MOS Microprocessors and Peripherals

1983 Data Book

ADVANCED MICRO DEVICES

Advanced Micro Devices

MOS Microprocessors and Peripherals Data Book

© 1983 Advanced Micro Devices, Inc.

Advanced Micro Devices reserves the right to make changes in its products without notice in order to improve design or performance characteristics. The company assumes no responsibility for the use of any circuits described herein.

901 Thompson Place, P.O. Box 3453, Sunnyvale, California 94088
(408) 732-2400 TWX: 910-339-9280 TELEX: 34-6306

Printed in U.S.A. 6/83 03845A-MMP

Index Section

NUMERIC INDEX

*These pages are included for reference only. For complete data sheets, refer to "Bipolar Microprocessor Logic and Interface" data book, 1983 edition.

*These pages are included for reference only. For complete data sheets, refer to "Bipolar Microprocessor Logic and Interface" data book, 1983 edition.

FUNCTIONAL INDEX

MOS Microprocessor Family Selection Guide

	8086/88	8085A	8085A-2	8080A	Z8001/2†	Z8001/2-A
Clock Period	200ns	320ns	200ns	480ns	250ns	165ns
Clock Generator	8284A	On-Chip	On-Chip	8224	8127	8127
Arithmetic Processing Unit	8087	9511A-1 9512-1	9511A-4 9512-1	9511A 9512	9511A-4 9512-1	9511A-4 9512-1
Interrupt Controller	8259A-5	9519A 8259A	9519A-4 8259A-5	9519A 8259A	9519A-1	9519A-1
DMA Controller	8089 9516A 9517A-5	9517A-4	9517A-5	9517A	8016 9517A-4	8016A
Dynamic Memory Controller	2964B	2964B	2964B	2964B	2964B	2964B
Serial I/O	8251A 8530A 8030A 8031A 8531A	8251A 8530 8531	8251A 8530A 8531A	8251A	8030 8031	8030A 8031A
Parallel I/O	8255A-5 8036A	8255A-5	8255A-5	8255A	8036	8036A
Counter Timer I/O	9513 8036A 8073	9513 8253-5	9513 8253-5	9513 8253	8073	8073
FIFO I/O	8038	8038	8038	8038	8038	8038
Data Ciphering Processor	8068	8068	8068	9518	8068	8068
Error Detection and Correction	2960	2960	2960	2960	2960	2960
Burst Error Processor	8065 9520	8065 9520	8065 9520	8065 9520	8065 9520	8065 9520
CRT Controller	8052				8052	8052A
I/O Processor	8089	N/A	N/A	N/A	N/A	N/A
RAM I/O	N/A	8155/6	8155/6-2	N/A	N/A	N/A
Bus Control	8288	N/A	N/A	N/A	N/A	N/A
Bus Latches	29841-6	29841-6	29841-6	29841-6	29841-6	29841-6
Bus Buffers	29827/28	29827/28	29827/28	29827/28	29827/28	29827/28
Bus Transceivers	29861-4	29861-4	29861-4	29861-4	29861-4	29861-4
EDC Buffers	2961/2	2961/2	2961/2	2961/2	2961/2	2961/2
RAM Drivers	2965/6	2965/6	2965/6	2965/6	2965/6	2965/6

†Z8000 is a trademark of Zilog, Inc.

iAPX86 Family

iAPX 186

High Integration 16-Bit Microprocessor

ADVANCE INFORMATION

DISTINCTIVE CHARACTERISTICS

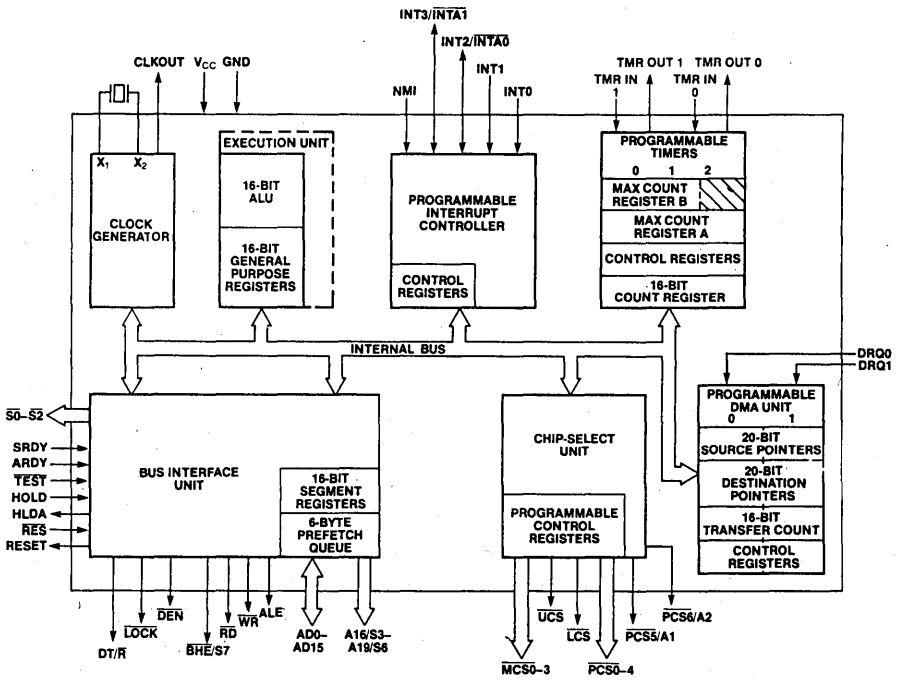
- Integrated feature set
  - Enhanced 8086-2 CPU
  - Clock generator
  - Two independent, high-speed DMA channels
  - Programmable interrupt controller
  - Three programmable 16-bit timers
  - Programmable memory and peripheral chip-select logic
  - Programmable wait state generator
  - Local bus controller
- High-performance 8MHz processor
  - Two times the performance of the standard iAPX 86
  - 4M byte/sec bus bandwidth interface
- Direct addressing capability to 1M byte of memory
- Completely object code compatible with all existing iAPX 86, 88 software
  - Ten new instruction types
- Compatible with 8282/83/86/87, 8288, 8289 bus support components
- Optional numeric processor extension
  - iAPX 186 with a high-performance 80-bit numeric data processor the 8087

GENERAL DESCRIPTION

The iAPX 186 (80186 part number) is a highly integrated 16-bit microprocessor. It effectively combines 15–20 of the most common iAPX 86 system components onto one. The 80186 provides two times greater throughput than the standard 5MHz iAPX 86. The iAPX 186 is upward compatible with iAPX 86 and 88 software and adds 10 new instruction types to the existing set.

The iAPX 186 comes in a 68-pin package and requires a single +5V power supply.

Figure 1. iAPX*186 Block Diagram



*iRMX, ICE and iAPX are trademarks of Intel Corporation. Reprinted by permission of Intel Corp. copyright 1983.

03551A-MMP

Figure 2. 80186 Pinout Diagram

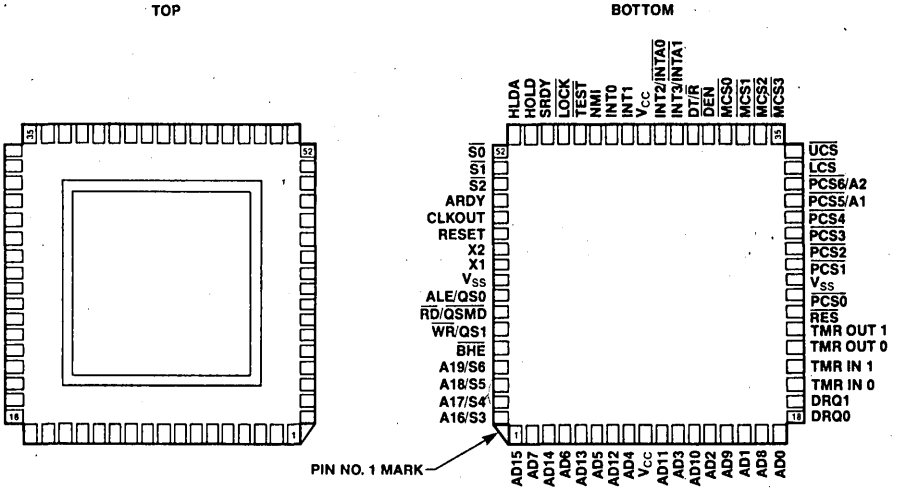


Table 1. 80186 Pin Description

Symbol	Pin No.	Type	Name and Function
V_{CC}, V_{CC}	9,43	I	System Power: +5 volt power supply.
V_{SS}, V_{SS}	26,60,	I	System Ground.
RESET	57	O	Reset Output indicates that the 80186 CPU is being reset, and can be used as a system reset. It is active HIGH, synchronized with the processor clock, and lasts an integer number of clock periods corresponding to the length of the $\overline{RES}$ signal.
X1, X2	59,58	I	Crystal Inputs, X1 and X2, provide an external connection for a fundamental mode parallel resonant crystal for the internal crystal oscillator. X1 can interface to an external clock instead of a crystal. The input or oscillator frequency is internally divided by two to generate the clock signal (CLKOUT).
CLKOUT	56	O	Clock Output provides the system with a 50% duty cycle waveform. All device pin timings are specified relative to CLKOUT. CLKOUT has sufficient MOS drive capabilities for the 8087 Numeric Processor Extension.
$\overline{RES}$	24	I	System Reset causes the 80186 to immediately terminate its present activity, clear the internal logic, and enter a dormant state. This signal may be asynchronous to the 80186 clock. The 80186 begins fetching instructions approximately 7 clock cycles after $\overline{RES}$ is returned HIGH. $\overline{RES}$ is required to be LOW for greater than 4 clock cycles and is internally synchronized. For proper initialization, the LOW-to-HIGH transition of $\overline{RES}$ must occur no sooner than 50 microseconds after power up. This input is provided with a Schmitt-trigger to facilitate power-on $\overline{RES}$ generation via an RC network. When $\overline{RES}$ occurs, the 80186 will drive the status lines to an inactive level for one clock, and then tri-state them.

Table 1. 80186 Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
$\overline{TEST}$	47	I	$\overline{TEST}$ is examined by the WAIT instruction. If the $\overline{TEST}$ input is HIGH when "WAIT" execution begins, instruction execution will suspend. $\overline{TEST}$ will be resampled until it goes LOW, at which time execution will resume. If interrupts are enabled while the 80186 is waiting for $\overline{TEST}$, interrupts will be serviced. This input is synchronized internally.
TMR IN 0, TMR IN1	20 21	I I	Timer Inputs are used either as clock or control signals, depending upon the programmed timer mode. These inputs are active HIGH (or LOW-to-HIGH transitions are counted) and internally synchronized.
TMR OUT 0, TMR OUT 1	22 23	O O	Timer outputs are used to provide single pulse or continuous waveform generation, depending upon the timer mode selected.
DRQ0 DRQ1	18 19	I I	DMA Request is driven HIGH by an external device when it desires that a DMA channel (Channel 0 or 1) perform a transfer. These signals are active HIGH, level-triggered, and internally synchronized.
NMI	46	I	Non-Maskable Interrupt is an edge-triggered input which causes a type 2 interrupt. NMI is not maskable internally. A transition from a LOW to HIGH initiates the interrupt at the next instruction boundary. NMI is latched internally. An NMI duration of one clock or more will guarantee service. This input is internally synchronized.
INT0, INT1, INT2/$\overline{INTA0}$ INT3/$\overline{INTA1}$	45,44 42 41	I I/O I/O	Maskable Interrupt Requests can be requested by strobing one of these pins. When configured as inputs, these pins are active HIGH. Interrupt Requests are synchronized internally. INT2 and INT3 may be configured via software to provide active-LOW interrupt-acknowledge output signals. All interrupt inputs may be configured via software to be either edge- or level-triggered. To ensure recognition, all interrupt requests must remain active until the interrupt is acknowleged. When iRMX mode is selected, the function of these pins changes (see Interrupt Controller section of this data sheet).
A19/S6, A18/S5, A17/S4, A16/S3	65–68	O O O O	Address Bus Outputs (16–19) and Bus Cycle Status (3–6) reflect the four most significant address bits during T_1. These signals are active HIGH. During T_2, T_3, T_W, and T_4, status information is available on these lines as encoded below: S3,S4, and S5 are defined as LOW during T_2–T_4.
AD15–AD0	10–17, 1–8	I/O	Address/Data Bus (0–15) signals constitute the time mutiplexed memory or I/O address (T_1) and data (T_2, T_3, T_W, and T_4) bus. The bus is active HIGH. A_0 is analogous to $\overline{BHE}$ for the lower byte of the data bus, pins D_7 through D_0. It is LOW during T_1 when a byte is to be transferred onto the lower portion of the bus in memory or I/O operations.
$\overline{BHE}$/S7	64	O	During T_1 the Bus High Enable signal should be used to determine if data is to be enabled onto the most significant half of the data bus, pins D_{15}–D_8. $\overline{BHE}$ is LOW during T_1 for read, write, and interrupt acknowledge cycles when a byte is to be transferred on the higher half of the bus. The S_7 status information is available during T_2, T_3, and T_4. S_7 is logically equivalent to $\overline{BHE}$. The signal is active LOW, and is tristated OFF during bus HOLD.

	Low	High
S6	Processor Cycle	DMA Cycle

$\overline{BHE}$ and A0 Encodings		
$\overline{BHE}$ Value	A0 Value	Function
0	0	Word Transfer
0	1	Byte Transfer on upper half of data bus (D15–D8)
1	0	Byte Transfer on lower half of data bus (D_7–D_0)
1	1	Reserved

Table 1. 80186 Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
ALE/QS0	61	O	Address Latch Enable/Queue Status 0 is provided by the 80186 to latch the address into the 8282/8283 address latches. ALE is active HIGH. Addresses are guaranteed to be valid on the trailing edge of ALE. The ALE rising edge is generated off the rising edge of the CLKOUT immediately preceding T_1 of the associated bus cycle, effectively one-half clock cycle earlier than in the standard 8086. The trailing edge is generated off the CLKOUT rising edge in T_1 as in the 8086. Note that ALE is never floated.
$\overline{WR}$/QS1	63	O	Write Strobe/Queue Status 1 indicates that the data on the bus is to be written into a memory or an I/O device. $\overline{WR}$ is active for T_2, T_3, and T_W of any write cycle. It is active LOW, and floats during "HOLD." It is driven HIGH for one clock during Reset, and then floated. When the 80186 is in queue status mode, the ALE/QS0 and $\overline{WR}$/QS1 pins provide information about processor/instruction queue interaction. QS1 / QS0 / Queue Operation 0 / 0 / No queue operation 0 / 1 / First opcode byte fetched from the queue 1 / 1 / Subsequent byte fetched from the queue 1 / 0 / Empty the queue
$\overline{RD}$/$\overline{QSMD}$	62	O	Read Strobe indicates that the 80186 is performing a memory or I/O read cycle. $\overline{RD}$ is active LOW for T_2, T_3, and T_W of any read cycle. It is guaranteed not to go LOW in T_2 until after the Address Bus is floated. $\overline{RD}$ is active LOW, and floats during "HOLD." $\overline{RD}$ is driven HIGH for one clock during Reset, and then the output driver is floated. A weak internal pull-up mechanism on the $\overline{RD}$ line hols it HIGH when the line is not driven. During RESET the pin is sampled to determine whether the 80186 should provide ALE, $\overline{WR}$, and $\overline{RD}$, or if the Queue-Status should be provided. $\overline{RD}$ should be connected to GND to provide Queue-Status data.
ARDY	55	I	Asynchronous Ready informs the 80186 that the addressed memory space or I/O device will complete a data transfer. The ARDY input pin will accept an asynchronous input, and is active HIGH. Only the rising edge is internally synchronized by the 80186. This means that the falling edge of ARDY must be synchronized to the 80186 clock. If connected to V_{CC}, no WAIT states are inserted. Asynchronous ready (ARDY) or synchronous ready (SRDY) must be active to terminate a bus cycle.
SRDY	49	I	Synchronous Ready must be synchronized externally to the 80186. The use of SRDY provides a relaxed system-timing specification on the Ready input. This is accomplished by eliminating the one-half clock cycle which is required for internally resolving the signal level when using the ARDY input. This line is active HIGH. If this line is connected to V_{CC}, no WAIT states are inserted. Asynchronous ready (ARDY) or synchronous ready (SRDY) must be active before a bus cycle is terminated.
$\overline{LOCK}$	48	O	$\overline{LOCK}$ output indicates that other system bus masters are not to gain control of the system bus while $\overline{LOCK}$ is active LOW. The $\overline{LOCK}$ signal is requested by the LOCK prefix instruction and is activated at the beginning of the first data cycle associated with the instruction following the LOCK prefix. It remains active until the completion of the instruction following the LOCK prefix. No prefetches will occur while $\overline{LOCK}$ is asserted. $\overline{LOCK}$ is active LOW, is driven HIGH for one clock during RESET, and then floated.

Table 1. 80186 Pin Description (Continued)

Symbol	Pin No.	Type	Name and Function
$\overline{S0}$,$\overline{S1}$,$\overline{S2}$	52–54	O	Bus cycle status $\overline{S0}$–$\overline{S2}$ are encoded to provide bus-transaction information: 80186 Bus Cycle Status Information $\overline{S2}$ / $\overline{S1}$ / $\overline{S0}$ / Bus Cycle Initiated 0 0 0 Interrupt Acknowledge 0 0 1 Read I/O 0 1 0 Write I/O 0 1 1 Halt 1 0 0 Instruction Fetch 1 0 1 Read Data from Memory 1 1 0 Write Data to Memory 1 1 1 Passive (no bus cycle) The status pins float during "HOLD." $\overline{S2}$ may be used as a logical M/$\overline{IO}$ indicator, and $\overline{S1}$ as a DT/$\overline{R}$ indicator. The status lines are driven HIGH for one clock during Reset, and then floated until a bus cycle begins.
HOLD (input) HLDA (output)	50 51	I O	HOLD indicates that another bus master is requesting the local bus. The HOLD input is active HIGH. HOLD may be asynchronous with respect to the 80186 clock. The 80186 will issue a HLDA in response to a HOLD request at the end of T_4 or T_I. Simultaneous with the issuance of HLDA, the 80186 will float the local bus and control lines. After HOLD is detected as being LOW, the 80186 will lower HLDA. When the 80186 needs to run another bus cycle, it will again drive the local bus and control lines.
$\overline{UCS}$	34	O	Upper Memory Chip Select is an active LOW output whenever a memory reference is made to the defined upper portion (1K–256K block) of memory. This line is not floated during bus HOLD. The address range activating $\overline{UCS}$ is software programmable.
$\overline{LCS}$	33	O	Lower Memory Chip Select is active LOW whenever a memory reference is made to the defined lower portion (1K–256K) of memory. This line is not floated during bus HOLD. The address range activating $\overline{LCS}$ is software programmable.
$\overline{MCS0}$–3	38,37,36,35	O	Mid-Range Memory Chip Select signals are active LOW when a memory reference is made to the defined mid-range portion of memory (8K–512K). These lines are not floated during bus HOLD. The address ranges activating $\overline{MCS0}$–3 are software programmable.
$\overline{PCS0}$–4	25,27–30	O	Peripheral Chip Select signals 0–4 are active LOW when a reference is made to the defined peripheral area (64K byte I/O space). These lines are not floated during bus HOLD. The address ranges activating $\overline{PCS0}$–4 are software programmable.
$\overline{PCS5}$/A1	31	O	Peripheral Chip Select 5 or Latched A1 may be programmed to provide a sixth peripheral chip select, or to provide an internally latched A1 signal. The address range activating $\overline{PCS5}$ is software programmable. When programmed to provide latched A1, rather than $\overline{PCS5}$, this pin will retain the previously latched value of A1 during a bus HOLD. A1 is active HIGH.
$\overline{PCS6}$/A2	32	O	Peripheral Chip Select 6 or Latched A2 may be programmed to provide a seventh peripheral chip select, or to provide an internally latched A2 signal. The address range activating $\overline{PCS6}$ is software programmable. When programmed to provide latched A2, rather than $\overline{PCS6}$, this pin will retain the previously latched value of A2 during a bus HOLD. A2 is active HIGH.
DT/$\overline{R}$	40	O	Data Transmit/Receive controls the direction of data flow through the external 8286/8287 data bus transceiver. When LOW, data is transferred to the 80186. When HIGH the 80186 places write data on the data bus.
$\overline{DEN}$	39	O	Data Enable is provided as an 8286/8287 data bus transceiver output enable. $\overline{DEN}$ is active LOW during each memory and I/O access. $\overline{DEN}$ is HIGH whenever DT/$\overline{R}$ changes state.

FUNCTIONAL DESCRIPTION

Introduction

The following Functional Description describes the base architecture of the iAPX 186. This architecture is common to the iAPX 86, 88, and 286 microprocessor families as well. The iAPX 186 is a very high integration 16-bit microprocessor. It combines 15–20 of the most common microprocessor system components onto one chip while providing twice the performance of the standard iAPX 86. The 80186 is object code compatible with the iAPX 86, 88 microprocessors and adds 10 new instruction types to the existing iAPX 86, 88 instruction set.

iAPX 186 BASE ARCHITECTURE

The iAPX 86, 88, 186, and 286 family all contain the same basic set of registers, instructions, and addressing modes. The 80186 processor is upward compatible with the 8086, 8088, and 80286 CPUs.

Register Set

The 80186 base architecture has fourteen registers as shown in Figures 3a and 3b. These registers are grouped into the following categories.

General Registers

Eight 16-bit general purpose registers used to contain arithmetic and logical operands. Four of these (AX, BX, CX, and DX) can be used as 16-bit registers or split into pairs of separate 8-bit registers.

Segment Registers

Four 16-bit special purpose registers select, at any given time, the segments of memory that are immediately addressable for code, stack, and data. (For usage, refer to Memory Organization.)

Base and Index Registers

Four of the general purpose registers may also be used to determine offset addresses of operands in memory. These registers may contain base addresses or indexes to particular locations within a segment. The addressing mode selects the specific registers for operand and address calculations.

Status and Control Registers

Two 16-bit special purpose registers record or alter certain aspects of the 80186 processor state. These are the Instruction Pointer Register, which contains the offset address of the next sequential instruction to be executed, and the Status Word Register, which contains status and control flag bits (see Figures 3a and 3b).

Status Word Description

The Status Word records specific characteristics of the result of logical and arithmetic instructions (bits 0, 2, 4, 6, 7, and 11) and controls the operation of the 80186 within a given operating mode (bits 8, 9, and 10). The Status Word Register is 16-bits wide. The function of the Status Word bits is shown in Table 2.

Figure 3a. 80186 General Purpose Register Set

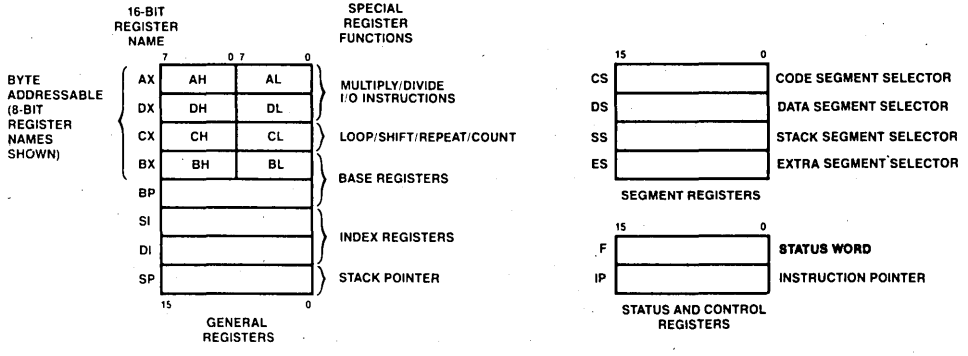


Figure 3b. Status Word Format

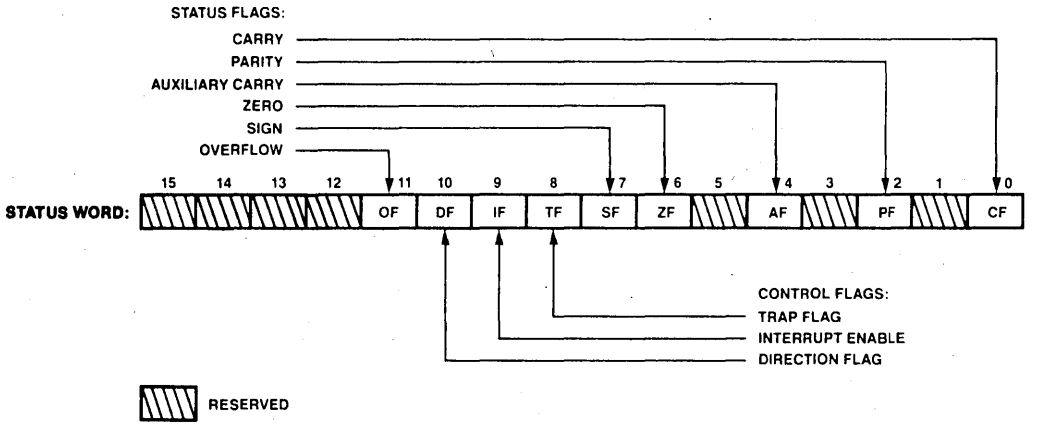


Table 2. Status Word Bit Functions

Bit Position	Name	Function
0	CF	Carry Flag—Set on high-order bit carry or borrow; cleared otherwise
2	PF	Parity Flag—Set if low-order 8 bits of result contain an even number of 1-bits; cleared otherwise
4	AF	Set on carry from or borrow to the low order four bits of AL; cleared otherwise
6	ZF	Zero Flag–Set if result is zero; cleared otherwise
7	SF	Sign Flag—Set equal to high-order bit of result (0 if positive, 1 if negative)
8	TF	Single Step Flag—Once set, a single step interrupt occurs after the next instruction executes. TF is cleared by the single step interrupt.
9	IF	Interrupt-enable Flag—When set, maskable interrupts will cause the CPU to transfer control to an interrupt vector specified location.
10	DF	Direction Flag—Causes string instructions to auto decrement the appropriate index register when set. Clearing DF causes auto increment.
11	OF	Overflow Flag—Set if the signed result cannot be expressed within the number of bits in the destination operand; cleared otherwise

Instruction Set

The instruction set is divided into seven categories: data transfer, arithmetic, shift/rotate/logical, string manipulation, control transfer, high-level instructions, and processor control. These categories are summarized in Figure 4.

An 80186 instruction can reference anywhere from zero to several operands. An operand can reside in a register, in the instruction itself, or in memory. Specific operand addressing modes are discussed later in this data sheet.

Memory Organization

Memory is organized in sets of segments. Each segment is a linear contiguous sequence of up to 64K (2^{16}) 8-bit bytes. Memory is addressed using a two-component address (a pointer) that consists of a 16-bit base segment and a 16-bit offset. The 16-bit base values are contained in one of four internal segment registers (code, data, stack, extra). The physical address is calculated by shifting the base value LEFT by four bits and adding the 16-bit offset value to yield a 20-bit physical address (see Figure 5). This allows for a 1 MByte physical address size.

All instructions that address operands in memory must specify the base segment and the 16-bit offset value. For speed and compact instruction encoding, the segment register used for physical address generation is implied by the addressing mode used (see Table 3). These rules follow the way programs are written (see Figure 6) as independent modules that require areas for code and data, a stack, and access to external data areas.

Special segment override instruction prefixes allow the implicit segment register selection rules to be overridden for special cases. The stack, data, and extra segments may coincide for simple programs.

Figure 4. iAPX 186 Instruction Set

GENERAL PURPOSE	
MOV	Move byte or word
PUSH	Push word onto stack
POP	Pop word off stack
PUSHA	Push all registers on stack
POPA	Pop all registers from stack
XCHG	Exchange byte or word
XLAT	Translate byte
INPUT/OUTPUT	
IN	Input byte or word
OUT	Output byte or word
ADDRESS OBJECT	
LEA	Load effective address
LDS	Load pointer using DS
LES	Load pointer using ES
FLAG TRANSFER	
LAHF	Load AH register from flags
SAHF	Store AH register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack

ADDITION	
ADD	Add byte or word
ADC	Add byte or word with carry
INC	Increment byte or word by 1
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition
SUBTRACTION	
SUB	Subtract byte or word
SBB	Subtract byte or word with borrow
DEC	Decrement byte or word by 1
NEG	Negate byte or word
CMP	Compare byte or word
AAS	ASCII adjust for subtraction
DAS	Decimal adjust for subtraction
MULTIPLICATION	
MUL	Multiply byte or word unsigned
IMUL	Integer multiply byte or word
AAM	ASCII adjust for multiply
DIVISION	
DIV	Divide byte or word unsigned
IDIV	Integer divide byte or word
AAD	ASCII adjust for division
CBW	Convert byte to word
CWD	Convert word to doubleword

MOVS	Move byte or word string
INS	Input bytes or word string
OUTS	Output bytes or word string
CMPS	Compare byte or word string
SCAS	Scan byte or word string
LODS	Load byte or word string
STOS	Store byte or word string
REP	Repeat
REPE/REPZ	Repeat while equal/zero
REPNE/REPNZ	Repeat while not equal/not zero

LOGICALS	
NOT	"Not" byte or word
AND	"And" byte or word
OR	"Inclusive or" byte or word
XOR	"Exclusive or" byte or word
TEST	"Test" byte or word
SHIFTS	
SHL/SAL	Shift logical/arithmetic left byte or word
SHR	Shift logical right byte or word
SAR	Shift arithmetic right byte or word
ROTATES	
ROL	Rotate left byte or word
ROR	Rotate right byte or word
RCL	Rotate through carry left byte or word
RCR	Rotate through carry right byte or word

FLAG OPERATIONS	
STC	Set carry flag
CLC	Clear carry flag
CMC	Complement carry flag
STD	Set direction flag
CLD	Clear direction flag
STI	Set interrupt enable flag
CLI	Clear interrupt enable flag
EXTERNAL SYNCHRONIZATION	
HLT	Halt until interrupt or reset
WAIT	Wait for $\overline{\text{TEST}}$ pin active
ESC	Escape to extension processor
LOCK	Lock bus during next instruction
NO OPERATION	
NOP	No operation
HIGH LEVEL INSTRUCTIONS	
ENTER	Format stack for procedure entry
LEAVE	Restore stack for procedure exit
BOUND	Detects values outside prescribed range

All mnemonics copyright Intel Corp.

Figure 4. iAPX 186 Instruction Set (continued)

CONDITIONAL TRANSFERS		UNCONDITIONAL TRANSFERS	
JA/JNBE	Jump if above/not below nor equal	CALL	Call procedure
JAE/JNB	Jump if above or equal/not below	RET	Return from procedure
JB/JNAE	Jump if below/not above nor equal	JMP	Jump
JBE/JNA	Jump if below or equal/not above		
JC	Jump if carry	**ITERATION CONTROLS**	
JE/JZ	Jump if equal/zero		
JG/JNLE	Jump if greater/not less nor equal	LOOP	Loop
JGE/JNL	Jump if greater or equal/not less	LOOPE/LOOPZ	Loop if equal/zero
JL/JNGE	Jump if less/not greater nor equal	LOOPNE/LOOPNZ	Loop if not equal/not zero
JLE/JNG	Jump if less or equal/not greater	JCXZ	Jump if register CX = 0
JNC	Jump if not carry		
JNE/JNZ	Jump if not equal/not zero	**INTERRUPTS**	
JNO	Jump if not overflow		
JNP/JPO	Jump if not parity/parity odd	INT	Interrupt
JNS	Jump if not sign	INTO	Interrupt if overflow
JO	Jump if overflow	IRET	Interrupt return
JP/JPE	Jump if parity/parity even		
JS	Jump if sign		

All mnemonics copyright Intel Corp.

To access operands that do not reside in one of the four immediately available segments, a full 32-bit pointer can be used to reload both the base (segment) and offset values.

Figure 5. Two Component Address

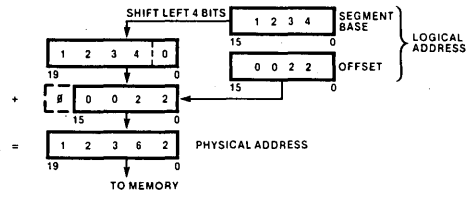
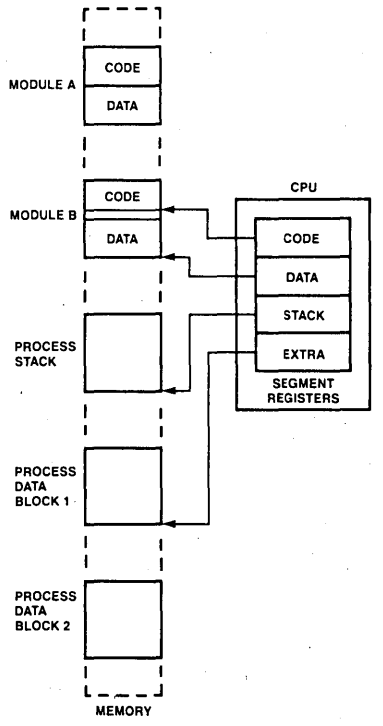


Table 3. Segment Register Selection Rules

Memory Reference Needed	Segment Register Used	Implicit Segment Selection Rule
Instructions	Code (CS)	Instruction prefetch and immediate data.
Stack	Stack (SS)	All stack pushes and pops; any memory references which use BP Register as a base register.
External Data (Global)	Extra (ES)	All string instruction references which use the DI register as an index.
Local Data	Data (DS)	All other data references.

Figure 6. Segmented Memory Helps Structure Software



Addressing Modes

The 80186 provides eight categories of addressing modes to specify operands. Two addressing modes are provided for instructions that operate on register or immediate operands:

- *Register Operand Mode:* The operand is located in one of the 8- or 16-bit general registers.
- *Immediate Operand Mode:* The operand is included in the instruction.

Six modes are provided to specify the location of an operand in a memory segment. A memory operand address consists of two 16-bit components: a segment base and an offset. The segment base is supplied by a 16-bit segment register either implicity chosen by the addressing mode or explicitly chosen by a segment override prefix. The offset, also called the effective address, is calculated by summing any combination of the following three address elements:

- the *displacement* (an 8- or 16-bit immediate value contained in the instruction);
- the *base* (contents of either the BX or BP base registers); and
- the *index* (contents of either the SI or DI index registers).

Any carry out from the 16-bit addition is ignored. Eight-bit displacements are sign extended to 16-bit values.

Combinations of these three address elements define the six memory addressing modes, described below.

- *Direct Mode:* The operand's offset is contained in the instruction as an 8- or 16-bit displacement element.
- *Register Indirect Mode:* The operand's offset is in one of the registers SI, DI, BX, or BP.
- *Based Mode:* The operand's offset is the sum of an 8- or 16-bit displacement and the contents of a base register (BX or BP).
- *Indexed Mode:* The operand's offset is the sum of an 8- or 16-bit displacement and the contents of an index register (SI or DI).
- *Based Indexed Mode:* The operand's offset is the sum of the contents of a base register and an index register.
- *Based Indexed Mode with Displacement:* The operand's offset is the sum of a base register's contents, an index register's contents, and an 8- or 16-bit displacement.

Data Types

The 80186 directly supports the following data types:

- *Integer:* A signed binary numeric value contained in an 8-bit byte or a 16-bit word. All operations assume a 2's complement representation. Signed 32 and 64 bit integers are supported using the 8087 Numeric Data Processor.
- *Ordinal:* An unsigned binary numeric value contained in an 8-bit byte or a 16-bit word.
- *Pointer:* A 16- or 32-bit quantity, composed of a 16-bit offset component or a 16-bit segment base component in addition to a 16-bit offset component.
- *String:* A contiguous sequence of bytes or words. A string may contain from 1K to 64K bytes.
- *ASCII:* A byte representation of alphanumeric and control characters using the ASCII standard of character representation.
- *BCD:* A byte (unpacked) representation of the decimal digits 0–9.
- *Packed BCD:* A byte (packed) representation of two decimal digits (0–9). One digit is stored in each nibble (4-bits) of the byte.
- *Floating Point:* A signed 32-, 64-, or 80-bit real number representation. (Floating point operands are supported using the iAPX 186/8087 Numeric Data Processor configuration.)

In general, individual data elements must fit within defined segment limits. Figure 7 graphically represents the data types supported by the iAPX 186.

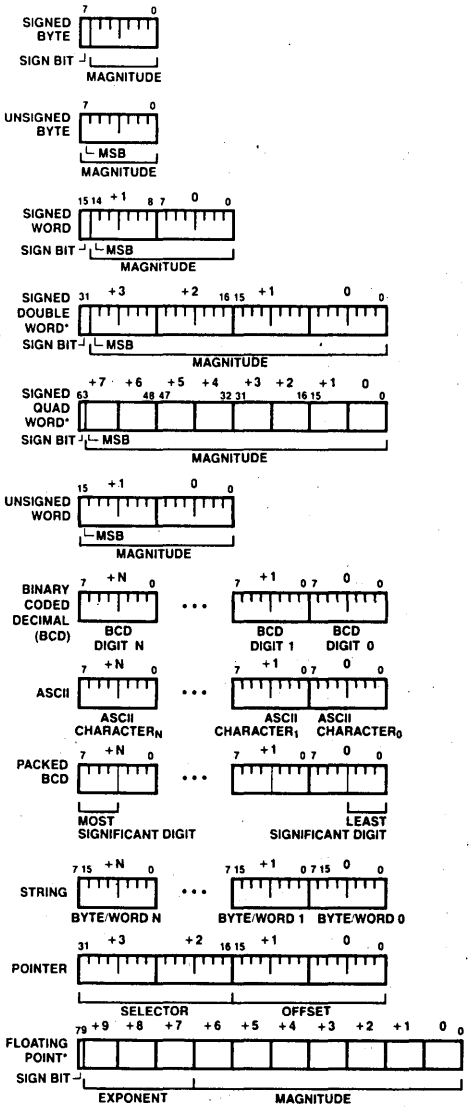
I/O Space

The I/O space consists of 64K 8-bit or 32K 16-bit ports. Separate instructions address the I/O space with either an 8-bit port address, specified in the instruction, or a 16-bit port address in the DX register. 8-bit port addresses are zero extended such that A_{15}-A_8 are LOW. I/O port addresses 00F8(H) through 00FF(H) are reserved.

Interrupts

An interrupt transfers execution to a new program location. The old program address (CS:IP) and machine state (Status Word) are saved on the stack to allow resumption of the interrupted program. Interrupts fall into three classes: hardware initiated, INT instructions, and instruction exceptions. Hardware initiated interrupts occur in response to an external input and are classified as non-maskable or maskable.

Figure 7. iAPX 186 Supported Data Types



NOTE:
*SUPPORTED BY iAPX 186 WITH A NUMERIC DATA PROCESSOR.

Programs may cause an interrupt with an INT instruction. Instruction exceptions occur when an unusual condition, which prevents further instruction processing, is detected while attempting to execute an instruction. If the exception was caused by executing an ESC instruction with the ESC trap bit set in the relocation register, the return instruction will point to the ESC instruction, or to the segment override prefix immediately preceding the ESC instruction if the prefix was present. In all other cases, the return address from an exception will point at the instruction immediately following the instruction causing the exception.

A table containing up to 256 pointers defines the proper interrupt service routine for each interrupt. Interrupts 0–31, some of which are used for instruction exceptions, are reserved. Table 4 shows the 80186 predefined types and default priority levels. For each interrupt, an 8-bit vector must be supplied to the 80186 which identifies the appropriate table entry. Exceptions supply the interrupt vector internally. In addition, internal peripherals and non-cascaded external interrupts will generate their own vectors through the internal interrupt controller. INT instructions contain or imply the vector and allow access to all 256 interrupts. Maskable hardware initiated interrupts supply the 8-bit vector to the CPU during an interrupt acknowledge bus sequence. Non-maskable hardware interrupts use a predefined internally supplied vector.

Interrupt Sources

The 80186 can service interrupts generated by software or hardware. The software interrupts are generated by specific instructions (INT, ESC, unused OP, etc.) or the results of conditions specified by instructions (array bounds check, INT0, DIV, IDIV, etc.). All interrupt sources are serviced by an indirect call through an element of a vector table. This vector table is indexed by using the interrupt vector type (Table 4), multiplied by four. All hardware-generated interrupts are sampled at the end of each instruction. Thus, the software interrupts will begin service first. Once the service routine is entered and interrupts are enabled, any hardware source of sufficient priority can interrupt the service routine in progress.

The software generated 80186 interrupts are described below.

DIVIDE ERROR EXCEPTION (TYPE 0)
Generated when a DIV or IDIV instruction quotient cannot be expressed in the number of bits in the destination.

Table 4. 80186 Interrupt Vectors

Interrupt Name	Vector Type	Default Priority	Related Instructions
Divide Error Exception	0	*1	DIV, IDIV
Single Step Interrupt	1	12**2	All
NMI	2	1	All
Breakpoint Interrupt	3	*1	INT
INT0 Detected Overflow Exception	4	*1	INT0
Array Bounds Exception	5	*1	BOUND
Unused-Opcode Exception	6	*1	Undefined Opcodes
ESC Opcode Exception	7	*1***	ESC Opcodes
Timer 0 Interrupt	8	2A****	
Timer 1 Interrupt	18	2B****	
Timer 2 Interrupt	19	2C****	
Reserved	9	3	
DMA 0 Interrupt	10	4	
DMA 1 Interrupt	11	5	
INT0 Interrupt	12	6	
INT1 Interrupt	13	7	
INT2 Interrupt	14	8	
INT3 Interrupt	15	9	

NOTES:

*1. These are generated as the result of an instruction execution.

**2. This is handled as in the 8086.

****3. All three timers constitute one source of request to the interrupt controller. The Timer interrupts all have the same default priority level with respect to all other interrupt sources. However, they have a defined priority ordering amongst themselves. (Priority 2A is higher priority than 2B.) Each Timer interrupt has a separate vector type number.

4. Default priorities for the interrupt sources are used only if the user does not program each source into a unique priority level.

***5. An escape opcode will cause a trap only if the proper bit is set in the peripheral control block relocation register.

SINGLE-STEP INTERRUPT (TYPE 1)
Generated after most instructions if the TF flag is set. Interrupts will not be generated after prefix instructions (e.g., REP), instructions which modify segment registers (e.g., POP DS), or the WAIT instruction.

NON-MASKABLE INTERRUPT—NMI (TYPE 2)
An external interrupt source which cannot be masked.

BREAKPOINT INTERRUPT (TYPE 3)
A one-byte version of the INT instruction. It uses 12 as an index into the service routine address table (because it is a type 3 interrupt).

INT0 DETECTED OVERFLOW EXCEPTION (TYPE 4)
Generated during an INT0 instruction if the 0F bit is set.

ARRAY BOUNDS EXCEPTION (TYPE 5)
Generated during a BOUND instruction if the array index is outside the array bounds. The array bounds are located in memory at a location indicated by one of the instruction operands. The other operand indicates the value of the index to be checked.

UNUSED OPCODE EXCEPTION (TYPE 6)
Generated if execution is attempted on undefined opcodes.

ESCAPE OPCODE EXCEPTION (TYPE 7)
Generated if execution is attempted of ESC opcodes (D8H–DFH). This exception will only be generated if a bit in the relocation register is set. The return address of this exception will point to the ESC instruction causing the exception. If a segment override prefix preceded the ESC instruction, the return address will point to the segment override prefix.

Hardware-generated interrupts are divided into two groups: maskable interrupts and non-maskable interrupts. The 80186 provides maskable hardware interrupt request pins INT0–INT3. In addition, maskable interrupts may be generated by the 80186 integrated DMA controller and the integrated timer unit. The vector types for these interrupts is shown in Table 4. Software enables these inputs by setting the interrupt flag bit (IF) in the Status Word. The interrupt controller is discussed in the peripheral section of this data sheet.

Further maskable interrupts are disabled while servicing an interrupt because the IF bit is reset as part of the response to an interrupt or exception. The saved Status Word will reflect the enable status of the processor prior to the interrupt. The interrupt flag will remain zero unless specifically set. The interrupt return instruction restores the Status Word, thereby restoring the original status of IF bit. If the interrupt return re-enables interrupts, and another interrupt is pending, the 80186 will immediately service the highest-priority interrupt pending, i.e., no instructions of the main line program will be executed.

Non-Maskable Interrupt Request (NMI)

A non-maskable interrupt (NMI) is also provided. This interrupt is serviced regardless of the state of the IF bit. A typical use of NMI would be to activate a power failure routine. The activation of this input

causes an interrupt with an internally supplied vector value of 2. No external interrupt acknowledge sequence is performed. The IF bit is cleared at the beginning of an NMI interrupt to prevent maskable interrupts from being serviced.

Single-Step Interrupt

The 80186 has an internal interrupt that allows programs to execute one instruction at a time. It is called the single-step interrupt and is controlled by the single-step flag bit (TF) in the Status Word. Once this bit is set, an internal single-step interrupt will occur after the next instruction has been executed. The interrupt clears the TF bit and uses an internally supplied vector of 1. The IRET instruction is used to set the TF bit and transfer control to the next instruction to be single-stepped.

Initialization and Processor Reset

Processor initialization or startup is accomplished by driving the $\overline{RES}$ input pin LOW. $\overline{RES}$ forces the 80186 to terminate all execution and local bus activity. No instruction or bus activity will occur as long as $\overline{RES}$ is active. After $\overline{RES}$ becomes inactive and an internal processing interval elapses, the 80186 begins execution with the instruction at physical location FFFF0(H). $\overline{RES}$ also sets some registers to predefined values as shown in Table 5.

Table 5. 80186 Initial Register State after RESET

Status Word	F002(H)
Instruction Pointer	0000(H)
Code Segment	FFFF(H)
Data Segment	0000(H)
Extra Segment	0000(H)
Stack Segment	0000(H)

iAPX 186 CLOCK GENERATOR

The iAPX 186 provides an on-chip clock generator for both internal and external clock generation. The clock generator features a crystal oscillator, a divide-by-two counter, synchronous and asynchronous ready inputs, and reset circuitry.

Oscillator

The oscillator circuit of the iAPX 186 is designed to be used with a parallel resonant fundamental mode crystal. This is used as the time base for the iAPX 186. The crystal frequency selected will be double the CPU clock frequency. Use of an LC or RC circuit is not recommended with this oscillator. If an external oscillator is used, it can be connected directly to input pin X1 in lieu of a crystal. The output of the oscillator is not directly available outside the iAPX 186. The recommended crystal configuration is shown in Figure 8.

Clock Generator

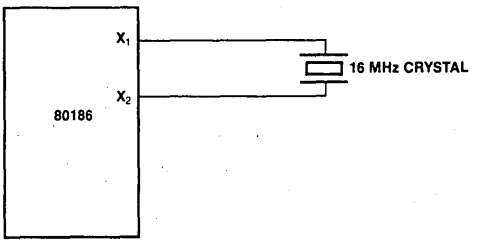
The iAPX 186 clock generator provides the 50% duty cycle processor clock for the iAPX 186. It does this by dividing the oscillator output by 2 forming the symmetrical clock. If an external oscillator is used, the state of the clock generator will change on the falling edge of the oscillator signal. The CLKOUT pin provides the processor clock signal for use outside the iAPX 186. This may be used to drive other system components. All timings are referenced to the output clock.

READY Synchronization

The iAPX 186 provides both synchronous and asynchronous ready inputs. Asynchronous ready synchronization is accomplished by circuitry which samples ARDY in the middle of T_2 and again in the middle of each T_W until ARDY is sampled HIGH. One-half CLKOUT cycle of resolution time is used. Full synchronization is performed only on the rising edge of ARDY, i.e., the falling edge of ARDY must be synchronized to the CLKOUT signal if it will occur during T_2 or T_W. HIGH-to-LOW transitions of ARDY must be performed synchronously to the CPU clock.

A second ready input (SRDY) is provided to interface with externally synchronized ready signals. This input is sampled at the end of T_2 and again at the end of each T_W until it is sampled HIGH. By using this input rather than the asynchronous ready input, the half-clock cycle resolution time penalty is eliminated.

Figure 8. Recommended iAPX 186 Crystal Configuration



This input must satisfy set-up and hold times to guarantee proper operation of the circuit.

In addition, the iAPX 186, as part of the integrated chip-select logic, has the capability to program WAIT states for memory and peripheral blocks. This is discussed in the Chip Select/Ready Logic description.

RESET Logic

The iAPX 186 provides both a $\overline{RES}$ input pin and a synchronized RESET pin for use with other system components. The $\overline{RES}$ input pin on the iAPX 186 is provided with hysteresis in order to facilitate power-on Reset generation via an RC network. RESET is guaranteed to remain active for at least five clocks given a $\overline{RES}$ input of at least six clocks. RESET may be delayed up to two and one-half clocks behind $\overline{RES}$.

Multiple iAPX 186 processors may be synchronized through the $\overline{RES}$ input pin, since this input resets both the processor and divide-by-two internal counter in the clock generator. In order to insure that the divide-by-two counters all begin counting at the same time, the active going edge of $\overline{RES}$ must satisfy a 25 ns setup time before the falling edge of the 80186 clock input. In addition, in order to insure that all CPUs begin executing in the same clock cycle, the reset must satisfy a 25 ns setup time before the rising edge of the CLKOUT signal of all the processors.

LOCAL BUS CONTROLLER

The iAPX 186 provides a local bus controller to generate the local bus control signals. In addition, it employs a HOLD/HLDA protocol for relinquishing the local bus to other bus masters. It also provides control lines that can be used to enable external buffers and to direct the flow of data on and off the local bus.

Memory/Peripheral Control

The iAPX 186 provides ALE, $\overline{RD}$, and $\overline{WR}$ bus control signals. The $\overline{RD}$ and $\overline{WR}$ signals are used to strobe data from memory to the iAPX 186 or to strobe data from the iAPX 186 to memory. The ALE line provides a strobe to address latches for the multiplexed address/data bus. The iAPX 186 local bus controller does not provide a memory/$\overline{I/O}$ signal. If this is required, the user will have to use the $\overline{S2}$ signal (which will require external latching), make the memory and I/O spaces nonoverlapping, or use only the integrated chip-select circuitry.

Transceiver Control

The iAPX 186 generates two control signals to be connected to 8286/8287 transceiver chips. This capability allows the addition of transceivers for extra buffering without adding external logic. These control lines, DT/$\overline{R}$ and $\overline{DEN}$, are generated to control the flow of data through the transceivers. The operation of these signals is shown in Table 6.

Table 6. Transceiver Control Signals Description

Pin Name	Function
$\overline{DEN}$ (Data Enable)	Enables the output drivers of the transceivers. It is active LOW during memory, I/O, or INTA cycles.
DT/$\overline{R}$ (Data Transmit/ Receive)	Determines the direction of travel through the transceivers. A HIGH level directs data away from the processor during write operations, while a LOW level directs data toward the processor during a read operation.

Local Bus Arbitration

The iAPX 186 uses a HOLD/HLDA system of local bus exchange. This provides an asynchronous bus exchange mechanism. This means multiple masters utilizing the same bus can operate at separate clock frequencies. The iAPX 186 provides a single HOLD/HLDA pair through which all other bus masters may gain control of the local bus. This requires external circuitry to arbitrate which external device will gain control of the bus from the iAPX 186 when there is more than one alternate local bus master. When the iAPX 186 relinquishes control of the local bus, it floats $\overline{DEN}$, $\overline{RD}$, $\overline{WR}$, $\overline{S0}$–$\overline{S2}$, $\overline{LOCK}$, AD0–AD15, A16–A19, $\overline{BHE}$, and DT/$\overline{R}$ to allow another master to drive these lines directly.

The iAPX 186 HOLD latency time, i.e., the time between HOLD request and HOLD acknowledge, is a function of the activity occurring in the processor when the HOLD request is received. A HOLD request is the highest-priority activity request which the processor may receive: higher than instruction fetching or internal DMA cycles. However, if a DMA cycle is in progress, the iAPX 186 will complete the transfer before relinquishing the bus. This implies that if a HOLD request is received just as a DMA transfer begins, the HOLD latency time can be as great as 4 bus cycles. This will occur if a DMA word transfer operation is taking place from an odd address to an odd address. This is a total of 16 clocks or more, if WAIT states are required. In addition, if locked transfers are performed, the HOLD latency time will be increased by the length of the locked transfer.

Local Bus Controller and Reset

Upon receipt of a RESET pulse from the $\overline{RES}$ input, the local bus controller will perform the following actions:

- Drive $\overline{DEN}$, $\overline{RD}$, and $\overline{WR}$ HIGH for one clock cycle, then float.

  NOTE: $\overline{RD}$ is also provided with an internal pull-up device to prevent the processor from inadvertently entering Queue Status mode during reset.

- Drive $\overline{S0}$–$\overline{S2}$ to the passive state (all HIGH) and then float.
- Drive $\overline{LOCK}$ HIGH and then float.
- Tristate AD0–15, A16–19, $\overline{BHE}$, DT/$\overline{R}$.
- Drive ALE LOW (ALE is never floated).
- Drive HLDA LOW.

INTERNAL PERIPHERAL INTERFACE

All the iAPX 186 integrated peripherals are controlled via 16-bit registers contained within an internal 256-byte control block. This control block may be mapped into either memory or I/O space. Internal logic will recognize the address and respond to the bus cycle. During bus cycles to internal registers, the bus controller will signal the operation externally (i.e., the $\overline{RD}$, $\overline{WR}$, status, address, data, etc., lines will be driven as in a normal bus cycle), but D_{15-0}, SRDY, and ARDY will be ignored. The base address of the control block must be on an even 256-byte boundary (i.e., the lower 8 bits of the base address are all zeros). All of the defined registers within this control block may be read or written by the 80186 CPU at any time. The location of any register contained within the 256-byte control block is determined by the current base address of the control block.

The control block base address is programmed via a 16-bit relocation register contained within the control block at offset FEH from the base address of the control block (see Figure 9). It provides the upper 12 bits of the base address of the control block. Note that mapping the control register block into an address range corresponding to a chip-select range is not recommended (the chip select circuitry is discussed later in this data sheet). In addition, bit 12 of this register determines whether the control block will be mapped into I/O or memory space. If this bit is 1, the control block will be located in memory space, whereas if the bit is 0, the control block will be located in I/O space. If the control register block is mapped into I/O space, the upper 4 bits of the base address must be programmed as 0 (since I/O addresses are only 16 bits wide).

In addition to providing relocation information for the control block, the relocation register contains bits which place the interrupt controller into iRMX mode, and cause the CPU to interrupt upon encountering ESC instructions. At RESET, the relocation register is set to 20FFH. This causes the control block to start at FF00H in I/O space. An offset map of the 256-byte control register block is shown in Figure 10.

The integrated iAPX 186 peripherals operate semi-autonomously from the CPU. Access to them for the most part is via software read/write of the control and data locations in the control block. Most of these registers can be both read and written. A few dedicated lines, such as interrupts and DMA request provide real-time communication between the CPU and peripherals as in a more conventional system utilizing discrete peripheral blocks. The overall interaction and function of the peripheral blocks has not substantially changed.

CHIP-SELECT/READY GENERATION LOGIC

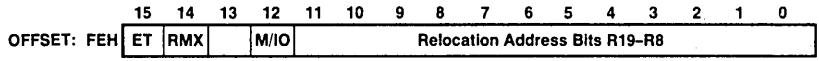
The iAPX 186 contains logic which provides programmable chip-select generation for both memories and peripherals. In addition, it can be programmed to provide READY (or WAIT state) generation. It can also provide latched address bits A1 and A2. The chip-select lines are active for all memory and I/O cycles in their programmed areas, whether they be generated by the CPU or by the integrated DMA unit.

Memory Chip Selects

The iAPX 186 provides 6 memory chip select outputs for 3 address areas: upper memory, lower memory, and midrange memory. One each is provided for upper memory and lower memory, while four are provided for midrange memory.

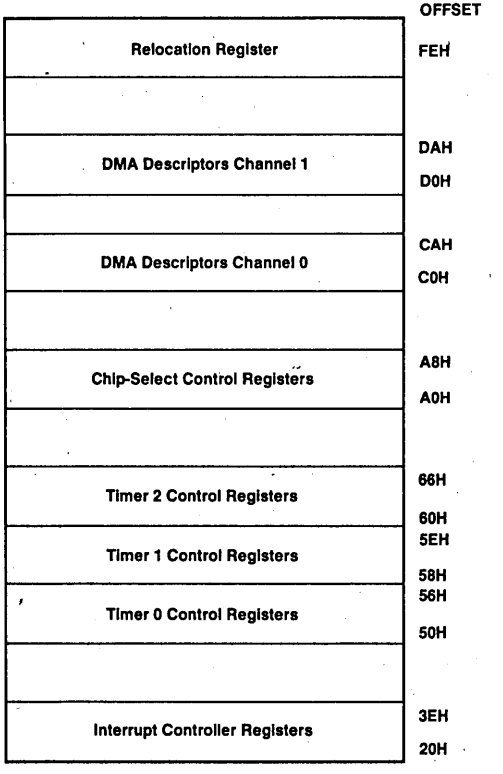
The range for each chip select is user-programmable and can be set to 2K, 4K, 8K, 16K, 32K, 64K, 128K (plus 1K and 256K for upper and lower chip selects). In addition, the beginning or base address of the midrange memory chip select may also be selected. Only one chip select may be programmed to be active for any memory location at a time. All chip select sizes are in bytes, whereas iAPX 186 memory is arranged in words. This means that if, for example, 16 64K x 1 memories are used, the memory block size will be 128K, not 64K.

Figure 9. Relocation Register



ET = ESC Trap / No ESC Trap (1/0)
M/IO = Register block located in Memory / I/O Space (1/0)
RMX = Normal Interrupt Controller mode / iRMX compatible Interrupt Controller mode (0/1)

Figure 10. Internal Register Map



Upper Memory $\overline{CS}$

The iAPX 186 provides a chip select, called $\overline{UCS}$, for the top of memory. The top of memory is usually used as the system memory because after reset the iAPX 186 begins executing at memory location FFFF0H.

The upper limit of memory defined by this chip select is always FFFFFH, while the lower limit is programmable. By programming the lower limit, the size of the select block is also defined. Table 7 shows the relationship between the base address selected and the size of the memory block obtained.

Table 7. UMCS Programming Values

Starting Address (Base Address)	Memory Block Size	UMCS Value (Assuming R0=R1=R2=0)
FFC00	1K	FFF8H
FF800	2K	FFB8H
FF000	4K	FF38H
FE000	8K	FE38H
FC000	16K	FC38H
F8000	32K	F838H
F0000	64K	F038H
E0000	128K	E038H
C0000	256K	C038H

The lower limit of this memory block is defined in the UMCS register (see Figure 11). This register is at offset A0H in the internal control block. The legal values for bits 6–13 and the resulting starting address and memory block sizes are given in Table 7. Any combination of bits 6–13 not shown in Table 7 will result in undefined operation. After reset, the UMCS register is programmed for a 1K area. It must be reprogrammed if a larger upper memory area is desired.

Any internally generated 20-bit address whose upper 16 bits are greater than or equal to UMCS (with bits 0–5 "0") will cause UCS to be activated. UMCS bits R2–R0 are used to specify READY mode for the area of memory defined by this chip-select register, as explained below.

Lower Memory $\overline{CS}$

The iAPX 186 provides a chip select for low memory called $\overline{LCS}$. The bottom of memory contains the interrupt vector table, starting at location 00000H.

The lower limit of memory defined by this chip select is always 0H, while the upper limit is programmable. By programming the upper limit, the size of the memory block is also defined. Table 8 shows the relationship between the upper address selected and the size of the memory block obtained.

Table 8. LMCS Programming Values

Upper Address	Memory Block Size	LMCS Value (Assuming R0=R1=R2=0)
003FFH	1K	0038H
007FFH	2K	0078H
00FFFH	4K	00F8H
01FFFH	8K	01F8H
03FFFH	16K	03F8H
07FFFH	32K	07F8H
0FFFFH	64K	0FF8H
1FFFFH	128K	1FF8H
3FFFFH	256K	3FF8H

The upper limit of this memory block is defined in the LMCS register (see Figure 12). This register is at offset A2H in the internal control block. The legal values for bits 6–15 and the resulting upper address and memory block sizes are given in Table 8. Any combination of bits 6–15 not shown in Table 8 will result in undefined operation. After reset, the LMCS register value is undefined. However, the $\overline{LCS}$ chip-select line will not become active until the LMCS register is accessed.

Any internally generated 20-bit address whose upper 16 bits are less than or equal to LMCS (with bits 0–5 "1") will cause $\overline{LCS}$ to be active. LMCS register bits R2–R0 are used to specify the READY mode for the area of memory defined by this chip-select register.

Mid-Range Memory $\overline{CS}$

The iAPX 186 provides four $\overline{MCS}$ lines which are active within a user-locatable memory block. This block can be located anywhere within the iAPX 186 1M byte memory address space exclusive of the areas defined by $\overline{UCS}$ and $\overline{LCS}$. Both the base address and size of this memory block are programmable.

The size of the memory block defined by the mid-range select lines, as shown in Table 9, is determined by bits 8–14 of the MPCS register (see Figure 13). This register is at location A8H in the internal control block. One and only one of bits 8–14 must be set at a time. Unpredictable operation of the $\overline{MCS}$ lines will otherwise occur. Each of the four chip-select lines is active for one of the four equal contiguous divisions of the mid-range block. Thus, if the total block size is 32K, each chip select is active for 8K of memory with $\overline{MCS0}$ being active for the first range and $\overline{MCS3}$ being active for the last range.

The EX and MS in MPCS relate to peripheral functionality as described in a later section

Table 9. MMCS Programming Values

Total Block Size	Individual Select Size	MMCS Bits 14–8
8K	2K	0000001B
16K	4K	0000010B
32K	8K	0000100B
64K	16K	0001000B
128K	32K	0010000B
256K	64K	0100000B
512K	128K	1000000B

The base address of the mid-range memory block is defined by bits 15–9 of the MMCS register (see Figure 14). This register is at offset A6H in the internal control block. These bits correspond to bits A19–A13 of the 20-bit memory address. Bits A12–A0 of the base address are always 0. The base address may be set at any integer multiple of the size of the total memory block selected. For example, if the mid-range block size is 32K (or the size of the block for which each $\overline{MCS}$ line is active is 8K), the block could be located at 10000H or 18000H, but not at 14000H, since the first few integer multiples of a 32K memory block are 0H, 8000H, 10000H, 18000H, etc. After reset, the contents of both of these registers is undefined. However, none of the $\overline{MCS}$ lines will be active until both the MMCS and MPCS registers are accessed.

Figure 11. UMCS Register

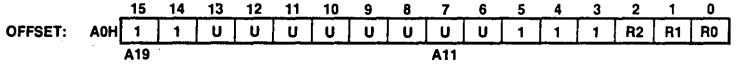


Figure 12. LMCS Register

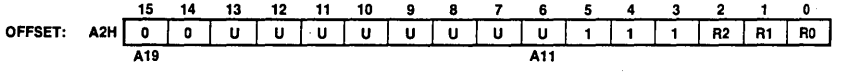


Figure 13. MPCS Register

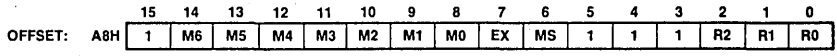
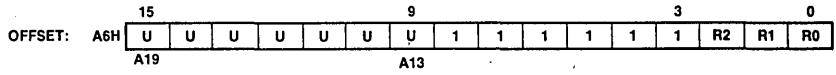


Figure 14. MMCS Register



MMCS bits R2–R0 specify READY mode of operation for all mid-range chip selects. All devices in mid-range memory must use the same number of WAIT states.

The 512K block size for the mid-range memory chip selects is a special case. When using 512K, the base address would have to be at either locations 00000H or 80000H. If it were to be programmed at 00000H when the $\overline{LCS}$ line was programmed, there would be an internal conflict between the $\overline{LCS}$ ready generation logic and the $\overline{MCS}$ ready generation logic. Likewise, if the base address were programmed at 80000H, there would be a conflict with the $\overline{UCS}$ ready generation logic. Since the $\overline{LCS}$ chip-select line does not become active until programmed, while the $\overline{UCS}$ line is active at reset, the memory base can be set only at 00000H. If this base address is selected, however, the $\overline{LCS}$ range must not be programmed.

Peripheral Chip Selects

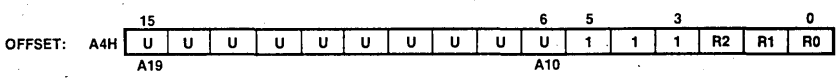
The iAPX 186 can generate chip selects for up to seven peripheral devices. These chip selects are active for seven contiguous blocks of 128 bytes above a programmable base address. This base address may be located in either memory or I/O space.

Seven $\overline{CS}$ lines called $\overline{PCS0}$–6 are generated by the iAPX 186. The base address is user-programmable; however it can only be a multiple of 1K bytes, i.e., the least significant 10 bits of the starting address are always 0.

$\overline{PCS5}$ and $\overline{PCS6}$ can also be programmed to provide latched address bits A1, A2. If so programmed, they cannot be used as peripheral selects. These outputs can be connected directly to the A0, A1 pins used for selecting internal registers of 8-bit peripheral chips. This scheme simplifies the hardware interface because the 8-bit registers of peripherals are simply treated as 16-bit registers located on even boundaries in I/O space or memory space where only the lower 8-bits of the register are significant: the upper 8-bits are "don't cares."

The starting address of the peripheral chip-select block is defined by the PACS register (see Figure 15). This register is located at offset A4H in the internal control block. Bits 15–6 of this register correspond to bits 19–10 of the 20-bit Programmable Base Address (PBA) of the peripheral chip-select block. Bits 9–0 of the PBA of the peripheral chip-select block are all zeros. If the chip-select block is located in I/O space, bits 12–15 must be programmed zero, since the I/O address is only 16 bits wide. Table 10 shows the address range of each peripheral chip select with respect to the PBA contained in PACS register.

Figure 15. PACS Register



The user should program bits 15–6 to correspond to the desired peripheral base location. PACS bits 0–2 are used to specify READY mode for $\overline{PCS0}$–$\overline{PCS3}$.

Table 10. PCS Address Ranges

PCS Line	Active between Locations
PCS0	PBA — PBA+127
PCS1	PBA+128 — PBA+255
PCS2	PBA+256 — PBA+383
PCS3	PBA+384 — PBA+511
PCS4	PBA+512 — PBA+639
PCS5	PBA+640 — PBA+767
PCS6	PBA+768 — PBA+895

The mode of operation of the peripheral chip selects is defined by the MPCS register (which is also used to set the size of the mid-range memory chip-select block, see Figure 16). This register is located at offset A8H in the internal control block. Bit 7 is used to select the function of $\overline{PCS5}$ and $\overline{PCS6}$, while bit 6 is used to select whether the peripheral chip selects are mapped into memory or I/O space. Table 11 describes the programming of these bits. After reset, the contents of both the MPCS and the PACS registers are undefined, however none of the PCS lines will be active until both of the MPCS and PACS registers are accessed.

Table 11. MS, EX Programming Values

Bit	Description
MS	1 = Peripherals mapped into memory space. 0 = Peripherals mapped into I/O space.
EX	0 = 5 $\overline{PCS}$ lines. A1, A2 provided. 1 = 7 $\overline{PCS}$ lines. A1, A2 are not provided.

MPCS bits 0–2 are used to specify READY mode for $\overline{PCS4}$–$\overline{PCS6}$ as outlined below.

READY Generation Logic

The iAPX 186 can generate a "READY" signal internally for each of the memory or peripheral $\overline{CS}$ lines. The number of WAIT states to be inserted for each peripheral or memory is programmable to provide 0–3 wait states for all accesses to the area for which the chip select is active. In addition, the iAPX 186 may be programmed to either ignore external READY for each chip-select range individually or to factor external READY with the integrated ready generator.

READY control consists of 3 bits for each $\overline{CS}$ line or group of lines generated by the iAPX 186. The interpretation of the ready bits is shown in Table 12.

Table 12. READY Bits Programming

R2	R1	R0	Number of WAIT States Generated
0	0	0	0 wait states, external RDY also used.
0	0	1	1 wait state inserted, external RDY also used.
0	1	0	2 wait states inserted, external RDY also used.
0	1	1	3 wait states inserted, external RDY also used.
1	0	0	0 wait states, external RDY ignored.
1	0	1	1 wait state inserted, external RDY ignored.
1	1	0	2 wait states inserted, external RDY ignored.
1	1	1	3 wait states inserted, external RDY ignored.

The internal ready generator operates in parallel with external READY, not in series if the external READY is used (R2 = 0). This means, for example, if the internal generator is set to insert two wait states, but activity on the external READY lines will insert four wait states, the processor will only insert four wait states, not six. This is because the two wait states generated by the internal generator overlapped the first two wait states generated by the external ready signal. Note that the external ARDY and SRDY lines are always ignored during cycles accessing internal peripherals.

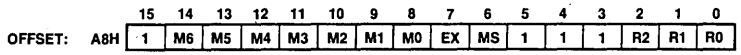
R2–R0 of each control word specifies the READY mode for the corresponding block, with the exception of the peripheral chip selects: R2–R0 of PACS set the $\overline{PCS0}$–3 READY mode, R2–R0 of MPCS set the $\overline{PCS4}$–6 READY mode.

Chip Select/Ready Logic and Reset

Upon reset, the Chip-Select/Ready Logic will perform the following actions:

- All chip-select outputs will be driven HIGH.
- Upon leaving RESET, the $\overline{UCS}$ line will be programmed to provide chip selects to a 1K block with the accompanying READY control bits set at 011 to

Figure 16. MPCS Register



allow the maximum number of internal wait states in conjunction with external Ready consideration (i.e., UMCS resets to FFFBH).

- No other chip select or READY control registers have any predefined values after RESET. They will not become active until the CPU accesses their control registers. Both the PACS and MPCS registers must be accessed before the $\overline{PCS}$ lines will become active.

DMA CHANNELS

The 80186 DMA controller provides two independent high-speed DMA channels. Data transfers can occur between memory and I/O spaces (e.g., Memory to I/O) or within the same space (e.g., Memory to Memory or I/O to I/O). Data can be transferred either in bytes (8 bits) or in words (16 bits) to or from even or odd addresses. Each DMA channel maintains both a 20-bit source and destination pointer which can be optionally incremented or decremented after each data transfer (by one or two depending on byte or word transfers). Each data transfer consumes 2 bus cycles (a minimum of 8 clocks), one cycle to fetch data and the other to store data. This provides a maximum data transfer rate of one Mword/sec or 2 MBytes/sec.

DMA Operation

Each channel has six registers in the control block which define each channel's specific operation. The control registers consist of a 20-bit Source pointer (2 words), a 20-bit Destination pointer (2 words), a 16-bit Transfer Counter, and a 16-bit Control Word. The format of the DMA Control Blocks is shown in Table 13. The Transfer Count Register (TC) specifies the number of DMA transfers to be performed. Up to 64K byte or word transfers can be performed with automatic termination. The Control Word defines the channel's operation (see Figure 18). All registers may be modified or altered during any DMA activity. Any changes made to these registers will be reflected immediately in DMA operation.

Table 13. DMA Control Block Format

Register Name	Register Address	
	Ch. 0	Ch. 1
Control Word	CAH	DAH
Transfer Count	C8H	D8H
Destination Pointer (upper 4 bits)	C6H	D6H
Destination Pointer	C4H	D4H
Source Pointer (upper 4 bits)	C2H	D2H
Source Pointer	C0H	D0H

Figure 17. DMA Unit Block Diagram

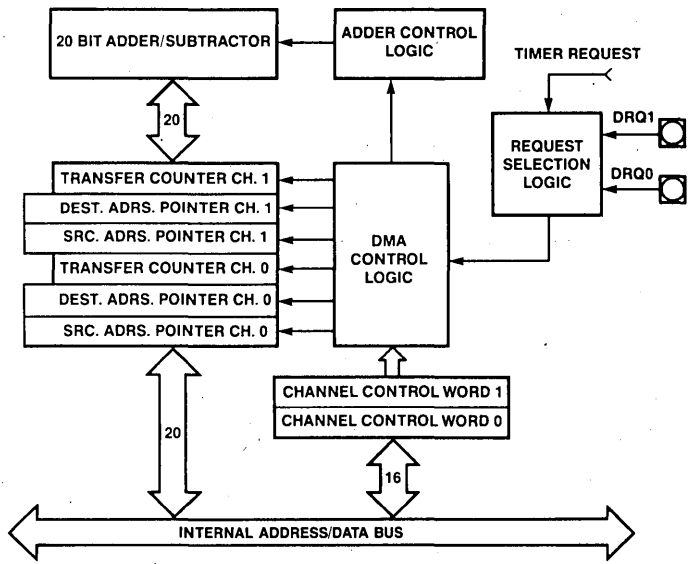
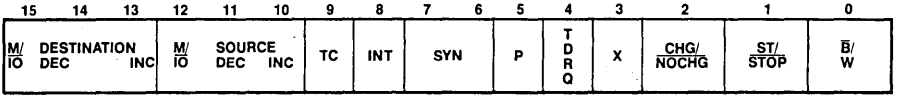


Figure 18. DMA Control Register



X = DON'T CARE.

DMA Channel Control Word Register

Each DMA Channel Control Word determines the mode of operation for the particular 80186 DMA channel. This register specifies:

- the mode of synchronization;
- whether bytes or words will be transferred;
- whether interrupts will be generated after the last transfer;
- whether DMA activity will cease after a programmed number of DMA cycles;
- the relative priority of the DMA channel with respect to the other DMA channel;
- whether the source pointer will be incremented, decremented, or maintained constant after each transfer;
- whether the source pointer addresses memory or I/O space;
- whether the destination pointer will be incremented, decremented, or maintained constant after each transfer; and
- whether the destination pointer will address memory or I/O space.

The DMA channel control registers may be changed while the channel is operating. However, any changes made during operation will affect the current DMA transfer.

DMA Control Word Bit Descriptions

$\overline{B}$/W: Byte/Word (0/1) Transfers.

ST/$\overline{STOP}$: Start/stop (1/0) Channel.

CHG/$\overline{NOCHG}$: Change/Do not change (1/0) ST/$\overline{STOP}$ bit. If this bit is set when writing to the control word, the ST/$\overline{STOP}$ bit will be programmed by the write to the control word. If this bit is cleared when writing the control word, the ST/$\overline{STOP}$ bit will not be altered. This bit is not stored; it will always be a 0 on read.

INT: Enable Interrupts to CPU on byte count termination.

TC: If set, DMA will terminate when the contents of the Transfer Count register reach zero. The ST/$\overline{STOP}$ bit will also be reset at this point if TC is set. If this bit is cleared, the DMA unit will decrement the transfer count register for each DMA cycle, but the DMA transfer will not stop when the contents of the TC register reach zero.

SYN: (2 bits)
00 No synchronization.
NOTE: The ST bit will be cleared automatically when the contents of the TC register reach zero regardless of the state of the TC bit.
01 Source synchronization.
10 Destination synchronization.
11 Unused.

SOURCE:INC Increment source pointer by 1 or 2 (depends on $\overline{B}$/W) after each transfer.

M/$\overline{IO}$ Source pointer is in M/IO space (1/0).

DEC Decrement source pointer by 1 or 2 (depends on $\overline{B}$/W) after each transfer.

DEST: INC Increment destination pointer by 1 or 2 ($\overline{B}$/W) after each transfer.

M/$\overline{IO}$ Destination pointer is in M/IO space (1/0).

DEC Decrement destination pointer by 1 or 2 (depending on $\overline{B}$/W) after each transfer.

P Channel priority—relative to other channel.
0 low priority.
1 high priority.
Channels will alternate cycles if both set at same priority level.

TDRQ	0: Disable DMA requests from timer 2. 1: Enable DMA requests from timer 2.
Bit 3	Bit 3 is not used.

If both INC and DEC are specified for the same pointer, the pointer will remain constant after each cycle.

DMA Destination and Source Pointer Registers

Each DMA channel maintains a 20-bit source and a 20-bit destination pointer. Each of these pointers takes up two full 16-bit registers in the peripheral control block. The lower four bits of the upper register contain the upper four bits of the 20-bit physical address (see Figure 18a). These pointers may be individually incremented or decremented after each transfer. If word transfers are performed the pointer is incremented or decremented by two. Each pointer may point into either memory or I/O space. Since the DMA channels can perform transfers to or from odd addresses, there is no restriction on values for the pointer registers. Higher transfer rates can be obtained if all word transfers are performed to even addresses, since this will allow data to be accessed in a single memory access.

