><td>P2</td><td>P3</td></tr> </table>	MULT	DIV	UART	CPU	OSC		236 RAM	CLOCK		P0	P1	P2	P3	<table border="1"> <tr><td>MULT</td><td>DIV</td><td>UART</td></tr> <tr><td>CPU</td><td colspan="2">DSP</td></tr> <tr><td>DAC</td><td colspan="2">PWM</td></tr> <tr><td>ADC</td><td colspan="2">SPI</td></tr> <tr><td>P2</td><td>P3</td><td>A15-0</td></tr> </table>	MULT	DIV	UART	CPU	DSP		DAC	PWM		ADC	SPI		P2	P3	A15-0	<table border="1"> <tr><td>88-Bit R.S. ECC</td><td>Buffer RAM Contrl.</td><td>AT/IDE Infc. P2 P3</td></tr> <tr><td colspan="2">Z86C95 MCU</td><td>8-Bit A/D 8-Bit D/A</td></tr> </table>	88-Bit R.S. ECC	Buffer RAM Contrl.	AT/IDE Infc. P2 P3	Z86C95 MCU		8-Bit A/D 8-Bit D/A	<table border="1"> <tr><td colspan="2">DSP</td></tr> <tr><td>512 RAM</td><td>4K ROM</td></tr> <tr><td colspan="2">16-bit MAC</td></tr> <tr><td>DATA I/O</td><td>RAM I/O</td></tr> </table>	DSP		512 RAM	4K ROM	16-bit MAC		DATA I/O	RAM I/O	<table border="1"> <tr><td>Z8</td><td>DSP</td></tr> <tr><td>26 KB ROM</td><td>4 KW ROM</td></tr> <tr><td>A/D</td><td>D/A</td></tr> <tr><td colspan="2">47 DIGITAL I/O</td></tr> </table>	Z8	DSP	26 KB ROM	4 KW ROM	A/D	D/A	47 DIGITAL I/O		<table border="1"> <tr><td colspan="3">2K ROM</td></tr> <tr><td colspan="3">CPU</td></tr> <tr><td>WDT</td><td colspan="2">124 RAM</td></tr> <tr><td>P0</td><td>P2</td><td>P3</td></tr> </table>	2K ROM			CPU			WDT	124 RAM		P0	P2	P3	<table border="1"> <tr><td colspan="4">4K/2K ROM</td></tr> <tr><td colspan="4">CPU</td></tr> <tr><td>UART</td><td colspan="3">124 RAM</td></tr> <tr><td>P0</td><td>P1</td><td>P2</td><td>P3</td></tr> </table>	4K/2K ROM				CPU				UART	124 RAM			P0	P1	P2	P3
MULT	DIV	UART																																																																																			
CPU	OSC																																																																																				
236 RAM	CLOCK																																																																																				
P0	P1	P2	P3																																																																																		
MULT	DIV	UART																																																																																			
CPU	DSP																																																																																				
DAC	PWM																																																																																				
ADC	SPI																																																																																				
P2	P3	A15-0																																																																																			
88-Bit R.S. ECC	Buffer RAM Contrl.	AT/IDE Infc. P2 P3																																																																																			
Z86C95 MCU		8-Bit A/D 8-Bit D/A																																																																																			
DSP																																																																																					
512 RAM	4K ROM																																																																																				
16-bit MAC																																																																																					
DATA I/O	RAM I/O																																																																																				
Z8	DSP																																																																																				
26 KB ROM	4 KW ROM																																																																																				
A/D	D/A																																																																																				
47 DIGITAL I/O																																																																																					
2K ROM																																																																																					
CPU																																																																																					
WDT	124 RAM																																																																																				
P0	P2	P3																																																																																			
4K/2K ROM																																																																																					
CPU																																																																																					
UART	124 RAM																																																																																				
P0	P1	P2	P3																																																																																		
<b>Part #</b>	<b>Z86C93</b>	<b>Z86C94/95</b>	<b>Z86C99 (Q2/92)</b>	<b>Z89C00 (Q1/92)</b>	<b>Z89C65/66 (Q1/92)</b>	<b>Z86C17</b>	<b>Z08614/E23/02</b>																																																																														
<b>Description</b>	Enhanced Z8®	Enhanced Z8® with DSP	Zilog Hard Disk Controller (ZHDC)	16-Bit Digital Signal Processor	Zilog Modem/Fax Controller (ZMFC)	Zilog Mouse Controller (ZMC™)	Zilog Keyboard Controller (ZKC™)																																																																														
<b>Process/Speed</b>	CMOS 24 MHz	CMOS 24 MHz	CMOS 24 MHz	CMOS 15 MHz	CMOS 10 MHz	CMOS 4 MHz	NMOS 4 MHz (8614/8602) CMOS 4 MHz OTP(86E23)																																																																														
<b>Features</b>	16x16 Multiply 1.7 µs 32x16 Divide 2.0 µs Full duplex UART 2 Standby Modes (STOP and HALT) 3 16-bit Counter/Timers Pin compatible to Z86C91	8-bit ADC, 8-bit DAC 16-bit Multiply/Divide Full duplex UART SPI (Serial Peripheral Interface) 3 Standby Modes (STOP/HALT/PAUSE) Pulse Width Modulator 3x16-bit timer 16-bit DSP slave processor 83 ns Mult./Accum.	3 volt operation Low profile package 88-bit Reed Solomon ECC Full AT/IDE bus interface 16-bit DSP servo controller Z8 Supervisory controller SRAM, DRAM buffer control 1.7 µs A/D 2.0 µs D/A	16-bit Mult./Accum. 2 data RAMs (256 words each) 24-bit ALU 4K word ROM 64Kx16 Ext. RAM 16-bit I/O Port 111 Instructions Single cycle clock instruction 3 INT, 2 COMP inputs Library of software macros available	Z8 controller with 24-byte ROM 24-bit DSP with 4K word ROM 8-bit A/D with AGC Library of software macros available 10-bit D/A (PWM) 47 I/O pins	Watch-Dog Timer (WDT) 2 Analog Comparators 2 Standby Modes (STOP and HALT) 2 Counter/Timers Auto Power-On Reset One Trip-Point Input Low Noise option Built-in pull-up resistor 3 volt operation	Full duplex UART Low EMI Mode 101/102 keyboard firmware available Open drain outputs 4K ROM (Z8614) 2K ROM (Z8602) 8K OTP ROM (Z86E23)																																																																														
<b>Package</b>	40-pin DIP 44-pin PLCC 48-pin VQFP	80-pin QFP 100-pin VQFP	144-pin QFP 120-pin VQFP	68-pin PLCC	68-pin PLCC	18-pin DIP 18-pin SOIC	40-pin DIP																																																																														
<b>Application</b>	Disk Drives Tape Drives Modems	Disk Drives Tape Drives Servo Control Motor Control	Disk Drives	General-Purpose DSP TMS 32010/20/25 applications	Multimedia Answering Machines Speech Storage and Transmission Modems FAXes	Mouse Track Balls	Keyboards																																																																														

### Video Products/Multimedia

Z8® Family Block Diagram	8K ROM	8K ROM	6K ROM	ROM	CHAR ROM	1K ROM	6K ROM
	4K CHAR ROM	4K CHAR ROM	3K CHAR ROM	CHAR ROM	COMMAND INTERPRETER	Z8 CPU	Z8 CPU
Part #	Z86C27	Z86127	Z86227	Z86327 (Q4/92)	Z86128	Z86L06	Z86L29
	Digital Television Controller (DTC™) (Fully Featured)	Digital Television Controller (DTC™) (Classic)	Digital Television Controller (DTC™) (Low-Cost)	Integrated Captioning Controller	Closed-Caption Controller	Low Voltage Controller	Low Voltage Controller
Description	Z8® Digital Television Controller MCU with logic functions needed for Television Controller, VCRs and Cable	Standard DTC features with reduced PWM outputs	Standard DTC features with reduced ROM, RAM, PWM outputs for greater economy	Advanced DTC family Integrated OSD and Closed Caption	Line 21 Controller (L21C™) for Closed Caption Television	18-pin Z8® Consumer Controller Processor (CCP™) low-voltage and low-current battery operation	18-pin Z8® Consumer Controller Processor (CCP™) low-voltage operations that require more ROM and output drive capabilities
Process/Speed	CMOS 4 MHz	CMOS 4 MHz	CMOS 4 MHz	CMOS 12 MHz	CMOS 12 MHz	Low Voltage CMOS 8 MHz	Low Voltage CMOS 8 MHz
Features	Z8/DTC Architecture 8K ROM, 256-byte RAM 160x7-bit video RAM On-Screen Display (OSD) video controller Programmable color, size, position attributes 13 PWMs for D/A conversion 128-character set 4Kx6-bit char. Gen. ROM Watch-Dog Timer (WDT) Brown-Out Protection 5 Ports/36 pins 2 Standby Modes Low EMI Mode	Z8/DTC Architecture 8K ROM, 256-byte RAM 160x7-bit video RAM OSD Video Controller Programmable color, size, position attributes 9 PWMs 128-character set 4Kx6-bit character generator ROM Watch-Dog Timer (WDT) Brown-Out Protection 4 Ports/28 pins 2 Standby Modes Low EMI Mode	Z8/DTC Architecture 6K ROM, 256-byte RAM 120x7-bit video RAM OSD on board Programmable color, size, position attributes 7 PWMs 96-character set 3Kx6-bit character generator ROM Watch-Dog Timer (WDT) Brown-Out Protection 3 Ports/20 pins 2 Standby Modes Low EMI Mode	4 Family Members: 24K OTP version 24K ROM version 16K ROM version 8K ROM version Highly integrated Z8 MCU with advanced OSD capability for supporting closed caption and other display functions	Conforms to FCC Line 21 format Parallel or serial modes Stand-alone operation On-board data sync and slicer On-board character generator Full attribute support: - Color - Blinking - Italic - Underline	Z8 Architecture 1K ROM Watch-Dog Timer 2 Analog Comparators with output option 2 Standby Modes 2 Counter/Timers Auto Power-On Reset 2 volt operation RC OSC option Low Noise option Brown-Out Protection High current drivers (2)	Z8 Architecture 6K ROM Watch-Dog Timer 2 Analog Comparators with output option 2 Standby Modes 2 Counter/Timers Auto Power-On Reset 2 volt operation RC OSC option Auto Latches Brown-Out Protection High-current LED drivers (4)
Package	64-pin DIP	64-pin DIP (52 pins Active)	40-pin DIP	64-pin DIP, 52-pin DIP 40-pin DIP	18-pin DIP	18-pin DIP 18-pin SOIC	18-pin DIP 18-pin SOIC
Application	High-end Television Cable/Satellite Receiver VCRs	Mid-level Television Cable/Satellite Receiver	Low-end Television Cable/Satellite Receiver	TVs, VCRs Satellite Receivers Cable TV Receivers Interactive TV Controllers	TVs, VCRs, Decoders	I.R. Controller Portable battery operations	I.R. Controller Portable battery operations




**Consumer  
Business Unit**

	Automotive		Auto/TAD	Telephone Answering Devices (TAD)																																																																					
<b>Z8® Family Block Diagram</b>	<table border="1"> <tr><td colspan="2">1K ROM</td></tr> <tr><td>SPI</td><td>CPU</td></tr> <tr><td>WDT</td><td>124 RAM</td></tr> <tr><td>P2</td><td>P3</td></tr> </table>	1K ROM		SPI	CPU	WDT	124 RAM	P2	P3	<table border="1"> <tr><td colspan="3">2K ROM</td></tr> <tr><td colspan="3">CPU</td></tr> <tr><td>WDT</td><td colspan="2">124 RAM</td></tr> <tr><td>P2</td><td>P3</td><td>P0</td></tr> </table>	2K ROM			CPU			WDT	124 RAM		P2	P3	P0	<table border="1"> <tr><td colspan="3">4K ROM</td></tr> <tr><td colspan="3">CPU</td></tr> <tr><td>WDT</td><td>236 RAM</td><td>P1</td></tr> <tr><td>P2</td><td>P3</td><td>P0</td></tr> </table>	4K ROM			CPU			WDT	236 RAM	P1	P2	P3	P0	<table border="1"> <tr><td>512 W RAM</td><td>8 RAM POINTERS</td></tr> <tr><td>16x16 MAC</td><td>4K W ROM</td></tr> <tr><td>STACK</td><td>REG</td></tr> <tr><td colspan="2">16-BIT I/O</td></tr> </table>	512 W RAM	8 RAM POINTERS	16x16 MAC	4K W ROM	STACK	REG	16-BIT I/O		<table border="1"> <tr><td>Z8</td><td>DSP</td></tr> <tr><td>24K ROM</td><td>4K ROM</td></tr> <tr><td>A/D</td><td>D/A</td></tr> <tr><td colspan="2">32 DIGITAL I/O</td></tr> </table>	Z8	DSP	24K ROM	4K ROM	A/D	D/A	32 DIGITAL I/O		<table border="1"> <tr><td>Z8</td><td>DSP</td></tr> <tr><td>24K ROM</td><td>4K ROM</td></tr> <tr><td>A/D</td><td>D/A</td></tr> <tr><td>31 DIGITAL I/O</td><td>EXT OUT</td></tr> </table>	Z8	DSP	24K ROM	4K ROM	A/D	D/A	31 DIGITAL I/O	EXT OUT	<table border="1"> <tr><td>Z8</td><td>DSP</td></tr> <tr><td>24K ROM</td><td>8K ROM</td></tr> <tr><td>RAM PORT</td><td>A/D</td></tr> <tr><td>RAM REFRESH</td><td>D/A</td></tr> <tr><td colspan="2">24 DIGITAL I/O</td></tr> </table>	Z8	DSP	24K ROM	8K ROM	RAM PORT	A/D	RAM REFRESH	D/A	24 DIGITAL I/O	
1K ROM																																																																									
SPI	CPU																																																																								
WDT	124 RAM																																																																								
P2	P3																																																																								
2K ROM																																																																									
CPU																																																																									
WDT	124 RAM																																																																								
P2	P3	P0																																																																							
4K ROM																																																																									
CPU																																																																									
WDT	236 RAM	P1																																																																							
P2	P3	P0																																																																							
512 W RAM	8 RAM POINTERS																																																																								
16x16 MAC	4K W ROM																																																																								
STACK	REG																																																																								
16-BIT I/O																																																																									
Z8	DSP																																																																								
24K ROM	4K ROM																																																																								
A/D	D/A																																																																								
32 DIGITAL I/O																																																																									
Z8	DSP																																																																								
24K ROM	4K ROM																																																																								
A/D	D/A																																																																								
31 DIGITAL I/O	EXT OUT																																																																								
Z8	DSP																																																																								
24K ROM	8K ROM																																																																								
RAM PORT	A/D																																																																								
RAM REFRESH	D/A																																																																								
24 DIGITAL I/O																																																																									
<b>Part #</b>	<b>Z86C06/Z86E09</b>	<b>Z86C08/Z86E08</b>	<b>Z86C30/E30 Z86C40/E40</b>	<b>Z89C00</b>	<b>Z89C65</b>	<b>Z89C66</b>	<b>Z89C67</b>																																																																		
<b>Description</b>	Z8® Consumer Controller Processor (CCP™) with 1K ROM, SPI E09 = OTP version (Q4/Q2)	Z8® MCU with 2K ROM E08 = OTP Version	Z8® Consumer Controller Processor (CCP™) with 4K ROM C30 = 28-pin C40 = 40-pin E30/E40 = OTP version	16-Bit Digital Signal Processor	Telephone Answering Controller with DSP LPC voice synthesis and DTMF detection	Telephone Answering Controller with DSP LPC voice synthesis and DTMF detection and external ROM/RAM interface	Telephone Answering Controller with digital voice encode and decode DTMF detection and full memory control interface																																																																		
<b>Process/Speed</b>	CMOS 8, 12 MHz	CMOS 8, 12 MHz	CMOS 8, 12 MHz	CMOS 10 MHz	CMOS 20 MHz	CMOS 20 MHz	CMOS 20 MHz																																																																		
<b>Features</b>	1K ROM, 124 RAM 2 Standby Modes 2 Counter/Timers ROM Protect 2 Ports, 14 I/O lines Serial Peripheral Interface Brown-Out Protection 2 Analog Comparators Low EMI option Watch-Dog Timer Auto Power-On Reset Low Power option	2K ROM, 124 RAM 2 Standby Modes 2 Counter/Timers ROM Protect 3 Ports, 14 I/O lines Brown-Out Protection 2 Analog Comparators Low EMI option Watch-Dog Timer Auto Power-On Reset	4K ROM, 236 RAM 2 Standby Modes 2 Counter/Timers ROM Protect RAM Protect 4 Ports (E6C40/E40) 3 Ports (E6C30/E30) Brown-Out Protection 2 Analog Comparators Low EMI (Z86E40) Watch-Dog Timer Auto Power-On Reset Low Power option	16-bit MAC 2 Data RAMs (256 Words each) 24-bit ALU 4K Word ROM 64Kx16 Ext. ROM 16-bit I/O Port 111 Instructions, most single-cycle Zero Overhead Pointers 3 Interrupts 2 User Outputs 2 User Inputs	Z8® Controller 24K Z8 ROM 16-bit DSP 4K DSP Word ROM 8-bit A/D with AGC DTMF macro available LPC macro available 10-bit PWM D/A Other DSP software options available 47 I/O Pins	Z8® Controller 64K External ROM 16-bit DSP 4K Word DSP ROM 8-bit A/D with AGC DTMF macro available LPC macro available 10-bit PWM D/A Other DSP software options available External ROM/RAM capability 31 I/O Pins	Z8® Controller 24K ROM 16-bit DSP 4K Word ROM 8-bit A/D with AGC DTMF macro available LPC macro available 10-bit PWM D/A Other DSP software options available ARAM/DRAM/ROM Controller & Interface Dual CODEC Interface																																																																		
<b>Package</b>	18-pin DIP 18-pin SOIC	18-pin DIP 18-pin SOIC	28-pin DIP 40-pin DIP 44-pin PLCC, QFP	68-pin PLCC	68-pin PLCC	68-pin PLCC	68-pin PLCC																																																																		
<b>Application</b>	Keyless Entry Window Control Door Control Dashboard Displays	Keyless Entry Window Control Door Control Dashboard Displays	Window Control Wiper Control Sunroof Control Security Systems TAD	General-Purpose DSP TMS 32010/20 Applications	Fully featured cassette answering machines with voice prompts and DTMF signaling	General-Purpose DSP applications in TAD and other high-performance 1-tape voice processors	Voice Processing DSP applications in TAD and other all digital voice processors																																																																		

# **Support Products Summary**



## Foreword

**Z**ilog recognizes the customer's need to successfully design and develop new systems or applications quickly with maximum reliability and performance. The company has developed new design/architectural support products to enhance the growing portfolio of new products in the datacommunication, intelligent peripheral control and microcontroller markets. This edition of the Support Products Summary features the latest component support tools for accomplishing these goals.

### New Products

Zilog has continued to introduce many new Superintegration™ CMOS components to fit multiple niche market designs. The company's proven core and cell library, one of the largest in the industry, is used to create Application Specific Standard Products (ASSPs) to meet the needs of specific market designs.

Zilog's support technology also allows customers to automate software design with innovative support tools such as the EPM™ (Electronic Programmer's Manual). The EPM Manual provides a turnkey modular approach to creating device driver software in "C" code. This approach provides an easy, flexible integration of datacommunication products into new system designs.

Choosing the right microprocessor/microcontroller architecture is key to

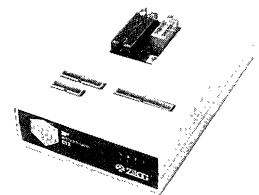
successful product development and development support. Zilog backs its customers with years of experience in engineering assistance, advanced integrated development tools, electronic interactive programs, training, documentation and much more.

### Z8® Microcontrollers

The Z8 family of microcontrollers offers optimum system cost/performance in specific markets such as mass storage, automotive, computer peripherals, speech processing and general purpose embedded control. Zilog's expertise has helped customers follow their product designs from inception through to completion. Time to market and maximum cost/performance are just a few of the important areas served by these support tools.

The new ICE BOX™ in-circuit emulation tools provide real-time diagnostic/test emulations and OTP programming support for Zilog's Z8 microcontrollers.

The emulator provides all the essential MCU timing and I/O circuitry to simplify user emulation of the prototype hardware/software products.



## Z80® Intelligent Peripherals

The Z80 family of intelligent peripheral controllers offers on-board intelligence for faxes, cellular phones, Local Area Network (LAN) network controllers, wireless controllers, printers, terminals, modems

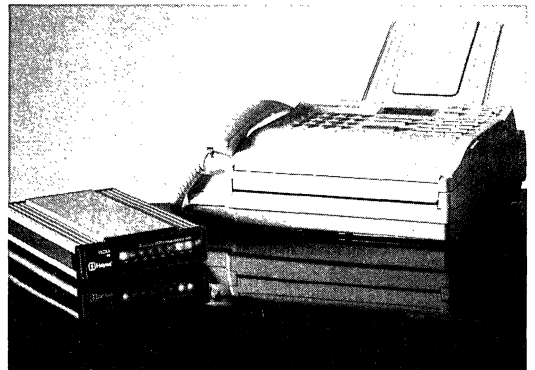


and general-purpose embedded control. To support the IPC line, Zilog offers evaluation kits that contain an assembled circuit board with supporting software documentation. These kits assist software and hardware development of customer designs.

The IPC family provides a software migration path for accelerating the customer design application from the lab through production assembly.

## Z8000® Datacommunications

The Z8000 datacommunications family has many innovative intelligent, multi-protocol components for the high bit-rate, serial datacommunications markets. These products provide customers with a tremendous selection and the most flexibility in protocols, system interfaces and data transfer rates in the industry. Zilog datacom controllers ensure quick, easy and flexible interconnection of hosts and peripherals in applications such as local area networks, metropolitan area networks, bridges, routers, gateways and wide area networks.