DMA Transfer Count Register

Each DMA channel maintains a 16-bit transfer count register (TC). This register is decremented after every DMA cycle, regardless of the state of the TC bit in the DMA Control Register. If the TC bit in the DMA control word is set, however, DMA activity will terminate when the transfer count register reaches zero.

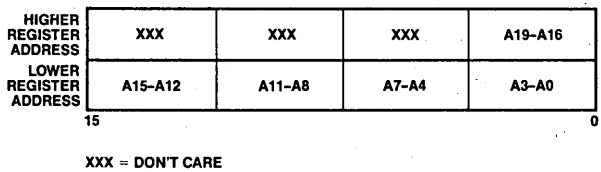
DMA Requests

Data transfers may be either source or destination synchronized, that is either the source of the data or the destination of the data may request the data transfer. In addition, DMA transfers may be unsynchronized; that is, the transfer will take place continually until the correct number of transfers has occurred. When source or unsynchronized transfers are performed, the DMA channel may begin another transfer immediately after the end of a previous DMA transfer. This allows a complete transfer to take place every 2 bus cycles or eight clock cycles (assuming no wait states). No prefetching occurs when destination synchronization is performed, however. Data will not be fetched from the source address until the destination device signals that it is ready to receive it. When destination synchronized transfers are requested, the DMA controller will relinquish control of the bus after every transfer. If no other bus activity is initiated, another DMA cycle will begin after two processor clocks. This is done to allow the destination device time to remove its request if another transfer is not desired. Since the DMA controller will relinquish the bus, the CPU can initiate a bus cycle. As a result, a complete bus cycle will often be inserted between destination synchronized transfers. These lead to the maximum DMA transfer rates shown in Table 14.

Table 14. Maximum DMA Transfer Rates

Type of Synchronization Selected	CPU Running	CPU Halted
Unsynchronized	2MBytes/sec	2MBytes/sec
Source Synch	2MBytes/sec	2MBytes/sec
Destination Synch	1.3MBytes/sec	1.5MBytes/sec

Figure 18a. DMA Memory Pointer Register Format



XXX = DON'T CARE

DMA Acknowledge

No explicit DMA acknowledge pulse is provided. Since both source and destination pointers are maintained, a read from a requesting source, or a write to a requesting destination, should be used as the DMA acknowledge signal. Since the chip-select lines can be programmed to be active for a given block of memory or I/O space, and the DMA pointers can be programmed to point to the same given block, a chip-select line could be used to indicate a DMA acknowledge.

DMA Priority

The DMA channels may be programmed such that one channel is always given priority over the other, or they may be programmed such as to alternate cycles when both have DMA requests pending. DMA cycles always have priority over internal CPU cycles except between locked memory accesses or word accesses the odd memory locations; however, an external bus hold takes priority over an internal DMA cycle. Because an interrupt request cannot suspend a DMA operation and the CPU cannot access memory during a DMA cycle, interrupt latency time will suffer during sequences of continuous DMA cycles. An NMI request, however, will cause all internal DMA activity to halt. This allows the CPU to quickly respond to the NMI request.

DMA Programming

DMA cycles will occur whenever the ST/STOP bit of the Control Register is set. If synchronized transfers are programmed, a DRQ must also have been generated. Therefore, the source and destination transfer pointers, and the transfer count register (if used) must be programmed before this bit is set.

Each DMA register may be modified while the channel is operating. If the CHG/NOCHG bit is cleared when the control register is written, the ST/STOP bit of the control register will not be modified by the write. If multiple channel registers are modified, it is recommended that a LOCKED string transfer be used to prevent a DMA transfer from occurring between updates to the channel registers.

DMA Channels and Reset

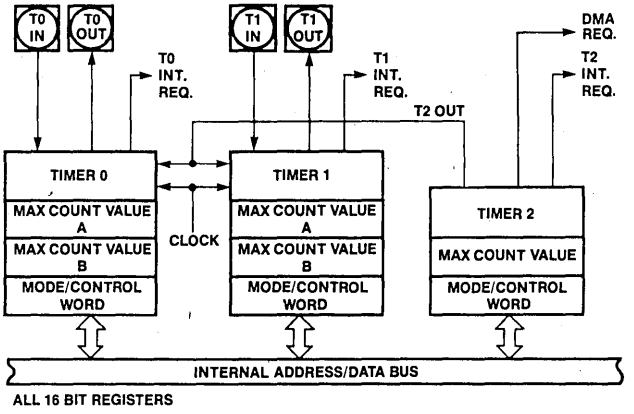
Upon RESET, the DMA channels will perform the following actions:

- The Start/Stop bit for each channel will be reset to STOP.
- Any transfer in progress is aborted.

TIMERS

The 80186 provides three internal 16-bit programmable timers (see Figure 19). Two of these are highly flexible and are connected to four external pins (2 per timer). They can be used to count external events, time external events, generate nonrepetitive waveforms, etc. The third timer is not connected to any external pins, and is useful for real-time coding and time delay applications. In addition, this third timer can be used as a prescaler to the other two, or as a DMA request source.

Figure 19. Timer Block Diagram



Timer Operation

The timers are controlled by 11 16-bit registers in the internal peripheral control block. The configuration of these registers is shown in Table 15. The count register contains the current value of the timer. It can be read or written at any time independent of whether the timer is running or not. The value of this register will be incremented for each timer event. Each of the timers is equipped with a MAX COUNT register, which defines the maximum count the timer will reach. After reaching the MAX COUNT register value, the timer count value will reset to zero during that same clock, i.e., the maximum count value is never stored in the count register itself. Timers 0 and 1 are, in addition, equipped with a second MAX COUNT register, which enables the timers to alternate their count between two different MAX COUNT values programmed by the user. If a single MAX COUNT register is used, the timer output pin will switch LOW for a single clock, 2 clocks after the maximum count value has been reached. In the dual MAX COUNT register mode, the output pin will indicate which MAX COUNT register is currently in use, thus allowing nearly complete freedom in selecting waveform duty cycles. For the timers with two MAX COUNT registers, the RIU bit in the control register determines which is used for the comparison.

Each timer gets serviced every fourth CPU-clock cycle, and thus can operate at speeds up to one-quarter the internal clock frequency (one-eighth the crystal rate). External clocking of the timers may be done at up to a rate of one-quarter of the internal CPU-clock rate (2 MHz for an 8 MHz CPU clock). Due to internal synchronization and pipelining of the timer circuitry, a timer output may take up to 6 clocks to respond to any individual clock or gate input. Since the count registers and the maximum count registers are all 16 bits wide, 16 bits of resolution are provided. Any Read or Write access to the timers will add one wait state to the minimum four-clock bus cycle. However, this is needed to synchronize and coordinate the internal data flows between the internal timers and the internal bus.

The timers have several programmable options.

- All three timers can be set to halt or continue on a terminal count.
- Timers 0 and 1 can select between internal and external clocks, alternate between MAX COUNT registers and be set to retrigger on external events.
- The timers may be programmed to cause an interrupt on terminal count.

These options are selectable via the timer mode/control word.

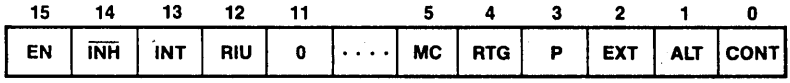
Timer Mode/Control Register

The mode/control register (see Figure 20) allows the user to program the specific mode of operation or check the current programmed status for any of the three integrated timers.

Table 15. Timer Control Block Format

Register Name	Register Offset		
	Tmr. 0	Tmr. 1	Tmr. 2
Mode/Control Word	56H	5EH	66H
Max Count B	54H	5CH	not present
Max Count A	52H	5AH	62H
Count Register	50H	58H	60H

Figure 20. Timer Mode/Control Register



ALT:
The ALT bit determines which of two MAX COUNT registers is used for count comparison. If ALT = 0, register A for that timer is always used, while if ALT = 1, the comparison will alternate between register A and register B when each maximum count is reached. This alternation allows the user to change one MAX COUNT register while the other is being used, and thus provides a method of generating non-repetitive waveforms. Square waves and pulse outputs of any duty cycle are a subset of available signals obtained by not changing the final count registers. The ALT bit also determines the function of the timer output pin. If ALT is zero, the output pin will go LOW for one clock, the clock after the maximum count is reached. If ALT is one, the output pin will reflect the current MAX COUNT register being used (0/1 for B/A).

CONT:
Setting the CONT bit causes the associated timer to run continuously, while resetting it causes the timer to halt upon maximum count. If CONT = 0 and ALT =1, the timer will count to the MAX COUNT register A value, reset, count to the register B value, reset, and halt.

EXT:
The external bit selects between internal and external clocking for the timer. The external signal may be asynchronous with respect to the 80186 clock. If this bit is set, the timer will count LOW-to-HIGH transitions on the input pin. If cleared, it will count an internal clock while using the input pin for control. In this mode, the function of the external pin is defined by the RTG bit. The maximum input to output transition latency time may be as much as 6 clocks. However, clock inputs may be pipelined as closely together as every 4 clocks without losing clock pulses.

P:
The prescaler bit is ignored unless internal clocking has been selected (EXT = 0). If the P bit is a zero, the timer will count at one-fourth the internal CPU clock rate. If the P bit is a one, the output of timer 2 will be used as a clock for the timer. Note that the user must initialize and start timer 2 to obtain the prescaled clock.

RTG:
Retrigger bit is only active for internal clocking (EXT = 0). In this case it determines the control function provided by the input pin.

If RTG = 0, the input level gates the internal clock on and off. If the input pin is HIGH, the timer will count; if the input pin is LOW, the timer will hold its value. As indicated previously, the input signal may be asynchronous with respect to the 80186 clock.

When RTG = 1, the input pin detects LOW-to-HIGH transitions. The first such transition starts the timer running, clearing the timer value to zero on the first clock, and then incrementing thereafter. Further transitions on the input pin will again reset the timer to zero, from which it will start counting up again. If CONT = 0, when the timer has reached maximum count, the EN bit will be cleared, inhibiting further timer activity.

EN:
The enable bit provides programmer control over the timer's RUN/HALT status. When set, the timer is enabled to increment subject to the input pin constraints in the internal clock mode (discussed previously). When cleared, the timer will be inhibited from counting. All input pin transitions during the time EN is zero will be ignored. If CONT is zero, the EN bit is automatically cleared upon maximum count.

INH:
The inhibit bit allows for selective updating of the enable (EN) bit. If INH is a one during the write to the mode/control word, then the state of the EN bit will be modified by the write. If INH is a zero during the write, the EN bit will be unaffected by the operation. This bit is not stored; it will always be a 0 on a read.

INT:
When set, the INT bit enables interrupts from the timer, which will be generated on every terminal count. If the timer is configured in dual MAX COUNT register mode, an interrupt will be generated each time the value in MAX COUNT register A is reached, and each time the value in MAX COUNT register B is reached. If this enable bit is cleared after the interrupt request has been generated, but before a pending interrupt is serviced, the interrupt request will still be in force. (The request is latched in the Interrupt Controller.)

MC:
The Maximum Count bit is set whenever the timer reaches its final maximum count value. If the timer is configured in dual MAX COUNT register mode, this bit will be set each time the value in MAX COUNT register A is reached, and each time the value in MAX COUNT register B is reached. This bit is set regardless of the timer's interrupt-enable bit. The MC bit gives the user the ability to monitor timer status through software instead of through interrupts.

RIU:
The Register In Use bit indicates which MAX COUNT register is currently being used for comparison to the timer count value. A zero value indicates register A. The RIU bit cannot be written, i.e., its value is not affected when the control register is written. It is always cleared when the ALT bit is zero.

Not all mode bits are provided for timer 2. Certain bits are hardwired as indicated below:

ALT = 0, EXT = 0, P = 0, RTG = 0, RIU = 0

Count Registers

Each of the three timers has a 16-bit count register. The current contents of this register may be read or written by the processor at any time. If the register is written into while the timer is counting, the new value will take effect in the current count cycle.

Max Count Registers

Timers 0 and 1 have two MAX COUNT registers, while timer 2 has a single MAX COUNT register. These contain the number of events the timer will count. In timers 0 and 1, the MAX COUNT register used can alternate between the two max count values whenever the current maximum count is reached. The condition which causes a timer to reset is equivalent between the current count value and the max count being used. This means that if the count is changed to be above the max count value, or if the max count value is changed to be below the current value, the timer will not reset to zero, but rather will count to its maximum value, "wrap around" to zero, then count until the max count is reached.

Timers and Reset

Upon RESET, the Timers will perform the following actions:

- All EN (Enable) bits are reset preventing timer counting.
- All SEL (Select) bits are reset to zero. This selects MAX COUNT register A, resulting in the Timer Out pins going HIGH upon RESET.

INTERRUPT CONTROLLER

The 80186 can receive interrupts from a number of sources, both internal and external. The internal interrupt controller serves to merge these requests on a priority basis, for individual service by the CPU. Internal interrupt sources (Timers and DMA channels) can be disabled by their own control registers or by mask bits within the interrupt controller. The 80186 interrupt controller has its own control registers that set the mode of operation for the controller.

The interrupt controller will resolve priority among requests that are pending simultaneously. Nesting is provided so interrupt service routines for lower priority interrupts may themselves be interrupted by higher priority interrupts. A block diagram of the interrupt controller is shown in Figure 21.

The interrupt controller has a special iRMX 86 compatibility mode that allows the use of the 80186 within the iRMX 86 operating system interrupt structure. The controller is set in this mode by setting bit 14 in the peripheral control block relocation register (see iRMX 86 Compatibility Mode section). In this mode, the internal 80186 interrupt controller functions as a "slave" controller to an external "master" controller. Special initialization software must be included to properly set up the 80186 interrupt controller in iRMX 86 mode.

NON-iRMX MODE OPERATION

Interrupt Controller External Interface

For external interrupt sources, five dedicated pins are provided. One of these pins is dedicated to NMI, non-maskable interrupt. This is typically used for power-fail interrupts, etc. The other four pins may function either as four interrupt input lines with internally generated interrupt vectors, as an interrupt line and an interrupt acknowledge line (called the "cascade mode") along with two other input lines with internally generated interrupt vectors, or as two interrupt input lines and two dedicated interrupt acknowledge ouput lines. When the interrupt lines are configured in cascade mode, the 80186 interrupt controller will not generate internal interrupt vectors.

External sources in the cascade mode use externally generated interrupt vectors. When an interrupt is acknowledged, two $\overline{\text{INTA}}$ cycles are initiated and the vector is read into the 80186 on the second cycle. The capability to interface to external 8259A programmable interrupt controllers is thus provided when the inputs are configured in cascade mode.

Interrupt Controller Modes of Operation

The basic modes of operation of the interrupt controller in non-iRMX mode are similar to the 8259A. The interrupt controller responds identically to internal interrupts in all three modes: the difference is only in the interpretation of function of the four external interrupt pins. The interrupt controller is set into one of these three modes by programming the correct bits in the INT0 and INT1 control registers. The modes of interrupt controller operation are as follows:

Fully Nested Mode

When in the fully nested mode four pins are used as direct interrupt requests. The vectors for these four inputs are generated internally. An in-service bit is provided for every interrupt source. If a lower-priority device requests an interrupt while the in-service bit (IS) is set, no interrupt will be generated by the interrupt controller. In addition, if another interrupt request occurs from the same interrupt source while the inservice bit is set, no interrupt will be generated by the interrupt controller. This allows interrupt service routines to operate with interrupts enabled without being themselves interrupted by lower-priority interrupts. Since interrupts are enabled, higher-priority interrupts will be serviced.

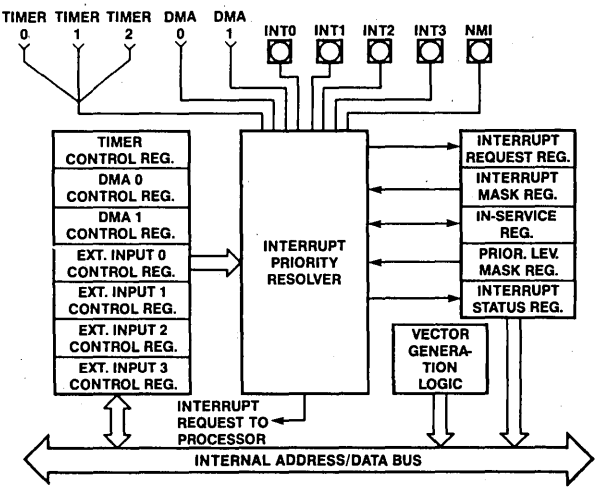
When a service routine is completed, the proper IS bit must be reset by writing the proper pattern to the EOI register. This is required to allow subsequent interrupts from this interrupt source and to allow servicing of lower-priority interrupts. An EOI command is issued at the end of the service routine just before the issuance of the return from interrupt instruction. If the fully nested structure has been upheld, the next highest-priority source with its IS bit set is then serviced.

Cascade Mode

The 80186 has four interrupt pins and two of them have dual functions. In the fully nested mode the four pins are used as direct interrupt inputs and the corresponding vectors are generated internally. In the cascade mode, the four pins are configured into interrupt input-dedicated acknowledge signal pairs. The interconnection is shown in Figure 22. INT0 is an interrupt input interfaced to an 8259A, while INT2/$\overline{\text{INTA0}}$ serves as the dedicated interrupt acknowledge signal to that peripheral. The same is true for INT1 and INT3/$\overline{\text{INTA1}}$. Each pair can selectively be placed in the cascade or non-cascade mode by programming the proper value into INT0 and INT1 control registers. The use of the dedicated acknowledge signals eliminates the need for the use of external logic to generate $\overline{\text{INTA}}$ and device select signals.

The primary cascade mode allows the capability to serve up to 128 external interrupt sources through the use of external master and slave 8259As. Three levels of priority are created, requiring priority resolution in the 80186 interrupt controller, the master 8259As, and the slave 8259As. If an external interrupt is serviced, one IS bit is set at each of these levels. When the interrupt service routine is completed, up to three end-of-interrupt commands must be issued by the programmer.

Figure 21. Interrupt Controller Block Diagram



Special Fully Nested Mode
This mode is entered by setting the SFNM bit in INT0 or INT1 control register. It enables complete nestability with external 8259A masters. Normally, an interrupt request from an interrupt source will not be recognized unless the in-service bit for that source is reset. If more than one interrupt source is connected to an external interrupt controller, all of the interrupts will be funneled through the same 80186 interrupt request pin. As a result, if the external interrupt controller receives a higher-priority interrupt, its interrupt will not be recognized by the 80186 controller until the 80186 in-service bit is reset. In special fully nested mode, the 80186 interrupt controller will allow interrupts from an external pin regardless of the state of the in-service bit for an interrupt source in order to allow multiple interrupts from a single pin. An in-service bit will continue to be set, however, to inhibit interrupts from other lower-priority 80186 interrupt sources.

Special procedures should be followed when resetting IS bits at the end of interrupt service routines. Software polling of the external master's IS register is required to determine if there is more than one bit set. If so, the IS bit in the 80186 remains active and the next interrupt service routine is entered.

Operation in a Polled Environment

The controller may be used in a polled mode if interrupts are undesirable. When polling, the processor disables interrupts and then polls the interrupt controller whenever it is convenient. Polling the interrupt controller is accomplished by reading the Poll Word (Figure 9). Bit 15 in the poll word indicates to the processor that an interrupt of high enough priority is requesting service. Bits 0–4 indicate to the processor the type vector of the highest-priority source requesting service. Reading the Poll Word causes the In-Service bit of the highest-priority source to be set.

It is desirable to be able to read the Poll Word information without guaranteeing service of any pending interrupt, i.e., not set the indicated in-service bit. The 80186 provides a Poll Status Word in addition to the conventional Poll Word to allow this to be done. Poll Word information is duplicated in the Poll Status Word, but reading the Poll Status Word does not set the associated in-service bit. These words are located in two adjacent memory locations in the register file.

Non-iRMX Mode Features

Programmable Priority
The user can program the interrupt sources into any of eight different priority levels. The programming is done by placing a 3-bit priority level (0–7) in the control register of each interrupt source. (A source with a priority level of 4 has higher priority over all priority levels from 5 to 7. Priority registers containing values lower than 4 have greater priority.) All interrupt sources have preprogrammed default priority levels (see Table 4).

If two requests with the same programmed priority level are pending at once, the priority ordering scheme shown in Table 4 is used. If the serviced interrupt routine reenables interrupts, it allows other requests to be serviced.

End-of-Interrupt Command
The end-of-interrupt (EOI) command is used by the programmer to reset the In-Service (IS) bit when an interrupt service routine is completed. The EOI command is issued by writing the proper pattern to the EOI register. There are two types of EOI commands, specific and nonspecific. The nonspecific command does not specify which IS bit is reset. When issued, the interrupt controller automatically resets the IS bit of the highest priority source with an active service routine. A specific EOI command requires that the programmer send the interrupt vector type to the interrupt controller indicating which source's IS bit is to be reset. This command is used when the fully nested structure has been disturbed or the highest priority IS bit that was set does not belong to the service routine in progress.

Trigger Mode
The four external interrupt pins can be programmed in either edge- or level-trigger mode. The control register for each external source has a level-trigger mode (LTM) bit. All interrupt inputs are active HIGH. In the edge sense mode or the level-trigger mode, the interrupt request must remain active (HIGH) until the interrupt request is acknowledged by the 80186 CPU. In the edge-sense mode, if the level remains high after the interrupt is acknowledged, the input is disabled and no further requests will be generated. The input level must go LOW for at least one clock cycle to reenable the input. In the level-trigger mode, no such provision is made: holding the interrupt input HIGH will cause continuous interrupt requests.

Interrupt Vectoring
The 80186 Interrupt Controller will generate interrupt vectors for the integrated DMA channels and the integrated Timers. In addition, the Interrupt Controller will generate interrupt vectors for the external interrupt lines if they are not configured in Cascade or Special Fully Nested Mode. The interrupt vectors generated are fixed and cannot be changed (see Table 4).

Interrupt Controller Registers

The Interrupt Controller register model is shown in Figure 23. It contains 15 registers. All registers can both be read or written unless specified otherwise.

In-Service Register
This register can be read from or written into. The format is shown in Figure 24. It contains the In-Service bit for each of the interrupt sources. The In-Service bit is set to indicate that a source's service routine is in progress. When an In-Service bit is set, the interrupt controller will not generate interrupts to the CPU when it receives interrupt requests from devices with a lower programmed priority level. The TMR bit is the In-Service bit for all three timers; the D0 and D1 bits are the In-Service bits for the two DMA channels; the I0–I3 are the In-Service bits for the external interrupt pins. The IS bit is set when the processor acknowledges an interrupt request either by an interrupt acknowledge or by reading the poll register. The IS bit is reset at the end of the interrupt service routine by an end-of-interrupt command issued by the CPU.

Interrupt Request Register
The internal interrupt sources have interrupt request bits inside the interrupt controller. The format of this register is shown in Figure 24. A read from this register yields the status of these bits. The TMR bit is the logical OR of all timer interrupt requests. D0 and D1 are the interrupt request bits for the DMA channels.

The state of the external interrupt input pins is also indicated. The state of the external interrupt pins is not a stored condition inside the interrupt controller, therefore the external interrupt bits cannot be written. The external interrupt request bits show exactly when an interrupt request is given to the interrupt controller, so if edge-triggered mode is selected, the bit in the register will be HIGH only after an inactive-to-active transition. For internal interrupt sources, the register bits are set when a request arrives and are reset when the processor acknowledges the requests.

Mask Register
This is a 16-bit register that contains a mask bit for each interrupt source. The format for this register is shown in Figure 24. A one in a bit position corresponding to a particular source serves to mask the source from generating interrupts. These mask bits are the exact same bits which are used in the individual control registers; programming a mask bit using the mask register will also change this bit in the individual control registers, and vice versa.

Figure 22. Cascade Mode Interrupt Connection

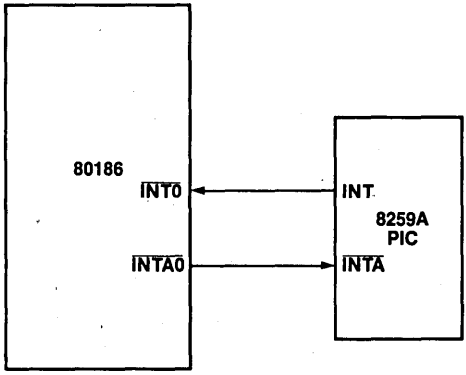


Figure 23. Interrupt Controller Registers (Non-IRMX 86 Mode)

	OFFSET
INT3 CONTROL REGISTER	3EH
INT2 CONTROL REGISTER	3CH
INT1 CONTROL REGISTER	3AH
INT0 CONTROL REGISTER	38H
DMA 1 CONTROL REGISTER	36H
DMA 0 CONTROL REGISTER	34H
TIMER CONTROL REGISTER	32H
INTERRUPT CONTROLLER STATUS REGISTER	30H
INTERRUPT REQUEST REGISTER	2EH
IN-SERVICE REGISTER	2CH
PRIORITY MASK REGISTER	2AH
MASK REGISTER	28H
POLL STATUS REGISTER	26H
POLL REGISTER	24H
EOI REGISTER	22H

Priority Mask Register

This register is used to mask all interrupts below particular interrupt priority levels. The format of this register is shown in Figure 25. The code in the lower three bits of this register inhibits interrupts of priority lower (a higher priority number) than the code specified. For example, 100 written into this register masks interrupts of level five (101), six (110), and seven (111). The register is reset to seven (111) upon RESET so all interrupts are unmasked.

Interrupt Status Register

This register contains general interrupt controller status information. The format of this register is shown in Figure 26. The bits in the status register have the following functions:

DHLT: DMA Halt Transfer; setting this bit halts all DMA transfers. It is automatically set whenever a non-maskable interrupt occurs, and it is reset when an IRET instruction is executed. The purpose of this bit is to allow prompt service of all non-maskable interrupts. This bit may also be set by the CPU.

IRTx: These three bits represent the individual timer interrupt request bits. These bits are used to differentiate the timer interrupts, since the timer IR bit in the interrupt request register is the "OR" function of all timer interrupt requests. Note that setting any one of these three bits initiates an interrupt request to the interrupt controller.

Figure 24. In-Service, Interrupt Request, and Mask Register Formats

15	14	• • •	10	9	8	7	6	5	4	3	2	1	0
0	0		0	0	0	I3	I2	I1	I0	D1	D0	0	TMR

Figure 25. Priority Mask Register Format

15	14	• • • • • • • • • •	3	2	1	0
0	0		0	PRM2	PRM1	PRM0

Figure 26. Interrupt Status Register Format

15	14	• • • • •	7	6	5	4	3	2	1	0
DHLT	0		0	0	0	0	0	IRT2	IRT1	IRT0

Timer, DMA 0, 1; Control Registers
These registers are the control words for all the internal interrupt sources. The format for these registers is shown in Figure 27. The three bit positions PR0, PR1, and PR2 represent the programmable priority level of the interrupt source. The MSK bit inhibits interrupt requests from the interrupt source. The MSK bits in the individual control registers are the exact same bits as are in the Mask Register; modifying them in the individual control registers will also modify them in the Mask Register, and vice versa.

INT0–INT3 Control Registers
These registers are the control words for the four external input pins. Figure 28 shows the format of the INT0 and INT1 Control registers; Figure 29 shows the format of the INT2 and INT3 Control registers. In cascade mode or special fully nested mode, the control words for INT2 and INT3 are not used.

The bits in the various control registers are encoded as follows:

PRO-2: Priority programming information. Highest

LTM: Level-trigger mode bit. 1 = level-triggered; 0 = edge-triggered. Interrupt Input levels are active high. In level-triggered mode, an interrupt is generated whenever the external line is high. In edge-triggered mode, an interrupt will be generated only when this level is preceded by an inactive-to-active transition on the line. In both cases, the level must remain active until the interrupt is acknowledged.

MSK: Mask bit, 1 = mask; 0 = nonmask.

C: Cascade mode bit, 1 = cascade; 0 = direct

SFNM: Special fully nested mode bit, 1 = SFNM; 0

EOI Register
The end of the interrupt register is a command register which can only be written into. The format of this register is shown in Figure 30. It initiates an EOI command when written to by the 80186 CPU.

The bits in the EOI register are encoded as follows:

S_x: Encoded information that specifies an interrupt source vector type as shown in Table 4. For example, to reset the In-Service bit for DMA channel 0, these bits should be set to 01010, since the vector type for DMA channel 0 is 10. Note that to reset the single In-Service bit for any of the three timers, the vector type for timer 0 (8) should be written in this register.

Figure 27. Timer/DMA Control Register Formats

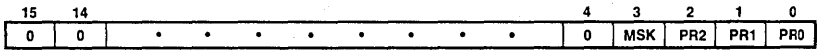


Figure 28. INT0/INT1 Control Register Formats

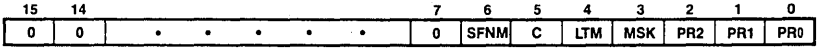


Figure 29. INT2/INT3 Control Register Formats

NSPEC/SPEC: A bit that determines the type of EOI command. Nonspecific = 1, Specific = 0.

Poll and Poll Status Registers

These registers contain polling information. The format of these registers is shown in Figure 31. They can only be read. Reading the Poll register constitutes a software poll. This will set the IS bit of the highest priority pending interrupt. Reading the poll status register will not set the IS bit of the highest priority pending interrupt; only the status of pending interrupts will be provided.

Encoding of the Poll and Poll Status register bits are as follows:

S_x: Encoded information that indicates the vector type of the highest priority interrupting source. Valid only when INTREQ = 1.

INTREQ: This bit determines if an interrupt request is present. Interrupt Request = 1; no Interrupt Request = 0.

iRMX 86 COMPATIBILITY MODE

This mode allows iRMX 86-80186 compatibility. The interrupt model of iRMX 86 requires one master and multiple slave 8259As in cascaded fashion. When iRMX mode is used, the internal 80186 interrupt controller will be used as a slave controller to an external master interrupt controller. The internal 80186 resources will be monitored through the internal interrupt controller, while the external controller functions as the system master interrupt controller.

Upon reset, the 80186 interrupt controller will be in the non-iRMX 86 mode of operation. To set the controller in the iRMX 86 mode, bit 14 of the Relocation Register should be set.

Because of pin limitations caused by the need to interface to an external 8259A master, the internal interrupt controller will no longer accept external inputs. There are however, enough 80186 interrupt controller inputs (internally) to dedicate one to each timer. In this mode, each timer interrupt source has its own mask bit, IS bit, and control word.

The iRMX 86 operating system requires peripherals to be assigned fixed priority levels. This is incompatible with the normal operation of the 80186 interrupt controller. Therefore, the initialization software must program the proper priority levels for each source. The required priority levels for the internal interrupt sources in iRMX mode are shown in Table 16.

Table 16. Internal Source Priority Level

Priority Level	Interrupt Source
0	Timer 0
1	(reserved)
2	DMA 0
3	DMA 1
4	Timer 1
5	Timer 2

These level assignments must remain fixed in the iRMX 86 mode of operation.

iRMX 86 Mode External Interface

The configuration of the 80186 with respect to an external 8259A master is shown in Figure 32. The INT0 input is used as the 80186 CPU interrupt input. INT3 functions as an output to send the 80186 slave-interrupt-request to one of the 8 master-PIC-inputs.

Figure 30. EOI Register Format

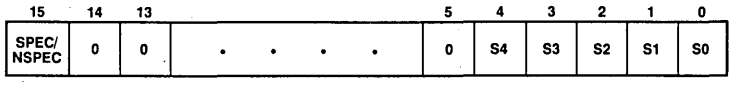


Figure 31. Poll Register Format

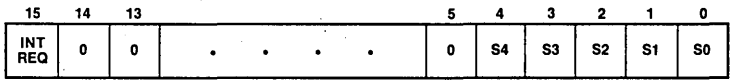
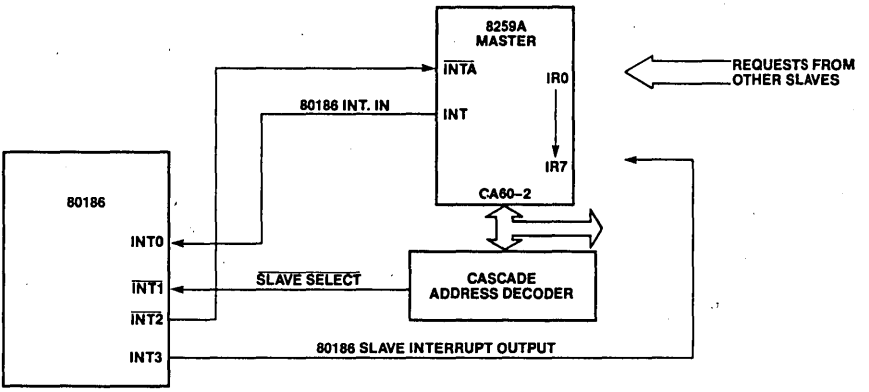


Figure 32. iRMX 86 Interrupt Controller Interconnection



Correct master-slave interface requires decoding of the slave addresses (CAS0–2). Slave 8259As do this internally. Because of pin limitations, the 80186 slave address will have to be decoded externally. $\overline{INT1}$ is used as a slave-select input. Note that the slave vector address is transferred internally, but the READY input must be supplied externally.

$\overline{INT2}$ is used as an acknowledge output, suitable to drive the $\overline{INTA}$ input of an 8259A.

Interrupt Nesting

iRMX 86 mode operation allows nesting of interrupt requests. When an interrupt is acknowledged, the priority logic masks off all priority levels except those with equal or higher priority.

Vector Generation in the iRMX 86 Mode

Vector generation in iRMX mode is exactly like that of an 8259A slave. The interrupt controller generates an 8-bit vector which the CPU multiplies by four and uses as an address into a vector table. The significant five bits of the vector are user-programmable while the lower three bits are generated by the priority logic. These bits represent the encoding of the priority level requesting service. The significant five bits of the vector are programmed by writing to the Interrupt Vector register at offset 20H.

Specific End-of-Interrupt

In iRMX mode the specific EOI command operates to reset an in-service bit of a specific priority. The user supplies a 3-bit priority-level value that points to an in-service bit to be reset. The command is executed by writing the correct value in the Specific EOI register at offset 22H.

Interrupt Controller Registers in the iRMX 86 Mode

All control and command registers are located inside the internal peripheral control block. Figure 33 shows the offsets of these registers.

End-of-Interrupt Register
The end-of-interrupt register is a command register which can only be written. The format of this register is shown in Figure 34. It initiates an EOI command when written by the 80186 CPU.

The bits in the EOI register are encoded as follows:

L_X: Encoded value indicating the priority of the IS bit to be reset.

In-Service Register
This register can be read from or written into. It contains the in-service bit for each of the internal

interrupt sources. The format for this register is shown in Figure 35. Bit positions 2 and 3 correspond to the DMA channels; positions 0, 4, and 5 correspond to the integral timers. The source's IS bit is set when the processor acknowledges its interrupt request.

Interrupt Request Register
This register indicates which internal peripherals have interrupt requests pending. The format of this register is shown in Figure 35. The interrupt request bits are set when a request arrives from an internal source, and are reset when the processor acknowledges the request.

Mask Register
This register contains a mask bit for each interrupt source. The format for this register is shown in Figure 35. If the bit in this register corresponding to a particular interrupt source is set, any interrupts from that source will be masked. These mask bits are exactly the same bits which are used in the individual control registers, i.e., changing the state of a mask bit in this register will also change the state of the mask bit in the individual interrupt control register corresponding to the bit.

Control Registers
These registers are the control words for all the internal interrupt sources. The format of these registers is shown in Figure 36. Each of the timers and both of the DMA channels have their own Control Register.

The bits of the Control Registers are encoded as follows:

pr_x: 3-bit encoded field indicating a priority level for the source; note that each source must be programmed at specified levels.

msk: mask bit for the priority level indicated by pr_x bits.

Figure 33. Interrupt Controller Registers (iRMX 86 Mode)

	OFFSET
LEVEL 5 CONTROL REGISTER (TIMER 2)	3AH
LEVEL 4 CONTROL REGISTER (TIMER 1)	38H
LEVEL 3 CONTROL REGISTER (DMA 1)	36H
LEVEL 2 CONTROL REGISTER (DMA 0)	34H
LEVEL 0 CONTROL REGISTER (TIMER 0)	32H
INTERRUPT-REQUEST REGISTER	2EH
IN-SERVICE REGISTER	2CH
PRIORITY-LEVEL MASK REGISTER	2AH
MASK REGISTER	28H
SPECIFIC EOI REGISTER	22H
INTERRUPT VECTOR REGISTER	20H

Figure 34. Specific EOI Register Format

15	14	13	. . . . .	8	7	6	5	4	3	2	1	0
0	0	0	. . . . .	0	0	0	0	0	0	L2	L1	L0

Figure 35. In-Service, Interrupt Request, and Mask Register Format

15	14	13	. . . . .	8	7	6	5	4	3	2	1	0
0	0	0	. . . . .	0	0	0	TMR2	TMR1	D1	D0	0	TMR0

Interrupt Vector Register
This register provides the upper five bits of tne interrupt vector address. The format of this register is shown in Figure 37. The interrupt controller itself provides the lower three bits of the interrupt vector as determined by the priority level of the interrupt request.

The format of the bits in this register is:

t_x: 5-bit field indicating the upper five bits of the vector address.

Priority-Level Mask Register
This register indicates the lowest priority-level interrupt which will be serviced.

The encoding of the bits in this register is:

m_x: 3-bit encoded field indication priority-level value. All levels of lower priority will be masked.

Interrupt Controller and Reset

Upon RESET, the interrupt controller will perform the following actions:

- All SFNM bits reset to 0, implying Fully Nested Mode.
- All PR bits in the various control registers set to 1. This places all sources at lowest priority (level 111).
- All LTM bits reset to 0, resulting in edge-sense mode.
- All Interrupt Service bits reset to 0.
- All Interrupt Request bits reset to 0.
- All MSK (Interrupt Mask) bits set to 1 (mask).
- All C (Cascade) bits reset to 0 (non-cascade).
- All PRM (Priority Mask) bits set to 1, implying no levels masked.
- Initialized to non-iRMX 86 mode.

Figure 36. Control Word Format

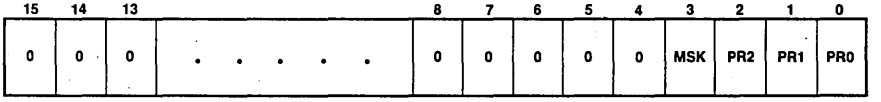


Figure 37. Interrupt Vector Register Format

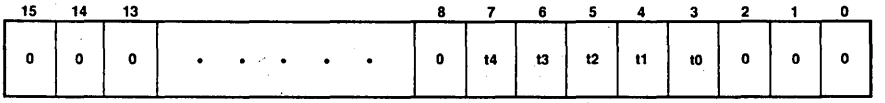


Figure 38. Priority Level Mask Register

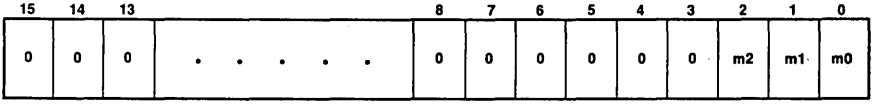


Figure 39. Typical iAPX 186 Computer

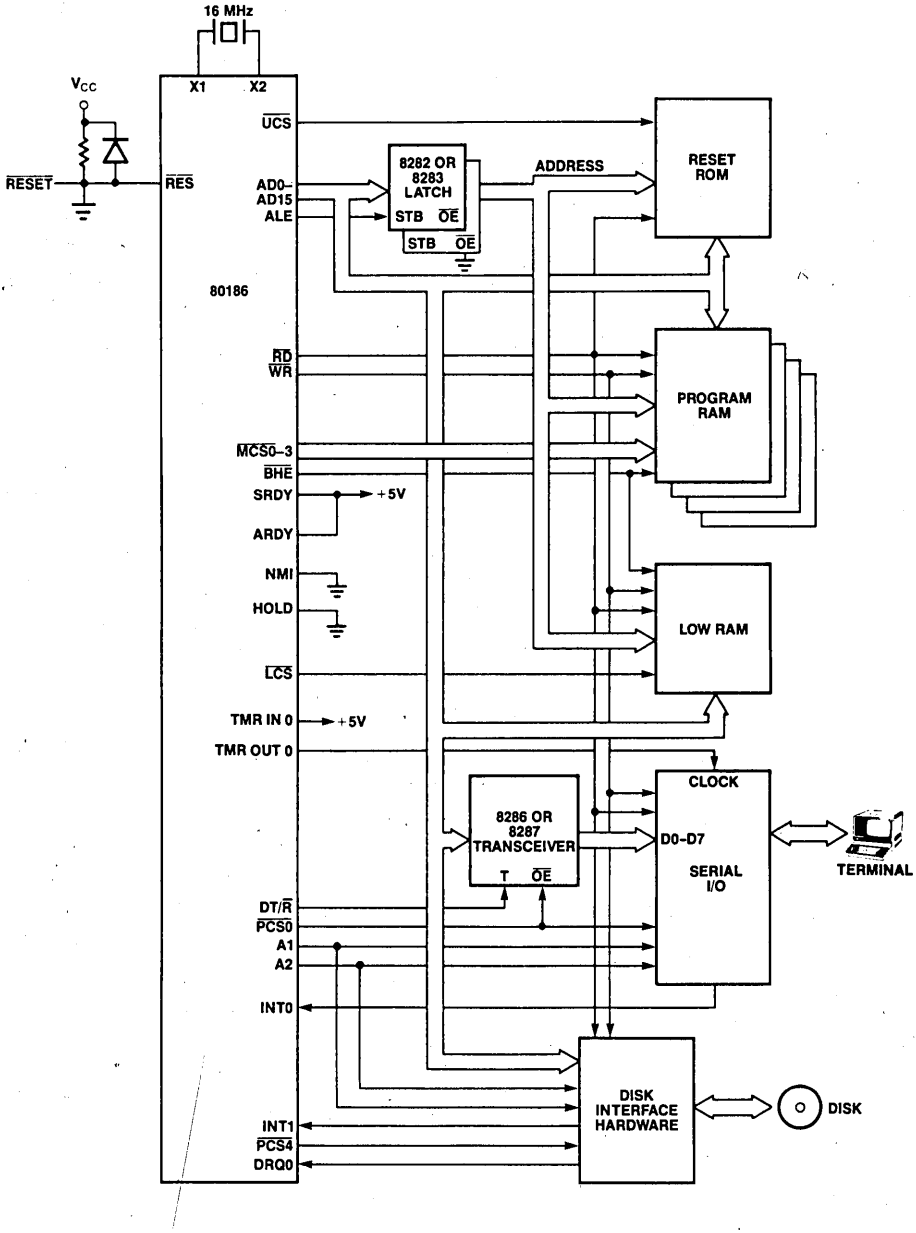
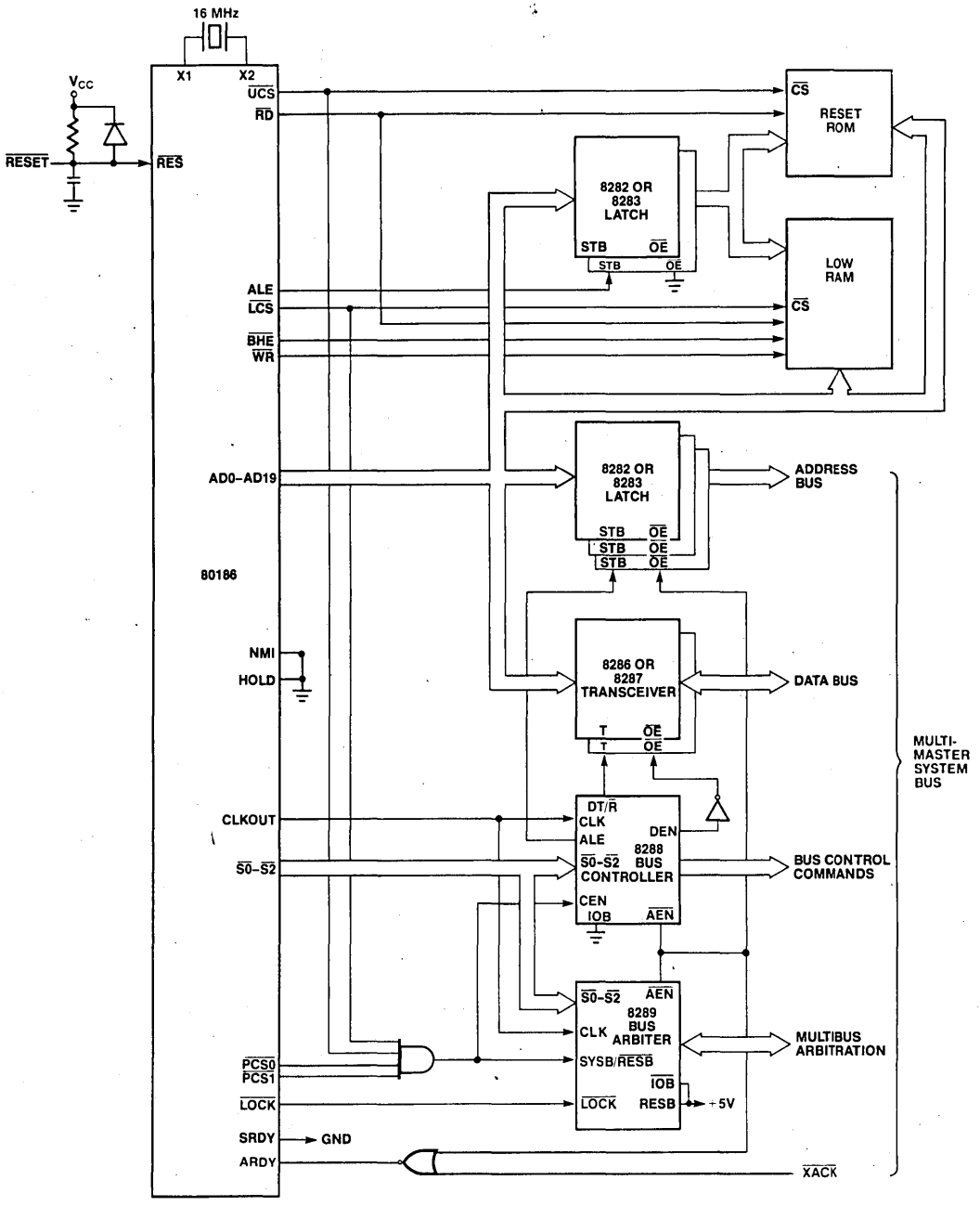


Figure 40. Typical iAPX 186 Multi-Master Bus Interface



PACKAGE

The 80186 is housed in a 68-pin, leadless JEDEC type A hermetic chip carrier. Figure 41 illustrates the package dimensions.

NOTE: The IDT 3M Textool 68-pin JEDEC Socket is required for ICE™ operation. See Figure 40 for details.

Figure 41. 80186 JEDEC Type A Package

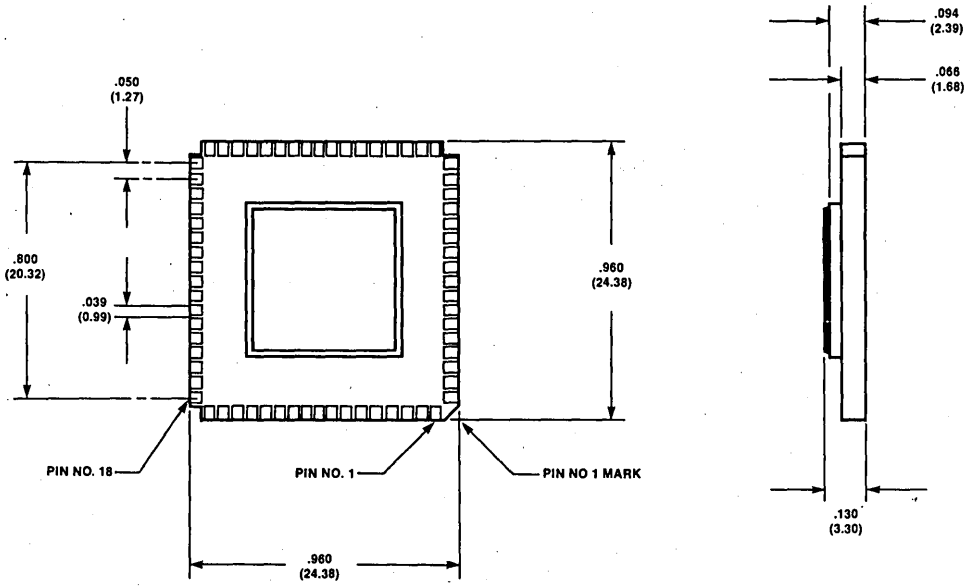
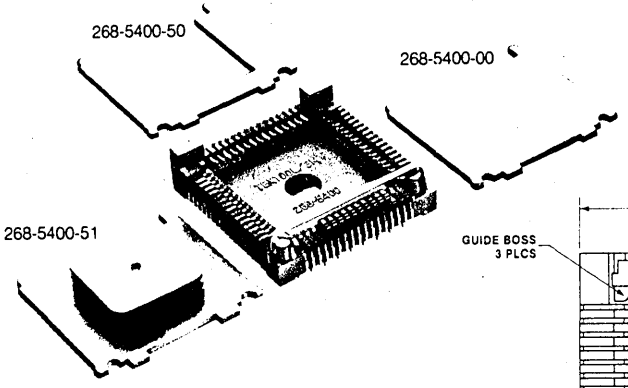
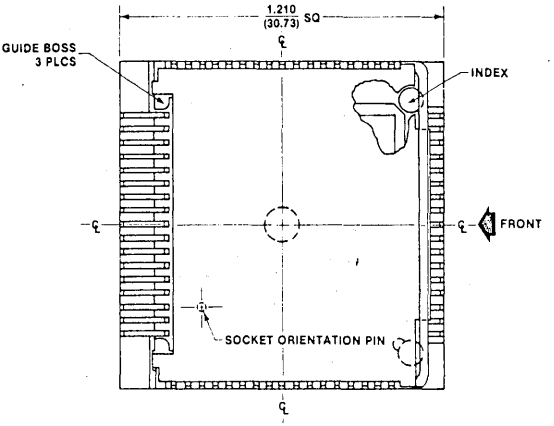
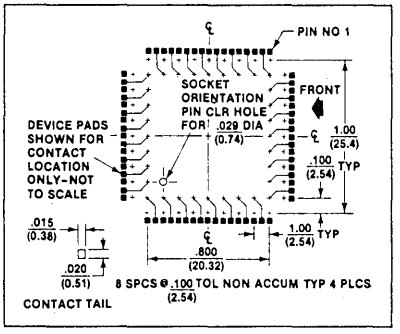


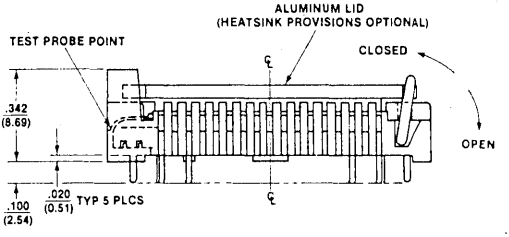
Figure 42. Textool 68 Lead Chip Carrier Socket





PC BOARD PATTERN





ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature under Bias ..... 0°C to 70°C
Storage Temperature ..... −65°C to +150°C
Voltage on Any Pin with
Respect to Ground ..... −1.0V to +7V
Power Dissipation ..... 3 Watt

NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

D.C. CHARACTERISTICS (T_A = 0°–70°C, V_{CC} = 5V ±10%)

Symbol	Parameter	Min.	Max.	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	+0.8	Volts	
V_{IH}	Input High Voltage (All except X1 and $\overline{RES}$)	2.0	V_{CC} + 0.5	Volts	
V_{IH1}	Input High Voltage ($\overline{RES}$)	TBD	V_{CC} + 0.5	Volts	
V_{OL}	Output Low Voltage		0.45	Volts	I_a = 2.5 mA for $\overline{S0}$–$\overline{S2}$ I_a = 2.0 mA for all other outputs
V_{OH}	Output High Voltage		2.4	Volts	I_{oa} = −400 μA
I_{CC}	Power Supply Current		550	mA	T_A = 25°C
I_{LI}	Input Leakage Current		±10	μA	0V < V_{IN} < V_{CC}
I_{LO}	Output Leakage Current		±10	μA	0.45V < V_{OUT} < V_{CC}
V_{CLO}	Clock Output Low		0.6	Volts	I_a = 2.5 mA
V_{CHO}	Clock Output High	4.0		Volts	I_{oa} = −200 μA
V_{CLI}	Clock Input Low Voltage	−0.5	0.6	Volts	
V_{CHI}	Clock Input High Voltage	3.9	V_{CC} + 1.0	Volts	
C_{IN}	Input Capacitance		10	pF	
C_{IO}	I/O Capacitance		20	pF	

PIN TIMINGS

A.C. CHARACTERISTICS (T_A = 0°–70°C, V_{CC} = 5V ± 10%)

80186 Timing Requirements All Timings Measured At 1.5 Volts Unless Otherwise Noted.

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TDVCL	Data in Setup (A/D)	20		ns	
TCLDX	Data in Hold (A/D)	10		ns	
TARYHCH	Asynchronous Ready (AREADY) active setup time*	20		ns	
TARYLCL	AREADY inactive setup time	35		ns	
TCHARYX	AREADY hold time	15		ns	
TSRYCL	Synchronous Ready (SREADY) transition setup time	35		ns	
TCLSRY	SREADY transition hold time	15		ns	
THVCL	HOLD Setup*	25		ns	
TINVCH	INTR, NMI, $\overline{TEST}$, TIMERIN, Setup*	25		ns	
TINVCL	DRQ0, DRQ1, Setup*	25		ns	

*To guarantee recognition at next clock.

A.C. CHARACTERISTICS (Continued)

80186 Master Interface Timing Responses

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TCLAV	Address Valid Delay	10	44	ns	C_L = 20–200 pF all outputs
TCLAX	Address Hold	10		ns	
TCLAZ	Address Float Delay	TCLAX	35	ns	
TCHCZ	Command Lines Float Delay		45	ns	
TCHCV	Command Lines Valid Delay (after float)		55	ns	
TLHLL	ALE Width	TCLCL-35		ns	
TCHLH	ALE Active Delay		35	ns	
TCHLL	ALE Inactive Delay		35	ns	
TLLAX	Address Hold to ALE Inactive	TCHCL-25		ns	
TCLDV	Data Valid Delay	10	44	ns	
TCHDX	Data Hold Time	10		ns	
TWHDX	Data Hold after $\overline{WR}$	TCLCL-40		ns	
TCVCTV	Control Active Delay1	10	70	ns	
TCHCTV	Control Active Delay2	10	55	ns	
TCVCTX	Control Inactive Delay	10	55	ns	
TAZRL	Address Float to $\overline{RD}$ Active	0		ns	
TCLRL	$\overline{RD}$ Active Delay	10	70	ns	
TCLRH	$\overline{RD}$ Inactive Delay	10	55	ns	
TRHAV	$\overline{RD}$ Inactive to Address Active	TCLCL-40		ns	
TCLHAV	HLDA Valid Delay	10	50	ns	
TRLRH	$\overline{RD}$ Width	2TCLCL-50		ns	
TWLWH	$\overline{WR}$ Width	2TCLCL-40		ns	
TAVAL	Address Valid to ALE Low	TCLCH-25		ns	
TCHSV	Status Active Delay	10	55	ns	
TCLSH	Status Inactive Delay	10	55	ns	
TCLTMV	Timer Output Delay		60	ns	100 pf max
TCLRO	Reset Delay		60	ns	

80186 Chip-Select Timing Responses

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TCLCSV	Chip-Select Active Delay		66	ns	
TCXCSX	Chip-Select Hold from Command Inactive	35		ns	
TCHCSX	Chip-Select Inactive Delay	10	35	ns	

80186 CLKIN Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TCKIN	CLKIN Period	62.5	250	ns	
TCKHL	CLKIN Fall Time		10	ns	3.5 to 1.0 volts
TCKLH	CLKIN Rise Time		10	ns	1.0 to 3.5 volts
TCLCK	CLKIN Low Time	25		ns	1.5 volts
TCHCK	CLKIN High Time	25		ns	1.5 volts

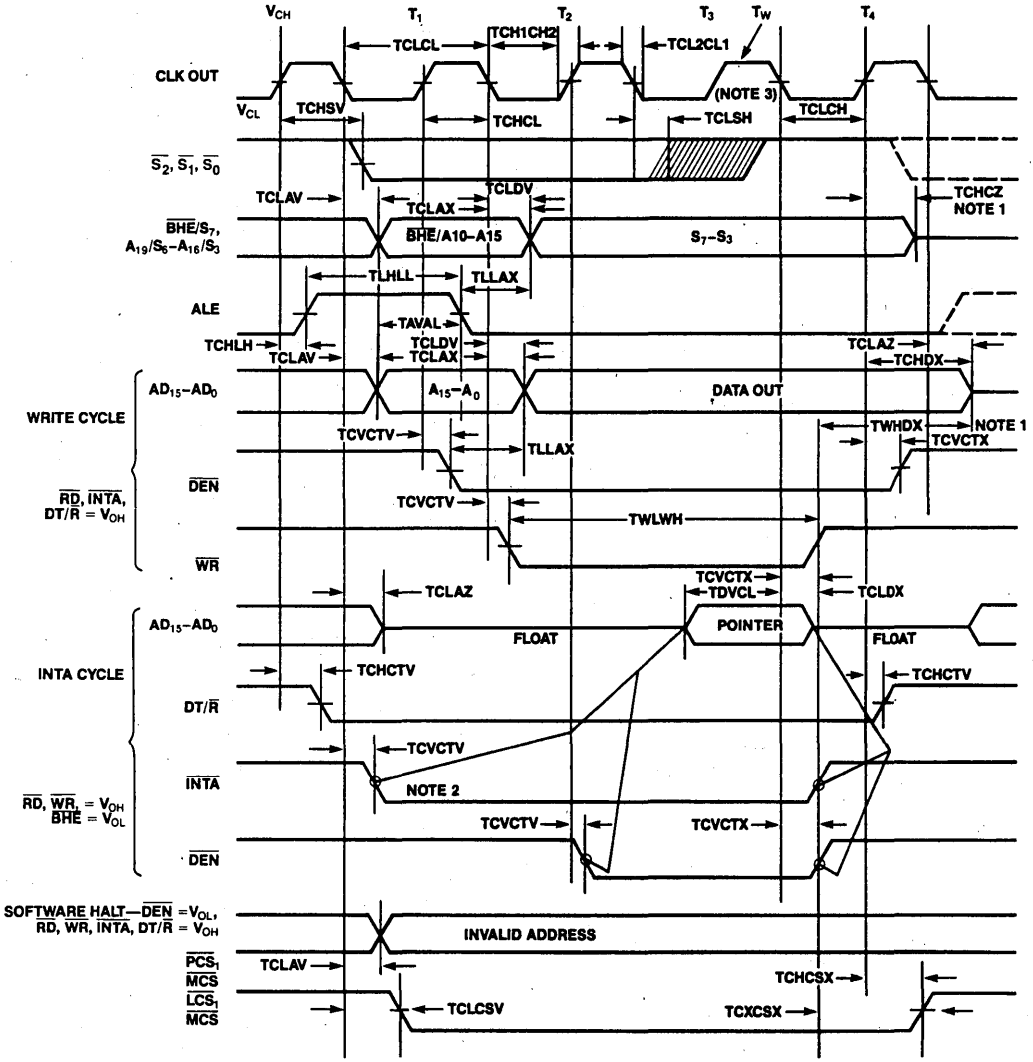
80186 CLKOUT Timing (200 pF load)

Symbol	Parameter	Min.	Max.	Units	Test Conditions
TCICO	CLKIN to CLKOUT Skew		50	ns	
TCLCL	CLKOUT Period	125	500	ns	
TCLCH	CLKOUT Low Time	55		ns	1.5 volts
TCHCL	CLKOUT High Time	55		ns	1.5 volts
TCH1CH2	CLKOUT Rise Time		15	ns	1.0 to 3.5 volts
TCL2CL1	CLKOUT Fall Time		15	ns	3.5 to 1.0 volts

All timings measured at 1.5 volts unless otherwise noted.

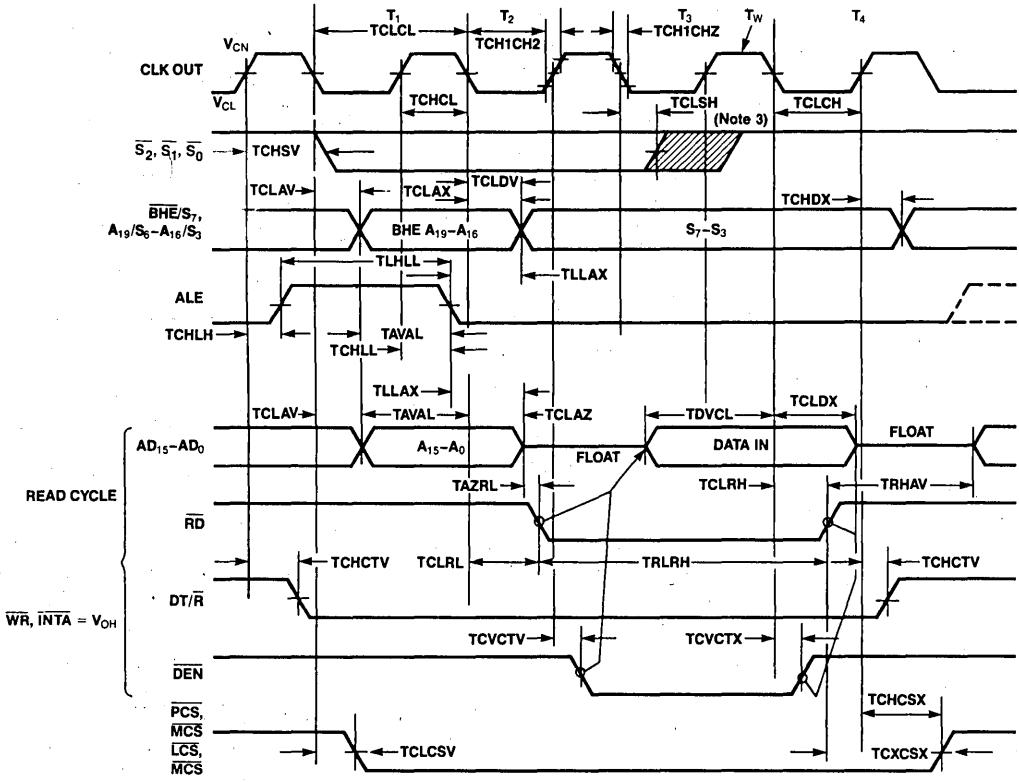
WAVEFORMS

MAJOR CYCLE TIMING



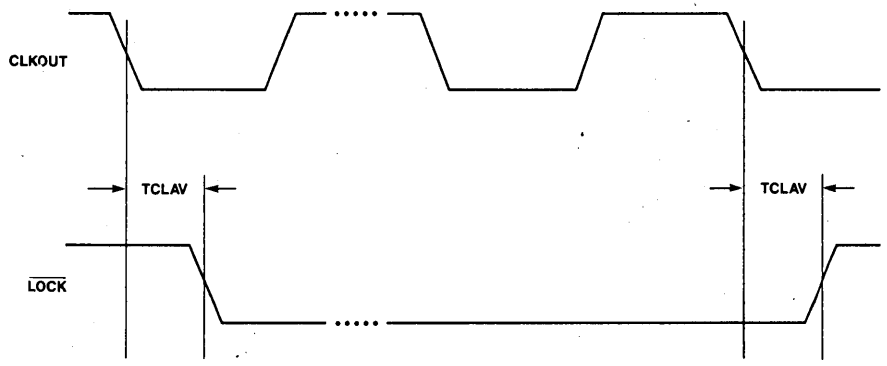
WAVEFORMS (Continued)

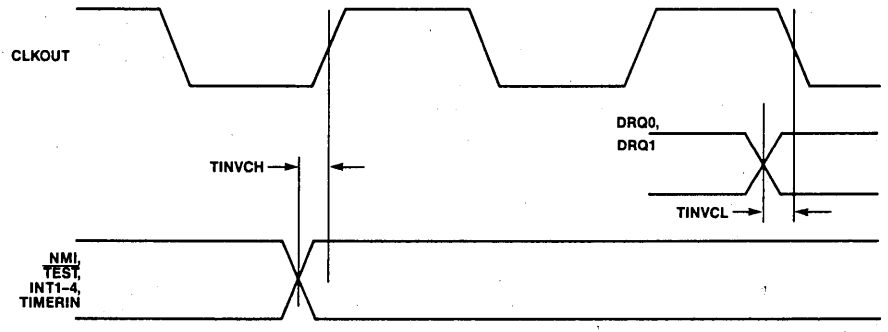
MAJOR CYCLE TIMING (Continued)



NOTES:
1. Following a Write cyclé, the Local Bus is floated by the 80186 only when the 80186 enters a "Hold Acknowledge" state.
2. INTA occurs one clock later in RMX-mode.
3. Status inactive just prior to T_4

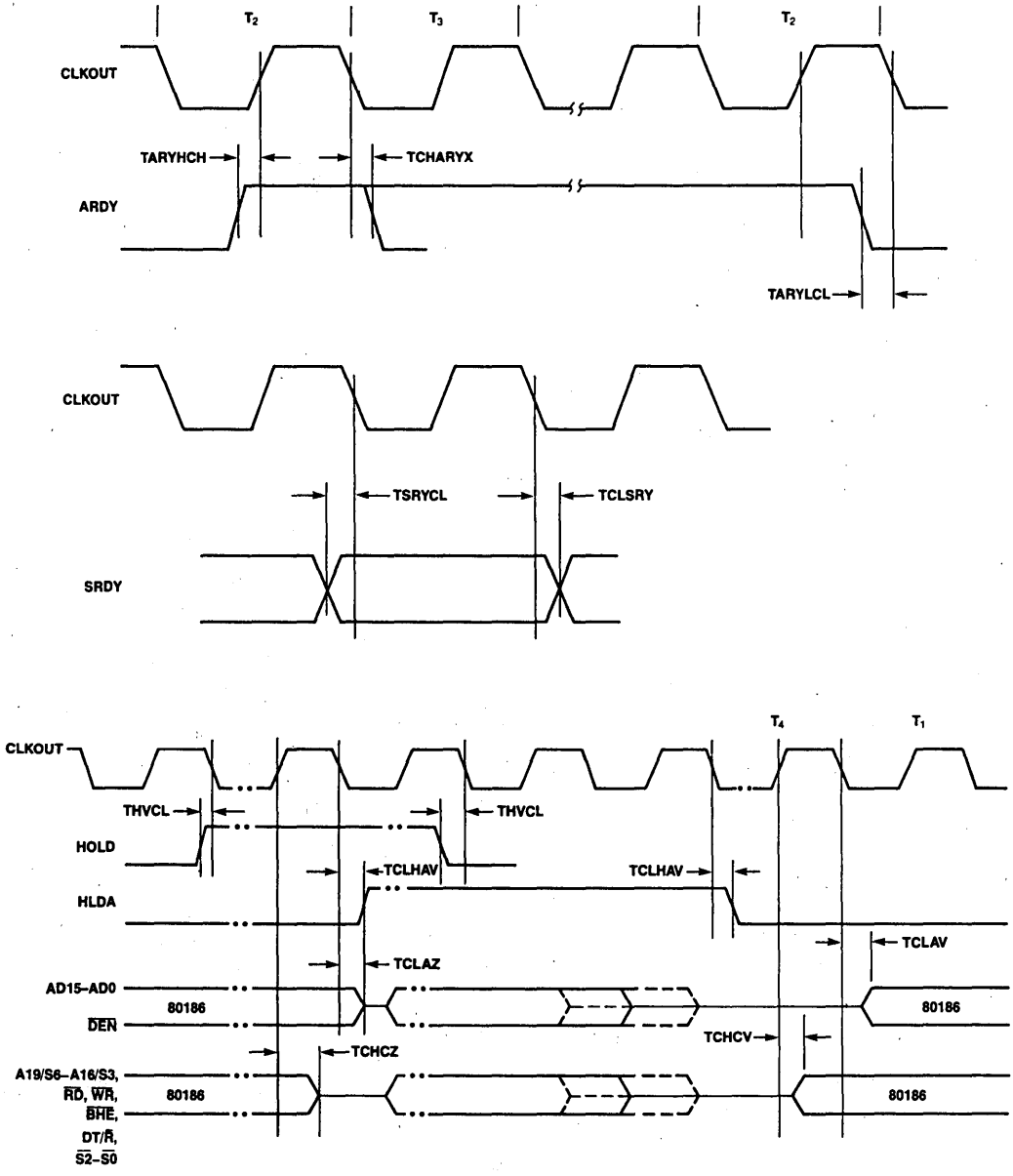
WAVEFORMS (Continued)





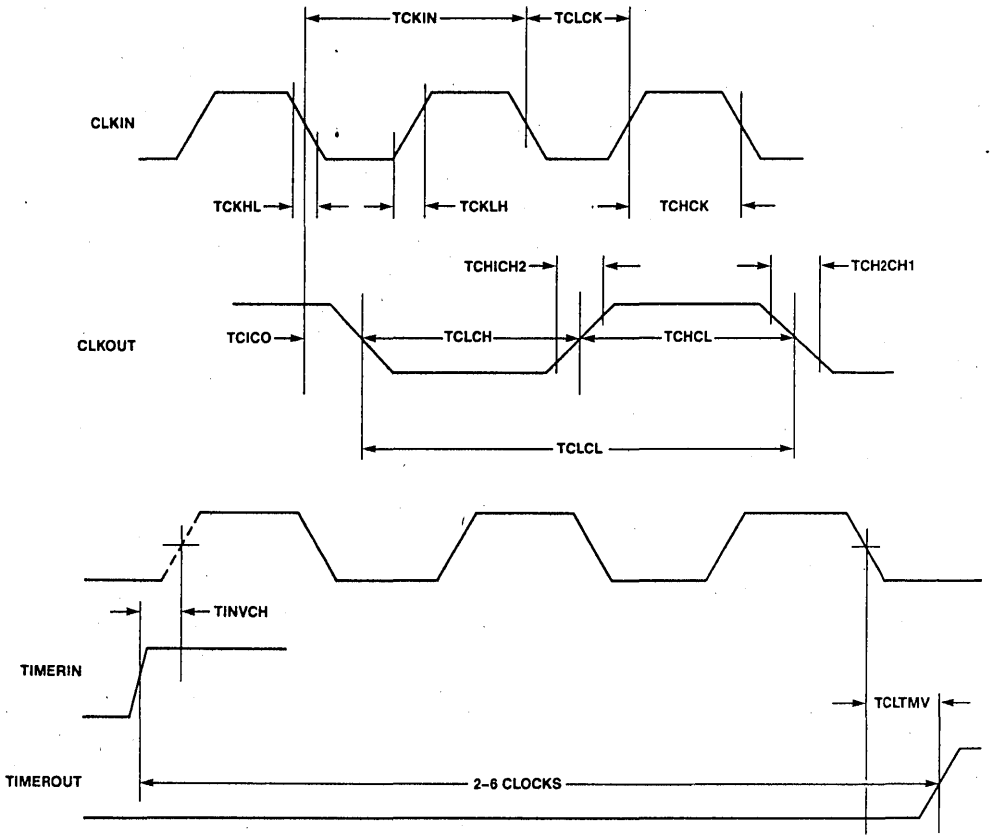
WAVEFORMS (Continued)

HOLD-HLDA TIMING



WAVEFORMS (Continued)

TIMER ON 80186



80186 INSTRUCTION TIMINGS

The following instruction timings represent the minimum execution time in clock cycles for each instruction. The timings given are based on the following assumptions:

- The opcode, along with any data or displacement required for execution of a particular instruction, has been prefetched and resides in the queue at the time it is needed.
- No wait states or bus HOLDS occur.
- All word-data is located on even-address boundaries.

All jumps and calls include the time required to fetch the opcode of the next instruction at the destination address.

All instructions which involve memory reference can require one (and in some cases, two) additional clocks above the minimum timings shown. This is due to the asynchronous nature of the handshake between the BIU and the Execution unit.

INSTRUCTION SET SUMMARY

FUNCTION	FORMAT	Clock Cycles	Comments
DATA TRANSFER			
MOV = Move:			
Register to Register/Memory	1 0 0 0 1 0 0 w | mod reg r/m	2/12	
Register/memory to register	1 0 0 0 1 0 1 w | mod reg r/m	2/9	
Immediate to register/memory	1 1 0 0 0 1 1 w | mod 0 0 0 r/m | data | data if w = 1	12–13	8/16-bit
Immediate to register	1 0 1 1 w reg | data | data if w = 1	3–4	8/16-bit
Memory to accumulator	1 0 1 0 0 0 0 w | addr-low | addr-high	9	
Accumulator to memory	1 0 1 0 0 0 1 w | addr-low | addr-high	8	
Register/memory to segment register	1 0 0 0 1 1 1 0 | mod 0 reg r/m	2/9	
Segment register to register/memory	1 0 0 0 1 1 0 0 | mod 0 reg r/m	2/11	
PUSH = Push:			
Memory	1 1 1 1 1 1 1 1 | mod 1 1 0 r/m	16	
Register	0 1 0 1 0 reg	10	
Segment register	0 0 0 reg 1 1 0	9	
Immediate	0 1 1 0 1 0 s 0 | data | data if s = 0	10	
PUSHA = Push All	0 1 1 0 0 0 0 0	36	
POP = Pop:			
Memory	1 0 0 0 1 1 1 1 | mod 0 0 0 r/m	20	
Register	0 1 0 1 1 reg	10	
Segment register	0 0 0 reg 1 1 1 | (reg ≠ 01)	8	
POPA = Pop All	0 1 1 0 0 0 0 1	51	
XCHG = Exchange:			
Register/memory with register	1 0 0 0 0 1 1 w | mod reg r/m	4/17	
Register with accumulator	1 0 0 1 0 reg	3	
IN = Input from:			
Fixed port	1 1 1 0 0 1 0 w | port	10	
Variable port	1 1 1 0 1 1 0 w	8	
OUT = Output to:			
Fixed port	1 1 1 0 0 1 1 w | port	9	
Variable port	1 1 1 0 1 1 1 w	7	
XLAT = Translate byte to AL	1 1 0 1 0 1 1 1	11	
LEA = Load EA to register	1 0 0 0 1 1 0 1 | mod reg r/m	6	
LDS = Load pointer to DS	1 1 0 0 0 1 0 1 | mod reg r/m | (mod ≠ 11)	18	
LES = Load pointer to ES	1 1 0 0 0 1 0 0 | mod reg r/m | (mod ≠ 11)	18	
LAHF = Load AH with flags	1 0 0 1 1 1 1 1	2	
SAHF = Store AH into flags	1 0 0 1 1 1 1 0	3	
PUSHF = Push flags	1 0 0 1 1 1 0 0	9	
POPF = Pop flags	1 0 0 1 1 1 0 1	8	

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright of Intel Corp. 1983.

INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	Clock Cycles	Comments
ARITHMETIC			
ADD = Add:			
Reg/memory with register to either	0 0 0 0 0 0 d w | mod reg r/m	3/10	
Immediate to register/memory	1 0 0 0 0 0 s w | mod 0 0 0 r/m | data | data if s w = 01	4/16	
Immediate to accumulator	0 0 0 0 0 1 0 w | data | data if w = 1	3/4	8/16-bit
ADC = Add with carry:			
Reg/memory with register to either	0 0 0 1 0 0 d w | mod reg r/m	3/10	
Immediate to register/memory	1 0 0 0 0 0 s w | mod 0 1 0 r/m | data | data if s w = 01	4/16	
Immediate to accumulator	0 0 0 1 0 1 0 w | data | data if w = 1	3/4	8/16-bit
INC = Increment:			
Register/memory	1 1 1 1 1 1 1 w | mod 0 0 0 r/m	3/15	
Register	0 1 0 0 0 reg	3	
SUB = Subtract:			
Reg/memory and register to either	0 0 1 0 1 0 d w | mod reg r/m	3/10	
Immediate from register/memory	1 0 0 0 0 0 s w | mod 1 0 1 r/m | data | data if s w = 0 1	4/16	
Immediate from accumulator	0 0 1 0 1 1 0 w | data | data if w = 1	3/4	8/16-bit
SBB = Subtract with borrow:			
Reg/memory and register to either	0 0 0 1 1 0 d w | mod reg r/m	3/10	
Immediate from register/memory	1 0 0 0 0 0 s w | mod 0 1 1 r/m | data | data if s w = 0 1	4/16	
Immediate from accumulator	0 0 0 1 1 1 0 w | data | data if w = 1	3/4	8/16-bit
DEC = Decrement:			
Register/memory	1 1 1 1 1 1 1 w | mod 0 0 1 r/m	3/15	
Register	0 1 0 0 1 reg	3	
CMP = Compare:			
Register/memory with register	0 0 1 1 1 0 1 w | mod reg r/m	3/10	
Register with register/memory	0 0 1 1 1 0 0 w | mod reg r/m	3/10	
Immediate with register/memory	1 0 0 0 0 0 s w | mod 1 1 1 r/m | data | data if s w = 0 1	3/10	
Immediate with accumulator	0 0 1 1 1 1 0 w | data | data if w = 1	3/4	8/16-bit
NEG = Change sign	1 1 1 1 0 1 1 w | mod 0 1 1 r/m	3	
AAA = ASCII adjust for add	0 0 1 1 0 1 1 1	8	
DAA = Decimal adjust for add	0 0 1 0 0 1 1 1	4	
AAS = ASCII adjust for subtract	0 0 1 1 1 1 1 1	7	
DAS = Decimal adjust for subtract	0 0 1 0 1 1 1 1	4	
MUL = Multiply (unsigned):	1 1 1 1 0 1 1 w | mod 1 0 0 r/m		
Register-Byte		26–28	
Register-Word		35–37	
Memory-Byte		32–34	
Memory-Word		41–43	
IMUL = Integer multiply (signed):	1 1 1 1 0 1 1 w | mod 1 0 1 r/m		
Register-Byte		25–28	
Register-Word		34–37	
Memory-Byte		31–34	
Memory-Word		40–43	
IMUL = Integer immediate multiply (signed)	0 1 1 0 1 0 s 1 | mod reg r/m | data | data if s = 0	22–25/29–32	
DIV = Divide (unsigned):	1 1 1 1 0 1 1 w | mod 1 1 0 r/m		
Register-Byte		29	
Register-Word		38	
Memory-Byte		35	
Memory-Word		44	

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright of Intel Corp. 1983.

INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	Clock Cycles	Comments
ARITHMETIC (Continued):			
IDIV = Integer divide (signed):	1 1 1 1 0 1 1 w | mod 1 1 1 r/m		
Register-Byte		44–52	
Register-Word		53–61	
Memory-Byte		50–58	
Memory-Word		59–67	
AAM = ASCII adjust for multiply	1 1 0 1 0 1 0 0 | 0 0 0 0 1 0 1 0	19	
AAD = ASCII adjust for divide	1 1 0 1 0 1 0 1 | 0 0 0 0 1 0 1 0	15	
CBW = Convert byte to word	1 0 0 1 1 0 0 0	2	
CWD = Convert word to double word	1 0 0 1 1 0 0 1	4	
LOGIC			
Shift/Rotate Instructions:			
Register/Memory by 1	1 1 0 1 0 0 0 w | mod TTT r/m	2/15	
Register/Memory by CL	1 1 0 1 0 0 1 w | mod TTT r/m		
Register/Memory by Count	1 1 0 0 0 0 0 w | mod TTT r/m | count	5+n/17+n	
	TTT Instruction 0 0 0 ROL 0 0 1 ROR 0 1 0 RCL 0 1 1 RCR 1 0 0 SHL/SAL 1 0 1 SHR 1 1 1 SAR		
AND = And:			
Reg/memory and register to either	0 0 1 0 0 0 d w | mod reg r/m	3/10	
Immediate to register/memory	1 0 0 0 0 0 0 w | mod 1 0 0 r/m | data | data if w = 1	4/16	
Immediate to accumulator	0 0 1 0 0 1 0 w | data | data if w = 1	3/4	8/16-bit
TEST = And function to flags, no result:			
Register/memory and register	1 0 0 0 0 1 0 w | mod reg r/m	3/10	
Immediate data and register/memory	1 1 1 1 0 1 1 w | mod 0 0 0 r/m | data | data if w = 1	4/10	
Immediate data and accumulator	1 0 1 0 1 0 0 w | data | data if w = 1	3/4	8/16-bit
OR = Or:			
Reg/memory and register to either	0 0 0 0 1 0 d w | mod reg r/m	3/10	
Immediate to register/memory	1 0 0 0 0 0 0 w | mod 0 0 1 r/m | data | data if w = 1	4/16	
Immediate to accumulator	0 0 0 0 1 1 0 w | data | data if w = 1	3/4	8/16-bit
XOR = Exclusive or:			
Reg/memory and register to either	0 0 1 1 0 0 d w | mod reg r/m	3/10	
Immediate to register/memory	1 0 0 0 0 0 0 w | mod 1 1 0 r/m | data | data if w = 1	4/16	
Immediate to accumulator	0 0 1 1 0 1 0 w | data | data if w = 1	3/4	8/16-bit
NOT = Invert register/memory	1 1 1 1 0 1 1 w | mod 0 1 0 r/m	3	
STRING MANIPULATION:			
MOVS = Move byte/word	1 0 1 0 0 1 0 w	8+8n	
CMPS = Compare byte/word	1 0 1 0 0 1 1 w	5+22n	
SCAS = Scan byte/word	1 0 1 0 1 1 1 w	5+15n	
LODS = Load byte/wd to AL/AX	1 0 1 0 1 1 0 w	6+11n	
STOS = Stor byte/wd from AL/A	1 0 1 0 1 0 1 w	6+9n	
INS = Input byte/wd from DX port	0 1 1 0 1 1 0 w	8	
OUTS = Output byte/wd to DX port	0 1 1 0 1 1 1 w	7	

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright of Intel Corp. 1983.

INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	Clock Cycles	Comments
STRING MANIPULATION (Continued):			
Repeated by count in CX			
MOVS = Move string	1 1 1 1 0 0 1 0 | 1 0 1 0 0 1 0 w	14	
CMPS = Compare string	1 1 1 1 0 0 1 z | 1 0 1 0 0 1 1 w	22	
SCAS = Scan string	1 1 1 1 0 0 1 z | 1 0 1 0 1 1 1 w	15	
LODS = Load string	1 1 1 1 0 0 1 0 | 1 0 1 0 1 1 0 w	12	
STOS = Store string	1 1 1 1 0 0 1 0 | 1 0 1 0 1 0 1 w	10	
INS = Input string	1 1 1 1 0 0 1 0 | 0 1 1 0 1 1 0 w	8+8n/14	Repeated/ Not Repeated
OUTS = Output string	1 1 1 1 0 0 1 0 | 0 1 1 0 1 1 1 w	8+8n/14	Repeated Not Repeated
CONTROL TRANSFER			
CALL = Call:			
Direct within segment	1 1 1 0 1 0 0 0 | disp-low | disp-high	14	
Register memory indirect within segment	1 1 1 1 1 1 1 1 | mod 0 1 0 r/m	13/19	
Direct intersegment	1 0 0 1 1 0 1 0 | segment offset | segment selector	23	
Indirect intersegment	1 1 1 1 1 1 1 1 | mod 0 1 1 r/m (mod ≠ 11)	38	
JMP = Unconditional jump:			
Short long	1 1 1 0 1 0 1 1 | disp-low	13	
Direct within segment	1 1 1 0 1 0 0 1 | disp-low | disp-high	13	
Register memory indirect within segment	1 1 1 1 1 1 1 1 | mod 1 0 0 r/m	11/17	
Direct intersegment	1 1 1 0 1 0 1 0 | segment offset | segment selector	13	
Indirect intersegment	1 1 1 1 1 1 1 1 | mod 1 0 1 r/m (mod ≠ 11)	26	
RET = Return from CALL:			
Within segment	1 1 0 0 0 0 1 1	16	
Within seg adding immed to SP	1 1 0 0 0 0 1 0 | data-low | data-high	18	
Intersegment	1 1 0 0 1 0 1 1	22	
Intersegment adding immediate to SP	1 1 0 0 1 0 1 0 | data-low | data-high	25	

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright of Intel Corp. 1983.

INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	Clock Cycles	Comments
CONTROL TRANSFER (Continued):			
JE/JZ = Jump on equal zero	0 1 1 1 0 1 0 0 | disp	4/13	13 if JMP taken 4 if JMP not taken
JL/JNGE = Jump on less not greater or equal	0 1 1 1 1 1 0 0 | disp	4/13	
JLE/JNG = Jump on less or equal not greater	0 1 1 1 1 1 1 0 | disp	4/13	
JB/JNAE = Jump on below not above or equal	0 1 1 1 0 0 1 0 | disp	4/13	
JBE/JNA = Jump on below or equal not above	0 1 1 1 0 1 1 0 | disp	4/13	
JP/JPE = Jump on parity parity even	0 1 1 1 1 0 1 0 | disp	4/13	
JO = Jump on overflow	0 1 1 1 0 0 0 0 | disp	4/13	
JS = Jump on sign	0 1 1 1 1 0 0 0 | disp	4/13	
JNE/JNZ = Jump on not equal not zero	0 1 1 1 0 1 0 1 | disp	4/13	
JNL/JGE = Jump on not less greater or equal	0 1 1 1 1 1 0 1 | disp	4/13	
JNLE/JG = Jump on not less or equal greater	0 1 1 1 1 1 1 1 | disp	4/13	
JNB/JAE = Jump on not below above or equal	0 1 1 1 0 0 1 1 | disp	4/13	
JNBE/JA = Jump on not below or equal above	0 1 1 1 0 1 1 1 | disp	4/13	
JNP/JPO = Jump on not par par odd	0 1 1 1 1 0 1 1 | disp	4/13	
JNO = Jump on not overflow	0 1 1 1 0 0 0 1 | disp	4/13	
JNS = Jump on not sign	0 1 1 1 1 0 0 1 | disp	4/13	
LOOP = Loop CX times	1 1 1 0 0 0 1 0 | disp	5/15	
LOOPZ/LOOPE = Loop while zero equal	1 1 1 0 0 0 0 1 | disp	6/16	
LOOPNZ/LOOPNE = Loop while not zero equal	1 1 1 0 0 0 0 0 | disp	6/16	
JCXZ = Jump on CX zero	1 1 1 0 0 0 1 1 | disp	16 5	JMP taken/ JMP not taken
ENTER = Enter Procedure	1 1 0 0 1 0 0 0 | data-low | data-high | L		
L = 0		15	
L = 1		25	
L > 1		22+16(n−1)	
LEAVE = Leave Procedure	1 1 0 0 1 0 0 1	8	
INT = Interrupt:			
Type specified	1 1 0 0 1 1 0 1 | type	47	
Type 3	1 1 0 0 1 1 0 0	45	if INT. taken/ if INT. not taken
INTO = Interrupt on overflow	1 1 0 0 1 1 1 0	48/4	
IRET = Interrupt return	1 1 0 0 1 1 1 1	28	
BOUND = Detect value out of range	0 1 1 0 0 0 1 0 | mod reg r/m	33–35	

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright of Intel Corp. 1983.

INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	Clock Cycles	Comments
PROCESSOR CONTROL			
CLC = Clear carry	1 1 1 1 1 0 0 0	2	
CMC = Complement carry	1 1 1 1 0 1 0 1	2	
STC = Set carry	1 1 1 1 1 0 0 1	2	
CLD = Clear direction	1 1 1 1 1 1 0 0	2	
STD = Set direction	1 1 1 1 1 1 0 1	2	
CLI = Clear interrupt	1 1 1 1 1 0 1 0	2	
STI = Set interrupt	1 1 1 1 1 0 1 1	2	
HLT = Halt	1 1 1 1 0 1 0 0	2	
WAIT = Wait	1 0 0 1 1 0 1 1	6	if $\overline{\text{test}}$ = 0
LOCK = Bus lock prefix	1 1 1 1 0 0 0 0	2	
ESC = Processor Extension Escape	1 0 0 1 1 T T T | mod LLL r/m (TTT LLL are opcode to processor extension)	6	

FOOTNOTES

The effective Address (EA) of the memory operand is computed according to the mod and r/m fields:

if mod = 11 then r/m is treated as a REG field

if mod = 00 then DISP = 0*, disp-low and disp-high are absent

if mod = 01 then DISP = disp-low sign-extended to 16-bits, disp-high is absent

if mod = 10 then DISP = disp-high: disp-low

if r/m = 000 then EA = (BX) + (SI) + DISP

if r/m = 001 then EA = (BX) + (DI) + DISP

if r/m = 010 then EA = (BP) + (SI) + DISP

if r/m = 011 then EA = (BP) + (DI) + DISP

if r/m = 100 then EA = (SI) + DISP

if r/m = 101 then EA = (DI) + DISP

if r/m = 110 then EA = (BP) + DISP*

if r/m = 111 then EA = (BX) + DISP

DISP follows 2nd byte of instruction (before data if required)

*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

SEGMENT OVERRIDE PREFIX

0 0 1 reg 1 1 0

reg is assigned according to the following:

reg	Segment Register
00	ES
01	CS
10	SS
11	DS

REG is assigned according to the following table:

16-Bit (w = 1)	8-Bit (w = 0)
000 AX	000 AL
001 CX	001 CL
010 DX	010 DL
011 BX	011 BL
100 SP	100 AH
101 BP	101 CH
110 SI	110 DH
111 DI	111 BH

The physical addresses of all operands addressed by the BP register are computed using the SS segment register. The physical addresses of the destination operands of the string primitive operations (those addressed by the DI register) are computed using the ES segment, which may not be overridden.

iAPX 286

High Performance Microprocessor with Memory Management and Protection

ADVANCE INFORMATION

DISTINCTIVE CHARACTERISTICS

- High performance 8 and 10MHz processor (up to six times iAPX 86)
- Large address space
  - 16 megabytes physical
  - 1 gigabyte virtual per task
- Integrated memory management, four-level memory protection and support for virtual memory and operating systems
- Two iAPX 86 upward compatible operating modes
  - iAPX 86 real address mode
  - Protected virtual address mode
- Optional processor extension
  - iAPX 286/287 high performance 80-bit numeric data processor
- High bandwidth bus interface (8 or 10 megabyte/sec)

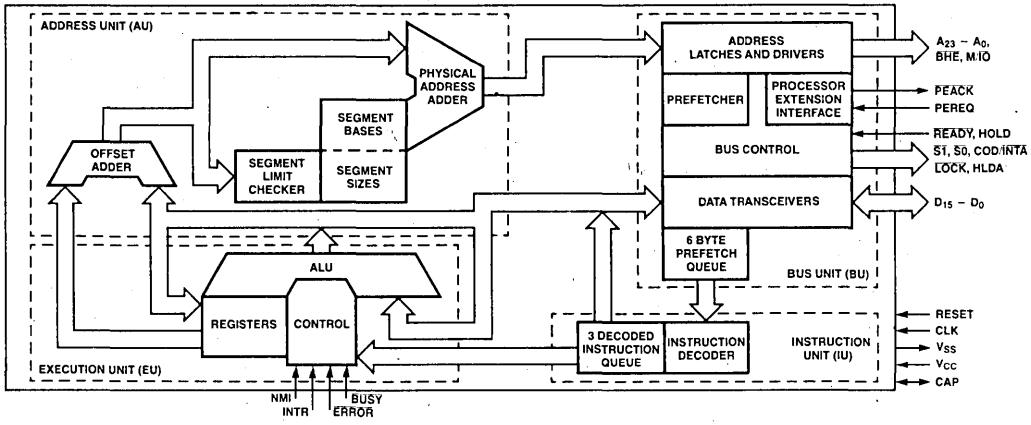
GENERAL DESCRIPTION

The iAPX 286 (80286 part number) is an advanced, high performance microprocessor with specially optimized capabilities for multiple user and multi-tasking systems. The 80286 has built-in memory protection that supports operating system and task isolation as well as program and data privacy within tasks. A 10MHz iAPX 286 provides up to six times greater throughput than the standard 5MHz iAPX 86. The 80286 includes memory management capabilities that map up to 2^{30} bytes (one gigabyte) of virtual address space per task into 2^{24} bytes (16 megabytes) of physical memory.

The iAPX 286 is upward compatible with iAPX 86 and 88 software. Using iAPX 86 real address mode, the 80286 is object code compatible with existing iAPX 86, 88 software. In protected virtual address mode, the 80286 is source code compatible with iAPX 86, 88 software and may require upgrading to use virtual addresses supported by the 80286's integrated memory management and protection mechanism. Both modes operate at full 80286 performance and execute a superset of the iAPX 86 and 88's instructions.

The 80286 provides special operations to support the efficient implementation and execution of operating systems. For example, one instruction can end execution of one task, save its state, switch to a new task, load its state, and start execution of the new task. The 80286 also supports virtual memory systems by providing a segment-not-present exception and restartable instructions.

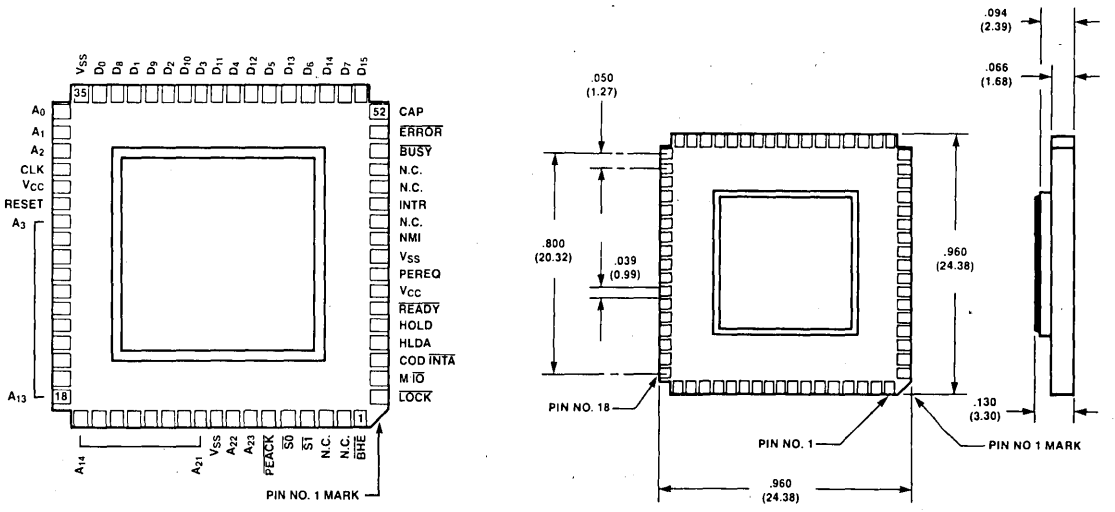
Figure 1. 80286 Internal Block Diagram



Reprinted by permission of Intel Corp., copyright 1983.

Figure 2. 80286 Pin Configuration

Component Pad View - Viewed From Underside of Component When Mounted on Board



Note: N.C. pads must not be connected.

Table 1. Pin Description

The following pin function descriptions are for the 80286 microprocessor:

Symbol	Type	Name and Function
CLK	I	**System Clock** provides the fundamental timing for iAPX 286 systems. It is a 16 MHz signal divided by two inside the 80286 to generate the 8 MHz processor clock. The internal divide-by-two circuitry can be synchronized to an external clock generator by a LOW to HIGH transition on the RESET input.
D_{15}-D_0	I/O	**Data Bus** inputs data during memory, I/O, and interrupt acknowledge read cycles; outputs data during memory and I/O write cycles. The data bus is active HIGH and floats to 3-state OFF during bus hold acknowledge.
A_{23}-A_0	O	**Address Bus** outputs physical memory and I/O port addresses. A0 is LOW when data is to be transferred on pins D_{7-0}. A_{23}-A_{16} are LOW during I/O transfers. The address bus is active HIGH and floats to 3-state OFF during bus hold acknowledge.
$\overline{BHE}$	O	**Bus High Enable** indicates transfer of data on the upper byte of the data bus, D_{15-8}. Eight-bit oriented devices assigned to the upper byte of the data bus would normally use $\overline{BHE}$ to condition chip select functions. $\overline{BHE}$ is active LOW and floats to 3-state OFF during bus hold acknowledge.

$\overline{BHE}$ and A0 Encodings

$\overline{BHE}$ Value	A0 Value	Function
0	0	Word transfer
0	1	Byte transfer on upper half of data bus (D_{15-8})
1	0	Byte transfer on lower half of data bus (D_{7-0})
1	1	Reserved

Table 1. Pin Description (Cont.)

Symbol	Type	Name and Function
$\overline{S1}$, $\overline{S0}$	O	**Bus Cycle Status** indicates initiation of a bus cycle and, along with M/IO and COD/INTA, defines the type of bus cycle. The bus is in a T_S state whenever one or both are LOW. S1 and S0 are active LOW and float to 3-state OFF during bus hold acknowledge.

80286 Bus Cycle Status Definition

COD/$\overline{INTA}$	M/$\overline{IO}$	$\overline{S1}$	$\overline{S0}$	Bus cycle initiated
0 (LOW)	0	0	0	Interrupt acknowledge
0	0	0	1	Reserved
0	0	1	0	Reserved
0	0	1	1	None; not a status cycle
0	1	0	0	IF A1 = 1 then halt; else shutdown
0	1	0	1	Memory data read
0	1	1	0	Memory data write
0	1	1	1	None; not a status cycle
1 (HIGH)	0	0	0	Reserved
1	0	0	1	I/O read
1	0	1	0	I/O write
1	0	1	1	None; not a status cycle
1	1	0	0	Reserved
1	1	0	1	Memory instruction read
1	1	1	0	Reserved
1	1	1	1	None; not a status cycle

M/$\overline{IO}$	O	**Memory/IO Select** distinguishes memory access from I/O access. If HIGH during T_S, a memory cycle or a halt/shutdown cycle is in progress. If LOW, an I/O cycle or an interrupt acknowledge cycle is in progress. M/$\overline{IO}$ floats to 3-state OFF during bus hold acknowledge.
COD/$\overline{INTA}$	O	**Code/Interrupt Acknowledge** distinguishes instruction fetch cycles from memory data read cycles. Also distinguishes interrupt acknowledge cycles from I/O cycles. COD/$\overline{INTA}$ floats to 3-state OFF during bus hold acknowledge.
$\overline{LOCK}$	O	**Bus Lock** indicates that other system bus masters are not to gain control of the system bus following the current bus cycle. The $\overline{LOCK}$ signal may be activated explicitly by the "LOCK" instruction prefix or automatically by 80286 hardware during memory XCHG instructions, interrupt acknowledge, or descriptor table access. $\overline{LOCK}$ is active LOW and floats to 3-state OFF during bus hold acknowledge.
$\overline{READY}$	I	**Bus Ready** terminates a bus cycle. Bus cycles are extended without limit until terminated by $\overline{READY}$ LOW. $\overline{READY}$ is an active LOW synchronous input requiring setup and hold times relative to the system clock be met for correct operation. $\overline{READY}$ is ignored during bus hold acknowledge.
HOLD HLDA	I O	**Bus Hold Request and Hold Acknowledge** control ownership of the 80286 local bus. The HOLD input allows another local bus master to request control of the local bus. When control is granted, the 80286 will float its bus drivers to 3-state OFF and then activate HLDA, thus entering the bus hold acknowledge condition. The local bus will remain granted to the requesting master until HOLD becomes inactive which results in the 80286 deactivating HLDA and regaining control of the local bus. This terminates the bus hold acknowledge condition. HOLD may be asynchronous to the system clock. These signals are active HIGH.
INTR	I	**Interrupt Request** requests the 80286 to suspend its current program execution and service a pending external request. Interrupt requests are masked whenever the interrupt enable bit in the flag word is cleared. When the 80286 responds to an interrupt request, it performs two interrupt acknowledge bus cycles to read an 8-bit interrupt vector that identifies the source of the interrupt. To assure program interruption, INTR must remain active until the first interrupt acknowledge cycle is completed. INTR is sampled at the beginning of each processor cycle and must be active HIGH at least two processor cycles before the current instruction ends in order to interrupt before the next instruction. INTR is level sensitive, active HIGH, and may be asynchronous to the system clock.
NMI	I	**Non-maskable Interrupt Request** interrupts the 80286 with an internally supplied vector value of 2. No interrupt acknowledge cycles are performed. The interrupt enable bit in the 80286 flag word does not affect this input. The NMI input is active HIGH, may be asynchronous to the system clock, and is edge triggered after internal synchronization. For proper recognition, the input must have been previously LOW for at least four system clock cycles and remain HIGH for at least four system clock cycles.

Table 1. Pin Description (Cont.)

<table>
<tr><th>Symbol</th><th>Type</th><th>Name and Function</th></tr>
<tr><td>PEREQ
PEACK</td><td>I
O</td><td>Processor Extension Operand Request and Acknowledge extend the memory management and protection capabilities of the 80286 to processor extensions. The PEREQ input requests the 80286 to perform a data operand transfer for a processor extension. The PEACK output signals the processor extension when the requested operand is being transferred. PEREQ is active HIGH and may be asynchronous to the system clock. PEACK is active LOW.</td></tr>
<tr><td>BUSY
ERROR</td><td>I
I</td><td>Processor Extension Busy and Error indicate the operating condition of a processor extension to the 80286. An active BUSY input stops 80286 program execution on WAIT and some ESC instructions until BUSY becomes inactive (HIGH). The 80286 may be interrupted while waiting for BUSY to become inactive. An active ERROR input causes the 80286 to perform a processor extension interrupt when executing WAIT or some ESC instructions. These inputs are active LOW and may be asynchronous to the system clock.</td></tr>
<tr><td>RESET</td><td>I</td><td>System Reset clears the internal logic of the 80286 and is active HIGH. The 80286 may be reinitialized at any time with a LOW to HIGH transition on RESET which remains active for more than 16 system clock cycles. During RESET active, the output pins of the 80286 enter the state shown below:
<table>
<tr><th colspan="2">80286 Pin State During Reset</th></tr>
<tr><th>Pin Value</th><th>Pin Names</th></tr>
<tr><td>1 (HIGH)</td><td>$\overline{S0}$, $\overline{S1}$, $\overline{PEACK}$, A23-A0, $\overline{BHE}$, $\overline{LOCK}$</td></tr>
<tr><td>0 (LOW)</td><td>M/$\overline{IO}$, COD/$\overline{INTA}$, HLDA</td></tr>
<tr><td>3-state OFF</td><td>D_{15}–D_0</td></tr>
</table>
Operation of the 80286 begins after a HIGH to LOW transition on RESET. The HIGH to LOW transition of RESET must be synchronous to the system clock. Approximately 50 system clock cycles are required by the 80286 for internal initializations before the first bus cycle to fetch code from the power-on execution address is performed.

A LOW to HIGH transition of RESET synchronous to the system clock, will begin a new processor cycle at the next HIGH to LOW transition of the system clock. The LOW to HIGH transition of RESET may be asynchronous to the system clock; however, in this case it can not be predetermined which phase of the processor clock will occur during the next system clock period. Synchronous LOW to HIGH transitions of RESET are only required for systems where the processor clock must be phase synchronous to another clock.</td></tr>
<tr><td>V_{SS}</td><td>I</td><td>System Ground: 0 VOLTS.</td></tr>
<tr><td>V_{CC}</td><td>I</td><td>System Power: +5 Volt Power Supply.</td></tr>
<tr><td>CAP</td><td>I</td><td>Substrate Filter Capacitor: a 0.047μf ± 20% 12V capacitor must be connected between this pin and ground. This capacitor filters the output of the internal substrate bias generator. A maximum DC leakage current of 1 μa is allowed through the capacitor.

For correct operation of the 80286, the substrate bias generator must charge this capacitor to its operating voltage. The capacitor chargeup time is 5 milliseconds (max.) after V_{CC} and CLK reach their specified AC and DC parameters. RESET may be applied to prevent spurious activity by the CPU during this time. After this time, the 80286 processor clock can be phase synchronized to another clock by pulsing RESET LOW synchronous to the system clock.</td></tr>
</table>

FUNCTIONAL DESCRIPTION

Introduction

The 80286 is an advanced, high-performance microprocessor with specially optimized capabilities for multiple user and multi-tasking systems. Depending on the application, the 80286's performance is up to six times faster than the standard 5 MHz 8086's, while providing complete upward software compatibility with Intel's iAPX 86, 88, and 186 family of CPU's.

The 80286 operates in two modes: iAPX 86 real address mode and protected virtual address mode. Both modes execute a superset of the iAPX 86 and 88 instruction set.

In iAPX 86 real address mode programs use real addresses with up to one megabyte of address space. Programs use virtual addresses in protected virtual address mode, also called protected mode. In protected mode, the 80286 CPU automatically maps 1 gigabyte of virtual addresses per task into a 16 megabyte real address space. This mode also provides memory protection to isolate the operating system and ensure privacy of each tasks' programs and data. Both modes provide the same base instruction set, registers, and addressing modes.

The following Functional Description describes first, the base 80286 architecture common to both modes, second, iAPX 86 real address mode, and third, protected mode.

iAPX 286/10 BASE ARCHITECTURE

The iAPX 86, 88, 186, and 286 CPU family all contain the same basic set of registers, instructions, and addressing modes. The 80286 processor is upward compatible with the 8086, 8088, and 80186 CPU's.

Register Set

The 80286 base architecture has fifteen registers as shown in Figure 3. These registers are grouped into the following four categories:

General Registers: Eight 16-bit general purpose registers used to contain arithmetic and logical operands. Four of these (AX, BX, CX, and DX) can be used either in their entirety as 16-bit words or split into pairs of separate 8-bit registers.

Segment Registers: Four 16-bit special purpose registers select, at any given time, the segments of memory that are immediately addressable for code, stack, and data. (For usage, refer to Memory Organization.)

Base and Index Registers: Four of the general purpose registers may also be used to determine offset addresses of operands in memory. These registers may contain base addresses or indexes to particular locations within a segment. The addressing mode determines the specific registers used for operand address calculations.

Status and Control Registers: Three 16-bit special purpose registers record or control certain aspects of the 80286 processor state. These include the Instruction Pointer, which contains the offset address of the next sequential instruction to be executed.

Figure 3. Register Set

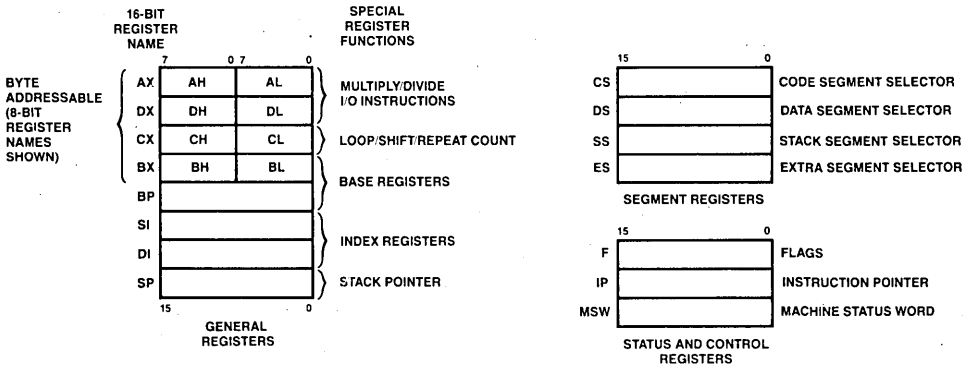
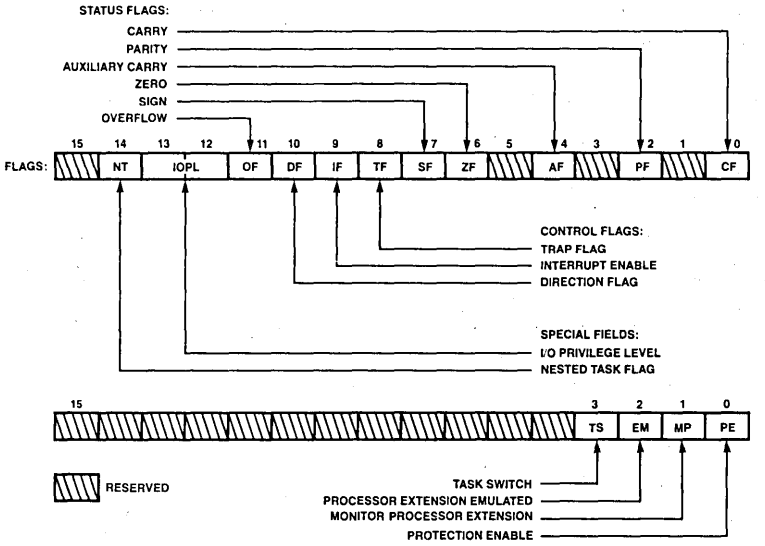


Figure 3a. Status and Control Register Bit Functions



Flags Word Description

The Flags word (Flags) records specific characteristics of the result of logical and arithmetic instructions (bits 0, 2, 4, 6, 7, and 11) and controls the operation of the 80286 within a given operating mode (bits 8 and 9). Flags is a 16-bit register. The function of the flag bits is given in Table 2.

Instruction Set

The instruction set is divided into seven categories: data transfer, arithmetic, shift/rotate/logical, string manipulation, control transfer, high level instructions, and processor control. These categories are summarized in Figure 4.

An 80286 instruction can reference zero, one, or two operands; where an operand resides in a register, in the instruction itself, or in memory. Zero-operand instructions (e.g. NOP and HLT) are usually one byte long. One-operand instructions (e.g. INC and DEC) are usually two bytes long but some are encoded in only one byte. One-operand instructions may reference a register or memory location. Two-operand instructions permit the following six types of instruction operations:

—Register to Register
—Memory to Register
—Immediate to Register
—Memory to Memory
—Register to Memory
—Immediate to Memory

Table 2. Flags Word Bit Functions

Bit Position	Name	Function
0	CF	Carry Flag—Set on high-order bit carry or borrow; cleared otherwise
2	PF	Parity Flag—Set if low-order 8 bits of result contain an even number of 1-bits; cleared otherwise
4	AF	Set on carry from or borrow to the low order four bits of AL; cleared otherwise
6	ZF	Zero Flag–Set if result is zero; cleared otherwise
7	SF	Sign Flag—Set equal to high-order bit of result (0 if positive, 1 if negative)
11	OF	Overflow Flag—Set if result is a too-large positive number or a too-small negative number (excluding sign-bit) to fit in destination operand; cleared otherwise
8	TF	Single Step Flag—Once set, a single step interrupt occurs after the next instruction executes. TF is cleared by the single step interrupt.
9	IF	Interrupt-enable Flag—When set, maskable interrupts will cause the CPU to transfer control to an interrupt vector specified location.
10	DF	Direction Flag—Causes string instructions to auto decrement the appropriate index registers when set. Clearing DF causes auto increment.

Two-operand instructions (e.g. MOV and ADD) are usually three to six bytes long. Memory to memory operations are provided by a special class of string instructions requiring one to three bytes. For detailed instruction formats and encodings refer to the instruction set summary at the end of this document.

Figure 4a. Data Transfer Instructions

GENERAL PURPOSE	
MOV	Move byte or word
PUSH	Push word onto stack
POP	Pop word off stack
PUSHA	Push all registers on stack
POPA	Pop all registers from stack
XCHG	Exchange byte or word
XLAT	Translate byte
INPUT/OUTPUT	
IN	Input byte or word
OUT	Output byte or word
ADDRESS OBJECT	
LEA	Load effective address
LDS	Load pointer using DS
LES	Load pointer using ES
FLAG TRANSFER	
LAHF	Load AH register from flags
SAHF	Store AH register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack

Figure 4b. Arithmetic Instructions

ADDITION	
ADD	Add byte or word
ADC	Add byte or word with carry
INC	Increment byte or word by 1
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition
SUBTRACTION	
SUB	Subtract byte or word
SBB	Subtract byte or word with borrow
DEC	Decrement byte or word by 1
NEG	Negate byte or word
CMP	Compare byte or word
AAS	ASCII adjust for subtraction
DAS	Decimal adjust for subtraction
MULTIPLICATION	
MUL	Multiply byte or word unsigned
IMUL	Integer multiply byte or word
AAM	ASCII adjust for multiply
DIVISION	
DIV	Divide byte or word unsigned
IDIV	Integer divide byte or word
AAD	ASCII adjust for division
CBW	Convert byte to word
CWD	Convert word to doubleword

Figure 4c. String Instructions

MOVS	Move byte or word string
INS	Input bytes or word string
OUTS	Output bytes or word string
CMPS	Compare byte or word string
SCAS	Scan byte or word string
LODS	Load byte or word string
STOS	Store byte or word string
REP	Repeat
REPE/REPZ	Repeat while equal/zero
REPNE/REPNZ	Repeat while not equal/not zero

Figure 4d. Shift/Rotate/Logical Instructions

LOGICALS	
NOT	"Not" byte or word
AND	"And" byte or word
OR	"Inclusive or" byte or word
XOR	"Exclusive or" byte or word
TEST	"Test" byte or word
SHIFTS	
SHL/SAL	Shift logical/arithmetic left byte or word
SHR	Shift logical right byte or word
SAR	Shift arithmetic right byte or word
ROTATES	
ROL	Rotate left byte or word
ROR	Rotate right byte or word
RCL	Rotate through carry left byte or word
RCR	Rotate through carry right byte or word

All mnemonics copyright Intel Corp.

Figure 4e. Program Transfer Instructions

CONDITIONAL TRANSFERS		UNCONDITIONAL TRANSFERS	
JA/JNBE	Jump if above/not below nor equal	CALL	Call procedure
JAE/JNB	Jump if above or equal/not below	RET	Return from procedure
JB/JNAE	Jump if below/not above nor equal	JMP	Jump
JBE/JNA	Jump if below or equal/not above		
JC	Jump if carry	**ITERATION CONTROLS**	
JE/JZ	Jump if equal/zero		
JG/JNLE	Jump if greater/not less nor equal	LOOP	Loop
JGE/JNL	Jump if greater or equal/not less	LOOPE/LOOPZ	Loop if equal/zero
JL/JNGE	Jump if less/not greater nor equal	LOOPNE/LOOPNZ	Loop if not equal/not zero
JLE/JNG	Jump if less or equal/not greater	JCXZ	Jump if register CX = 0
JNC	Jump if not carry	**INTERRUPTS**	
JNE/JNZ	Jump if not equal/not zero		
JNO	Jump if not overflow	INT	Interrupt
JNP/JPO	Jump if not parity/parity odd	INTO	Interrupt if overflow
JNS	Jump if not sign	IRET	Interrupt return
JO	Jump if overflow		
JP/JPE	Jump if parity/parity even		
JS	Jump if sign		

Figure 4f. Processor Control Instructions

FLAG OPERATIONS	
STC	Set carry flag
CLC	Clear carry flag
CMC	Complement carry flag
STD	Set direction flag
CLD	Clear direction flag
STI	Set interrupt enable flag
CLI	Clear interrupt enable flag
EXTERNAL SYNCHRONIZATION	
HLT	Halt until interrupt or reset
WAIT	Wait for $\overline{\text{BUSY}}$ not active
ESC	Escape to extension processor
LOCK	Lock bus during next instruction
NO OPERATION	
NOP	No operation
EXECUTION ENVIRONMENT CONTROL	
LMSW	Load machine status word
SMSW	Store machine status word

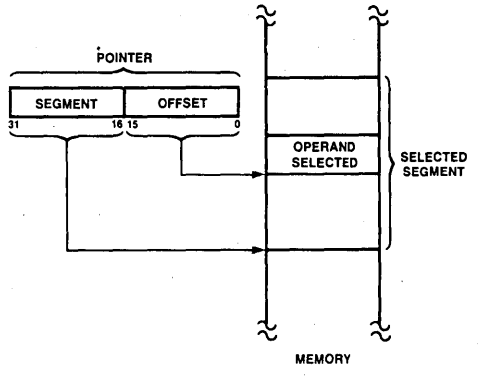
Figure 4g. High Level Instructions

ENTER	Format stack for procedure entry
LEAVE	Restore stack for procedure exit
BOUND	Detects values outside prescribed range

Memory Organization

Memory is organized as sets of variable length segments. Each segment is a linear contiguous sequence of up to 64K (2^{16}) 8-bit bytes. Memory is addressed using a two-component address (a pointer) that consists of a 16-bit segment selector, and a 16-bit offset. The segment selector indicates the desired segment in memory. The offset component indicates the desired byte address within the segment.

Figure 5. Two Component Address



All mnemonics copyright Intel Corp.

Table 3. Segment Register Selection Rules

Memory Reference Needed	Segment Register Used	Implicit Segment Selection Rule
Instructions	Code (CS)	Automatic with instruction prefetch
Stack	Stack (SS)	All stack pushes and pops. Any memory reference which uses BP as a base register.
Local Data	Data (DS)	All data references except when relative to stack or string destination
External (Global) Data	Extra (ES)	Alternate data segment and destination of string operation

All instructions that address operands in memory must specify the segment and the offset. For speed and compact instruction encoding, segment selectors are usually stored in the high speed segment registers. An instruction need specify only the desired segment register and an offset in order to address a memory operand.

Most instructions need not explicitly specify which segment register is used. The correct segment register is automatically chosen according to the rules of Table 3. These rules follow the way programs are written (see Figure 6) as independent modules that require areas for code and data, a stack, and access to external data areas.

Special segment override instruction prefixes allow the implicit segment register selection rules to be overridden for special cases. The stack, data, and extra segments may coincide for simple programs. To access operands that do not reside in one of the four immediately available segments, either a full 32-bit pointer can be used or a new segment selector must be loaded.

Addressing Modes

The 80286 provides a total of eight addressing modes for instructions to specify operands. Two addressing modes are provided for instructions that operate on register or immediate operands:

Register Operand Mode: The operand is located in one of the 8 or 16-bit general registers.

Immediate Operand Mode. The operand is included in the instruction.

Six modes are provided to specify the location of an operand in a memory segment. A memory operand address consists of two 16-bit components: segment selector and offset. The segment selector is supplied by a segment register either implicitly chosen by the addressing mode or explicitly chosen by a segment override prefix. The offset is calculated by summing any combination of the following three address elements:

the **displacement** (an 8 or 16-bit immediate value contained in the instruction)

the **base** (contents of either the BX or BP base registers)

the **index** (contents of either the SI or DI index registers)

Any carry out from the 16-bit addition is ignored. Eight-bit displacements are sign extended to 16-bit values.

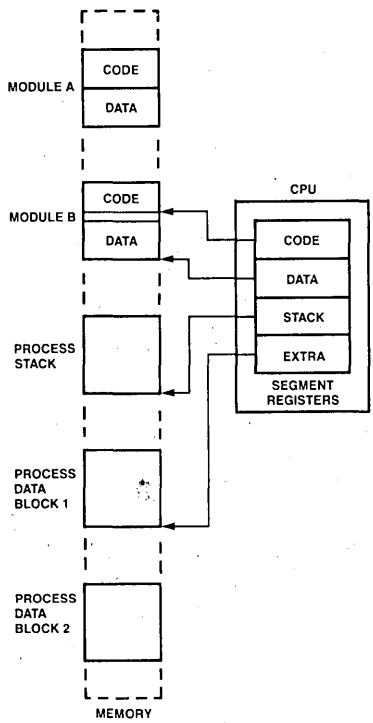
Combinations of these three address elements define the six memory addressing modes, described below.

Direct Mode: The operand's offset is contained in the instruction as an 8 or 16-bit displacement element.

Register Indirect Mode: The operand's offset is in one of the registers SI, DI, BX, or BP.

Based Mode: The operand's offset is the sum of an 8 or 16-bit displacement and the contents of a base register (BX or BP).

Figure 6. Segmented Memory Helps Structure Software



Indexed Mode: The operand's offset is the sum of an 8 or 16-bit displacement and the contents of an index register (SI or DI).

Based Indexed Mode: The operand's offset is the sum of the contents of a base register and an index register.

Based Indexed Mode with Displacement: The operand's offset is the sum of a base register's contents, an index register's contents, and an 8 or 16-bit displacement.

Data Types

The 80286 directly supports the following data types:

Integer: A signed binary numeric value contained in an 8-bit byte or a 16-bit word. All operations assume a 2's complement representation. Signed 32 and 64-bit integers are supported using the iAPX 287 Numeric Data Processor.

Ordinal: An unsigned binary numeric value contained in an 8-bit byte or 16-bit word.

Pointer: A 32-bit quantity, composed of a segment selector component and an offset component. Each component is a 16-bit word.

String: A contiguous sequence of bytes or words. A string may contain from 1 byte to 64K bytes.

ASCII: A byte representation of alphanumeric and control characters using the ASCII standard of character representation.

BCD: A byte (unpacked) representation of the decimal digits 0–9.

Packed BCD: A byte (packed) representation of two decimal digits 0–9 storing one digit in each nibble of the byte.

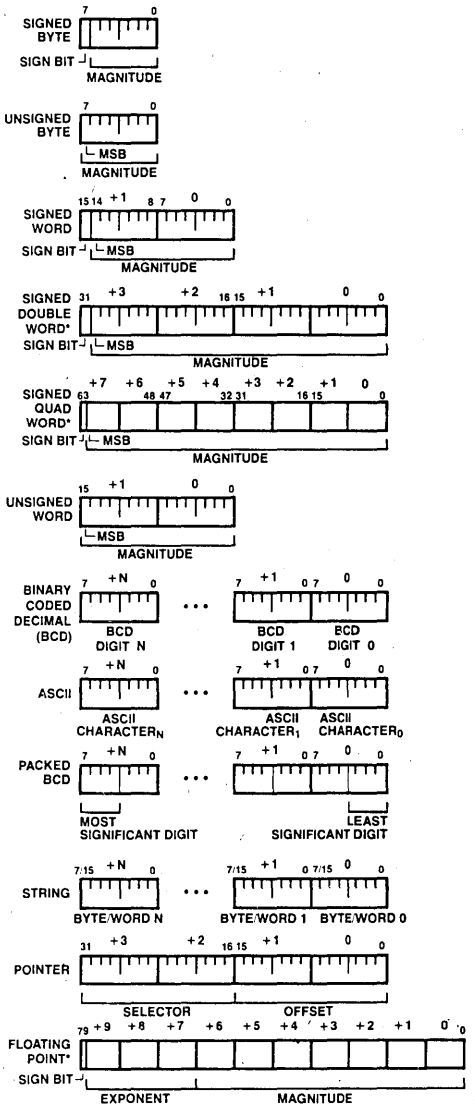
Floating Point: A signed 32, 64, or 80-bit real number representation. (Floating point operands are supported using the iAPX 287 Numeric Processor configuration.)

Figure 7 graphically represents the data types supported by the iAPX 286.

I/O Space

The I/O space consists of 64K 8-bit or 32K 16-bit ports. I/O instructions address the I/O space with either an 8-bit port address, specified in the instruction, or a 16-bit port address in the DX register. 8-bit port addresses are zero extended such that A_{15}–A_8 are LOW. I/O port addresses 00F8(H) through 00FF(H) are reserved.

Figure 7. iAPX 286 Supported Data Types



*Supported by iAPX 286/L287 Numeric Data Processor Configuration

Table 4. Interrupt Vector Assignments

Function	Interrupt Number	Related Instructions	Return Address Before Instruction Causing Exception?
Divide error exception	0	DIV, IDIV	Yes
Single step interrupt	1	All	
NMI interrupt	2	All	
Breakpoint interrupt	3	INT	
INTO detected overflow exception	4	INTO	No
BOUND range exceeded exception	5	BOUND	Yes
Invalid opcode exception	6	Any undefined opcode	Yes
Processor extension not available exception	7	ESC or WAIT	Yes
Reserved	8–15		
Processor extension error interrupt	16	ESC or WAIT	
Reserved	17–31		
User defined	32–255		

Interrupts

An interrupt transfers execution to a new program location. The old program address (CS:IP) and machine state (Flags) are saved on the stack to allow resumption of the interrupted program. Interrupts fall into three classes: hardware initiated, INT instructions, and instruction exceptions. Hardware initiated interrupts occur in response to an external input and are classified as non-maskable or maskable. Programs may cause an interrupt with an INT instruction. Instruction exceptions occur when an unusual condition, which prevents further instruction processing, is detected while attempting to execute an instruction. The return address from an exception will always point at the instruction causing the exception and include any leading instruction prefixes.

A table containing up to 256 pointers defines the proper interrupt service routine for each interrupt. Interrupts 0–31, some of which are used for instruction exceptions, are reserved. For each interrupt, an 8-bit vector must be supplied to the 80286 which identifies the appropriate table entry. Exceptions supply the interrupt vector internally. INT instructions contain or imply the vector and allow access to all 256 interrupts. Maskable hardware initiated interrupts supply the 8-bit vector to the CPU during an interrupt acknowledge bus sequence. Non-maskable hardware interrupts use a predefined internally supplied vector.

MASKABLE INTERRUPT (INTR)

The 80286 provides a maskable hardware interrupt request pin, INTR. Software enables this input by setting the interrupt flag bit (IF) in the flag word. All 224 user-defined interrupt sources can share this input, yet they can retain separate interrupt handlers. An 8-bit vector read by the CPU during the interrupt acknowledge sequence (discussed in System Interface section) identifies the source of the interrupt.

Further maskable interrupts are disabled while servicing an interrupt by resetting the IF but as part of the response to an interrupt or exception. The saved flag word will reflect the enable status of the processor prior to the interrupt. Until the flag word is restored to the flag register, the interrupt flag will be zero unless specifically set. The interrupt return instruction includes restoring the flag word, thereby restoring the original status of IF.

NON-MASKABLE INTERRUPT REQUEST (NMI)

A non-maskable interrupt input (NMI) is also provided. NMI has higher priority than INTR. A typical use of NMI would be to activate a power failure routine. The activation of this input causes an interrupt with an internally supplied vector value of 2. No external interrupt acknowledge sequence is performed.

While executing the NMI servicing procedure, the 80286 will service neither further NMI requests, INTR requests, nor the processor extension segment overrun interrupt until an interrupt return (IRET) instruction is executed or the CPU is reset. If NMI occurs while currently servicing an NMI, its presence will be saved for servicing after executing the first IRET instruction. IF is cleared at the beginning of an NMI interrupt to inhibit INTR interrupts.

SINGLE STEP INTERRUPT

The 80286 has an internal interrupt that allows programs to execute one instruction at a time. It is called the single step interrupt and is controlled by the single step flag bit (TF) in the flag word. Once this bit is set, an internal single step interrupt will occur after the next instruction has been executed. The interrupt clears the TF bit and uses an internally supplied vector of 1. The IRET instruction is used to set the TF bit and transfer control to the next instruction to be single stepped.

Interrupt Priorities

When simultaneous interrupt requests occur, they are processed in a fixed order as shown in Table 5. Interrupt processing involves saving the flags, return address, and setting CS:IP to point at the first instruction of the interrupt handler. If other interrupts remain enabled they are processed before the first instruction of the current interrupt handler is executed. The last interrupt processed is therefore the first one serviced.

Table 5. Interrupt Processing Order

Order	Interrupt
1	INT instruction or exception
2	Single step
3	NMI
4	Processor extension segment overrun
5	INTR

Initialization and Processor Reset

Processor initialization or start up is accomplished by driving the RESET input pin HIGH. RESET forces the 80286 to terminate all execution and local bus activity. No instruction or bus activity will occur as long as RESET is active. After RESET becomes inactive and an internal processing interval elapses, the 80286 begins execution in real address mode with the instruction at physical location FFFFF0(H). RESET also sets some registers to predefined values as shown as shown in Table 6.

Table 6. 80286 Initial Register State after RESET

Flag word	0002(H)
Machine Status Word	FFF0(H)
Instruction pointer	FFF0(H)
Code segment	F000(H)
Data segment	0000(H)
Extra segment	0000(H)
Stack segment	0000(H)

Machine Status Word Description

The machine status word (MSW) records when a task switch takes place and controls the operating mode of the 80286. It is a 16-bit register of which the lower four bits are used. One bit places the CPU into protected mode, while the other three bits, as shown in Table 7, control the processor extension interface. After RESET, this register contains FFF0(H) which places the 80286 in iAPX 86 real address mode.

Table 7. MSW Bit Functions

Bit Position	Name	Function
0	PE	Protected mode Enable places the 80286 into protected mode and can not be cleared except by RESET.
1	MP	Monitor Processor extension allows WAIT instructions to cause a processor extension not present exception (number 7).
2	EM	Emulate processor extension causes a processor extension not present exception (number 7) on ESC instructions to allow emulating a processor extension.
3	TS	Task Switched indicates the next instruction using a processor extension will cause exception 7, allowing software to test whether the current processor extension context belongs to the current task.

The LMSW and SMSW instructions can load and store the MSW in real address mode. The recommended use of TS, EM, and MP is shown in Table 8.

Table 8. Recommended MSW Encodings For Processor Extension Control

TS	MP	EM	Recommended Use	Instructions Causing Exception
0	0	0	iAPX 86 real address mode only. Initial encoding after RESET. iAPX 286 operation is identical to iAPX 86, 88.	None
0	0	1	No processor extension is available. Software will emulate its function.	ESC
1	0	1	No processor extension is available. Software will emulate its function. The current processor extension context may belong to another task.	ESC
0	1	0	A processor extension exists.	None
1	1	0	A processor extension exists. The current processor extension context may belong to another task. The exception on WAIT allows software to test for an error pending from a previous processor extension operation.	ESC or WAIT

Halt

The HLT instruction stops program execution and prevents the CPU from using the local bus until restarted. Either NMI, INTR with IF = 1, or RESET will force the 80286 out of halt. If interrupted, the saved CS:IP will point to the next instruction after the HLT.

iAPX 86 REAL ADDRESS MODE

The 80286 executes a fully upward-compatible superset of the 8086 instruction set in real address mode. In real address mode the 80286 is object code compatible with 8086 and 8088 software. The real address mode architecture (registers and addressing modes) is exactly as described in the iAPX 286 Base Architecture section of this Functional Description.

Memory Size

Physical memory is a contiguous array of up to 1,048,576 bytes (one megabyte) addressed by pins A_0 through A_{19} and $\overline{BHE}$. A_{20} through A_{23} are ignored.

Memory Addressing

In real address mode the processor generates 20-bit physical addresses directly from a 20-bit segment base address and a 16-bit offset.

The selector portion of a pointer is interpreted as the upper 16 bits of a 20-bit segment address. The lower four bits of the 20-bit segment address are always zero. Segment addresses, therefore, begin on multiples of 16 bytes. See Figure 8 for a graphic representation of address formation.

All segments in real address mode are 64K bytes in size and may be read, written, or executed. An exception or interrupt can occur if data operands or instructions attempt to wrap around the end of a segment (e.g. a word with its low order byte at offset FFFF(H) and its high order byte at offset 0000(H)). If, in real address mode, the information contained in a segment does not use the full 64K bytes, the unused end of the segment may be overlayed by another segment to reduce physical memory requirements.

Reserved Memory Locations

The 80286 reserves two fixed areas of memory in real address mode (see Figure 9); system initialization area and interrupt table area. Locations from addresses FFFF0(H) thorugh FFFFF(H) are reserved for system initialization. Initial execution begins at location FFFF0(H). Locations 00000(H) through 003FF(H) are reserved for interrupt vectors.

Figure 8. iAPX 86 Real Address Mode Address Calculation

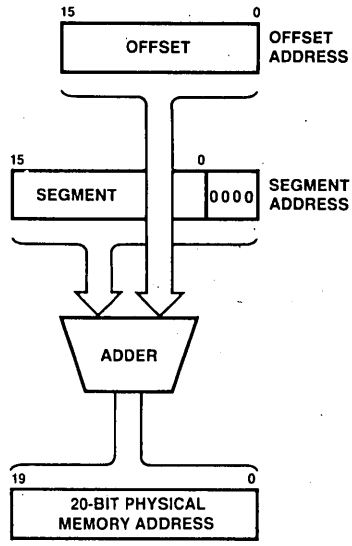


Figure 9. iAPX 86 Real Address Mode Initially Reserved Memory Locations

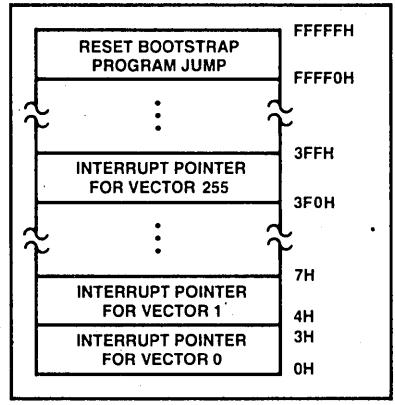


Table 9. Real Address Mode Addressing Interrupts

Function	Interrupt Number	Related Instructions	Return Address Before Instruction?
Interrupt table limit too small exception	8	INT vector is not within table limit	Yes
Processor extension segment overrun interrupt	9	ESC with memory operand extending beyond offset FFFF(H)	No
Segment overrun exception	13	Word memory reference with offset = FFFF(H) or an attempt to execute past the end of a segment	Yes

Interrupts

Table 9 shows the interrupt vectors reserved for exceptions and interrupts which indicate an addressing error. The exceptions leave the CPU in the state existing before attempting to execute the failing instruction (except for PUSH, POP, PUSHA, or POPA). Refer to the next section on protected mode initialization for a discussion on exception 8.

Protected Mode Initialization

To prepare the 80286 for protected mode, the LIDT instruction is used to load the 24-bit interrupt table base and 16-bit limit for the protected mode interrupt table. This instruction can also set a base and limit for the interrupt vector table in real address mode. After reset, the interrupt table base is initialized to 000000(H) and its size set to 03FF(H). These values are compatible with iAPX 86, 88 software. LIDT should only be executed in preparation for protected mode.

Shutdown

Shutdown occurs when a severe error is detected that prevents further instruction processing by the CPU. Shutdown and halt are externally signalled via a halt bus operation. They can be distinguished by A_1 HIGH for halt and A_1 LOW for shutdown. In real address mode, shutdown can occur under two conditions:

- Exceptions 8 or 13 happen and the IDT limit does not include the interrupt vector.
- A CALL, INT, or POP instruction attempts to wrap around the stack segment when SP is not even.

An NMI input can bring the CPU out of shutdown if the IDT limit is at least 000F(H) and SP is greater than 0005(H), otherwise shutdown can only be exited via the RESET input.

PROTECTED VIRTUAL ADDRESS MODE

The 80286 executes a fully upward-compatible superset of the 8086 instruction set in protected virtual address mode (protected mode). Protected mode also provides memory management and protection mechanisms and associated instructions.

The 80286 enters protected virtual address mode from real address mode by setting the PE (Protection Enable) bit of the machine status word with the Load Machine Status Word (LMSW) instruction. Protected mode offers extended physical and virtual memory address space, memory protection mechanisms, and new operations to support operating systems and virtual memory.

All registers, instructions, and addressing modes described in the iAPX 286 Base Architecture section of this Functional Description remain the same. Programs for the iAPX 86, 88, 186, and real address mode 80286 can be run in protected mode; however, embedded constants for segment selectors are different.

Memory Size

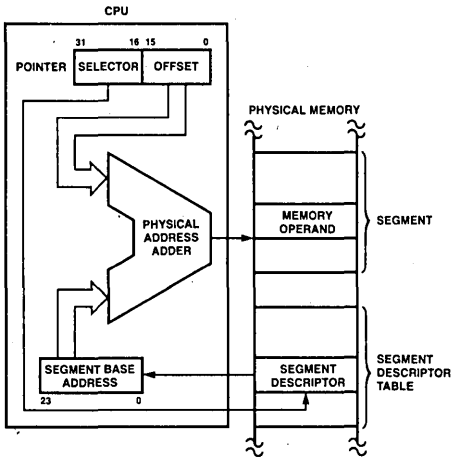
The protected mode 80286 provides a 1 gigabyte virtual address space per task mapped into a 16 megabyte physical address space defined by the address pins A_{23}–A_0 and $\overline{BHE}$. The virtual address space may be larger than the physical address space since any use of an address that does not map to a physical memory location will cause a restartable exception.

Memory Addressing

As in real address mode, protected mode uses 32-bit pointers, consisting of 16-bit selector and offset components. The selector, however, specifies an index into a memory resident table rather than the upper 16-bits of a real memory address. The 24-bit base address of the

desired segment is obtained from the tables in memory. The 16-bit offset is added to the segment base address to form the physical address as shown in Figure 10. The tables are automatically referenced by the CPU whenever a segment register is loaded with a selector. All iAPX 286 instructions which load a segment register will reference the memory based tables without additional software. The memory based tables contain 8 byte values called descriptors.

Figure 10. Protected Mode Memory Addressing

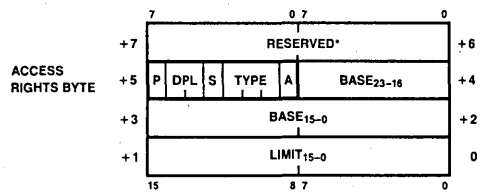


DESCRIPTORS

Descriptors define the use of memory. Special types of descriptors also define new functions for transfer of control and task switching. The 80286 has segment descriptors for code, stack and data segments, and system control descriptors for special system data segments and control transfer operations. Descriptor accesses are performed as locked bus operations to assure descriptor integrity in multi-processor systems.

CODE AND DATA SEGMENT DESCRIPTORS

Besides segment base addresses, code and data descriptors contain other segment attributes including segment size (1 to 64K bytes), access rights (read only, read/write, execute only, and execute/read), and presence in memory (for virtual memory systems) (See Figure 11). Any segment usage violating a segment attribute indicated by the segment descriptor will prevent the memory cycle and cause an exception or interrupt.



*Must be set to 0 for compatability with iAPX 386.

Figure 11. Code and Data Segment Descriptors

Access Rights Byte Definition

Bit Position	Name		Function	
7	Present (P)	P = 1	Segment is mapped into physical memory.	
		P = 0	No mapping to physical memory exists, base and limit are not used.	
6–5	Descriptor Privilege Level (DPL)		Segment privilege attribute used in privilege tests.	
4	Segment Descriptor (S)	S = 1	Code or Data segment descriptor	
		S = 0	Non-segment descriptor	
3	Executable (E)	E = 0	Data segment descriptor type is:	Type Field Definition — Data Segment
2	Expansion Direction (ED)	ED = 0	Grow up segment, offsets must be ≤ limit.	
		ED = 1	Grow down segment, offsets must be > limit.	
1	Writeable (W)	W = 0	Data segment may not be written into.	
		W = 1	Data segment may be written into.	
3	Executable (E)	E = 1	Code Segment Descriptor type is:	Type Field Definition — Code Segment
2	Conforming (C)	C = 1	Code segment may only be executed when CPL ≥ DPL.	
1	Readable (R)	R = 0	Code segment may not be read.	
		R = 1	Code segment may be read.	
0	Accessed (A)	A = 0	Segment has not been accessed.	
		A = 1	Segment selector has been loaded into segment register or used by selector test instructions.	

Code and data are stored in two types of segments: code segments and data segments. Both types are identified and defined by segment descriptors. Code segments are identified by the executable (E) bit set to 1 in the descriptor access rights byte. The access rights byte of both code and data segment descriptor types have three fields in common: present (P) bit, Descriptor Privilege Level (DPL), and accessed (A) bit. If P = 0, any attempted use of this segment will cause a not-present exception. DPL specifies the privilege level of the segment descriptor. DPL effects when the descriptor may be used by a task (refer to privilege discussion below). The A bit shows whether the segment has been previously accessed for usage profiling, a necessity for virtual memory systems. The CPU will always set this bit when accessing the descriptor.

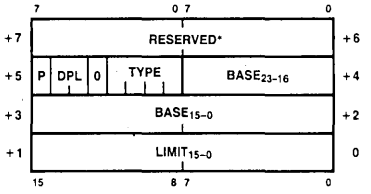
Data segments (S = 1, E = 0) may be either read-only or read-write as controlled by the W bit of the access rights byte. Read-only (W = 0) data segments may not be written into. Data segments may grow in two directions, as determined by the Expansion Direction (ED) bit: upwards (ED = 0) for data segments, and downwards (ED = 1) for a segment containing a stack. The limit field for a data segment descriptor is interpreted differently depending on the ED bit (see Figure 11).

A code segment (S = 1, E = 1) may be execute-only or execute/read as determined by the Readable (R) bit. Code segments may never be written into and execute-only code segments (R = 0) may not be read. A code segment may also have an attribute called conforming (C). A conforming code segment may be shared by programs that execute at different privilege levels. The DPL of a conforming code segment defines the range of privilege levels at which the segment may be executed (refer to privilege discussion below).

SYSTEM CONTROL DESCRIPTORS

In addition to code and data segment descriptors, the protected mode 80286 defines system control descriptors. These descriptors define special system data segments and control transfer mechanisms in the protected environment. The special system data segment descriptors define segments which contain tables of descriptors (Local Descriptor Table Descriptor) and segments which contain the execution state of a task (Task State Segment Descriptor).

The control transfer descriptors are call gates, task gates, interrupt gates and trap gates. Gates provide a level of indirection between the source and destination of the control transfer. This indirection allows the CPU to automatically perform protection checks and control the entry point of the destination. Call gates are used to change privilege levels (see Privilege), task gates are used to perform a task switch, and interrupt and trap



*Must be set to 0 for compatability with iAPX 386.

Figure 12. System Segment Format

System Segment Descriptor Fields

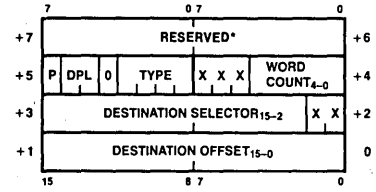
Name	Value	Description
TYPE	1 2 3	Available Task State Segment Local Descriptor Table Descriptor Busy Task State Segment
P	0 1	Descriptor contents are not valid Descriptor contents are valid
DPL	0–3	Descriptor Privilege Level
BASE	24-bit number	Base Address of special system data segment in real memory
LIMIT	16-bit number	Offset of last byte in segment

gates are used to specify interrupt service routines. The interrupt gate disables interrupts (resets IF) while the trap gate does not.

Figure 12 gives the formats for the special system data segment descriptors. The descriptors contain a 24-bit base address of the segment and a 16-bit limit. The access byte defines the type of descriptor, its state and privilege level. The descriptor contents are valid and the segment is in physical memory if P = 1. If P = 0, the segment is not valid. The DPL field is only used in Task State Segment descriptors and indicates the privilege level at which the descriptor may be used (see Privilege). Since the Local Descriptor Table descriptor may only be used by a special privileged instruction, the DPL field is not used. Bit 4 of the access byte is 0 to indicate that it is a system control descriptor. The type field specifies the descriptor type as indicated in Figure 12.

Figure 13 shows the format of the gate descriptors. The descriptor contains a destination pointer that points to the descriptor of the target segment and the entry point offset. The destination selector in an interrupt gate, trap gate, and call gate must refer to a code segment descriptor. These gate descriptors contain the entry point to prevent a program from constructing and using an illegal entry point. Task gates may only refer to a task state segment. Since task gates invoke a task switch, the destination offset is not used in the task gate.

Exception 13 is generated when the gate is used if a destination selector does not refer to the correct de-



*Must be set to 0 for compatability with iAPX 386.

Figure 13. Gate Descriptor Format

Gate Descriptor Fields

Name	Value	Description
TYPE	4 5 6 7	–Call Gate –Task Gate –Interrupt Gate –Trap Gate
P	0 1	–Descriptor Contents are not valid –Descriptor Contents are valid
DPL	0–3	Descriptor Privilege Level
WORD COUNT	0–31	Number of words to copy from callers stack to called procedures stack. Only used with call gate.
DESTINATION SELECTOR	16-bit selector	Selector to the target code segment (Call, Interrupt or Trap Gate) Selector to the target task state segment (Task Gate)
DESTINATION OFFSET	16-bit offset	Entry point within the target code segment

scriptor type. The word count field is used in the call gate descriptor to indicate the number of parameters (0–31 words) to be automatically copied from the caller's stack to the stack of the called routine when a control transfer changes privilege levels. The word count field is not used by any other gate descriptor.

The access byte format is the same for all gate descriptors. P = 1 indicates that the gate contents are valid. P = 0 indicates the contents are not valid and causes exception 11 if referenced. DPL is the descriptor privilege level and specifies when this descriptor may be used by a task (refer to privilege discussion below). Bit 4 must equal 0 to indicate a system control descriptor. The type field specifies the descriptor type as indicated in Figure 13.

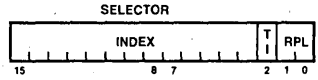
SEGMENT DESCRIPTOR CACHE REGISTERS

A segment descriptor cache register is assigned to each of the four segment registers (CS, SS, DS, ES). Segment descriptors are automatically loaded (cached) into a segment descriptor cache register (Figure 14) whenever the associated segment register is loaded with a selector. Only segment descriptors may be loaded into segment descriptor cache registers. Once loaded, all references to that segment of memory use the cached descriptor information instead of reaccessing memory. The descriptor cache registers are not visible to programs. No instructions exist to store their contents. They only change when a segment register is loaded.

SELECTOR FIELDS

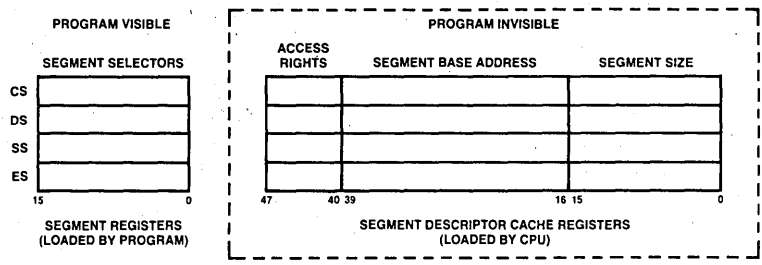
A protected mode selector has three fields: descriptor entry index, local or global descriptor table indicator (TI), and selector privilege (RPL) as shown in Figure 15. These fields select one of two memory based tables of descriptors, select the appropriate table entry and allow high-speed testing of the selector's privilege attribute (refer to privilege discussion below).

Figure 15. Selector Fields



BITS	NAME	FUNCTION
1–0	REQUESTED PRIVILEGE LEVEL (RPL)	INDICATES SELECTOR PRIVILEGE LEVEL DESIRED
2	TABLE INDICATOR (TI)	TI = 0 USE GLOBAL DESCRIPTOR TABLE (GDT) TI = 1 USE LOCAL DESCRIPTOR TABLE (LDT)
15–3	INDEX	SELECT DESCRIPTOR ENTRY IN TABLE

Figure 14. Descriptor Cache Registers

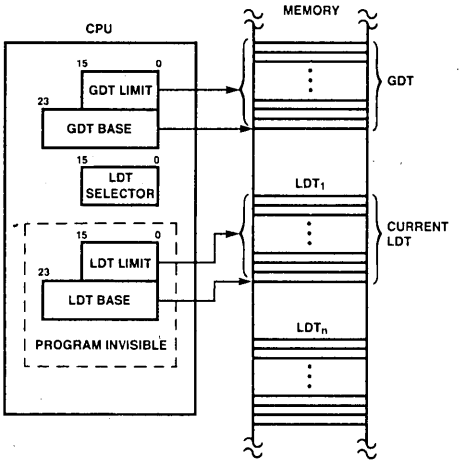


LOCAL AND GLOBAL DESCRIPTOR TABLES

Two tables of descriptors, called descriptor tables, contain all descriptors accessible by a task at any given time. A descriptor table is a linear array of up to 8192 descriptors. The upper 13 bits of the selector value are an index into a descriptor table. Each table has a 24-bit base register to locate the descriptor table in physical memory and a 16-bit limit register that confine descriptor access to the defined limits of the table as shown in Figure 16. A restartable exception (13) will occur if an attempt is made to reference a descriptor outside the table limits.

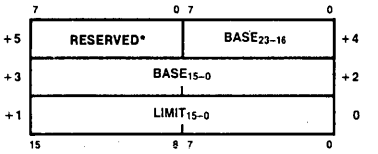
One table, called the Global Descriptor Table (GDT), contains descriptors available to all tasks. The other table, called the Local Descriptor Table (LDT), contains descriptors that can be private to a task. Each task may have its own private LDT. The GDT may contain all descriptor types except interrupt and trap descriptors. The LDT may contain only segment, task gate, and call gate descriptors. A segment cannot be accessed by a task if its segment descriptor does not exist in either descriptor table at the time of access.

Figure 16. Local and Global Descriptor Table Definition



The LGDT and LLDT instructions load the base and limit of the global and local descriptor tables. LGDT and LLDT are protected. They may only be executed by trusted programs operating at level 0. The LGDT instruction loads a six byte field containing the 16-bit table limit and 24-bit base address of the Global Descriptor Table as shown in Figure 17. The LLDT instruction loads a selector which refers to a Local Descriptor Table descriptor containing the base address and limit for an LDT, as shown in Figure 12.

Figure 17. Global Descriptor Table and Interrupt Descriptor Data Type

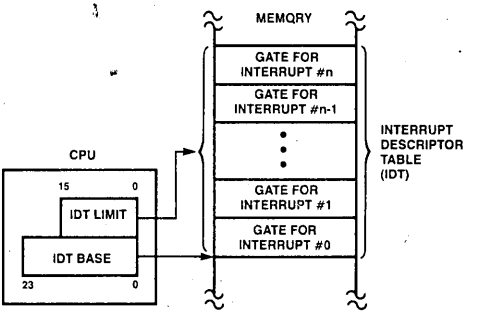


*Must be set to 0 for compatability with iAPX 386.

INTERRUPT DESCRIPTOR TABLE

The protected mode 80286 has a third descriptor table, called the Interrupt Descriptor Table (IDT) (see Figure 18), used to define up to 256 interrupts. It may contain only task gates, interrupt gates and trap gates. The IDT (Interrupt Descriptor Table) has a 24-bit base and 16-bit limit register in the CPU. The protected LIDT instruction loads these registers with a six byte value of identical form to that of the LGDT instruction (see Figure 17 and Protected Mode Initialization).

Figure 18. Interrupt Descriptor Table Definition

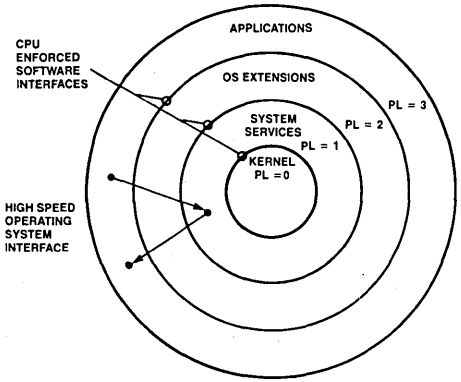


References to IDT entries are made via INT instructions, external interrupt vectors, or exceptions. The IDT must be at least 256 bytes in size to allocate space for all reserved interrupts.

Privilege

The 80286 has a four-level hierarchical privilege system which controls the use of privileged instructions and access to descriptors (and their associated segments) within a task. Four-level privilege, as shown in Figure 19, is an extension of the user/supervisor mode commonly found in minicomputers. The privilege levels are numbered 0 through 3. Level 0 is the most privileged level. Privilege

Figure 19. Hierarchical Privilege Levels



levels provide protection within a task. (Tasks are isolated by providing private LDT's for each task.) Operating system routines, interrupt handlers, and other system software can be included and protected within the virtual address space of each task using the four levels of privilege. Tasks may also have a separate stack for each privilege level.

Tasks, descriptors, and selectors have a privilege level attribute that determines whether the descriptor may be used. Task privilege effects the use of instructions and descriptors. Descriptor and selector privilege only effect access to the descriptor.

TASK PRIVILEGE

A task always executes at one of the four privilege levels. The task privilege level at any specific instant is called the Current Privilege Level (CPL) and is defined by the lower two bits of the CS register. CPL cannot change during execution in a single code segment. A task's CPL may only be changed by control transfers through gate descriptors to a new code segment (See Control Transfer). Tasks begin executing at the CPL value specified by the code segment when the task is initiated via a task switch operation. A task executing at Level 0 can access all data segments defined in the GDT and the task's LDT and is considered the most trusted level. A task executing at Level 3 has the most restricted access to data and is considered the least trusted level.

DESCRIPTOR PRIVILEGE

Descriptor privilege is specified by the Descriptor Privilege Level (DPL) field of the descriptor access byte. DPL specifies the least trusted task privilege level (CPL) at which a task may access the descriptor. Descriptors with DPL = 0 are the most protected. Only tasks executing at privilege level 0 (CPL = 0) may access them. Descriptors with DPL = 3 are the least protected (i.e. have the least restricted access) since tasks can access them when CPL = 0, 1, 2, or 3. This rule applies to all descriptors, except LDT descriptors.

SELECTOR PRIVILEGE

Selector privilege is specified by the Requested Privilege Level (RPL) field in the least significant two bits of a selector. Selector RPL may establish a less trusted privilege level than the current privilege level for the use of a selector. This level is called the task's effective privilege level (EPL). RPL can only reduce the scope of a task's access to data with this selector. A task's effective privilege is the numeric maximum of RPL and CPL. A selector with RPL = 0 imposes no additional restriction on its use while a selector with RPL = 3 can only refer to segments at privilege Level 3 regardless of the task's CPL. RPL is generally used to verify that pointer parameters passed to a more trusted procedure are not allowed to use data at a more privileged level than the caller (refer to pointer testing instructions).