The innovative EPM is among the support tools offered for Zilog's datacommunications controllers.

To find out more about Zilog's growing family of Superintegration products and support tools, contact your local Zilog sales office or authorized distributor today.

## Table of Contents

### 1 Microcontrollers

Kit-to-Part Cross Reference Matrix .....	1-2
Z0860000ZCO Z8 Development Kit .....	1-3
Z0860000ZDP Z8600 Adapter Kit .....	1-4
Z0860200ZCO Z08602 101/102 Keyboard Kit .....	1-5
Z0860200ZDP Z08602 Adapter Kit .....	1-6
Z86C0800ZCO Z86C08 Application Kit .....	1-7
Z86C0800ZDP Z86C08 Adapter Kit .....	1-8
Z86E0800ZPR Z86E08 Programmer Kit .....	1-9
Z86C1200ZEM Z86C12 C Series ICE BOX Emulator Kit .....	1-10
Z86C1900ZCO Z86C19 Universal/I.R. Remote Control Kit .....	1-11
Z86C1900ZEM Z86C19 Evaluation Kit .....	1-12
Z86E0600ZDP Z86E06 Converter Kit .....	1-13
Z86E2100ZDF Z86E21 QFP OTP Program Adapter Kit .....	1-14
Z86E2100ZDP Z86E21 DIP OTP Program Adapter Kit .....	1-15
Z86E2100ZDV Z86E21 PLCC OTP Program Adapter Kit .....	1-16
Z86E2101ZDF Z86E21 QFP OTP Program Converter Kit .....	1-17
Z86E2101ZDV Z86E21 PLCC OTP Program Converter Kit .....	1-18
Z86E2300ZDP Z86E23 DIP OTP Program Adapter Kit .....	1-19
Z86C2700ZCO Z86C27 Application Kit .....	1-20
Z86C2701ZEM Z86C27 ICE BOX Emulator Kit .....	1-21
Z86C6200ZEM Z86C62 ICE BOX Emulator Kit .....	1-22
Z86E3000ZDP Z86E30 DIP OTP Program Adapter Kit .....	1-23
Z86E4000ZDF Z86E40 QFP OTP Program Adapter Kit .....	1-24
Z86E4000ZDP Z86E40 DIP OTP Program Adapter Kit .....	1-25
Z86E4000ZDV Z86E40 PLCC OTP Program Adapter Kit .....	1-26
Z86E4001ZDF Z86E40 QFP OTP Program Converter Kit .....	1-27
Z86E4001ZDV Z86E40 PLCC OTP Program Converter Kit .....	1-28
Z86C5000ZEM Z86C50 ICE BOX Emulator Kit .....	1-29
Z86C9300ZEM Z86C93 ICE BOX Emulator Kit .....	1-30
Z0880000ZCO Super8™ Development Kit .....	1-31
Z8 S Series Emulators, Base Units and Pods .....	1-32
Z86C5000ZPD Z86C50 S Series Emulator Pod	
Z86C1200ZPD Z86C12 S Series Emulator Pod	
Z86C9300ZPD Z86C93 S Series Emulator Pod	
Z86C0000ZUSP064 S Series Emulator Base Unit (64 Kbyte Program Memory)	
Z86C0000ZUSP128 S Series Emulator Base Unit (128 Kbyte Program Memory)	
Z86C0000ZUSP256 S Series Emulator Base Unit (256 Kbyte Program Memory)	

# SUPPORT PRODUCTS SUMMARY



## 2 Intelligent Peripherals

Kit-to-Part Cross Reference Matrix .....	2-2
Z8018000ZCO Z180/SCC Evaluation Kit .....	2-3
Z8018100ZCO Z80181 Evaluation Kit .....	2-4
Z8018101ZCO Z181™ MPU LLAP Evaluation Kit .....	2-4 A&B
Z8018100ZDP Z80181 Emulator Adaptor Kit .....	2-5
Z84C1100ZCO Z84C11 Evaluation Kit .....	2-6
Z84C1500ZCO Z84C15 Evaluation Kit .....	2-7
Z84C5000ZCO Z84C50 Evaluation Kit .....	2-8
Z84C9000ZCO Z84C01 Evaluation Kit .....	2-9
ZEPMIP00001 Z80181 Electronic Programmer's Manual .....	2-10

## 3 Datacommunications

Kit-to-Part Cross Reference Matrix .....	3-2
Z8523000ZCO Z85230 Evaluation Kit .....	3-3
Z16C0100ZCO Z16C01/20/30 Evaluation Kit .....	3-4
Z16C3001ZCO Z16C30/Z16C33 Evaluation Kit .....	3-5
Z8018600ZCO Z80186 Evaluation Board .....	3-6
ZEPMDC00001 USC™ Electronic Programmer's Manual .....	3-7
ZEPMDC00002 SCC Electronic Programmer's Manual .....	3-8

# **Microcontrollers**



# KIT-TO-PART CROSS REFERENCE MATRIX

## APPLICATIONS BOARDS TOOL BOX Family

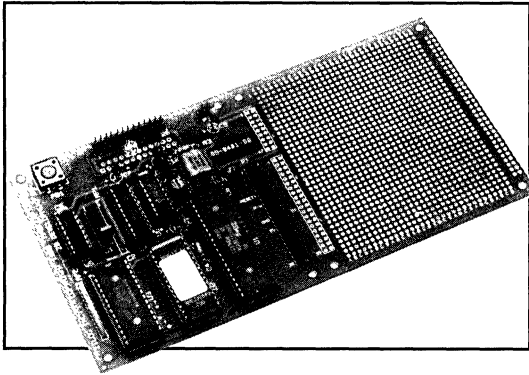
## EMULATORS ICE BOX™ Family

Part Number	Application Kits										Adapter Kits										Emulation Kits		C Series				S Series Pod			
	Z0860000ZCO	Z0860200ZCO	Z86C08800ZCO	Z86C1900ZCO	Z86C2700ZCO	Z0860000ZCO	Z860000ZDP	Z860200ZDP	Z86C0880ZDP	Z86E2100ZDP	Z86E2100ZDV	Z86E2100ZDF	Z86E300ZDP	Z86E400ZDP	Z86E400ZDV	Z86E400ZDF	Z86E4001ZDP	Z86E4001ZDV	Z86E4001ZDF	Z86C0800ZDP	Z86C1900ZEM	Z86E0880ZPR	Z86C1200ZEM	Z86C5000ZEM	Z86C8800ZEM	Z86C2701ZEM	Z86C6200ZEM	Z86C1200ZDp	Z86C8800ZDp*	Z86C9400ZDp
Z08600	●					●																								
Z08601	●																													
Z08602		●					●																							
Z08604																														●
Z08611	●																													
Z86C06																							●							●*
Z86C08		●						●															●			●†				●*
Z06E08		●						●															●			●†				
Z86C09			●																					●						
Z86C12																														
Z86C19			●																											●
Z86C21	●																													●
Z86E21	●								●	●	●	●	●	●	●	●	●	●	●	●	●	●	●			●				●
Z86C27			●																											
Z86E30											●																			●
Z86E40												●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Z86C61																							●							●
Z86C62																														
Z86C90																														●
Z86C91	●																													●
Z86C93	●																													●
Z86C94																														●
Z86C96																														
Z86C97			●																											
Z08800				●																										

\* Excluding SPI  
 † Zilog Z86C0800ZDP Required  
 ‡ Z86C0000ZUSP Base Unit Required

# Z0860000ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C91, Z86C21, Z8600, Z8601, Z8611**

### DESCRIPTION

The Z8<sup>®</sup> Development Kit can be used for several purposes. As an evaluation tool, one can learn the Z8 instruction set plus the manipulation of the Z8 MCU's interrupt vectors and register set. Secondly, the Z8 Development Kit is designed to aid the user in constructing specific applications using the Z8 microcontroller.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ 50 mA

#### Dimensions

Width: 4.0 in. (10.2 cm)

Length: 8.0 in. (20.3 cm)

#### Serial Interface

RS-232C @ 9600 baud

### KIT CONTENTS

#### Z8 Development Board

CMOS Z86C91 MPU

12 MHz Crystal

(32K)/8K x 8 EPROM

(32K)/8K x 8 Static RAM

RS-232C PC Interface

Z86C91 Expansion Header

#### Cables

25-Pin RS-232 Cable

#### Software (IBM<sup>®</sup> PC Platform)

Z8/Super8<sup>™</sup> Assembler and Utilities

Host Communication Package

Monitor Instructions

Tutorial

Sample Z86C91 Application Software

#### Documentation

Microcontrollers Data Book

Z8 Development Kit User Guide

Z8 Cross Assembler User Guide

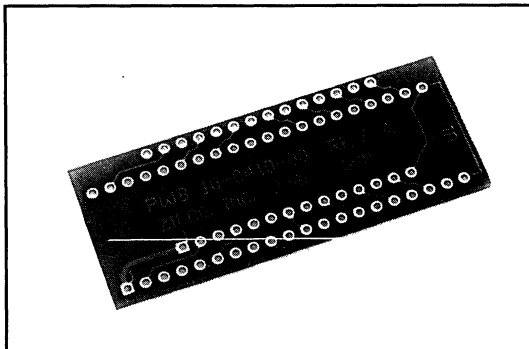
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z0860000ZCO

# Z0860000ZDP

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z8600**

### DESCRIPTION

The Z8600 Adapter Kit allows a standard Z8<sup>®</sup> emulator to emulate a Z8600 microcontroller by converting a 40-pin Z8 pin out MCU to a 28-pin Z8 pin out MCU.

### SPECIFICATIONS

#### Dimensions

Width: 0.9 in. (2.3 cm)

Length: 2.2 in. (5.4 cm)

### KIT CONTENTS

**Z8600 Adapter Board**

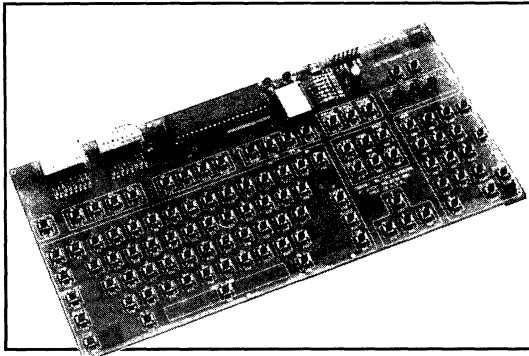
PC Board

### ORDERING INFORMATION

**Part No:** Z0860000ZDP

# Z0860200ZCO

## PRODUCT SPECIFICATION



## SUPPORTED DEVICES

**Z08602**

## DESCRIPTION

The kit contains an assembled circuit board, Z08602 with keyboard, ROM-code, and documentation to help the user become familiar with the features of the Z08602 keyboard controller.

The Z08602 microcontroller is designed into a 101/102 PC keyboard circuit to control all scan codes, line status modes, scan timing and communication between the keyboard and PC.

## SPECIFICATIONS

### Power Requirements

+5 Vdc @ .2 A (supplied by PC)

### Dimensions

Width: 4.6 in. (11.7 cm)

Length: 9.3 in. (23.6 cm)

## KIT CONTENTS

### Z08602 101/102 Keyboard

NMOS Z08602 MPU

2 MHz Crystal

101/102 Keyboard Option

3 LEDs

Two 8-Position Dip Switches

6-Pin Communication Header

### Software (IBM® PC Platform)

Contact Zilog for Licensing of Keyboard Source Code

### Documentation

Microcontrollers Data Book

Z8602 Application Note

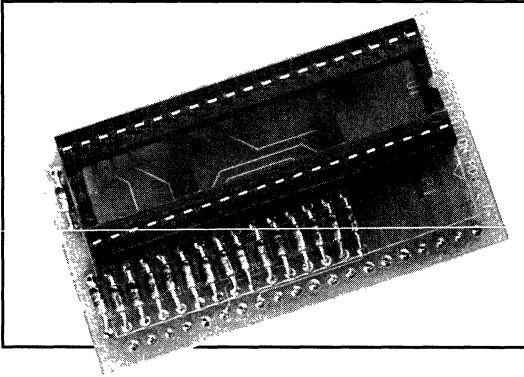
Z8602 Product Specification

## ORDERING INFORMATION

**Part No:** Z0860200ZCO

# Z0860200ZDP

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z08602**

### DESCRIPTION

The Z08602 adapter board is a tool used to adapt a standard Z8601 type device or emulation system to a Z8602 target socket.

### SPECIFICATIONS

#### Dimensions

Width: 1.3 in. (3.3 cm)

Length: 2.3 in. (5.8 cm)

### KIT CONTENTS

#### Z08602 Adapter Board

40-Pin Z08601/Z08611 MPU Socket

40-Pin Z08602 Connector

#### Documentation

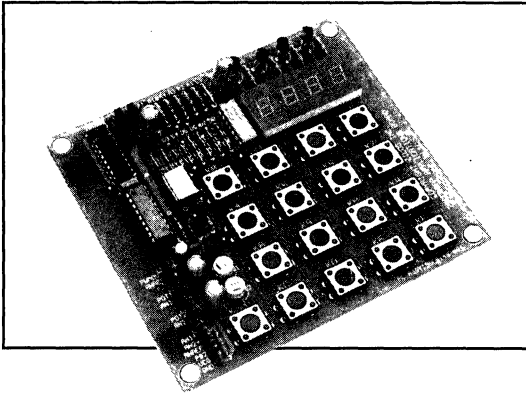
Z08602 Adapter Kit User Guide

### ORDERING INFORMATION

**Part No:** Z0860200ZDP

# Z86C0800ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C08**

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to help the user become familiar with the features of the Z86C08 microcontroller.

The Applications Board is used to demonstrate the advantages and versatility of the 18-pin Z8 device. Included is simple hardware and software that demonstrates the implementation of WDT, HALT, and STOP mode, low cost D to A and A to D conversion techniques.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ 50 mA

#### Dimensions

Width: 4.4 in. (11.2 cm)

Length: 4.8 in. (12.2 cm)

### KIT CONTENTS

#### Z86C08 Application Board

CMOS Z86C08 MPU

4 MHz Crystal

Four 7-segment LED Displays

17-Key Keypad

#### Software (IBM® PC Platform)

Application Source Code

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

#### Documentation

Microcontrollers Data Book

Z8 Cross Assembler User Guide

MOBJ Link/Loader User Guide

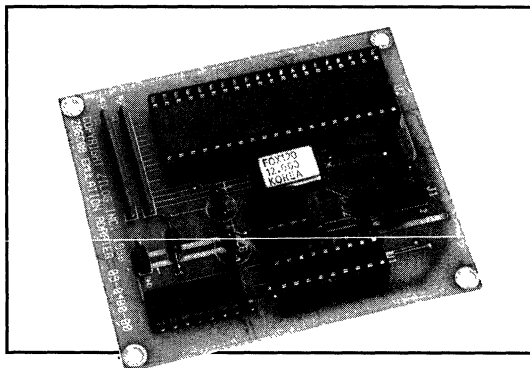
Z86C08 Application Kit User Guide

### ORDERING INFORMATION

**Part No:** Z86C0800ZCO

# Z86C0800ZDP

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C08**

### DESCRIPTION

The Z86C08 adapter board converts the Z8<sup>®</sup> MCU from a 40-pin pin out to an 18-pin pin out. This adapter board allows a standard Z8 emulation device to emulate the Z86C08. The Z86C08 Adapter Board is placed between the Z8 emulator and the user's target socket. The board does not emulate the watchdog timer function.

### SPECIFICATIONS

#### Dimensions

Width: 2.5 in. (6.4 cm)

Length: 2.9 in. (7.4 cm)

### KIT CONTENTS

#### Z86C08 Adapter Board

40-Pin Z8 MPU Socket

18-Pin Z86C08 Socket

12 MHz Crystal

#### Cables

18-Pin Z86C08 Emulation Cable

#### Documentation

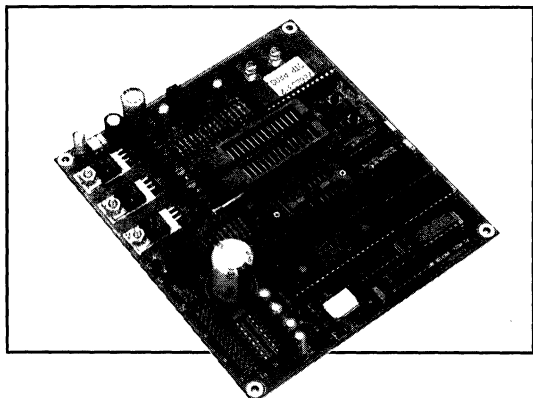
Z86C08 Adapter Kit User Guide

### ORDERING INFORMATION

**Part No:** Z86C0800ZDP

# Z86E0800ZPR

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

#### Z86E08

### DESCRIPTION

The Kit contains an assembled circuit board, software, and documentation to program the Z86E08 OTP.

The Z86E08 is an OTP version of the Z86C08 single-chip microcontroller housed in an 18-pin DIP. It offers the same architecture and all the features of the Z86C08. The Z86E08 also offers low noise and ROM protect options, which can be programmed by the programmer.

### SPECIFICATIONS

#### Power Requirements

+15 Vdc @ 1 A  
Or 12-15 Vac @ 1 A.

#### Dimensions

Width: 4.9 in. (12.4 cm)  
Length: 5.4 in. (13.7 cm)

### KIT CONTENTS

#### Z86E08 Programmer Board

CMOS Z86C91 MPU  
7.3728 MHz Crystal  
8K x 8 EPROM  
(32K)/8K x 8 ZIF Socket (for user EPROM)  
Z86E08 ZIF Socket  
Two 7805 Voltage Regulators  
7812 Voltage Regulator  
Bridge Rectifier  
2 LEDs  
2 Key Switches

#### Software (IBM® PC Platform)

Programming source code

#### Documentation

Z86E08 Product Specification  
Z86E08 Kit User Guide

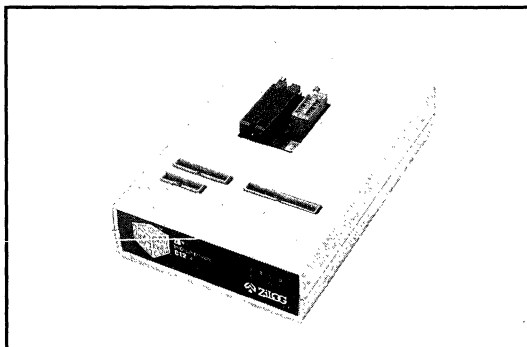
### ORDERING INFORMATION

**Part No:** Z86E0800ZPR



# Z86C1200ZEM

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C08, Z86E08, Z86C00, Z86C10, Z86C11  
Z86C20, Z86C21, Z86E21, Z86C61, Z86C91**

### DESCRIPTION

The Z86C1200ZEM is a member of Zilog's ICE BOX product family of in-circuit emulators. The ICE BOX -C12 provides emulation and OTP programming support for Zilog's Z8<sup>®</sup> microcontroller. The Emulator provides all the essential MCU timing and I/O circuitry which simplifies user emulation of the prototype hardware/software product. The Emulator can be connected to a serial port COM 1 or COM 2 of the host computer (IBM XT, AT compatible).

### SPECIFICATIONS

#### Emulation Specification

Maximum Emulation Speed 16 MHz

#### Power Requirements

+5 Vdc @ 1.0 A

#### Dimensions

Width: 6.0 in. (15.2 cm)

Length: 8.8 in. (22.4 cm)

#### Serial Interface

RS-232C @ 19200 baud

### KIT CONTENTS

#### Z86C12 Emulator

Z8 Emulation Base Board  
CMOS Z86C9120PSC  
8K x 8 EPROM  
(Programmed with Debug Monitor)  
EPM5128 EPLD  
32K x 8 Static RAM  
3 64K x 4 Static RAM  
RS-232C Interface  
Reset Switch

#### Z86C12 Emulation Daughter Board

EPM5032 EPLD  
16 MHz CMOS Z86C1216GSE ICE Chip  
40-, 18-Pin ZIF OTP Sockets  
80-, 60-, 40-Pin Target Connectors

#### Components

Z86E21 12PSC

Z86E08 12PSC

#### Cables

12", 40-Pin DIP Emulation Pod  
12", 28-Pin DIP Emulation Pod  
12", 18-Pin DIP Emulation Pod  
48" Power Cable  
15" Power Cable with Banana Plugs  
60" DB 25 RS-232C Cable

#### Software (IBM<sup>®</sup> PC Platform)

Z8/Z80<sup>®</sup>/Z8000<sup>®</sup> Cross Assembler  
MOBJ Link/Loader  
Host Package

#### Documentation

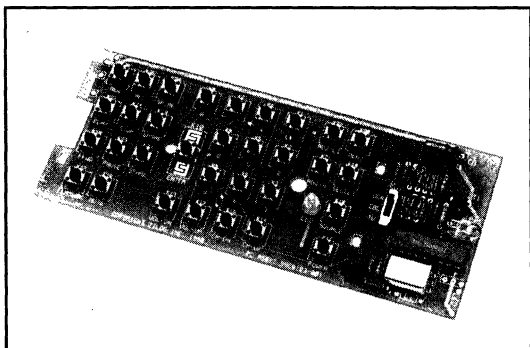
Emulator User Guide  
Z8 Cross Assembler User Guide  
MOBJ Link/Loader User Guide  
Registration Card

### ORDERING INFORMATION

**Part No:** Z86C1200ZEM

# Z86C1900ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

Z86C09, Z86C19

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation for the universal I.R. transmitter application. The transmitter can be set up to operate most models of remote-controlled TVs, VCRs and Cable TV Decoders even if they are in different brands.

With the set-up feature and the easy operation capability, the Universal I.R. Transmitter can be used to replace several remote controllers. The documentation contains the look up codes of each corresponding brand.

### SPECIFICATIONS

#### Power Requirements

$+3 < V_{CC} < +5$  Vdc

#### Dimensions

Width: 2.3 in. (5.8 cm)

Length: 5.5 in. (14.0 cm)

### KIT CONTENTS

#### Z86C19 Universal I.R. Remote Control Board

CMOS Z86C19 MPU (With Mask ROM)

8 MHz Crystal

32 Key Switches

#### Software (IBM® PC Platform)

Source code available with factory approval.

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

#### Documentation

Microcontroller Data Book

Z8 Cross Assembler User Guide

MOBJ Link/Loader User Guide

Z86C09/19 Product Specification

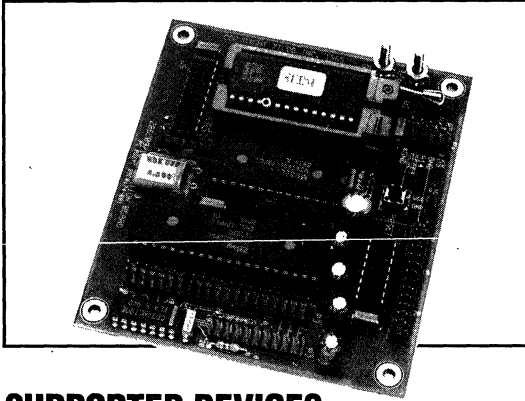
Z86C19 I.R. Application Kit User Guide

### ORDERING INFORMATION

**Part No:** Z86C1900ZCO

# Z86C1900ZEM

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z8604, Z86C09, Z86C19, Z86C90**

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the maskROM Z86C09/19 and ROMless Z86C90 devices.

The supplied cross assembler and link/loader package allows full assembly language programming support. A board resident debug monitor program allows object code to be down-loaded and subsequently debugged.

Code targeted for the Z86C09/19 device may be verified in the target application before submitting to Zilog for production masking.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .5 A

#### Dimensions

Width: 3.5 in. (8.9 cm)

Length: 4.0 in. (10.2 cm)

#### Serial Interface

RS-232C @ 9600 baud

### KIT CONTENTS

#### Z86C19 Evaluation Board

CMOS Z86C90 MPU

8 MHz Crystal

(32K)/8K x 8 ZIF Socket (supplied with  
Debug Monitor EPROM)

(32K)/8K x 8 Static RAM

RS-232C PC Interface

Z86C90 Expansion Header

Z86C09/19 Emulation Header

#### Cables

25-Pin RS-232 Cable

18-Pin Z86C19 Emulation Cable

#### Software (IBM® PC Platform)

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Resident Debug Monitor Source Code

Z86C09 Example Software

#### Documentation

Microcontrollers Data Book

Z86C09/19 Product Specification

Z86C30/40/90 Product Specification

Z86C19ZEM Kit User Guide

Z8 Cross Assembler User Guide

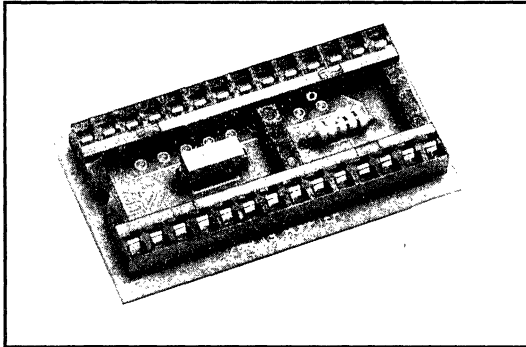
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z86C1900ZEM

# Z86E0600ZDP

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

Z86E06/09/19

### DESCRIPTION

The Z86E06 Program Converter Board is a simple adapter which converts the 28-pin footprint of the Zilog Z86E30 OTP chip to the 18-pin DIP configuration of the Z86E06/09/19 OTP chip. The converter supports all the functions of the Z86E06/09/19 except for SPI function.

### SPECIFICATIONS

#### Dimensions

Width: 0.8 in. (2.0 cm)

Length: 1.5 in. (3.8 cm)

### KIT CONTENTS

#### Z86E06 Program Converter Board

28-Pin Z86E30 MCU Socket

18-Pin Z86E06/09/19 Connector

#### Cables

25-Pin RS-232 Cable

18-Pin Z86C19 Emulation Cable

#### Documentation

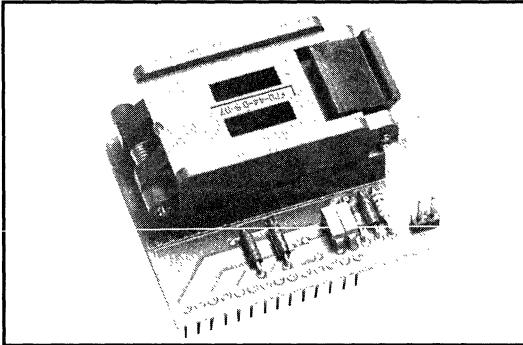
Z86E06 OTP Converter Kit User Guide

### ORDERING INFORMATION

**Part No:** Z86E0600ZDP

# Z86E2100ZDF

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86E21**

### DESCRIPTION

The Z86E21 QFP OTP Program Adapter Kit allows the 2764A standard EPROM programmer to program the Z86E21 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

+12.5 Vdc @ .5 A

#### Dimensions

Width: 1.75 in. (4.4 cm)

Length: 2.20 in. (5.6 cm)

### KIT CONTENTS

#### Z86E21 QFP OTP Program Adapter Board

44-Pin QFP ZIF Socket

28-Pin Connector

#### Documentation

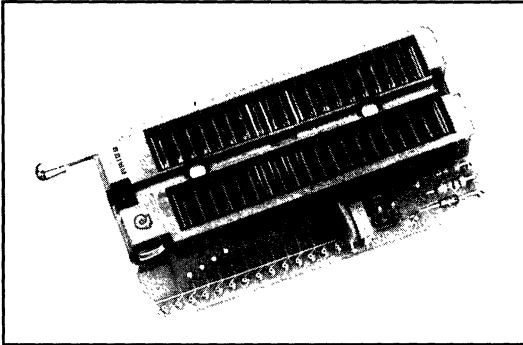
Z86E2100ZDF Adapter User Guide

### ORDERING INFORMATION

**Part No:** Z86E2100ZDF

# Z86E2100ZDP

## PRODUCT SPECIFICATION



## SUPPORTED DEVICES

**Z86E21**

## DESCRIPTION

The Z86E21 DIP OTP Program Adapter Kit allows the 2764A standard EPROM programmer to program the Z86E21 OTP microcontroller.

## SPECIFICATIONS

### Power Requirements

+12.5 Vdc @ .5 A

### Dimensions

Width: 1.4 in. (3.6 cm)

Length: 2.6 in. (6.6 cm)

## KIT CONTENTS

### Z86E21 OTP Program Adapter Board

40-Pin DIP ZIF Socket

28-Pin Connector

### Documentation

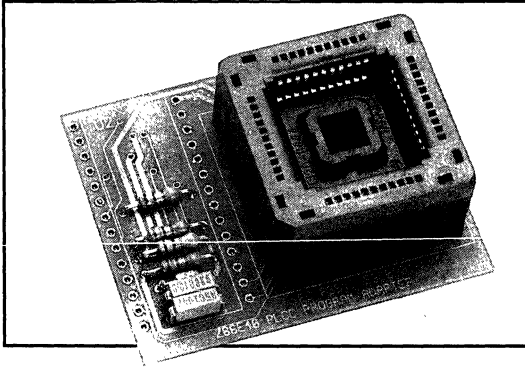
Z86E21ZDP Adapter User Guide

## ORDERING INFORMATION

**Part No:** Z86E2100ZDP

# Z86E2100ZDV

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86E21**

### DESCRIPTION

The Z86E21 PLCC OTP Program Adapter Kit allows the 2764A standard EPROM programmer to program the Z86E21 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

+12.5 Vdc @ .5 A

#### Dimensions

Width: 1.75 in. (4.4 cm)

Length: 2.20 in. (5.6 cm)

### KIT CONTENTS

#### Z86E21 PLCC OTP Program Adapter Board

44-Pin PLCC ZIF Socket

28-Pin Connector

#### Documentation

Z86E2100ZDV Adapter User Guide

### ORDERING INFORMATION

**Part No:** Z86E2100ZDV

# Z86E2101ZDF

## PRODUCT SPECIFICATION



Photograph  
Not Available  
At This Time

### SUPPORTED DEVICES

Z86E21

### DESCRIPTION

The Z86E21 OTP Program Converter Kit converts a 44-pin QFP package to a 40-pin DIP package, which allows the C12 ICE BOX™ to program the 44-pin QFP Z86E21 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

Not applicable.

#### Dimensions

Width: 2.0 in. (5.1 cm)

Length: 2.1 in. (5.3 cm)

### KIT CONTENTS

#### Z86E21 OTP Program Converter Board

44-Pin QFP ZIF Socket

40-Pin Connector

#### Documentation

Not applicable.

### ORDERING INFORMATION

**Part No:** Z86E2101ZDF



# Z86E2101ZDV

## PRODUCT SPECIFICATION



Photograph  
Not Available  
At This Time

### SUPPORTED DEVICES

Z86E21

### DESCRIPTION

The Z86E21 OTP Program Converter Kit converts a 44-pin PLCC package to a 40-pin DIP package, which allows the C12 ICE BOX™ to program the 44-pin PLCC Z86E21 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

Not applicable.

#### Dimensions

Width: 1.8 in. (4.6 cm)

Length: 2.1 in. (5.3 cm)

### KIT CONTENTS

#### Z86E21 OTP Program Converter Board

44-Pin PLCC ZIF Socket

40-Pin Connector

#### Documentation

Not applicable.

### ORDERING INFORMATION

**Part No:** Z86E2101ZDV

# Z86E2300ZDP

## PRODUCT SPECIFICATION



Photograph  
Not Available  
At This Time

### SUPPORTED DEVICES

Z86E23

### DESCRIPTION

The Z86E23 OTP Program Adapter Kit allows a Z86E21 OTP programmer to program the 40-pin DIP Z86E23 One-Time-Programmable microcontroller.

### SPECIFICATONS

#### Power Requirements

Not applicable

#### Dimensions

Width: 1.4 in. (3.6 cm)

Length: 2.6 in. (6.6 cm)

### KIT CONTENTS

#### Z86E23 OTP Program Adapter Board

40-Pin ZIF Socket

40-Pin Connector

### DOCUMENTATION

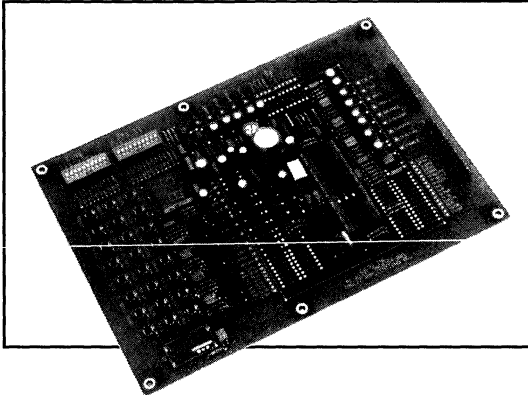
Z86E23ZDP Adapter User Guide

### ORDERING INFORMATION

**Part No:** Z86E2300ZDP

# Z86C2700ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C27, Z86C97**

### DESCRIPTION

The Z86C2700ZCO Application Kit is specifically designed for users to evaluate the hardware and software of Zilog's Z86C27 Digital Television Controller (DTC™). The Z86C2700ZCO Application Kit can be used with an Z86C2700ZEM Emulation Adapter Board to develop application code.

### SPECIFICATIONS

#### Power Requirements

Supplied by Television Set

#### Dimensions

Width: 6.2 in. (15.7 cm)

Length: 8.6 in. (21.8 cm)

### KIT CONTENTS

#### Z86C27 Application Board

CMOS Z86C27 MPU Socket  
4 MHz Crystal  
24 Key Multiplexed Keypad  
Two 7-segment LED Displays  
8 LEDs  
13 PWMs  
Low Pass Filter Interface  
PLL Interface

#### Documentation

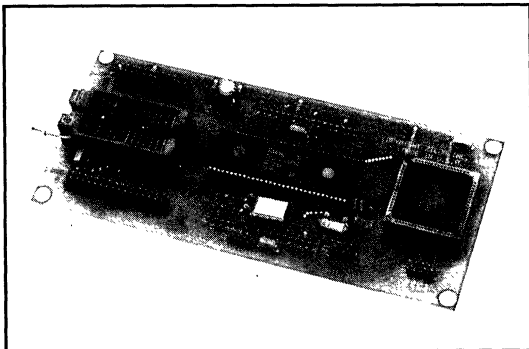
Microcontrollers Data Book  
Z86C27 Application Kit User Guide

### ORDERING INFORMATION

**Part No:** Z86C2700ZCO

# Z86C2701ZEM

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C27, Z86127, Z86C97**

### DESCRIPTION

The Z86C2701ZEM In-Circuit-Emulator (ICE BOX™) provides emulation support for Zilog's DTC™ Family of ICs. The emulator provides all the essential MCU timing and I/O circuitry which simplifies user emulation of the prototype hardware/software product. The emulator can be connected to a serial port COM 1 or COM 2 of the host computer (IBM® XT, AT compatible). An EPROM ZIF socket is provided for character ROM (CGROM) verification.

The Z86C2701ZEM daughter board can also be used in a standalone mode. An EPROM socket is provided for applying the application software. In Addition, it can be used in conjunction with an Orion Unilab™ 8620 Emulator to generate program code and provide a simple connector interconnection for this purpose.

### SPECIFICATIONS

#### Emulation Specification

Maximum emulation speed 16 MHz

#### Power Requirements

+5 Vdc @ 1.0 A

#### Dimensions

Width: 6.0 in. (15.2 cm)

Length: 8.8 in. (22.4 cm)

### KIT CONTENTS

#### Z86C12 Emulator

Z8 Emulation Base Board

CMOS Z86C9120PSC

8K x 8 EPROM (Programmed  
with Debug Monitor)

32K x 8 Static RAM

3 64K x 4 Static RAM

RS-232C Interface

Reset Switch

Z86C27 Emulation Daughter Board

16 MHz CMOS Z86C1216GSE ICE Chip

8K x 8 Static RAM

28-Pin ZIF Socket

6 HP-16500A Logic Analysis

System Interface Connectors

60/60 Pin Target Connectors

#### Cables

12" Z86C27 64-Pin DIP Emulation Pod

15" Power Cable with Banana Plugs

60" DB 25 RS-232C Cable

#### Software (IBM PC Platform)

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Host Package

#### Documentation

Emulator User Guide

Z8 Cross Assembler User Guide

MOBJ Link/Loader User Guide

Registration Card

### ORDERING INFORMATION

**Part No:** Z86C2701ZEM

# Z86C6200ZEM

## PRODUCT SPECIFICATION



Photograph  
Not Available  
At This Time

### SUPPORTED DEVICES

**Z86C62, Z86C96**

### DESCRIPTION

The Z86C6200ZEM is a member of Zilog's ICE BOX™ product family of in-circuit emulators. The ICE BOX C62 provides emulation for Zilog's Z86C62 (ROM device) and Z86C96 (ROMless device) micro-controllers. This includes all the essential MCU timing and I/O circuitry which simplifies user emulation of the prototype hardware/software product. The Emulator can be connected to a serial port COM 1 or COM 2 of the host computer (IBM® XT, AT, 386, 486 compatible).

### SPECIFICATIONS

#### Emulation Specification

Maximum emulation speed 16 MHz

#### Power Requirements

+5 Vdc @ .5 A

#### Dimensions

Width: 6.0 in. (15.2 cm)

Length: 8.8 in. (22.4 cm)

#### Serial Interface

RS-232C @ 19200 baud

### KIT CONTENTS

#### Z86C62 Emulator

Z8® Emulation Base Board

CMOS Z86C9120PSC

8K x 8 EPROM (Programmed  
with Debug Monitor)

32K x 8 Static RAM

3 64K x 4 Static RAM

RS-232C Interface

Reset Switch

#### Z86C62 Emulation Daughter Board

20 MHz CMOS Z86C9620VSC ICE Chip

5 HP-16500A Logic Analysis

System Interface Connectors

80/60 Pin Target Connector

#### Cables

12", Z86C96 68-Pin PLCC Emulation Pod

12", Z86C62 64-Pin DIP Emulation Pod

48" Power Cable

15" Power Cable with Banana Plugs

60" DB 25 RS-232C Cable

#### Software (IBM PC Platform)

Z8/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Host Package

#### Documentation

Emulator User Guide

Z8 Cross Assembler User Guide

MOBJ Link/Loader User Guide

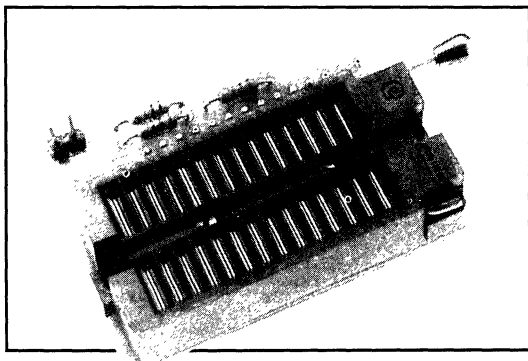
Registration Card

### ORDERING INFORMATION

**Part No:** Z86C620ZEM

# Z86E3000ZDP

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86E30**

### DESCRIPTION

The Z86E30 DIP OTP Program Adapter Kit allows a standard EPROM programmer to program the Z86E30 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

+12.5 Vdc @ .5A

#### Dimensions

Width: 1.45 in. (3.68 cm)

Length: 2.0 in. (5.08 cm)

### KIT CONTENTS

#### Z86E30 OTP Program Adapter Board

28-Pin DIP ZIF Socket

28-Pin Connector

#### Documentation

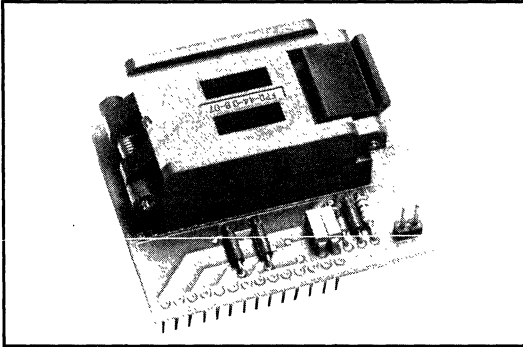
Z86E30ZDP Adapter User Guide

### ORDERING INFORMATION

**Part No:** Z86E3000ZDP

# Z86E4000ZDF

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86E40**

### DESCRIPTION

The Z86E40 QFP OTP Program Adapter Kit allows a standard EPROM programmer to program the Z86E40 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

+12.5 Vdc @ .5 A

#### Dimensions

Width: 1.75 in. (4.4 cm)

Length: 2.20 in. (5.6 cm)

### KIT CONTENTS

#### Z86E40 QFP OTP Program Adapter Board

44-Pin QFP ZIF Socket

28-Pin Connector

#### Documentation

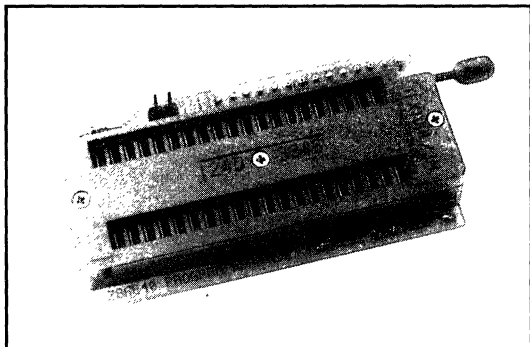
Z86E4000ZDF Adapter User Guide

### ORDERING INFORMATION

**Part No:** Z86E4000ZDF

# Z86E4000ZDP

## PRODUCT SPECIFICATION



### **KIT CONTENTS**

#### **Z86E40 DIP OTP Program Adapter Board**

40-Pin DIP ZIF Socket

28-Pin Connector

#### **Documentation**

Z86E40ZDP Adapter User Guide

### **ORDERING INFORMATION**

**Part No:** Z86E4000ZDP

### **SUPPORTED DEVICES**

**Z86E40**

### **DESCRIPTION**

The Z86E40 DIP OTP Program Adapter Kit allows a standard EPROM programmer to program the Z86E40 OTP microcontroller.

### **SPECIFICATIONS**

#### **Power Requirements**

+12.5 Vdc @ .5 A

#### **Dimensions**

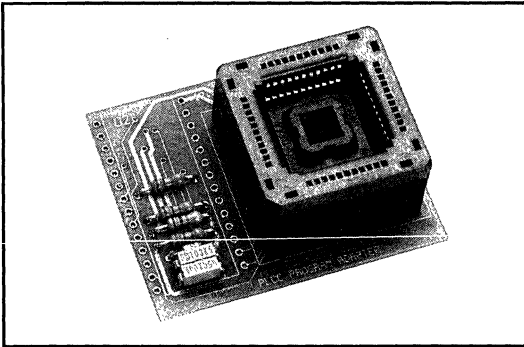
Width: 1.4 in. (3.6 cm)

Length: 2.6 in. (6.6 cm)



# Z86E4000ZDV

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86E40**

### DESCRIPTION

The Z86E40 PLCC DIP OTP Program Adapter Kit allows a standard EPROM programmer to program the Z86E40 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

+12.5 Vdc @ .5 A

#### Dimensions

Width: 1.6 in. (4.1 cm)

Length: 2.0 in. (5.1 cm)

### KIT CONTENTS

#### Z86E40 PLCC OTP Program Adapter Board

40-Pin ZIF Socket

28-Pin Connector

#### Documentation

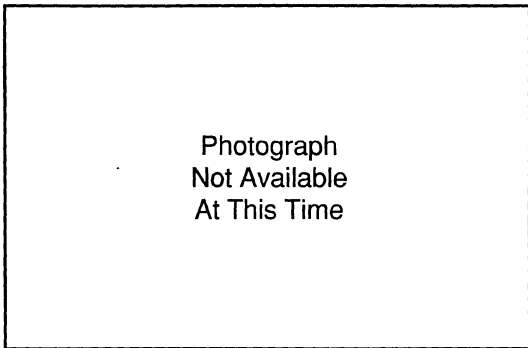
Z86E4000ZDV Adapter User Guide

### ORDERING INFORMATION

**Part No:** Z86E4000ZDV

# Z86E4001ZDF

## PRODUCT SPECIFICATION



### **KIT CONTENTS**

#### **Z86E40 OTP Program Converter Board**

44-Pin QFP ZIF Socket

40-Pin Connector

#### **Documentation**

Not applicable.