Descriptor Access and Privilege Validation

Determining the ability of a task to access a segment involves the type of segment to be accessed, the instruction used, the type of descriptor used and CPL, RPL, and DPL. The two basic types of segment accesses are control transfer (selectors loaded into CS) and data (selectors loaded into DS, ES or SS).

DATA SEGMENT ACCESS

Instructions that load selectors into DS and ES must refer to a data segment descriptor or readable code segment descriptor. The CPL of the task and the RPL of the selector must be the same as or more privileged (numerically equal to or lower than) than the descriptor DPL. In general, a task can only access data segments at the same or less privileged levels than the CPL or RPL (whichever is numerically higher) to prevent a program from accessing data it cannot be trusted to use.

An exception to the rule is a readable conforming code segment. This type of code segment can be read from any privilege level.

If the privilege checks fail (e.g. DPL is numerically less than the maximum of CPL and RPL) or an incorrect type of descriptor is referenced (e.g. gate descriptor or execute only code segment) exception 13 occurs. If the segment is not present, exception 11 is generated.

Instructions that load selectors into SS must refer to data segment descriptors for writable data segments. The descriptor privilege (DPL) and RPL must equal CPL. All other descriptor types or a privilege level violation will cause exception 13. A not present fault causes exception 12.

CONTROL TRANSFER

Four types of control transfer can occur when a selector is loaded into CS by a control transfer operation (see Table 10). Each transfer type can only occur if the operation which loaded the selector references the correct descriptor type. Any violation of these descriptor usage rules (e.g. JMP through a call gate or RET to a Task State Segment) will cause exception 13.

The ability to reference a descriptor for control transfer is also subject to rules of privilege. A CALL or JUMP instruction may only reference a code segment descriptor with DPL equal to the task CPL or a conforming segment with DPL of equal or greater privilege than CPL. The RPL of the selector used to reference the code descriptor must have as much privilege as CPL.

RET and IRET instructions may only reference code segment descriptors with descriptor privilege equal to or less privileged than the task CPL. The selector loaded into CS is the return address from the stack. After the return, the selector RPL is the task's new CPL. If CPL changes, the old stack pointer is popped after the return address.

When a JMP or CALL references a Task State Segment descriptor, the descriptor DPL must be the same or less privileged than the task's CPL. Reference to a valid Task State Segment descriptor causes a task switch (see Task Switch Operation). Reference to a Task State Segment descriptor at a more privileged level than the task's CPL generates exception 13.

When an instruction or interrupt references a gate descriptor, the gate DPL must have the same or less privilege than the task CPL. If DPL is at a more privileged level than CPL, exception 13 occurs. If the destination selector contained in the gate references a code segment descriptor, the code segment descriptor DPL must be the same or more privileged than the task CPL. If not, Exception 13 is issued. After the control transfer, the code segment descriptors DPL is the task's new CPL. If the destination selector in the gate references a task state segment, a task switch is automatically performed (see Task Switch Operation).

The privilege rules on control transfer require:

—JMP or CALL direct to a code segment (code segment descriptor) can only be to a conforming segment with DPL of equal or greater privilege than CPL or a non-conforming segment at the same privilege level.

—interrupts within the task or calls that may change privilege levels, can only transfer control through a gate at the same or a less privileged level than CPL to a code segment at the same or more privileged level than CPL.

—return instructions that don't switch tasks can only return control to a code segment at the same or less privileged level.

—task switch can be performed by a call, jump or interrupt which references either a task gate or task state segment at the same or less privileged level.

Table 10. Descriptor Types Used for Control Transfer

Control Transfer Types	Operation Types	Descriptor Referenced	Descriptor Table
Intersegment within the same privilege level	JMP, CALL, RET, IRET*	Code Segment	GDT/LDT
Intersegment to the same or higher privilege level Interrupt within task may change CPL.	CALL	Call Gate	GDT/LDT
	Interrupt Instruction, Exception, External Interrupt	Trap or Interrupt Gate	IDT
Intersegment to a lower privilege level (changes task CPL)	RET, IRET*	Code Segment	GDT/LDT
Task Switch	CALL, JMP	Task State Segment	GDT
	CALL, JMP	Task Gate	GDT/LDT
	IRET** Interrupt Instruction, Exception, External Interrupt	Task Gate	IDT

*NT (Nested Task bit of flag word) = 0
**NT (Nested Task bit of flag word) = 1

PRIVILEGE LEVEL CHANGES

Any control transfer that changes CPL within the task, causes a change of stacks as part of the operation. Initial values of SS:SP for privilege levels 0, 1, and 2 are kept in the task state segment (refer to Task Switch Operation). During a JMP or CALL control transfer, the new stack pointer is loaded into the SS and SP registers and the previous stack pointer is pushed onto the new stack.

When returning to the original privilege level, its stack is restored as part of the RET or IRET instruction operation. For subroutine calls that pass parameters on the stack and cross privilege levels, a fixed number of words, as specified in the gate, are copied from the previous stack to the current stack. The inter-segment RET instruction with a stack adjustment value will correctly restore the previous stack pointer upon return.

Protection

The 80286 includes mechanisms to protect critical instructions that affect the CPU execution state (e.g. HLT) and code or data segments from improper usage. These mechanisms are grouped under the term "protection" and have three forms:

Restricted usage of segments (e.g. no write allowed to read-only data segments). The only segments available for use are defined by descriptors in the Local Descriptor Table (LDT) and Global Descriptor Table (GDT).

Restricted access to segments via the rules of privilege and descriptor usage.

Privileged instructions or operations that may only be executed at certain privilege levels as determined by the CPL and I/O Privilege Level (IOPL). The IOPL is defined by bits 14 and 13 of the flag word.

These checks are performed for all instructions and can be split into three categories: segment load checks (Table 11), operand reference checks (Table 12), and privileged instruction checks (Table 13). Any violation of the rules shown will result in an exception. A not-present exception related to the stack segment causes exception 12.

The IRET and POPF instructions do not perform some of their defined functions if CPL is not of sufficient privilege (numerically small enough). No exceptions or other indication are given when these conditions occur.

The IF bit is not changed if CPL > IOPL.

The IOPL field of the flag word is not changed if CPL > 0.

Table 11
Segment Register Load Checks

Error Description	Exception Number
Descriptor table limit exceeded	13
Segment descriptor not-present	11 or 12
Privilege rules violated	13
Invalid descriptor/segment type segment register load: —Read only data segment load to SS —Special control descriptor load to DS, ES, SS —Execute only segment load to DS, ES, SS —Data segment load to CS —Read/Execute code segment load to SS	13

Table 12 Operand Reference Checks

Error Description	Exception Number
Write into code segment	13
Read from execute-only code segment	13
Write to read-only data segment	13
Segment limit exceeded[1]	12 or 13

Note 1: Carry out in offset calculations is ignored.

Table 13. Privileged Instruction Checks

Error Description	Exception Number
CPL ≠ 0 when executing the following instructions: LIDT, LLDT, LGDT, LTR, LMSW, CTS, HLT	13
CPL > IOPL when executing the following instructions: INS, IN, OUTS, OUT, STI, CLI, LOCK	13

EXCEPTIONS

The 80286 detects several types of exceptions and interrupts, in protected mode (see Table 14). Most are restartable after the exceptional condition is removed. Interrupt handlers for most exceptions receive an error code, pushed on the stack after the return address, that identifies the selector involved (0 if none). The return address normally points to the failing instruction, including all leading prefixes. For a processor extension segment overrun exception, the return address will not point at the ESC instruction that caused the exception; however, the processor extension registers may contain the address of the failing instruction.

Table 14. Protected Mode Exceptions

Interrupt Vector	Function	Return Address At Failing Instruction?	Always Restart-able?	Error Code on Stack?
8	Double exception detected	Yes	No	Yes
9	Processor extension segment overrun	No	No	No
10	Invalid task state segment	Yes	Yes	Yes
11	Segment not present	Yes	Yes	Yes
12	Stack segment overrun or segment not present	Yes	Yes[1]	Yes
13	General protection	Yes	No	Yes

Note 1: When a PUSHA or POPA instruction attempts to wrap around the stack segment, the machine state after the exception will not be restartable. This condition is identified by the value of the saved SP being either 0000(H), 0001(H), FFFE(H), or FFFF(H).

All these checks are performed for all instructions and can be split into three categories: segment load checks (Table 11), operand reference checks (Table 12), and privileged instruction checks (Table 13). Any violation of the rules shown will result in an exception. A not-present exception related to the stack segment causes exception 12.

Special Operations

TASK SWITCH OPERATION

The 80286 provides a built-in task switch operation which saves the entire 80286 execution state (registers, address space, and a link to the previous task), loads a new execution state, and commences execution in the new task. Like gates, the task switch operation is invoked by executing an inter-segment JMP or CALL instruction which refers to a Task State Segment (TSS) or task gate descriptor in the GDT or LDT. An INT n instruction, exception, or external interrupt may also invoke the task switch operation by selecting a task gate descriptor in the associated IDT descriptor entry.

The TSS descriptor points at a segment (see Figure 20) containing the entire 80286 execution state while a task gate descriptor contains a TSS selector. The limit field must be > 002B(H).

Each task must have a TSS associated with it. The current TSS is identified by a special register in the 80286 called the Task Register (TR). This register contains a selector referring to the task state segment descriptor that defines the current TSS. A hidden base and limit register associated with TR are loaded whenever TR is loaded with a new selector.

The IRET instruction is used to return control to the task that called the current task or was interrupted. Bit 14 in the flag register is called the Nested Task (NT) bit. It controls the function of the IRET instruction. If NT = 0, the IRET instruction performs the regular current task return; when NT = 1, IRET performs a task switch operation back to the previous task.

When a CALL or INT instruction initiates a task switch, the old and new TSS will be marked busy and the back link field of the new TSS set to the old TSS selector. The NT bit of the new task is set by CALL or INT initiated task switches. An interrupt that does not cause a task switch will clear NT. NT may also be set or cleared by POPF or IRET instructions.

The task state segment is marked busy by changing the descriptor type field from Type 1 to Type 3. Use of a selector that references a busy task state segment causes Exception 13.

PROCESSOR EXTENSION CONTEXT SWITCHING

The context of a processor extension (such as the 80287 numerics processor) is not changed by the task switch operation. A processor extension context need only be changed when a different task attempts to use the processor extension (which still contains the context of a previous task). The 80286 detects the first use of a processor extension after a task switch by causing the processor extension not present exception (7). The interrupt handler may then decide whether a context change is necessary.

Whenever the 80286 switches tasks, it sets the Task Switched (TS) bit of the MSW. TS indicates that a processor extension context may belong to a different task than the current one. The processor extension not present exception (7) will occur when attempting to execute an ESC or WAIT instruction if TS = 1 and a processor extension is present (MP = 1 in MSW).

POINTER TESTING INSTRUCTIONS

The iAPX 80286 provides several instructions to speed pointer testing and consistency checks for maintaining system integrity (see Table 15). These instructions use the memory management hardware to verify that a selector value refers to an appropriate segment without risking an exception. A condition flag indicates whether use of the selector or segment will cause an exception.

Figure 20. Task State Segment and TSS Registers

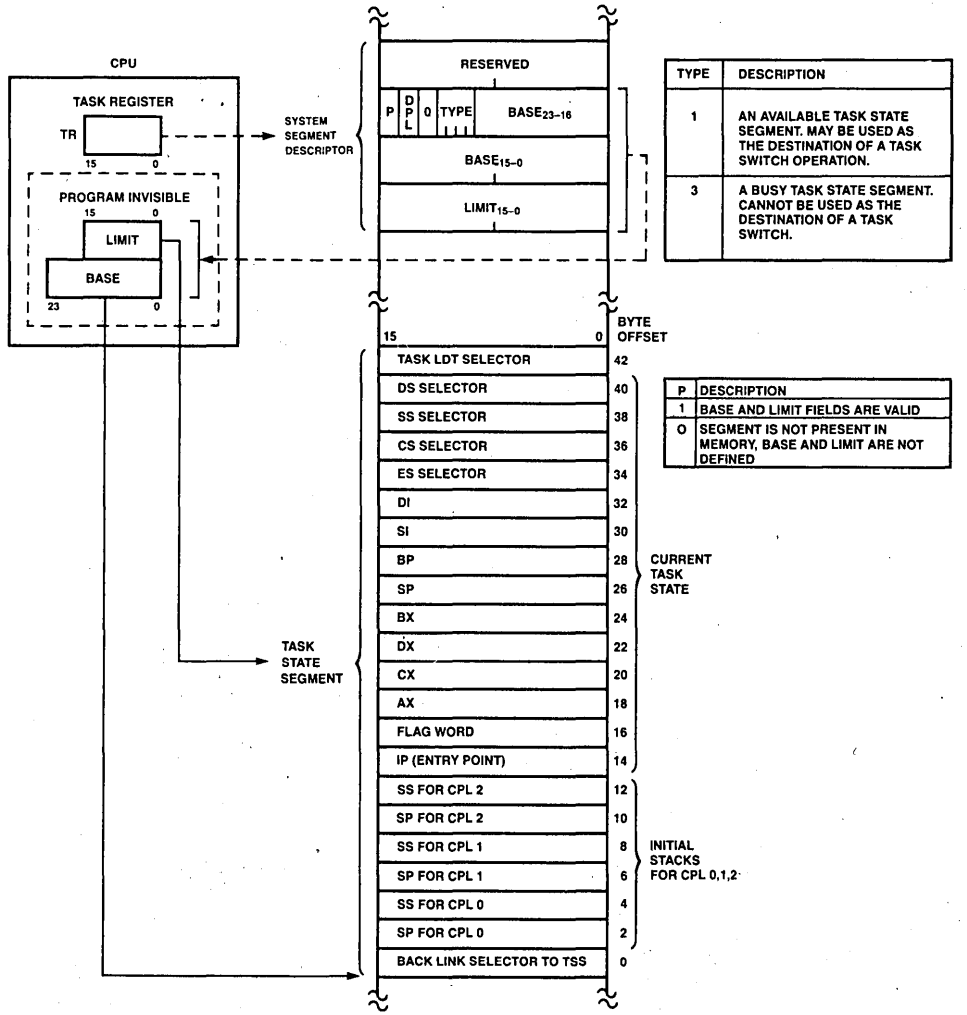


Table 15. Pointer Test Instructions

Instruction	Operands	Function
ARPL	Selector, Register	Adjust Requested Privilege Level: adjusts the RPL of the selector to the numeric maximum of current selector RPL value and the RPL value in the register. Set zero flag if selector RPL was changed.
VERR	Selector	VERify for Read: sets the zero flag if the segment referred to by the selector can be read.
VERW	Selector	VERify for Write: sets the zero flag if the segment referred to by the selector can be written.
LSL	Register, Selector	Load Segment Limit: reads the segment limit into the register if privilege rules and descriptor type allow. Set zero flag if successful.
LAR	Register, Selector	Load Access Rights: reads the descriptor access rights byte into the register if privilege rules allow. Set zero flag if successful.

DOUBLE FAULT AND SHUTDOWN

If two separate exceptions are detected during a single instruction execution, the 80286 performs the double fault exception (8). If an exception occurs during processing of the double fault exception, the 80286 will enter shutdown. During shutdown no further instructions or exceptions are processed. Either NMI (CPU remains in protected mode) or RESET (CPU exits protected mode) can force the 80286 out of shutdown. Shutdown is externally signalled via a HALT bus operation with A_1 HIGH.

PROTECTED MODE INITIALIZATION

The 80286 initially executes in real address mode after RESET. To allow initialization code to be placed at the top of physical memory, A_{23-20} will be HIGH when the 80286 performs memory references relative to the CS register, until CS is changed. A_{23-20} will be zero for references to the DS, ES, or SS segments. Changing CS in real address mode will force A_{23}–A_{20} LOW whenever using CS thereafter. The initial CS:IP value of FF00:FFF0 provides 64K bytes of code space for initialization code without changing CS.

Before placing the 80286 into protected mode, several registers must be initialized. The GDT and IDT base registers must refer to a valid GDT and IDT. After executing the LMSW instruction to set PE, the 80286 must immediately execute an intra-segment JMP instruction to clear the instruction queue of instructions decoded in real address mode.

To force the 80286 CPU registers to match the initial protected mode state assumed by software, execute a JMP instruction with a selector referring to the initial TSS used in the system. This will load the task register, local descriptor table register, segment registers and initial general register state. The TR should point at a valid TSS since a task switch operation involves saving the current task state.

SYSTEM INTERFACE

The 80286 system interface appears in two forms: a local bus and a system bus. The local bus consists of address, data, status, and control signals at the pins of the CPU. A system bus is any buffered version of the local bus. A system bus may also differ from the local bus in terms of coding of status and control lines and/or timing and loading of signals. The iAPX 286 family includes several devices to generate standard system buses such as the IEEE 796 standard Multibus™.

Bus Interface Signals and Timing

The iAPX 286 microsystem local bus interfaces the 80286 to local memory and I/O components. The interface has 24 address lines, 16 data lines, and 8 status and control signals.

The 80286 CPU, 82284 clock generator, 82288 bus controller, 82289 bus arbiter, 8286/7 transceivers, and 8282/3 latches provide a buffered and decoded system bus interface. The 82284 generates the system clock and synchronizes $\overline{\text{READY}}$ and RESET. The 82288 converts bus operation status encoded by the 80286 into command and bus control signals. The 82289 bus arbiter generates multibus bus arbitration signals. These components can provide the timing and electrical power drive levels required for most system bus interfaces including the multibus.

Physical Memory and I/O Interface

A maximum of 16 megabytes of physical memory can be addressed in protected mode. One megabyte can be addressed in real address mode. Memory is accessible as bytes or words. Words consist of any two consecutive bytes addressed with the least significant byte stored in the lowest address.

Byte transfers occur on either half of the 16-bit local data bus. Even bytes are accessed over D_{7-0} while odd bytes are transferred over D_{15-8}. Even-addressed words are transferred over D_{15-0} in one bus cycle, while odd-addressed words require two bus operations. The first transfers data on D_{15-8}, and the second transfers data on D_{7-0}. Both byte data transfers occur automatically, transparent to software.

Two bus signals, A_0 and $\overline{\text{BHE}}$, control transfers over the lower and upper halves of the data bus. Even address

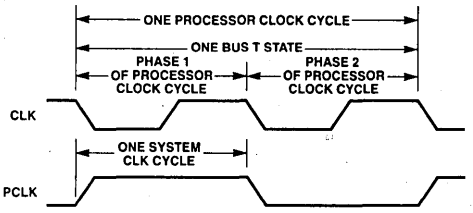
byte transfers are indicated by A_0 LOW and $\overline{BHE}$ HIGH. Odd address byte transfers are indicated by A_0 HIGH and $\overline{BHE}$ LOW. Both A_0 and $\overline{BHE}$ are LOW for even address word transfers.

The I/O address space contains 64K addresses in both modes. The I/O space is accessible as either bytes or words, as is memory. Byte wide peripheral devices may be attached to either the upper or lower byte of the data bus. Byte-wide I/O devices attached to the upper data byte (D_{15-8}) are accessed with odd I/O addresses. Devices on the lower data byte are accessed with even I/O addresses. An interrupt controller such as the 8259A must be connected to the lower data byte (D_{7-0}) for proper return of the interrupt vector.

Bus Operation

The 80286 uses a double frequency system clock (CLK input) to control bus timing. All signals on the local bus are measured relative to the system CLK input. The CPU divides the system clock by 2 to produce the internal processor clock, which determines bus state. Each processor clock is composed of two system clock cycles named phase 1 and phase 2. The 82284 clock generator output (PCLK) identifies the next phase of the processor clock. (See Figure 21.)

Figure 21. System and Processor Clock Relationships

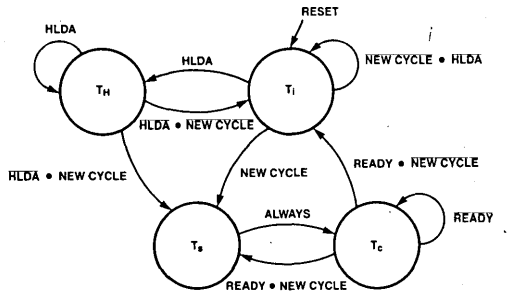


Six types of bus operations are supported; memory read, memory write, I/O read, I/O write, interrupt acknowledge, and halt/shutdown. Data can be transferred at a maximum rate of one word per two processor clock cycles.

The iAPX 286 bus has three basic states: idle (T_i), send status (T_s), and perform command (T_c). The 80286 CPU also has a fourth local bus state called hold (T_h). T_h indicates that the 80286 has surrendered control of the local bus to another bus master in response to a HOLD request.

Each bus state is one processor clock long. Figure 22 shows the four 80286 local bus states and allowed transitions.

Figure 22. 80286 Bus States



Bus States

The idle (T_i) state indicates that no data transfers are in progress or requested. The first active state, T_s is signalled by either status line $\overline{S1}$ or $\overline{S0}$ going LOW also identifying phase 1 of the processor clock. During T_s, the command encoding, the address, and data (for a write operation) are available on the 80286 output pins. The 82288 bus controller decodes the status signals and generates Multibus compatible read/write command and local transceiver control signals.

After T_s, the perform command (T_c) state is entered. Memory or I/O devices respond to the bus operation during T_c, either transferring read data to the CPU or accepting write data. T_c states may be repeated as often as necessary to assure sufficient time for the memory or I/O device to respond. The $\overline{READY}$ signal determines whether T_c is repeated.

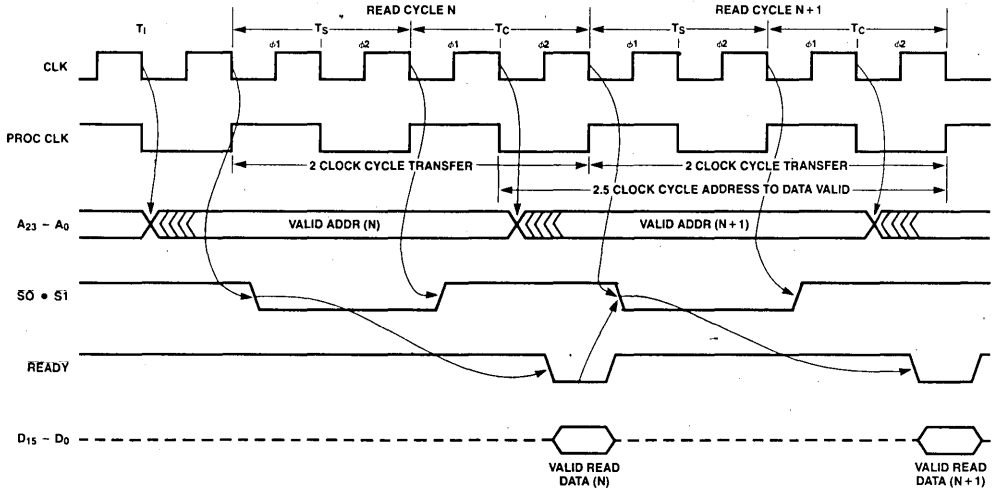
During hold (T_h), the 80286 will float all address, data, and status output pins enabling another bus master to use the local bus. The 80286 HOLD input signal is used to place the 80286 into the T_h state. The 80286 HLDA output signal indicates that the CPU has entered T_h.

Pipelined Addressing

The 80286 uses a local bus interface with pipelined timing to allow as much time as possible for data access. Pipelined timing allows bus operations to be performed in two processor cycles, while allowing each individual bus operation to last for three processor cycles.

The timing of the address outputs is pipelined such that the address of the next bus operation becomes available during the current bus operation. Or in other words, the first clock of the next bus operation is overlapped with the last clock of the current bus operation. Therefore, address decode and routing logic can operate in ad-

Figure 23. Basic Bus Cycle



vance of the next bus operation. External address latches may hold the address stable for the entire bus operation, and provide additional AC and DC buffering.

The 80286 does not maintain the address of the current bus operation during all T_C states. Instead, the address for the next bus operation may be emitted during phase 2 of any T_C. The address remains valid during phase 1 of the first T_C to guarantee hold time, relative to ALE, for the address latch inputs.

Bus Control Signals

The 82288 bus controller provides control signals; address latch enable (ALE), Read/Write commands, data transmit/receive (DT/$\overline{R}$), and data enable (DEN) that control the address latches, data transceivers, write enable, and output enable for memory and I/O systems.

The Address Latch Enable (ALE) output determines when the address may be latched. ALE provides at least one system CLK period of address hold time from the end of the previous bus operation until the address for the next bus operation appears at the latch outputs. This address hold time is required to support Multibus® and common memory systems.

The data bus transceivers are controlled by 82288 outputs Data Enable (DEN) and Data Transmit/Receive (DT/$\overline{R}$). DEN enables the data transceivers; while DT/$\overline{R}$ controls transceiver direction. DEN and DT/$\overline{R}$ are timed to prevent bus contention between the bus master, data bus transceivers, and system data bus tranceivers.

Command Timing Controls

Two system timing customization options, command extension and command delay, are provided on the iAPX 286 local bus.

Command extension allows additional time for external devices to respond to a command and is analogous to inserting wait states on the 8086. External logic can control the duration of any bus operation such that the operation is only as long as necessary. The $\overline{READY}$ input signal can extend any bus operation for as long as necessary.

Command delay allows an increase of address or write data setup time to system bus command active for any bus operation by delaying when the system bus command becomes active. Command delay is controlled by the 82288 CMDLY input. After T_S, the bus controller samples CMDLY at each failing edge of CLK. If CMDLY is HIGH, the 82288 will not activate the command signal. When CMDLY is LOW, the 82288 will activate the command signal. After the command becomes active, the CMDLY input is not sampled.

When a command is delayed, the available response time from command active to return read data or accept write data is less. To customize system bus timing, an address decoder can determine which bus operations require delaying the command. The CMDLY input does not affect the timing of ALE, DEN, or DT/$\overline{R}$.

Figure 24. CMDLY Controls and Leading Edge of the Command

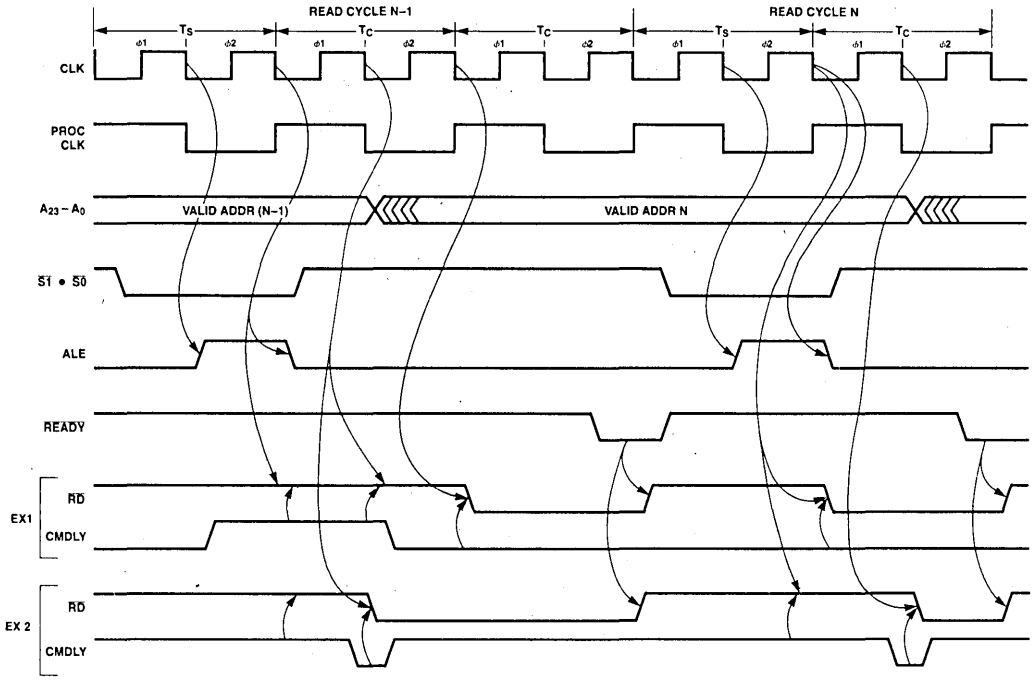


Figure 24 illustrates four uses of CMDLY. Example 1 shows delaying the read command two system CLKs for cycle N-1 and no delay for cycle N, and example 2 shows delaying the read command one system CLK for cycle N-1 and one system CLK delay for cycle N.

Bus Cycle Termination

At maximum transfer rates, the iAPX 286 bus alternates between the status and command states. The bus status signals become inactive after T_s so that they may correctly signal the start of the next bus operation after the completion of the current cycle. No external indication of T_c exists on the iAPX 286 local bus. The bus master and bus controller enter T_c directly after T_s and continue executing T_c cycles until terminated by $\overline{\text{READY}}$.

$\overline{\text{READY}}$ Operation

The current bus master and 82288 bus controller terminate each bus operation simultaneously to achieve maximum bus bandwidth. Both are informed in advance by $\overline{\text{READY}}$ active which identifies the last T_c cycle of the current bus operation. The bus master and bus controller must see the same sense of the $\overline{\text{READY}}$ signal, thereby requiring $\overline{\text{READY}}$ be synchronous to the system clock.

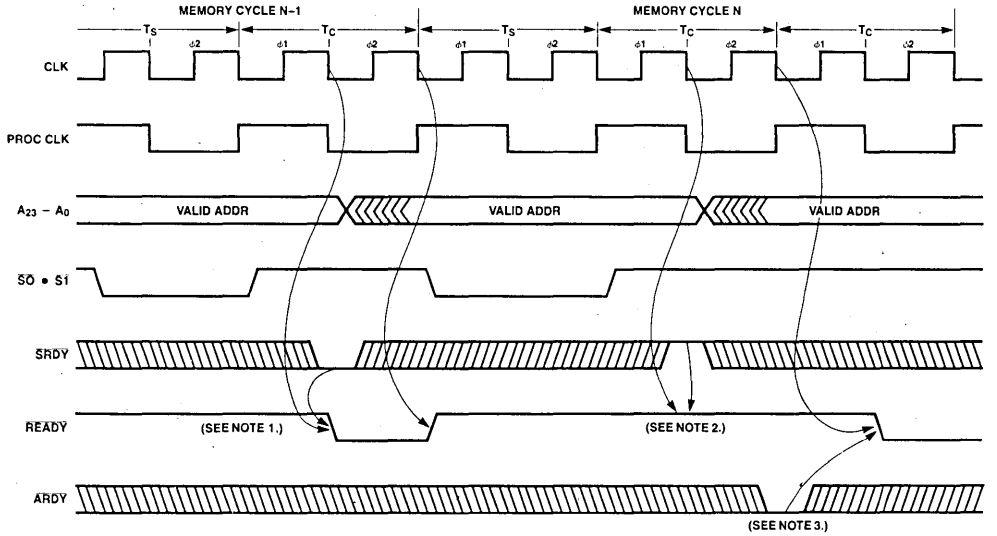
Synchronous Ready

The 82284 clock generator provides $\overline{\text{READY}}$ synchronization from both synchronous and asynchronous sources (see Figure 25). The synchronous ready input ($\overline{\text{SRDY}}$) of the clock generator is sampled with the falling edge of CLK at the end of phase 1 of each T_c. The state of $\overline{\text{SRDY}}$ is then broadcast to the bus master and bus controller via the $\overline{\text{READY}}$ output line.

Asynchronous Ready

Many systems have devices or subsystems that are asynchronous to the system clock. As a result, their ready outputs cannot be guaranteed to meet the 82284 $\overline{\text{SRDY}}$ setup and hold time requirements. The 82284 asynchronous ready input ($\overline{\text{ARDY}}$) is designed to accept such signals. The $\overline{\text{ARDY}}$ input is sampled at the beginning of each T_c cycle by 82284 synchronization logic. This provides a system CLK cycle time to resolve its value before broadcasting it to the bus master and bus controller.

Figure 25. Synchronous and Asynchronous Ready



NOTES:
1. $\overline{\text{SRDYEN}}$ is active low
2. If $\overline{\text{SRDYEN}}$ is high, the state of $\overline{\text{SRDY}}$ will not effect $\overline{\text{READY}}$
3. $\overline{\text{ARDYEN}}$ is active low

Each ready input of the 82284 has an enable pin ($\overline{\text{SRDYEN}}$ and $\overline{\text{ARDYEN}}$) to select whether the current bus operation will be terminated by the synchronous or asynchronous ready. Either of the ready inputs may terminate a bus operation. These enable inputs are active low and have the same timing as their respective ready inputs. Address decode logic usually selects whether the current bus operation should be terminated by $\overline{\text{ARDY}}$ or $\overline{\text{SRDY}}$.

Data Bus Control

Figures 26, 27, and 28 show how the DT/$\overline{\text{R}}$, DEN, data bus, and address signals operate for different combinations of read, write, and idle bus operations. DT/$\overline{\text{R}}$ goes active (LOW) for a read operaton. DT/$\overline{\text{R}}$ remains HIGH before, during, and between write operations.

The data bus is driven with write data during the second phase of T_S. The delay in write data timing allows the read data drivers, from a previous read cycle, sufficient time to enter 3-state OFF before the 80286 CPU begins driving the local data bus for write operations. Write data will always remain valid for one system clock past the last T_C to provide sufficient hold time for Multibus or other similar memory or I/O systems. During write-read or write-idle sequences the data bus enters 3-state OFF during the second phase of the processor cycle after the last T_C. In a write-write sequence the data bus does not enter 3-state OFF between T_C and T_S.

Bus Usage

The 80286 local bus may be used for several functions: instruction data transfers, data transfers by other bus masters, instruction fetching, processor extension data transfers, interrupt acknowledge, and halt/shutdown. This section describes local bus activities which have special signals or requirements.

Figure 26. Back to Back Read-Write Cycles

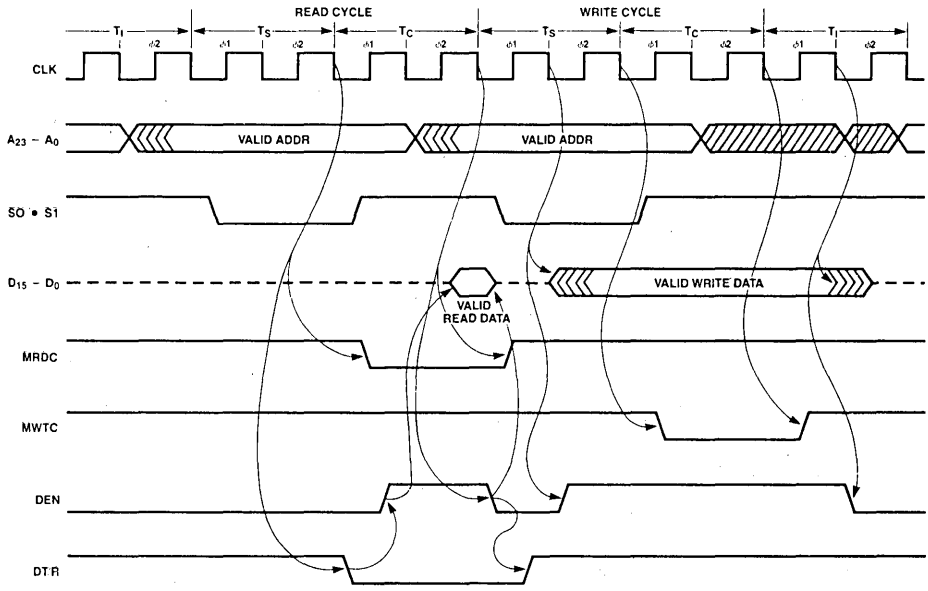


Figure 27. Back to Back Write-Read Cycles

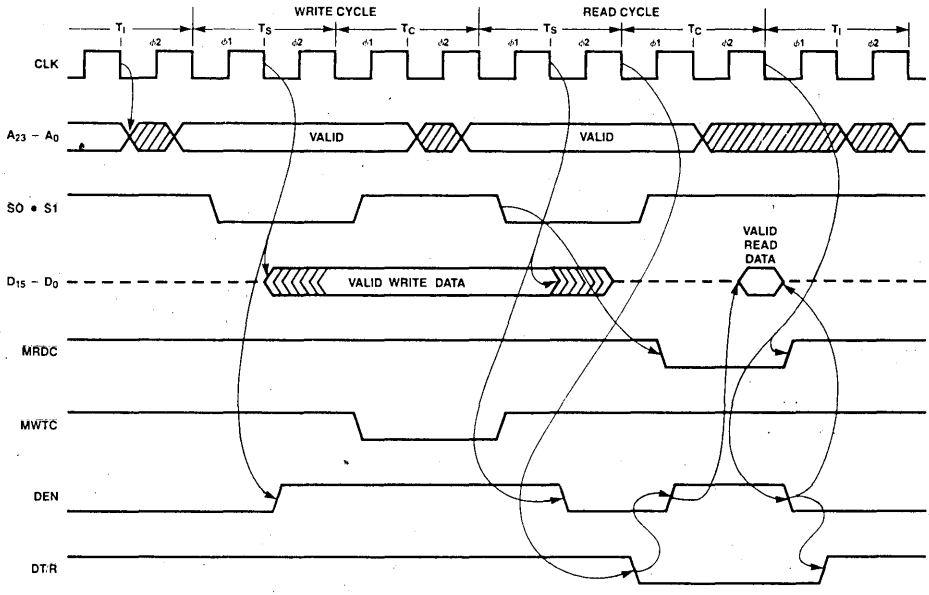
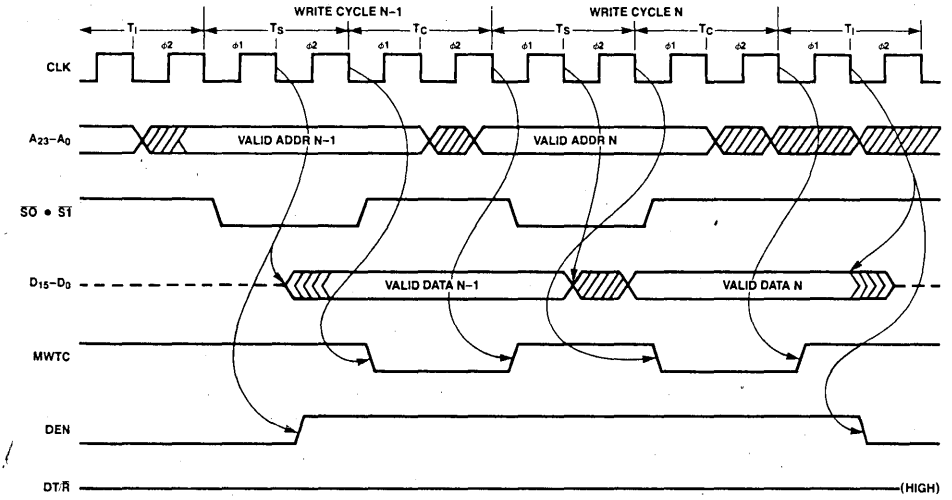


Figure 28. Back to Back Write-Write Cycles



HOLD and HLDA

HOLD and HLDA allow another bus master to gain control of the local bus by placing the 80286 bus into the T_h state. The sequence of events required to pass control between the 80286 and another local bus master are shown in Figure 29.

In this example, the 80286 is initially in the T_h state as signaled by HLDA being active. Upon leaving T_h, as signaled by HLDA going inactive, a write operation is started. During the write operation another local bus master requests the local bus from the 80286 as shown by the HOLD signal. After completing the write operation, the 80286 performs one T_i bus cycle, to guarantee write data hold time, then enters T_h as signaled by HLDA going active.

The CMDLY signal and $\overline{ARDY}$ ready are used to start and stop the write bus command, respectively. Note that $\overline{SRDY}$ must be inactive or disabled by $\overline{SRDYEN}$ to guarantee $\overline{ARDY}$ will terminate the cycle.

Instruction Fetching

The 80286 Bus Unit (BU) will fetch instructions ahead of the current instruction being executed. This activity is called prefetching. It occurs when the local bus would otherwise be idle and obeys the following rules:

A prefetch bus operation starts when at least two bytes of the 6-byte prefetch queue are empty.

The prefetcher normally performs word prefetches independent of the byte alignment of the code segment base in physical memory.

The prefetcher will perform only a byte code fetch operation for control transfers to an instruction beginning on a numerically odd physical address.

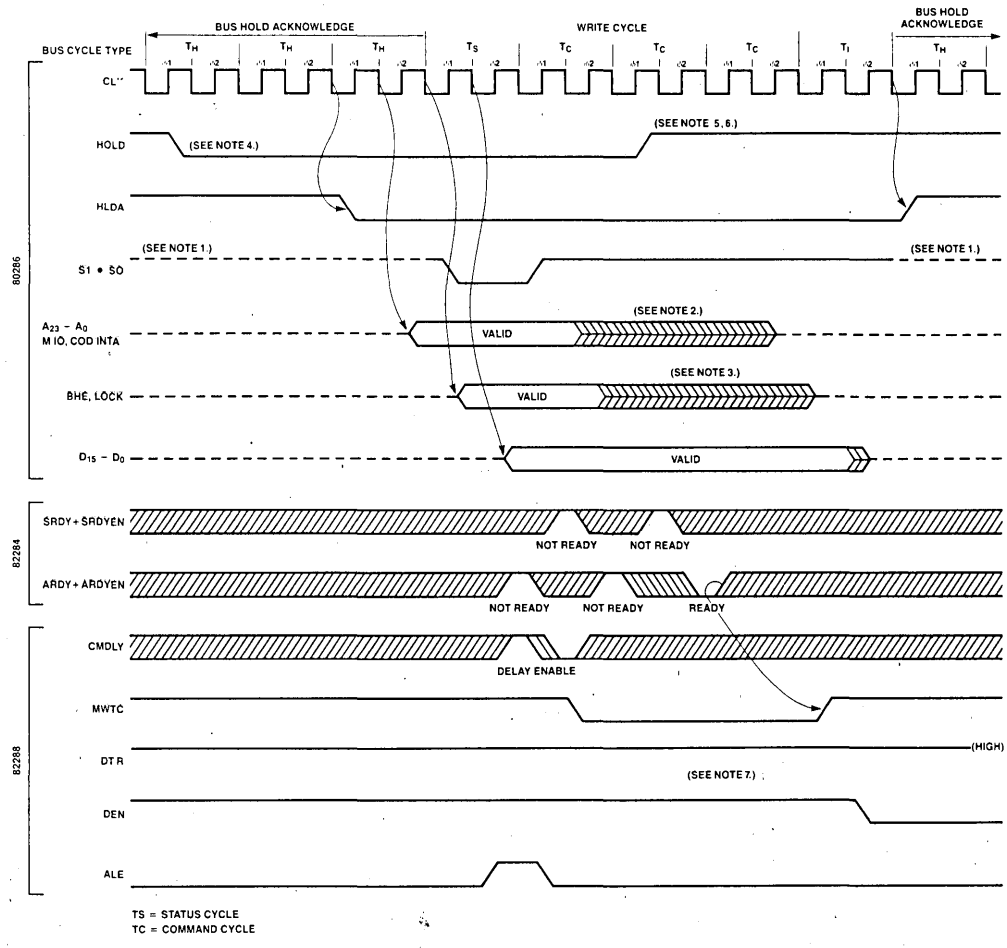
Prefetching stops whenever a control transfer or HLT instruction is decoded by the IU and placed into the instruction queue.

In real address mode, the prefetcher may fetch up to 5 bytes beyond the last control transfer or HLT instruction in a code segment.

In protected mode, the prefetcher will never cause a segment overrun exception. The prefetcher stops at the last physical memory word of the code segment. Exception 13 will occur if the program attempts to execute beyond the last full instruction in the code segment.

If the last byte of a code segment appears on an even physical memory address, the prefetcher will read the next physical byte of memory (perform a word code fetch). The value of this byte is ignored and any attempt to execute it causes exception 13.

Figure 29. Multibus Write Terminated by Asynchronous Ready with Bus Hold



NOTES:

1. Status lines are not driven by 80286, yet remain high due to pullup resistors in 82288 and 82289 during HOLD state.
2. Address, M/$\overline{IO}$ and COD/$\overline{INTA}$ may start floating during any TC depending on when internal 80286 bus arbiter decides to release bus to external HOLD. The float starts in $\phi 2$ of TC.
3. $\overline{BHE}$ and $\overline{LOCK}$ may start floating after the end of any TC depending on when internal 80286 bus arbiter decides to release bus to external HOLD.
4. The minimum HOLD ↓ to HLDA ↓ time is shown. Maximus is one T_H longer.
5. The earliest HOLD ↑ time is shown which will always allow a subsequent memory cycle if pending.
6. The minimum HOLD ↑ to HLDA ↑ time is shown. Maximum is a function of the instruction, type of bus cycle and other machine status (i.e., Interrupts, Waits, Lock, etc.)
7. Asynchronous ready allows termination of the cycle. Synchronous ready does not signal ready in this example. Synchronous ready state is ignored after ready is signaled via the asynchronous input.

Processor Extension Transfers

The processor extension interface uses I/O port addresses 00F8(H), 00FA(H), and 00FC(H) which are part of the I/O port address range and is a reserved area. An ESC instruction with EM = 0 and TS = 0 will perform I/O bus operations to one or more of these I/O port addresses independent of the value of IOPL and CPL.

ESC instructions with memory references enable the CPU to accept PEREQ inputs for processor extension operand transfers. The CPU will determine the operand starting address and read/write status of the instruction. For each operand transfer, two or three bus operations are performed, one word transfer with I/O port address 00FA(H) and one or two bus operations with memory. Three bus operations are required for each word operand aligned on an odd byte address.

Interrupt Acknowledge Sequence

Figure 30 illustrates an interrupt acknowledge sequence performed by the 80286 in response to an INTR input. An interrupt acknowledge sequence consists of two INTA bus operations. The first allows a master 8259A Programmable Interrupt Controller (PIC) to determine which if any of its slaves should return the interrupt vector. An eight bit vector is read by the 80286 during the second INTA bus operation to select an interrupt handler routine from the interrupt table.

The Master Cascade Enable (MCE) signal of the 82288 is used to enable the cascade address drivers, during INTA bus operations (See Figure 30), onto the local address bus for distribution to slave interrupt controllers via the system address bus. The 80286 emits the $\overline{LOCK}$ signal (active LOW) during T_s of the first INTA bus operation. A local bus "hold" request will not be honored until the end of the second INTA bus operation.

Three idle processor clocks are provided by the 80286 between INTA bus operations to allow for the minimum INTA to INTA time and CAS (cascade address) out delay of the 8259A. The second INTA bus operation must always have at least one extra T_c state added via logic controlling $\overline{READY}$. A_{23}–A_0 are in 3-state OFF until after the first T_c state of the second INTA bus operation. This prevents bus contention between the cascade address drivers and CPU address drivers. The extra T_c state allows time for the 80286 to resume driving the address lines for subsequent bus operations.

Local Bus Usage Priorities

The 80286 local bus is shared among several internal units and external HOLD requests. In case of simultaneous requests, their relative priorities are:

(Highest) Any transfers which assert $\overline{LOCK}$ either explicitly (via the LOCK instruction prefix) or implicitly (i.e. segment descriptor access, interrupt acknowledge sequence, or an XCHG with memory).

The second of the two byte bus operations required for an odd aligned word operand.

Local bus request via HOLD input.

Processor extension data operand transfer via PEREQ input.

Data transfer performed by EU as part of an instruction.

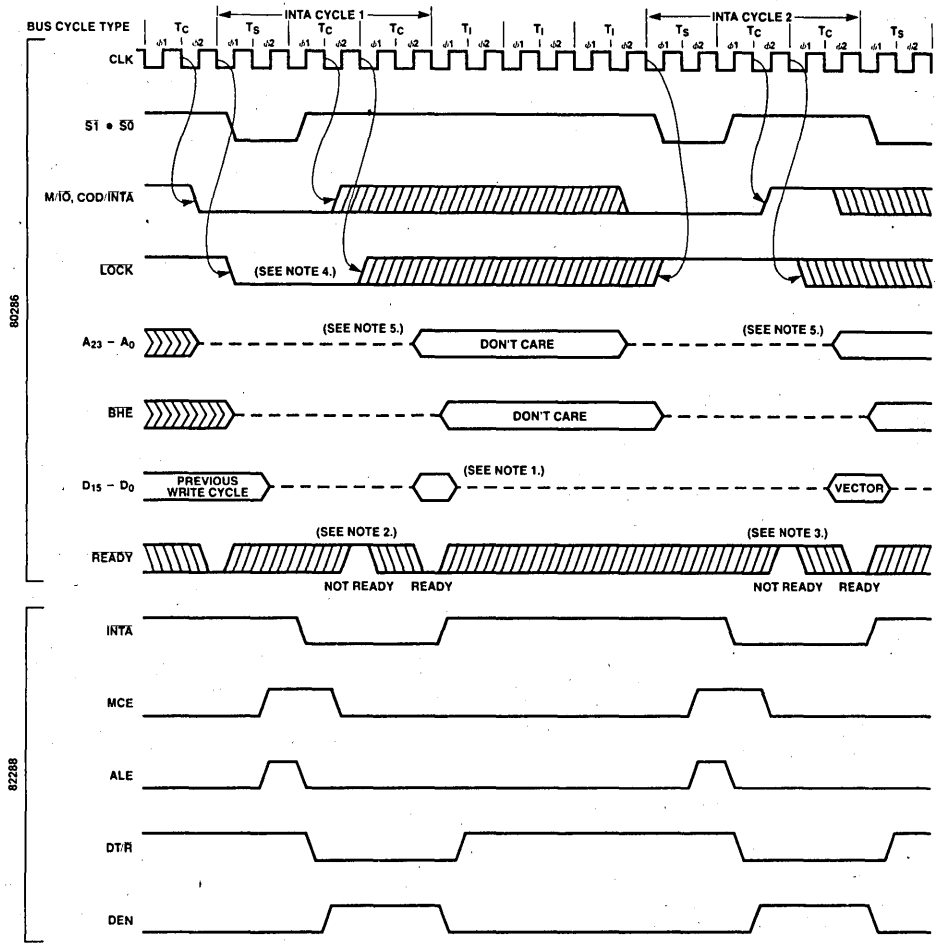
(Lowest) An instruction prefetch request from BU. The EU will inhibit prefetching two processor clocks in advance of any data transfers to minimize waiting by EU for a prefetch to finish.

Halt or Shutdown Cycles

The 80286 externally indicates halt or shutdown conditions as a bus operation. These conditions occur due to a HLT instruction or multiple protection exceptions while attempting to execute one instruction. A halt or shutdown bus operation is signalled when $\overline{S1}$, $\overline{S0}$ and COD/$\overline{INTA}$ are LOW and M/$\overline{IO}$ is HIGH. A_1 HIGH indicates halt, and A_1 LOW indicates shutdown. The 82288 bus controller does not issue ALE, nor is $\overline{READY}$ required to terminate a halt or shutdown bus operation.

During halt or shutdown, the 80286 may service PEREQ or HOLD requests. A processor extension segment overrun exception during shutdown will inhibit further service of PEREQ. Either NMI or RESET will force the 80286 out of either halt or shutdown. An INTR, if interrupts are enabled, or a processor extension segment overrun exception will also force the 80286 out of halt.

Figure 30. Interrupt Acknowledge Sequence



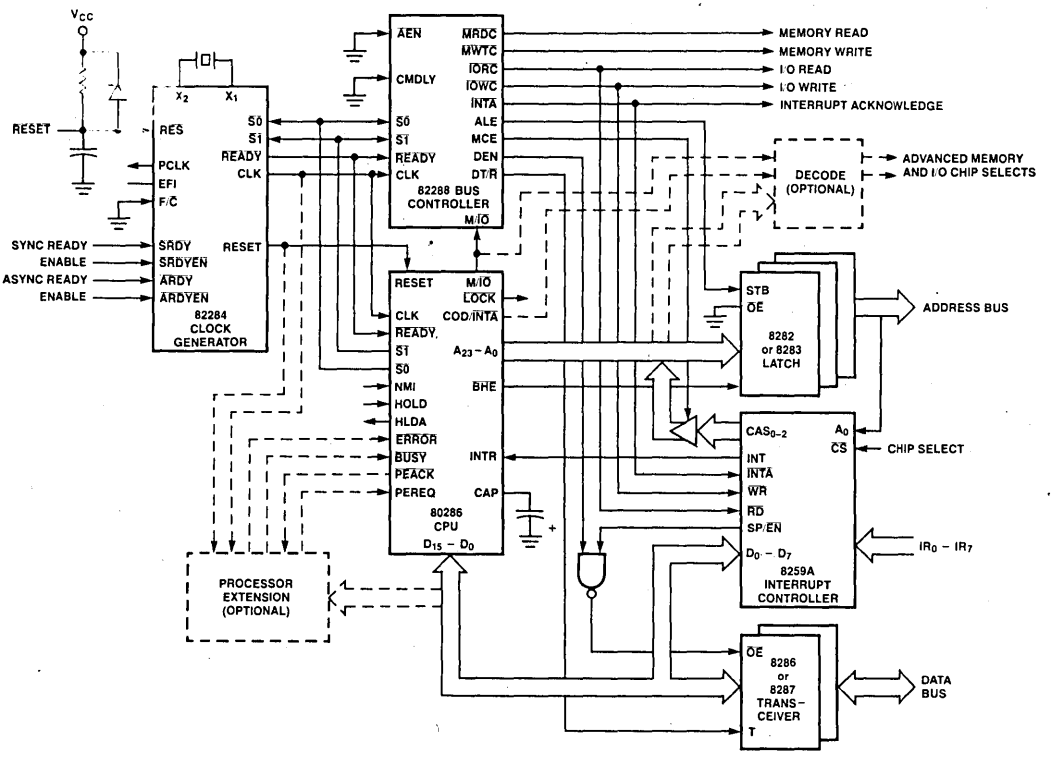
NOTES:

1. Data is ignored.
2. First INTA cycle should have at least one wait state inserted to meet 8259A minimum INTA pulse width.
3. Second INTA cycle must have at least one wait state inserted since the CPU will not drive $A_{23} - A_0$, $\overline{BHE}$, and $\overline{LOCK}$ until after the first TC state.

   The CPU imposed one/clock delay prevents bus contention between cascade address buffer being disabled by MCE ↓ and address outputs.

   Without the wait state, the 80286 address will not be valid for a memory cycle started immediately after the second INTA cycle. The 8259A also requires one wait state for minimum INTA pulse width.
4. $\overline{LOCK}$ is active for the first INTA cycle to prevent the 82289 from releasing the bus between INTA cycles in a multi-master system.
5. $A_{23} - A_0$ exits 3-state OFF during $\phi 2$ of the second T_C in the INTA cycle.

Figure 31. Basic iAPX 286 System Configuration



SYSTEM CONFIGURATIONS

The versatile bus structure of the iAPX 286 microsystem, with a full complement of support chips, allows flexible configuration of a wide range of systems. The basic configuration, shown in Figure 31, is similar to an iAPX 86 maximum mode system. It includes the CPU plus an 8259A interrupt controller, 82284 clock generator, and the 82288 Bus Controller. The iAPX 86 latches (8282 and 8283) and transceivers (8286 and 8287) may be used in an iAPX 286 microsystem.

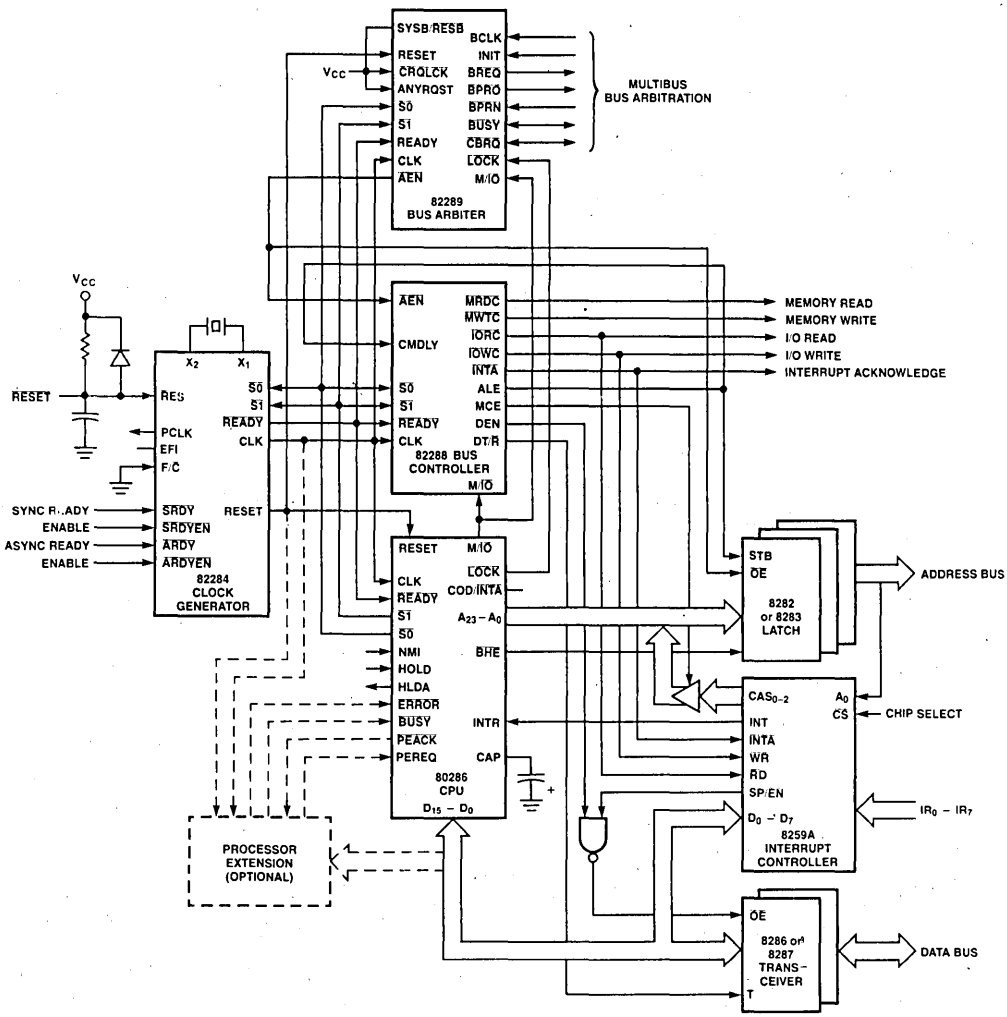
As indicated by the dashed lines in Figure 31, the ability to add processor extensions is an integral feature of iAPX 286 microsystems. The processor extension interface allows external hardware to perform special functions and transfer data concurrent with CPU execution of other instructions. Full system integrity is maintained because the 80286 supervises all data transfers and instruction execution for the processor extension.

The iAPX 286 with the 80287 numeric processor extension (NPX) uses this interface. The iAPX 286/287 has all the instructions and data types of an iAPX 86/87 or iAPX 88/87. The 80287 NPX can perform numeric calculations and data transfers concurrently with CPU program execution. Numerics code and data have the same integrity as all other information protected by the iAPX 286 protection mechanism.

The 80286 can overlap chip select decoding and address propagation during the data transfer for the previous bus operation. This information is latched into the 8282/3's by ALE during the middle of a T_S cycle. The latched chip select and address information remains stable during the bus operation while the next cycles address is being decoded and propagated into the system. Decode logic can be implemented with a high speed bipolar PROM.

The optional decode logic shown in Figure 31 takes advantage of the overlap between address and data of the 80286 bus cycle to generate advanced memory and IO-select signals. This minimizes system performance

Figure 32. Multibus System Bus Interface



degradation caused by address propogation and decode delays. In addition to selecting memory and I/O, the advanced selects may be used with configurations supporting local and system buses to enable the appropriate bus interface for each bus cycle. The COD/$\overline{\text{INTA}}$ and M/$\overline{\text{IO}}$ signals are applied to the decode logic to distinguish between interrupt, I/O, code and data bus cycles.

By adding the 82289 bus arbiter chip the 80286 provides a Multibus system bus interface as shown in Figure 32. The ALE output of the 82288 for the Multibus bus is connected to its CMDLY input to delay the start of commands one system CLK as required to meet Multibus address and write data setup times. This arrangement will add at least one extra T_C state to each bus operation which uses the Multibus.

A second 82288 bus controller and additional latches and transceivers could be added to the local bus of Figure 32. This configuration allows the 80286 to support an on-board bus for local memory and peripherals, and the Multibus for system bus interfacing.

Figure 33. iAPX 286 System Configuration with Dual-Ported Memory

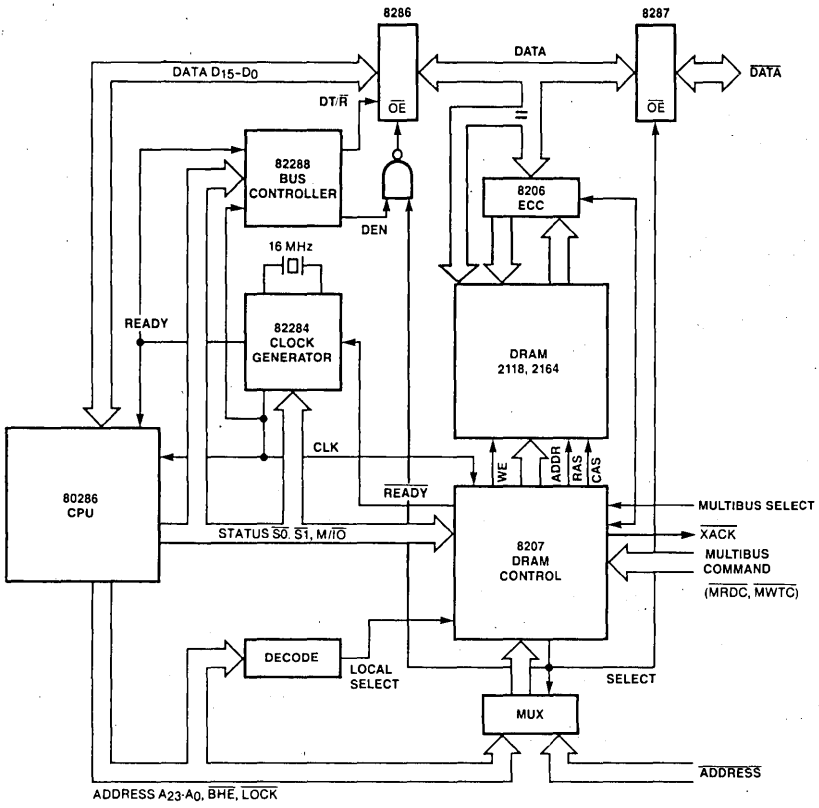
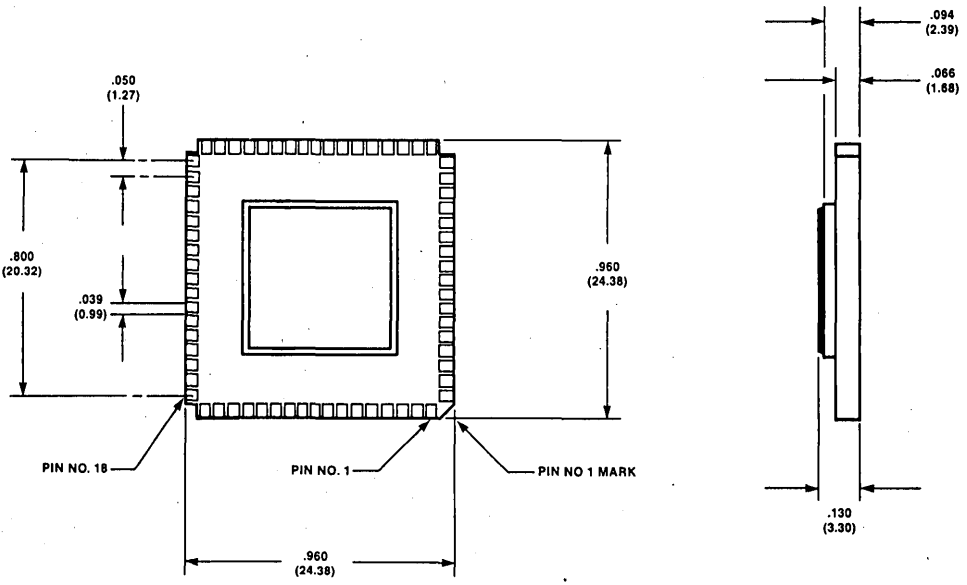


Figure 33 shows the addition of dual ported dynamic memory between the Multibus system bus and the iAPX 286 local bus. The dual port interface is provided by the 8207 Dual Port DRAM Controller. The 8207 runs synchronously with the CPU to maximize throughput for local memory references. It also arbitrates between requests from the local and system buses and performs functions such as refresh, initialization of RAM, and read/modify/write cycles. The 8207 combined with the 8206 Error Checking and Correction memory controller provide for single bit error correction. The dual-ported memory can be combined with a standard Multibus system bus interface to maximize performance and protection in multiprocessor system configurations.

PACKAGE

The 80286 is packaged in a 68-pin, leadless JEDEC type A hermetic chip carrier. Figure 34 illustrates the package, and Figure 2 shows the pinout.

Figure 34. 80286 JEDEC Type A Package



ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias . . . . . 0°C to 70°C
Storage Temperature . . . . . −65°C to +150°C
Voltage on Any Pin with
Respect to Ground . . . . . −0.3 to +7V
Power Dissipation . . . . . 3.6 Watt

**NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

D.C. CHARACTERISTICS (80286: T_A = 0°C to 70°C, V_{CC} = 5V ± 10%)

Symbol	Parameter	Min.	Max.	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	+0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}+0.5	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 3.0 mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = −400 µA
I_{CC}	Power Supply Current		600	mA	T_A = 25°C
I_{LI}	Input Leakage Current		±10	µA	0V ≤ V_{IN} ≤ V_{CC}
I_{LO}	Output Leakage Current		±10	µA	0.45V ≤ V_{OUT} ≤ V_{CC}
V_{CL}	Clock Input Low voltage	−0.5	+0.6	V	
V_{CH}	Clock Input HIgh Voltage	3.8	V_{CC}+1.0	V	
C_{IN}	Capacitance of Inputs (All input except CLK)		10	pF	fc = 1 MHz
C_O	Capacitance of I/O or outputs		20	pF	fc = 1 MHz
C_{CLK}	Capacitance of CLK Input		12	pF	f_c = 1 MHz

A.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5V ± 10%)

80286 Timing Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
1	System clock period	62.5	250	ns	
2	System clock low time	15	230	ns	at .6 Volts
3	System clock high time	20	235	ns	at 3.2 Volts
4	Asynchronous input setup time	20		ns	See note 1
5	Asynchronous input hold time	20		ns	See note 1
6	RESET setup time	20		ns	
7	RESET hold time	0		ns	
8	Read data in setup time	10		ns	
9	Read data in hold time	5		ns	
10	READY setup time	38.5		ns	
11	READY hold time	25		ns	
12	STATUS/$\overline{\text{PEACK}}$ valid delay	0	40	ns	C_L = 100 Pfd max
13	Address valid delay	0	60	ns	
14	Write data valid delay	0	50	ns	
15	Address/Status/Data float delay	0	60	ns	
16	HLDA valid delay	0	60	ns	

82284 Timing Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
17	$\overline{\text{SRDY}}$/$\overline{\text{SRDYEN}}$ setup time	15		ns	
18	$\overline{\text{SRDY}}$/$\overline{\text{SRDYEN}}$ hold time	0		ns	
19	$\overline{\text{ARDY}}$/$\overline{\text{ARDYEN}}$ setup time	0		ns	See note 1
20	$\overline{\text{ARDY}}$/$\overline{\text{ARDYEN}}$ hold time	16		ns	See note 1.
21	PCLK delay	0	40	ns	C_L = 75 pfd I_{OL} = 5.25 ma I_{OH} = −1.05 ma

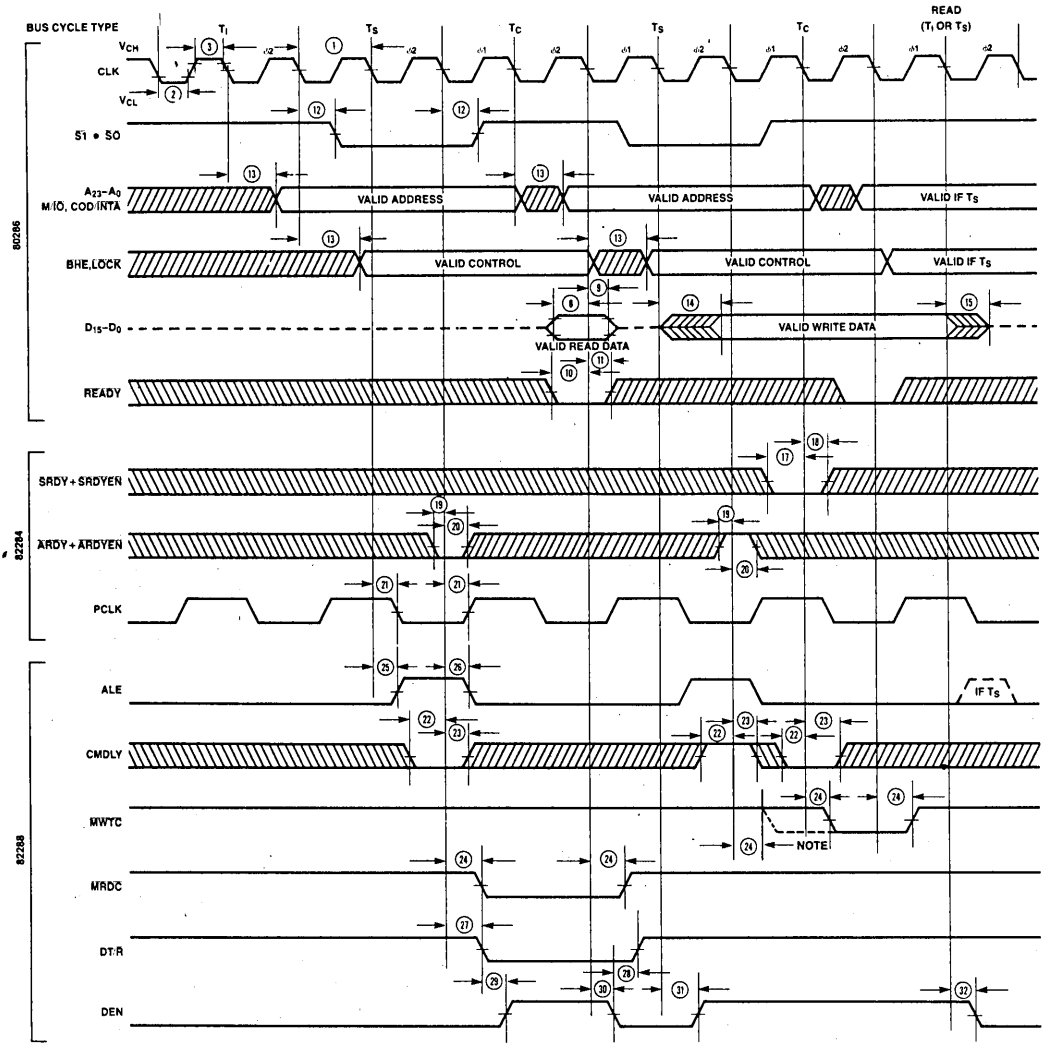
NOTE 1: These times are given for testing purposes to assure a predetermined action.

82288 Timing Requirements

Symbol	Parameter	Min.	Max.	Units	Test Conditions
22	CMDLY setup time	20		ns	
23	CMDLY hold time	0		ns	
24	Command delay	3	15	ns	C_L = 300 pfd max I_{OL} = 32 ma max I_{OH} = −5 ma max
25	ALE active delay	3	15	ns	C_L = 80 pfd max I_{OL} = 16 ma max I_{OH} = −1 ma max
26	ALE inactive delay	0	20	ns	
27	DT/$\overline{\text{R}}$ read active delay	0	20	ns	
28	DT/$\overline{\text{R}}$ read inactive delay	10	40	ns	
29	DEN read active delay	10	50	ns	
30	DEN read inactive delay	3	15	ns	
31	DEN write active delay	0	30	ns	
32	DEN write inactive delay	3	30	ns	

WAVEFORMS

MAJOR CYCLE TIMING

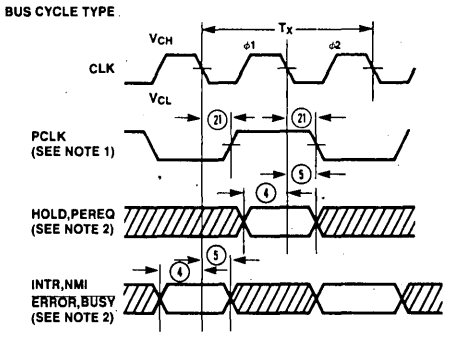


NOTE:

1. MWTC is valid at this point only if CMDLY is low.

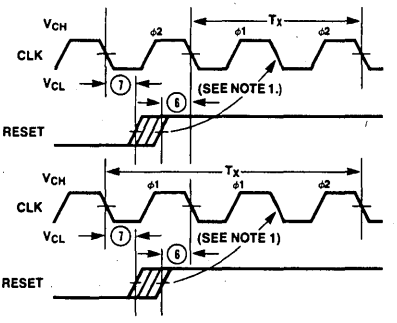
WAVEFORMS (Continued)

80286 ASYNCHRONOUS INPUT SIGNAL TIMING



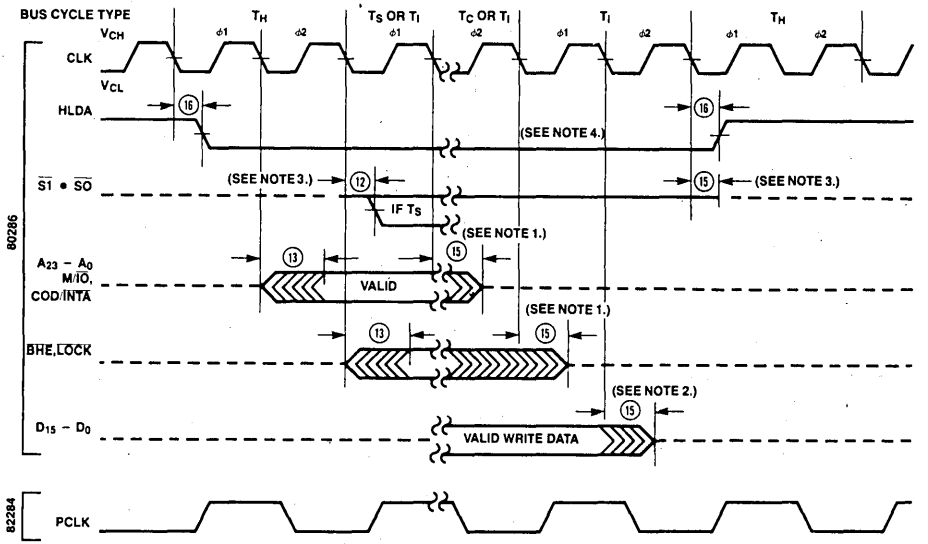
NOTES:
1. PCLK indicates which processor cycle phase will occur on the next CLK . PCLK may not indicate the correct phase until the first bus cycle is performed.
2. These inputs are asynchronous. The setup and hold times shown assure recognition for testing purposes.

80286 RESET INPUT TIMING AND SUBSEQUENT PROCESSOR CYCLE PHASE



NOTE 1: When RESET meets the setup time shown, the next CLK will start or repeat $\phi 1$ of a processor cycle.

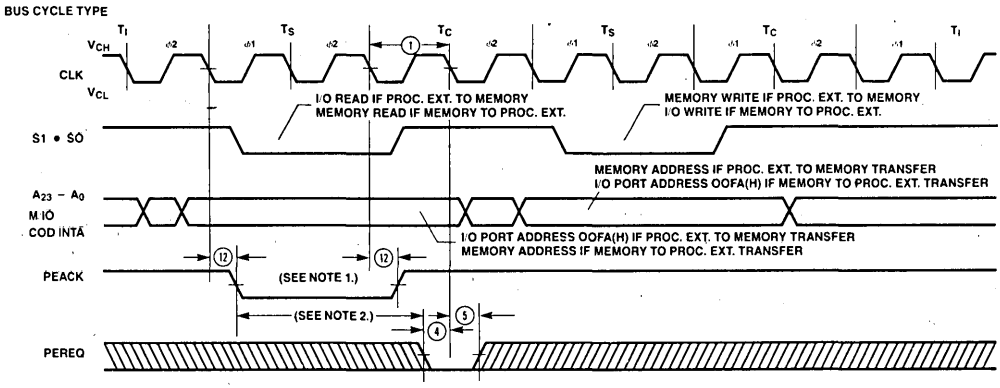
EXITING AND ENTERING HOLD



NOTES:
1. These signals may still be driven by the 80286 during the time shown. The worst case in terms of latest float time is shown.
2. The data bus will be driven as shown if the last cycle before T_I in the diagram was a write T_C.
3. The 80286 floats its status pins during T_H. External pullup resistors (in 82288) keep these signals high.
4. For HOLD request set up to HLDA, refer to Figure 29.