### **ORDERING INFORMATION**

**Part No:** Z86E4001ZDF

### **SUPPORTED DEVICES**

**Z86E40**

### **DESCRIPTION**

The Z86E40 OTP Program Converter Kit converts a 44-pin QFP package to a 40-pin DIP package, which allows the C50 ICE BOX™ to program the 44-pin QFP Z86E40 OTP microcontroller.

### **SPECIFICATIONS**

#### **Power Requirements**

Not applicable.

#### **Dimensions**

Width: 2.05 in. (5.2 cm)

Length: 2.10 in. (5.3 cm)

# Z86E4001ZDV

## PRODUCT SPECIFICATION



Photograph  
Not Available  
At This Time

### SUPPORTED DEVICES

**Z86E40**

### DESCRIPTION

The Z86E40 OTP Program Converter Kit converts a 44-pin PLCC package to a 40-pin DIP package, which allows the C50 ICE BOX™ to program the 44-pin PLCC Z86E40 OTP microcontroller.

### SPECIFICATIONS

#### Power Requirements

Not applicable.

#### Dimensions

Width: 1.8 in. (4.6 cm)

Length: 2.1 in. (5.3 cm)

### KIT CONTENTS

#### Z86E40 OTP Program Converter Board

40-Pin PLCC ZIF Socket

40-Pin Connector

#### Documentation

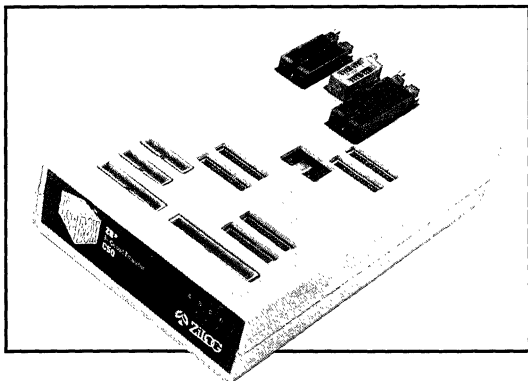
Not applicable.

### ORDERING INFORMATION

**Part No:** Z86E4001ZDV

# Z86C5000ZEM

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z86C06, Z86C09/19, Z86E09, Z86C30,  
Z86E30, Z86C40, Z86E40, Z86C89, Z86C90**

### DESCRIPTION

The Z86C5000ZEM is a member of Zilog's ICE BOX™ product family of in-circuit emulators. The ICE BOX -C50 provides emulation and OTP programming support for Zilog's CCP™ (Consumer Controller Processor) microcontroller. The Emulator provides all the essential MCU timing and I/O circuitry which simplifies user emulation of the prototype hardware/software product. The Emulator can be connected to a serial port COM 1 or COM 2 of the host computer (IBM XT, AT compatible).

### SPECIFICATIONS

#### Emulation Specification

Maximum Emulation Speed 16 MHz

#### Power Requirements

+5 Vdc @ 1.0 A

#### Dimensions

Width: 6.0 in. (15.2 cm)

Length: 8.8 in. (22.4 cm)

#### Serial Interface

RS-232C @ 19200 baud

### KIT CONTENTS

#### Z86C50 Emulator

Z8® Emulation Base Board  
CMOS Z86C9120PSC  
8K x 8 EPROM (Programmed  
with Debug Monitor)  
32K x 8 Static RAM  
3 64K x 4 Static RAMs  
RS-232C Interface  
Reset Switch

#### Z86C50 Emulation Daughter Board

20 MHz CMOS Z86C5020GSE ICE Chip  
2K x 8 Static RAM  
40/28/18 Pin ZIF OTP Sockets  
6 HP-16500A Logic Analysis  
System Interface Connectors  
80-, 60-, 40-Pin Target Connectors

#### Components

Z86E4012PSC  
Z86E3012PSC

#### Cables

12", 40-Pin DIP Emulation Cable  
12", 28-Pin DIP Emulation Cable  
12", 18-Pin DIP Emulation Cable  
48" Power Cable  
15" Power Cable with Banana Plugs  
60" DB 25 RS-232C Cable

#### Software (IBM PC Platform)

Z8/Z80®/Z8000® Cross Assembler  
MOBJ Link/Loader  
Host Package

#### Documentation

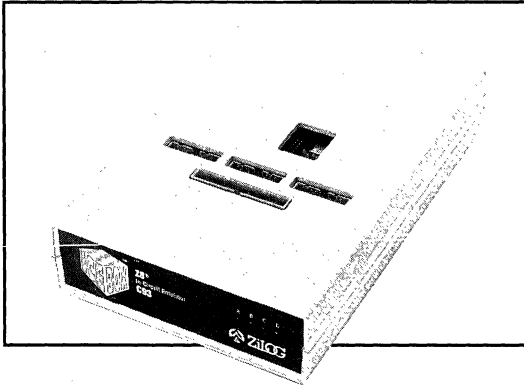
Emulator User Guide  
Z8 Cross Assembler User Guide  
MOBJ Link/Loader User Guide  
Registration Card

### ORDERING INFORMATION

**Part No:** Z86C5000ZEM

# Z86C9300ZEM

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

#### Z86C93

### DESCRIPTION

The Z86C9300ZEM is a member of Zilog's ICE BOX™ product family of in-circuit emulators. The ICE BOX -C93 provides emulation for Zilog's Z86C93 microcontroller. This includes all the essential MCU timing and I/O circuitry which simplifies user emulation of the prototype hardware/software product. The Emulator can be connected to a serial port COM 1 or COM 2 of the host computer (IBM® XT, AT, 386, 486 compatible).

### SPECIFICATIONS

#### Emulation Specification

Maximum Emulation Speed 16 MHz

#### Power Requirements

+5 Vdc @ .5A

#### Dimensions

Width: 6.0 in. (15.2 cm)

Length: 8.8 in. (22.4 cm)

#### Serial Interface

RS-232C @ 19200 baud

### KIT CONTENTS

#### Z86C93 Emulator

Z8® Emulation Base Board

CMOS Z86C9120PSC

8K x 8 EPROM

(Programmed with Debug Monitor)

32K x 8 Static RAM

3 64K x 4 Static RAMs

RS-232C Interface

Reset Switch

Z86C93 Emulation Daughter Board

20 MHz CMOS Z86C9320GSE ICE Chip

3 HP-16500A Logic Analysis

System Interface Connectors

80-Pin Target Connector

#### Cables

12" Z86C93 44-Pin PLCC Emulation Pod

48" Power Cable

15" Power Cable with Banana Plugs

60" DB 25 RS-232C Cable

#### Software (IBM PC Platform)

Z8/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Host Package

#### Documentation

Emulator User Guide

Z8 Cross Assembler User Guide

MOBJ Link/Loader User Guide

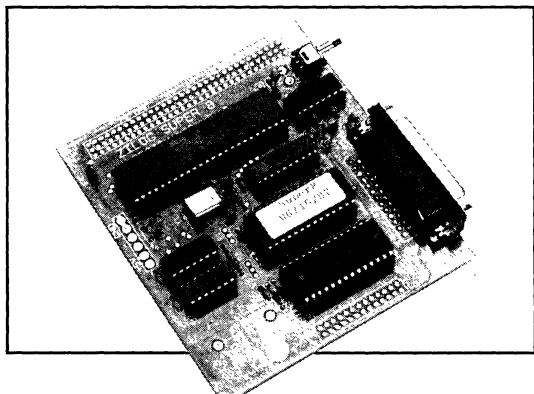
Registration Card

### ORDERING INFORMATION

Part No: Z86C9300ZEM

# Z0880000ZCO

## PRODUCT SPECIFICATION



### **KIT CONTENTS**

#### **Super8 Development Board**

NMOS Z08800 Super8 MPU  
20 MHz Crystal  
(32K)/8K x 8 EPROM  
(32K)/8K x 8 Static RAM  
RS-232C PC Interface

#### **Software (IBM PC Platform)**

Z8®/Super8 Assembler/Utilities  
Host Communication Package  
Monitor Instructions  
Tutorial  
Sample Z08802 Application Software

#### **Documentation**

Microcontrollers Data Book  
Super8 Development Kit User Guide  
Z8 Cross Assembler User Guide  
MOBJ Link/Loader User Guide

### **ORDERING INFORMATION**

**Part No:** Z0880000ZCO

### **SUPPORTED DEVICES**

**Z08800**

### **DESCRIPTION**

The Super8™ Development Kit can be used for several purposes. As an evaluation tool, the user can learn the Super8's instruction set plus the manipulation of the Super8's interrupt vectors and register set. Secondly, the Super8 Development Kit is designed to aid the user in constructing specific applications using the Super8 microcontroller. Lastly, application prototypes may be run using the Super8 Development Kit.

### **SPECIFICATIONS**

#### **Power Requirements**

+5 Vdc @ .4 A

#### **Dimensions**

Width: 4.0 in. (10.2 cm)

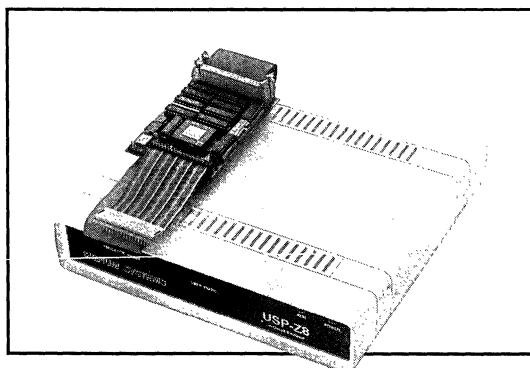
Length: 4.5 in. (11.4 cm)

#### **Serial Interface**

RS-232C @ 9600 baud

# Z8 S Series Emulators

## BASE UNITS AND PODS



### DESCRIPTION

The system comprises three base unit options, (64K, 128K, or 256K of emulation program ROM), and four pod options which allow the emulation of various Z8 microcontrollers. Features include real-time transparent emulation up to 20 MHz, in-line symbolic assembler and disassembler, real-time hardware breakpoints, eight channel user logic analyser, external trigger input and outputs, trace display and memory display/edit during execution, and window or command driven user interface.

### SPECIFICATIONS

#### Microcontrollers Emulated:

Z86C1200ZPD	Z86C00, Z86C10, Z86C20, Z86C11, Z86C21, Z86E21, Z86C91, Z86C61
Z86C5000ZPD	Z86C09, Z86C19, Z86C30, Z86C40, Z86C90
Z86C9300ZPD	Z86C93
Z86C9400ZPD	Z86C94
Z86C9500ZPD	Z86C95

#### Maximum Emulation Speed:

Up to 30 MHz (microcontroller dependent)

#### Size:

260 mm wide, 260 mm deep, 64 mm high

#### Operating Temperature:

0°C to +40°C

#### Storage Temperature:

-10°C to +65°C

#### Operating Humidity:

0 to 90%

#### Maximum Emulation Program Memory:

64 Kbytes with Z86C0000ZUSP064  
128 Kbytes with Z86C0000ZUSSP128  
256 Kbytes with Z86C0000ZUSP256

#### Maximum Emulation Data Memory:

64 Kbytes

#### Program Memory Mapping:

1K blocks

#### Pass Counters:

Two, 16-bit each

#### Trace Buffer:

32K - 80 bits

#### Sequencer:

Hardware, 8 levels

#### User Probe:

Eight channel logic input  
One trigger input  
Seven trigger outputs (Events, Pass Counters, Sequencer)

#### Host Interface:

Asynchronous RS-232C  
9600/115 KBaud  
XON/XOFF support

#### File Upward/Downward Format:

Zilog MUFOM (EEE 695-1985)  
Intel® HEX  
Intel AOMF  
2500AD® Software

---

## MINIMUM HOST REQUIREMENTS

- IBM® compatible PC/XT/AT/386 or PS-2
- 640 Kbyte memory
- 20 Mbyte hard disk
- RS-232 serial port (COM 1 or COM 2)
- Mouse (serial or bus)
- MDA, CGA, EGA, or VGA video adaptor

## MINIMUM EMULATION SUPPORT

- One base unit
- One emulation pod

### Part Numbers

#### Base Unit

Z86C0000ZUSP064  
Z86C0000ZUSP128  
Z86C0000ZUSP256  
Z86C9400ZUSP064  
Z86C9500ZUSP064

#### Emulation Pod

Z86C9300ZPD  
Z86C1200ZPD  
Z86C5000ZPD  
Z86C9400ZPD  
Z86C9500ZDP





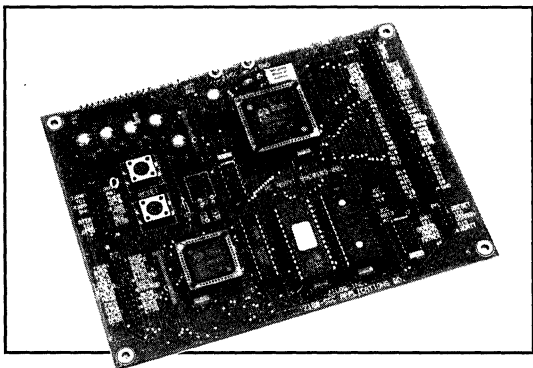
# **Intelligent Peripherals**

## Kit-To-Part Cross Reference Matrix

	Z801800ZCO	Z8018100ZCO	Z8018100ZDP	Z84C1100ZCO	Z84C1500ZCO	Z84C5000ZCO	Z84C9000ZCO	ZEPMIP00001
Z80180	●							●
Z80181		●	●					●
Z84C01						●		
Z84C11				●				
Z84C15					●			
Z84C43				●				
Z84C50						●		
Z84C90						●	●	
Z85C30	●							

# Z8018000ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

Z80180, Z85C30

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the Z80180 and Z85C30 devices.

The supplied cross assembler and link/loader package allows full assembly language programming support. A board resident debug monitor program allows executable code to be down-loaded and subsequently debugged.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .65 A

#### Dimensions

Width: 6.5 in.

Length: 5.0 in.

#### Serial Interface

RS-232C @ 9600 baud

### KIT CONTENTS

#### Z180/SCC Evaluation Board

CMOS Z80180 MPU  
19.6608 MHz Crystal  
Socketed (32K)/8K x 8 EPROM  
(programmed with Debug Monitor)  
Socketed (32K)/8K x 8 Static RAM  
RS-232C PC Interface  
Z80180 Expansion Header  
Z85C30 Expansion Header  
Reset switch  
Nmi switch

#### Cables

25-Pin RS-232 Cable

#### Software (IBM® PC Platform)

Z8®/Z80®/Z8000® Cross Assembler  
MOBJ Link/Loader  
Resident Debug Monitor Source Code  
Z80180 Example Software

#### Documentation

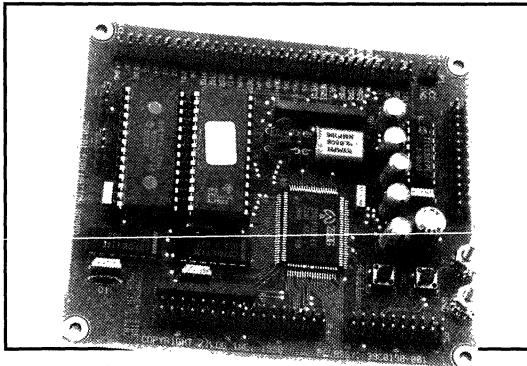
Z180/SCC Kit User Guide  
Z80180 Technical Manual  
Z8030/8530 Technical Manual  
Z80 Cross Assembler User Guide  
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z8018000ZCO

# Z8018100ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z80181**

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the Z80181 device.

The supplied cross assembler and link/loader package allows full assembly language support. A board resident debug monitor program allows executable code to be down-loaded and subsequently debugged.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .65 A

#### Dimensions

Width: 3.65 in. (9.27 cm)

Length: 4.20 in. (10.67 cm)

#### Serial Interface

RS-232C @ 9600 bits/sec.

### KIT CONTENTS

#### Z180181 Evaluation Board

CMOS Z80181 MPU

19.6608 MHz Crystal

Socketed (32K) /8K x 8 EPROM

(programmed with Debug Monitor)

Socketed (32K) /8K x 8 Static RAM

RS-232C Interface

Z80181 MPU Expansion Header

Z80181 Peripheral Signal

Expansion Header

Reset switch

Nmi switch

#### Cables

25-Pin RS-232 Cable

#### Software (IBM® PC Platform)

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Resident Debug Monitor Source Code

Z80181 Example Software

#### Documentation

Z80181 Kit User Guide

Z80181 Product Specification

Z8030/8530 Technical Manual

Z80 Cross Assembler User Guide

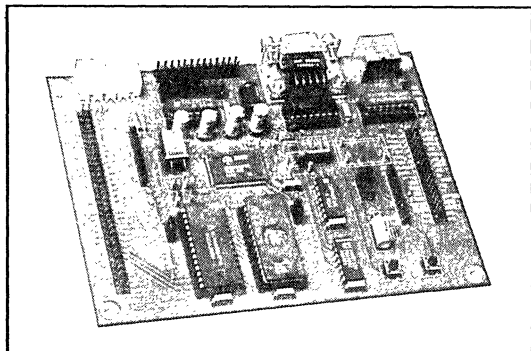
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z8018100ZCO

# Z8018101ZCO

## PRODUCT SPECIFICATION



### DESCRIPTION

The Z181™ MPU LLAP Evaluation Kit contains assembled Z181 LLAP Evaluation Boards, software and documentation to support AppleTalk® LocalTalk™ Link Access Protocol (LLAP) on Zilog's Z80181 (Z181).

The purpose of the Z181 LLAP Evaluation Board is to demonstrate one possible hardware configuration when implementing LLAP on the Z181 and to provide a platform from which to execute an example LLAP application program. This example LLAP application program (mainecho.c) allows the board to transmit and receive LLAP packets to another board.

### Z181 LLAP Driver Description

The kit contains source codes and documentation for the Z181 LLAP driver\*. The "User Guide, LLAP Driver for the Z181" describes the driver for the Z181 and explains how the driver is to be used. The "Design, LLAP Driver for the Z181" explains how the driver works.

In general, the Z181 LLAP Driver implements the LLAP protocol described in "Inside AppleTalk" by Sidhu et.al. e.g., proper ENQ, ACK, RTS, CTS transmission/reception.

Note: \*Licensing agreement for this software is required. Please contact Zilog's sales office for further information.

### SPECIFICATIONS FOR LLAP EVALUATION BOARD

#### Power Requirements

+5 Vdc @ 0.50A

#### Dimensions

Width: 4.4 in.

Length: 5.8 in.

#### Serial Interfaces

RS-232C

RS-422 LocalTalk DIN-8 and DB-9 interface

### LLAP Driver Resource Usage and Hardware Requirements

Approximately 4.5 Kbytes of program memory (ROM or RAM) written partly in Z181 assembler and in ANSI C (Microtec).

Approximately 128 bytes of data memory (RAM).

TxD and RxD of the Z181's SCC is connected to the RS-422 differential drivers.

The /REQ pin from the Z181's SCC is connected to the Z181 DMA's /DREQ1.

A 3.6864 MHz crystal is attached to the RTxC and SYNC pins for LLAP clocking.

The Z181 uses a 10.0 MHz clock.

Memory accesses require no added wait states.



---

## **KIT CONTENTS**

Two Z8018101ZCO LLAP Evaluation Boards, each has

- CMOS Z181™ MPU @ 10 MHz
- 20 MHz Crystal (for MPU)
- 3.684 MHz Crystal (for LLAP)
- Socketed 8Kx8 EPROM (containing mainecho.c)
- Socketed 8Kx8 Static RAM
- RS-232C Interface
- RS-422 Interface for LocalTalk™ DIN-8 or DB-9 connection
- Z181 MPU Expansion Header
- Z181 Peripheral Expansion Header
- Reset switch
- NMI switch

## **Software**

Z181 LLAP Driver Source Code Diskette\*

## **Hardware**

Two unsocketed 8Kx8 EPROM containing demo.c  
Two LocalTalk DB-9 connection modules  
Two power supply cables  
Two DB-25 connectors

## **Documentation**

User Guide, LLAP Driver for the Z181 Design, LLAP Driver for the Z181 Z181 Product Specification  
Z85C30 Product Specification

## **ORDERING INFORMATION**

**Part No:** Z8018101ZCO

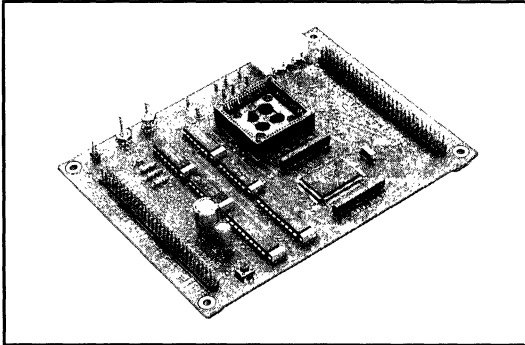
Note: \*Licensing agreement for this software is required. Please contact Zilog's Sales Office for further information.

## **Trademarks**

AppleTalk® A Registered Trademark of Apple Computer, Inc.  
LocalTalk™ A Trademark of Apple Computer, Inc.

# Z8018100ZDP

## PRODUCT SPECIFICATION



### Z80181 Adapter Board

68-Pin PLCC Z180™ socket

Power Supply cable

### Documentation

Z80181 Emulator Adapter Kit User Guide

## ORDERING INFORMATION

**Part No:** Z8018100ZDP

## SUPPORTED DEVICES

Z80181

## DESCRIPTION

The Z80181 Emulator Adapter enables development and debugging of user target systems which utilize the Z80181 chip plus the use of a Z80180 In-Circuit Emulator that provides a 68-pin PLCC probe. It serves as a bridge between a Z80180 In-Circuit Emulator and a user target system.

## SPECIFICATIONS

### Power Requirements

$V_{CC} = 5.0V$

### Dimensions

Width: 4.10 in.

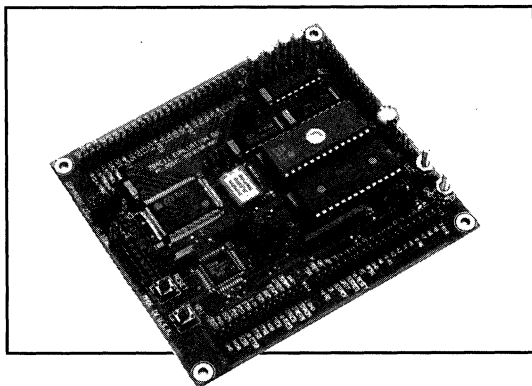
Length: 5.40 in.

## KIT CONTENTS



# Z84C1100ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

Z84C11, Z84C43

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the Z84C11 and Z84C43 devices.

The supplied cross assembler and link/loader package allows full assembly language programming support. A board resident debug monitor program allows executable code to be down-loaded and subsequently debugged.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .36 A

#### Dimensions

Width: 4.5 in. (8.9 cm)

Length: 4.0 in. (10.2 cm)

#### Serial Interface

RS-232C @ 9600 baud

### KIT CONTENTS

#### Z84C11 Evaluation Board

CMOS Z84C11 MPU

CMOS Z84C43 SIO

19.6608 Mhz Crystal

Socketed (32K)/8K x 8 EPROM

(supplied with Debug Monitor EPROM)

Socketed (32K)/8K x 8 Static RAM

RS-232C PC Interface

Z80 signal Expansion Header

Z84C11 I/O signal Expansion Header

Reset switch

Nmi switch

#### Cables

25-Pin RS-232 Cable

#### Software (IBM® PC Platform)

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Resident Debug Monitor Source Code

#### Documentation

Z84C11 Kit User Guide

Z84C11 Product Specification

Z80 Cross Assembler User Guide

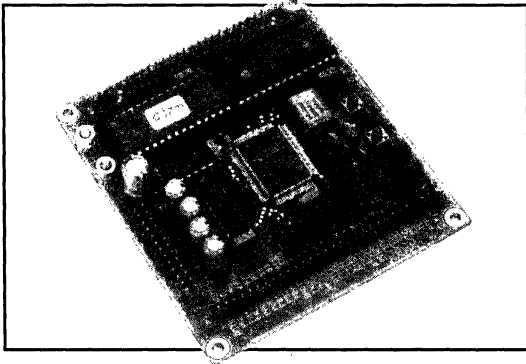
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z84C1100ZCO

# Z84C1500ZCO

## PRODUCT SPECIFICATION



### KIT CONTENTS

#### Z84C15 Evaluation Board

CMOS Z84C15 MPU  
19.6608 MHz Crystal  
Socketed (32K)/8K x 8 EPROM  
(programmed with Debug Monitor)  
Socketed (32K)/8K x 8 Static RAM  
RS-232C PC Interface  
Z80<sup>®</sup> MPU Signal Expansion Header  
Z84C15 I/O Expansion Header  
Reset switch  
Nmi switch

#### Cables

25-Pin RS-232 Cable

#### Software (IBM<sup>®</sup> PC Platform)

Z8<sup>®</sup>/Z80/Z8000<sup>®</sup> Cross Assembler  
MOBJ Link/Loader  
Resident Debug Monitor Source Code

#### Documentation

Z84C15 Kit User Guide  
Z84C15 Product Specification  
Z80 Cross Assembler User Guide  
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z84C1500ZCO

### SUPPORTED DEVICES

Z84C15

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the Z84C15 device.

The supplied cross assembler and link/loader package allows full assembly language programming support. A board resident debug monitor program allows executable code to be down-loaded and subsequently debugged.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .39 A

#### Dimensions

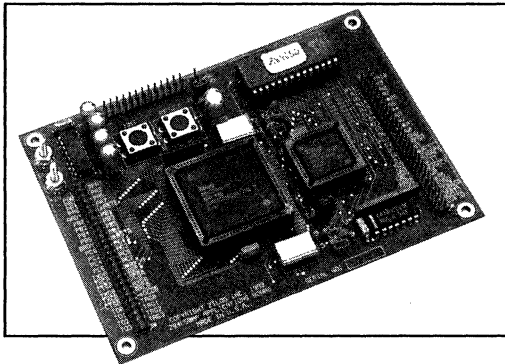
Width: 3.9 in. (8.9 cm)  
Length: 3.5 in. (10.2 cm)

#### Serial Interface

RS-232C @ 9600 baud

# Z84C5000ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z84C50, Z84C90**

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the Z84C50 and Z84C90 devices.

The supplied cross assembler and link/loader package allows full assembly language programming support. A board resident debug monitor program allows executable code to be down-loaded and subsequently debugged.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .47 A

#### Dimensions

Width: 5.5 in. (8.9 cm)

Length: 4.0 in. (10.2 cm)

#### Serial Interface

RS-232C @ 9600 baud

### KIT CONTENTS

#### Z84C50 Evaluation Board

CMOS Z84C50 MPU

CMOS Z84C90 KIO™ IC

8 MHz Crystal

(32K)/8K x 8 EPROM

(programmed with Debug Monitor)

RS-232C PC Interface

Z84C90 Signal Expansion Header

Z84C50 Signal Expansion Header

Reset switch

Nmi switch

#### Cables

25-Pin RS-232 Cable

#### Software (IBM® PC Platform)

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Resident Debug Monitor Source Code

Example Software

#### Documentation

Z84C50 Kit User Guide

Z84C50 Product Specification

Z80 Cross Assembler User Guide

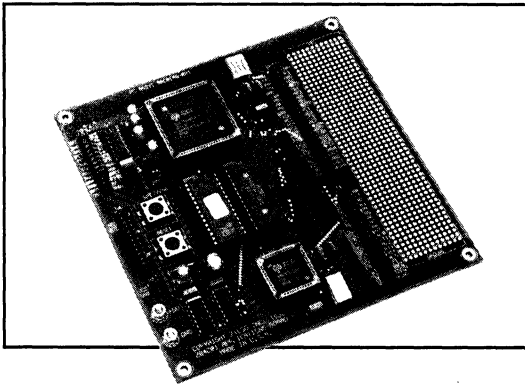
MOBJ Link/Loader User Guide

### ORDERING INFORMATION

**Part No:** Z84C5000ZCO

# Z84C9000ZCO

## PRODUCT SPECIFICATION



## SUPPORTED DEVICES

**Z84C01, Z84C90**

## DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software and hardware development for the Z84C01 MPU and Z84C90 KIO™ devices.

The supplied cross assembler and link/loader package allows full assembly language programming support. A board resident debug monitor program allows executable code to be down-loaded and subsequently debugged.

Code targeted for the Z84C01 and/or Z84C90 device may be verified for the target application before prototype build.

## SPECIFICATIONS

### Power Requirements

+5 Vdc @ .45 A

### Dimensions

Width: 6.0 in. (8.9 cm)

Length: 6.0 in. (10.2 cm)

### Serial Interface

RS-232C @ 9600 baud

## KIT CONTENTS

### Z84C01 Evaluation Board

CMOS Z84C01 MPU

CMOS Z84C90 KIO™ IC

8 MHz Crystal

Socketed (32K)/8K x 8 EPROM

(programmed with Debug Monitor)

Socketed (32K)/8K x 8 Static RAM

RS-232C PC Interface

Z84C90 Signal Expansion Header

Z84C01 Signal Expansion Header

Bread Board Area

Reset switch

Nmi switch

### Cables

25-Pin RS-232 Cable

### Software (IBM® PC Platform)

Z8®/Z80®/Z8000® Cross Assembler

MOBJ Link/Loader

Resident Debug Monitor Source Code

Example Software

### Documentation

Z84C90 Kit User Guide

Intelligent Peripheral Controllers Data Book

Z80 Cross Assembler User Guide

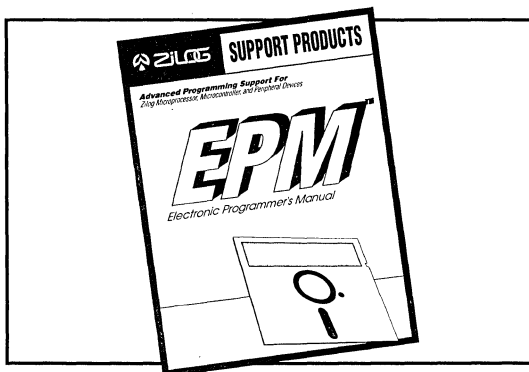
MOBJ Link/Loader User Guide

## ORDERING INFORMATION

**Part No:** Z4C9000ZCO

# ZEPMIP00001

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z80180/Z180, Z80181/Z181**

### DESCRIPTION

The EPM™ Electronic Programmer's Manual provides on-line documentation on Zilog's Microprocessor Unit Z80180/Z180 and Z80181/Z181 register sets and device operation. Its code generation features make it a most valuable tool for the programmer. The EPM Manual helps you set the registers to ensure that the device operates with your specified settings. Once you have selected the field values as a series of C function calls, or as an assembler table, you can include this output in any software that utilizes the device.

### SPECIFICATIONS

#### Minimum Hardware Requirements

IBM® PC/AT with available 512K RAM  
5.25 inch, high density, or 3.5 inch,  
high density floppy disk drive  
Hard disk drive  
Color monitor

#### Minimum Operating System

MS-DOS, version 3.0 or later

### KIT CONTENTS

#### Software (IBM® PC Platform)

2 EPM Floppy Diskettes: 5.25 inch,  
high density and 3.5 inch, high density

#### Documentation

EPM User's Guide  
Z80180/Z180 Technical Manual  
Z80181/Z181 Preliminary Product Spec.  
Z8030/Z8530 SCC Technical Manual  
EPM Registration Reply Card

### ORDERING INFORMATION

**Part No:** ZEPMIP00001

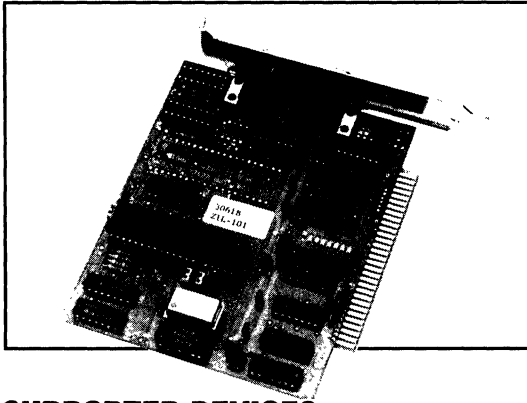
## **Datacom ICs**

## Kit-To-Part Cross Reference Matrix

	Z8018000ZCO	Z8018100ZCO	Z85C3000ZCO	Z16C0100ZCO	Z16C3001ZCO	Z8018600ZCO	ZEPMDC00001	ZEPMDC00002
Z16C01				●				
Z16C20				●				
Z16C30				●	●	●	●	
Z16C31						●	●	
Z16C32						●	●	
Z16C33					●	●	●	
Z16C35						●		●
Z8530			●			●		●
Z85C30	●	●	●			●		●
Z85230			●			●		●
Z85233			●			●		●
Z8030						●		●
Z80C30						●		●
Z80230						●		●

# Z8523000ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z8530, Z85C30, Z85230, Z85233**

### DESCRIPTION

The kit contains an assembled PC/XT/AT circuit board with one high speed serial port, selectively driven by RS-232C or RS-422 line drivers. The kit also contains software and documentation to support software and hardware development for Zilog's ESCC™ device.

The board illustrates the use of Zilog's ESCC in a variety of communication applications such as SDLC/HDLC, and high speed ASYNC.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .5 A

#### Dimensions

Width: 4 in. (10.16 cm)

Length: 5 in. (12.70 cm)

#### Serial Interface

A DB25 port selectively driven by RS-232C or RS-422 at selectable baud rates.

### KIT CONTENTS

#### Z85230 Evaluation Board

CMOS Z85230 ESCC

RS-232C and RS-422 line drivers

DB25 connector

#### Software (IBM® PC Platform)

Source and executable codes to run the ESCC in SDLC/HDLC and ASYNC modes using DMA, Interrupt and polling methods. All codes are written in C and compiled using the Microsoft® Quick C compiler.

#### Documentation

Z85230 Product Specifications

Z85230 Technical Manuals

Z8523000ZCO User Guide

Sealevel™ User's Manual

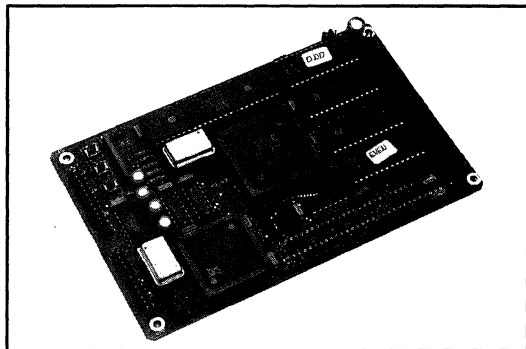
### ORDERING INFORMATION

**Part No:** Z8523000ZCO



# Z16C0100ZCO

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

Z16C01, Z16C20, Z16C30

### DESCRIPTION

The kit contains an assembled circuit board, software and documentation to support software, and hardware development for the Z16C01 CPU, Z16C20 GLU (General Logic Unit) and Z16C30 USC™ (Universal Serial Controller).

The supplied cross C compiler, assembler and link/loader package allows full C and assembly language programming support. A board resident debug monitor program and its PC based counterpart allow executable code to be down-loaded and subsequently debugged.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .5 A

#### Dimensions

Single Euro Card Format  
Width: 3.94 in. (10 cm)  
Length: 6.30 in. (16 cm)

#### Serial Interface

RS-232C @ 9600 baud

### KIT CONTENTS

Z16C01/20/30 Evaluation Board  
CMOS Z16C01 CPU  
CMOS Z16C20 GLU  
CMOS Z16C30 USC  
20 MHz Crystal  
20 MHz Oscillator  
Two (64K)/8K x 8 EPROMs  
(programmed with Debug Monitor)  
Two (32K)/(8K) x 8 Static RAM  
RS-232C PC Interface  
Z16C01 Expansion Bus Connector

#### Cables

25-Pin RS-232C Cable

#### Software (IBM® PC Platform)

Z8000® Cross C Compiler  
Z8®/Z80®/Z8000 Cross Assembler  
MOBJ Link/Loader  
Resident Debug Monitor Source Code  
Host Package Source Code  
Z16C20 Example Software

#### Documentation

Z8000 CPU Technical Manual  
Z8000 CPU Programmer's Pocket Guide  
Z16C20 GLU Product Specification  
Z16C30 USC Technical Manual  
Z16C0100ZCO Kit User Manual  
CC8K C Compiler User Guide  
Z8000 Cross Assembler User Guide  
MOBJ Link/Loader User Guide

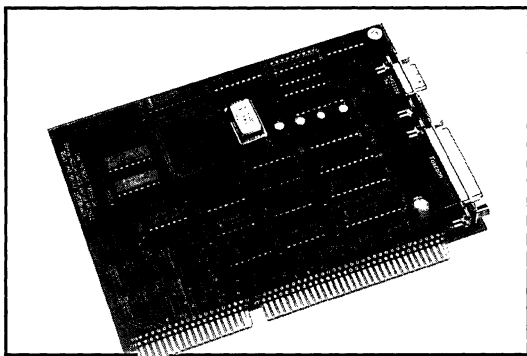
### ORDERING INFORMATION

Part No: Z16C0100ZCO

- \$250 sealed

# Z16C3001ZCO

## PRODUCT SPECIFICATION



### KIT CONTENTS

#### Z16C30/Z16C33 Evaluation Board

CMOS Z16C30 USC and Z16C33 MUSC  
RS-232C and RS-422 line drivers  
DB9 and DB25 Interfaces

#### Software (IBM® PC Platform)

Source and executable codes to run the USC or MUSC in SDLC/HDLC or ASYNC mode. All codes are written and compiled using Microsoft® Quick C 2.5.

#### Documentation

Z16C30 and Z16C33 Product Specifications  
Z16C30/Z16C33 Technical Manual  
Z16C3001ZCO Kit User Guide

### ORDERING INFORMATION

**Part No:** Z16C3001ZCO

### SUPPORTED DEVICES

**Z16C30, Z16C33**

### DESCRIPTION

The kit contains an assembled PC/XT/AT circuit board with two high-speed serial connections, DB9 and DB25 connectors selectively driven by RS-232 or RS-422 line drivers. The kit also contains software and documentation to support software and hardware development for Zilog's USC™ and MUSC™ devices.

The board illustrates the use of Zilog's USC and MUSC devices in a variety of communication applications such as ASYNC, SDLC/HDLC and high-speed ASYNC.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .5 A

#### Dimensions

Width: 4.5 in. (11.43 cm)

Length: 6.5 in. (16.51 cm)

#### Serial Interface

DB9 and DB25 connectors selectively driven by RS-232C or RS-422 at selectable baud rates.

# Z8018600ZCO

## PRODUCT SPECIFICATION



Photograph  
Not Available  
At This Time

### SUPPORTED DEVICES

**Z8X30, Z85230, Z16C31, Z16C32, Z8XC30,  
Z16C33, Z16C30, Z16C35**

### DESCRIPTION

The kit contains an assembled circuit board, software, and documentation to support the evaluation and development of code for Zilog's Z85C30 SCC, Z85230 ESCC™, Z16C30 USC™, Z16C33 MUSC™, and the Z16C35 ISCC™. The purpose of the board is to illustrate how the datacom family interfaces and communicates with the 80186 CPU. This will help potential customers evaluate Zilog's data communication controllers in an Intel® environment. A board-resident monitor program allows code to be downloaded and executed.

### SPECIFICATIONS

#### Power Requirements

+5 Vdc @ .50A

#### Dimensions

Width: 8.4 in. (21.34 cm)

Length: 9.3 in. (23.62 cm)

#### Serial Interfaces

RS-232C, RS-422

### KIT CONTENTS

#### Z8018600ZCO Evaluation Board

Intel 80186 Integrated 16-bit MPU @ 16 MHz  
CMOS Z85230 ESCC  
CMOS Z16C30 USC  
CMOS Z16C32 IUSC  
CMOS Z16C35 ISCC  
2 (64K) 8Kx8 EPROMs  
6 (256K) 32Kx8 SRAMs  
RS-232C, RS-422, and Apple® LocalTalk™  
line drivers  
DB9, DB25, and DIN 8 Interfaces

#### Cables

1 25-pin RS-232C Cable  
14 Jumper Wires

#### Software (IBM® PC Platform)

Resident Monitor for download and  
execution (80186 Assembler source code)  
PC-board terminal emulator  
Z85230, Z16C30/33, Z16C31, and  
Z16C35 Examples Software (All codes  
written in "C" and compiled using  
the Microtec® C compiler.)

#### Documentation

Z85230 ESCC Product Specification/  
Technical Manual  
Z16C30/33 (M)USC Product Specification/  
Technical Manual  
Z16C35 ISCC Product Specification/  
Technical Manual  
Datacom Evaluation Board Application Note

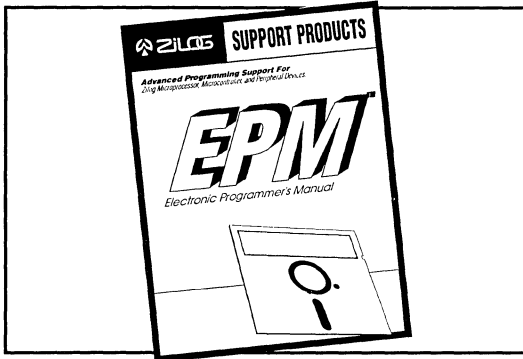
### ORDERING INFORMATION

**Part No:** Z8018600ZCO

*Zilog - Sealed  
\$580*

# ZEPMDC00001

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z16C30, Z16C33, Z16C31**

### DESCRIPTION

The EPM™ Electronic Programmer's Manual provides on-line documentation on Zilog's USC™ Universal Serial Communications Controller (Z16C30), MUSC™ Mono-Universal Serial Controller (Z16C33), and the IUSC™ Integrated Universal Serial Controller (Z16C31) register sets and device operation. Its code generation features make it a most valuable tool for the programmer. The EPM helps you set the registers to ensure that the device operates with your specified settings. Once you have selected the field values as a series of C function calls, or as an assembler table, you can include this output in any software that utilizes the device.