WAVEFORMS (Continued)

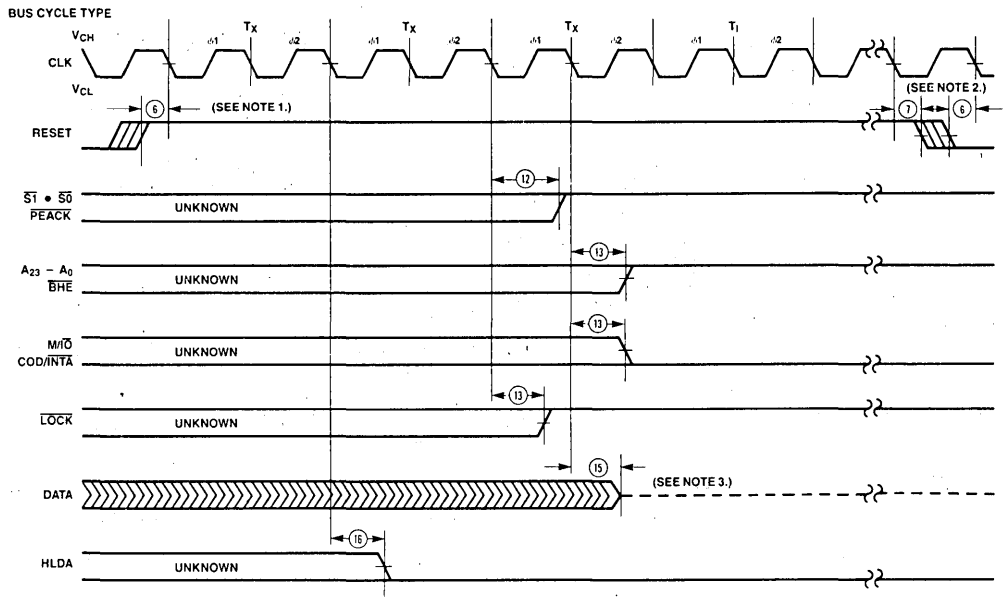
80286 PEREQ/$\overline{\text{PEACK}}$ TIMING REQUIRED PEREQ TIMING FOR ONE TRANSFER ONLY



NOTES:

1. $\overline{\text{PEACK}}$ always goes active during the first bus operation of a processor extension data operand transfer sequence. The first bus operation will be either a memory read at operand address or I/O read at port address OOFA(H).
2. To prevent a second processor extension data operand transfer, the worst case maximum time (Shown above) is: $3X(1) - (11)_{max.} - (4)_{min.}$. The actual, configuration dependent; maximum time is: $3X(1) - (11)_{max.} - (4)_{min.} + A X 2 X (1)$.
   A is the number of extra T_C states added to either the first or second bus operation of the processor extension data operand transfer sequence.

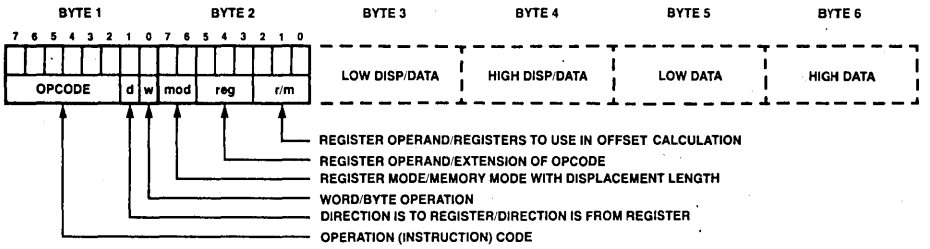
INITIAL 80286 PIN STATE DURING RESET



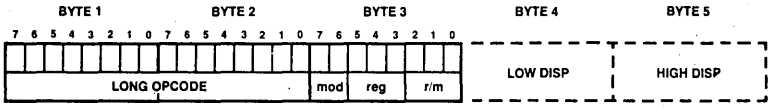
NOTES:

1. Setup time for RESET ↓ may be violated with the consideration that $\phi 1$ of the processor clock may begin one system CLK period later.
2. Setup and hold times for RESET ↑ must be met for proper operation.
3. The data bus is only guaranteed to be in 3-state OFF at the time shown.

Figure 35. 80286 Instruction Format Examples



A. SHORT OPCODE FORMAT EXAMPLE



B. LONG OPCODE FORMAT EXAMPLE

80286 INSTRUCTION SET SUMMARY

Instruction Timing Notes

The instruction clock counts listed below establish the maximum execution rate of the 80286. With no delays in bus cycles, the actual clock count of an 80286 program will average 5% more than the calculated clock count, due to instruction sequences which execute faster than they can be fetched from memory.

To calculate elapsed times for instruction sequences, multiply the sum of all instruction clock counts, as listed in the table below, by the processor clock period. An 8 MHz processor clock has a clock period of 125 nanoseconds and requires an 80286 system clock (CLK input) of 16 MHz.

Instruction Clock Count Assumptions

1. The instruction has been prefetched, decoded, and is ready for execution. Control transfer instruction clock counts include all time required to fetch, decode, and prepare the next instruction for execution.
2. Bus cycles do not require wait states.
3. There are no processor extension data transfer or local bus HOLD requests.
4. No exceptions occur during instruction execution.

Instruction Set Summary Notes

Addressing displacements selected by the MOD field are not shown. If necessary they appear after the instruction fields shown.

Above/below refers to unsigned value

Greater refers to positive signed value

Less refers to less positive (more negative) signed values

if d = 1 then to register; if d = 0 then from register

if w = 1 then word instruction; if w = 0 then byte instruction

if s = 0 then 16-bit immediate data form the operand

if s = 1 then an immediate data byte is sign-extended to form the 16-bit operand

x don't care

z used for string primitives for comparison with ZF FLAG

If two clock counts are given, the smaller refers to a register operand and the larger refers to a memory operand

* = add one clock if offset calculation requires summing 3 elements

n = number of times repeated

m = number of bytes of code in next instruction

Level (L)—Lexical nesting level of the procedure

The following comments describe possible exceptions, side effects, and allowed usage for instructions in both operating modes of the 80286.

REAL ADDRESS MODE ONLY

1. This is a protected mode instruction. Attempted execution in real address mode will result in an undefined opcode exception (6).
2. A segment overrun exception (13) will occur if a word operand reference at offset FFFF(H) is attempted.
3. This instruction may be executed in real address mode to initialize the CPU for protected mode.
4. The IOPL and NT fields will remain 0.
5. Processor extension segment overrun interrupt (9) will occur if the operand exceeds the segment limit.

EITHER MODE

6. An exception may occur, depending on the value of the operand.
7. $\overline{\text{LOCK}}$ is automatically asserted regardless of the presence or absence of the LOCK instruction prefix.

PROTECTED VIRTUAL ADDRESS MODE ONLY

8. The destination of an INT, JMP, CALL, RET or IRET instruction must be in the defined limit of a code segment or a general protection exception (13) occurs.
9. A general protection exception (13) will occur if the memory operand can not be used due to either a segment limit or access rights violation. If a stack segment limit is violated, a stack segment overrun exception (12) occurs.
10. For segment load operations, the CPL, RPL, and DPL must agree with privilege rules to avoid an exception. The segment must be present to avoid a not-present exception (11). If the SS register is the destination, and a segment not-present violation occurs, a stack exception (12) occurs.
11. All segment descriptor accesses in the GDT or LDT made by this instruction will automatically assert $\overline{\text{LOCK}}$ to maintain descriptor integrity in multiprocessor systems.
12. JMP, CALL, INT, RET, IRET instructions referring to another code segment will cause a general protection exception (13) if any privilege rule is violated.
13. A general protection exception (13) occurs if CPL $\neq$ 0.
14. A general protection exception (13) occurs if CPL > IOPL.
15. The IF field of the flag word is not updated if CPL > IOPL. The IOPL field is updated only if CPL = 0.
16. Any violation of privilege rules as applied to the selector operand do not cause a protection exception; rather, the instruction does not return a result and the zero flag is cleared.
17. If the starting address of the memory operand violates a segment limit, or an invalid access is attempted, a general protection exception (13) will occur before the ESC instruction is executed. A stack segment overrun exception (12) will occur if the stack limit is violated by the operand's starting address. If a segment limit is violated during an attempted data transfer then a processor extension segment overrun exception (9) occurs.

80286 INSTRUCTION SET SUMMARY

FUNCTION	FORMAT	CLOCK COUNT: Real Address Mode	CLOCK COUNT: Protected Virtual Address Mode	COMMENTS: Real Address Mode	COMMENTS: Protected Virtual Address Mode
DATA TRANSFER					
MOV = Move:					
Register to Register/Memory	1 0 0 0 1 0 0 w | mod reg r/m	2,3*	2,3*	2	9
Register/memory to register	1 0 0 0 1 0 1 w | mod reg r/m	2,5*	2,5*	2	9
Immediate to register/memory	1 1 0 0 0 1 1 w | mod 0 0 0 r/m | data | data if w = 1	2,3*	2,3*	2	9
Immediate to register	1 0 1 1 w reg | data | data if w = 1	2	2		
Memory to accumulator	1 0 1 0 0 0 0 w | addr-low | addr-high	5	5	2	9
Accumulator to memory	1 0 1 0 0 0 1 w | addr-low | addr-high	3	3	2	9
Register/memory to segment register	1 0 0 0 1 1 1 0 | mod 0 reg r/m	2,5*	17,19*	2	9,10,11
Segment register to register/memory	1 0 0 0 1 1 0 0 | mod 0 reg r/m	2,3*	2,3*	2	9
PUSH = Push:					
Memory	1 1 1 1 1 1 1 1 | mod 1 1 0 r/m	5*	5*	2	9
Register	0 1 0 1 0 reg	3	3	2	9
Segment register	0 0 0 reg 1 1 0	3	3	2	9
Immediate	0 1 1 0 1 0 s 0 | data | data if s = 0	3	3	2	9
PUSHA = Push All	0 1 1 0 0 0 0 0	17	17	2	9
POP = Pop:					
Memory	1 0 0 0 1 1 1 1 | mod 0 0 0 r/m	5*	5*	2	9
Register	0 1 0 1 1 reg	5	5	2	9
Segment register	0 0 0 reg 1 1 1 | (reg ≠ 01)	5	20	2	9,10,11
POPA = Pop All	0 1 1 0 0 0 0 1	19	19	2	9
XCHG = Exchange:					
Register/memory with register	1 0 0 0 0 1 1 w | mod reg r/m	3,5*	3,5*	2,7	7,9
Register with accumulator	1 0 0 1 0 reg	3	3		
IN = Input from:					
Fixed port	1 1 1 0 0 1 0 w | port	5	5		14
Variable port	1 1 1 0 1 1 0 w	5	5		14
OUT = Output to:					
Fixed port	1 1 1 0 0 1 1 w | port	3	3		14
Variable port	1 1 1 0 1 1 1 w	3	3		14
XLAT = Translate byte to AL	1 1 0 1 0 1 1 1	5	5		9
LEA = Load EA to register	1 0 0 0 1 1 0 1 | mod reg r/m	3*	3*		
LDS = Load pointer to DS	1 1 0 0 0 1 0 1 | mod reg r/m | (mod ≠ 11)	7*	21*	2	9,10,11
LES = Load pointer to ES	1 1 0 0 0 1 0 0 | mod reg r/m | (mod ≠ 11)	7*	21*	2	9,10,11
LAHF = Load AH with flags	1 0 0 1 1 1 1 1	2	2		
SAHF = Store AH into flags	1 0 0 1 1 1 1 0	2	2		
PUSHF = Push flags	1 0 0 1 1 1 0 0	3	3	2	9
POPF = Pop flags	1 0 0 1 1 1 0 1	5	5	2,4	9,15

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright Intel Corp., 1983.

80286 INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	CLOCK COUNT: Real Address Mode	CLOCK COUNT: Protected Virtual Address Mode	COMMENTS: Real Address Mode	COMMENTS: Protected Virtual Address Mode
ARITHMETIC					
ADD = Add:					
Reg/memory with register to either	000000dw | mod reg r/m	2,7*	2,7*	2	9
Immediate to register/memory	100000sw | mod000 r/m | data | data if sw=01	3,7*	3,7*	2	9
Immediate to accumulator	0000010w | data | data if w=1	3	3		
ADC = Add with carry:					
Reg/memory with register to either	000100dw | mod reg r/m	2,7*	2,7*	2	9
Immediate to register/memory	100000sw | mod010 r/m | data | data if sw=01	3,7*	3,7*	2	9
Immediate to accumulator	0001010w | data | data if w=1	3	3		
INC = Increment:					
Register/memory	1111111w | mod000 r/m	2,7*	2,7*	2	9
Register	01000 reg	2	2		
SUB = Subtract:					
Reg/memory and register to either	001010dw | mod reg r/m	2,7*	2,7*	2	9
Immediate from register/memory	100000sw | mod101 r/m | data | data if sw=01	3,7*	3,7*	2	9
Immediate from accumulator	0010110w | data | data if w=1	3	3		
SBB = Subtract with borrow:					
Reg/memory and register to either	000110dw | mod reg r/m	2,7*	2,7*	2	9
Immediate from register/memory	100000sw | mod011 r/m | data | data if sw=01	3,7*	3,7*	2	9
Immediate from accumulator	0001110w | data | data if w=1	3	3		
DEC = Decrement:					
Register/memory	1111111w | mod001 r/m	2,7*	2,7*	2	9
Register	01001 reg	2	2		
CMP = Compare:					
Register/memory with register	0011101w | mod reg r/m	2,6*	2,6*	2	9
Register with register/memory	0011100w | mod reg r/m	2,7*	2,7*	2	9
Immediate with register/memory	100000sw | mod111 r/m | data | data if sw=01	3,6*	3,6*	2	9
Immediate with accumulator	0011110w | data | data if w=1	3	3		
NEG = Change sign	1111011w | mod011 r/m	2	7*	2	7
AAA = ASCII adjust for add	00110111	3	3		
DAA = Decimal adjust for add	00100111	3	3		
AAS = ASCII adjust for subtract	00111111	3	3		
DAS = Decimal adjust for subtract	00101111	3	3		
MUL = Multiply (unsigned):	1111011w | mod100 r/m				
Register-Byte		13	13		
Register-Word		21	21		
Memory-Byte		16*	16*	2	9
Memory-Word		24*	24*	2	9
IMUL = Integer multiply (signed):	1111011w | mod101 r/m				
Register-Byte		13	13		
Register-Word		21	21		
Memory-Byte		16*	16*	2	9
Memory-Word		24*	24*	2	9
IMUL = Integer immediate multiply (signed)	011010s1 | mod reg r/m | data | data if s=0	21,24*	21,24*	2	9
DIV = Divide (unsigned):	1111011w | mod110 r/m				
Register-Byte		14	14		
Register-Word		22	22		
Memory-Byte		17*	17*	2,6	6,9
Memory-Word		25*	25*	2,6	6,9

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright Intel Corp., 1983.

80286 INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	CLOCK COUNT: Real Address Mode	CLOCK COUNT: Protected Virtual Address Mode	COMMENTS: Real Address Mode	COMMENTS: Protected Virtual Address Mode
ARITHMETIC (Continued):					
IDIV = Integer divide (signed):	1 1 1 1 0 1 1 w | mod 1 1 1 r/m				
Register-Byte		17	17		
Register-Word		25	25		
Memory-Byte		20*	20*	2	9
Memory-Word		28*	28*	2	9
AAM = ASCII adjust for multiply	1 1 0 1 0 1 0 0 | 0 0 0 0 1 0 1 0	16	16		
AAD = ASCII adjust for divide	1 1 0 1 0 1 0 1 | 0 0 0 0 1 0 1 0	14	14		
CBW = Convert byte to word	1 0 0 1 1 0 0 0	2	2		
CWD = Convert word to double word	1 0 0 1 1 0 0 1	2	2		
LOGIC					
Shift/Rotate Instructions:					
Register/Memory by 1	1 1 0 1 0 0 0 w | mod TTT r/m	2,7*	2,7*	2	9
Register/Memory by CL	1 1 0 1 0 0 1 w | mod TTT r/m	5+n,8+n*	5+n,8+n*	2	9
Register/Memory by Count	1 1 0 0 0 0 0 w | mod TTT r/m | count	5+n,8+n*	5+n,8+n*	2	9
	TTT Instruction: 0 0 0 ROL; 0 0 1 ROR; 0 1 0 RCL; 0 1 1 RCR; 1 0 0 SHL/SAL; 1 0 1 SHR; 1 1 1 SAR				
AND = And:					
Reg/memory and register to either	0 0 1 0 0 0 d w | mod reg r/m	2,7*	2,7*	2	9
Immediate to register/memory	1 0 0 0 0 0 0 w | mod 1 0 0 r/m | data | data if w = 1	3,7*	3,7*	2	9
Immediate to accumulator	0 0 1 0 0 1 0 w | data | data if w = 1	3	3		
TEST = And function to flags, no result:					
Register/memory and register	1 0 0 0 0 1 0 w | mod reg r/m	2,6*	2,6*	2	9
Immediate data and register/memory	1 1 1 1 0 1 1 w | mod 0 0 0 r/m | data | data if w = 1	3,6*	3,6*	2	9
Immediate data and accumulator	1 0 1 0 1 0 0 w | data | data if w = 1	3	3		
OR = Or:					
Reg/memory and register to either	0 0 0 0 1 0 d w | mod reg r/m	2,7*	2,7*	2	9
Immediate to register/memory	1 0 0 0 0 0 0 w | mod 0 0 1 r/m | data | data if w = 1	3,7*	3,7*	2	9
Immediate to accumulator	0 0 0 0 1 1 0 w | data | data if w = 1	3	3		
XOR = Exclusive or:					
Reg/memory and register to either	0 0 1 1 0 0 d w | mod reg r/m	2,7*	2,7*	2	9
Immediate to register/memory	1 0 0 0 0 0 0 w | mod 1 1 0 r/m | data | data if w = 1	3,7*	3,7*	2	9
Immediate to accumulator	0 0 1 1 0 1 0 w | data | data if w = 1	3	3		
NOT = Invert register/memory	1 1 1 1 0 1 1 w | mod 0 1 0 r/m	2,7*	2,7*	2	9
STRING MANIPULATION:					
MOVS = Move byte/word	1 0 1 0 0 1 0 w	5	5	2	9
CMPS = Compare byte/word	1 0 1 0 0 1 1 w	8	8	2	9
SCAS = Scan byte/word	1 0 1 0 1 1 1 w	7	7	2	9
LODS = Load byte/wd to AL/AX	1 0 1 0 1 1 0 w	5	5	2	9
STOS = Stor byte/wd from AL/A	1 0 1 0 1 0 1 w	3	3	2	9
INS = Input byte/wd from DX port	0 1 1 0 1 1 0 w	5	5	2	9,14
OUTS = Output byte/wd to DX port	0 1 1 0 1 1 1 w	5	5	2	9,14

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright Intel Corp., 1983.

80286 INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	CLOCK COUNT: Real Address Mode	CLOCK COUNT: Protected Virtual Address Mode	COMMENTS: Real Address Mode	COMMENTS: Protected Virtual Address Mode
STRING MANIPULATION (Continued):					
Repeated by count in CX					
MOVS = Move string	1 1 1 1 0 0 1 0 | 1 0 1 0 0 1 0 w	5+4n	5+4n	2	9
CMPS = Compare string	1 1 1 1 0 0 1 z | 1 0 1 0 0 1 1 w	5+9n	5+9n	2	9
SCAS = Scan string	1 1 1 1 0 0 1 z | 1 0 1 0 1 1 1 w	5+8n	5+8n	2	9
LODS = Load string	1 1 1 1 0 0 1 0 | 1 0 1 0 1 1 0 w	5+4n	5+4n	2	9
STOS = Store string	1 1 1 1 0 0 1 0 | 1 0 1 0 1 0 1 w	4+3n	4+3n	2	9
INS = Input string	1 1 1 1 0 0 1 0 | 0 1 1 0 1 1 0 w	5+4n	5+4n	2	9,14
OUTS = Output string	1 1 1 1 0 0 1 0 | 0 1 1 0 1 1 1 w	5+4n	5+4n	2	9,14
CONTROL TRANSFER					
CALL = Call:					
Direct within segment	1 1 1 0 1 0 0 0 | disp-low | disp-high	7+m	7+m	2	8
Register/memory indirect within segment	1 1 1 1 1 1 1 1 | mod 0 1 0 r/m	7+m,11+m*	7+m,11+m*	2	8,9
Direct intersegment	1 0 0 1 1 0 1 0 | segment offset	13+m	26+m	2	8,11,12
	segment selector				
Protected Mode Only (Direct Intersegment):					
Via call gate to same privilege level			41+m		8,11,12
Via call gate to different privilege level, no parameters			82+m		8,11,12
Via call gate to different privilege level, x parameters			86+4x+m		8,11,12
Via TSS			177+m		8,11,12
Via task gate			182+m		8,11,12
Indirect intersegment	1 1 1 1 1 1 1 1 | mod 0 1 1 r/m (mod ≠ 11)	16+m	29+m*	2	8,9,11,12
Protected Mode Only (Indirect Intersegment):					
Via call gate to same privilege level			44+m*		8,9,11,12
Via call gate to different privilege level, no parameters			83+m*		8,9,11,12
Via call gate to different privilege level, x parameters			90+4x+m*		8,9,11,12
Via TSS			180+m*		8,9,11,12
Via task gate			185+m*		8,9,11,12
JMP = Unconditional jump:					
Short/long	1 1 1 0 1 0 1 1 | disp-low	7+m	7+m		8
Direct within segment	1 1 1 0 1 0 0 1 | disp-low | disp-high	7+m	7+m		8
Register/memory indirect within segment	1 1 1 1 1 1 1 1 | mod 1 0 0 r/m	7+m,11+m*	7+m,11+m*	2	8,9
Direct intersegment	1 1 1 0 1 0 1 0 | segment offset	11+m	23+m		8,11,12
	segment selector				
Protected Mode Only (Direct Intersegment):					
Via call gate to same privilege level			38+m		8,11,12
Via TSS			175+m		8,11,12
Via task gate			180+m		8,11,12
Indirect intersegment	1 1 1 1 1 1 1 1 | mod 1 0 1 r/m (mod ≠ 11)	15+m*	26+m*	2	8,9,11,12
Protected Mode Only (Indirect Intersegment):					
Via call gate to same privilege level			41+m*		8,9,11,12
Via TSS			178+m*		8,9,11,12
Via task gate			183+m*		8,9,11,12
RET = Return from CALL:					
Within segment	1 1 0 0 0 0 1 1	11+m	11+m	2	8,9
Within seg adding immed to SP	1 1 0 0 0 0 1 0 | data-low | data-high	11+m	11+m	2	8,9
Intersegment	1 1 0 0 1 0 1 1	15+m	25+m	2	8,9,11,12
Intersegment adding immediate to SP	1 1 0 0 1 0 1 0 | data-low | data-high	15+m		2	8,9,11,12
Protected Mode Only (RET):					
To different privilege level			55+m		

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright Intel Corp., 1983.

80286 INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	CLOCK COUNT: Real Address Mode	CLOCK COUNT: Protected Virtual Address Mode	COMMENTS: Real Address Mode	COMMENTS: Protected Virtual Address Mode
CONTROL TRANSFER (Continued):					
JE/JZ = Jump on equal zero	0 1 1 1 0 1 0 0 | disp	7 + m or 3	7 + m or 3		8
JL/JNGE = Jump on less not greater or equal	0 1 1 1 1 1 0 0 | disp	7 + m or 3	7 + m or 3		8
JLE/JNG = Jump on less or equal not greater	0 1 1 1 1 1 1 0 | disp	7 + m or 3	7 + m or 3		8
JB/JNAE = Jump on below not above or equal	0 1 1 1 0 0 1 0 | disp	7 + m or 3	7 + m or 3		8
JBE/JNA = Jump on below or equal not above	0 1 1 1 0 1 1 0 | disp	7 + m or 3	7 + m or 3		8
JP/JPE = Jump on parity parity even	0 1 1 1 1 0 1 0 | disp	7 + m or 3	7 + m or 3		8
JO = Jump on overflow	0 1 1 1 0 0 0 0 | disp	7 + m or 3	7 + m or 3		8
JS = Jump on sign	0 1 1 1 1 0 0 0 | disp	7 + m or 3	7 + m or 3		8
JNE/JNZ = Jump on not equal not zero	0 1 1 1 0 1 0 1 | disp	7 + m or 3	7 + m or 3		8
JNL/JGE = Jump on not less greater or equal	0 1 1 1 1 1 0 1 | disp	7 + m or 3	7 + m or 3		8
JNLE/JG = Jump on not less or equal greater	0 1 1 1 1 1 1 1 | disp	7 + m or 3	7 + m or 3		8
JNB/JAE = Jump on not below above or equal	0 1 1 1 0 0 1 1 | disp	7 + m or 3	7 + m or 3		8
JNBE/JA = Jump on not below or equal above	0 1 1 1 0 1 1 1 | disp	7 + m or 3	7 + m or 3		8
JNP/JPO = Jump on not par par odd	0 1 1 1 1 0 1 1 | disp	7 + m or 3	7 + m or 3		8
JNO = Jump on not overflow	0 1 1 1 0 0 0 1 | disp	7 + m or 3	7 + m or 3		8
JNS = Jump on not sign	0 1 1 1 1 0 0 1 | disp	7 + m or 3	7 + m or 3		8
LOOP = Loop CX times	1 1 1 0 0 0 1 0 | disp	8 + m or 4	8 + m or 4		8
LOOPZ/LOOPE = Loop while zero equal	1 1 1 0 0 0 0 1 | disp	8 + m or 4	8 + m or 4		8
LOOPNZ/LOOPNE = Loop while not zero equal	1 1 1 0 0 0 0 0 | disp	8 + m or 4	8 + m or 4		8
JCXZ = Jump on CX zero	1 1 1 0 0 0 1 1 | disp	8 + m or 4	8 + m or 4		8
ENTER = Enter Procedure	1 1 0 0 1 0 0 0 | data-low | data-high | L				
L = 0		11	11	2	9
L = 1		15	15	2	9
L > 1		16 - 4(L - 1)	16 - 4(L - 1)	2	9
LEAVE = Leave Procedure	1 1 0 0 1 0 0 1	5	5	2	9
INT = Interrupt:					
Type specified	1 1 0 0 1 1 0 1 | type	23 + m		2	
Type 3	1 1 0 0 1 1 0 0	23 + m		2	
INTO = Interrupt on overflow	1 1 0 0 1 1 1 0	24 - m or 3 (3 if no interrupt)	24 - m or 3 (3 if no interrupt)	2	
Protected Mode Only:					
Via interrupt or trap gate to same privilege level			40 + m		8,11,12
Via interrupt or trap gate to fit different privilege level			78 + m		8,11,12
Via Task Gate			167 - m		8,11,12
IRET = Interrupt return	1 1 0 0 1 1 1 1	17 + m	31 + m	2,4	8,9,11,12,15
Protected Mode Only:					
To different privilege level			55 + m		8,9,11,12,15
To different task (NT = 1)			169 - m		8,9,11,12
BOUND = Detect value out of range	0 1 1 0 0 0 1 0 | mod reg r m	13*	13* (Use INT clock count if exception 5)	2,6	6,8,9,11,12

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright Intel Corp., 1983.

80286 INSTRUCTION SET SUMMARY (Continued)

FUNCTION	FORMAT	CLOCK COUNT: Real Address Mode	CLOCK COUNT: Protected Virtual Address Mode	COMMENTS: Real Address Mode	COMMENTS: Protected Virtual Address Mode
PROCESSOR CONTROL					
CLC = Clear carry	1 1 1 1 1 0 0 0	2	2		
CMC = Complement carry	1 1 1 1 0 1 0 1	2	2		
STC = Set carry	1 1 1 1 1 0 0 1	2	2		
CLD = Clear direction	1 1 1 1 1 1 0 0	2	2		
STD = Set direction	1 1 1 1 1 1 0 1	2	2		
CLI = Clear interrupt	1 1 1 1 1 0 1 0	3	3		14
STI = Set interrupt	1 1 1 1 1 0 1 1	2	2		14
HLT = Halt	1 1 1 1 0 1 0 0	2	2		13
WAIT = Wait	1 0 0 1 1 0 1 1	3	3		
LOCK = Bus lock prefix	1 1 1 1 0 0 0 0	0	0		14
CTS = Clear task switched flag	0 0 0 0 1 1 1 1 | 0 0 0 0 0 1 1 0	2	2	3	13
ESC = Processor Extension Escape (TTT LLL are opcode to processor extension)	1 0 0 1 1 T T T | mod LLL r/m	9-20*	9-20*	5	17
PROTECTION CONTROL					
LGDT = Load global descriptor table register	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 1 | mod 0 1 0 r/m	11*	11*	2,3	9,13
SGDT = Store global descriptor table register	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 1 | mod 0 0 0 r/m	11*	11*	2,3	9
LIDT = Load interrupt descriptor table register	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 1 | mod 0 1 1 r/m	12*	12*	2,3	9,13
SIDT = Store interrupt descriptor table register	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 1 | mod 0 0 1 r/m	12*	12*	2,3	9
LLDT = Load local descriptor table register from register memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 | mod 0 1 0 r/m		17,19*	1	9,11,13
SLDT = Store local descriptor table register to register/memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 | mod 0 0 0 r/m		2,3*	1	9
LTR = Load task register from register/memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 | mod 0 1 1 r/m		17,19*	1	9,11,13
STR = Store task register to register memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 | mod 0 0 1 r/m		2,3*	1	9,11,13
LMSW = Load machine status word from register/memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 1 | mod 1 1 0 r/m	3,6*	3,6*	2,3	9,13
SMSW = Store machine status word	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 1 | mod 1 0 0 r/m	2,3*	2,3*	2,3	9
LAR = Load access rights from register/memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 1 0 | mod reg r/m		14,16*	1	9,16
LSL = Load segment limit from register/memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 1 1 | mod reg r/m		14,16*	1	9,16
ARPL = Adjust requested privilege level: from register/memory	0 1 1 0 0 0 1 1 | mod reg r/m		10*,11*	2	9
VERR = Verify read access: register/memory	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 | mod 1 0 0 r/m		14,16*	1	9,16
VERR = Verify write access:	0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 | mod 1 0 1 r/m		14,16*	1	9,16

Shaded areas indicate instructions not available in iAPX 86, 88 microsystems.

All mnemonics copyright Intel Corp., 1983.

Footnotes

The effective Address (EA) of the memory operand is computed according to the mod and r/m fields:

if mod = 11 then r/m is treated as a REG field
if mod = 00 then DISP = 0*, disp-low and disp-high are absent
if mod = 01 then DISP = disp-low sign-extended to 16-bits, disp-high is absent
if mod = 10 then DISP = disp-high: disp-low

if r/m = 000 then EA = (BX) + (SI) + DISP
if r/m = 001 then EA = (BX) + (DI) + DISP
if r/m = 010 then EA = (BP) + (SI) + DISP
if r/m = 011 then EA = (BP) + (DI) + DISP
if r/m = 100 then EA = (SI) + DISP
if r/m = 101 then EA = (DI) + DISP
if r/m = 110 then EA = (BP) + DISP*
if r/m = 111 then EA = (BX) + DISP

DISP follows 2nd byte of instruction (before data if required)

*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

SEGMENT OVERRIDE PREFIX

0	0	1	reg	1	1	0

reg is assigned according to the following:

reg	Segment Register
00	ES
01	CS
10	SS
11	DS

REG is assigned according to the following table:

16-Bit (w = 1)	8-Bit (w = 0)
000 AX	000 AL
001 CX	001 CL
010 DX	010 DL
011 BX	011 BL
100 SP	100 AH
101 BP	101 CH
110 SI	110 DH
111 DI	111 BH

The physical addresses of all operands addressed by the BP register are computed using the SS segment register. The physical addresses of the destination operands of the string primitive operations (those addressed by the DI register) are computed using the ES segment, which may not be overridden.

8086/8086-1/8086-2

16-Bit Microprocessor
iAPX-86 Family

DISTINCTIVE CHARACTERISTICS

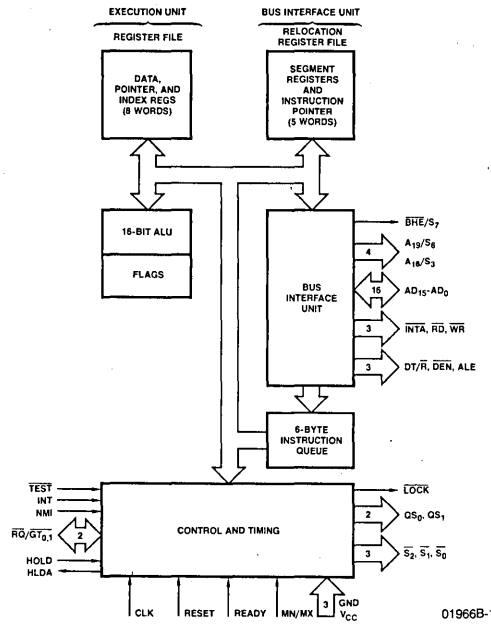
- Directly addresses up to 1 Mbyte of memory
- 24 operand addressing modes
- Efficient implementation of high level languages
- Instruction set compatible with 8080 software
- Bit, byte, word, and block operations
- 8 and 16-bit signed and unsigned arithmetic in binary or decimal
- Multibus* system interface
- Three speed options
  - 5MHz for 8086
  - 8MHz for 8086-2
  - 10MHz for 8086-1

GENERAL DESCRIPTION

The 8086 is a general purpose 16-bit microprocessor CPU. Its architecture is built around thirteen 16-bit registers and nine 1-bit flags. The CPU operates on 16-bit address spaces, and can directly address up to 1 megabyte using offset addresses within four distinct memory segments, designated as code, data, stack and extra code. The 8086 implements a powerful instruction set with 24 operand addressing modes. This instruction set is compatible with that of the 8080 and 8085. In addition, the 8086 is particularly effective in executing high level languages.

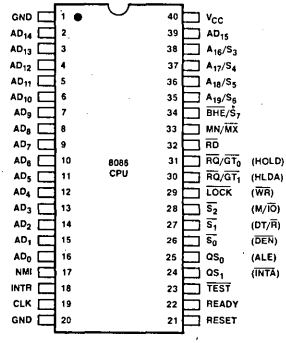
The 8086 can operate in minimum and maximum modes. Maximum mode offloads certain bus control functions to a peripheral device and allows the CPU to operate efficiently in a multi-processor system. The CPU and its high performance peripherals are Multibus* compatible. The 8086 is implemented in N-channel, depletion load, silicon gate technology and is contained in a 40-pin CerDIP or Molded DIP package.

Figure 1. Block Diagram



01966B-1

Figure 2. Pin Configuration
D-40-1, P-40-1



Note: Pin 1 is marked for orientation.

01966B-2

ORDERING INFORMATION

Package Type	Ambient Temperature	Clock Frequency		
		5MHz	8MHz	10MHz
Molded DIP Hermetic DIP	$0°C \leq T_A \leq 70°C$	P8086 D8086	P8086-2 D8086-2	P8086-1 D8086-1
Hermetic DIP	$-40°C \leq T_A \leq 85°C$	ID8086		
Hermetic DIP	$-55°C \leq T_A \leq +125°C$	MD8086B		

Multibus is a registered trademark of Intel Corp.

01966B-MMP

FUNCTIONAL ORGANIZATION

The 8086 CPU is internally organized into two processing units. These two units are the Bus Interface Unit (BIU) and the Execution Unit (EU). A block diagram of this organization is shown in Figure 1.

The BIU performs instruction fetch and queuing, operand fetch and store, address relocation, and basic bus control. The EU receives operands and instructions from the BIU and processes them on a 16-bit ALU. The EU accesses memory and peripheral devices through requests to the BIU. The BIU generates physical addresses in memory using the 4 segment registers and offset values.

The BIU and EU usually operate asynchronously. This permits the 8086 to overlap execution fetch and execution. Up to 6 instruction bytes can be queued. The instruction queue acts as a FIFO buffer for instructions, from which the EU extracts instruction bytes as required.

MEMORY ORGANIZATION

The 8086 addresses up to 1 megabyte of memory. The address space is organized as a linear array, from 00000 to FFFFF in hexadecimal. Memory is subdivided into segments of 64K bytes each. There are 4 segments: code, stack, data, and extra (usually employed as an extra data segment). Each segment thus contains information of a similar type. Selection of a destination segment is automatically performed using the rules in the table below. This segmentation makes memory more easily relocatable and supports a more structured programming style.

Physical addresses in memory are generated by selecting the appropriate segment, obtaining the segment "base" address from the segment register, shifting the base address 4 digits to the left, and then adding this base to the "offset" address. For programming code, the offset address is obtained from the instruction pointer. For operands, the offset address is calculated in several ways, depending upon information contained in the addressing mode. Memory organization and address generation are shown in Figure 3a.

Certain memory locations are reserved for specific CPU operations. These are shown in Figure 3b. Addresses FFFFOH through FFFFFH are reserved for operations which include a jump to the initial program loading routine. After RESET, the CPU will always begin execution at location FFFFOH, where the jump must be located.

Addresses 00000H through OO3FFH are reserved for interrupt operations. The service routine of each of the 256 possible interrupt types is signaled by a 4-byte pointer. The pointer elements must be stored in reserved memory addresses before the interrupts are invoked.

Figure 3a. Memory Organization

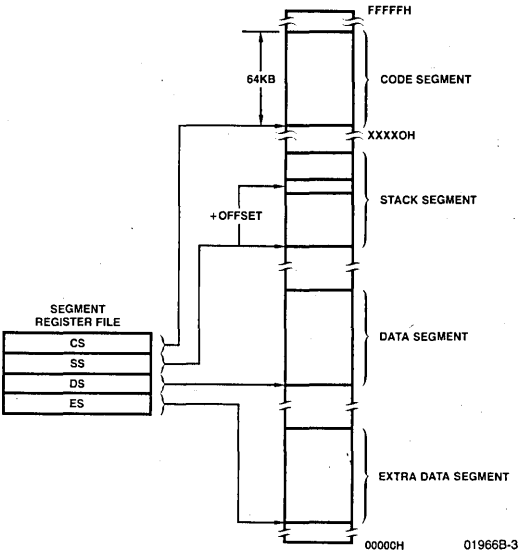
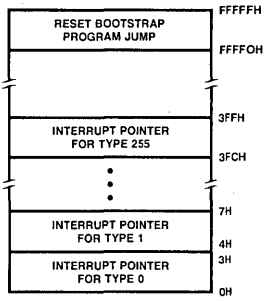


Figure 3b. Reserved Memory Locations



Memory Reference Need	Segment Register Used	Segment Selection Rule
Instructions	CODE (CS)	Automatic for all prefetching of instructions.
Stack	STACK (SS)	All stack pushes and pops, and all memory references relative to BP base register except data references.
Local Data	DATA (DS)	Data references which are relative to the stack, the destination of a string operation, or explicitly overriden.
External (Global) Data	EXTRA (ES)	Destination of string operations, when they are explicitly selected using a segment override.

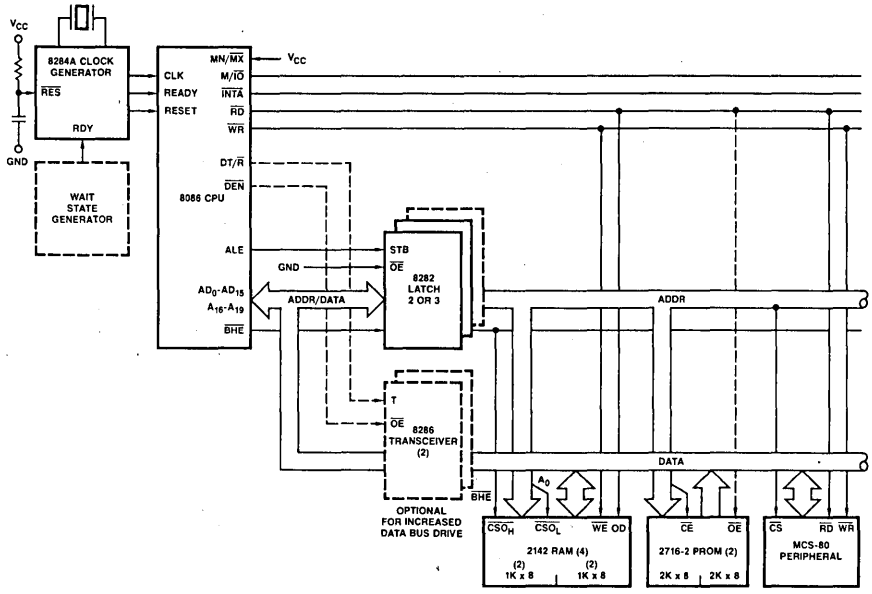
MINIMUM AND MAXIMUM MODES

The 8086 has two system configurations, minimum and maximum mode. The CPU has a strap pin, MN/$\overline{MX}$, which defines the system configuration. The status of this strap pin defines the function of pin numbers 24 through 31.

When MN/$\overline{MX}$ is strapped to GND, the 8086 operates in maximum mode. The operations of pins 24 through 31 are redefined. In maximum mode, several bus timing and control functions are "off-loaded" to the 8288 bus controller, thus freeing up the CPU. The CPU communicates status information to the 8288 through pins S_0, S_1, and S_2. In maximum mode, the 8086 can operate in a multiprocessor system, using the LOCK signal within a Multibus format.

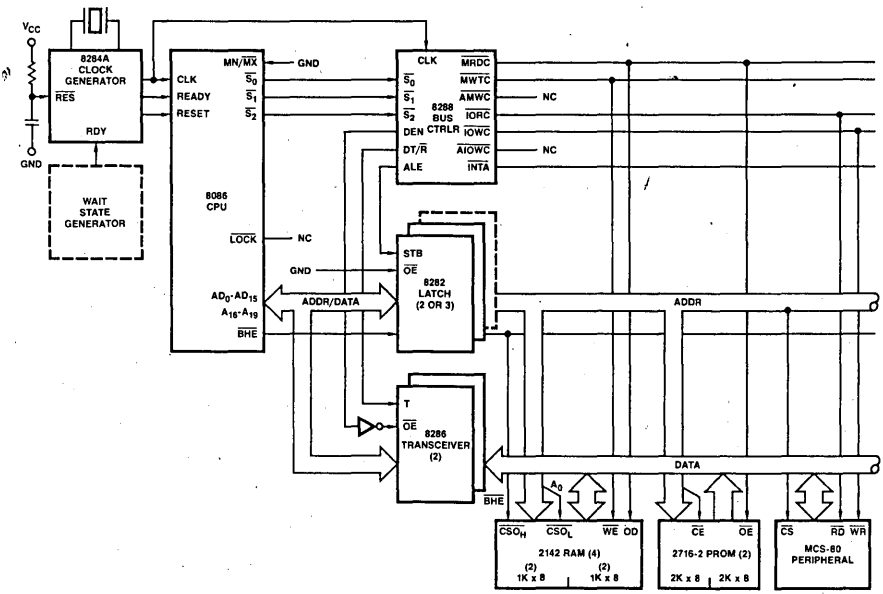
When MN/$\overline{MX}$ is strapped to V_{CC}, the 8086 operates in minimum mode. The CPU sends bus control signals itself through pins 24 through 31. This is shown in Figure 2 (in parentheses). Examples of minimum and maximum mode systems are shown in Figure 4.

Figure 4a. Minimum Mode 8086 Typical Configuration



01966B-5

Figure 4b. Maximum Mode 8086 Typical Configuration



01966B-6

BUS OPERATION

The 8086 has a combined address and data bus, commonly referred to as "a time multiplexed bus." This technique provides the most efficient use of pins on the processor while permitting the use of a standard 40-lead package. This bus can be used throughout the system with address latching provided on memory and I/O modules. The bus can also be demultiplexed at the processor with a single set of address latches if a standard non-multiplexed bus is desired for the system.

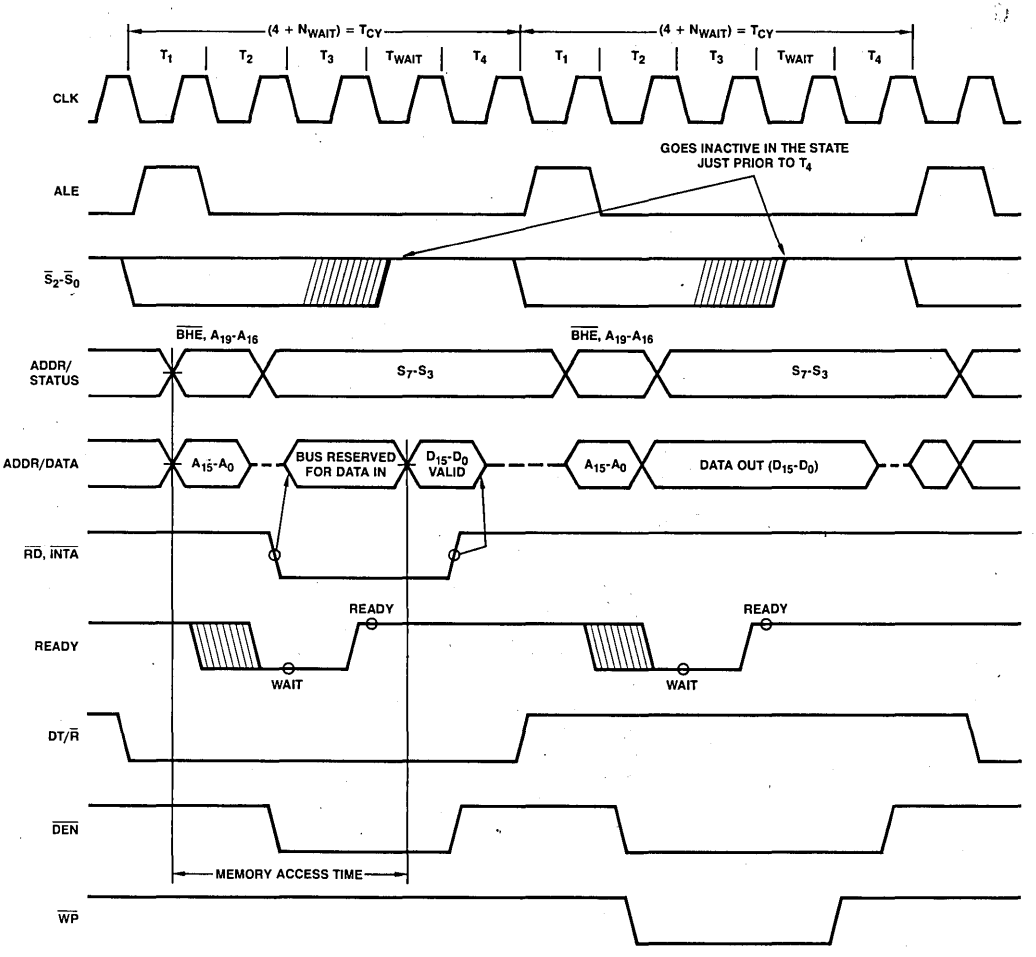
Each bus cycle consists of at least four CLK cycles. These are referred to as T_1, T_2, T_3 and T_4 (see Figure 5). The address is sent from the processor during T_1. Data transfer occurs on the bus during T_3 and T_4. T_2 is used for changing the direction of the bus during read operations. In the event that a "NOT READY" indication is given by the addressed device, "Wait" states (T_W) are inserted between T_3 and T_4. Each inserted "Wait" state is of the same duration as a CLK cycle. "Idle" states (T_I) or inactive CLK cycles can occur between 8086 bus cycles. The processor uses these cycles for internal housekeeping.

During T_1 of any bus cycle the ALE (Address Latch Enable) signal is emitted (by either the processor or the 8288 bus controller, depending on the MN/MX strap). At the trailing edge of this pulse, a valid address and certain status information for the cycle may be latched.

I/O ADDRESSING

8086 I/O operations can address up to a maximum of 64K I/O byte registers or 32K I/O word registers. The I/O address appears in the same format as the memory address on bus lines A_{15}-A_0. The address lines A_{19}-A_{16} are zero in I/O operations. I/O instructions which use register DX as a pointer have full address capability. Direct I/O instructions directly address one or two of the 256 I/O byte locations in page 0 of the I/O address space. I/O ports are addressed in the same manner as memory locations.

Figure 5. Basic System Timing



01966B-7

EXTERNAL INTERFACE

PROCESSOR RESET AND INITIALIZATION

Processor initialization or start up is accomplished with activation (HIGH) of the RESET pin. The 8086 RESET is required to be HIGH for greater than 4 CLK cycles. The 8086 will terminate operations on the high-going edge of RESET and will remain dormant as long as RESET is HIGH. The low-going transition of RESET triggers an internal reset sequence for approximately 10 CLK cycles. After this interval the 8086 operates normally beginning with the instruction in absolute location FFFF0H (see Figure 3B). The details of this operation are explained in the Instruction Set description of the MCS-86 Family User's Manual. The RESET input is internally synchronized to the processor clock. At initialization the HIGH-to-LOW transition of RESET must occur no sooner than 50μs after power-up, to allow complete initialization of the 8086.

NMI may not be asserted prior to the 2nd CLK cycle following the end of RESET.

INTERRUPT OPERATIONS

Interrupt operations fall into two classes; software or hardware initiated. The software initiated interrupts and software aspects of hardware interrupts are described in the Instruction Set description. Hardware interrupts are either non-maskable or maskable.

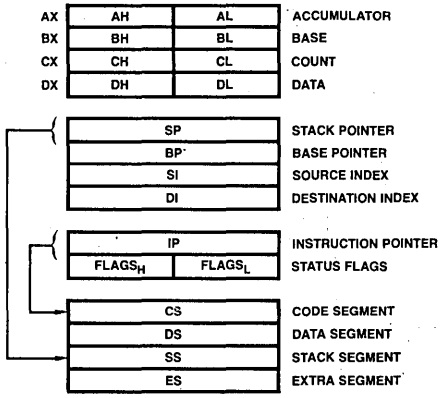
Interrupts transfer control to a new program location. A 256-element table containing address pointers to the interrupt service program locations resides in absolute locations 0 through 3FFH (see Figure 3b), which are reserved for this purpose. Each element in the table is 4 bytes in size and corresponds to an interrupt "type". An interrupting device supplies an 8-bit type number, during the interrupt acknowledge sequence, which is used to "vector" through the appropriate element to the new interrupt service program location.

BASIC SYSTEM TIMING

Typical system configurations for the processor operating in minimum mode and in maximum mode are shown in Figures 4a and 4b, respectively. In minimum mode, the processor emits bus control signals in a manner similar to the 8085. In maximum mode, the processor emits coded status information which the 8288 bus controller uses to generate MULTIBUS compatible bus control signals. Figure 5 illustrates the signal timing relationships.

$\overline{BHE}$	A_0	Characteristics
0	0	Whole word
0	1	Upper byte from/to odd address
1	0	Lower byte from/to even address
1	1	None

Figure 6. 8086 Register Model



01966B-8

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias	0 to 70°C
Storage Temperature	−65 to +150°C
Voltage on Any Pin with Respect to Ground	−1.0 to +7V
Power Dissipation	2.5 Watt

*NOTICE: Stresses above those listed under "Absolute Maxium Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS

(8086: T_A = 0 to 70°C, T_A = −40 to 85°C, T_A = −55 to 125°C, V_{CC} = 5V ± 10%)
(8086-1: T_A = 0 to 70°C, V_{CC} = 5V ± 5%)
(8086-2: T_A = 0 to 70°C, V_{CC} = 5V ± 5%)

Parameter	Description	Test Conditions	Min	Max	Units
V_{IL}	Input Low Voltage		−0.5	+0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC} +0.5	V
V_{OL}	Output Low Voltage	I_{OL} = 2.5 mA		0.45	V
V_{OH}	Output High Voltage	I_{OH} = −400μA	2.4		V
I_{CC}	Power Supply Current: 8086 8086-1 8086-2	T_A = 25°C		340 360 350	mA
I_{LI}	Input Leakage Current	0V ≤ V_{IN} ≤ V_{CC}		± 10	μA
I_{LO}	Output Leakage Current	0.45V ≤ V_{OUT} ≤ V_{CC}		± 10	μA
V_{CL}	Clock Input Low Voltage		−0.5	+0.6	V
V_{CH}	Clock Input High Voltage		3.9	V_{CC} + 1.0	V
C_{IN}	Capacitance of Input Buffer (All input except AD_0-AD_{15}, $\overline{RQ}/\overline{GT}$)	fc = 1 MHz		15	pF
C_{IO}	Capacitance of I/O Buffer (AD_0-AD_{15}, $\overline{RQ}/\overline{GT}$)	fc = 1 MHz		15	pF

AC CHARACTERISTICS

(8086: T_A = 0 to 70°C, V_{CC} = 5V ± 10%)
(8086-1: T_A = 0 to 70°C, V_{CC} = 5V ± 5%)
(8086-2: T_A = 0 to 70°C, V_{CC} = 5V ± 5%)

MINIMUM COMPLEXITY SYSTEM TIMING REQUIREMENTS

Parameter	Description	Test Conditions	8086		8086-1 (Preliminary)		8086-2		Units
			Min	Max	Min	Max	Min	Max	
TCLCL	CLK Cycle Period		200	500	100	500	125	500	ns
TCLCH	CLK Low Time		(2/3 TCLCL) −15		(2/3 TCLCL) −14		(2/3 TCLCL) −15		ns
TCHCL	CLK High Time		(1/3 TCLCL) +2		(1/3 TCLCL) +6		(1/3 TCLCL) +2		ns
TCH1CH2	CLK Rise Time	From 1.0 to 3.5V		10		10		10	ns
TCL2CL1	CLK Fall Time	From 3.5 to 1.0V		10		10		10	ns
TDVCL	Data in Setup Time		30		5		20		ns
TCLDX	Data in Hold Time		10		10		10		ns
TR1VCL	RDY Setup Time into 8284A (See Notes 1, 2)		35		35		35		ns
TCLR1X	RDY Hold Time into 8284A (See Notes 1, 2)		0		0		0		ns
TRYHCH	READY Setup Time into 8086		(2/3 TCLCL) −15		53		(2/3 TCLCL) −15		ns
TCHRYX	READY Hold Time into 8086		30		20		20		ns
TRYLCL	READY Inactive to CLK (See Note 3)		−8		−10		−8		ns
THVCH	HOLD Setup Time		35		20		20		ns
TINVCH	INTR, NMI, $\overline{\text{TEST}}$ Setup Time (See Note 2)		30		15		15		ns
TILIH	Input Rise Time (Except CLK)	From 0.8 to 2.0V		20		20		20	ns
TIHIL	Input Fall Time (Except CLK)	From 2.0 to 0.8V		12		12		12	ns

AC CHARACTERISTICS (Cont.)

TIMING RESPONSES

Parameter	Description	Test Conditions	8086		8086-1 (Preliminary)		8086-2		Units
			Min	Max	Min	Max	Min	Max	
TCLAV	Address Valid Delay	*C_L = 20-100pF for all 8086 Outputs (In addition to 8086 self load)	10	110	10	50	10	60	ns
TCLAX	Address Hold Time		10		10		10		ns
TCLAZ	Address Float Delay		TCLAX	80	10	40	TCLAX	50	ns
TLHLL	ALE Width		TCLCH −20		TCLCH −10		TCLCH −10		ns
TCLLH	ALE Active Delay			80		40		50	ns
TCHLL	ALE Inactive Delay			85		45		55	ns
TLLAX	Address Hold Time to ALE Inactive		TCHCL −10		TCHCL −10		TCHCL −10		ns
TCLDV	Data Valid Delay		10	110	10	50	10	60	ns
TCHDX	Data Hold Time		10		10		10		ns
TWHDX	Data Hold Time After WR		TCLCH −30		TCLCH −25		TCLCH −30		ns
TCVCTV	Control Active Delay 1		10	110	10	50	10	70	ns
TCHCTV	Control Active Delay 2		10	110	10	45	10	60	ns
TCVCTX	Control Inactive Delay		10	110	10	50	10	70	ns
TAZRL	Address Float to READ Active		0		0		0		ns
TCLRL	$\overline{RD}$ Active Delay		10	165	10	70	10	100	ns
TCLRH	$\overline{RD}$ Inactive Delay		10	150	10	60	10	80	ns
TRHAV	$\overline{RD}$ Inactive to Next Address Active		TCLCL −45		TCLCL −35		TCLCL −40		ns
TCLHAV	HLDA Valid Delay		10	160	10	60	10	100	ns
TRLRH	$\overline{RD}$ Width		2TCLCL −75		2TCLCL −40		2TCLCL −50		ns
TWLWH	$\overline{WR}$ Width		2TCLCL −60		2TCLCL −35		2TCLCL −40		ns
TAVAL	Address Valid to ALE Low		TCLCH −60		TCLCH −35		TCLCH −40		ns
TOLOH	Output Rise Time	From 0.8 to 2.0V		20		20		20	ns
TOHOL	Output Fall Time	From 2.0 to 0.8V		12		12		12	ns

Notes: 1. Signal at 8284A shown for reference only.
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.
3. Applies only to T2 state (8 ns into T3).

AC CHARACTERISTICS (Cont.)

TIMING RESPONSES

Parameter	Description	Test Conditions	8086		8086-1 (Preliminary)		8086-2		Units
			Min	Max	Min	Max	Min	Max	
TCLML	Command Active Delay (See Note 1)		10	35	10	35	10	35	ns
TCLMH	Command Inactive Delay (See Note 1)		10	35	10	35	10	35	ns
TRYHSH	READY Active to Status Passive (See Note 3)			110		45		65	ns
TCHSV	Status Active Delay		10	110	10	45	10	60	ns
TCLSH	Status Inactive Delay		10	130	10	55	10	70	ns
TCLAV	Address Valid Delay		10	110	10	50	10	60	ns
TCLAX	Address Hold Time		10		10		10		ns
TCLAZ	Address Float Delay		TCLAX	80	10	40	TCLAX	50	ns
TSVLH	Status Valid to ALE High (See Note 1)			15		15		15	ns
TSVMCH	Status Valid to MCE High (See Note 1)			15		15		15	ns
TCLLH	CLK Low to ALE Valid (See Note 1)			15		15		15	ns
TCLMCH	CLK Low to MCE High (See Note 1)			15		15		15	ns
TCHLL	ALE Inactive Delay (See Note 1)	C_L = 20-100pF for all 8086 Outputs (In addition to 8086 self load)		15		15		15	ns
TCLMCL	MCE Inactive Delay (See Note 1)			15		15		15	ns
TCLDV	Data Valid Delay		10	110	10	50	10	60	ns
TCHDX	Data Hold Time		10		10		10		ns
TCVNV	Control Active Delay (See Note 1)		5	45	5	45	5	45	ns
TCVNX	Control Inactive Delay (See Note 1)		10	45	10	45	10	45	ns
TAZRL	Address Float to Read Active		0		0		0		ns
TCLRL	$\overline{RD}$ Active Delay		10	165	10	70	10	100	ns
TCLRH	$\overline{RD}$ Inactive Delay		10	150	10	60	10	80	ns
TRHAV	$\overline{RD}$ Inactive to Next Address Active		TCLCL −45		TCLCL −35		TCLCL −40		ns
TCHDTL	Direction Control Active Delay (See Note 1)			50		50		50	ns
TCHDTH	Direction Control Inactive Delay (See Note 1)			30		30		30	ns
TCLGL	GT Active Delay		0	85	0	45	0	50	ns
TCLGH	GT Inactive Delay		0	85	0	45	0	50	ns
TRLRH	$\overline{RD}$ Width		2TCLCL −75		2TCLCL −40		2TCLCL −50		ns
TOLOH	Output Rise Time	From 0.8 to 2.0V		20		20		20	ns
TOHOL	Output Fall Time	From 2.0 to 0.8V		12		12		12	ns

AC CHARACTERISTICS (Cont.)

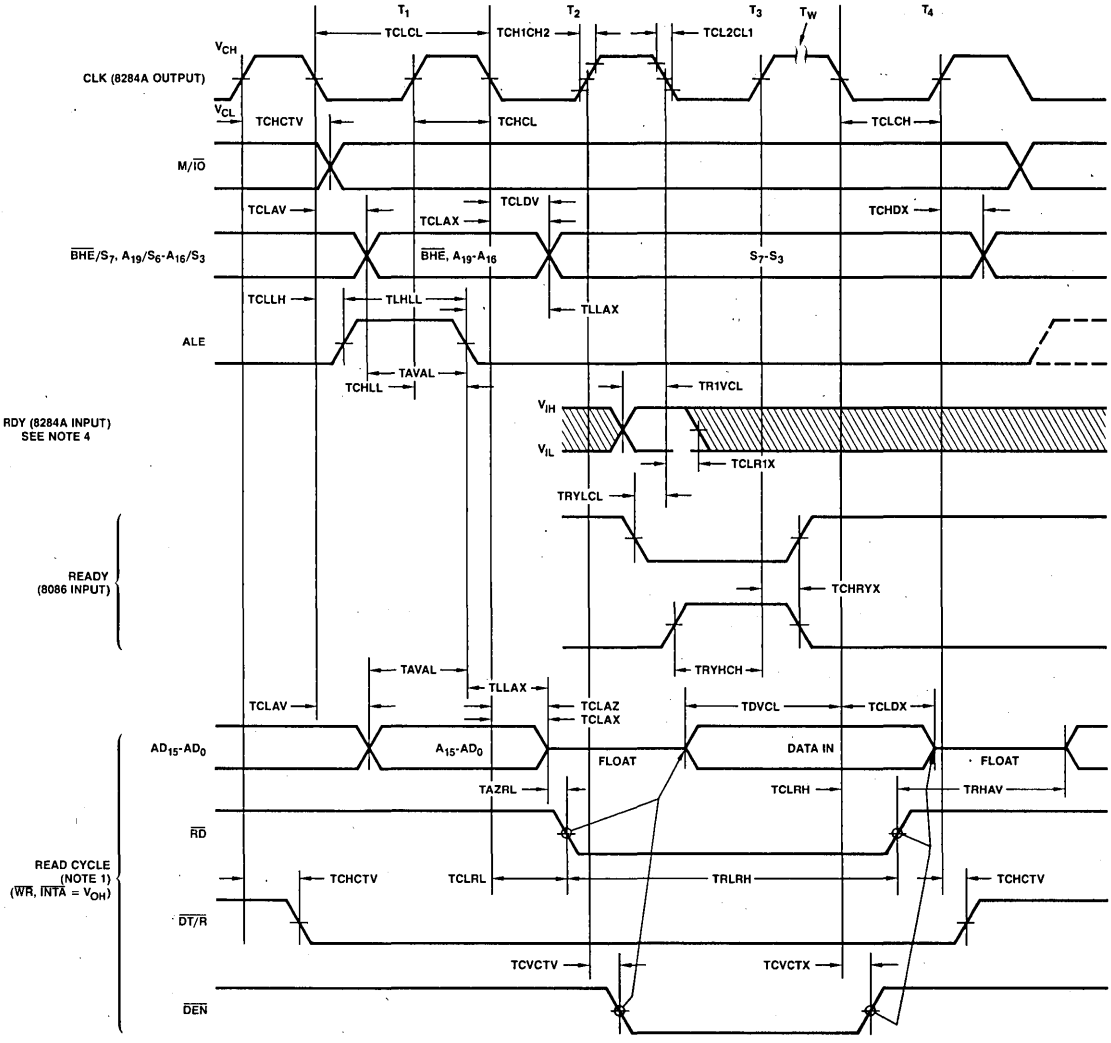
MAX MODE SYSTEM (USING 8288 BUS CONTROLLER) TIMING REQUIREMENTS

Parameter	Description	Test Conditions	8086		8086-1 (Preliminary)		8086-2		Units
			Min	Max	Min	Max	Min	Max	
TCLCL	CLK Cycle Period		200	500	100	500	125	500	ns
TCLCH	CLK Low Time		(2/3 TCLCL) −15		(2/3 TCLCL) −14		(2/3 TCLCL) −15		ns
TCHCL	CLK High Time		(1/3 TCLCL) +2		(1/3 TCLCL) +6		(1/3 TCLCL) +2		ns
TCH1CH2	CLK Rise Time	From 1.0 to 3.5V		10		10		10	ns
TCL2CL1	CLK Fall Time	From 3.5 to 1.0V		10		10		10	ns
TDVCL	Data in Setup Time		30		5		20		ns
TCLDX	Data in Hold Time		10		10		10		ns
TR1VCL	RDY Setup Time into 8284A (See Notes 1, 2)		35		35		35		ns
TCLR1X	RDY Hold Time into 8284A (See Notes 1, 2)		0		0		0		ns
TRYHCH	READY Setup Time into 8086		(2/3 TCLCL) −15		53		(2/3 TCLCL) −15		ns
TCHRYX	READY Hold Time into 8086		30		20		20		ns
TRYLCL	READY Inactive to CLK (See Note 4)		−8		−10		−8		ns
TINVCH	Setup Time for Recognition (INTR, NMI, $\overline{\text{TEST}}$ (See Note 2)		30		15		15		ns
TGVCH	$\overline{\text{RQ}}/\overline{\text{GT}}$ Setup Time		30		12		15		ns
TCHGX	$\overline{\text{RQ}}$ Hold Time into 8086		40		20		30		ns
TILIH	Input Rise Time (Except CLK)	From 0.8 to 2.0V		20		20		20	ns
TIHIL	Input Fall Time (Except CLK)	From 2.0 to 0.8V		12		12		12	ns

Notes: 1. Signal at 8284A or 8288 shown for reference only.
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.
3. Applies only to T3 and wait states.
4. Applies only to T2 state (8ns into T3).

WAVEFORMS

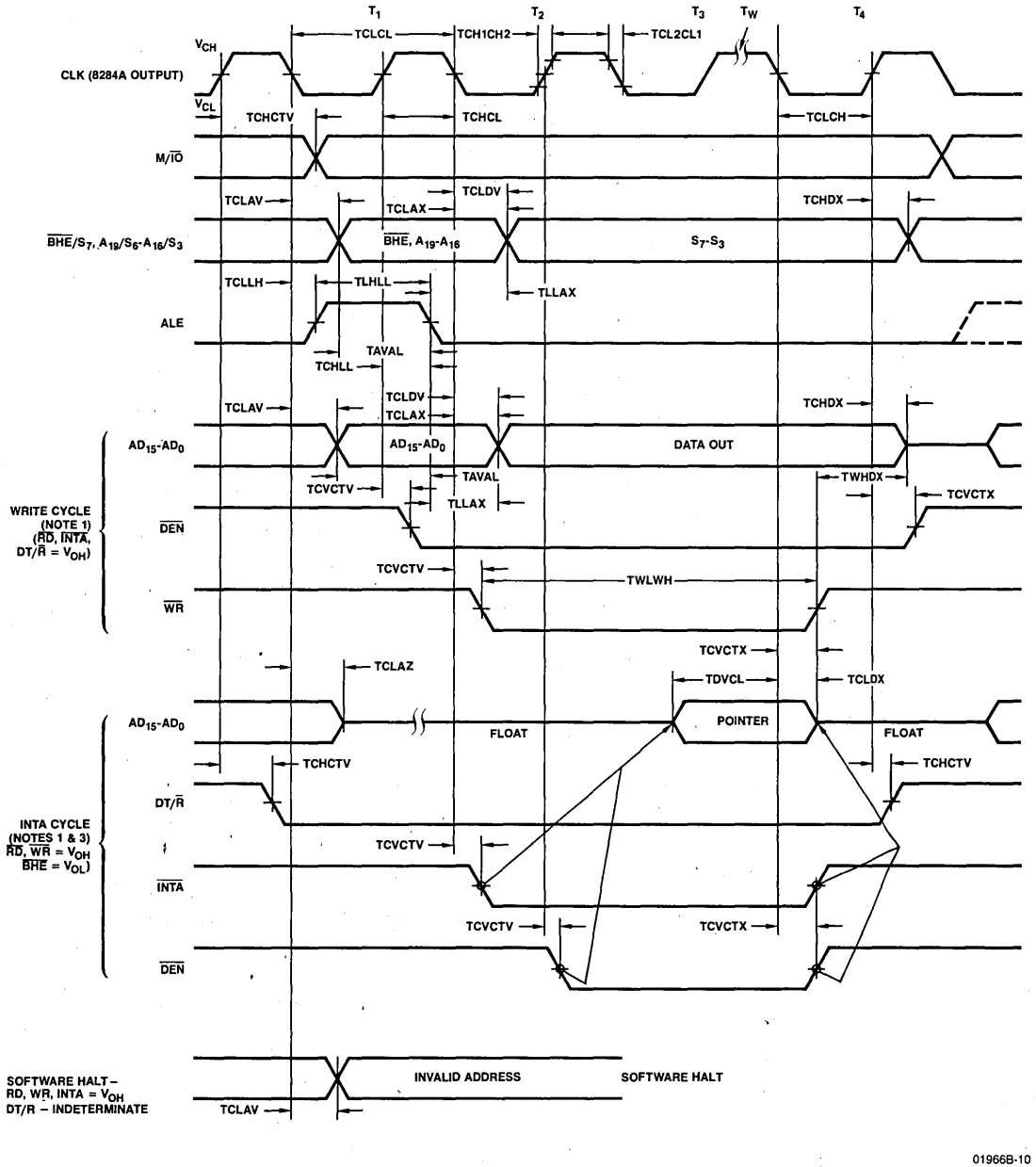
MINIMUM MODE



01966B-9

WAVEFORMS (Cont.)

MINIMUM MODE

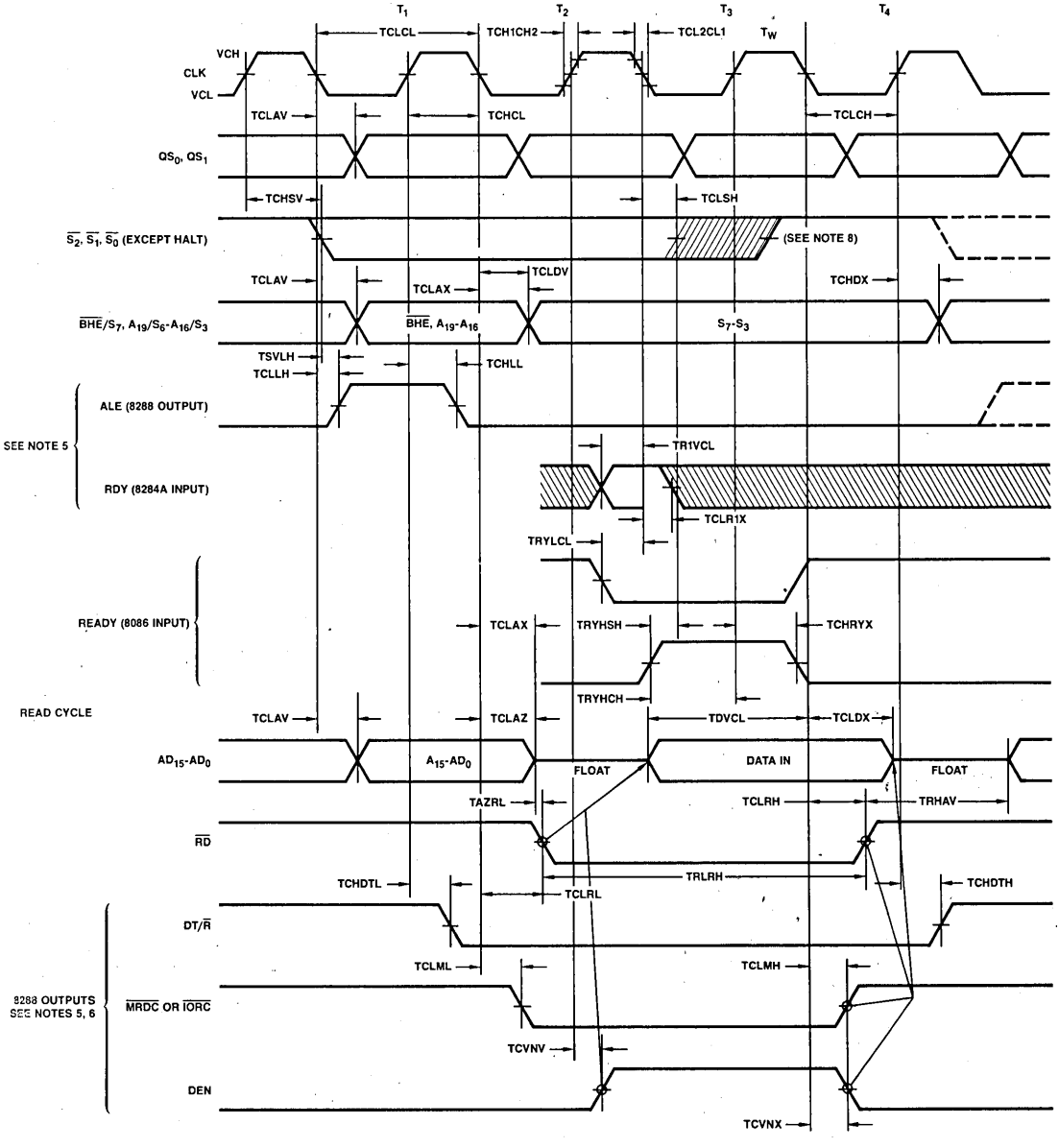


01966B-10

Notes:
1. All signals switch between V_{OH} and V_{OL} unless otherwise specified.
2. RDY is sampled near the end of T_2, T_3, T_W to determine if T_W machines states are to be inserted.
3. Two INTA cycles run back-to-back. The 8086 LOCAL ADDR/DATA BUS is floating during both INTA cycles. Control signals shown for second INTA cycle.
4. Signals at 8284A are shown for reference only.
5. All timing measurements are made at 1.5V unless otherwise noted.

WAVEFORMS (Cont.)

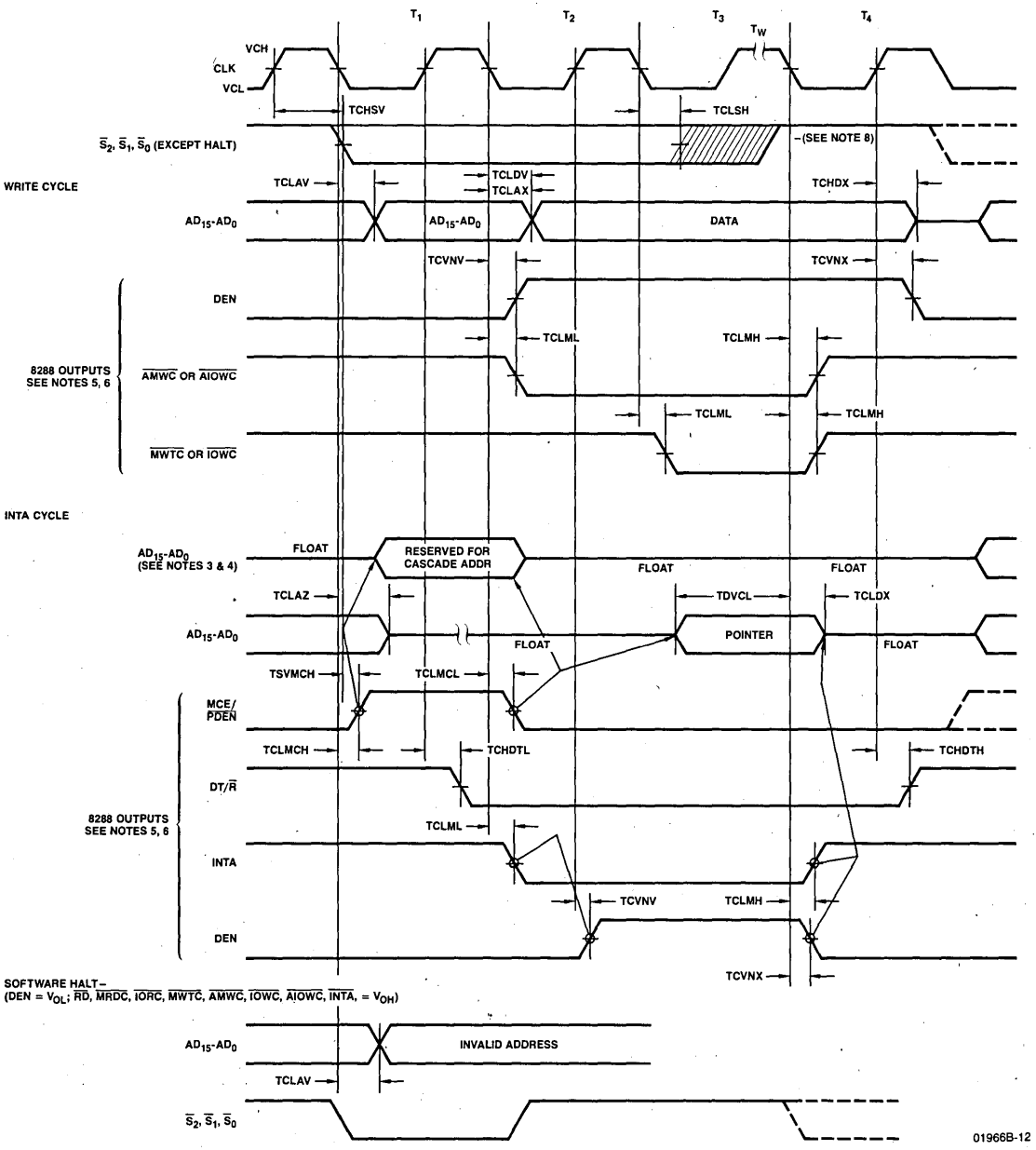
MAXIMUM MODE



01966B-11

WAVEFORMS (Cont.)

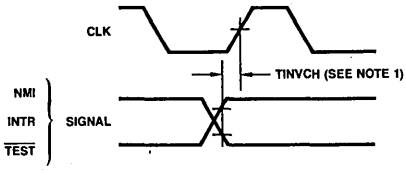
MAXIMUM MODE (Cont.)



Notes:
1. All signals switch between V_{OH} and V_{OL} unless otherwise specified.
2. RDY is sampled near the end of T_2, T_3, T_W to determine if T_W machines states are to be inserted.
3. Cascade address is valid between first and second INTA cycle.
4. Two INTA cycles run back-to-back. The 8086 LOCAL ADDR/DATA BUS is floating during both INTA cycles. Control for pointer address is shown for second INTA cycle.
5. Signals at 8284A or 8288 are shown for reference only.
6. The issuance of the 8288 command and control signals ($\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$, $\overline{IORC}$, $\overline{IOWC}$, $\overline{AIOWC}$, $\overline{INTA}$ and DEN) lags the active high 8288 CEN.
7. All timing measurements are made at 1.5V unless otherwise noted.
8. Status inactive in state just prior to T_4.

WAVEFORMS (Cont.)

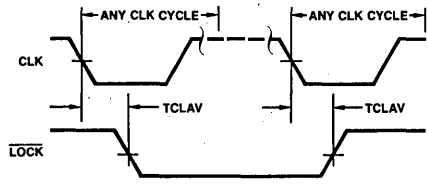
ASYNCHRONOUS SIGNAL RECOGNITION



Note: Setup Requirements for Asychronous signals only to guarantee recognition at next CLK.

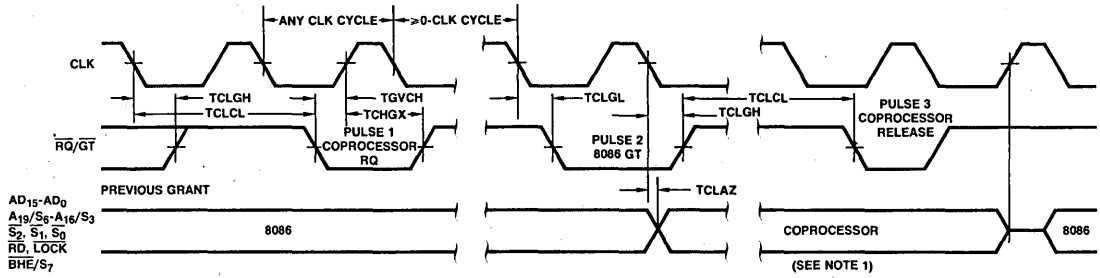
01966B-13

BUS LOCK SIGNAL TIMING (MAXIMUM MODE ONLY)



01966B-14

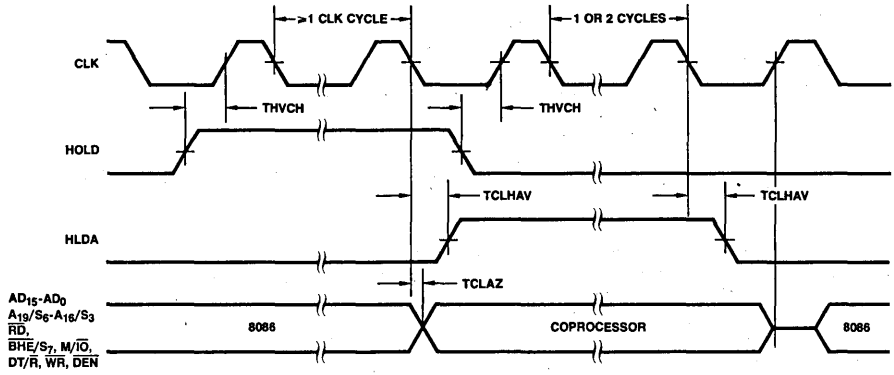
REQUEST/GRANT SEQUENCE TIMING (MAXIMUM MODE ONLY)



Notes: The Coprocessor may not drive the buses outside the region shown without risking contention.

01966B-15

HOLD/HOLD ACKNOWLEDGE TIMING (MINIMUM MODE ONLY)



01966B-16

TABLE 1. PIN DESCRIPTION

The following pin function descriptions are for 8086 systems in either minimum or maximum mode. The "Local Bus" in these descriptions is the direct multiplexed bus interface connection to the 8086 (without regard to additional bus buffers).

Symbol	Pin No.	Type	Name and Function
AD_{15}-AD_0	2-16, 39	I/O	**Address Data Bus:** These lines constitute the time multiplexed memory/IO address (T_1) and data (T_2, T_3, T_W, T_4) bus. A_0 is analogous to $\overline{BHE}$ for the lower byte of the data bus, pins D_7-D_0. It is LOW during T_1 when a byte is to be transferred on the lower portion of the bus in memory or I/O operations. Eight-bit oriented devices tied to the lower half would normally use A_0 to condition chip select functions. (See $\overline{BHE}$.) These lines are active HIGH and float to 3-state OFF during interrupt acknowledge and local bus "hold acknowledge."
A_{19}/S_6, A_{18}/S_5, A_{17}/S_4, A_{16}/S_3	35-38	O	**Address/Status:** During T_1 these are the four most significant address lines for memory operations. During I/O operations these lines are LOW. During memory and I/O operations, status information is available on these lines during T_2, T_3, T_W, and T_4. The status of the interrupt enable FLAG bit (S_5) is updated at the beginning of each CLK cycle. A_{17}/S_4 and A_{16}/S_3 are encoded as shown. This information indicates which relocation register is presently being used for data accessing. These lines float to 3-state OFF during local bus "hold acknowledge."
$\overline{BHE}/S_7$	34	O	**Bus High Enable/Status:** During T_1 the bus high enable signal ($\overline{BHE}$) should be used to enable data onto the most significant half of the data bus, pins D_{15}-D_8. Eight-bit oriented devices tied to the upper half of the bus would normally use $\overline{BHE}$ to condition chip select functions. $\overline{BHE}$ is LOW during T_1 for read, write, and interrupt acknowledge cycles when a byte is to be transferred on the high portion of the bus. The S_7 status information is available during T_2, T_3, and T_4. The signal is active LOW, and floats to 3-state OFF in "hold." It is LOW during T_1 for the first interrupt acknowledge cycle.
$\overline{RD}$	32	O	**Read:** Read strobe indicates that the processor is performing a memory of I/O read cycle, depending on the state of the S_2 pin. This signal is used to read devices which reside on the 8086 local bus. $\overline{RD}$ is active LOW during T_2, T_3 and T_W of any read cycle, and is guaranteed to remain HIGH in T_2 until the 8086 local bus has floated. This signal floats to 3-state OFF in "hold acknowledge."
READY	22	I	**READY:** Is the acknowledgment from the addressed memory or I/O device that it will complete the data transfer. The READY signal from memory/IO is synchronized by the 8284A Clock Generator to form READY. This signal is active HIGH. The 8086 READY input is not synchronized. Correct operation is not guaranteed if the setup and hold times are not met.
INTR	18	I	**Interrupt Request:** Is a level triggered input which is sampled during the last clock cycle of each instruction to determine if the processor should enter into an interrupt acknowledge operation. A subroutine is vectored to via an interrupt vector lookup table located in system memory. It can be internally masked by software resetting the interrupt enable bit. INTR is internally synchronized. This signal is active HIGH.
$\overline{TEST}$	23	I	**$\overline{TEST}$:** Input is examined by the "Wait" instruction. If the $\overline{TEST}$ input is LOW execution continues, otherwise the processor waits in an "Idle" state. This input is synchronized internally during each clock cycle on the leading edge of CLK.
NMI	17	I	**Non-maskable Interrupt:** an edge triggered input which causes a type 2 interrupt. A subroutine is vectored to via an interrupt vector lookup table located in system memory. NMI is not maskable internally by software. A transition from a LOW to HIGH initiates the interrupt at the end of the current instruction. This input is internally synchronized.
RESET	21	I	**Reset:** causes the processor to immediately terminate its present activity. The signal must be active HIGH for at least four clock cycles. It restarts execution, as described in the Instruction Set description, when RESET returns LOW. RESET is internally synchronized.
CLK	19	I	**Clock:** provides the basic timing for the processor and bus controller. It is asymmetric with a 33% duty cycle to provide optimized internal timing.
V_{CC}	40		**V_{CC}:** +5V power supply pin.
GND	1, 20		**Ground**
MN/$\overline{MX}$	33	I	**Minimum/Maximum:** indicates what mode the processor is to operate in. The two modes are discussed in the following sections.

A_{17}/S_4	A_{16}-S_3	Characteristics
0 (LOW)	0	Alternate Data
0	1	Stack
1 (HIGH)	0	Code or None
1	1	Data
S_6 is 0 (LOW)		

$\overline{BHE}$	A_0	Characteristics
0	0	Whole word
0	1	Upper byte from/ to odd address
1	0	Lower byte from/ to even address
1	1	None

TABLE 1. PIN DESCRIPTION (Cont.)

The following pin function descriptions are for the 8086/8288 system in maximum mode (i.e., MN/$\overline{MX}$ = V_{SS}). Only the pin functions which are unique to maximum mode are described; all other pin functions are as described above.

Symbol	Pin No.	Type	Name and Function
$\overline{S_2}$, $\overline{S_1}$, $\overline{S_0}$	26-28	O	**Status:** active during T_4, T_1, and T_2 and is returned to the passive state (1, 1, 1) during T_3 or during T_W when READY is HIGH. This status is used by the 8288 Bus Controller to generate all memory and I/O access control signals. Any change by S_2, S_1, or S_0 during T_4 is used to indicate the beginning of a bus cycle, and the return to the passive state in T_3 or T_W is used to indicate the end of a bus cycle. These signals float to 3-state OFF in "hold acknowledge." These status lines are encoded as shown.
$\overline{RQ}/\overline{GT_0}$, $\overline{RQ}/\overline{GT_1}$	30, 31	I/O	**Request/Grant:** pins are used by other local bus masters to force the processor to release the local bus at the end of the processor's current bus cycle. Each pin is bidirectional with $\overline{RQ}/\overline{GT_0}$ having higher priority than $\overline{RQ}/\overline{GT_1}$. $\overline{RQ}/\overline{GT}$ has an internal pull-up resistor so may be left unconnected. The request/grant sequence is as follows (see Figure 9): 1. A pulse of 1 CLK wide from another local bus master indicates a local bus request ("hold") to the 8086 (pulse 1). 2. During a T_4 or T_1 clock cycle, a pulse 1 CLK wide from the 8086 to the requesting master (pulse 2), indicates that the 8086 has allowed the local bus to float and that it will enter the "hold acknowledge" state at the next CLK. The CPU's bus interface unit is disconnected logically from the local bus during "hold acknowledge." 3. A pulse 1 CLK wide from the requesting master indicates to the 8086 (pulse 3) that the "hold" request is about to end and that the 8086 can reclaim the local bus at the next CLK. Each master-master exchange of the local bus is a sequence of 3 pulses. There must be one dead CLK cycle after each bus exchange. Pulses are active LOW. If the request is made while the CPU is performing a memory cycle, it will release the local bus during T_4 of the cycle when all the following conditions are met: 1. Request occurs on or before T_2. 2. Current cycle is not the low byte of a word (on an odd address). 3. Current cycle is not the first acknowledge of an interrupt acknowledge sequence. 4. A locked instruction is not currently executing. If the local bus is idle when the request is made the two possible events will follow: 1. Local bus will be released during the next clock. 2. A memory cycle will start within 3 clocks. Now the four rules for a currently active memory cycle apply with condition number 1 already satisfied.
$\overline{LOCK}$	29	O	**$\overline{LOCK}$:** output indicates that other system bus masters are not to gain control of the system bus while $\overline{LOCK}$ is active LOW. The $\overline{LOCK}$ signal is activated by the "LOCK" prefix instruction and remains active until the completion of the next instruction. This signal is active LOW, and floats to 3-state OFF in "hold acknowledge."
QS_1, QS_0	24, 25	O	**Queue Status:** The queue status is valid during the CLK cycle after which the queue operation is performed. QS_1 and QS_0 provide status to allow external tracking of the internal 8086 instruction queue.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Characteristics
0 (LOW)	0	0	Interrupt Acknowledge
0	0	1	Read I/O Port
0	1	0	Write I/O Port
0	1	1	Halt
1 (HIGH)	0	0	Code Access
1	0	1	Read Memory
1	1	0	Write Memory
1	1	1	Passive

TABLE 1. PIN DESCRIPTION (Cont.)

The following pin function descriptions are for the 8086 in minimum mode (i.e., MN/MX = V_{CC}). Only the pin functions which are unique to minimum mode are described; all other pin functions are as described above.

Symbol	Pin No.	Type	Name and Function
M/$\overline{IO}$	28	O	**Status line:** logically equivalent to S_2 in the maximum mode. It is used to distinguish a memory access from an I/O access. M/IO becomes valid in the T_4 preceding a bus cycle and remains valid until the final T_4 of the cycle (M = HIGH, IO = LOW). M/$\overline{IO}$ floats to 3-state OFF in local bus "hold acknowledge."
$\overline{WR}$	29	O	**Write:** indicates that the processor is performing a write memory or write I/O cycle, depending on the state of the M/IO signal. WR is active for T_2, T_3 and T_W of any write cycle. It is active LOW, and floats to 3-state OFF in local bus "hold acknowledge."
$\overline{INTA}$	24	O	**$\overline{INTA}$:** is used as a read strobe for interrupt acknowledge cycles. It is active LOW during T_2, T_3 and T_W of each interrupt acknowledge cycle.
ALE	25	O	**Address Latch Enable:** provided by the processor to latch the address into 8282/8283 address latch. It is a HIGH pulse active during T_1 of any bus cycle. Note that ALE is never floated.
DT/$\overline{R}$	27	O	**Data Transmit/Receive:** needed in minimum system that desires to use an 8286/8287 data bus transceiver. It is used to control the direction of data flow through the transceiver. Logically DT/$\overline{R}$ is equivalent to $\overline{S}_1$ in the maximum mode, and its timing is the same as for M/$\overline{IO}$. (T = HIGH, R = LOW.) This signal floats to 3-state OFF in local bus "hold acknowledge."
$\overline{DEN}$	26	O	**Data Enable:** provided as an output enable for the 8286/8287 in a minimum system which uses the transceiver. $\overline{DEN}$ is active LOW during each memory and I/O access and for INTA cycles. For a read or $\overline{INTA}$ cycle it is active from the middle of T_2 until the middle of T_4, while for a write cycle it is active from the beginning of T_2 until the middle of T_4. $\overline{DEN}$ floats to 3-state OFF in local bus "hold acknowledge."
HOLD, HLDA	31, 30	I/O	**HOLD:** indicates that another master is requesting a local bus "hold." To be acknowledged HOLD must be active HIGH. The processor receiving the "hold" request will issue HLDA (HIGH) as an acknowledgement in the middle of a T_4 or T_1 clock cycle. Simultaneous with the issuance of HLDA the processor will float the local bus and control lines. After HOLD is detected as being LOW, the processor will LOWer HLDA, and when the processor needs to run another cycle, it will again drive the local bus and control lines. The same rules as for $\overline{RQ}$/$\overline{GT}$ apply regarding when the local bus will be released. HOLD is not asynchronous input. External synchronization should be provided if the system cannot otherwise guarantee the setup time.

8087

Numeric Data Processor
iAPX86 Family

DISTINCTIVE CHARACTERISTICS

- High performance arithmetic and transcendental functions in hardware
- Supports 8-, 16-, 32-, 64-bit integer
- Performs 32-, 64-, 80-bit floating point calculations conforming to IEEE standard
- Standard 8086 instruction set and addressing modes
- Built-in exception handling functions
- Multibus* system compatible

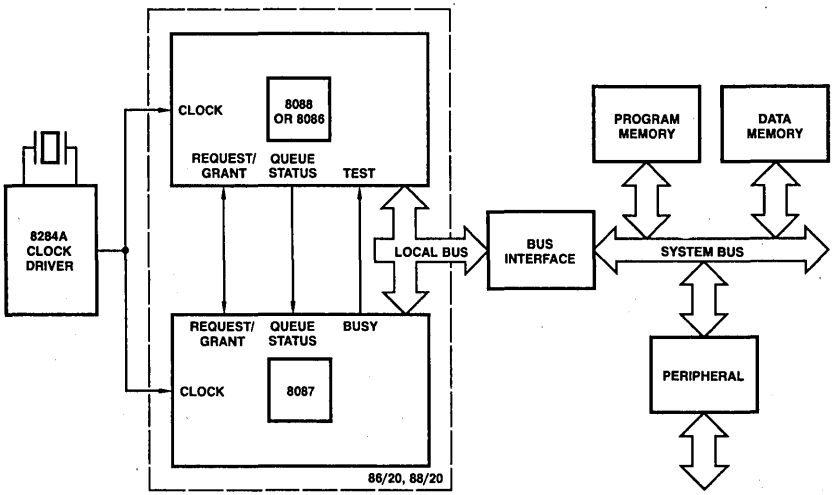
GENERAL DESCRIPTION

The 8087 is designed to do high performance numeric processing in hardware. It operates as the coprocessor to an 8086 or 8088 CPU, and can improve numeric throughput by a factor of 100 over the stand-alone CPU. It is programmed with the same instruction set as the 8086/88.

The 8087 does trigonometric, logarithmic, and exponential functions, which are essential in many scientific and military applications. The 8087 can also process BCD numbers up to 18 digits with no round-off error.

The 8087 is built in N-channel depletion load technology, in a 40 pin package.

BLOCK DIAGRAM



MMC-161

ORDERING INFORMATION

Package Type	Ambient Temperature Spec	Order Number
Hermetic Dip	$0°C \leq T_A \leq 70°C$	D8087

*Multibus is a registered trademark of Intel Corporation.

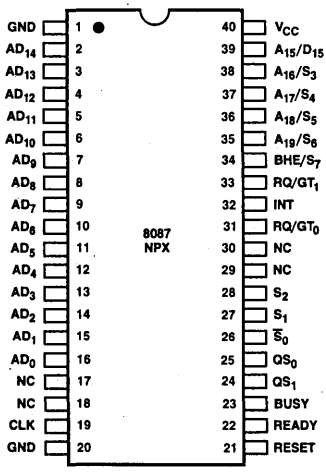
PIN DESCRIPTION

Symbol	Type	Name and Function
AD_{15}-AD_0	I/O	**Address Data:** These lines constitute the time multiplexed memory address (T_1) and data (T_2, T_3, T_W, T_4) bus. A_0 is analogous to $\overline{BHE}$ for the lower byte of the data bus, pins D_7-D_0. It is LOW during T_1 when a byte is to be transferred on the lower portion of the bus in memory operations. Eight-bit oriented devices tied to the lower half of the bus would normally use A_0 to condition chip select functions. These lines are active HIGH. They are input/output lines for 8087 driven bus cycles and are inputs which the 8087 monitors when the 8086/8088 is in control of the bus.
A_{19}/S_6, A_{18}/S_5, A_{17}/S_4, A_{16}/S_3	I/O	**Address Memory:** During T_1 these are the four most significant address lines for memory operations. During memory operations, status information is available on these lines during T_2, T_3, T_W, and T_4. For 8087 controlled bus cycles, S_6, S_4, and S_3 are reserved and currently one (HIGH), while S_5 is always LOW. These lines are inputs which the 8087 monitors when the 8086/8088 is in control of the bus.
$\overline{BHE}/S_7$	I/O	**Bus High Enable:** During T_1 the bus high enable signal ($\overline{BHE}$) should be used to enable data onto the most significant half of the data bus, pins D_{15}-D_8. Eight-bit oriented devices tied to the upper half of the bus would normally use $\overline{BHE}$ to condition chip select functions. $\overline{BHE}$ is LOW during T_1 for read and write cycles when a byte is to be transferred on the high portion of the bus. The S_7 status information is available during T_2, T_3, T_W, and T_4. The signal is active LOW. S_7 is an input which the 8087 monitors during 8086/8088 controlled bus cycles.
S_2, S_1, S_0	I/O	**Status:** For 8087 driven bus cycles, these status lines are encoded as follows: $\overline{S_2}$ $\overline{S_1}$ $\overline{S_0}$ 0 (LOW) X X Unused 1 (HIGH) 0 0 Unused 1 0 1 Read Memory 1 1 0 Write Memory 1 1 1 Passive Status is driven active during T_4, remains valid during T_1 and T_2, and is returned to the passive state (1, 1, 1) during T_3 or during T_W when READY is HIGH. This status is used by the 8288 Bus Controller to generate all memory access control signals. Any change in $\overline{S_2}$, $\overline{S_1}$, or $\overline{S_0}$ during T_4 is used to indicate the beginning of a bus cycle, and the return to the passive state in T_3 or T_W is used to indicate the end of a bus cycle. These signals are monitored by the 8087 when the 8086/8088 is in control of the bus.
$\overline{RQ}/\overline{GT}_0$	I/O	**Request/Grant:** This request/grant pin is used by the NDP to gain control of the local bus from the CPU for operand transfers or on behalf of another bus master. It must be connected to one of the two processor request/grant pins. The request/grant sequence on this pin is as follows: 1. A pulse one clock wide is passed to the CPU to indicate a local bus request by either the 8087 or the master connected to the 8087 $\overline{RQ}/\overline{GT}_1$ pin. 2. The NDP waits for the grant pulse and when it is received will either initiate bus transfer activity in the clock cycle following the grant or pass the grant out on the $\overline{RQ}/\overline{GT}_1$ pin in this clock if the initial request was for another bus master. 3. The 8087 will generate a release pulse to the CPU one clock cycle after the completion of the last NDP bus cycle or on receipt of the release pulse from the bus master on $\overline{RQ}/\overline{GT}_1$.
$\overline{RQ}/\overline{GT}_1$	I/O	**Request/Grant:** This request/grant pin is used by another local bus master to force the NDP to release the local bus at the end of the processor's current bus cycle. If the NDP is not in control of the bus when the request is made the request/grant sequence is passed through the NDP on the $\overline{RQ}/\overline{GT}_0$ pin one cycle later. Subsequent grant and release pulses are also passed through the NDP with a two and one clock delay, respectivley, for resynchronization. $\overline{RQ}/\overline{GT}_1$ has an internal pullup resistor, and so may be left unconnected. If the NDP has control of the bus the request/grant sequence is as follows: 1. A pulse 1 CLK wide from another local bus master indicates a local bus request to the 8087 (pulse 1). 2. During the NDP's next T_4 or T_1 a pulse 1 CLK wide from the 8087 to the requesting master (pulse 2) indicates that the 8087 has allowed the local bus to float and that it will enter the "RQ/GT acknowledge" state at the next CLK. The NDP's control unit is disconnected logically from the local bus during "RQ/GT acknowledge." 3. A pulse 1 CLK wide from the requesting master indicates to the 8087 (pulse 3) that the "RQ/GT" request is about to end and that the 8087 can reclaim the local bus at the next CLK. Each master-master exchange of the local bus is sequence of 3 pulses. There must be one dead CLK cycle after each bus exchange. Pulses are active LOW.
QS_1, QS_0	I	**QS_1, QS_0:** QS_1 and QS_0 provide the 8087 with status to allow tracking of the CPU instruction queue. QS_1 QS_0 0 (LOW) 0 No Operation 0 1 First Byte of Op Code from Queue 1 (HIGH) 0 Empty the Queue 1 1 Subsequent Byte from Queue
INT	O	**Interrupt:** This line is used to indicate that an unmasked exception has occurred during numeric instruction execution when 8087 interrupts are enabled. This signal is typically routed to an 8259A. INT is active HIGH.
BUSY	O	**Busy:** This signal indicates that the 8087 NEU is executing a numeric instruction. It is connected to the CPU's $\overline{TEST}$ pin to provide CPU-NDP synchronization. In the case of an unmasked exception BUSY remains active until the exception is cleared. BUSY is active HIGH.

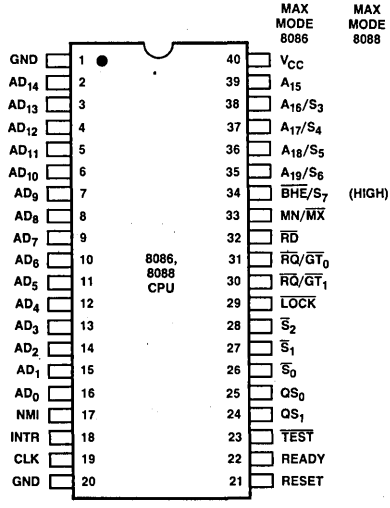
PIN DESCRIPTION (Cont.)

Symbol	Type	Name and Function
READY	I	**Ready:** READY is the acknowledgment from the addressed memory device that it will complete the data transfer. The RDY signal from memory is synchronized by the 8284A Clock Generator to form READY. This signal is active HIGH.
RESET	I	**Reset:** RESET causes the processor to immediately terminate its present activity. The signal must be active HIGH for at least four clock cycles. RESET is internally synchronized.
CLK	I	**Clock:** The clock provides the basic timing for the processor and bus controller. It is asymmetric with a 33% duty cycle to provide optimized internal timing.
V_{CC}		**Power:** V_{CC} is the +5V power supply pin.
GND		**Ground:** GND are the ground pins.

CONNECTION DIAGRAMS
Top Views
D-40



MMC-162



MMC-163

Note: Pin 1 is marked for orientation.

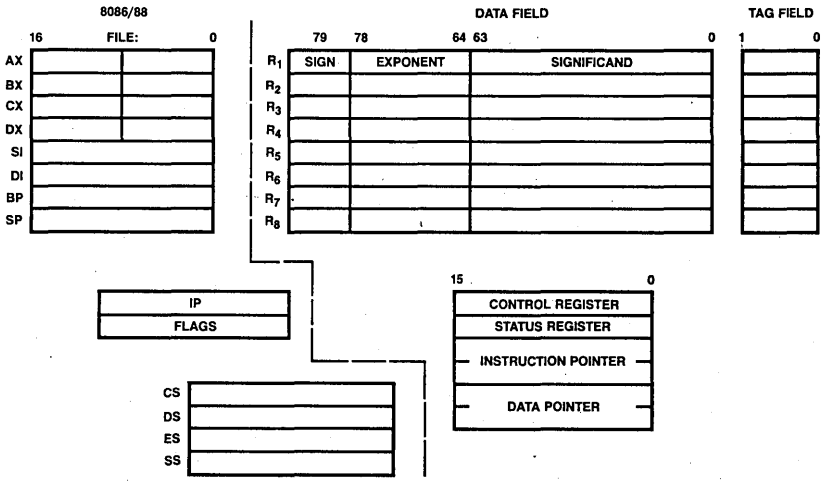
FUNCTIONAL DESCRIPTION

The 8087 is a numeric processor extension that provides arithmetic and logical instruction support for a variety of numeric data types. It also executes numerous built-in transcendental functions (e.g., tangent and log functions). The 8087 executes instructions as a coprocessor to a maximum mode 8086 or 8088. Figure 3 presents the registers of the 8087 plus CPU combination. Table 2 shows the range of data types supported by the NDP. The 8087 is treated as an extension to the CPU, providing register, data types, control, and instruction capabilities at the hardware level. At the programmers level the CPU and NDP is viewed as a single unified processor.

System Configuration

As a coprocessor to an 8086 or 8088, the 8087 is wired in parallel with the CPU as shown in Figure 4. The CPU's status ($\overline{S_0}$-$\overline{S_2}$) and queue status lines (QS_0-QS_1) enable the 8087 to monitor and decode instructions in synchronization with the CPU and without any CPU overhead. Once started the 8087 can process in parallel with, and independent of the host CPU. The NPX can interrupt the CPU when it detects an error or exception. The 8087's interrupt request line is typically routed to the CPU through an 8259A Programmable Interrupt Controller.

Figure 3. 8087 Register Architecture



MMC-164

Table 2. 8087 Data Types

Data Formats	Range	Precision	Most Significant Byte (7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0 | 7 0)
Byte Integer	10^2	8 Bits	I_7 I_0 Two's Complement
Word Integer	10^4	16 Bits	I_{15} I_0 Two's Complement
Short Integer	10^9	32 Bits	I_{31} I_0 Two's Complement
Long Integer	10^{18}	64 Bits	I_{63} I_0 Two's Complement
Packed BCD	10^{18}	18 Digits	S — D_{17} D_{16} ... D_1 D_0
Short Real	$10^{\pm 38}$	24 Bits	S E_7 E_0 F_1 F_{23} F_0 Implicit
Long Real	$10^{\pm 308}$	53 Bits	S E_{10} E_0 F_1 F_{52} F_0 Implicit
Temporary Real	$10^{\pm 4932}$	64 Bits	S E_{14} E_0 F_0 F_{63}

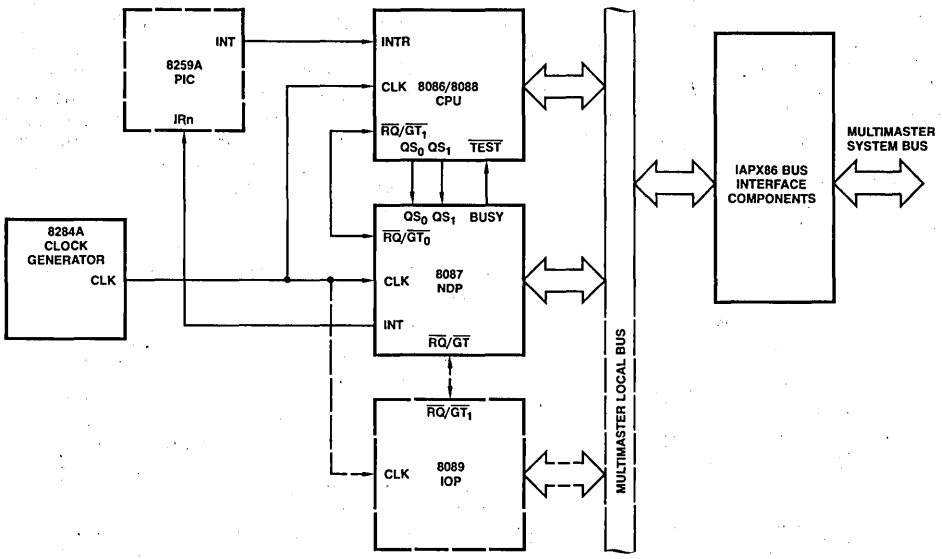
Integer: 1

Packed BCD: $(-1)^S (D_{17}...D_0)$

Real: $(-1)^S (2^{E\text{-}BIAS})(F_0 \bullet F_1...)$

Bias = 127 for Short Real
1023 for Long Real
16383 for Temp Real

Figure 4. NDP System Configuration



The 8087 uses one of the request/grant lines (typically $\overline{RQ}/\overline{GT}_1$) to obtain control of the local bus for data transfers. The other request/grant line is available for general system use (for instance by an I/O processor in LOCAL mode). A bus master can also be connected to the 8087's RQ/GT_1 line. In this configuration the 8087 will pass the request/grant handshake signals between the CPU and the attached master when the 8087 is not in control of the bus and will relinquish the bus to the master directly when the 8087 is in control. In this way two additional masters can be configured; one will share the 8086 bus with the 8087 on a first come first served basis, and the second will be guaranteed to be higher in priority than the 8087.

As Figure 4 shows, all processors utilize the same clock generator and system bus interface components.

Bus Operation

The 8087 bus structure, operation and timing are identical to all other processors in the 8086 family. The address is time multiplexed with the data on the first 16/8 lines of the address/data bus. A_{16} through A_{19} are time multiplexed with four status lines S_3-S_6. S_3, S_4 and S_6 are always one (high) for 8087 driven bus cycles while S_5 is always zero (low). When the 8087 is monitoring CPU bus cycles (passive mode) S_6 is also monitored by the 8087 to differentiate 8086/8088 activity from that of a local I/O processor or any other local bus master. (The 8086/8088 must be the only processor on the local bus to drive S_6 low.) S_7 is multiplexed with and has the same value as $\overline{BHE}$ for all 8087 bus cycles.

The first three status lines, $\overline{S}_0$-$\overline{S}_2$, are used with an 8288 bus controller to determine the type of bus cycle being run:

$\overline{S}_2$	$\overline{S}_1$	$\overline{S}_0$	
0	X	X	Unused
1	0	0	Unused
1	0	1	Memory Data Read
1	1	0	Memory Data Write
1	1	1	Passive (no bus cycle)

Programming Interface

The NDP includes the standard 8086/88 instruction set for general data manipulation and program control. It also includes 68 numeric instructions for extended precision integer, floating point, trigonometric, logarithmic, and exponential functions. Sample execution times for several NDP functions are shown in Figure 4.

Any instruction executed by the NDP is the combined result of the CPU and NPX activity. The CPU and NPX have specialized functions and registers providing fast concurrent operation. The CPU controls overall program execution while the NPX uses the coprocessor interface to recognize and perform numeric operations.

Table 2 lists the eight data types the 8087 supports and presents the format for each type. Internally, the NPX holds all numbers in the temporary real format. Load and store instructions automatically convert operands represented in memory as 16-, 32-, or 64-bit integers, 32- or 64-bit floating point numbers or 18-digit packed BCD numbers into temporary real format and vice versa. The NDP also provides the capability to control round off, underflow, and overflow errors in each calculation.

Computations in the NPX use the processor's register stack. These eight 80-bit registers provide the equivalent capacity of 20 32-bit registers. The NPX register set can be accessed as a stack, with instructions operating on the top one or two stack elements, or as a fixed register set, with instructions operating on explicitly designated registers.

All 8087 instructions appear as ESCAPE instructions to the host CPU. Assembly language programs are written in ASM-86, the 8086/88 assembly language. Table 3 gives the execution times of some typical numeric instructions.

NUMERIC PROCESSOR EXTENSION ARCHITECTURE

As shown in Figure 5, the 8087 is internally divided into two processing elements, the control unit (CU) and the numeric execution unit (NEU). The NEU executes all numeric instructions, while the CU receives and decodes instructions, reads and writes

Figure 5. 8087 Block Diagram

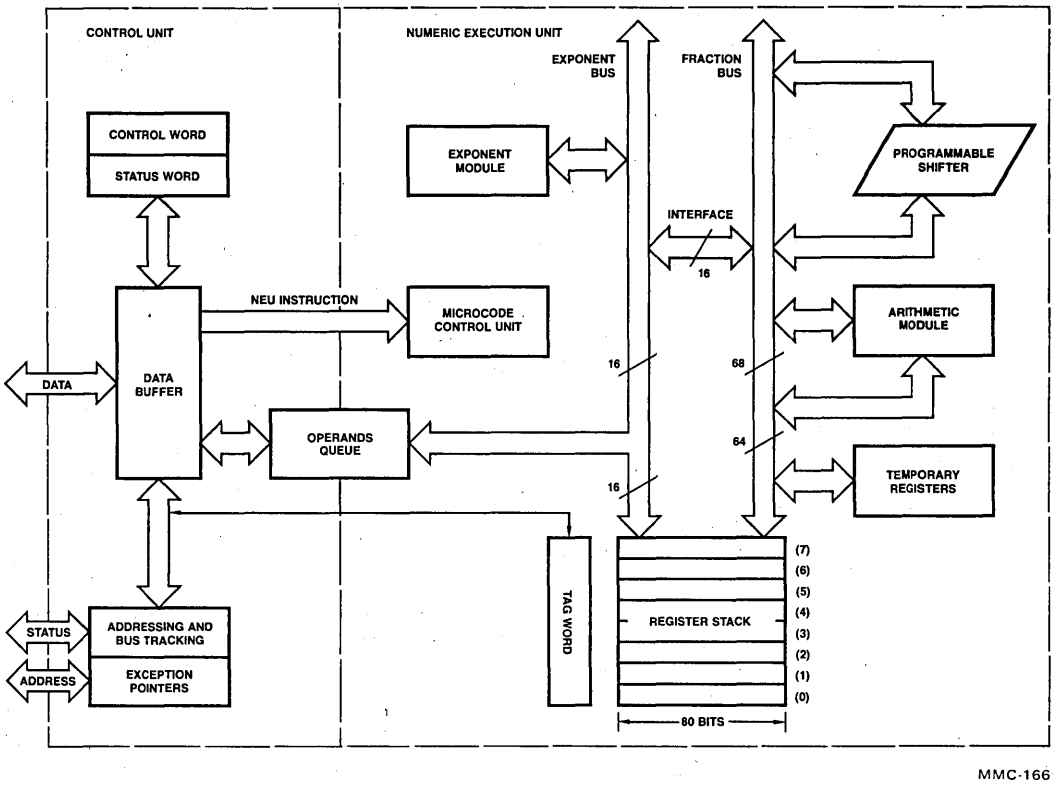


TABLE 3. EXECUTION TIME FOR SELECTED 8087 NUMERIC INSTRUCTIONS AND CORRESPONDING 8086 EMULATION

Floating Point Instruction	Approximate Execution Time (μs)	
	8087 (5MHz Clock)	8086 Emulation
Add/Subtract Magnitude	14/18	1,600
Multiply (single precision)	19	1,600
Multiply (extended precision)	27	2,100
Divide	39	3,200
Compare	9	1,300
Load (double precision)	10	1,700
Store (double precision)	21	1,200
Square Root	36	19,600
Tangent	90	13,000
Exponentiation	100	17,000

memory operands and executes NPX control instructions. The two elements are able to operate independently of one another, allowing to CU to maintain synchronization with the CPU while the NEU is busy processing a numeric instruction.

Control Unit

The CU keeps the 8087 operating in synchronization with its host CPU. 8087 instructions are intermixed with CPU instructions in a single instruction stream. The CPU fetches all instructions from memory; by monitoring the status signals ($\overline{S}_0$-$\overline{S}_2$, S_6) emitted by the CPU, the NPX control unit determines when an 8086 instruction is being fetched. The CU monitors the Data bus in parallel with the CPU to obtain instructions that pertain to the 8087.

The CU maintains an instruction queue that is identical to the queue in the host CPU. The CU automatically determines if the CPU is an 8086 or an 8088 immediately after reset (by monitoring the $\overline{BHE}/S_7$ line) and matches its queue length accordingly. By monitoring the CPU's queue status lines (QS_0, QS_1), the CU obtains and decodes instructions from the queue in synchronization with the CPU.

A numeric instruction appears as an ESCAPE instruction to the 8086 or 8088 CPU. Both the CPU and NPX decode and execute the ESCAPE instruction together. The 8087 only recognizes the numeric instructions shown in Table 5. The start of a numeric operation is accomplished when the CPU executes the ESCAPE instruction. The instruction may or may not identify a memory operand.

The CPU does, however, distinguish between ESC instructions that reference memory and those that do not. If the instruction refers to a memory operand, the CPU calculates the operand's address using any one of its available addressing modes, and then performs a "dummy read" of the word at that location. (Any location within the 1M byte address space is allowed.) This is a normal read cycle except that the CPU ignores the data it receives. If the ESC instruction does not contain a memory reference (e.g., an 8087 stack operation), the CPU simply proceeds to the next instruction.

An 8087 Instruction can have one of three memory reference options: (1) not reference memory; (2) load an operand from memory into the 8087; or (3) store an operand from the 8087 into memory. If no memory reference is required, the 8087 simply executes its instruction. If a memory reference is required, the CU uses a "dummy read" cycle initiated by the CPU to capture and save the address that the CPU places on the bus. If the instruction is a load, the CU additionally captures the data word when it becomes available on the local data bus. If data required is longer than one word, the CU immediately obtains the bus from the CPU using the request/grant protocol and reads the rest of the information in consecutive bus cycles. In a store operation, the CU captures and saves the store address as in a load, and ignores the data word that follows in the "dummy read" cycle. When the 8087 is ready to perform the store, the CU obtains the bus from the CPU and writes the operand starting at the specified address.

Numeric Execution Unit

The NEU executes all instructions that involve the register stack; these include arithmetic, logical, transcendental, constant and data transfer instructions. The data path in the NEU is 84 bits wide (68 fraction bits, 15 exponent bits and a sign bit) which allows internal operand transfers to be performed at very high speeds.

When the NEU begins executing an instruction, it activates the 8087 BUSY signal. This signal can be used in conjunction with the CPU WAIT instruction to resynchronize both processors when the NEU has completed its current instruction.

Register Set

The 8087 register set is shown in Figure 3. Each of the eight data registers in the 8087's register stack is 80 bits wide and is divided into "fields" corresponding to the NDP's temporary real data type.

At a given point in time the TOP field in the control word identifies the current top-of-stack register. A "push" operation decrements TOP by 1 and loads a value into the new top register. A "pop" operation stores the value from the current top register and then increments TOP by 1. The 8087 register stack grows "down" toward lower-addressed registers.

Instructions may address the data registers either implicitly or explicitly. Many instructions operate on the register at the top of the stack. These instructions implicitly address the register pointed to by the TOP. Other instructions allow the programmer to explicitly specify the register which is to be used. Explicit register addressing is "top-relative."

Status Word

The status word shown in Figure 6 reflects the overall state of the 8087; it may be stored in memory and then inspected by CPU code. The status word is a 16-bit register divided into fields as shown in Figure 6. The busy bit (bit 15) indicates whether the NEU is either executing an instruction or has an interrupt request pending (B = 1), or is idle (B = 0). Several instructions which store and manipulate the status word are executed exclusively by the CU, and these do not set the busy bit themselves.

The four numeric condition code bits (C_0-C_3) are similar to the flags in a CPU: various instructions update these bits to reflect the outcome of NDP operations. The effect of these instructions on the condition code bits is summarized in Table 4.

Bits 14-12 of the status word point to the 8087 register that is the current top-of-stack (TOP) as described above.

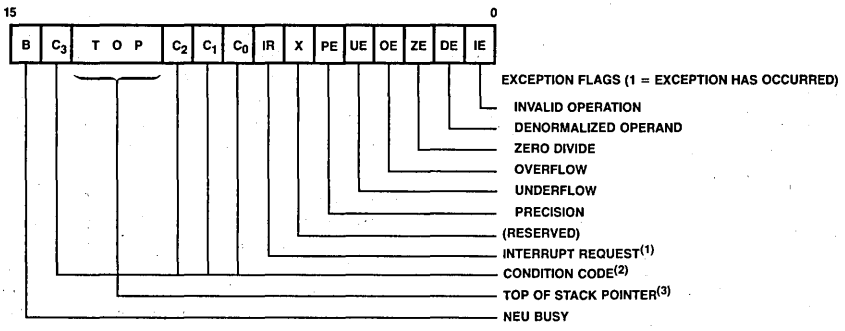
Bit 7 is the interrupt request bit. This bit is set if any unmasked exception bit is set and cleared otherwise.

Bits 5-0 are set to indicate that the NEU has detected an exception while executing an instruction.

Tag Word

The tag word marks the content of each register as shown in Figure 7. The principal function of the tag word is to optimize the NDP's performance. The tag word can be used, however, to interpret the contents of 8087 registers.

Figure 6. 8087 Status Word



(1)IR is set if any unmasked exception bit is set, cleared otherwise.

(2)See Table 3 for condition code interpretation.

(3)Top Values:

000 = Register 0 is Top of Stack.
001 = Register 1 is Top of Stack.
•
•
•
111 = Register 7 is Top of Stack.

MMC-167

TABLE 4. CONDITION CODE INTERPRETATION

Instruction	C_3	C_2	C_1	C_0	Interpretation
Compare, Test	0	X	X	0	A > B
	0	X	X	1	A < B
	1	X	X	0	A = B
	1	X	X	1	A ? B (not comparable)
Remainder	U	0	U	U	Complete reduction
	U	1	U	U	Incomplete reduction
Examine	0	0	0	0	Valid, positive, unnormalized
	0	0	0	1	Invalid, positive, exponent = 0
	0	0	1	0	Valid, negative, unnormalized
	0	0	1	1	Invalid, negative, exponent = 0
	0	1	0	0	Valid, positive, normalized
	0	1	0	1	Infinity, positive
	0	1	1	0	Valid, negative, normalized
	0	1	1	1	Infinity, negative
	1	0	0	0	Zero, positive
	1	0	0	1	Empty
	1	0	1	0	Zero, negative
	1	0	1	1	Empty
	1	1	0	0	Invalid, positive, exponent = 0
	1	1	0	1	Empty
	1	1	1	0	Invalid, negative, exponent = 0
	1	1	1	1	Empty

X = value is not affected by instruction
U = value is undefined following instruction

Figure 7. 8087 Tag Word

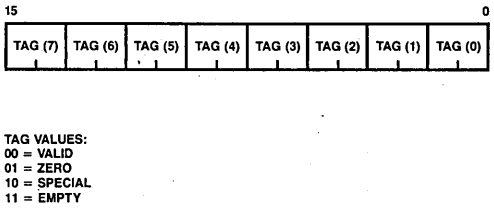
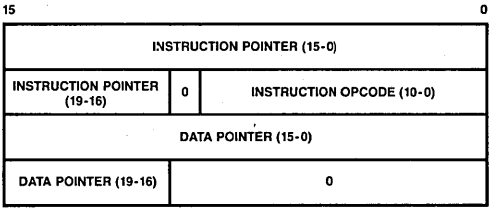


Figure 8. 8087 Instruction and Data Pointers



Instruction and Data Pointers

The instruction and data pointers (see Figure 8) are provided for user-written error handlers. Whenever the 8087 executes an NEU instruction, the CU saves the instruction address, the operand address (if present) and the instruction opcode. 8087 instructions can store this data into memory.

Control Word

The 8087 provides several processing options which are selected by loading a word from memory into the control word. Figure 9 shows the format and encoding of the fields in the control word.

The low order byte of this control word configures 8087 interrupts and exception masking. Bits 5-0 of the control word contain individual masks for each of the six exceptions that the 8087 recognizes and bit 7 contains a general mask bit for all 8087 interrupts. The high order byte of the control word configures the 8087 operating mode including precision, rounding, and infinity controls. The precision control bits (bits 9-8) can be used to set the 8087 internal operating precision at less than the default of temporary real precision. This can be useful in providing compatibility with earlier generation arithmetic processors of smaller precision than the 8087. The rounding control bits (bits 11-10) provide for directed rounding and true chop as well as the unbiased round to the nearest mode specified in the proposed IEEE standard. Control over closure of the number space at infinity is also provided (either affine closure, $\pm\infty$, or projective closure, ∞, is treated as unsigned, may be specified).

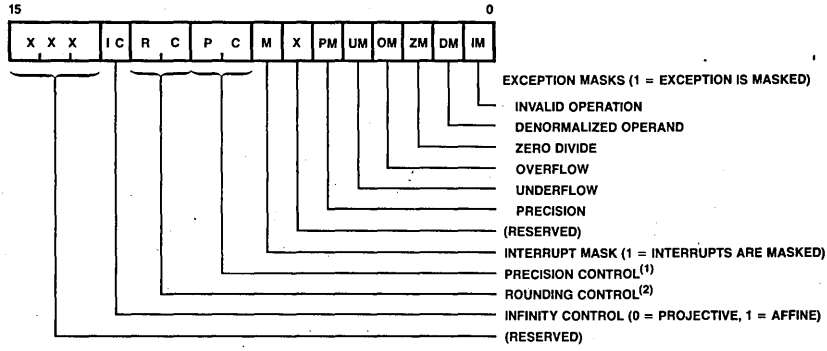
Exception Handling

The 8087 detects six different exception conditions that can occur during instruction execution. Any or all exceptions will cause an interrupt if unmasked and interrupts are enabled.

If interrupts are disabled the 8087 will simply continue execution regardless of whether the host clears the exception. If a specific exception class is masked and that exception occurs, however, the 8087 will post the exception in the status register and perform an on-chip default exception handling procedure, thereby allowing processing to continue. The exceptions that the 8087 detects are the following:

1. INVALID OPERATION: Stack overflow, stack underflow, indeterminate form (0/0, $\infty - \infty$, etc.) or the use of a Non-Number (NAN) as an operand. An exponent value is reserved and any bit pattern with this value in the exponent field is termed a Non-Number and causes this exception. If this exception is masked, the 8087's default response is to generate a specific NAN called INDEFINITE, or to propagate already existing NANs as the calculation result.
2. OVERFLOW: The result is too large in magnitude to fit the specified format. The 8087 will generate an encoding for infinity if this exception is masked.
3. ZERO DIVISOR: The divisor is zero while the dividend is a non-infinite, non-zero number. Again, the 8087 will generate an encoding for infinity if this exception is masked.
4. UNDERFLOW: The result is non-zero but too small in magnitude to fit in the specified format. If this exception is masked the 8087 will denormalize (shift right) the fraction until the exponent is in range. This process is called gradual underflow.
5. DENORMALIZED OPERAND: At least one of the operands or the result is denormalized; it has the smallest exponent but a non-zero significand. Normal processing continues if this exception is masked off.
6. INEXACT RESULT: If the true result is not exactly representable in the specified format, the result is rounded according to the rounding mode, and this flag is set. If this exception is masked, processing will simply continue.

Figure 9. 8087 Control Word



(1)Precision Control
- 00 = 24 bits
- 01 = Reserved
- 10 = 53 bits
- 11 = 64 bits

(2)Rounding Control
- 00 = Round to Nearest or Even
- 01 = Round Down (toward $-\infty$)
- 10 = Round Up (toward $+\infty$)
- 11 = Chop (truncate toward zero)

MMC-170

MAXIMUM RATINGS*

Ambient Temperature Under Bias	0 to 70°C
Storage Temperature	−65 to +150°C
Voltage on Any Pin with Respect to Ground	−1.0 to +7.0V
Power Dissipation	3.0W

*Stresses above those listed under Maximum Ratings may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS (T_A = 0 to 70°C, V_{CC} = +5V ±10%)

Parameter	Description	Test Conditions	Min	Max	Units
V_{IL}	Input Low Voltage		−0.5	+0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC} +0.5	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0μA		0.45	V
V_{OH}	Output High Voltage	I_{OH} = −400μA	2.4		V
I_{CC}	Power Supply Current	T_A = 25°C		475	mA
I_{LI}	Input Leakage Current	0V ≤ V_{IN} ≤ V_{CC}		±10	μA
I_{LO}	Output Leakage Current	0.45V ≤ V_{OUT} ≤ V_{CC}		±10	μA
V_{CL}	Clock Input Low Voltage		−0.5	+0.6	V
V_{CH}	Clock Input High Voltage		3.9	V_{CC} +1.0	V
C_{IN}	Capacitance of Inputs	fc = 1MHz		10	pF
C_{IO}	Capacitance of I/O Buffer (AD_{0-15}, A_{16}-A_{19}, BHE, S_2-S_0, RQ/GT) and CLK	fc = 1MHz		15	pF
C_{OUT}	Capacitance of Outputs BUSY, INT	fc = 1MHz		10	pF

AC CHARACTERISTICS (T_A = 0 to 70°C, V_{CC} = +5V ±10%)

TIMING REQUIREMENTS

Parameter	Description	Test Conditions	Min	Max	Units
TCLCL	CLK Cycle Period		200	500	ns
TCLCH	CLK Low Time		(2/3 TCLCL) −15		ns
TCHCL	CLK High Time		(1/3 TCLCL) +2		ns
TCH_1CH_2	CLK Rise Time	From 1.0 to 3.5V		10	ns
TCL_2CL_1	CLK Fall Time	From 3.5 to 1.0V		10	ns
TDVCL	Data In Setup Time		30		ns
TCLDX	Data In Hold Time		10		ns
TRYHCH	READY Setup Time		(2/3 TCLCL) −15		ns
TCHRYX	READY Hold Time		30		ns
TRYLCL	READY Inactive to CLK (See Note 3)		−8		ns
TGVCH	RQ/GT Setup Time		30		ns
TCHGX	RQ/GT Hold Time		40		ns
TQVCL	QS_{0-1} Setup Time		30		ns
TCLQX	QS_{0-1} Hold Time		10		ns
TSACH	Status Active Setup Time		30		ns
TSNCL	Status Inactive Setup Time		30		ns
TILIH	Input Rise Time (Except CLK)	From 0.8 to 2.0V		20	ns
TIHIL	Input Fall Time (Except CLK)	From 2.0 to 0.8V		12	ns

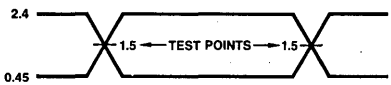
AC CHARACTERISTICS (Cont.)

TIMING RESPONSES

Parameter	Description	Test Conditions	Min	Max	Units
TCLML	Command Active Delay (See Note 1)	C_L = 20-100pF for all 8087 outputs (In addition to 8087 self-load)	10	35	ns
TCLMH	Command Inactive Delay (See Note 1)		10	35	ns
TRYHSH	Ready Active to Status Passive (See Note 2)			110	ns
TCHSV	Status Active Delay		10	110	ns
TCLSH	Status Inactive Delay		10	130	ns
TCLAV	Address Valid Delay		10	110	ns
TCLAX	Address Hold Time		10		ns
TCLAZ	Address Float Delay		TCLAX	80	ns
TSVLH	Status Valid to ALE High (See Note 1)			15	ns
TCLLH	CLK Low to ALE Valid (See Note 1)			15	ns
TCHLL	ALE Inactive Delay (See Note 1)			15	ns
TCLDV	Data Valid Delay		10	110	ns
TCHDX	Data Hold Time		10		ns
TCVNV	Control Active Delay (See Note 1)		5	45	ns
TCVNX	Control Inactive Delay (See Note 1)		10	45	ns
TCHBV	BUSY and INT Valid Delay		10	150	ns
TCHDTL	Direction Control Active Delay (See Note 1)			50	ns
TCHDTH	Direction Control Inactive Delay (See Note 1)			30	ns
TCLGL	RQ/GT Active Delay	C_L = 40pF (in addition to 8087 self-load)	0	85	ns
TCLGH	RQ/GT Inactive Delay		0	85	ns
TOLOH	Output Rise Time	From 0.8 to 2.0V		20	ns
TOHOL	Output Fall Time	From 2.0 to 0.8V		12	ns

Notes: 1. Signal at 8284A or 8288 shown for reference only.
2. Applies only to T_3 and wait states.
3. Applies only to T_2 state (8ns into T_3).

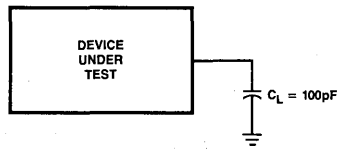
AC TESTING INPUT, OUTPUT WAVEFORM



MMC-171

AC testing: inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." The clock is driven at 4.3V and 0.25V timing measurements are made at 1.5V for both a logic "1" and "0."

AC TESTING LOAD CIRCUIT

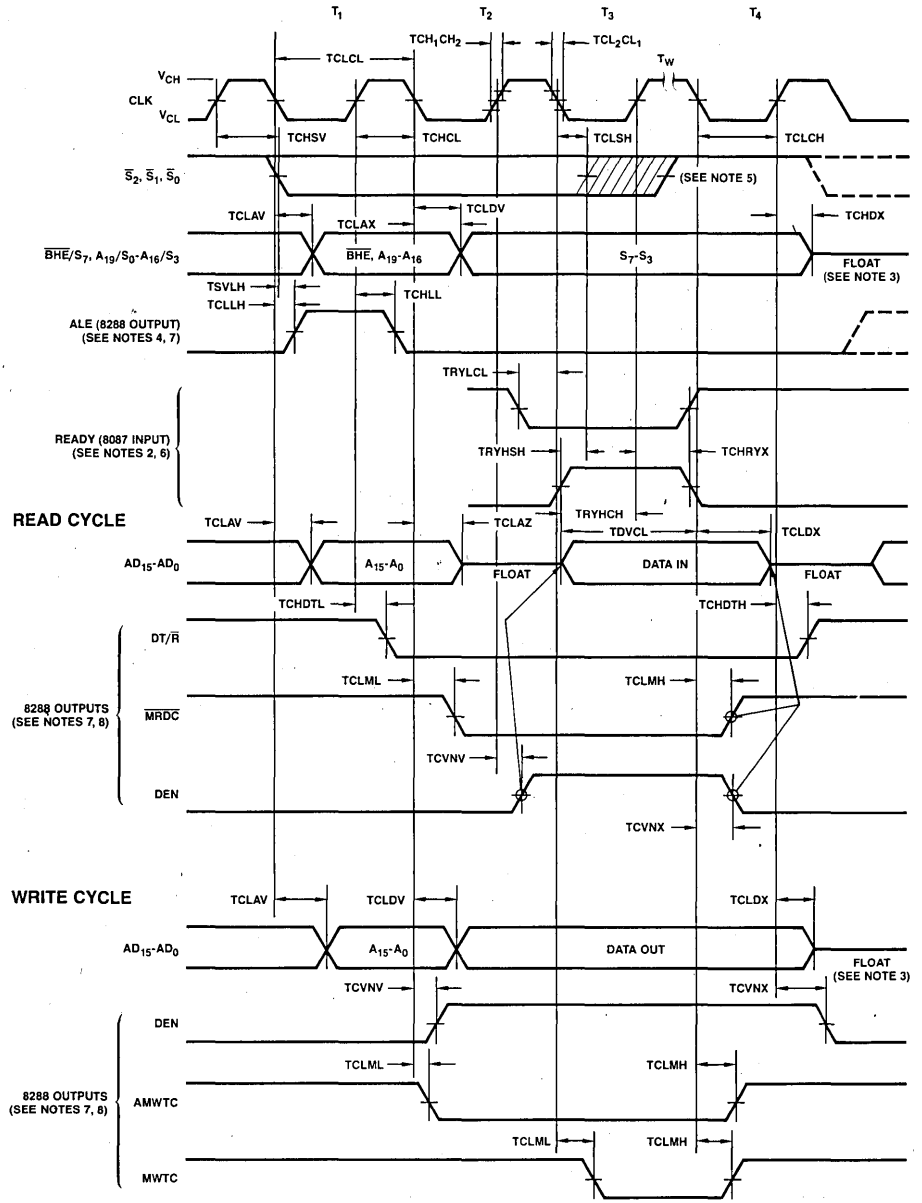


MMC-172

C_L includes jig capacitance

WAVEFORMS

MASTER MODE

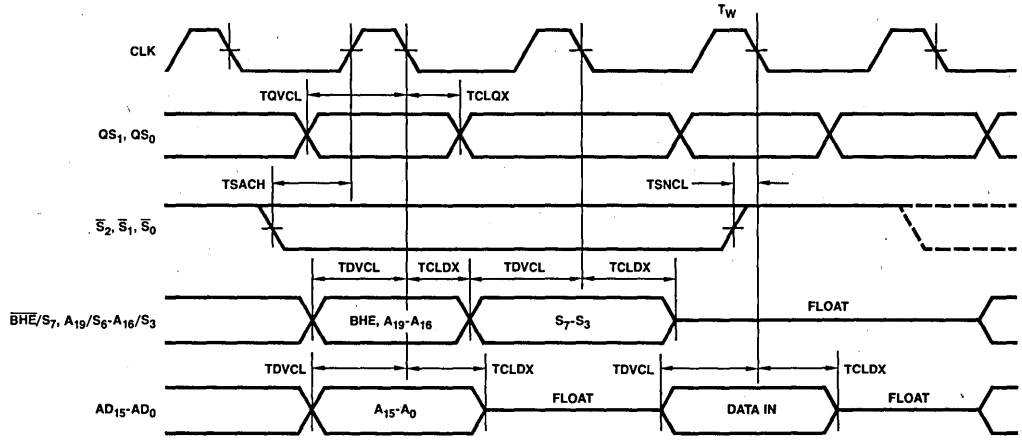


MMC-173

Notes:
1. All signals switch between V_{OL} and V_{OH} unless otherwise specified.
2. RDY is sampled near the end of T_2, T_3 and T_W to determine if T_W machine states are to be inserted.
3. The local bus floats only if the 8087 is returning control to the 8086/8088.
4. ALE rises at later of (TSVLH, TCLLH).
5. Status inactive in state just prior to T_4.
6. Ready should remain active until $S_{0\text{-}2}$ become inactive.
7. Signals at 8284A or 8288 are shown for reference only.
8. The issuance of 8288 command and control signals $\overline{\text{MRDC}}$, $\overline{\text{MWTC}}$, $\overline{\text{AMWC}}$ and DEN lags the active high 8288 CEN.
9. All timing measurements are made at 1.5V unless otherwise noted.

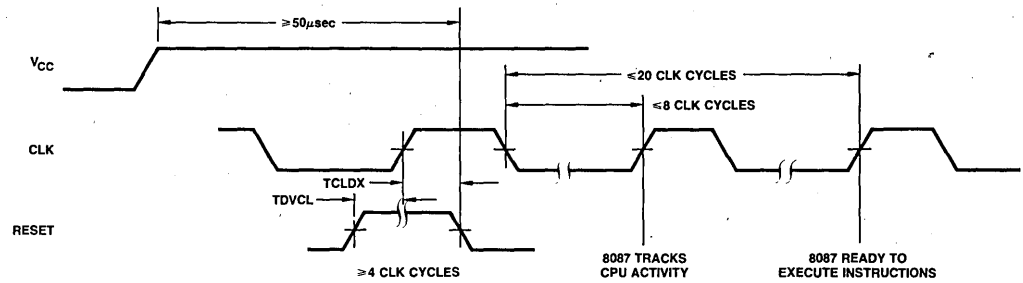
WAVEFORMS (Cont.)

PASSIVE MODE



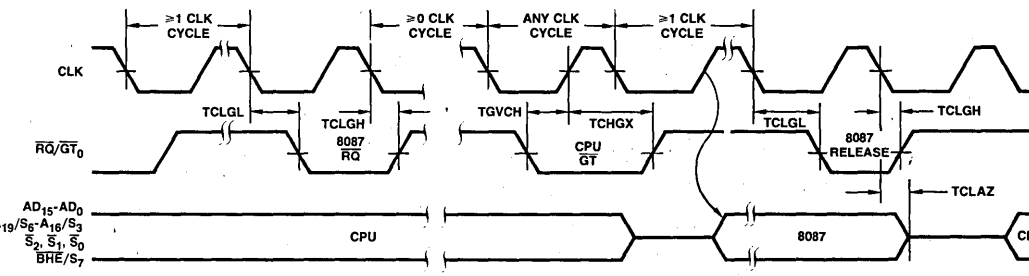
MMC-174

RESET



MMC-175

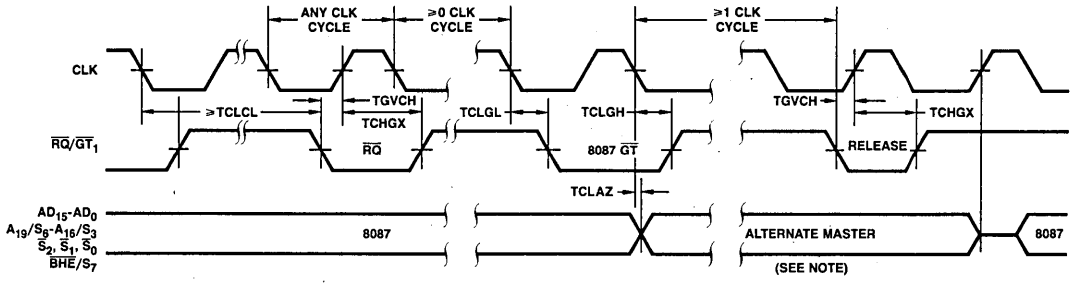
REQUEST/GRANT$_0$



MMC-176

WAVEFORMS (Cont.)

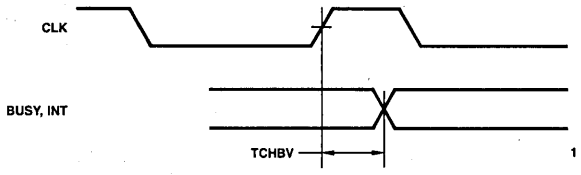
REQUEST/GRANT$_1$



MMC-177

Note: Alternate master may not drive the buses outside of the region shown without risking contention.

BUSY AND INTERRUPT



MMC-178

8088/8088-2

iAPX86 Family
8-Bit Microprocessor CPU

DISTINCTIVE CHARACTERISTICS

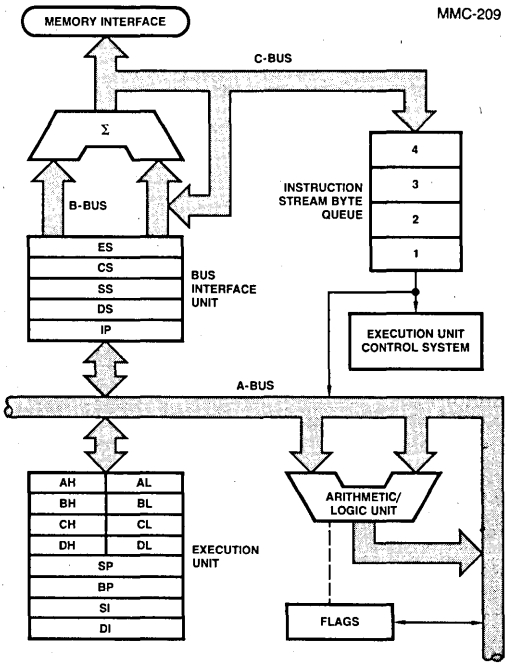
- 8-bit data bus, 16-bit internal architecture
- Directly addresses 1 Mbyte of memory
- Software compatible with 8086 CPU
- Byte, word, and block operations
- 24 operand addressing modes
- Powerful instruction set
- Efficient high level language implementation
- Two block options: 5MHz 8088
  8MHz 8088-2

GENERAL DESCRIPTION

The 8088 CPU is an 8-bit processor designed around the 8086 internal structure. Most functions of the 8088 are identical to the equivalent 8086 functions. The pinout is slightly different. The 8088 handles the external bus the same way the 8086 does, but handling only 8 bits at a time. Sixteen-bit words are fetched or written in two consecutive bus cycles. Both processors will appear identical to the software engineer, with the exception of execution time.

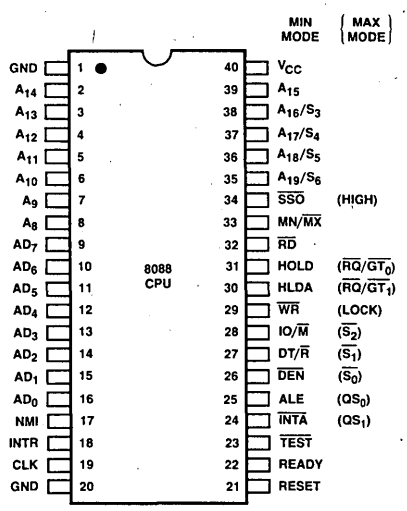
The 8088 is made with N-channel silicon gate technology and is packaged in a 40-pin Plastic or Ceramic DIP.

BLOCK DIAGRAM



CONNECTION DIAGRAM
Top View

D-40
P-40



Note: Pin 1 is marked for orientation. MMC-210

ORDERING INFORMATION

Package Type	Ambient Temperature	Order Number
Molded DIP	0°C ≤ T_A ≤ 70°C	P8088
Hermetic DIP		D8088 D8088-2
Hermetic DIP	−40°C ≤ T_A ≤ 85°C	ID8088 ID8088-2
Hermetic DIP	−55°C ≤ T_A ≤ 125°C	MD8088B MD8088-2B

02338B-MMP

THE 8088 COMPARED TO THE 8086

- The queue length is 4 bytes in the 8088, whereas the 8086 queue contains 6 bytes, or three words. The queue was shortened to prevent overuse of the bus by the BIU when prefetching instructions. This was required because of the additional time necessary to fetch instructions 8 bits at a time.
- To further optimize the queue, the prefetching algorithm was changed. The 8088 BIU will fetch a new instruction to load into the queue each time there is a 1 byte hole (space available) in the queue. The 8086 waits until a 2-byte space is available.
- The internal execution time of the instruction set is affected by the 8-bit interface. All 16-bit fetches and writes from/to memory take an additional four clock cycles. The CPU is also limited by the speed of instruction fetches. This latter problem only occurs when a series of simple operations occur. When the more sophisticated instructions of the 8088 are being used, the queue has time to fill and the execution proceeds as fast as the execution unit will allow.

The 8088 and 8086 are completely software compatible by virture of their identical execution units. Software that is system dependent may not be completely transferable, but software that is not system dependent will operate equally as well on an 8088 or an 8086.

The hardware interface of the 8088 contains the major differences between the two CPUs. The pin assignments are nearly identical, however, with the following functional changes:

- $A_8 - A_{15}$ – These pins are only address outputs on the 8088. These address lines are latched internally and remain valid throughout a bus cycle in a manner similar to the 8085 upper address lines.
- $\overline{BHE}$ has no meaning on the 8088 and has been eliminated.
- $\overline{SSO}$ provides the $\overline{SO}$ status information in the minimum mode. This output occurs on pin 34 in minimum mode only. $DT/\overline{R}$, $IO/\overline{M}$, and $\overline{SSO}$ provide the complete bus status in minimum mode.
- $IO/\overline{M}$ has been inverted to be compatible with the MCS-85 bus structure.
- ALE is delayed by one clock cycle in the minimum mode when entering HALT, to allow the status to be latched with ALE.

I/O ADDRESSING

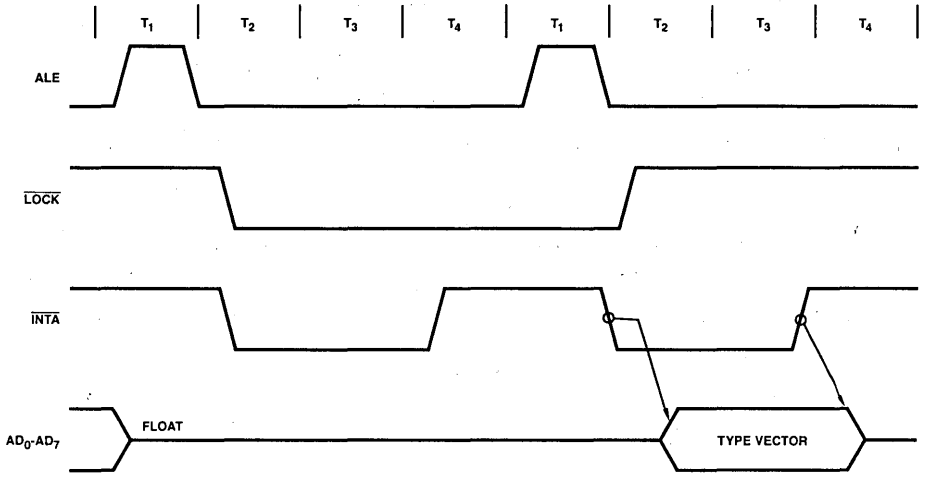
In the 8088, I/O operations can address up to a maximum of 64K I/O registers. The I/O address appears in the same format as the memory address on bus lines $A_{15} - A_0$. The address lines $A_{19} - A_{16}$ are zero in I/O operations. The variable I/O instructions, which use register DX as a pointer, have full address capability, while the direct I/O instructions directly address one or two of the 256 I/O byte locations in page 0 of the I/O address space. I/O ports are addressed in the same manner as memory locations.

Designers familiar with the 8085 or upgrading an 8085 design should note that the 8085 addresses I/O with an 8-bit address on both halves of the 16-bit address bus. The 8088 uses a full 16-bit address of its lower 16 address lines.

BUS OPERATION

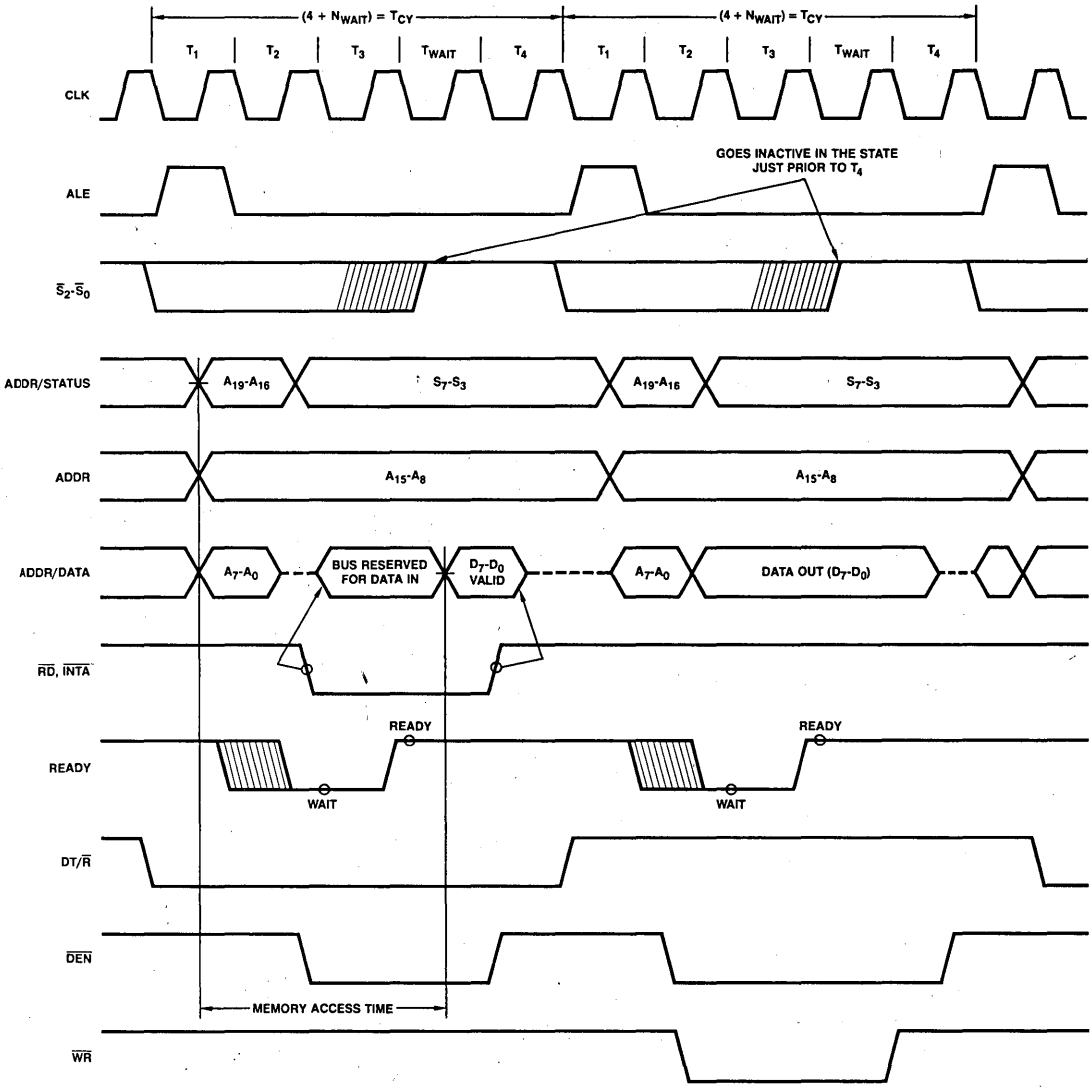
The 8088 address/data bus is broken into three parts – the lower eight address/data bits ($AD_0 - AD_7$), the middle eight address bits ($A_8 - A_{15}$), and the upper four address bits ($A_{16} - A_{19}$). The address/data bits and the highest four address bits are time multiplexed. This technique provides the most efficient use of pins on the processor, permitting the use of a standard 40 lead package. The middle eight address bits are not multiplexed, i.e. they remain valid throughout each bus cycle. In addition, the bus can be demultiplexed at the processor with a single address latch if a standard, non-multiplexed bus is desired for the system.