### SPECIFICATIONS

#### Minimum Hardware Requirements

IBM PC/AT with available 512K RAM  
5.25 inch, high density, or 3.5 inch,  
high density floppy disk drive  
Hard disk drive  
Color monitor

#### Minimum Operating System

MS-DOS, version 3.0 or later

### KIT CONTENTS

#### Software (IBM® PC Platform)

2 EPM Floppy Diskettes: 5.25 inch,  
high density and 3.5 inch, high density

#### Documentation

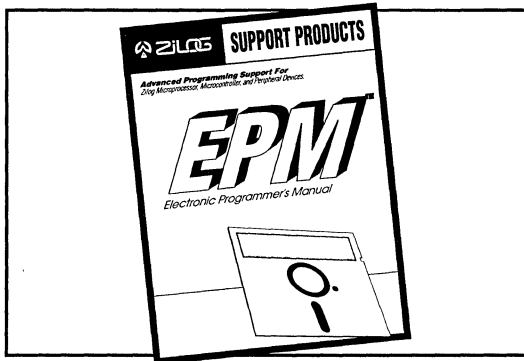
EPM User's Guide  
USC/MUSC Technical Manual  
IUSC Technical Manual  
EPM Registration Reply Card

### ORDERING INFORMATION

**Part No:** ZEPMDC00001

# ZEPMDC00002

## PRODUCT SPECIFICATION



### SUPPORTED DEVICES

**Z8X30, Z8XC30, Z8X230, Z16C35**

### DESCRIPTION

The EPM™ Electronic Programmer's Manual provides on-line documentation on Zilog's Serial Communications Controller family of devices (Z08X30 NMOS SCC, Z8XC30 CMOS SCC, Z8X230 ESCC, Z16C35 ISCC™ controller): register set and operation of the device. Its code generation features make it a most valuable tool for the programmer. The EPM Manual helps you set the registers to ensure that the device operates with your specified settings. Once you have selected values for the registers, the EPM Manual lets you save the field values as a series of C function calls or as an assembler table. You can include this output in any software that utilizes the device.

### SPECIFICATIONS

#### Minimum Hardware Requirements

IBM® PC/AT with available 512K RAM  
5.25 inch, high density, or 3.5 inch,  
high density floppy disk drive  
Hard disk drive  
Color monitor

#### Minimum Operating System

MS-DOS, version 3.0 or later

### KIT CONTENTS

#### Software (IBM® PC Platform)

2 EPM Floppy Diskette: 5.25 inch,  
high density and 3.5 inch, high density

#### Documentation

EPM User's Guide  
SCC Technical Manual  
ESCC Technical Manual  
ISCC Technical Manual  
EPM Registration Reply Card

### ORDERING INFORMATION

**Part No:** ZEPMDC00002

# Z80 & Z80180 HARDWARE AND SOFTWARE SUPPORT

## Hardware Support

Company	Name	Phone
American Automation	Emulator	(714) 731-1661
Applied Microsystems	Emulator (symbolic debug)	(206) 882-2000
Hewlett-Packard	Emulator pods for HP 64000/UX/PC	(800) 4HP-DATA
Huntsville Microsystems	Emulator	(205) 881-6005
Micromint	SB180,SB180FX, BCC180,RTC180	(800) 635-3355
MicroTek	Z80 Emulator, RS-232 compatible	(213) 321-2121
MicroWorks	Prototyping board	(408) 997-1644
Orion Instruments	Logic Analyzer and ROM emulator	(415) 327-8800
Pentica Systems, Inc.	Emulator	(617) 577-1101
Softaid	Emulator, ICEBOX, ICE Analyzer (symbolic debug)	(800) 433-8812
Sophia Systems	Emulator, SA2000	(415) 493-6700
Versalogic	Z80 STD Bus circuit board	(503) 485-8575
Z-World	IBM PC Development Bd.	(916) 753-3722
Zaxtek	Emulator	(714) 474-1170
Zilog	Z180+SCC (Z8018000ZCO)	*Call Zilog*
Zilog	Application Board Z80181 Eval. Kit (Z8018100ZCO)	*Call Zilog*
Zilog	Z84C11 Eval. Kit (Z84C1100ZCO)	*Call Zilog*
Zilog	Z84C15 Eval. Kit (Z84C1500ZCO)	*Call Zilog*
Zilog	Z84C50+KIO Application Bd. (Z84C5000ZCO)	*Call Zilog*
Zilog	Z84C01+KIO Development Bd. (Z84C9000ZCO)	*Call Zilog*

### 68 PLCC Socket Manufacturers:

Methode, TNB, ITT, Cannon, Precicontact

### 64 Shrink DIP Socket Manufacturers:

TI, Bevar, Yamaichi

### 44/80/100 pin QFP:

ZIF (Zero Insertion Force) sockets for prototyping may be obtained from Yamaichi Electronics, (408) 450-0797

## Assemblers and Cross Assemblers

Company	Host/Comments	Phone
2500AD	IBM PC, CP/M, VAX	(800) 843-8144
Allen Ashley	IBM PC	(818) 793-5748
American Automation	IBM PC	(714) 731-1661
AnyWare Engineering	IBM PC, Macintosh	(303) 442-0556
Avocet Systems	IBM PC	(800) 448-8500
Enertec, Inc.	IBM PC, VAX	(215) 362-0966
Hewlett-Packard	HP 64000/64000UX/ 64000-PC Systems	(800) 4HP-DATA
Huntsville	IBM PC, Sun Micro,	(205) 881-6005
Logisoft	IBM PC	(408) 773-8465
Microsystems	Apollo	
Micro Computer Tools	IBM PC	(415) 825-4200
Micro Dialects, Inc.	Macintosh	(513) 271-9100
Softaid	IBM PC	(800) 433-8812
Softools	IBM PC	(410) 750-3733
Software Develop- ment Systems	Uniware,IBM,VAX	(708) 971-8170
Z-World	Unix/VMS,Sun,Apollo IBM PC	(916) 753-3722

## Simulators/Applications Software

Company	Host/Comments	Phone
Avocet Systems	Simulator/IBM PC	(800) 448-8500
Lear Com Company	Simulator/IBM PC	(303) 232-2226
Logisoft	8080 to Z80 Translator	(408) 773-8465
Micromint	Z-System OS	(800) 635-3355
Softaid	Z80180 Guide, IBM PC diskette	(800) 433-8812
The AG Group	LLAP Dvmnt/Apple	(510) 937-7900
Z-World	Simulator/IBM PC	(916) 753-3722

---

# Z80, Z80180 & Z80280 HARDWARE AND SOFTWARE SUPPORT, MARCH 1992

---

## Z80 & Z80180 High Level Language Compilers

Company	Language Host	Phone
2500AD	C, IBM PC, CP/M, VAX	(800) 843-8144
American Automation Archimedes	C, IBM PC C, IBM PC, Sun, VAX, HP	(714) 731-1661 (415) 567-4010
Avocet Systems Laboratory	C, IBM PC Forth IBM PC Microsystems	(800) 448-8500 (213) 306-7412
Microtek Lab, Inc.	C, Pascal IBM PC, Sun Micro, VAX	(213) 321-2121
MPE	Forth IBM PC	(716) 461-9187
Softaid	MT-Basic IBM PC	(800) 433-8812
Software Development Systems	C, IBM PC, VAX, Sun, Apollo	(708) 971-8170
Z-World	Dynamic C, IBM PC	(916) 753-3722

**Note:**

Z80/64180 software is also compatible with the Z80180.

## Z80280 Hardware Support

Company	System Name	Phone
Computer Design Solutions Softaid	STD Buscard & Z80 Dvmnt. Board Z80 ICE Analyzer	(704) 876-2346 (800) 433-8812

**68 PLCC Socket Manufacturers:**

Methods, TNB, ITT, Cannon, Precicontact

## Assemblers and Cross Assemblers:

Company	Host/Comments	Phone
2500AD	IBM PC, CP/M, VAX	(800) 843-8144
Computer Design Solutions	IBM PC	(704) 876-2346

## Simulators

Company	Host/Comments	Phone
Micro Methods, Inc.	IBM PC (ZRPM)	(503) 861-1765

## High Level Language Compilers

Company	Language Host	Phone
2500AD	C, IBM PC, CP/M, VAX	(800) 843-8144
Computer Design Solutions	C, IBM PC	(704) 876-2346

**Note:** Z80 Software is object code compatible with the Z280.

This is NOT a complete list of hardware and software vendors who support Zilog products. Please contact the Zilog sales office nearest you if what you are looking for is not on this list. This list is for reference only and is not an endorsement for any company.

## Communications Software Support

SCC Physical Layer Drivers and Upper Layer Software

Company	Software	Phone
Trillium	SCC Physical Layer X.25, ISDN, Frame Relay Networking Layers.	(310) 479-0500

**Military Qualified Classic/Datacom Products**

Zilog P/N	Description	Speed	Package	883C	SMD P/N	JAN P/N
Z0840002CMB/J	Z80 CPU	2MHz	40-Pin DIP	Qualed		M38510/48002BQA
Z0840004CMB/J	Z80 CPU	4MHz	40-Pin DIP	Qualed		M38510/48001BQA
Z0842002CMB	Z80 PIO	2MHz	40-Pin DIP	Qualed	8418602QA	
Z0842004CMB	Z80 PIO	4MHz	40-Pin DIP	Qualed	8418601QA	
Z0843002CMB	Z80CTC	2MHz	28-Pin DIP	Qualed	8301602XA	
Z0843004CMB	Z80 CTC	4MHz	28-Pin DIP	Qualed	8301601XA	
Z0844002CMB	Z80 SIO	2MHz	40-Pin DIP	Qualed		
Z0844004CMB	Z80 SIO	4MHz	40-Pin DIP	Qualed		
Z0844102CMB	Z80 SIO	2MHz	40-Pin DIP	Qualed		
Z0844104CMB	Z80 SIO	4MHz	40-Pin DIP	Qualed		
Z0844202CMB	Z80 SIO	2MHz	40-Pin DIP	Qualed	8301502QA	
Z0844204CMB	Z80 SIO	4MHz	40-Pin DIP	Qualed	8301501QA	
Z84C0006CMB/J	Z80 CPU	6MHz	40-Pin DIP	Qualed		M38510/48501BQA
Z84C2006CMB	Z80 PIO	6MHz	40-Pin DIP	Qualed	5962-8951401QA	
Z84C3006CMB	Z80 CTC	6MHz	28-Pin DIP	Qualed	5962-8951501XA	
Z84C4006CMB	Z80 SIO	6MHz	40-Pin DIP	Qualed		
Z84C4106CMB	Z80 SIO	6MHz	40-Pin DIP	Qualed		
Z84C4206CMB	Z80 SIO	6MHz	40-Pin DIP	Qualed	5962-8951601QA	
Z84C9008GMB	Z80 KIO	8MHz	84-Pin PGA	Qualed		
Z8018006GMB	Z180 MPU	6MHz	68-Pin PGA	Qualed		
Z8018008GMB	Z180 MPU	8MHz	68-Pin PGA	Qualed		
Z0803004CMB	Z-BUS SCC	4 MHz	40-Pin DIP	Qualed	5962-8551802QA	N/A
Z0803004LMB	Z-BUS SCC	4 MHz	44-Pin LCC	Qualed	5962-8551802YA	N/A
Z0803006CMB	Z-BUS SCC	6 MHz	40-Pin DIP	Qualed	5962-8551801QA	N/A
Z0803006LMB	Z-BUS SCC	6 MHz	44-Pin LCC	Qualed	5962-8551801YA	N/A
Z0853004CMB	Z8530 SCC	4 MHz	40-Pin DIP	Qualed	5962-8752702QA	N/A
Z0853004LMB	Z8530 SCC	4 MHz	44-Pin LCC	Qualed	5962-8752702YA	N/A
Z0853006CMB	Z8530 SCC	6 MHz	40-Pin DIP	Qualed	5962-8752701QA	N/A
Z0853006LMB	Z8530 SCC	6 MHz	44-Pin LCC	Qualed	5962-8752701YA	N/A
Z85C3006CMB/J	Z85C30 CMOS SCC	6 MHz	40-Pin DIP	Qualed	5962-8868901QA	M38510/48601BQA
Z85C3006LMB	Z85C30 CMOS SCC	6 MHz	44-Pin LCC	Qualed	5962-8868901YA	N/A
Z85C3008CMB/J	Z85C30 CMOS SCC	8 MHz	40-Pin DIP	Qualed	5962-8868902QA	M38510/48602BQA
Z85C3008LMB	Z85C30 CMOS SCC	8 MHz	44-Pin LCC	Qualed	5962-8868902YA	N/A
Z85C3010CMB	Z85C30 CMOS SCC	10 MHz	40-Pin DIP	Qualed	5962-8868907QA	N/A
Z85C3010NMB	Z85C30 CMOS SCC	10MHz	44-Pin Cerquad	Q1 '92		N/A
Z85C3010LMB	Z85C30 CMOS SCC	10 MHz	44-Pin LCC	Qualed	5962-8868907YA	N/A
Z16C3010GMB	CMOS USC	10 MHz	68-Pin PGA	Qualed	Q1 '91	N/A
Z8523008CMB	CMOS ESCC	8 MHz	40-Pin DIP	Qualed	Q1 '92	N/A
Z8523008LMB	CMOS ESCC	8 MHz	44-Pin DIP	Qualed	Q1 '92	N/A
Z8523010CMB	CMOS ESCC	10 MHz	40-Pin DIP	Qualed	Q1 '92	N/A
Z8523010LMB	CMOS ESCC	10 MHz	44-Pin LCC	Qualed	Q1 '92	N/A
Z8523016CMB	CMOS ESCC	16.0 MHz	40-Pin DIP	Qualed	Q1 '92	N/A
Z8523016LMB	CMOS ESCC	16.0 MHz	44-Pin LCC	Qualed	Q1 '92	N/A



---



## PACKAGE INFORMATION

### ORDERING CODES

#### PACKAGE

##### PREFERRED

D = Cerdip

P = Plastic DIP

V = Plastic Leaded Chip Carrier

##### LONGER LEAD TIME

A = VQFP (Very Small QFP)

C = Ceramic Sidebrazed

E = Ceramic Window Lid

F = Plastic Quad Flatpack

G = Ceramic PGA (Pin Grid Array)

K = Cerdip Window Lid

L = Ceramic LCC (Leadless Chip Carrier)

N = Cerquad

R = Ceramic Protopak

S = SOIC (Small Outline Integrated Circuit)

T = Low Profile Protopak

#### ENVIRONMENTAL

##### PREFERRED

C = Plastic Standard

E = Hermetic Standard

F = Protopak Standard

##### LONGER LEAD TIME

A = Hermetic Stressed

B = 883 Class B Military

D = Plastic Stressed

J = JAN 38510 Military

#### TEMPERATURE

##### PREFERRED

S = 0°C to +70°C

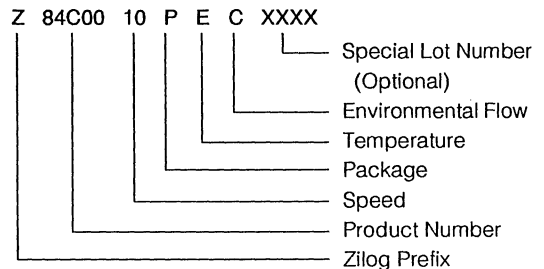
##### LONGER LEAD TIME

E = -40°C to +100°C

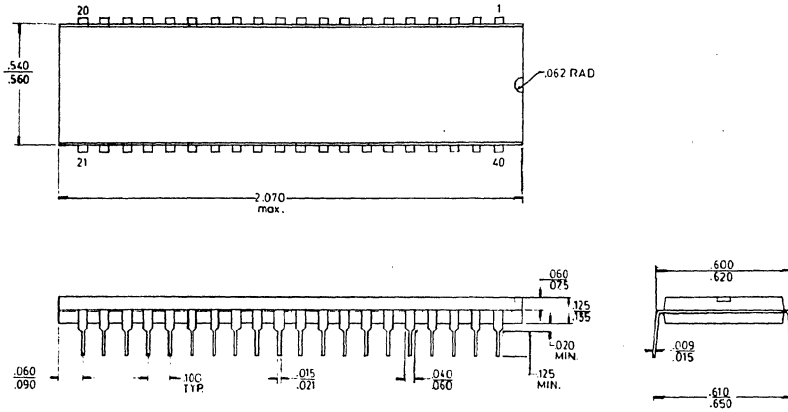
M = -55°C to +125°C

#### EXAMPLE

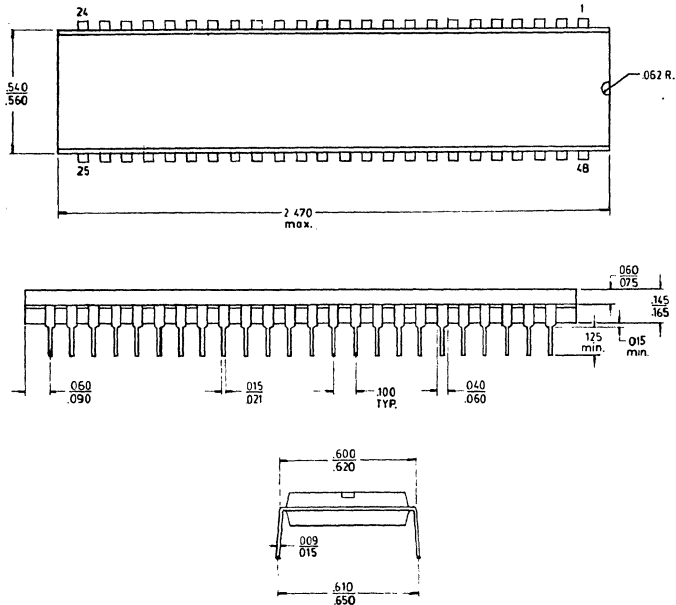
Z84C0010PEC is a CMOS 8400, 10 MHz, Plastic, -40°C to 100°C, Plastic Standard Flow.



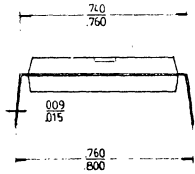
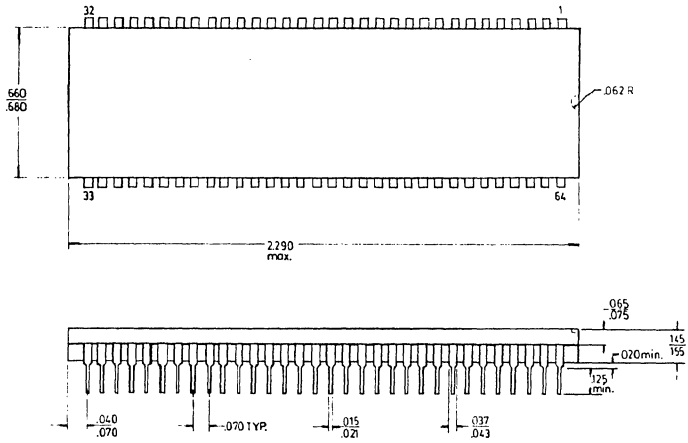
# PACKAGE INFORMATION



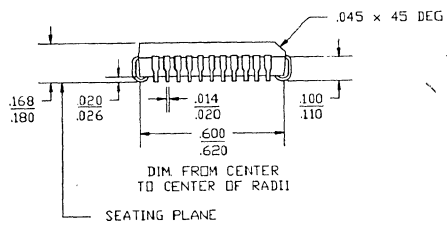
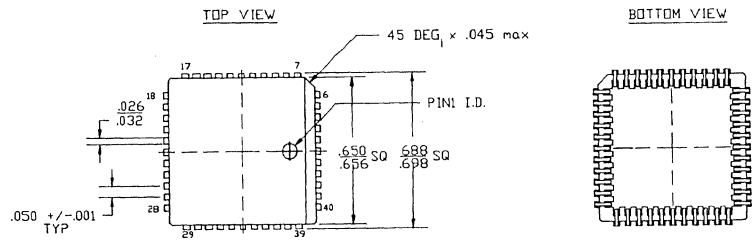
**40-Lead PDIP**



**48-Lead PDIP**



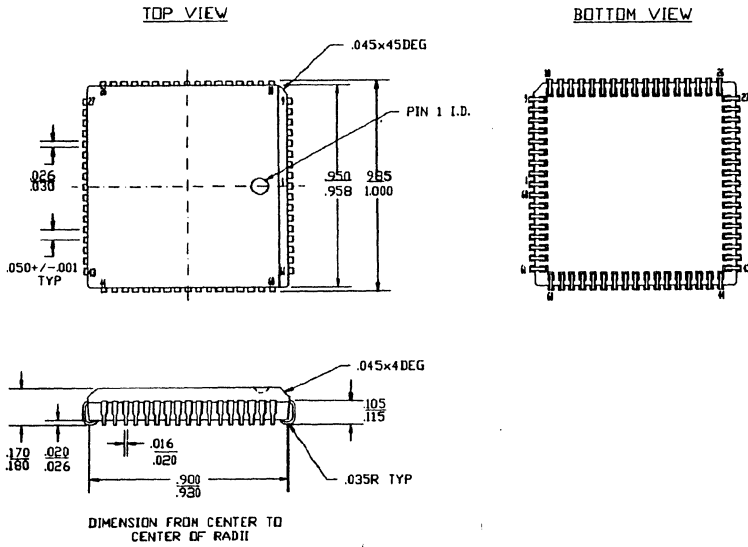
**64-Lead PDIP**



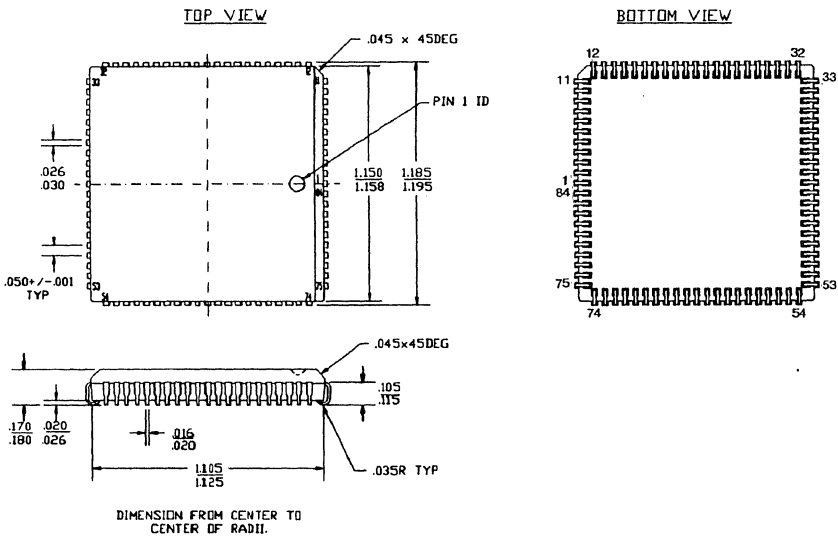
- NOTES:  
 1. ALL DIMENSIONS IN INCH.  
 2. LEADS ARE COPLANAR WITHIN .004 IN RANGE.