Figure 1. Interrupt Acknowledge Sequence



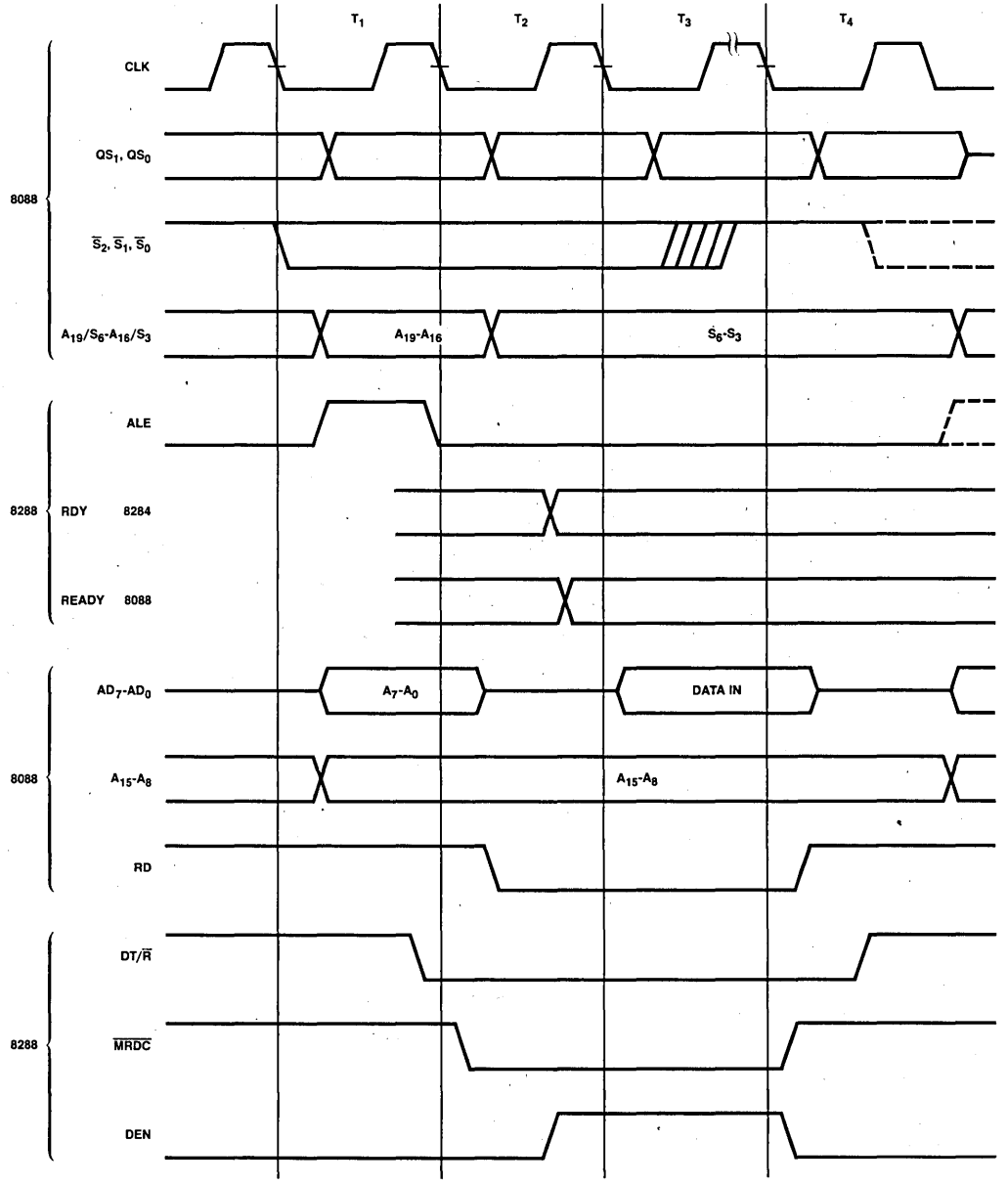
MMC-211

Figure 2. Basic System Timing



MMC-212

Figure 3. Medium Complexity System Timing



MMC-213

MAXIMUM RATINGS

Storage Temperature	−65 to +150°C
Voltage on any Pin with Respect to Ground	−1.0 to +7V
Power Dissipation	2.5W

Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS

(8088: T_A = 0 to +70°C, −40 to +85°C, −55 to +125°C, V_{CC} = 5V ±10%)
(8088-2: T_A = 0 to +70°C, −40 to +85°C, −55 to +125°C, V_{CC} = 5V ±10%)

Parameter	Description		Test Conditions	Min	Max	Units
V_{IL}	Input Low Voltage			−0.5	+0.8	V
V_{IH}	Input High Voltage			2.0	V_{CC} + 0.5	V
V_{OL}	Output Low Voltage		I_{OL} = 2.0 mA		0.45	V
V_{OH}	Output High Voltage		I_{OH} = −400μA	2.4		V
I_{CC}	Power Supply Current:	8088	T_A = 25°C		340	mA
		8088-2			350	
I_{LI}	Input Leakage Current		0V ≤ V_{IN} ≤ V_{CC}		±10	μA
I_{LO}	Output Leakage Current		0.45V ≤ V_{OUT} ≤ V_{CC}		±10	μA
V_{CL}	Clock Input Low Voltage			−0.5	+0.6	V
V_{CH}	Clock Input High Voltage			3.9	V_{CC} + 1.0	V
C_{IN}	Capacitance of Input Buffer (All input except AD_0−AD_7, RQ/GT)		fc = 1 MHz		15	pF
C_{IO}	Capacitance of I/O Buffer (AD_0−AD_7, RQ/GT)		fc = 1 MHz		15	pF

AC CHARACTERISTICS (8088: T_A = 0 to 70°C, V_{CC} = 5V ±10%)
(8088-2: T_A = 0 to 70°C, V_{CC} = 5V ±5%)

MINIMUM COMPLEXITY SYSTEM TIMING REQUIREMENTS

Parameter	Description	Test Conditions	8088		8088-2		Units
			Min	Max	Min	Max	
TCLCL	CLK Cycle Period		200	500	125	500	ns
TCLCH	CLK Low Time		(2/3 TCLCL) −15		(2/3 TCLCL)−15		ns
TCHCL	CLK High Time		(1/3 TCLCL) +2		(1/3 TCLCL) +2		ns
TCH1CH2	CLK Rise Time	From 1.0 to 3.5V		10		10	ns
TCL2CL1	CLK Fall Time	From 3.5 to 1.0V		10		10	ns
TDVCL	Data in Setup Time		30		20		ns
TCLDX	Data in Hold Time		10		10		ns
TR1VCL	RDY Setup Time into 8284 (See Notes 1, 2)		35		35		ns
TCLR1X	RDY Hold Time into 8284 (See Notes 1, 2)		0		0		ns
TRYHCH	READY Setup Time into 8088		(2/3 TCLCL) −15		(2/3 TCLCL) −15		ns
TCHRYX	READY Hold Time into 8088		30		20		ns
TRYLCL	READY Inactive to CLK (See Note 3)		−8		−8		ns
THVCH	HOLD Setup Time		35		20		ns
TINVCH	INTR, NMI, $\overline{\text{TEST}}$ Setup Time (See Note 2)		30		15		ns
TILIH	Input Rise Time (Except CLK)	From 0.8 to 2.0V		20		20	ns
TIHIL	Input Fall Time (Except CLK)	From 2.0 to 0.8V		12		12	ns

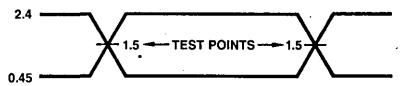
AC CHARACTERISTICS (Cont.)

TIMING RESPONSES

Parameter	Description	Test Conditions	8088		8088-2		Units
			Min	Max	Min	Max	
TCLAV	Address Valid Delay	C_L = 20-100pF for all 8088 outputs (in addition to internal loads)	10	110	10	60	ns
TCLAX	Address Hold Time		10		10		ns
TCLAZ	Address Float Delay		TCLAX	80	TCLAX	50	ns
TLHLL	ALE Width		TCLCH −20		TCLCH −10		ns
TCLLH	ALE Active Delay			80		50	ns
TCHLL	ALE Inactive Delay			85		55	ns
TLLAX	Address Hold Time to ALE Inactive		TCHCL −10		TCHCL −10		ns
TCLDV	Data Valid Delay		10	110	10	60	ns
TCHDX	Data Hold Time		10		10		ns
TWHDX	Data Hold Time After $\overline{WR}$		TCLCH −30		TCLCH −30		ns
TCVCTV	Control Active Delay 1		10	110	10	70	ns
TCHCTV	Control Active Delay 2		10	110	10	60	ns
TCVCTX	Control Inactive Delay		10	110	10	70	ns
TAZRL	Address Float to READ Active		0		0		ns
TCLRL	$\overline{RD}$ Active Delay		10	165	10	100	ns
TCLRH	$\overline{RD}$ Inactive Delay		10	150	10	80	ns
TRHAV	$\overline{RD}$ Inactive to Next Address Active		TCLCL −45		TCLCL −40		ns
TCLHAV	HLDA Valid Delay		10	160	10	100	ns
TRLRH	$\overline{RD}$ Width		2TCLCL −75		2TCLCL −50		ns
TWLWH	$\overline{WR}$ Width		2TCLCL −60		2TCLCL −40		ns
TAVAL	Address Valid to ALE Low		TCLCH −60		TCLCH −40		ns
TOLOH	Output Rise Time	From 0.8 to 2.0V		20		20	ns
TOHOL	Output Fall Time	From 2.0 to 0.8V		12		12	ns

AC TESTING INPUT, OUTPUT WAVEFORM

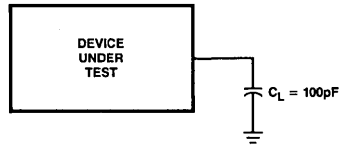
Input/Output



MMC-214

AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." The clock is driven at 4.3V and 0.25V. Timing measurements are made at 1.5V for both a logic "1" and "0."

AC TESTING LOAD CIRCUIT



MMC-215

C_L Includes JIG Capacitance.

AC CHARACTERISTICS (Cont.)

MAX MODE SYSTEM (USING 8288 BUS CONTROLLER) TIMING REQUIREMENTS

Parameter	Description	Test Conditions	8088		8088-2		Units
			Min	Max	Min	Max	
TCLCL	CLK Cycle Period		200	500	125	500	ns
TCLCH	CLK Low Time		(2/3 TCLCL) −15		(2/3 TCLCL) −15		ns
TCHCL	CLK High Time		(1/3 TCLCL) +2		(1/3 TCLCL) +2		ns
TCH1CH2	CLK Rise Time	From 1.0 to 3.5V		10		10	ns
TCL2CL1	CLK Fall Time	From 3.5 to 1.0V		10		10	ns
TDVCL	Data in Setup Time		30		20		ns
TCLDX	Data in Hold Time		10		10		ns
TR1VCL	RDY Setup Time into 8284 (See Notes 1, 2)		35		35		ns
TCLR1X	RDY Hold Time into 8284 (See Notes 1, 2)		0		0		ns
TRYHCH	READY Setup Time into 8088		(2/3 TCLCL) −15		(2/3 TCLCL) −15		ns
TCHRYX	READY Hold Time into 8088		30		20		ns
TRYLCL	READY Inactive to CLK (See Note 4)		−8		−8		ns
TINVCH	Setup Time for Recognition (INTR, NMI, TEST (See Note 2)		30		15		ns
TGVCH	RQ/GT Setup Time		30		15		ns
TCHGX	RQ Hold Time into 8086		40		30		ns
TILIH	Input Rise Time (Except CLK)	From 0.8 to 2.0V		20		20	ns
TIHIL	Input Fall Time (Except CLK)	From 2.0 to 0.8V		12		12	ns

Notes: 1. Signal at 8284 or 8288 shown for reference only.
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.
3. Applies only to T_2 state (8ns into T_3 state).
4. Applies only to T_2 state (8ns into T_3 state).

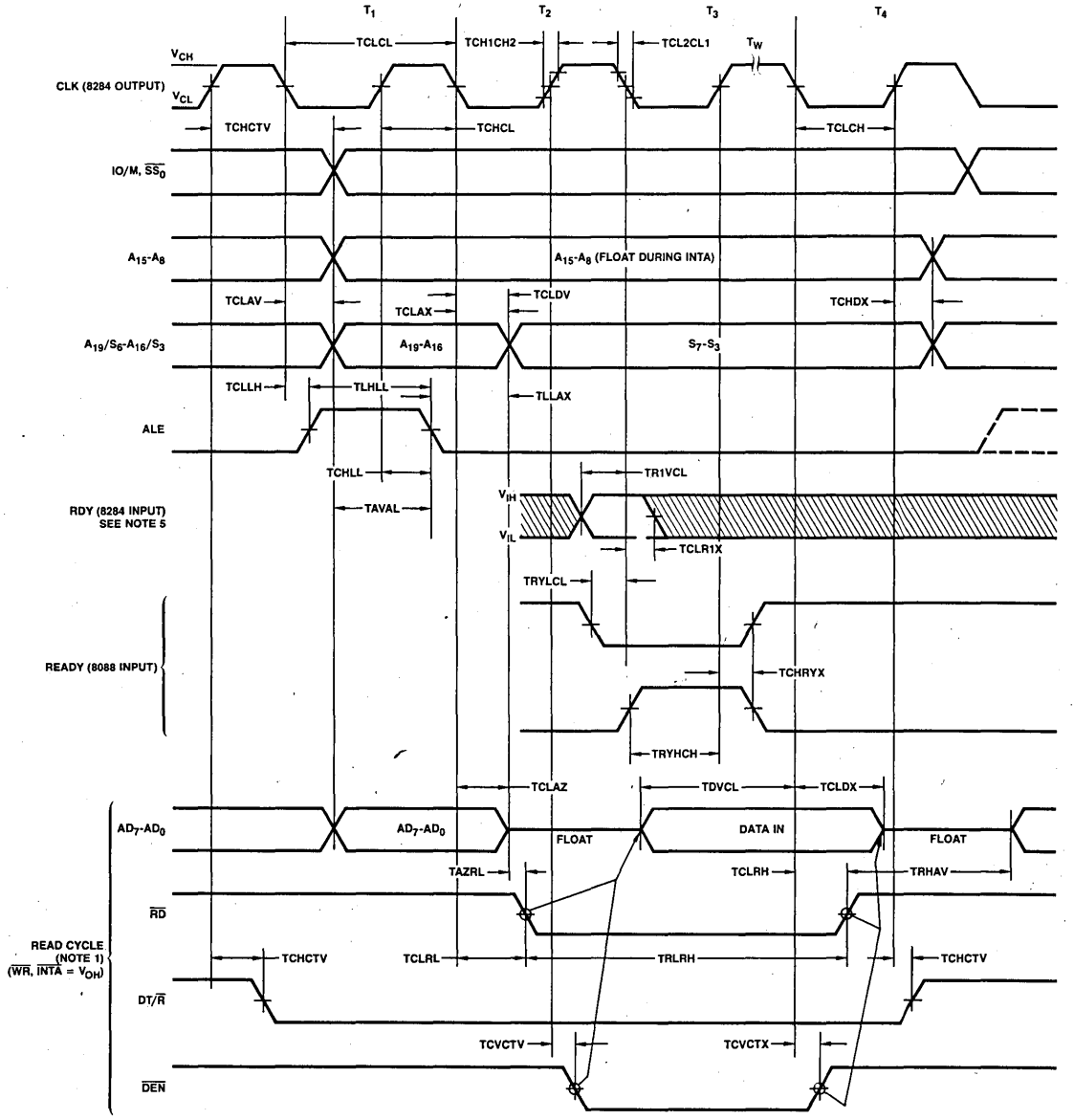
AC CHARACTERISTICS (Cont.)

TIMING RESPONSES

Parameter	Description	Test Conditions	8088		8088-2		Units
			Min	Max	Min	Max	
TCLML	Command Active Delay (See Note 1)		10	35	10	35	ns
TCLMH	Command Inactive Delay (See Note 1)		10	35	10	35	ns
TRYHSH	READY Active to Status Passive (See Note 3)			110		65	ns
TCHSV	Status Active Delay		10	110	10	60	ns
TCLSH	Status Inactive Delay		10	130	10	70	ns
TCLAV	Address Valid Delay		10	110	10	60	ns
TCLAX	Address Hold Time		10		10		ns
TCLAZ	Address Float Delay		TCLAX	80	TCLAX	50	ns
TSVLH	Status Valid to ALE High (See Note 1)			15		15	ns
TSVMCH	Status Valid to MCE High (See Note 1)			15		15	ns
TCLLH	CLK Low to ALE Valid (See Note 1)			15		15	ns
TCLMCH	CLK Low to MCE High (See Note 1)			15		15	ns
TCHLL	ALE Inactive Delay (See Note 1)	C_L = 20-100pF for all 8088 outputs (in addition to internal loads)		15		15	ns
TCLMCL	MCE Inactive Delay (See Note 1)			15		15	ns
TCLDV	Data Valid Delay		10	110	10	60	ns
TCHDX	Data Hold Time		10		10		ns
TCVNV	Control Active Delay (See Note 1)		5	45	5	45	ns
TCVNX	Control Inactive Delay (See Note 1)		10	45	10	45	ns
TAZRL	Address Float to Read Active		0		0		ns
TCLRL	RD Active Delay		10	165	10	100	ns
TCLRH	RD Inactive Delay		10	150	10	80	ns
TRHAV	RD Inactive to Next Address Active		TCLCL −45		TCLCL −40		ns
TCHDTL	Direction Control Active Delay (See Note 1)			50		50	ns
TCHDTH	Direction Control Inactive Delay (See Note 1)			30		30	ns
TCLGL	GT Active Delay			110		50	ns
TCLGH	GT Inactive Delay			85		50	ns
TRLRH	RD Width		2TCLCL −75		2TCLCL −50		ns
TOLOH	Output Rise Time	From 0.8 to 2.0V		20		20	ns
TOHOL	Output Fall Time	From 2.0 to 0.8V		12		12	ns

WAVEFORMS

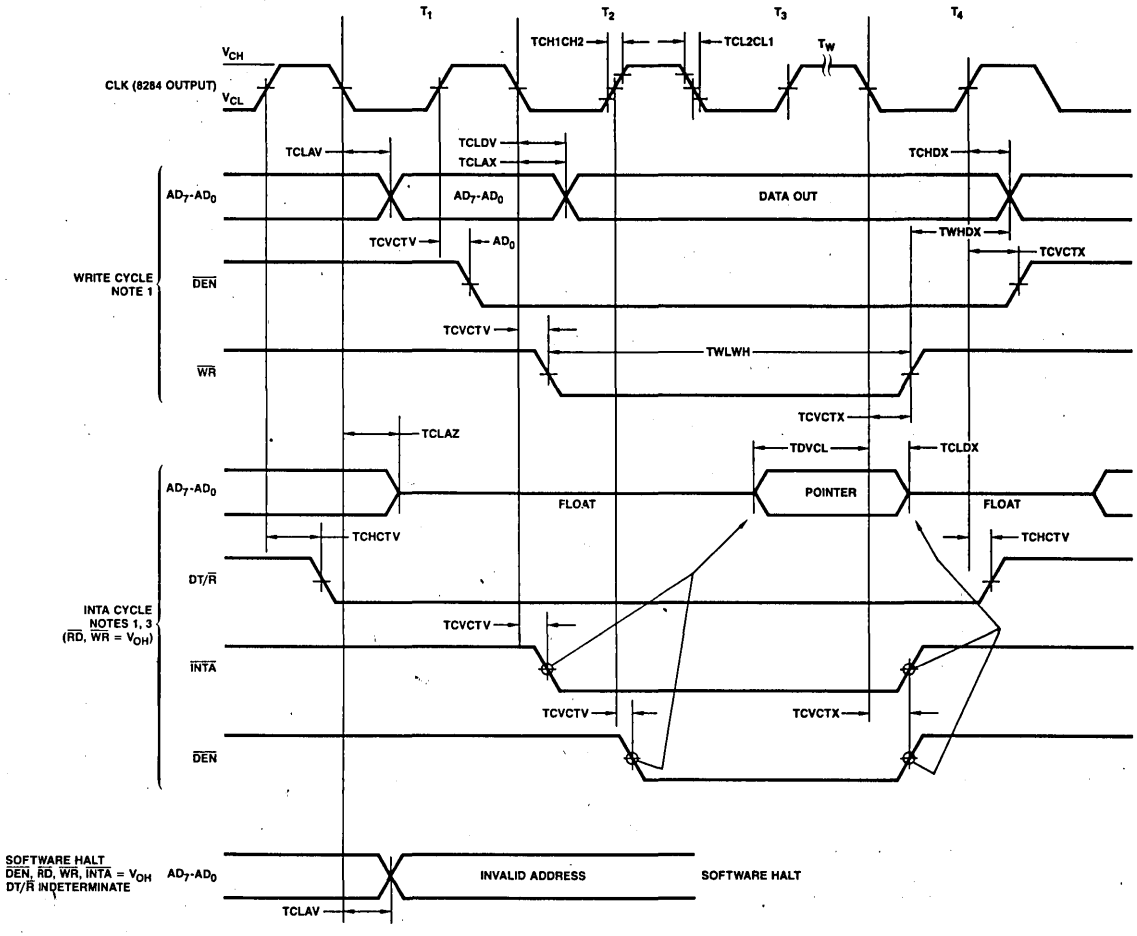
BUS TIMING – MINIMUM MODE SYSTEM



MMC-216

WAVEFORMS (Cont.)

BUS TIMING – MINIMUM MODE SYSTEM (Cont.)

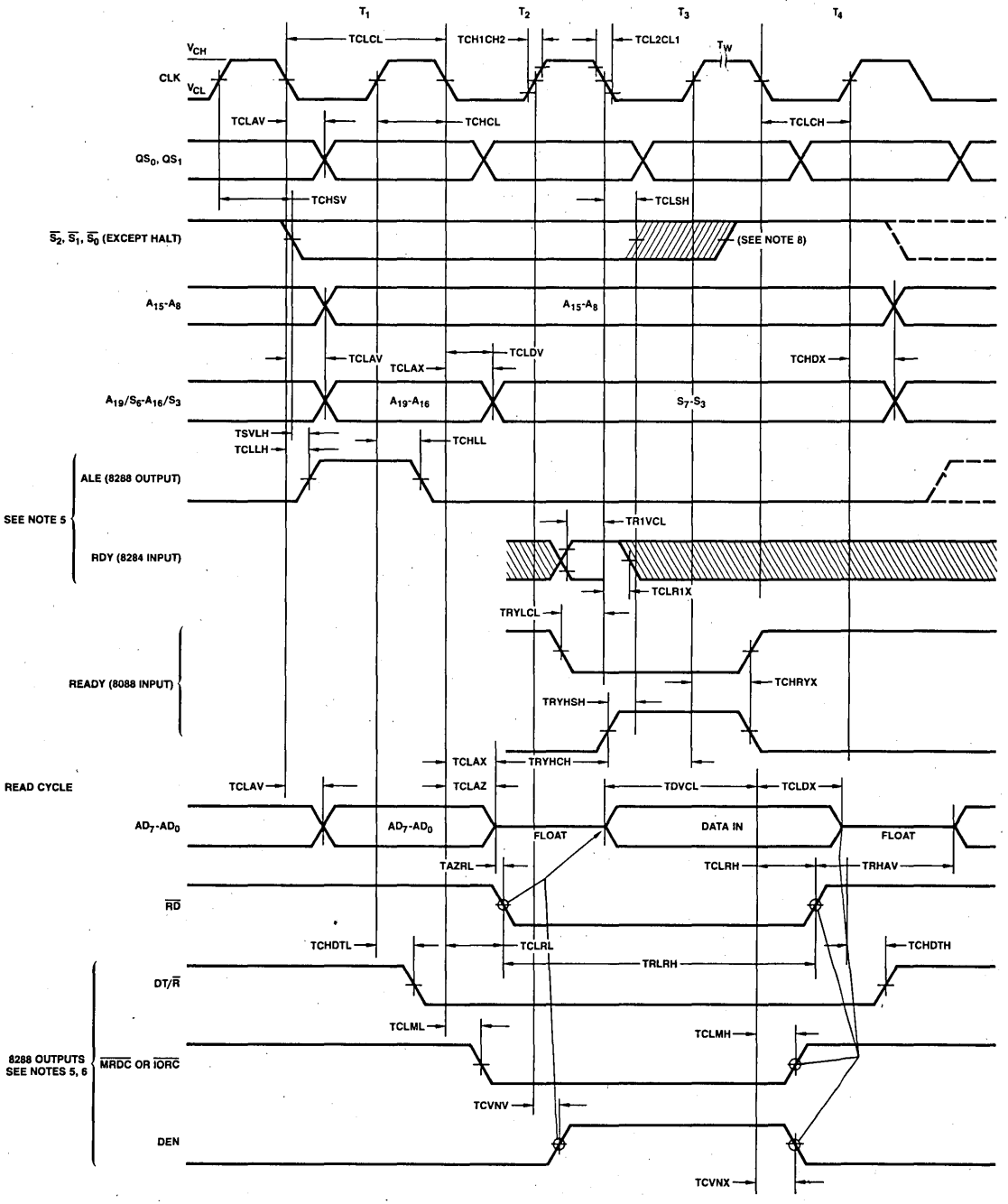


MMC-217

Notes: 1. All signals switch between V_{OH} and V_{OL} unless otherwise specified.
2. RDY is sampled near the end of T_2, T_3, T_W to determine if T_W machines states are to be inserted.
3. Two INTA cycles run back-to-back. The 8088 local ADDR/DATA bus is floating during both INTA cycles. Control signals are shown for the second INTA cycle.
4. Signals at 8284 are shown for reference only.
5. All timing measurements are made at 1.5V unless otherwise noted.

WAVEFORMS (Cont.)

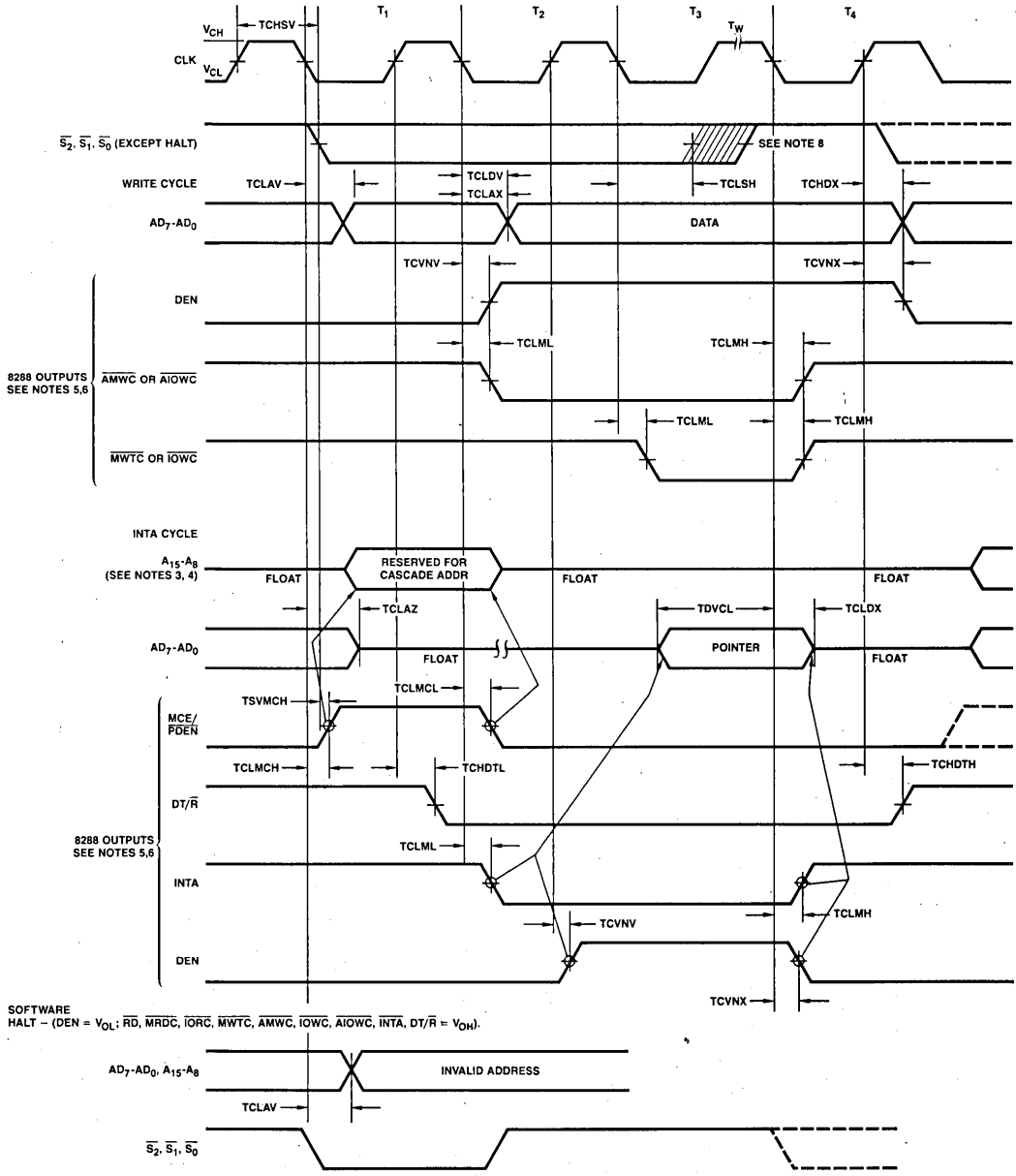
BUS TIMING – MAXIMUM MODE



MMC-218

WAVEFORMS (Cont.)

BUS TIMING – MAXIMUM MODE SYSTEM (USING 8288)

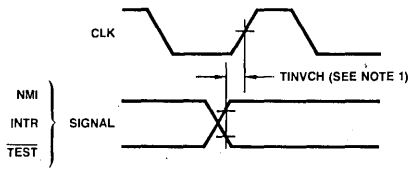


Notes:
1. All signals switch between V_{OH} and V_{OL} unless otherwise specified.
2. RDY is sampled near the end of T_2, T_3, T_W to determine if T_W machines states are to be inserted.
3. Cascade address is valid between first and second INTA cycles.
4. Two INTA cycles run back-to-back. The 8088 local ADDR/DATA bus is floating during both INTA cycles. Control for pointer address is shown for second INTA cycle.
5. Signals at 8284 or 8288 are shown for reference only.
6. The issuance of the 8288 command and control signals ($\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$, $\overline{IORC}$, $\overline{IOWC}$, $\overline{AIOWC}$, $\overline{INTA}$, and DEN) lags the active high 8288 CEN.
7. All timing measurements are made at 1.5V unless otherwise noted.
8. Status inactive in state just prior to T_4.

MMC-219

WAVEFORMS (Cont.)

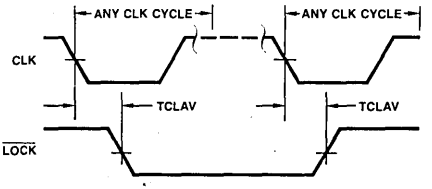
ASYNCHRONOUS SIGNAL RECOGNITION



MMC-220

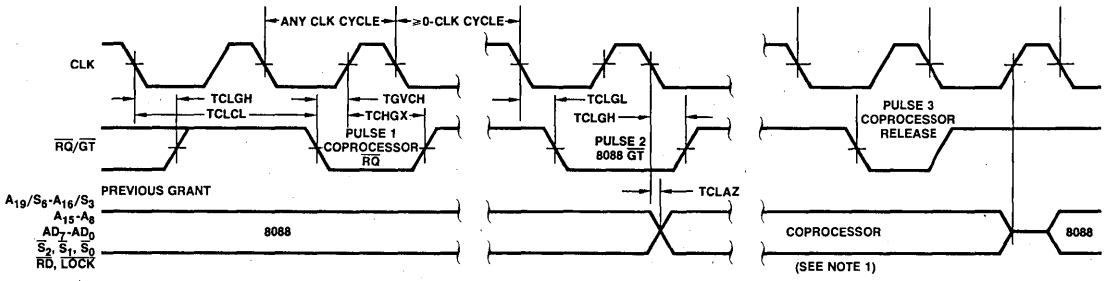
Note: Setup requirements for asynchronous signals only to guarantee recognition at next CLK.

BUS LOCK SIGNAL TIMING (MAXIMUM MODE ONLY)



MMC-221

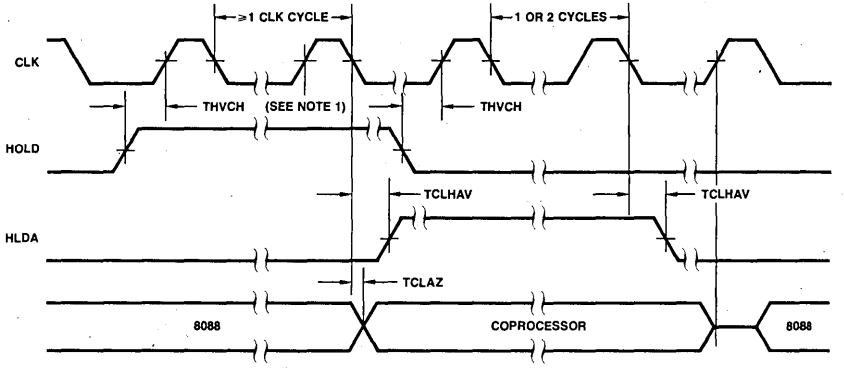
REQUEST/GRANT SEQUENCE TIMING (MAXIMUM MODE ONLY)



MMC-222

Note 1: The coprocessor may not drive the buses outside the region shown without risking contention.

HOLD/HOLD ACKNOWLEDGE TIMING (MINIMUM MODE ONLY)



MMC-223

TABLE 1. PIN DESCRIPTION

The following pin function descriptions are for 8088 systems in either minimum or maximum mode. The "local bus" in these descriptions is the direct multiplexed bus interface connection to the 8088 (without regard to additional bus buffers).

Symbol	Pin No.	Type	Name and Function
AD_7-AD_0	9-16	I/O	**Address Data Bus:** These lines constitute the time multiplexed memory/IO address (T_1) and data (T_2, T_3, T_W, and T_4) bus. These lines are active HIGH and float to 3-state OFF during interrupt acknowledge and local bus "hold acknowledge."
$A_{15}-A_8$	2-8, 39	O	**Address Bus:** These lines provide address bits 8 through 15 for the entire bus cycle (T_1-T_4). These lines do not have to be latched by ALE to remain valid. $A_{15}-A_8$ are active HIGH and float to 3-state OFF during interrupt acknowledge and local bus "hold acknowledge."
A_{19}/S_6, A_{18}/S_5, A_{17}/S_4 A_{16}/S_3	34-38	O	**Address/Status:** During T_1, these are the four most significant address lines for memory operations. During I/O operations, these lines are LOW. During memory and I/O operations, status information is available on these lines during T_2, T_3, T_W, and T_4. S_6 is always LOW. The status of the interrupt enable flag bit (S_5) is updated at the beginning of each clock cycle. S_4 and S_3 are encoded as shown. This information indicates which segment register is presently being used for data accessing. These lines float to 3-state OFF during local bus "hold acknowledge."
$\overline{RD}$	32	O	**Read:** Read strobe indicates that the processor is performing a memory or I/O read cycle, depending on the state of the IO/$\overline{M}$ pin or S_2. This signal is used to read devices which reside on the 8088 local bus. $\overline{RD}$ is active LOW during T_2, T_3 and T_W of any read cycle, and is guaranteed to remain HIGH in T_2 until the 8088 local bus has floated. This signal floats to 3-state OFF in "hold acknowledge."
READY	22	I	**READY:** is the acknowledgement from the addressed memory or I/O device that it will complete the data transfer. The RDY signal from memory or I/O is synchronized by the 8284 clock generator to form READY. This signal is active HIGH. The 8088 READY input is not synchronized. Correct operation is not guaranteed if the setup and hold times are not met.
INTR	18	I	**Interrupt Request:** is a level triggered input which is sampled during the last clock cycle of each instruction to determine if the processor should enter into an interrupt acknowledge operation. A subroutine is vectored to via an interrupt vector lookup table located in system memory. It can be internally masked by software resetting the interrupt enable bit. INTR is internally synchronized. This signal is active HIGH.
$\overline{TEST}$	23	I	**$\overline{TEST}$:** input is examined by the "wait for test" instruction. If the $\overline{TEST}$ input is LOW, execution continues, otherwise the processor waits in an "idle" state. This input is synchronized internally during each clock cycle on the leading edge of CLK.
NMI	17	I	**Non-Maskable Interrupt:** is an edge triggered input which causes a type 2 interrupt. A subroutine is vectored to via an interrupt vector lookup table located in system memory. NMI is not maskable internally by software. A transition from a LOW to HIGH initiates the interrupt at the end of the current instruction. This input is internally synchronized.
RESET	21	I	**RESET:** causes the processor to immediately terminate its present activity. The signal must be active HIGH for at least four clock cycles. It restarts execution, as described in the instruction set description, when RESET returns LOW. RESET is internally synchronized.
CLK	19	I	**Clock:** provides the basic timing for the processor and bus controller. It is asymmetric with a 33% duty cycle to provide optimized internal timing.
V_{CC}	40		**V_{CC}:** is the +5V ±10% power supply pin.
GND	1, 20		**GND:** are the ground pins.
MN/$\overline{MX}$	33	I	**Minimum/Maximum:** indicates what mode the processor is to operate in. The two modes are discussed in the following sections.

S_4	S_3	Characteristics
0 (LOW)	0	Alternate Data
0	1	Stack
1 (HIGH)	0	Code or None
1	1	Data
S_6 is 0 (LOW)		

TABLE 1. PIN DESCRIPTION (Cont.)

The following pin function descriptions are for the 8088 minimum mode (i.e., MN/MX = V_{CC}). Only the pin functions which are unique to minimum mode are described; all other pin functions are as described above.

Symbol	Pin No.	Type	Name and Function
IO/$\overline{M}$	28	O	**Status Line:** is an inverted maximum mode $\overline{S_2}$. It is used to distinguish a memory access from an I/O access. IO/$\overline{M}$ becomes valid in the T_4 preceding a bus cycle and remains valid until the final T_4 of the cycle (I/O = HIGH, M = LOW). IO/$\overline{M}$ floats to 3-state OFF in local bus "hold acknowledge."
$\overline{WR}$	29	O	**Write:** strobe indicates that the processor is performing a write memory or write I/O cycle, depending on the state of the IO/$\overline{M}$ signal. WR is active for T_2, T_3, and T_W of any write cycle. It is active LOW, and floats to 3-state OFF in local bus "hold acknowledge."
$\overline{INTA}$	24	O	**INTA:** is used as a read strobe for interrupt acknowledge cycles. It is active LOW during T_2, T_3 and T_W of each interrupt acknowledge cycle.
ALE	25	O	**Address Latch Enable:** is provided by the processor to latch the address into 8282/8283 address latch. It is a HIGH pulse active during clock low of T_1 of any bus cycle. Note that ALE is never floated.
DT/$\overline{R}$	27	O	**Data Transmit/Receive:** is needed in a minimum system that desires to use an 8286/8287 data bus transceiver. It is used to control the direction of data flow through the transceiver. Logically DT/$\overline{R}$ is equivalent to $\overline{S_1}$ in the maximum mode, and its timing is the same as for IO/$\overline{M}$ (T = HIGH, R = LOW.) This signal floats to 3-state OFF in local bus "hold acknowledge."
$\overline{DEN}$	26	O	**Data Enable:** is provided as an output enable for the 8286/8287 in a minimum system which uses the transceiver. $\overline{DEN}$ is active LOW during each memory and I/O access and for $\overline{INTA}$ cycles. For a read or $\overline{INTA}$ cycle, it is active from the middle of T_2 until the middle of T_4, while for a write cycle, it is active from the beginning of T_2 until the middle of T_4. $\overline{DEN}$ floats to 3-state OFF during local bus "hold acknowledge."
HOLD, HLDA	30, 31	I, O	**HOLD:** indicates that another master is requesting a local bus "hold." To be acknowledged, HOLD must be active HIGH. The processor receiving the "hold" request will issue HLDA (HIGH) as an acknowledgement, in the middle of a T_4 or T_1 clock cycle. Simultaneous with the issuance of HLDA the processor will float the local bus and control lines. After HOLD is detected as being LOW, the processor lowers HLDA, and when the processor needs to run another cycle, it will again drive the local bus and control lines. Hold is not an asynchronous input. External synchronization should be provided if the system cannot otherwise guarantee the setup time.
$\overline{SSO}$	34	O	**Status Line:** is logically equivalent to $\overline{SO}$ in the maximum mode. The combination of $\overline{SSO}$, IO/$\overline{M}$ and DT/$\overline{R}$ allows the system to completely decode the current bus cycle status.

IO/$\overline{M}$	DT/$\overline{R}$	$\overline{SSO}$	Characteristics
1 (HIGH)	0	0	Interrupt Acknowledge
1	0	1	Read I/O port
1	1	0	Write I/O port
1	1	1	Halt
0 (LOW)	0	0	Code Access
0	0	1	Read memory
0	1	0	Write memory
0	1	1	Passive

TABLE 1. PIN DESCRIPTION (Cont.)

The following pin function descriptions are for the 8088, 8228 system in maximum mode (i.e., MN/MX = GND). Only the pin functions which are unique to maximum mode are described; all other pin functions are as described above.

Symbol	Pin No.	Type	Name and Function
$\overline{S_2}, \overline{S_1}, \overline{S_0}$	26-28	O	**Status:** is active during clock high of T_4, T_1, and T_2 and is returned to the passive state (1, 1, 1) during T_3 or during T_W when READY is HIGH. This status is used by the 8288 bus controller to generate all memory and I/O access control signals. Any change by $\overline{S_2}$, $\overline{S_1}$, or $\overline{S_0}$ during T_4 is used to indicate the beginning of a bus cycle, and the return to the passive state in T_3 or T_W is used to indicate the end of a bus cycle. These signals float to 3-state OFF during "hold acknowledge". During the first clock cycle after RESET becomes active, these signals are active HIGH. After this first clock, they float to 3-state OFF.
$\overline{RQ}/\overline{GT_0}$, $\overline{RQ}/\overline{GT_1}$	30, 31	I/O	**Request/Grant:** pins are used by other local bus masters to force the processor to release the local bus at the end of the processor's current bus cycle. Each pin is bidirectional with $\overline{RQ}/\overline{GT_0}$ having higher priority than $\overline{RQ}/\overline{GT_1}$. $\overline{RQ}/\overline{GT}$ has an internal pull-up resistor, so may be left unconnected. The request/grant sequence is as follows (See Figure 8): 1. A pulse of one CLK wide from another local bus master indicates a local bus request ("hold") to the 8088 (pulse 1). 2. During a T_4 or T_1 clock cycle, a pulse one clock wide from the 8088 to the requesting master (pulse 2), indicates that the 8088 has allowed the local bus to float and that it will enter the "hold acknowledge" state at the next CLK. The CPU's bus interface unit is disconnected logically from the local bus during "hold acknowledge." The same rules as for HOLD/HOLDA apply as for when the bus is released. 3. A pulse one CLK wide from the requesting master indicates to the 8088 (pulse 3) that the "hold" request is about to end and that the 8088 can reclaim the local bus at the next CLK. The CPU then enters T_4. Each master-master exchange of the local bus is a sequence of three pulses. There must be one idle CLK cycle after each bus exchange. Pulses are active LOW. If the request is made while the CPU is performing a memory cycle, it will release the local bus during T_4 of the cycle when all the following conditions are met: 1. Request occurs on or before T_2. 2. Current cycle is not the low bit of a word. 3. Current cycle is not the first acknowledge of an interrupt acknowledge sequence. 4. A locked instruction is not currently executing. If the local bus is idle when the request is made the two possible events will follow: 1. Local bus will be released during the next clock. 2. A memory cycle will start within 3 clocks. Now the four rules for a currently active memory cycle apply with condition number 1 already satisfied.
$\overline{LOCK}$	29	O	**$\overline{LOCK}$:** indicates that other system bus masters are not to gain control of the system bus while $\overline{LOCK}$ is active (LOW). The $\overline{LOCK}$ signal is activated by the "LOCK" prefix instruction and remains active until the completion of the next instruction. This signal is active LOW, and floats to 3-state off in "hold acknowledge."
QS_1, QS_0	24, 25	O	**Queue Status:** provide status to allow external tracking of the internal 8088 instruction queue. The queue status is valid during the CLK cycle after which the queue operation is performed.
–	34	O	Pin 34 is always HIGH in the maximum mode.

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Characteristics
0 (LOW)	0	0	Interrupt Acknowledge
0	0	1	Read I/O Port
0	1	0	Write I/O Port
0	1	1	Halt
1 (HIGH)	0	0	Code Access
1	0	1	Read Memory
1	1	0	Write Memory
1	1	1	Passive

QS_1	QS_0	Characteristics
0 (LOW)	0	No Operation
0	1	First Byte of Opcode from Queue
1 (HIGH)	0	Empty the Queue
1	1	Subsequent Byte from Queue

8089

I/O Processor
iAPX86 Family

DISTINCTIVE CHARACTERISTICS

- High speed DMA capability
- Two DMA channels
- Removes I/O software overhead from 8086/8088
- 1 megabyte addressability
- 1.25 Mbyte/sec transfer rate
- Memory-based communication with CPU
- Allows mixed interface of 8- and 16-bit peripherals
- LOCAL or REMOTE I/O processing modes
- Multibus compatible system interface

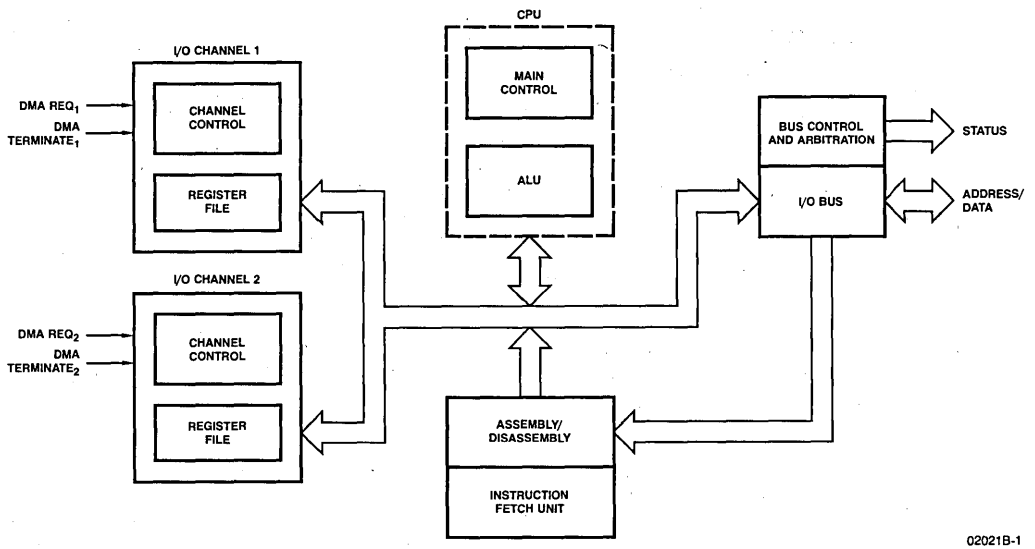
GENERAL DESCRIPTION

The 8089 is a high performance I/O processor designed for the 8086 Family. It supports versatile DMA functions and maintains peripheral components, to offload I/O overhead from the CPU. The IOP communicates with the CPU through shared memory blocks and 2 control lines.

The 8089 IOP can operate in LOCAL mode sharing the same bus and buffer with the CPU, or in REMOTE mode on a separate local bus. In REMOTE mode the 8089 is compatible with any 8080 or 8085 CPU as well.

The 8089 IOP is particularly effective in I/O intensive applications like filter and buffer management, CRT control, and other communications tasks. It is implemented in N-channel depletion load silicon, packaged in a 40-pin DIP.

BLOCK DIAGRAM



ORDERING INFORMATION

Package Type	Ambient Temperature Spec	Order Number
Hermetic Dip	$0°C \leq T_A \leq 70°C$	D8089-3

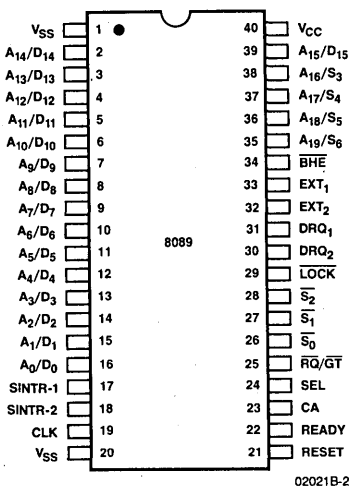
Multibus is a trademark of Intel Corp.

02021B-MMP

TABLE 1. PIN DESCRIPTION

Symbol	Type	Name and Function
A_0-A_{15}/ D_0-D_{15}	I/O	**Multiplexed Address and Data Bus:** The function of these lines are defined by the state of $\overline{S}_0$, $\overline{S}_1$ and $\overline{S}_2$ lines.
A_{16}-A_{19}/ S_3-S_6	O	**Address and Status:** Multiplexed most significant address lines and status information. The address lines are active only when addressing memory. Otherwise, the status lines are active.
$\overline{BHE}$	O	**Bus High Enable:** The Bus High Enable is used to enable data operations on the most significant half of the data bus (D_8-D_{15}).
$\overline{S}_0$, $\overline{S}_1$, $\overline{S}_2$	O	**Status:** These are the status pins that define the IOP activity during any given cycle. The status lines are utilized by the bus controller and bus arbiter to generate all memory and I/O control signals.
READY	I	**Ready:** The ready signal received from the addressed device indicates that the device is ready for data transfer.
$\overline{LOCK}$	O	**Lock:** The lock output signal indicates to the bus controller that the bus is needed for more than one contiguous cycle.
RESET	I	**Reset:** The receipt of a reset signal causes the IOP to suspend all its activities and enter an idle state until a channel attention is received.
CLK	I	**Clock:** Clock provides all timing needed for internal IOP operation.
CA	I	**Channel Attention:** Gets the attention of the IOP
SEL	I	**Select:** The first CA received after system reset informs the IOP via the SEL line, whether it is a Master or Slave and starts the initialization sequence.
DRQ1-2	I	**Data Request:** DMA request inputs which signal the IOP that a peripheral is ready to transfer/receive data.
$\overline{RQ/GT}$	I/O	**Request Grant:** Request Grant implements the communication dialogue required to arbitrate the use of the system bus or I/O bus.
SINTR1-2	O	**Signal Interrupt:** Signal Interrupt outputs from channels 1 and 2 respectively.
EXT1-2	I	**External Terminate:** External terminate inputs for channels 1 and 2 respectively.
V_{CC}		**Voltage:** +5 volt power input.
V_{SS}		**Ground.**

CONNECTION DIAGRAM
Top View
D-40



Note: Pin 1 is marked for orientation.

OPERATING MODES

Shown in Figure 3 is the 8089 in a LOCAL configuration. The 8086/88 CPU is used in its maximum mode. The 8089 and CPU reside on the same local bus, sharing the same set of system buffers. Peripherals located on the system bus can be addressed by either the CPU or the 8089. The 8089 requests the use of the LOCAL bus by means of the RQ/GT line. When the CPU relinquishes the system bus, the 8089 uses the same bus control, latches and transceiver components to generate the system address, control and data lines.

A typical REMOTE configuration is shown in Figure 4. In this mode, the IOP's bus is physically separated from the system bus by means of system buffers/latches. The IOP maintains its own local bus and can operate out of local or system memory. The system bus interface contains the following components: 8282 latches, 8286 transceiver, 8288 bus controller, and 8283 bus arbiter.

The peripheral devices PER_1 and PER_2 are supported on their own data and address bus. The 8089 communicates with the peripherals without affecting system bus operation. Optional buffers may be used on the local bus when capacitive loading conditions so dictate.

COMMUNICATION MECHANISM

Communication between the CPU and IOP is performed through messages prepared in shared memory. The CPU can cause the 8089 to execute a program by placing it in the 8089's memory space and/or directing the 8089's attention to it by asserting a hardware Channel Attention (CA) signal to the IOP, activating the proper I/O channel. Communication from the IOP to the processor can be performed in a similar manner via a system interrupt (SINTR 1, 2); if the CPU has enabled interrupts for this purpose.

The Control Block furnishes bus control initialization for the IOP operation (CCW or Channel Control Word) and provides pointers to the Parameter Block or "data" memory for both channels 1 and 2.

The Parameter Block contains the address of the Task Block and acts as a message center between the IOP and CPU. Parameters or variable information is passed from the CPU to its IOP in this block to customize the software interface to the peripheral device. It is also used for transferring data and status information between the IOP and CPU.

The Task Block contains the instructions for the respective channel. This block can reside on the local bus of the IOP, allowing the IOP to operate concurrently with the CPU, or reside in system memory.

Register Set

The 8089 maintains separate registers for its two I/O channels as well as some common registers (see Figure 6). There are sufficient registers for each channel to sustain its own DMA transfers, and process its own instruction stream.

Bus Operation

The 8089 utilizes the same bus structure as the 8086 and 8088 in their maximum mode configurations (see Figure 7).

The data bandwidth of the IOP is a function of the physical bus width of the system and I/O busses. Table 2 gives the bandwidth, latency and bus utilization of the 8089. The system bus is assumed to be 16-bits wide with either an 8-bit peripheral (under byte column) or 16-bit peripheral (word column) being shown.

The latency refers to the worst case response time by the IOP to a DMA request, without the bus arbitration times. Notice that the word transfer allows 50% more bandwidth. This occurs since three bus cycles are required to map 8-bit data into a 16-bit location, versus two for a 16-bit to 16-bit transfer. Note that it is possible to fully saturate the system bus in the LOCAL mode whereas in the REMOTE mode this is reduced to a maximum of 50%.

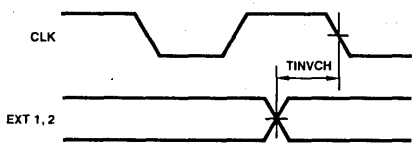
TABLE 2. ACHIEVABLE 5MHz 8089 OPERATIONS

	Local		Remote	
	Byte	Word	Byte	Word
Bandwidth	830 KB/S	1250 KB/S	830 KB/S	1250 KB/S
Latency	1.0/2.4 μsec*	1.0/2.4 μsec*	1.0/2.4 μsec*	1.0/2.4 μsec*
System Bus Utilization	2.4μsec Per Transfer	1.6μsec Per Transfer	0.8μsec Per Transfer	0.8μsec Per Transfer

*2.4μsec if interleaving with other channel and no wait states. 1μsec if channel is waiting for request.

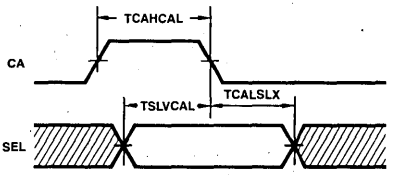
WAVEFORMS

EXTERNAL TERMINATE SETUP



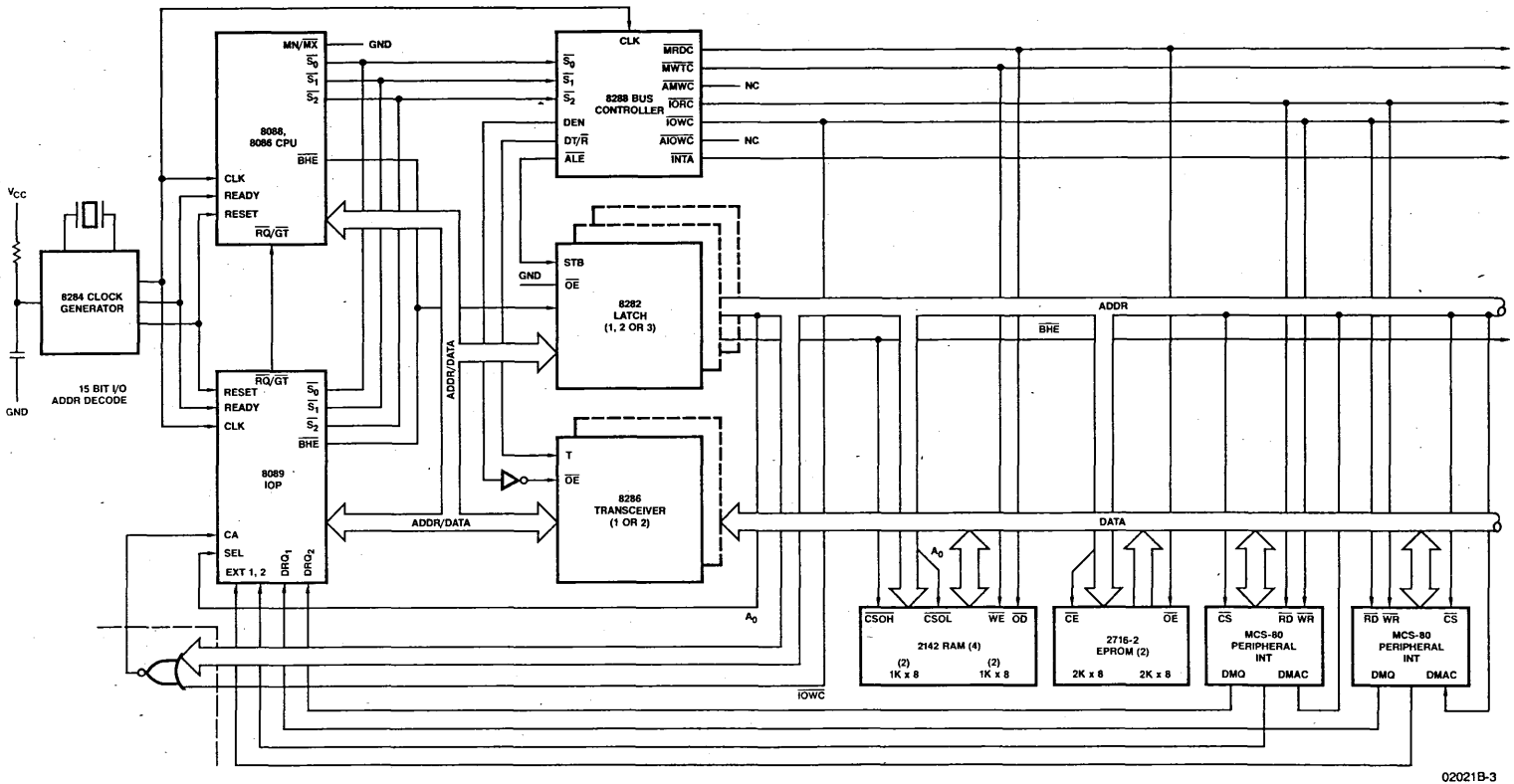
02021B-15

SEL SETUP AND TIMING



02021B-16

Figure 3. Typical Configuration with 8089 in LOCAL Mode, 8088, 8086 in MAX Mode



Note: Only one latch is needed if configured with 8086 and only 64K addressing is used. Only one transceiver is needed if using a physical 8-bit data bus (8088).

Figure 4. Typical REMOTE Configuration

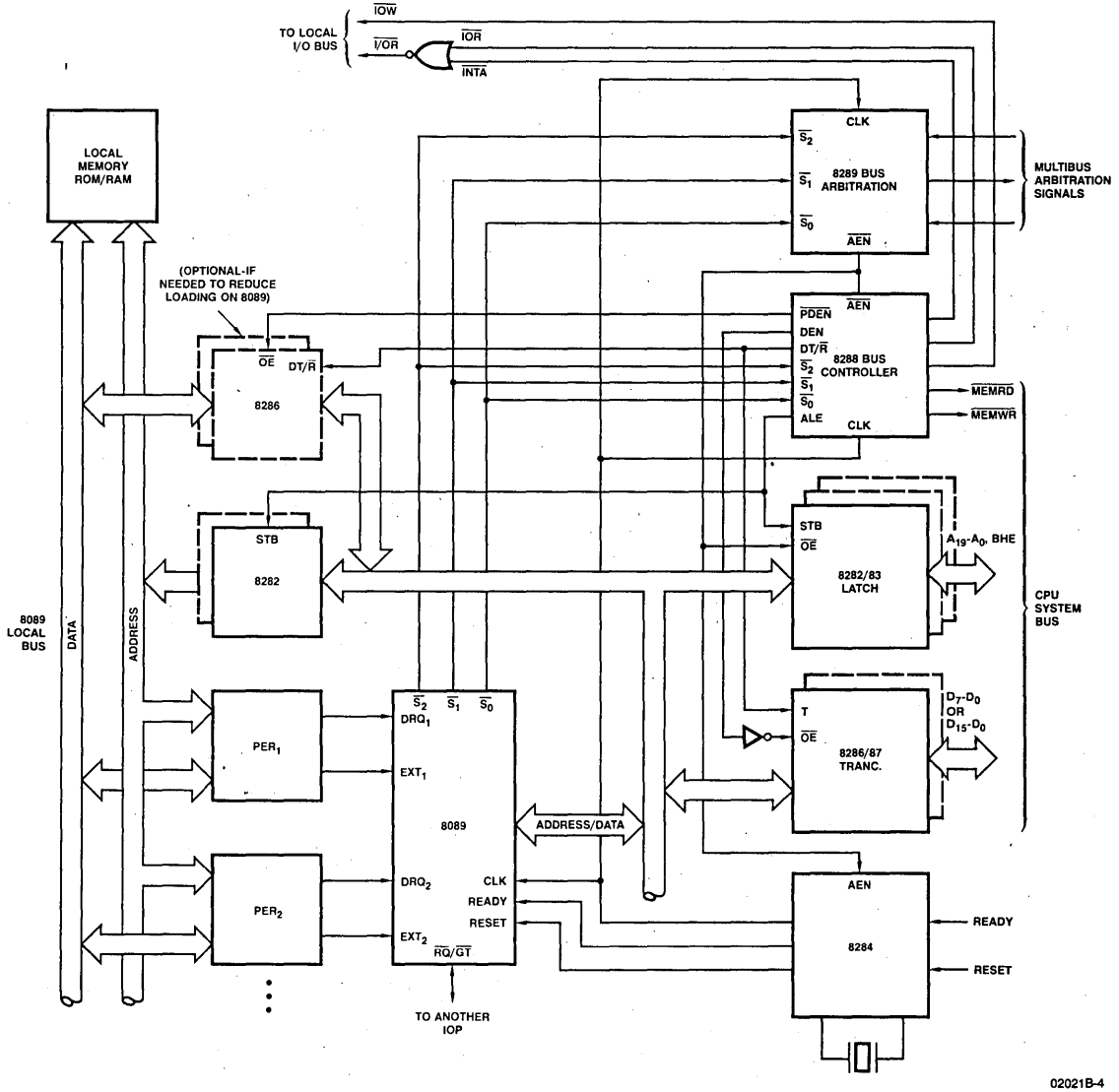


Figure 5. Communication Data Structure Hierarchy

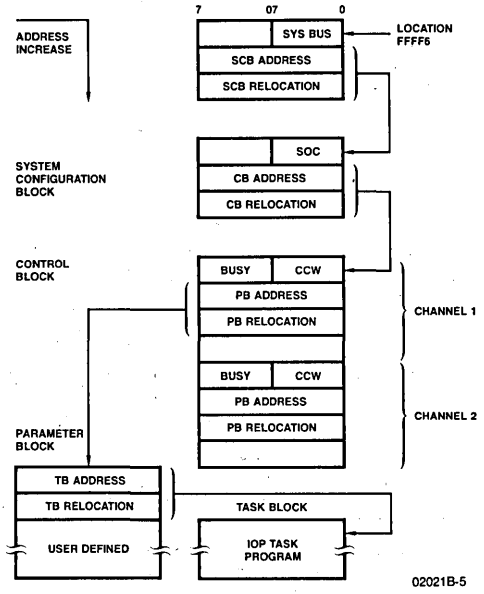


Figure 6. Register Model

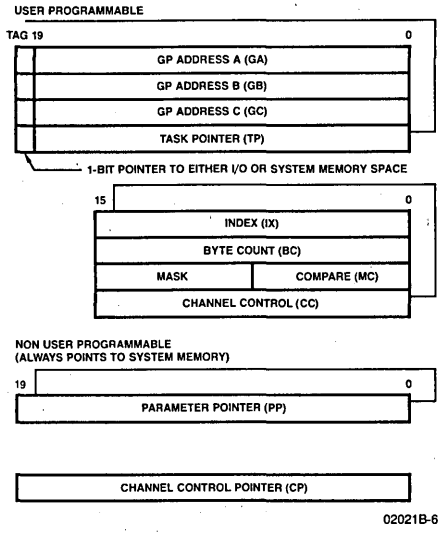
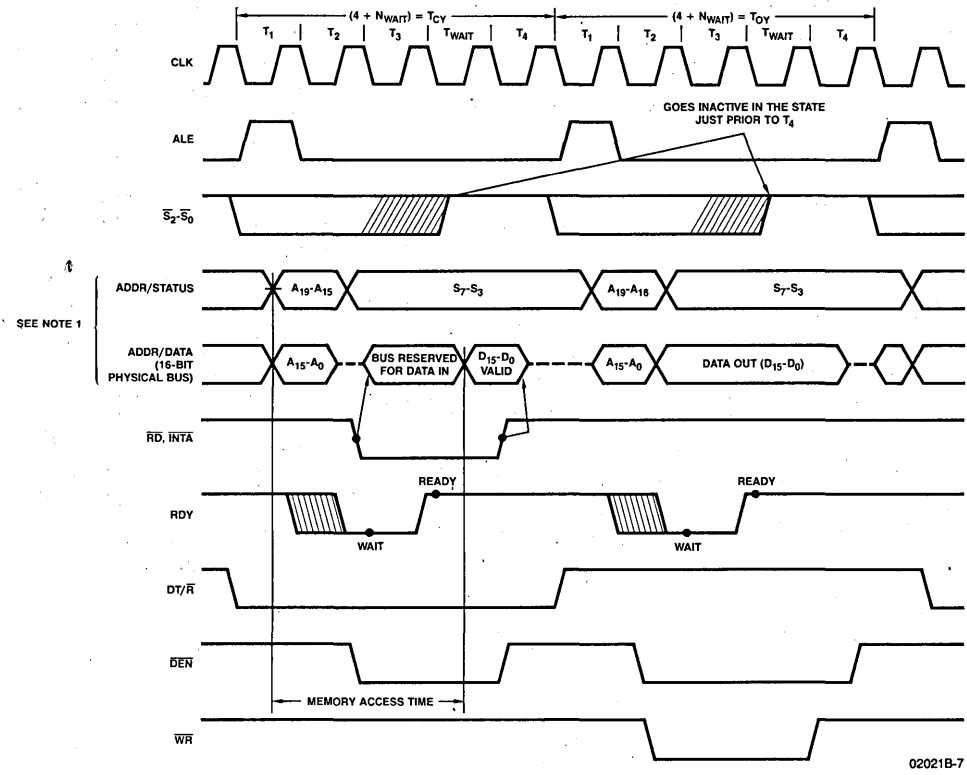


Figure 7. 8089 Bus Operation



Note: 1. BHE is stable (i.e., non multiplexed) throughout each transfer cycle. A_8-A_{15} are also stable on transfers to a physical 8-bit bus.

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias	0 to 70°C
Storage Temperature	−65 to +150°C
Voltage on Any Pin with Respect to Ground	−1.0 to +7V
Power Dissipation	2.5 Watt

*Notice: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS (T_A = 0 to 70°C, V_{CC} = 5V ±10%)

Parameter	Description	Test Conditions	Min	Max	Units
V_{IL}	Input Low Voltage		−0.5	+0.8	V
V_{IH}	Input High Voltage		2.0	V_{CC} +1.0	V
V_{OL}	Output Low Voltage	I_{OL} = 2.0mA		0.45	V
V_{OH}	Output High Voltage	I_{OH} = −400μA	2.4		V
I_{CC}	Power Supply Current	T_A = 25°C		350	mA
I_{LI}	Input Leakage Current (See Note 1)	0V < V_{IN} < V_{CC}		±10	μA
I_{LO}	Output Leakage Current	0.45V ≤ V_{OUT} ≤ V_{CC}		±10	μA
V_{CL}	Clock Input Low Voltage		−0.5	+0.6	V
V_{CH}	Clock Input High Voltage		3.9	V_{CC} +1.0	V
C_{IN}	Capacitance of Input Buffer (All input except AD_0-AD_{15}, $\overline{RQ}/\overline{GT}$)	fc = 1MHz		15	pF
C_{IO}	Capacitance of I/O Buffer (AD_0-AD_{15}, $\overline{RQ}/\overline{GT}$)	fc = 1MHz		15	pF

AC CHARACTERISTICS (T_A = 0 to 70°C, V_{CC} = 5V ±10%)

8089/8086 MAX MODE SYSTEM (USING 8288 BUS CONTROLLER) TIMING REQUIREMENTS

Parameter	Description	Test Conditions	Min	Max	Units
TCLCL	CLK Cycle Period		200	500	ns
TCLCH	CLK Low Time		(2/3 TCLCL) −15		ns
TCHCL	CLK High Time		(1/3 TCLCL) +2		ns
TCH1CH2	CLK Rise Time	From 1.0 to 3.5V		10	ns
TCL2CL1	CLK Fall Time	From 3.5 to 1.0V		10	ns
TDVCL	Data In Setup Time		30		ns
TCLDX	Data In Hold Time		10		ns
TR1VCL	RDY Setup Time into 8284 (See Notes 1, 2)		35		ns
TCLR1X	RDY Hold Time into 8284 (See Notes 1, 2)		0		ns
TRYHCH	READY Setup Time into 8089		(2/3 TCLCL) −15		ns
TCHRYX	READY Hold Time into 8089		30		ns
TRYLCL	READY Inactive to CLK (See Note 4)		−8		ns
TINVCH	Setup Time Recognition (DRQ 1, 2 RESET, Ext 1, 2) (See Note 2)		30		ns
TGVCH	$\overline{RQ}/\overline{GT}$ Setup Time		30		ns
TCAHCAL	CA Width		95		ns
TSLVCAL	SEL Setup Time		75		ns
TCALSLX	SEL Hold Time		0		ns
TCHGX	GT Hold Time into 8089		40		ns
TILIH	Input Rise Time (Except CLK)	From 0.8 to 2.0V		20	ns
TIHIL	Input Fall Time (Except CLK)	From 2.0 to 0.8V		12	ns

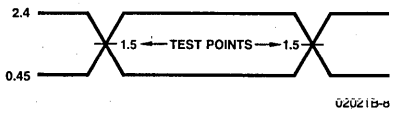
AC CHARACTERISTICS (Cont.) (T_A = 0 to 70°C, V_{CC} = 5V ±10%)

TIMING RESPONSES

Parameter	Description	Test Conditions	Min	Max	Units
TCLML	Command Active Delay (See Note 1)	C_L = 80pF	10	35	ns
TCLMH	Command Inactive Delay (See Note 1)	C_L = 150pF	10	35	ns
TRYHSH	READY Active to Status Passive (See Note 3)			110	ns
TCHSV	Status Active Delay		10	110	ns
TCLSH	Status Inactive Delay		10	130	ns
TCLAV	Address Valid Delay		10	110	ns
TCLAX	Address Hold Time		10		ns
TCLAZ	Address Float Delay		TCLAX	80	ns
TSVLH	Status Valid to ALE High (See Note 1)			15	ns
TCLLH	CLK Low to ALE Valid (See Note 1)			15	ns
TCHLL	ALE Inactive Delay (See Note 1)			15	ns
TCLDV	Data Valid Delay		10	110	ns
TCHDX	Data Hold Time		10		ns
TCVNV	Control Active Delay (See Note 1)		5	45	ns
TCVNX	Control Inactive Delay (See Note 1)		10	45	ns
TCHDTL	Direction Control Active Delay (See Note 1)			50	ns
TCHDTH	Direction Control Inactive Delay (See Note 1)			30	ns
TCLGL	$\overline{RQ}$ Active Delay	C_L = 100pF	0	85	ns
TCLGH	$\overline{RQ}$ Inactive Delay	Note 5: C_L = 30pF		85	ns
TCLSRV	SINTR Valid Delay	C_L = 100pF		150	ns
TOLOH	Output Rise Time	From 0.8 to 2.0V		20	ns
TOHOL	Output Fall Time	From 2.0 to 0.8V		12	ns

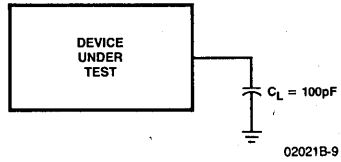
Notes: 1. Signal at 8284 or 8288 shown for reference only.
2. Setup requirement for asynchronous signal only to guarantee recognition at next CLK.
3. Applies only to T_3 and T_W states.
4. Applies only to T_2 state.
5. Applies only if RQ/GT mode 1 C_L = 30pF, 2.7kΩ pull up to V_{CC}.

AC TESTING INPUT, OUTPUT WAVEFORM



AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." The clock is driven at 4.3V and 0.25V. Timing measurements are made at 1.5V for both a logic "1" and "0."

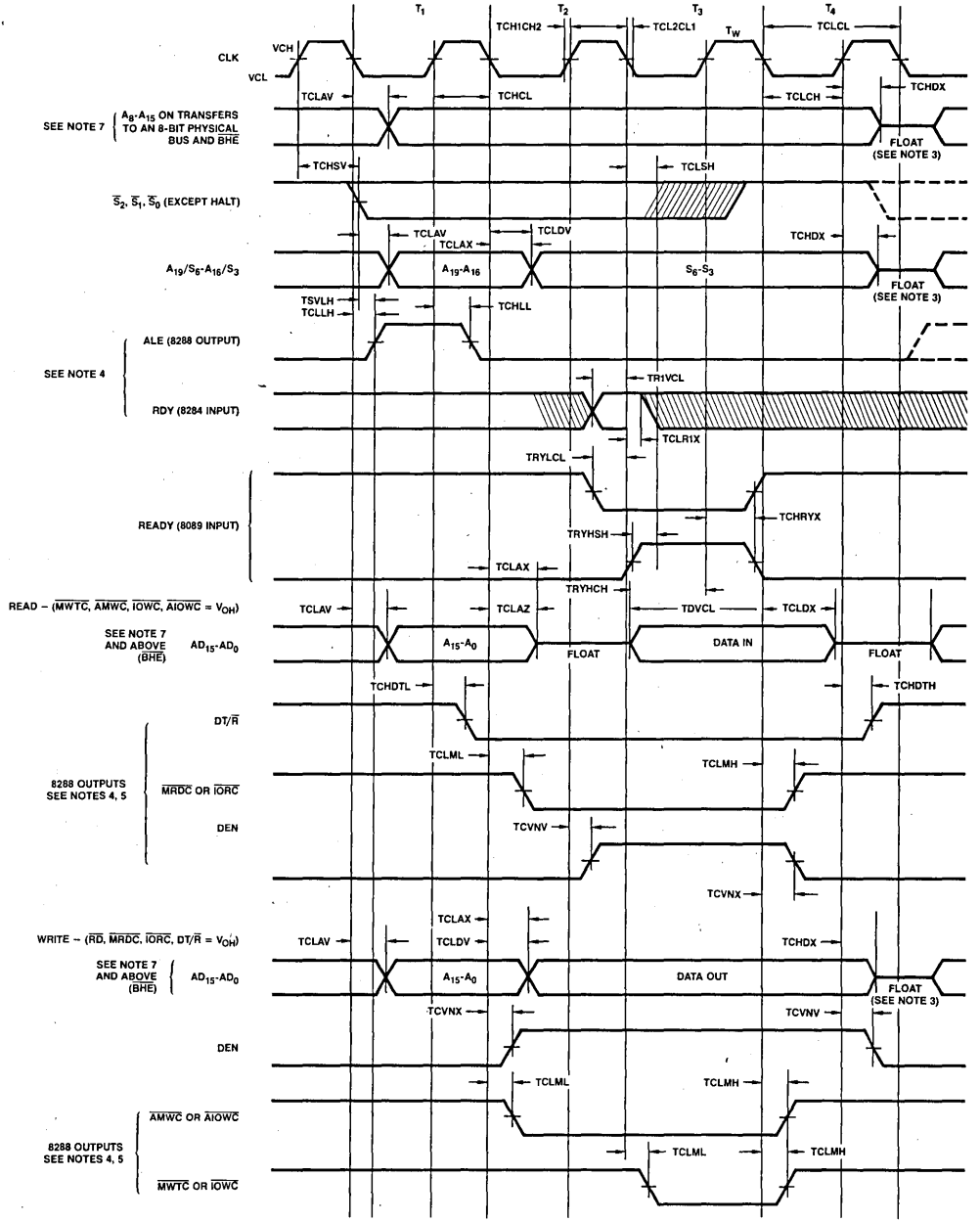
AC TESTING LOAD CIRCUIT



C_L = 100pF
C_L includes jig capacitance

WAVEFORMS

8089 BUS TIMING USING 8288

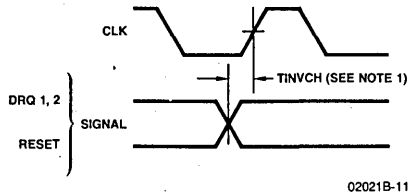


02021B-10

Notes:
1. All signals switch between V_{OH} and V_{OL} unless otherwise specified.
2. RDY is sampled near the end of T_2, T_3, T_W to determine if T_W machine states are to be inserted.
3. Following a write cycle data remains valid on the 8089 local bus until a local bus master decides to run another bus cycle. The local bus is floated by the 8089 when the 8089 enters a request bus acknowledge state.
4. Signals at 8284 or 8288 are shown for reference only.
5. The issuance of the 8288 command and control signals ($\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$, $\overline{IORC}$, $\overline{IOWC}$, $\overline{AIOWC}$, $\overline{INTA}$, and DEN) lags the active high 8288 CEN.
6. All timing measurements are made at 1.5V unless otherwise noted.
7. A_0-A_{15} are stable on transfers to an 8-bit physical data bus i.e., A_8-A_{15} don't float on a read from an 8-bit physical bus or multiplex with data on a write to an 8-bit physical bus $\overline{BHE}$ is stable (non multiplexed) for all transfers.