**44-Lead PLCC**

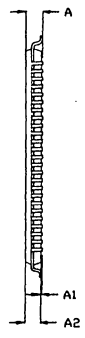
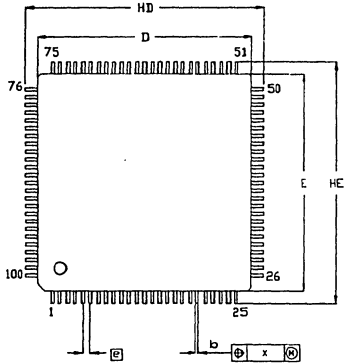
**PACKAGE INFORMATION (Continued)**



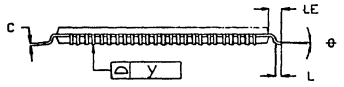
**68-Lead PLCC**



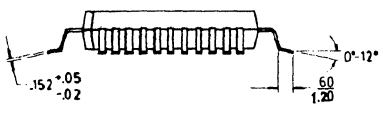
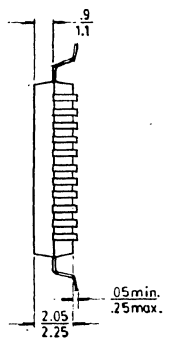
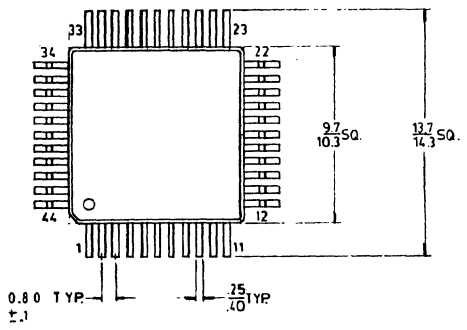
**84-Lead PLCC**



SYMBOL	DIMENSIONS IN MM.		
	MIN.	NOM.	MAX.
A	-	-	1.60
A1	-	-	0.20
A2	-	1.40	-
b	-	0.20	-
c	0.10	0.15	0.20
D	-	14.0	-
E	-	14.0	-
⌀	-	0.50	-
L	-	0.60	-
LE	-	1.00	-
HD	-	16.00	-
HE	-	16.00	-
x	-	-	0.15
y	-	-	0.10
⌀	0*	-	5*

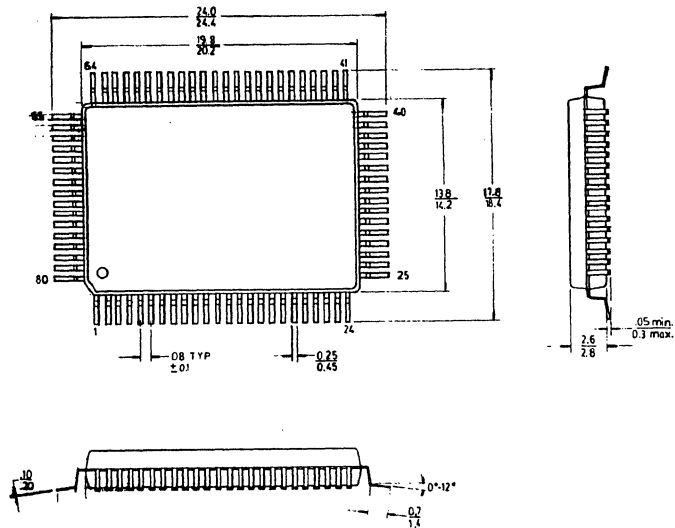


100-Lead VQFP

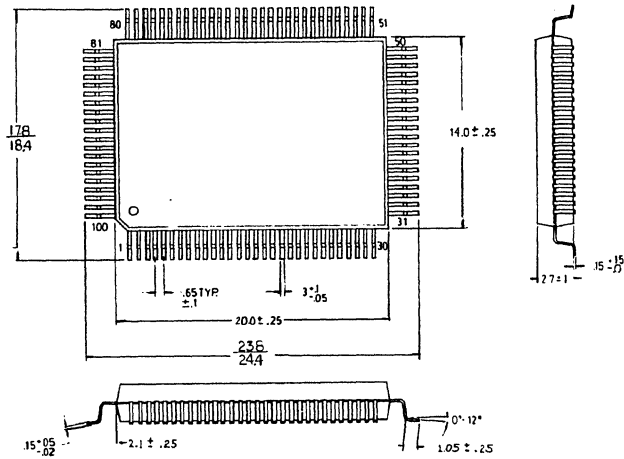


44-Lead QFP

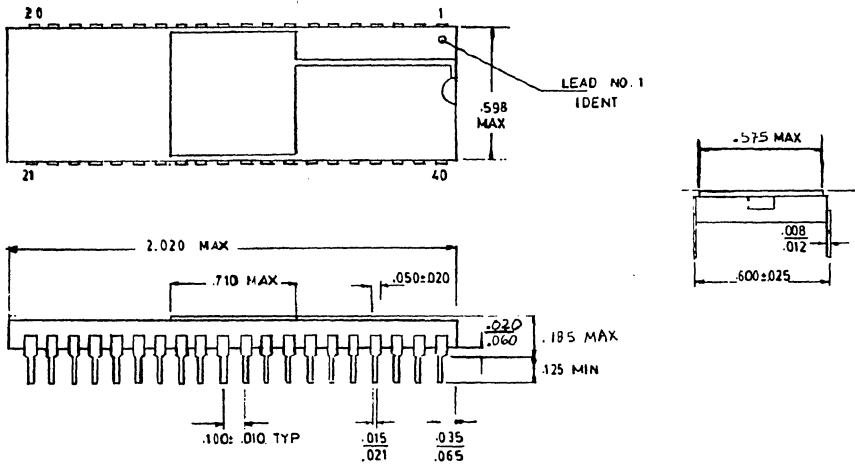
**PACKAGE INFORMATION** (Continued)



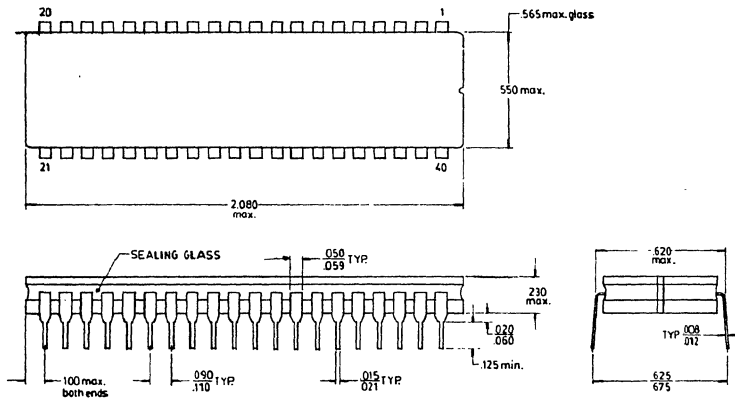
**80-Lead QFP**



**100-Lead QFP**



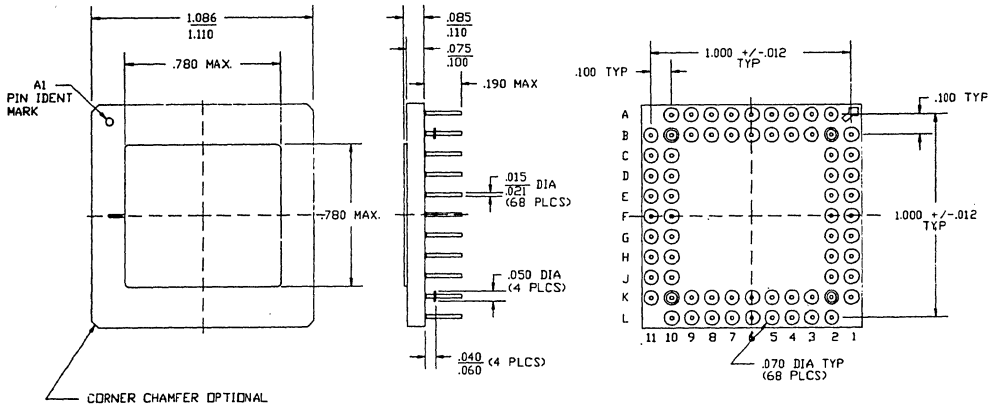
**40-Lead Sidebrazed CERDIP**



**40-Lead CERDIP**



**PACKAGE INFORMATION (Continued)**



**68-Lead PGA**

---

## ORDERING INFORMATION

### NMOS Z80 CPU 4,6,8 MHz

40-Pin DIP	44-Pin PLCC
Z0840004DSE	Z0840004VSC
Z0840004PSC	Z0840006VSC
Z0840006DSE	Z0840008VSC
Z0840006PSC	
Z0840008PSC	

### CMOS Z80 CPU 6,8,10,20 MHz

40-Pin DIP	44-Pin PLCC	44-Pin QFP
Z84C0006DDEE	Z84C0006VEE	Z84C0006FEC
Z84C0006PEC	Z84C0008VEE	Z84C0010FEC
Z84C0008PEC	Z84C0010VEE	Z84C0020FEC
Z84C0010PEC	Z84C0020VEE	
Z84C0020PEC		

### NMOS Z80 DMA 4 MHz

40-Pin DIP	44-Pin PLCC
Z0841004PSC	Z0841004VSC
Z0841004DSE	

### CMOS Z80 DMA 6,8 MHz

40-Pin DIP	44-Pin PLCC	44-Pin QFP
Z84C1006PEC	Z84C1006VEE	Z84C1006FEC
Z84C1008PEC	Z84C1008VEE	

### NMOS Z80 PIO 4,6 MHz

40-Pin DIP	44-Pin PLCC
Z0842004DSE	Z0842004VSC
Z0842004PSC	Z0842006VSC
Z0842006DSE	
Z0842006PSC	

### CMOS Z80 PIO 6,8,10 MHz

40-Pin DIP	44-Pin PLCC	44-Pin QFP
Z84C2006DDEE	Z84C2006VEE	Z84C2006FEC
Z84C2006PEC	Z84C2008VEE	Z84C2010FEC
Z84C2008PEC	Z84C2010VEE	
Z84C2010PEC		

### NMOS Z80 CTC 4,6 MHz

40-Pin DIP	44-Pin PLCC
Z0843004DSE	Z0843004VSC
Z0843004PSC	Z0843006VSC
Z0843006DSE	
Z0843006PSC	

### CMOS Z80 CTC 6,8,10 MHz

40-Pin DIP	44-Pin PLCC	44-Pin QFP
Z84C3006DDEE	Z84C3006VEE	Z84C3006FEC
Z84C3006PEC	Z84C3008VEE	Z84C3010FEC
Z84C3008PEC	Z84C3010VEE	
Z84C3010PEC		

### NMOS Z80 DART 4,6 MHz

40-Pin DIP
Z0847004PSC
Z0847006PSC

### NMOS Z80 SIO 4,6 MHz

40-Pin DIP	44-Pin PLCC
Z0844004DSE	Z0844404VSC
Z0844004PSC	Z0844406VSC
Z0844104DSE	
Z0844104PSC	
Z0844204DSE	
Z0844204PSC	
Z0844006DSE	
Z0844006PSC	
Z0844106DSE	
Z0844106PSC	
Z0844206DSE	
Z0844206PSC	

### CMOS Z80 SIO 6,8,10 MHz

40-Pin DIP	44-Pin PLCC	44-Pin QFP
Z84C4006DDEE	Z84C4406VEE	Z84C4306FEC
Z84C4006PEC	Z84C4408VEE	Z84C4310FEC
Z84C4106PEC	Z84C4410VEE	
Z84C4206DDEE		
Z84C4206PEC		
Z84C4008PEC		
Z84C4208PEC		
Z84C4010PEC		
Z84C4010DEC		
Z84C4110PEC		
Z84C4210PEC		
Z84C4108PEC		

### CMOS Z80 CPU+OSC 6,10 MHz

44-Pin PLCC	44-Pin QFP
Z84C0106VEE	Z84C0106FEC
Z84C0110VEE	Z84C0110FEC



---

## ORDERING INFORMATION

### CMOS USC 10, 20 Mbits/sec

68-pin PLCC	68-pin PGA
Z16C3010VSC	Z16C3010GEE
Z16C3010VEC	
Z16C3020VSC	

### CMOS IUSC 20 Mbits/sec

68-pin PLCC
Z16C3220VSC

### CMOS MUSC 10 Mbits/sec

68-pin PLCC
Z16C3310VSC

### CMOS ISCC 10, 16 MHz

68-pin PLCC
Z16C3510VSC
Z16C3516VSC

### CMOS DPLL 10, 20 MHz

28-pin P-DIP
Z16C5010PSC
Z16C5020PSC

### CMOS SCSI 3 Mbytes/sec

48-Pin PDIP	44-Pin PLCC
Z53C8003PSC	Z53C8003VSC

### CMOS SCSI 1.5Mbytes/sec

40-pin P-DIP	44-pin PLCC
Z0538010PSC	Z0538010VSC

### CMOS Z-Bus ESCC 10, 16 MHz

40-Pin PDIP	44-Pin PLCC
Z8023010PSC	Z8023010VSC
Z8023016PSC	Z8023016VSC

### CMOS ESCC 8, 10, 16, 20 MHz

40-pin P-DIP	44-pin PLCC
Z8523008PSC/PEC	Z8523008VSC/VEC
Z8523010PSC/PEC	Z8523010VSC/VEC
Z8523016PSC/PEC	Z8523016VSC/VEC
Z8523020PSC	Z8523020VSC

### CMOS Z-Bus SCC 8,10, MHz

40-pin DIP	44-pin PLCC
Z80C3008PSC	Z80C3008VSC
Z80C3010PSC	Z80C3010VSC

### CMOS SCC 8, 10, 16 MHz

40-pin DIP	44-pin PLCC
Z85C3008PSC	Z85C3008VSC
Z85C3008PEC	Z85C3008VEC
Z85C3008CEE	Z85C3010VSC
Z85C3010PSC	Z85C3010VEC
Z85C3010PEC	Z85C3016VSC
Z85C3010CEE	
Z85C3016DEA	

### NMOS Z-Bus SCC 6, 8 MHz

40-pin DIP	44-pin PLCC
Z0803006PSC	Z0803006VSC
Z0803006DSE	Z0803008VSC
Z0803008PSC	
Z0803008DSE	

### NMOS SCC 6, 8 MHz

40-pin DIP	44-pin PLCC
Z0853004PSC	Z0853006VSC
Z0853006PSC	Z0853008VSC
Z0853006PEC	
Z0853006DSE	
Z0853008PSC	
Z0853008DSE	
Z0853008DEA	

### CMOS SCSCI 10 MHz

68-pin PLCC
Z85C8010VSC

#### Notes:

For military grade devices and the package types other than listed above, please contact your local Zilog sales office.

Please check the availability before placing order.



---

## ZILOG'S QUALITY AND RELIABILITY PROGRAM

### Introduction

Zilog Corporation has an excellent reputation for the quality and reliability of its products.

Zilog's Quality and Reliability Program is based on careful study of the principles laid down by such pioneers as W. E. Deming and J. M. Juran; perhaps even more important, observation of the practical implementation of those principles in Japanese, European and American manufacturing facilities.

The Zilog program begins with employee involvement. Whether the judgement of our performance is based on perfection with incoming inspection, trouble free service in the field, or timely and accurate customer service, we recognize that our employees ultimately control these factors. Hence, our Quality Program is broadly shared throughout the organization.

#### 1. Harmony Between Design and Process

High product quality and reliability in VLSI products are possible only if there is structural harmony between product design and the manufacturing process. Great care is taken to assure that the statistical process control limits observed within the manufacturing plants, properly guardband the design technology used to configure the circuit and layout in Zilog's automated design methodology.

Through use of a technique which we call Process Templating, the technology file in the automated design system is periodically updated to assure that product design parameters fall within the statistical control limits with which the process is actually operated. In simple terms, the Process Template is the profile displayed by the process evaluation parameters which are automatically recorded from the test patterns on wafers as they proceed through the production line. These parameters are translated into the design technology file attributes so every product design bears a key and lock relationship to the process.

#### 2. Training

Product Design and Processing are people dependent. Zilog training emphasizes the fundamentals involved in design for quality and reliability.

Customer Service, an important aspect of Zilog's quality performance as a vendor, also depends upon our people clearly understanding their jobs, and our obligations to our customers. This too is part of the training curriculum administered by Zilog.

---

### 3. Order Acknowledgement Policy

One definition of vendor quality performance is that the vendor "does what he promises or acknowledges." Reliability and quality warranties are met only if Zilog and the customer are in agreement on product and delivery specifications. Zilog makes an extra effort to assure that the customer is fully informed by providing documents with its purchase order acknowledgements that clearly state what Zilog understands the specifications to be.

### 4. Test Guardbanding

No physical attribute is absolute. Customers' test methods may differ from Zilog's due to variations in test equipment, temperature or specification interpretation. To assure that every Zilog product performs to full customer expectations, Zilog uses a "waterfall" methodology in its testing. The first electrical tests made on the circuit, at the wafer probe operation, are guardbanded to the final test specifications. The final test specifications, in turn, are guardbanded to the quality control outgoing sample. The quality control outgoing sample is guardbanded to the customer procurement or data sheet specifications. This technique of "waterfall" guardbanding assures that circuits which may be marginal to the customer's expectations are eliminated in the manufacturing process long before they get to the shipping container.

### 5. Probe at Temperature

Semiconductor devices tend to exhibit their most limited performance at the highest operating temperature. Therefore, it is Zilog's policy that all chips are tested at high temperature the very first time they are electrically screened, at the wafer probe station. The circuits are tested again at their upper operating temperature limit in the 100% final test operation.

### 6. Process Characterization

Before release to production, every process is thoroughly characterized by an exhaustive series of pilot production runs and tests which identify the statistical, electrical, and mechanical limits of which that particular process regime is capable. This documentation, which fills a large looseleaf binder for each process, is maintained as the historical record or "footprint" for that particular regime.

Process recharacterization is done any time there is a major process or manufacturing site change, and the resulting documentation is then added to the characterization history. Once the process is fully characterized, the frequent test site evaluation and process template data demonstrates that the process remains in specification.

---

## 7. Product Characterization

Every Zilog product design is evaluated over extremes of operating temperature, supply voltage, and clock frequencies, prior to production release. This information permits the proper guardbanding of the test program waterfall and identification of any marginal "corners" in design tolerances.

A product characterization report, which summarizes the more important tolerances identified in the process of this exhaustive product design evaluation, is available to Zilog's customers.

## 8. Process Qualification

Zilog also qualifies every process prior to production by an exhaustive stress sequence performed on test chips and on representative products. Once a process regime is qualified, a process requalification is performed any time there is a major process change, or whenever the process template statistical quality limits are significantly exceeded or adjusted.

## 9. Product Qualification

In addition to characterization, every new Zilog product design is fully qualified by a comprehensive series of life, electrical, and environmental tests before release to production. Again, a qualification report is available to our customers which summarizes certain key life and environmental data taken in the course of these evaluations. Whenever possible, industry standard environmental and life tests are employed.

## 10. PPM Measurement, Direct and Indirect

It is frequently said that if you want to improve something, you need to put a measure on it. Therefore, Zilog measures its outgoing quality "parts per million" by the maintenance of careful records on the statistical sampling of production lots prepared for shipment. This information is then translated by our statisticians to a statement of our parts per million (or parts per billion) outgoing quality performance.

Of course, it is one thing for Zilog to think it is doing a good job in outgoing product quality and it is another for a customer to agree. Therefore, we ask certain key customers to provide us with their incoming inspection data which helps us calibrate our outgoing performance in terms of the actual results in the field. The fact that Zilog has been awarded "ship to stock" status by many customers testifies to our success in this area.

## 11. FIT Measurement Direct and Indirect

Just as Zilog records its outgoing quality in terms of parts per million, it also measures its outgoing product reliability in terms of "FITs" or failures per billion device hours, using the



---

results of weekly operating life test measurements on the circuits, performed in accordance with the standard specifications.

#### 12. Field Quality Engineers

It is frequently said that the customer is always right. If the customer has an application quality or reliability problem while using a Zilog product, whether it is Zilog's responsibility or not, we believe that we have a responsibility to resolve it. Therefore, Zilog maintains a force of skilled Applications Engineers who are also trained as field quality engineers and are available on immediate call to consult at the customer's locations on any problems they may be experiencing with Zilog product performance.

#### 13. Product Analysis

As noted earlier, we feel that a customer problem is a Zilog problem. Accordingly, Product Analysis facilities, staffed by experienced professionals, exist at each Zilog site to provide rapid evaluation of in-process and in-field rejects to determine the cause and provide corrective action through a feedback loop into the production, design, and applications process. Zilog is pleased to share product analysis reports on specific products with the customer upon request.

#### 14. Test Site Step-Stress

The process evaluation test sites on the wafer are packaged and subjected to step-stress testing. Any drift in parameters under severe conditions of stress outside the norm is taken as an indication of possible process contamination or variation.

#### 15. Statistical Process Control

Zilog employs Statistical Process Control at all critical process steps. Deviations from norms must be evaluated by a Q/R review board.

#### 16. Perfection Plus Program

Zilog employees actively participate in meetings in which methods which will enable a department to do its job more perfectly are proposed, reviewed, and adopted. Employees who have made suggestions proudly wear the Zilog Perfection Plus pin.

#### 17. Zilog Vendor of the Year Award

Zilog is proud of the many quality and performance awards it has received from its customers. In turn, Zilog makes an annual award to the vendor who has done the best overall job for Zilog.



# LITERATURE GUIDE

## Z8®/SUPER8™ MICROCONTROLLER FAMILY

Databook	Part No	Unit Cost
Z8 Microcontrollers Databook (includes the following documents)	DC-8275-04	5.00
<b>Z8 CMOS Microcontrollers</b> Z86C00/C10/C20 MCU OTP Product Specification Z86C06 Z8 CCP™ Preliminary Product Specification Z86C08 8-Bit MCU Product Specification Z86E08 Z8 OTP MCU Product Specification Z86C09/19 Z8 CCP Product Specification Z86E19 Z8 OTP MCU Advance Information Specification Z86C11 Z8 MCU Product Specification Z86C12 Z8 ICE Product Specification Z86C21 Z8 MCU Product Specification Z86E21/Z86E22 OTP Product Specification Z86C30 Z8 CCP Product Specification Z86E30 Z8 OTP CCP Product Specification Z86C40 Z8 CCP Product Specification Z86E40 Z8 OTP CCP Product Specification Z86C27/97 Z8 DTC™ Product Specification Z86127 Low-Cost Digital Television Controller Adv. Info. Spec. Z86C50 Z8 CCP ICE Advance Information Specification Z86C61 Z8 MCU Advance Information Specification Z86C62 Z8 MCU Advance Information Specification Z86C89/C90 CMOS Z8 CCP Product Specification Z86C91 Z8 ROMless MCU Product Specification Z86C93 Z8 ROMless MCU Preliminary Product Specification Z86C94 Z8 ROMless MCU Product Specification Z86C96 Z8 ROMless MCU Advance Information Specification Z88C00 CMOS Super8 MCU Advance Information Specification		
<b>Z8 NMOS Microcontrollers</b> Z8600 Z8 MCU Product Specification Z8601/03/11/13 Z8 MCU Product Specification Z8602 8-Bit Keyboard Controller Preliminary Product Spec. Z8604 8-Bit MCU Product Specification Z8612 Z8 ICE Product Specification Z8671 Z8 MCU With BASIC/Debug Interpreter Product Spec. Z8681/82 Z8 MCU ROMless Product Specification Z8691 Z8 MCU ROMless Product Specification Z8800/01/20/22 Super8 ROMless/ROM Product Specification		
<b>Peripheral Products</b> Z86128 Closed-Captioned Controller Adv. Info. Specification Z765A Floppy Disk Controller Product Specification Z5380 SCSI Product Specification Z53C80 SCSI Advance Information Specification		
<b>Z8 Application Notes and Technical Articles</b> Zilog Family On-Chip Oscillator Design Z86E21 Z8 Low Cost Thermal Printer Z8 Applications for I/O Port Expansions Z86C09/19 Low Cost Z8 MCU Emulator Z8602 Controls A 101/102 PC/Keyboard The Z8 MCU Dual Analog Comparator The Z8 MCU In Telephone Answering Systems Z8 Subroutine Library A Comparison of MCU Units Z86xx Interrupt Request Registers Z8 Family Framing A Programmer's Guide to the Z8 MCU Memory Space and Register Organization		
<b>Super8 Application Notes and Technical Articles</b> Getting Started with the Zilog Super8 Polled Async Serial Operations with the Super8 Using the Super8 Interrupt Driven Communications Using the Super8 Serial Port with DMA Generating Sine Waves with Super8 Generating DTMF Tones with Super8 A Simple Serial Parallel Converter Using the Super8		
<b>Additional Information</b> Z8 Support Products Zilog Quality and Reliability Report Literature List Package Information Ordering Information		



# LITERATURE GUIDE

## Z8®/SUPER8™ MICROCONTROLLER FAMILY (Continued)

Z8 Product Specifications, Technical Manuals and Users Guides	Part No	Unit Cost
Z86C06 CMOS Z8 CCP Preliminary Product Specification	DC-2563-00	N/C
Z86C08 CMOS Z8 8-Bit Microcontroller Product Specification and Addendum	DC-2527-02	N/C
Z86E08 CMOS Z8 8-Bit Microcontroller Product Specification and Addendum	DC-2542-01	N/C
Z86C09/C19 CMOS Z8 8-Bit Microcontroller Product Specification	DC-2506-02	N/C
Z86C11 CMOS Z8 Microcontroller Product Specification	DC-2572-01	N/C
Z86C12 Z8 ICE Product Specification	DC-2553-01	N/C
Z86C17 CMOS Z8 Microcontroller Preliminary Specification	DC-2597-00	N/C
Z86C21 CMOS Z8 Microcontroller Product Specification	DC-2568-01	N/C
Z86E21 CMOS OTP Microcontroller Product Specification	DC-2514-01	N/C
Z86E23 CMOS Z8 OTP Microcontroller Preliminary Product Specification	DC-2598-00	N/C
Z86C27/97 Z8 DTC™ Product Specification	DC-2561-01	N/C
Z86127 Low-Cost Digital Television Controller Preliminary Product Specification	DC-2574-00	N/C
86227 40-Pin Low-Cost Digital Television Controller Preliminary Product Specification	DC-3002-00	N/C
Z86C30 CMOS Z8 8-Bit Microcontroller Product Specification	DC-2509-03	N/C
Z86E30 CMOS Z8 OTP CCP Product Specification	DC-2573-01	N/C
Z86C40 CMOS CCP Product Specification	DC-2550-01	N/C
Z86E40 CMOS OTP CCP Product Specification	DC-2571-01	N/C
Z86C50 CMOS Z8 CCP ICE Advance Information Specification	DC-2559-0A	N/C
Z86C61/62/96 CMOS Z8 Microcontroller Preliminary Product Specification	DC-2587-00	N/C
Z86C90/C89 ROMless CMOS Z8 8-Bit Microcontroller Product Specification	DC-2506-01	N/C
Z86C91 Z8 CMOS ROMless Microcontroller Product Specification	DC-2566-01	N/C
Z86C93 CMOS Z8 ROMless Microcontroller Product Specification	DC-2508-01	N/C
Z86C95 CMOS Z8 Digital Signal Processor Advanced Information Specification	DC-3000-00	N/C
Z88C00 CMOS Super8 ROMless Microcontroller Advance Information Specification	DC-2551-0A	N/C
Z8602 NMOS Z8 8-Bit Microcomputer Keyboard Controller Preliminary Product Specification	DC-2525-01	N/C
Z8604 NMOS Z8 8-Bit Microcontroller Preliminary Product Specification	DC-2524-03	N/C
Z8614 NMOS Z8 8-Bit MCU Keyboard Controller Preliminary Product Specification	DC-2576-00	N/C
Z86128 Closed-Captioned Controller Preliminary Product Specification	DC-2570-00	N/C
Z86L06 Low Voltage CMOS Preliminary Product Specification	DC-2564-01	N/C
Z86L29 6K Infrared (IR) Remote (ZIRC) Controller Advanced Information Specification	DC-2602-0A	N/C
asm S8 Super8/Z8 Cross Assembler User's Guide	DC-8267-05	3.00
Z8 Microcontrollers Technical Manual	DC-8291-01	5.00

Z8 Application Notes	Part No	Unit Cost
The Z8 MCU In Telephone Answering Systems	DC-2514-01	N/C
Z8602 Controls A 101/102 PC/Keyboard	DC-2521-01	N/C
The Z8 MCU Dual Analog Comparator	DC-2516-01	N/C
Z86C09/19 Low Cost Z8 MCU Emulator	DC-2537-01	N/C
Z8 Applications for I/O Port Expansions	DC-2539-01	N/C
Z86E21 Z8 Low Cost Thermal Printer	DC-2541-01	N/C
Zilog Family On-Chip Oscillator Design	DC-2496-01	N/C
Using the Zilog Z86C06 SPI Bus	DC-2584-01	N/C
Interfacing LCDs to the Z8	DC-2592-01	N/C
X-10 Compatible Infrared (IR) Remote Control	DC-2591-01	N/C
Z86C17 In-Mouse Applications	DC-3001-01	N/C



# LITERATURE GUIDE

## VOLUME I DATABOOK

### Z80®/Z180™/Z280®/Z8000® and Datacom Family

Part No

Unit Cost

Volume I Databook

DC-2610-01

5.00

#### **Discrete Z80® Family**

Z8400/C00 NMOS/CMOS Z80® CPU Product Specification  
Z8410/C10 NMOS/CMOS Z80 DMA Product Specification  
Z8420/C20 NMOS/CMOS Z80 PIO Product Specification  
Z8430/C30 NMOS/CMOS Z80 CTC Product Specification  
Z8440/Z84C40 NMOS/CMOS Z80 SIO Product Specification

Z53C80 SCSI Product Specification  
Z85C80 SCSI/SCC Product Specification  
Z16C30 USC™ Product Specification  
Z16C32 IUSC™ Product Specification  
Z16C33 MUSC™ Product Specification  
Z16C50 DDPLL™ Product Specification

#### **Embedded Controllers**

Z84C01 Z80 CPU with CGC Product Specification  
Z84C50 RAM80™ Preliminary Product Specification  
Z8470 Z80 DART Product Specification  
Z84C90 CMOS Z80 KIO™ Product Specification  
Z84011/C11 PIO Parallel I/O Product Specification  
Z84013/015 Z84C13/C15 IPC/EIPC™ Product Specification  
Z80180/Z8S180 Z180 MPU Product Specification  
Z80181 Z10™ Controller Product Specification  
Z280™ MPU Preliminary Product Specification

#### **Technical Articles**

Z80 Questions and Answers  
Z180 Questions and Answers  
SCC Questions and Answers  
ESCC Questions and Answers  
ISCC Questions and Answers

#### **Serial Communications Controllers**

Z8030/Z8530 Z-BUS® SCC Product Specification  
Z80C30/Z85C30 SCC Product Specification  
Z85230 ESCC™ Product Specification  
Z80230 Z-BUS ESCC Product Specification  
Z16C35 ISCC™ Product Specification  
Z5380 SCSI Product Specification

#### **Additional Information**

Superintegration Products Guide  
Support Product Summary  
Product Support  
Military Qualified Products  
Quality and Reliability  
Literature Guide  
Package Information  
Ordering Information



# LITERATURE GUIDE

<b>Z80/Z180/Z280 Product Specifications, Technical Manuals and Users Guides</b>	<b>Part No</b>	<b>Unit Cost</b>
Z80 CPU Central Processing Unit Technical Manual	DC-0029-04	3.00
Z80 Family Programmer's Reference Guide	DC-0012-04	3.00
Z80 DMA Direct Memory Access Technical Manual	DC-2013-A0	3.00
Z80 PIO Parallel Input/Output Technical Manual	DC-0008-03	3.00
Z80 CTC Counter/Timer Circuit Technical Manual	DC-0036-03	3.00
Z80 SIO Serial I/O Technical Manual	DC-3033-01	3.00
Z80180 Z180 MPU Microprocessor Unit Technical Manual	DC-8276-03	3.00
Z280 MPU Microprocessor Unit Technical Manual	DC-8224-03	3.00
Z80181 Z181 SAC™ Smart Access Controller Preliminary Product Specification	DC-2519-02	N/C
Z84013/15, Z84C13/C15 CMOS IPC™ Intelligent Peripheral Controller Product Specification	DC-2507-04	N/C
Z84011/C11 PIO Parallel I/O Controller Product Specification	DC-2526-02	N/C
Z84C00 20 MHz Z80 CPU Central Processing Unit Preliminary Product Specification	DC-2523-02	N/C
Z84C50 Z80 RAM80 Z80 CPU/2K SRAM Preliminary Product Specification	DC-2498-01	N/C

<b>Z80/Z180/Z280 Application Notes</b>	<b>Part No</b>	<b>Unit Cost</b>
Z180/SCC™ Serial Communications Controller Interface at 10 MHz	DC-2521-02	N/C
Z80 Using the 84C11/C13/C15 in place of the 84011/013/015	DC-2499-02	N/C
LocalTalk Link Access Protocol Using the Z80181	DC-2589-01	N/C
A Fast Z80 Embedded Controller	DC-2578-01	N/C



# LITERATURE GUIDE

## Z8000® MICROPROCESSOR FAMILY

Z8000 Product Specifications, Technical Manuals and Users Guides	Part No	Unit Cost
Z8000 CPU Central Processing Unit Technical Manual	DC-2010-06	3.00
Z8010 MMU Memory Management Unit Technical Manual	DC-2015-A0	3.00
Z8030/Z8530 SCC Serial Communication Controller Technical Manual	DC-2057-06	3.00
Z8036 Z-CIO/Z8536 CIO Counter/Timer and Parallel Input/Output Technical Manual	DC-2091-02	3.00
Z8038 Z8000 Z-FIO FIFO Input/Output Interface Technical Manual	DC-2051-01	3.00
Z8000 CPU Central Processing Unit Programmer's Pocket Guide	DC-0122-03	3.00
Z5380 SCSI Small Computer System Interface Preliminary Product Specification	DC-2477-01	N/C
Z80C30/Z85C30 CMOS SCC Serial Communication Controller Product Specification	DC-2442-05	N/C
Z85230 ESCC™ Enhanced Serial Communication Controller Product Specification	DC-2596-01	N/C
Z85233 EMSCC Enhanced Mono Serial Communication Controller Preliminary Product Specification	DC-2590-00	N/C
Z85C80 SCSCI™ Serial Communication and Small Computer Interface Preliminary Product Specification	DC-2534-02	N/C
Z16C01/2/3 CPU Central Processing Unit Preliminary Product Specification	DC-2504-02	N/C
Z16C30 CMOS USC™ Universal Serial Controller Preliminary Product Specification	DC-2492-03	N/C
Z16C30 USC Universal Serial Controller Preliminary Technical Manual	DC-8296-01	3.00
Z16C30/Z16C33 CMOS USC/MUSC™ Universal Serial Controller Technical Manual	DC-8285-01	3.00
Z16C30/Z16C33 CMOS USC/MUSC Universal Serial Controller Addendum	DC-8285-01A	N/C
Z16C31 IUSC™ Integrated Universal Serial Controller Product Specification	DC-2544-02	N/C
Z16C33 CMOS MUSC Mono-Universal Serial Controller Preliminary Product Specification	DC-2517-03	N/C
Z16C35 CMOS ISCC™ Integrated Serial Communication Controller Product Specification	DC-2515-03	N/C
Z16C35 ISCC Integrated Serial Communication Controller Technical Manual	DC-8286-01	3.00
Z16C35 ISCC Integrated Serial Communication Controller Addendum	DC-8286-01A	N/C
Z16C50 DDPLL™ Dual Digital Phase-Locked Loop Preliminary Product Specification	DC-2540-01	N/C
Z85230/Z80230 ESCC Enhanced Serial Communication Controller Technical Manual	DC-8288-01	3.00
Z53C80 Small Computer System Interface (SCSI) Product Specification	DC-2575-01	N/C
Z80230 Z-Bus® ESCC Enhanced Serial Communication Controller Preliminary Product Specification	DC-2603-00	N/C
Z8000 Application Notes	Part No	Unit Cost
Z16C30 Using the USC in Military Applications	DC-2536-01	N/C
Z16C35 ISCC Interface to Intel and Motorola Microprocessors	DC-2522-02	N/C
Datcom IUSC/MUSC Time Slot Assigner	DC-2497-02	N/C
Datcom Evaluation Board Using The Zilog Family With The 80186 CPU	DC-2560-02	N/C
ESCC Enhancements Over The SCC	DC-2555-01	N/C
Z16C30 USC - Design a Serial Board for Multiple Protocols	DC-2554-01	N/C
Integrating Serial Data and SCSI Peripheral Control on one Chip	DC-2594-01	N/C



# LITERATURE GUIDE

## MILITARY COMPONENTS FAMILY

Military Specifications	Part No	Unit Cost
Z8681 ROMless Microcomputer Military Product Specification	DC-2392-02	N/C
Z8001/8002 Military Z8000 CPU Central Processing Unit Military Product Specification	DC-2342-03	N/C
Z8581 Military CGC Clock Generator and Controller Military Product Specification	DC-2346-01	N/C
Z8030 Military Z8000 Z-SCC Serial Communications Controller Military Product Specification	DC-2388-02	N/C
Z8530 Military SCC Serial Communications Controller Military Product Specification	DC-2397-02	N/C
Z8036 Military Z8000 Z-CIO Counter/Timer Controller and Parallel I/O Military Electrical Specification	DC-2389-01	N/C
Z8038/8538 Military FIO FIFO Input/Output Interface Unit Military Product Specification	DC-2463-02	N/C
Z8536 Military CIO Counter/Timer Controller and Parallel I/O Military Electrical Specification	DC-2396-01	N/C
Z8400 Military Z80 CPU Central Processing Unit Military Electrical Specification	DC-2351-02	N/C
Z8420 Military PIO Parallel Input/Output Controller Military Product Specification	DC-2384-02	N/C
Z8430 Military CTC Counter/Timer Circuit Military Electrical Specification	DC-2385-01	N/C
Z8440/1/2/4 Z80 SIO Serial Input/Output Controller Military Product Specification	DC-2386-02	N/C
Z80C30/85C30 Military CMOS SCC Serial Communications Controller Military Product Specification	DC-2478-02	N/C
Z84C00 CMOS Z80 CPU Central Processing Unit Military Product Specification	DC-2441-02	N/C
Z84C20 CMOS Z80 PIO Parallel Input/Output Military Product Specification	DC-2384-02	N/C
Z84C30 CMOS Z80 CTC Counter/Timer Circuit Military Product Specification	DC-2481-01	N/C
Z84C40/1/2/4 CMOS Z80 SIO Serial Input/Output Military Product Specification	DC-2482-01	N/C
Z16C30 CMOS USC Universal Serial Controller Military Preliminary Product Specification	DC-2531-01	N/C
Z16C01/2 CPU Central Processing Unit Military Product Specification	DC-2532-01	N/C
Z80180 Z180 MPU Microprocessor Unit Military Product Specification	DC-2538-01	N/C
Z84C90 CMOS KIO Serial/Parallel/Counter Timer Preliminary Military Product Specification	DC-2502-00	N/C
Z85230 ESCC Enhanced Serial Communication Controller Military Product Specification	DC-2595-00	N/C

## GENERAL LITERATURE

Catalogs, Handbooks and Users Guides	Part No	Unit Cost
Superintegration Short Form Catalog 1991	DC-5472-08	N/C
Quality and Reliability Report	DC-2475-08	N/C
Superintegration Products Guide	DC-5499-05	N/C
The Handling and Storage of Surface Mount Devices User's Guide	DC-5500-02	N/C
Support Products Summary	DC-2545-03	N/C
Universal Object File Utilities User's Guide	DC-8236-04	3.00
Zilog 1991 Annual Report	DC-1991-AR	N/C
Microcontroller Quick Reference Folder	DC-5508-01	N/C

## ZILOG DOMESTIC SALES OFFICES AND TECHNICAL CENTERS

### CALIFORNIA

Agoura ..... 818-707-2160  
Campbell ..... 408-370-8120  
Tustin ..... 714-838-7800

### COLORADO

Boulder ..... 303-494-2905

### FLORIDA

Largo ..... 813-585-2533

### GEORGIA

Norcross ..... 404-448-9370

### ILLINOIS

Schaumburg ..... 708-517-8080

### MINNESOTA

Minneapolis ..... 612-944-0737

### NEW HAMPSHIRE

Nashua ..... 603-888-8590

### NORTH CAROLINA

Raleigh ..... 919-790-7706

### OHIO

Independence ..... 216-447-1480

### PENNSYLVANIA

Ambler ..... 215-653-0230

### TEXAS

Dallas ..... 214-987-9987

### WASHINGTON

Seattle ..... 206-523-3591

## INTERNATIONAL SALES OFFICES

### CANADA

Toronto ..... 416-673-0634

### GERMANY

Munich ..... 49-89-672-045  
Sömmerda ..... 37-626-23906

### JAPAN

Tokyo ..... 81-3-3587-0528

### HONG KONG

Kowloon ..... 852-7238979

### KOREA

Seoul ..... 82-2-552-5401

### SINGAPORE

Singapore ..... 65-2357155

### TAIWAN

Taipei ..... 886-2-741-3125

### UNITED KINGDOM

Maidenhead ..... 44-628-392-00

© 1992 by Zilog, Inc. All rights reserved. No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Zilog, Inc. The information in this document is subject to change without notice. Devices sold by Zilog, Inc. are covered by warranty and patent indemnification provisions appearing in Zilog, Inc. Terms and Conditions of Sale only. Zilog, Inc. makes no warranty, express, statutory, implied or

by description, regarding the information set forth herein or regarding the freedom of the described devices from intellectual property infringement. Zilog, Inc. makes no warranty of merchantability or fitness for any purpose. Zilog, Inc. shall not be responsible for any errors that may appear in this document. Zilog, Inc. makes no commitment to update or keep current the information contained in this document.



Zilog, Inc. 210 E. Hacienda Ave., Campbell, CA 95008-6600, Tel: (408) 370-8000, Telex 910-338-7621, FAX 408 370-8056

DC 2610-01