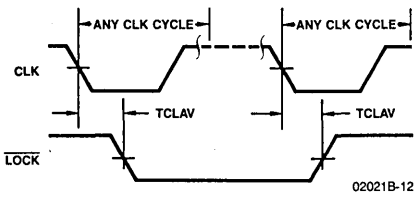
WAVEFORMS (Cont.)

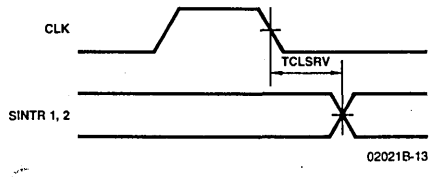
ASYNCHRONOUS SIGNAL RECOGNITION



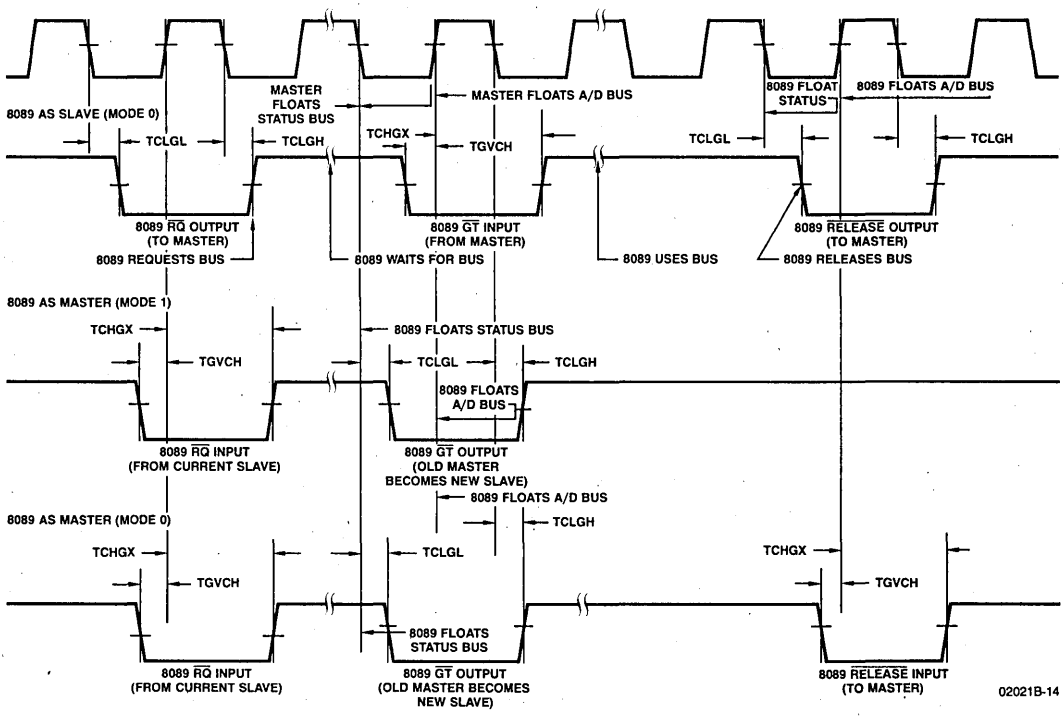
Notes: 1. Setup requirements for asynchronous signals only to guarantee recognition at next CLK.
2. All inputs except CA are latched on a CLK edge. The CA input is negative edge triggered.
3. DRQ becoming active greater than 30ns after the rising edge of CLK will guarantee non-recognition until the next rising clock edge.

BUS LOCK SIGNAL TIMING AND SINTR





REQUEST/GRANT SEQUENCE



8237A

Multimode DMA Controller

DISTINCTIVE CHARACTERISTICS

- Four independent DMA channels, each with separate registers for Mode Control, Current Address, Base Address, Current Word Count and Base Word Count.
- Transfer modes: Block, Demand, Single Word, Cascade
- Independent autoinitialization of all channels
- Memory-to-memory transfers
- Memory block initialization
- Address increment or decrement
- Master system disable
- Enable/disable control of individual DMA requests
- Directly expandable to any number of channels
- End of Process input for terminating transfers
- Software DMA requests
- Independent polarity control for DREQ and DACK signals
- Compressed timing option speeds transfers – up to 2M words/second
- +5 volt power supply
- Advanced N-channel silicon gate MOS technology
- 40 pin Hermetic DIP package
- New 8237A-5 5MHz version for higher-speed CPU compatibility

GENERAL DESCRIPTION

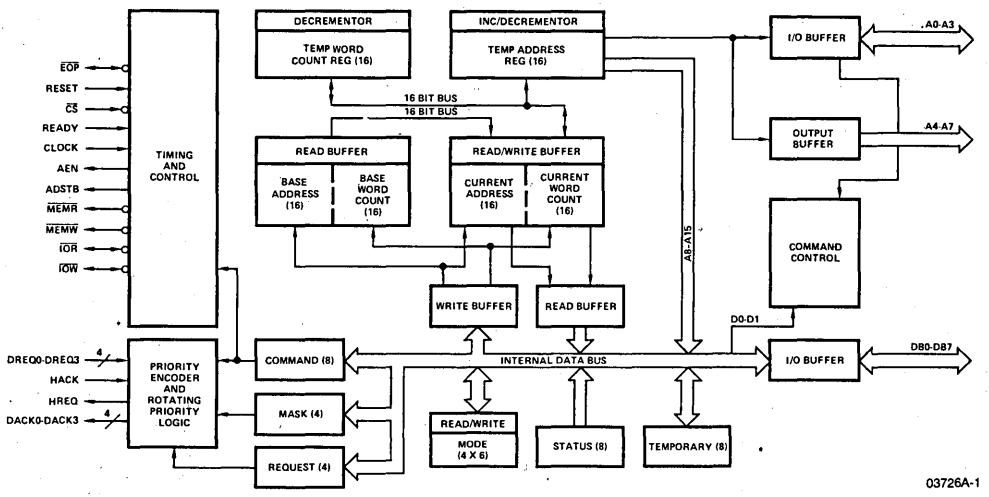
The 8237A Multimode Direct Memory Access (DMA) Controller is a peripheral interface circuit for microprocessor systems. It is designed to improve system performance by allowing external devices to directly transfer information to or from the system memory. Memory-to-memory transfer capability is also provided. The 8237A offers a wide variety of programmable control features to enhance data throughput and system optimization and to allow dynamic reconfiguration under program control.

The 8237A is designed to be used in conjunction with an external 8-bit address register such as the Am74LS373. It contains four independent channels and may be expanded to any number of channels by cascading additional controller chips.

The three basic transfer modes allow programmability of the types of DMA service by the user. Each channel can be individually programmed to Autoinitialize to its original condition following an End of Process (EOP).

Each channel has a full 64K address and word count capability. An external EOP signal can terminate a DMA or memory-to-memory transfer. This is useful for block search or compare operations using external comparators or for intelligent peripherals to abort erroneous services.

BLOCK DIAGRAM



03726A-1

ORDERING INFORMATION

Package Type	Ambient Temperature	Maximum Clock Frequency		
		3MHz	4MHz	5MHz
Hermetic DIP/ Molded DIP	$0°C \leq T_A \leq +70°C$	8237ADC/PC	8237A-4DC/PC	8237A-5DC/PC
Hermetic DIP	$-40°C \leq T_A \leq +85°C$	8237ADI		
Hermetic DIP	$-55°C \leq T_A \leq +125°C$	8237ADMB		

03726A-MMP

MAXIMUM RATINGS above which useful life may be impaired

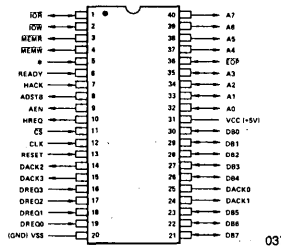
Storage Temperature	−65 to +150°C
Ambient Temperature Under Bias	−55 to +125°C
Supply Voltage	−0.5 to +7.0V
All Signal Voltages with Respect to Ground	−0.5 to +7.0V
Power Dissipation (Package Limitation)	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	T_A	V_{CC}
8237ADC/PC	0 to +70°C	5.0V ±5%
8237A-4DC/PC	0 to +70°C	5.0V ±5%
8237A-5DC/PC	0 to +70°C	5.0V ±5%
8237ADI	−40 to +85°C	5.0V ±10%
8237ADMB	−55 to +125°C	5.0V ±10%

CONNECTION DIAGRAM — Top View
D-40-1, P-40-1



03726A-4

ELECTRICAL CHARACTERISTICS over operating range (Note 1)

Parameters	Description	Test Conditions	Min	Typ	Max	Units
V_{OH}	Output HIGH Voltage	I_{OH} = −200μA	2.4			Volts
		I_{OH} = −100μA, (HREQ Only)	3.3			
V_{OL}	Output LOW Voltage	I_{OL} = 2.0mA (Data bus)			.45	Volts
		I_{OL} = 3.2mA (Other outputs)			.45	
V_{IH}	Input HIGH Voltage		2.0		V_{CC} + 0.5	Volts
V_{IL}	Input LOW Voltage		−0.5		0.8	Volts
I_{LD}	Input Load Current	$GND \leq V_{IN} \leq V_{CC}$	−10		+10	μA
I_{OLK}	Output Leakage Current	$V_{CC} \leq V_{OUT} \leq GND + .40$	−10		+10	μA
I_{CC}	Supply Current	T_A = +25°C		65	130	mA
		T_A = 0°C		75	150	
		T_A = −55°C			175	
C_{OUT}	Output Capacitance	f_c = 1.0MHz, Inputs = 0V		4	8	pF
C_{IN}	Input Capacitance			8	15	pF
C_{IO}	I/O Capacitance			10	18	pF

NOTES:

1. Typical values are for T_A = 25°C, nominal supply voltage and nominal processing parameters.
2. Input timing parameters assume transition times of 20ns or less. Waveform measurement points for both input and output signals are 2.0V for High and 0.8V for Low, unless otherwise noted.
3. Output loading is 1 Standard TTL gate plus 50pF capacitance unless noted otherwise.
4. The new $\overline{IOW}$ or $\overline{MEMW}$ pulse width for normal write will be t_{CY}-100ns and for extended write will be $2t_{CY}$-100ns. The net $\overline{IOR}$ or $\overline{MEMR}$ pulse width for normal read will be $2t_{CY}$-50ns and for compressed read will be t_{CY}-50ns.
5. t_{DQ} is specified for two different output HIGH levels. t_{DQ1} is measured at 2.0V. t_{DQ2} is measured at 3.3V. The value for t_{DQ2} assumes an external 3.3kΩ pull-up resistor connected from HREQ to V_{CC}.
6. DREQ should be held active until DACK is returned.
7. DREQ and DACK signals may be active High or active Low. Timing diagrams assume the active High mode.
8. Output loading on the data bus is 1 Standard TTL gate plus 15pF for the minimum value and 1 Standard TTL gate plus 100pF for the maximum value.
9. Successive read and/or write operations by the external processor to program or examine the controller must be timed to allow at least 600ns for the 8237A, at least 450ns for the 8237A-4, and 400ns for the 8237A-5 as recovery time between active read or write pulses.
10. Parameters are listed in alphabetical order.
11. Pin 5 is an input that should always be at a logic High level. An internal pull-up resistor will establish a logic High when the pin is left floating. Alternatively, pin 5 may be tied to V_{CC}.
12. Signals $\overline{READ}$ and $\overline{WRITE}$ refer to $\overline{IOR}$ and $\overline{MEMW}$ respectively for peripheral-to-memory DMA operations and to $\overline{MEMR}$ and $\overline{IOW}$ respectively for memory-to-peripheral DMA operations.
13. If N wait states are added during the write-to-memory half of a memory-to-memory transfer, this parameter will increase by N (t_{CY}).

SWITCHING CHARACTERISTICS

Active Cycle (Notes 2, 3, 10, 11 and 12)

Parameters	Description	8237A Min	8237A Max	8237A-4 Min	8237A-4 Max	8237A-5 Min	8237A-5 Max	Units
t_{AEL}	AEN HIGH from CLK LOW (S_1) Delay Time		300		225		200	ns
t_{AET}	AEN LOW from CLK HIGH (S_1) Delay Time		200		150		130	ns
t_{AFAB}	ADR Active to Float Delay from CLK HIGH		150		120		90	ns
t_{AFC}	$\overline{READ}$ or $\overline{WRITE}$ Float from CLK HIGH		150		120		120	ns
t_{AFDB}	DB Active to Float Delay from CLK HIGH		250		190		170	ns
t_{AHR}	ADR from $\overline{READ}$ HIGH Hold Time	t_{CY}-100		t_{CY}-100		t_{CY}-100		ns
t_{AHS}	DB from ADSTB LOW Hold Time	50		40		30		ns
t_{AHW}	ADR from WRITE HIGH Hold Time	t_{CY}-50		t_{CY}-50		t_{CY}-50		ns
t_{AK}	DACK Valid from CLK LOW Delay Time		250		220		170	ns
	$\overline{EOP}$ HIGH from CLK HIGH Delay Time		250		190		170	
	$\overline{EOP}$ LOW to CLK HIGH Delay Time		250		190		100	
t_{ASM}	ADR Stable from CLK HIGH		250		190		170	ns
t_{ASS}	DB to ADSTB LOW Setup Time	100		100		100		ns
t_{CH}	Clock High Time (Transitions ≤ 10ns)	120		100		80		ns
t_{CL}	Clock Low Time (Transitions ≤ 10ns)	150		110		68		ns
t_{CY}	CLK Cycle Time	320		250		200		ns
t_{DCL}	CLK HIGH to $\overline{READ}$ or $\overline{WRITE}$ LOW Delay (Note 4)		270		200		190	ns
t_{DCTR}	$\overline{Read}$ HIGH from CLK HIGH (S_4) Delay Time (Note 4)		270		210		190	ns
t_{DCTW}	$\overline{WRITE}$ HIGH from CLK HIGH (S_4) Delay Time (Note 4)		200		150		130	ns
t_{DQ1}	HREQ Valid from CLK HIGH Delay Time (Note 5)		160		120		120	ns
t_{DQ2}			250		190		120	
t_{EPS}	$\overline{EOP}$ LOW from CLK LOW Setup Time	60		45		40		ns
t_{EPW}	$\overline{EOP}$ Pulse Width	300		225		220		ns
t_{FAAB}	ADR Float to Active Delay from CLK HIGH		250		190		170	ns
t_{FAC}	$\overline{READ}$ or $\overline{WRITE}$ Active from CLK HIGH		200		150		150	ns
t_{FADB}	DB Float to Active Delay from CLK HIGH		300		225		200	ns
t_{HS}	HACK Valid to CLK HIGH Setup Time	100		75		75		ns
t_{IDH}	Input Data from $\overline{MEMR}$ HIGH Hold Time	0		0		0		ns
t_{IDS}	Input Data to $\overline{MEMR}$ HIGH Setup Time	250		190		170		ns
t_{ODH}	Output Data from $\overline{MEMW}$ HIGH Hold Time	20		20		10		ns
t_{ODV}	Output Data Valid to $\overline{MEMW}$ HIGH (Note 13)	200		125		130		ns
t_{QS}	DREQ to CLK LOW (S_1, S_4) Setup Time	120		90		0		ns
t_{RH}	CLK to READY LOW Hold Time	20		20		20		ns
t_{RS}	READY to CLK LOW Setup Time	100		60		75		ns
t_{STL}	ADSTB HIGH from CLK HIGH Delay Time		200		150		130	ns
t_{STT}	ADSTB LOW from CLK HIGH Delay Time		140		110		90	ns

SWITCHING CHARACTERISTICS

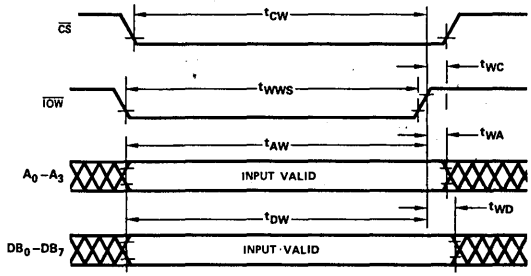
Program Condition (Idle Cycle)

(Notes 2, 3, 10 and 11)

Parameters	Description	8237A Min	8237A Max	8237A-4 Min	8237A-4 Max	8237A-5 Min	8237A-5 Max	Units
t_{AR}	ADR Valid or $\overline{CS}$ LOW to $\overline{READ}$ LOW	50		50		50		ns
t_{AW}	ADR Valid to $\overline{WRITE}$ HIGH Setup Time	200		150		160		ns
t_{CW}	CS LOW to $\overline{WRITE}$ HIGH Setup Time	200		150		160		ns
t_{DW}	Data Valid to $\overline{WRITE}$ HIGH Setup Time	200		150		160		ns
t_{RA}	ADR or $\overline{CS}$ Hold from $\overline{READ}$ HIGH	0		0		0		ns
t_{RDE}	Data Access from $\overline{READ}$ LOW (Note 8)		200		200		140	ns
t_{DRF}	DB Float Delay from $\overline{READ}$ HIGH	20	100	20	100	0	70	ns
t_{RSTD}	Power Supply HIGH to RESET LOW Setup Time	500		500		500		μs
t_{RSTS}	RESET to First $\overline{IOWR}$	2		2		2		t_{CY}
t_{RSTW}	RESET Pulse Width	300		300		300		ns
t_{RW}	$\overline{READ}$ Width	300		250		200		ns
t_{WA}	ADR from $\overline{WRITE}$ HIGH Hold Time	20		20		20		ns
t_{WC}	$\overline{CS}$ HIGH from $\overline{WRITE}$ HIGH Hold Time	20		20		20		ns
t_{WD}	Data from $\overline{WRITE}$ HIGH Hold Time	30		30		30		ns
t_{WWS}	Write Width	200		200		160		ns

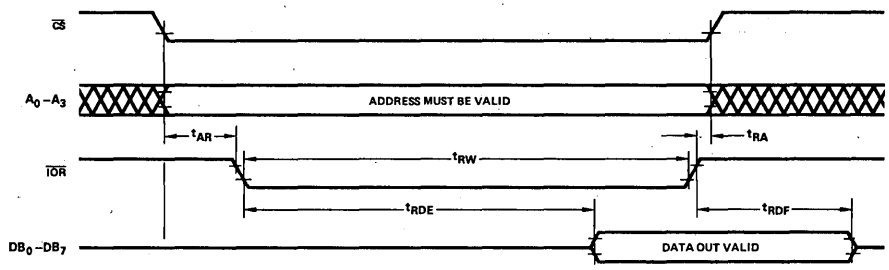
SWITCHING WAVEFORMS

Timing Diagram 1. Program Condition Write Timing (Note 9)



03726A-2

Timing Diagram 2. Program Condition Read Cycle (Note 9)



03726A-3

8251 • 9551

Programmable Communications Interface
Advanced MOS/LSI

DISTINCTIVE CHARACTERISTICS

- Separate control and transmit register input buffers
- 8080A/9080A compatible
- Synchronous or asynchronous serial data transfer
- Parity, overrun and framing errors detected
- Half or full duplex signalling
- Character length of 5, 6, 7 or 8 bits
- Internal or external synchronization
- Odd parity, even parity or no parity bit
- Modem interface controlled by processor
- Programmable Sync pattern
- Fully TTL compatible logic levels
- Commercial and military temperature range operation
- Ion-implanted N-channel silicon gate MOS technology

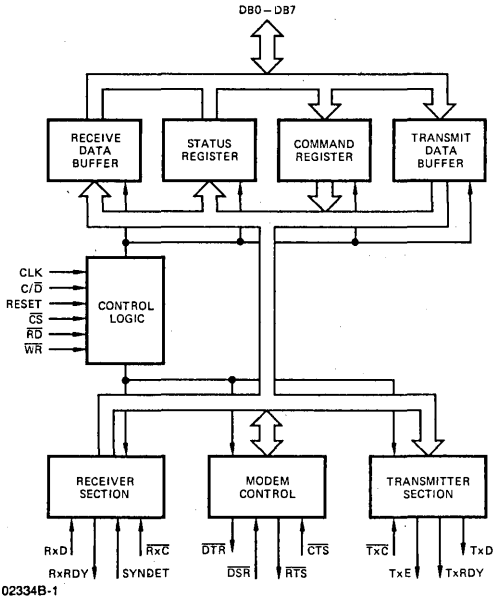
GENERAL DESCRIPTION

The 8251/9551 is a programmable serial data communication interface that provides an Universal Synchronous/Asynchronous Receiver/Transmitter (USART) function. It is normally used as a peripheral device for an associated processor, and may be programmed by the processor to operate in a variety of standard serial communication formats.

The device accepts parallel data from the CPU, formats and serializes the information based on its current operating mode, and then transmits the data as a serial bit stream. Simultaneously, serial data can be received, converted into parallel form, de-formated, and then presented to the CPU. The USART can operate in an independent full duplex mode.

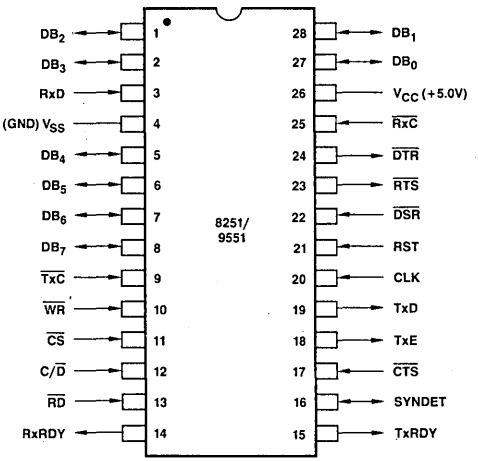
Data, Control, operation and format options are all selected by commands from an associated processor. This provides an unusual degree of flexibility and allows the 8251/9551 to service a wide range of communication disciplines and applications.

BLOCK DIAGRAM



02334B-1

CONNECTION DIAGRAM
Top View
D-28, P-28



02334B-2

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Package Type	Ambient Temperature Specification	8251 2.4MHz	9551 Standard 2.4MHz	9551 Upgraded 2.6MHz
Molded DIP	$0°C \leq T_A \leq 70°C$	P8251	Am9551PC	Am9551-4PC
Hermetic DIP		D8251	Am9551DC	Am9551-4DC
Hermetic DIP	$-40°C \leq T_A \leq 85°C$	ID8251	Am9551DI	Am9551-4DI
Hemertic DIP	$-55°C \leq T_A \leq 125°C$	MD8251B	Am9551DMB	

02334B-MMP

INTERFACE SIGNAL DESCRIPTION

Data Bus

The 9551 uses an 8 bit bi-directional data bus to exchange information with an associated processor. Internally, data is routed between the data bus buffers and the transmitter section or receiver section as selected by the Read ($\overline{RD}$) or Write ($\overline{WR}$) control inputs.

Chip Select ($\overline{CS}$)

The active low Chip Select input allows the 9551 to be individually selected from other devices within its address range. When Chip Select is high, reading or writing is inhibited, and the data bus output is in it's high impedance state.

Reset

The 9551 will assume an idle state when a high level is applied to the Reset input. When the Reset is returned Low, the 9551 will remain in the idle state until it receives a new mode control instruction.

Read ($\overline{RD}$)

The active low Read input enables data to be transferred from the 9551 to the processor.

Write ($\overline{WR}$)

The active low Write input enables data to be transferred from the processor to the 9551.

Control/Data (C/$\overline{D}$)

During a Read operation, if this input is at a high level the status byte will be read, and if it is at a low level the receive data will be read by the processor. When a Write operation is being performed, this input will indicate to the 9551 that the bus information being written is a command if C/$\overline{D}$ is high and data if C/$\overline{D}$ is low.

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	0	1	0	9551 DATA → DATA BUS
0	1	0	0	DATA BUS → 9551 DATA
1	0	1	0	9551 STATUS → DATA BUS
1	1	0	0	DATA BUS → 9551 COMMAND
X	X	X	1	DATA BUS → THREE-STATE

Clock (CLK)

This input is used for internal timing within the 9551. It does not control the transmit or receive rate. However, it should be at least 30 times the receive or transmit rate in the synchronous mode and 4.5 times the receive or transmit rate in the asynchronous mode. The CLK frequency is also restricted by both an upper and a lower bound. This input is often connected to a clock from the associated processor.

Receiver Data (RxD)

Serial data is received from the communication line on this input.

Receiver Clock ($\overline{RxC}$)

The serial data on input RxD is clocked into the 9551 by the $\overline{RxC}$ clock signal. In the synchronous mode, $\overline{RxC}$ is determined by the baud rate and supplied by the modem. In the asynchronous mode, $\overline{RxC}$ is 1, 16, or 64 times the baud rate as selected in the mode control instruction. Data is sampled by the 9551 on the rising edge of $\overline{RxC}$.

Receiver Ready (RxRDY)

The RxRDY output signal indicates to the processor that data has been shifted into the receiver buffer from the receiver section and may be read. The signal is active high and will be reset when the buffer is read by the processor. RxRDY can be activated only if the receiver enable (RxE) has been set in the command register, even though the receiver may be running. If the processor does not read the receiver buffer before the next character is shifted from the receiver section then an overrun error will be indicated in the status buffer.

Sync Detect (SYNDET)

This signal is used only in the synchronous mode. It can be either an output or input depending on whether the program is set for internal or external synchronization. As an output, a high level indicates when the sync character has been detected in the received data stream after the Internal Synchronization mode has been programmed. If the 9551 is programmed to utilize two sync characters, then SYNDET will go to a high level when the last bit of the second sync character is received. SYNDET is reset when the status buffer is read or when a Reset signal is activated. SYNDET will perform as an input when the External Synchronization mode is programmed. External logic can supply a positive-going signal to indicate to the 9551 that synchronization has been attained. This will cause it to initialize the assembly of characters on the next falling edge of $\overline{RxC}$. To successfully achieve synchronization the SYNDET signal should be maintained in a high condition for at least one full period of $\overline{RxC}$.

Transmit Data (TxD)

Serial data is transmitted to the communication line on this output.

Transmitter Clock ($\overline{TxC}$)

The serial data on TxD is clocked out with the $\overline{TxC}$ signal. The relationship between clock rate and baud rate is similar to that for RxC. Data is shifted out of the 9551 on the falling edge of $\overline{TxC}$.

Transmitter Ready (TxRDY)

The TxRDY output signal goes high when data in the Transmit Data Buffer has been shifted into the transmitter section allowing the Transmit Data Buffer to accept the next byte from the processor. TxRDY will be reset when information is written into the Transmit Data Buffer. Loading command register also resets TxRDY. TxRDY will be available on this output pin only when the 9551 is enable to transmit (CTS=0, TxEN=1). However, the TxRDY bit in the status Buffer will always be set when the Transmit Data Buffer is empty regardless of the state of TxEN and CTS.

Transmitter Empty (TxE)

The TxE output signal goes high when the Transmitter section has transmitted its data and is empty. The signal will remain high until a new data byte is shifted from the Transmit Data Buffer to the Transmitter section. In the synchronous mode if the processor does not load a new byte into the buffer in time, TxE will, independent of the status of the TxEN bit in the command register, momentarily go to a high level as SYNC characters are loaded into the Transmitter Section.

Data Terminal Ready ($\overline{DTR}$)

This signal is a general purpose output which reflects the state of bit 1 in the Command instruction. It is commonly connected to an associated modem to indicate that the 9551 is ready.

Data Set Ready ($\overline{DSR}$)

This is a general purpose input signal and forms part of the status byte that may be read by the processor. DSR is generally used as a response to DTR, by the Modem, to indicate that it is ready. The signal acts only as a flag and does not control any internal logic.

Request to Send ($\overline{RTS}$)

This is a general purpose output, similar to $\overline{DTR}$, and reflects the state of bit 5 in Command Instruction. It is normally used to initiate a data transmission by requesting the modem to prepare to send.

INTERFACE SIGNAL DESCRIPTION (Cont.)

Clear to Send ($\overline{CTS}$)
This is a general purpose input signal used to enable the 8251/9551 to transmit data if the TxEN bit in the Command byte is a one. $\overline{CTS}$ is generally used as a response to RTS by a modem to indicate that transmission may begin. Designers not using $\overline{CTS}$ in their systems should remember to tie it low so that 8251/9551 data transmission will not be disabled

OPERATION AND PROGRAMMING

The microcomputer program controlling the 9551 performs these tasks:

- Outputs control codes
- Inputs status
- Outputs data to be transmitted
- Inputs data which have been received

Control codes determine the mode in which the 9551 will operate and are used to set or reset control signals output by the 9551.

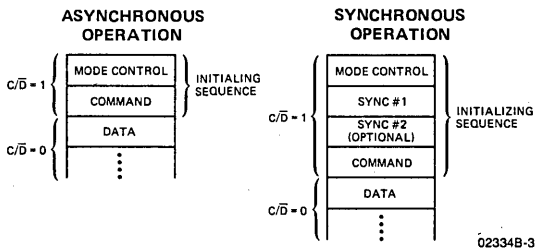
The Status register contents will be read by the program monitoring this device's operation in order to determine error conditions, when and how to read data, write data or output control codes. Program logic may be based on reading status bit levels, or control signals may be used to request interrupts.

INITIALIZING THE 9551

The 9551 may be initialized following a system reset or prior to starting a new serial I/O sequence. The USART must be reset following power up and subsequently may be reset at any time following completion of one activity and preceding a new set of operations. Following a reset, the 9551 enters an idle state in which it can neither transmit nor receive data.

The 9551 is initialized with two, three or four control words from the processor. Figure 1 shows the sequence of control words needed to initialize the 9551, for synchronous or for asynchronous operation. Note that in asynchronous operation a mode control is output to the device followed by a command. For synchronous operation, the mode control is followed by one or two SYNC characters, and then a command.

Figure 1. Control Word Sequence for Initialization.



Only a single address is set aside for mode control bytes, command bytes and SYNC character bytes. For this to be possible, logic internal to the chip directs control information to its proper destination based on the sequence in which it is received. Following a reset, the first control code output is interpreted as a mode control. If the mode control specifies synchronous operation, then the next one or two bytes (as determined by the mode byte) output as control codes will be interpreted as SYNC characters. For either asynchronous or synchronous operation, the next byte output as a control code is interpreted as a command. All subsequent bytes output as control codes are interpreted as commands. There are two ways in which control logic may return to anticipating a mode control input; following an external Reset signal or following an internal Reset command.

MODE CONTROL CODES

The 9551 interprets mode control codes as illustrated in Figures 2 and 3.

Control code bits 0 and 1 determine whether synchronous or asynchronous operation is specified. A non-zero value in bits 0 and 1 specifies asynchronous operation and defines the relationship between data transfer baud rate and receiver or transmitter clock rate. Asynchronous serial data may be received or transmitted on every clock pulse, on every 16th clock pulse, or on every 64th clock pulse. A zero in both bits 0 and 1 defines the mode of operation as synchronous.

For synchronous and asynchronous modes, control bits 2 and 3 determine the number of data bits which will be present in each data character.

For synchronous and asynchronous modes, bits 4 and 5 determine whether there will be a parity bit in each character, and if so, whether odd or even parity will be adopted. Thus in synchronous mode a character will consist of five, six, seven or eight data bits, plus an optional parity bit. In asynchronous mode, the data unit will consist of five, six, seven or eight data bits, an optional parity bit, a preceding start bit, plus 1, 1½, or 2 trailing stop bits. Interpretation of subsequent bits differs for synchronous or asynchronous modes.

Figure 2. Synchronous Mode Control Code.

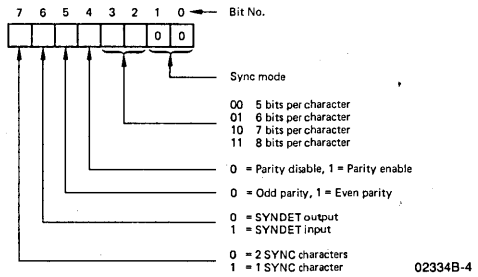
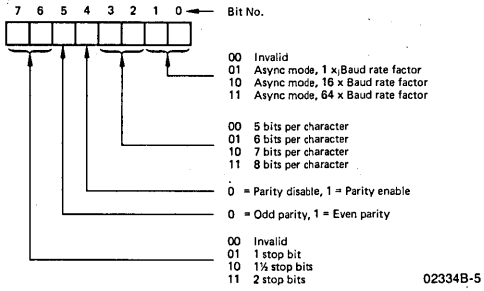


Figure 3. Asynchronous Mode Control Code.



OPERATION AND PROGRAMMING (Cont.)

Control code bits 6 and 7 in asynchronous mode determine how many stop bits will trail each data unit. 1½ stop bits can only be specified with a 16x or 64x baud rate factor. In these two cases, the half stop bit will be equivalent to 8 or 32 clock pulses, respectively.

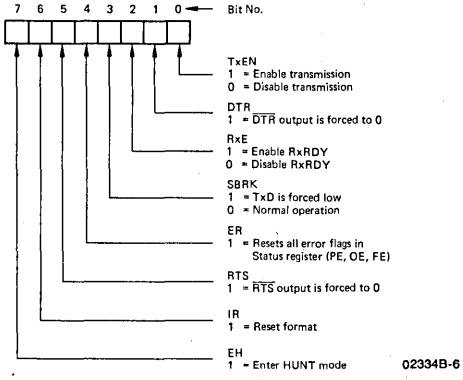
In synchronous mode, control bits 6 and 7 determine how character synchronization will be achieved. When SYNDET is an output, internal synchronization is specified; one or two SYNC characters, as specified by control bit 7, must be detected at the head of a data stream in order to establish synchronization.

COMMAND WORDS

Command words are used to initiate specific functions within the 9551 such as, "reset all error flags" or "start searching for sync". Consequently, Command Words may be issued by the microprocessor to the 9551 at any time during the execution of a program in which specific functions are to be initiated within the communication circuit.

Figure 4 shows the format for the Command Word.

Figure 4. 9551 Control Command.



Bit 0 of the Command Word is the Transmit Enable bit (TxEN). Data transmission from the 9551 cannot take place unless TxEN is set in the command register. Figure 5 defines the way in which TxEN, TxE and TxRDY combine to control transmitter operations.

Bit 1 is the Data Terminal Ready (DTR) bit. When the DTR command bit is set, the $\overline{DTR}$ output connection is active (low). DTR is used to advise a modem that the data terminal is prepared to accept or transmit data.

Bit 2 is the Receiver Enable Command bit (RxE). RxE is used to enable the RxRDY output signal. RxE prevents the RxRDY signal from being generated to notify the prcessor that a complete character is framed in the Receive Character Buffer. It does not inhibit the assembly of data characters at the input, however. Consequently, if communication circuits are active, characters will be assembled by the receiver and transferred to the Receiver Character Buffer. If RxE is disabled, the overrun error (OE) will probably be set; to insure proper operation, the overrun error is usually reset with the same command that enables RxE.

Bit 3 is the Send Break Command bit (SBRK). When SBRK is set, the transmitter output (TxD) is interrupted and a continuous binary "0" level, (spacing) is applied to the TxD output signal. The break will continue until a subsequent Command Word is sent to the 9551 to remove SBRK.

Bit 4 is the Error Reset bit (ER). When a Command Word is transmitted with the ER bit set, all three error flags in the Status Register are reset. Error Reset occurs when the Command Word is loaded into the 9551. No latch is provided in the Command Register to save the ER command bit.

Bit 5, the Request To Send Command bit (RTS), sets a latch to reflect the RTS signal level. The output of this latch is created independently of other signals in the 9551. As a result, data transfers may be made by the microprocessor to the Transmit Register, and data may be actively transmitted to the communication line through TxD regardless of the status of RTS.

Bit 6, the Internal Reset (IR), causes the 9551 to return to the Idle mode. All functions within the 9551 cease and no new operation can be resumed until the circuit is reinitialized. If the operating mode is to be altered during the execution of a microprocessor program, the 9551 must first be reset. Either the external reset connection can be activated, or the Internal Reset Command can be sent to the 9551. Internal Reset is a momentary function performed only when the command is issued.

Bit 7 is the Enter Hunt command bit (EH). The Enter Hunt mode command is only effective for the 9551 when it is operating in the Synchronous mode. EH causes the receiver to stop assembling characters at the RxD input and start searching for the prescribed sync pattern. Once the "Enter Hunt" mode has been initiated, the search for the sync pattern will continue indefinitely until EH is reset when a subsequent Command Word is sent, when the IR command is sent to the 9551, or when SYNC characters are recognized.

Figure 5. Operation of the Transmitter Section as a Function of TxE, TxRDY and TxEN.

TxEN	Txe	TxRDY	
1	1	1	Transmit Output Register and Transmit Character Buffer empty. TxD continues to mark if 9551 is in the asynchronous mode. TxD will send Sync pattern if 9551 is in the Synchronous Mode. Data can be entered into Buffer.
1	0	1	Transmit Output Register is shifting a character. Transmit Character Buffer is available to receive a new byte from the processor.
1	1	0	Transmit Register has finished sending. A new character is waiting for transmission. This is a transient condition.
1	0	0	Transmit Register is currently sending and an additional character is stored in the Transmit Character Buffer for transmission.
0	0/1	0/1	Transmitter is disabled.

STATUS REGISTER

The Status Register maintains information about the current operational status of the 9551. Figure 6 shows the format of the Status Register.

TxRDY signals the processor that the Transmit Character Buffer is empty and that the 9551 can accept a new character for transmission.

OPERATION AND PROGRAMMING (Cont.)

RxRDY signals the processor that a completed character is holding in the Receive Character Buffer Register for transfer to the processor.

TxE signals the processor that the Transmit Register is empty.

PE is the Parity Error signal indicating to the CPU that the character stored in the Receive Character Buffer was received with an incorrect number of binary "1" bits.

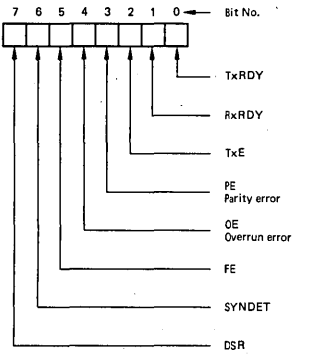
OE is the receiver Overrun Error. OE is set whenever a byte stored in the Receiver Character Register is overwritten with a new byte before being transferred to the processor.

FE is the character framing error which indicates that the asynchronous mode byte stored in the Receiver Character Buffer was received with incorrect character bit format, as specified by the current mode.

SYNDET is the synchronous mode status bit associated with internal sync detection.

DSR is the status bit set by the external Data Set Ready signal to indicate that the communication Data Set is operational. All status bits are set by the functions described for them. SYNDET is reset whenever the processor reads the Status Register. OE, FE, PE are reset only by command.

Figure 6. The 9551 Status Register.



MAXIMUM RATINGS Above which the useful life may be impaired

Storage Temperature	−65°C to +150°C
Ambient Temperature Under Bias	−55°C to +125°C
V_{CC} with Respect to V_{SS}	−0.5V to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5V to +7.0V
Power Dissipation	1.0W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	Ambient Temperature	V_{CC}	V_{SS}
D8251, P8251 Am9551DC, Am9551PC Am9551-4DC, Am9551-4PC	0 to 70°C	5.0V ±5%	0V
ID8251 Am9551DI Am9551-4DI	−40 to 85°C	5.0V ±10%	0V
MD8251B Am9551DMB	−55 to 125°C	5.0V ±10%	0V

ELECTRICAL CHARACTERISTICS Over Operating Range (Note 1)

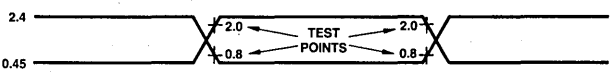
Parameters	Description	Test Conditions	8251 Min.	8251 Typ.	8251 Max.	9551 Min.	9551 Typ.	9551 Max.	Units
V_{OH}	Output HIGH Voltage	I_{OH} = −200μA				2.4			Volts
		I_{OH} = −100μA	2.4						
V_{OL}	Output LOW Voltage	I_{OL} = 3.2mA (Note 6)						0.45	Volts
		I_{OL} = 1.6mA			0.45				
V_{IH}	Input HIGH Voltage		2.2		V_{CC}	2.2		V_{CC}	Volts
V_{IL}	Input LOW Voltage		−0.5		0.8	−0.5		0.8	Volts
I_{LI}	Input Load Current	$V_{SS} \leq V_{IN} \leq V_{CC}$			10			10	μA
I_{DL}	Data Bus Leakage	V_{OUT} = 0.45V			−50			−50	μA
		$V_{OUT} = V_{CC}$			10			10	
I_{CC}	V_{CC} Supply Current	T_A = +25°C		45			45		mA
		T_A = 0°C			80			80	
		T_A = −40°C, −55°C			120			120	
C_O	Output Capacitance	fc = 1.0MHz, Inputs = 0V						15	pF
C_I	Input Capacitance				10			10	pF
$C_{I/O}$	I/O Capacitance				20			20	pF

SWITCHING CHARACTERISTICS OVER OPERATING RANGE (Note 2)

Parameters	Description		8251 Min.	8251 Max.	9551 Min.	9551 Max.	9551-4 Min.	9551-4 Max.	Units
t_{AR}	$\overline{CS}$, C/$\overline{D}$ Stable to $\overline{READ}$ Low Set-up Time		50		50		50		ns
t_{AW}	$\overline{CS}$, C/$\overline{D}$ Stable to $\overline{WRITE}$ Low Set-up Time		20		20		20		ns
t_{CR}	$\overline{DSR}$, $\overline{CTS}$ to $\overline{READ}$ Low Set-up Time			16		16		16	tCY
t_{CY}	Clock Period		.420	1.35	.380	1.35	.380	1.35	μs
t_{DF}	$\overline{READ}$ High to Data Bus Off Delay		25	200	25	200	25	200	ns
t_{DTx}	TxC Low to TxD Delay			1.0		1.0		1.0	μs
t_{DW}	Data to $\overline{WRITE}$ High Set-up Time		200		150		100		ns
t_{ES}	External SYNDET to RxC Low Set-up Time			16		16		16	tCY
t_{HRx}	Sampling Pulse to Rx Data Hold Time		2.0		2.0		2.0		μs
t_{IS}	Data Bit (Center) to Internal SYNDET Delay			25		25		25	tCY
$t_{\phi W}$	Clock Pulse Width		220	0.7tCY	175	0.7tCY	175	0.7tCY	ns
t_R, t_F	Clock Rise & Fall Time		0	50	0	50	0	50	ns
t_{RA}	$\overline{READ}$ High to $\overline{CS}$, C/D Hold Time		5.0		5.0		5.0		ns
t_{RD}	$\overline{READ}$ Low to Data Bus On Delay			350		250		180	ns
t_{RPD}	Receiver Clock High Time	1x Baud Rate	15		15		15		tCY
		16x & 64x Baud Rate	3.0		3.0		3.0		
t_{RPW}	Receiver Clock Low Time	1x Baud Rate	12		12		12		tCY
		16x & 64x Baud Rate	1.0		1.0		1.0		
t_{RR}	$\overline{READ}$ Pulse Width		430		380		250		ns
t_{RV}	Time Between $\overline{WRITE}$ Pulses During Initialization (Note 3)		6.0		6.0		6.0		tCY
t_{Rx}	Data Bit (Center) to RxRDY Delay			20		20		20	tCY
t_{SRx}	Rx Data to Sampling Pulse Set-up Time		2.0		2.0		2.0		μs
t_{TPD}	Transmitter Clock High Time	1x Baud Rate	15		15		15		tCY
		16x & 64x Baud Rate	3.0		3.0		3.0		
t_{TPW}	Transmitter Clock Low Time	1x Baud Rate	12		12		12		tCY
		16x & 64x Baud Rate	1.0		1.0		1.0		
t_{Tx}	Data Bit (Center) to TxRDY Delay			16		16		16	tCY
t_{TxE}	Data Bit (Center) to Tx EMPTY Delay			16		16		16	tCY
t_{WA}	$\overline{WRITE}$ High to $\overline{CS}$, C/D Hold Time		20		20		20		ns
t_{WC}	$\overline{WRITE}$ High to TxE, $\overline{DTR}$, $\overline{RTS}$ Delay			16		16		16	tCY
t_{WD}	$\overline{WRITE}$ High to Data Hold Time		40		40		40		ns
t_{WW}	$\overline{WRITE}$ Pulse Width		400		380		250		ns
f_{Rx}	Receiver Clock Frequency	1x Baud Rate	DC	56	DC	56	DC	56	kHz
		16x & 64x Baud Rate	DC	520	DC	520	DC	520	
f_{Tx}	Transmitter Clock Frequency	1x Baud Rate	DC	56	DC	56	DC	56	kHz
		16x & 64x Baud Rate	DC	520	DC	520	DC	520	

Notes:
1. Typical values are for T_A = 25°C, nominal supply voltage and nominal processing parameters.
2. Test conditions include: transition times ≤ 20ns, output loading of 1 TTL gate plus 100pF, input and output timing reference levels of 0.8V and 2.0V.
3. This time period between write pulses is specified for initialization purposes only; when MODE, SYNC 1, SYNC 2, COMMAND and first DATA BYTE are written into the 9551. Subsequent writing of both COMMAND and DATA are only allowed when TxRDY = 1.
4. Reset Pulse Width = 6tCY min.
5. Switching Characteristic parameters are listed in alphabetical order.
6. The maximum Input Low Current (I_{OL}) is 1.6mA at V_{OL} = .45V max over the military temperature range (−55 to +125°C) and V_{CC} = 5V ±10%.

AC TESTING INPUT, OUTPUT WAVEFORM

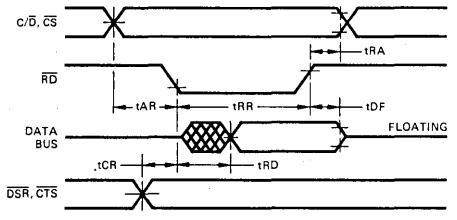


02334B-8

AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." Timing measurements are made at 2.0V for a logic "1" and 0.8V for a logic "0."

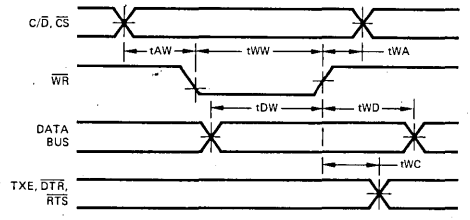
SWITCHING WAVEFORMS

READ OPERATION



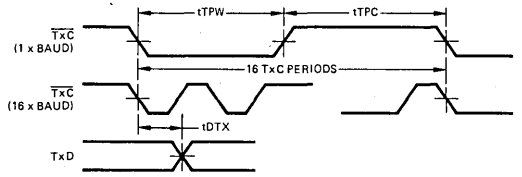
02334B-9

WRITE OPERATION



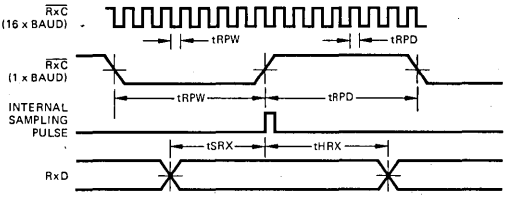
02334B-10

TRANSMITTER CLOCK AND DATA



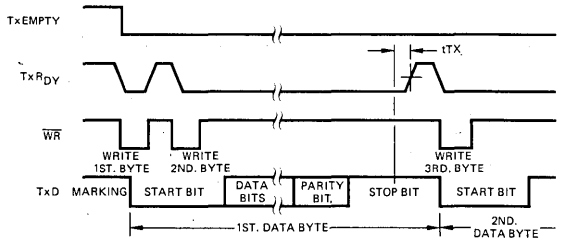
02334B-11

RECEIVER CLOCK AND DATA



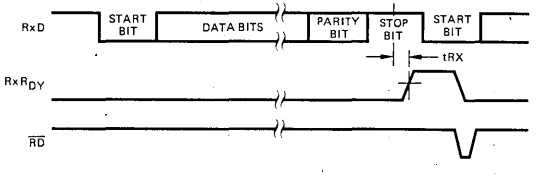
02334B-12

TxRDY TIMING (ASYNC MODE)



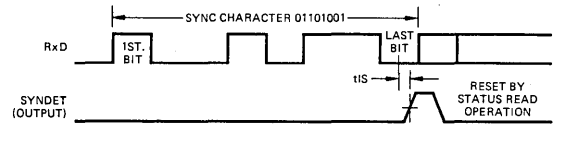
02334B-13

RxRDY TIMING (ASYNC MODE)



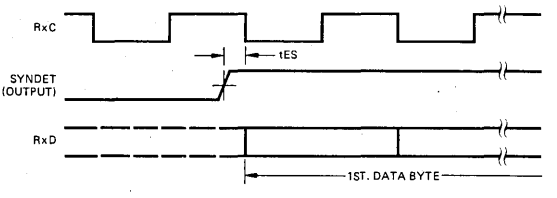
02334B-14

INTERNAL SYNC DETECT (SYNC MODE ONLY)



02334B-15

EXTERNAL SYNC DETECT (SYNC MODE ONLY)



02334B-16

8251A

Programmable Communication Interface

DISTINCTIVE CHARACTERISTICS

- Synchronous operation up to 64K baud
- Asynchronous operation up to 19.2K baud
- Full duplex, double-buffered transmitter and receiver
- Fully programmable with several speed and character modes
- Error detection for parity, overrun, and framing
- Compatible with 8080/85/86/88 microprocessors
- Single +5V supply and TTL clock
- False start bit detection; automatic break detect and handling.
- Supports bi-sync

GENERAL DESCRIPTION

The AMD 8251A is the enhanced version of the industry standard 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART) designed for data communications with 8-bit, 16-bit, and single-chip microprocessors. The 8251A is used as a peripheral device and is programmed by the CPU to operate using serial data transmission techniques. The 8251A interfaces easily with a modem.

The USART accepts data characters from the CPU in parallel format and then converts them into a continuous serial data stream for transmission. Simultaneously, it can receive serial data streams and convert them into parallel data characters for the CPU. The 8251A communicates with the CPU via direct control lines and 8-bit control words on the system bus. The CPU can query the USART status at any time.

The 8251A is fabricated with a N-channel silicon gate process and is packaged in a plastic or ceramic 28-pin DIP.

Figure 1. Block Diagram

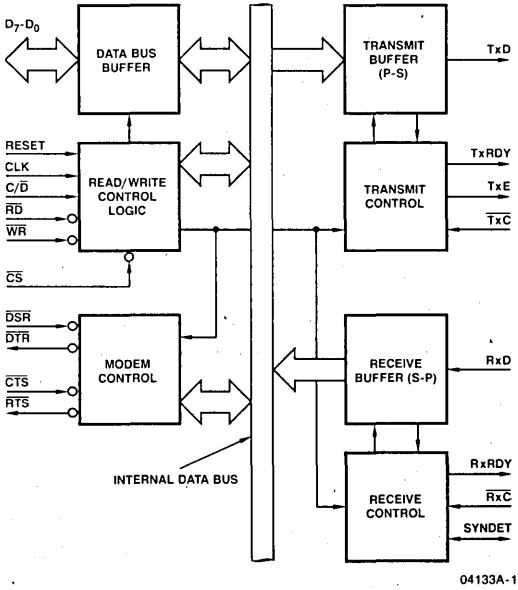
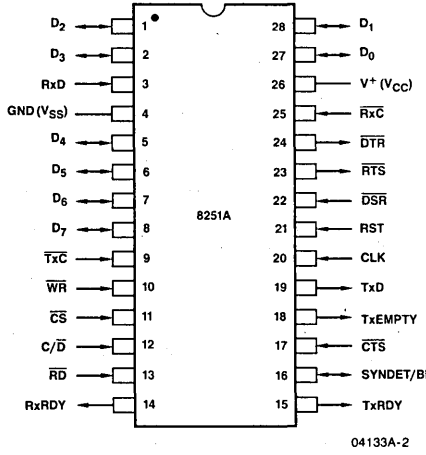


Figure 2. Pin Configuration

D-28
P-28



Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Package Type	Ambient Temperature Specification	Device Number 3MHz
Molded DIP	$0°C \leq T_A \leq 70°C$	P8251A
Hermetic DIP		D8251A
Hermetic DIP	$-40°C \leq T_A \leq +85°C$	ID8251A
Hermetic DIP	$-55°C \leq T_A \leq +125°C$	MD8251AB

FEATURES AND ENHANCEMENTS

The 8251A is an advanced design of the industry standard USART 8251. The 8251A operates with a wide range of microprocessors and microcomputers.

The 8251A incorporates all the key features of the 8251/9551 and has the following additional features and enhancements:

- 8251A has double-buffered data paths with separate I/O registers for control, status, Data In, and Data Out, which considerably simplifies control programming and minimizes CPU overhead.
- In asynchronous operations, the Receiver detects and handles "break" automatically, relieving the CPU of this task.
- A refined Rx initialization prevents the Receiver from starting when in "break" state, preventing unwanted interrupts from a disconnected USART.
- At the conclusion of a transmission, TxD line will always return to the marking state unless SBRK is programmed.
- Tx Enable logic enhancement prevents a Tx Disable command from halting transmission until all data previously written has been transmitted. The logic also prevents the transmitter from turning off in the middle of a word.
- When External Sync Detect is programmed, Internal Sync Detect is disabled, and an External Sync Detect status is provided via a flip-flop which clears itself upon a status read.
- Possibility of a false sync detect is minimized by ensuring that if double character sync is programmed, the characters be contiguously detected and also by clearing the Rx register to all ones whenever Enter Hunt command is issued in Sync mode.
- As long as the 8251A is not selected, the $\overline{RD}$ and $\overline{WR}$ do not affect the internal operation of the device.
- The 8251A Status can be read at any time but the status update will be inhibited during status read.
- The 8251A is free from extraneous glitches and has enhanced AC and DC characteristics, providing higher speed and better operating margins.
- Synchronous baud rate from DC to 64K.

MAXIMUM RATINGS

Ambient Temperature Under Bias	−55 to 125°C
Storage Temperature	−65 to +150°C
Voltage to Any Pin with Respect to Ground	−0.5 to +7.0V
Power Dissipation	1.0W

Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

OPERATING RANGE

Part Number	T_A	V_{CC}	V_{SS}
P8251A D8251A	$0°C \leq T_A \leq 70°C$	5.0V ±5%	0V
ID8251A	$-40°C \leq T_A \leq 85°C$	5.0V ±10%	0V
MD8251AB	$-55°C \leq T_A \leq 125°C$	5.0V ±10%	0V

DC CHARACTERISTICS

Com'l T_A = 0 to 70°C, V_{CC} = 5V ±5%
Ind T_A = −40 to 85°C, V_{CC} = 5V ±10%
Mil T_A = −55 to 125°C, V_{CC} = 5V ±10%

Parameter	Description	Test Conditions		Com'l and Ind Min	Com'l and Ind Max	Military Min	Military Max	Units
V_{IL}	Input Low Voltage			−0.5	0.8	−0.5	0.8	V
V_{IH}	Input High Voltage			2.0	V_{CC}	2.2	V_{CC}	V
V_{OL}	Output Low Voltage	I_{OL} = 2.2mA			0.45		0.45	V
V_{OH}	Output High Voltage	I_{OL} = −400μA		2.4		2.4		V
I_{OFL}	Output Float Leakage	$V_{OUT} = V_{CC}$ to 0.45V			±10		±10	μA
I_{IL}	Input Leakage	$V_{IN} = V_{CC}$ to 0.45V			±10		±10	μA
I_{CC}	Power Supply Current	All Outputs = High	Com'l		100		150	mA
			Ind		150		150	

CAPACITANCE (T_A = 25°C, V_{CC} = GND = 0V)

Parameter	Description	Test Conditions	Min	Max	Units
C_{IN}	Input Capacitance	fc = 1MHz		10	pF
$C_{I/O}$	I/O Capacitance	Unmeasured Pins Returned to GND		20	pF

AC CHARACTERISTICS (GND = 0V)

Com'l T_A = 0 to 70°C, V_{CC} = 5V ±5%
Ind T_A = −40 to 85°C, V_{CC} = 5V ±10%
Mil T_A = −55 to 125°C, V_{CC} = 5V ±10%

Bus Parameters (Note 1)

Parameter	Description	Test Conditions	Com'l and Ind Min	Com'l and Ind Max	Military Min	Military Max	Units
READ CYCLE							
t_{AR}	Address Stable Before $\overline{READ}$ ($\overline{CS}$, C/$\overline{D}$)	Note 2	0		75		ns
t_{RA}	Address Hold Time for $\overline{READ}$ ($\overline{CS}$, C/$\overline{D}$)	Note 2	0		75		ns
t_{RR}	$\overline{READ}$ Pulse Width		250		300		ns
t_{RD}	Data Delay from $\overline{READ}$	3, C_L = 150pF		200		280	ns
t_{DF}	$\overline{READ}$ to Data Floating		10	100	5	120	ns
WRITE CYCLE							
t_{AW}	Address Stable Before $\overline{WRITE}$		0		75		
t_{WA}	Address Hold Time for $\overline{WRITE}$		0		75		
t_{WW}	$\overline{WRITE}$ Pulse Width		250		300		
t_{DW}	Data Setup Time for $\overline{WRITE}$		150		200		
t_{WD}	Data Hold Time for $\overline{WRITE}$		20		80		
t_{RV}	Recovery Time Between WRITES	Note 4	6		6		

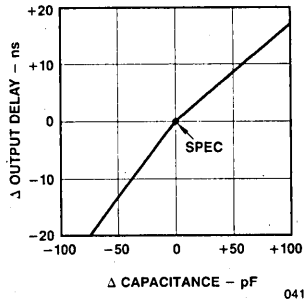
AC CHARACTERISTICS (Cont.) (GND = 0V)

Com'l T_A = 0 to 70°C, V_{CC} = 5V ±5%
Ind T_A = −40 to 85°C, V_{CC} = 5V±10%
Mil T_A = −55 to 125°C, V_{CC} = 5V ±10%

Parameter	Description	Test Conditions	Com'l and Ind		Military		Units
			Min	Max	Min	Max	
OTHER TIMINGS							
t_{CY}	Clock Period	Notes 5, 6	320	1350	320	1350	ns
t_{ϕ}	Clock High Pulse Width		120	$t_{CY}-90$	150	$t_{CY}-100$	ns
t_{ϕ}	Clock Low Pulse Width		90		100		ns
t_R, t_F	Clock Rise and Fall Time			20		20	ns
t_{DTx}	TxD Delay from Falling Edge of TxC			1		1	μs
f_{Tx}	Transmitter Input Clock Frequency 1x Baud Rate		DC	64	DC	64	kHz
	16x Baud Rate		DC	310	DC	310	kHz
	64x Baud Rate		DC	615	DC	615	kHz
t_{TPW}	Transmitter Input Clock Pulse Width 1x Baud Rate		12		12		t_{CY}
	16x and 64x Baud Rate		1		1		t_{CY}
t_{TPD}	Transmitter Input Clock Pulse Delay 1x Baud Rate		15		15		t_{CY}
	16x and 64x Baud Rate		3		3		t_{CY}
f_{Rx}	Receiver Input Clock Frequency 1x Baud Rate		DC	64	DC	64	kHz
	16x Baud Rate		DC	310	DC	310	kHz
	64x Baud Rate		DC	615	DC	615	kHz
t_{RPW}	Receiver Input Clock Pulse Width 1x Baud Rate		12		12		t_{CY}
	16x and 64x Baud Rate		1		1		t_{CY}
t_{RPD}	Receiver Input Clock Pulse Delay 1x Baud Rate		15		15		t_{CY}
	16x and 64x Baud Rate		3		3		t_{CY}
t_{TxRDY}	TxRDY Pin Delay from Center of Last Bit	Note 7		8		8	t_{CY}
$t_{TxRDY\ CLEAR}$	TxRDY ↓ from Leading Edge of $\overline{WR}$	Note 7		400		6	t_{CY}
t_{RxRDY}	RxRDY Pin Delay from Center of Last Bit	Note 7		26		24	t_{CY}
$t_{RxRDY\ CLEAR}$	RxRDY ↓ from Leading Edge of $\overline{RD}$	Note 7		400		6	t_{CY}
t_{IS}	Internal SYNDET Delay from Rising Edge of $\overline{RxC}$	Note 7		26		24	t_{CY}
t_{ES}	External SYNDET Setup Time After Rising Edge of $\overline{RxC}$	Note 7	18		16		t_{CY}
$t_{TxEMPTY}$	TxEMPTY Delay from Center of Last Bit	Note 7	20		20		t_{CY}
t_{WC}	Control Delay from Rising Edge of WRITE (TxEn, $\overline{DTR}$, $\overline{RTS}$)	Note 7	8		8		t_{CY}
t_{CR}	Control to READ Setup Time ($\overline{DSR}$, $\overline{CTS}$)	Note 7	20		20		t_{CY}

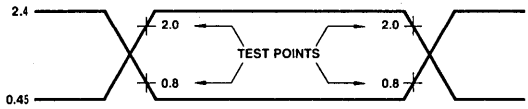
Notes: 1. AC timings measured V_{OH} = 2.0 V_{OL} = 2.0, V_{OL} = 0.8, and with load circuit of Figure 1.
2. Chip Select (CS) and Command/Data (C/D) are considered as Addresses.
3. Assumes that Address is valid before R_D↓.
4. This recovery time is for after a Mode Instruction only. Write Data is allowed only when TxRDY = 1. Recovery time between Writes for Asynchronous Mode is 8 t_{CY} and for Synchronous Mode is 16 t_{CY}.
5. The TxC and RxC frequencies have the following limitations with respect to CLK: For 1x Baud Rate, f_{Tx} or $f_{Rx} \leq 1/(30\ t_{CY})$: For 16x and 64x Baud Rate, f_{Tx} or $f_{Rx} \leq 1/(4.5\ t_{CY})$.
6. Reset Pulse Width = 6 t_{CY} minimum; System Clock must be running during Reset.
7. Status update can have a maximum delay of 28 clock periods from the event affecting the status.

TYPICAL Δ OUTPUT DELAY VS. Δ CAPACITANCE – pF



04133A-3

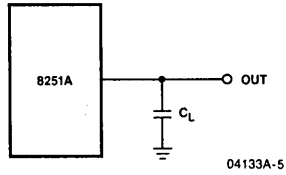
AC TESTING INPUT, OUTPUT WAVEFORM



04133A-4

AC testing: Inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." Timing measurements are made at 2.0V for a logic "1" and 0.8V for a logic "0."

AC TESTING LOAD CIRCUIT

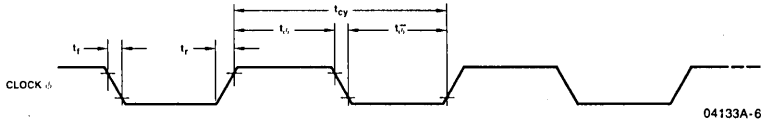


04133A-5

C_L = 150pF

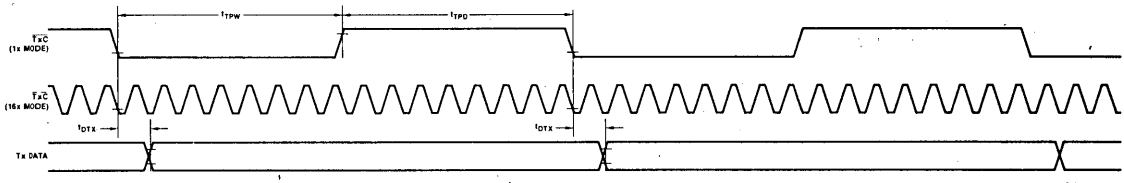
WAVEFORMS

SYSTEM CLOCK INPUT



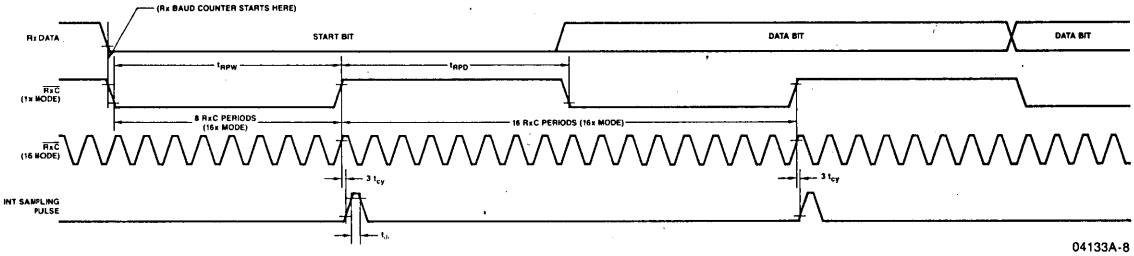
04133A-6

TRANSMITTER CLOCK AND DATA



04133A-7

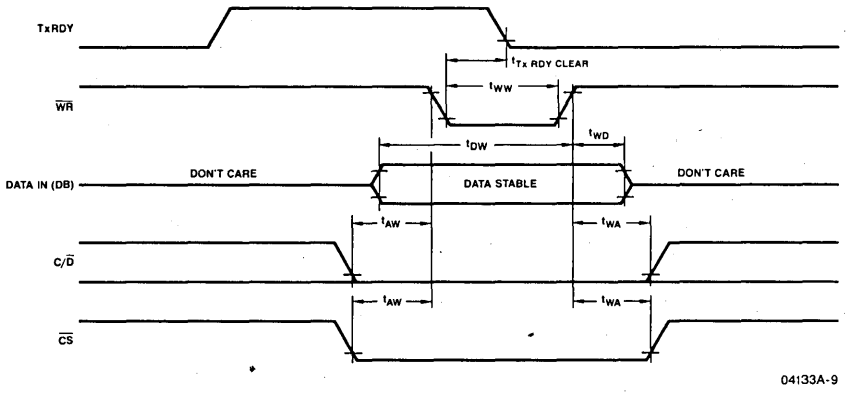
RECEIVER CLOCK AND DATA



04133A-8

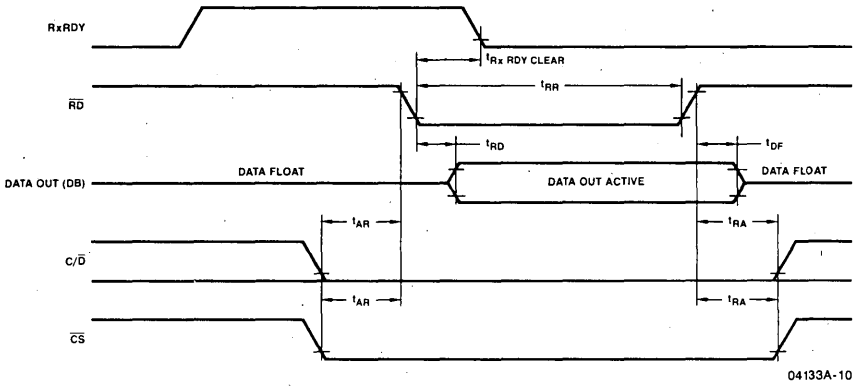
WAVEFORMS (Cont.)

WRITE DATA CYCLE (CPU → USART)



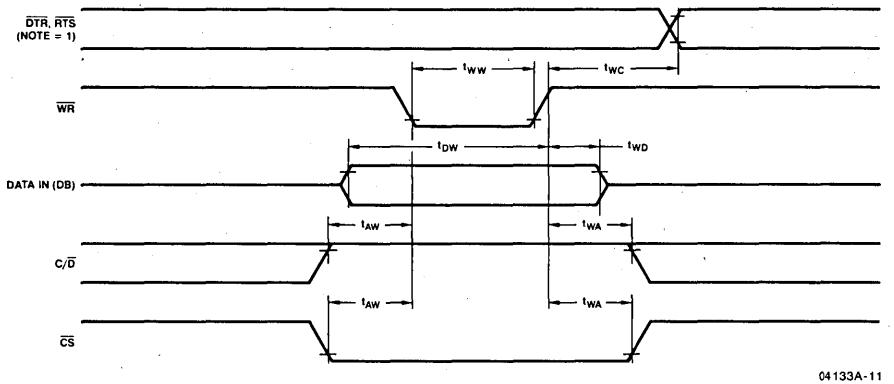
04133A-9

READ DATA CYCLE (CPU ← USART)



04133A-10

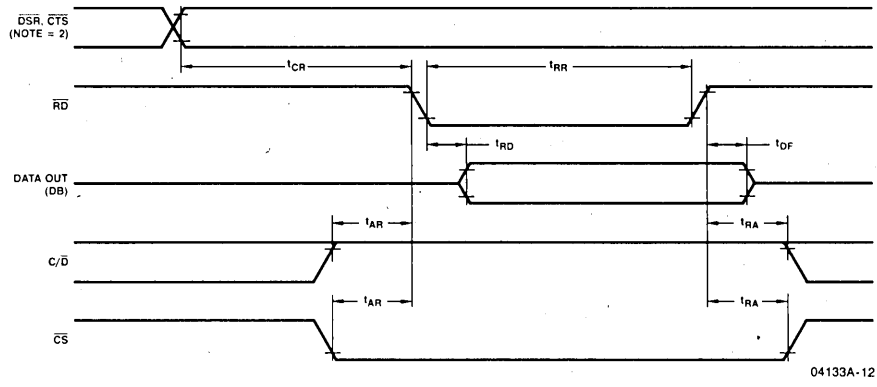
WRITE CONTROL OR OUTPUT PORT CYCLE (CPU → USART)



04133A-11

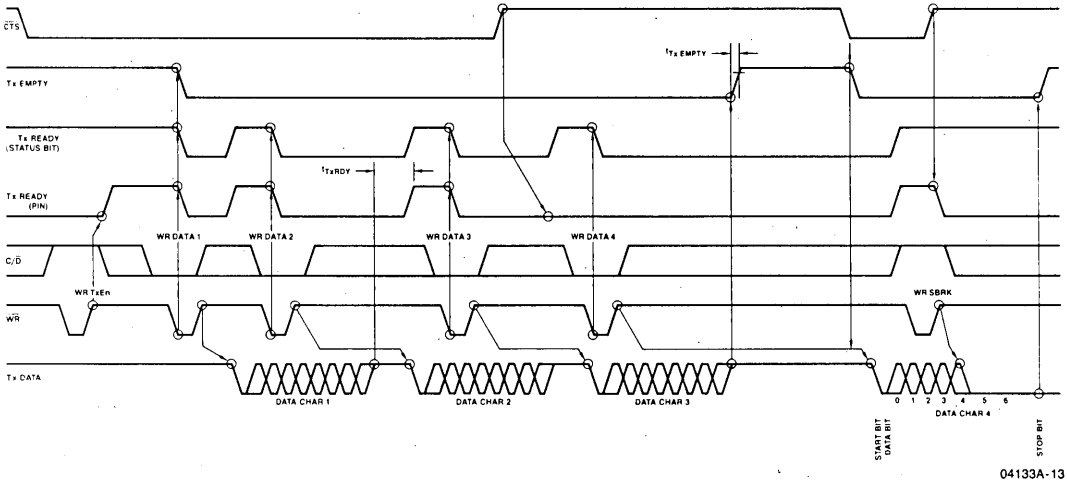
WAVEFORMS (Cont.)

READ CONTROL OR INPUT PORT (CPU ↓ USART)



Notes: 1. T_{WC} includes the response timing of a control byte.
2. T_{CR} includes the effect of CTS on the TxENable circuitry.

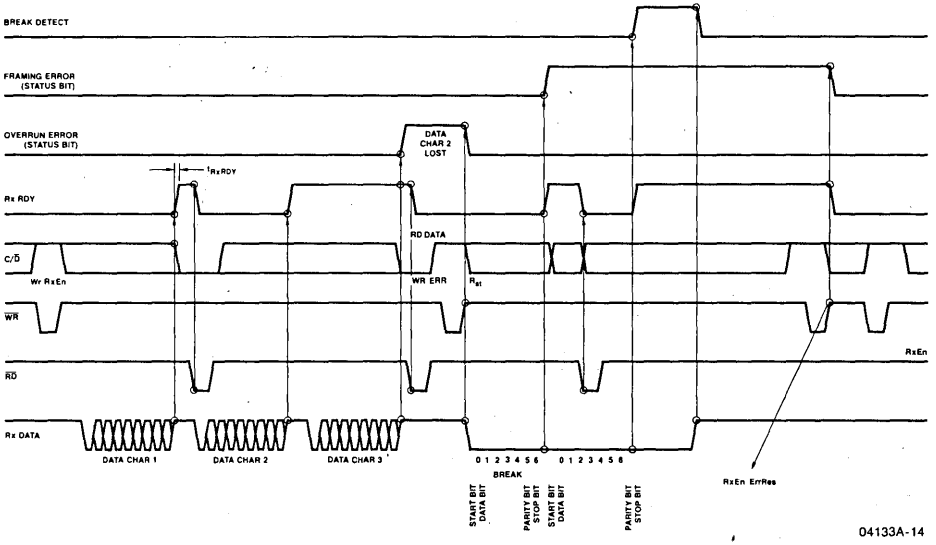
TRANSMITTER CONTROL AND FLAG TIMING (ASYNC MODE)



Example format = 7 bit character with parity and 2 stop bits.

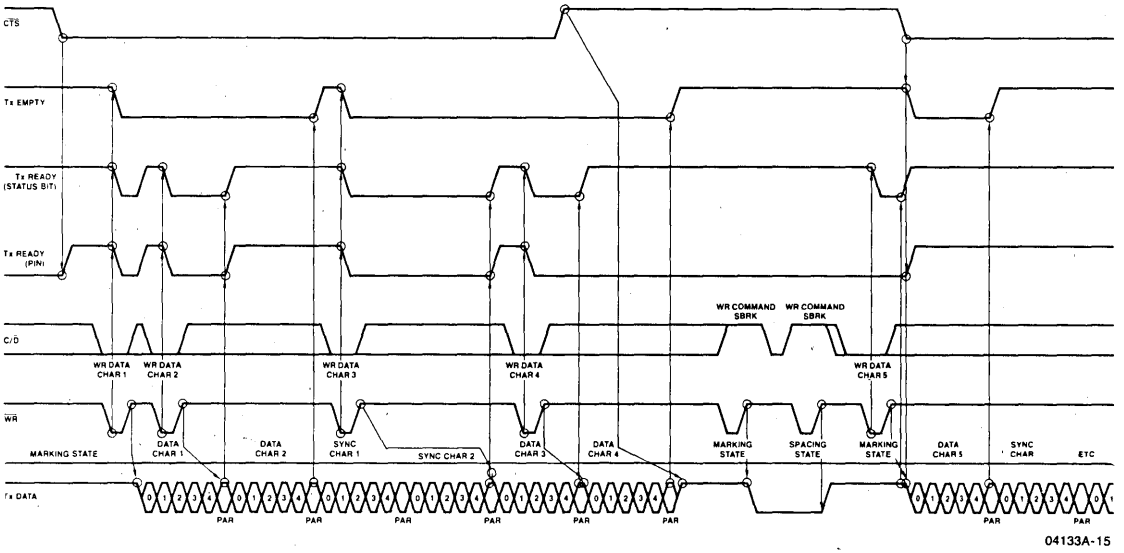
WAVEFORMS (Cont.)

RECEIVER CONTROL AND FLAG TIMING (ASYNC MODE)



Example Format = 7 bit character with parity and 2 stop bits.

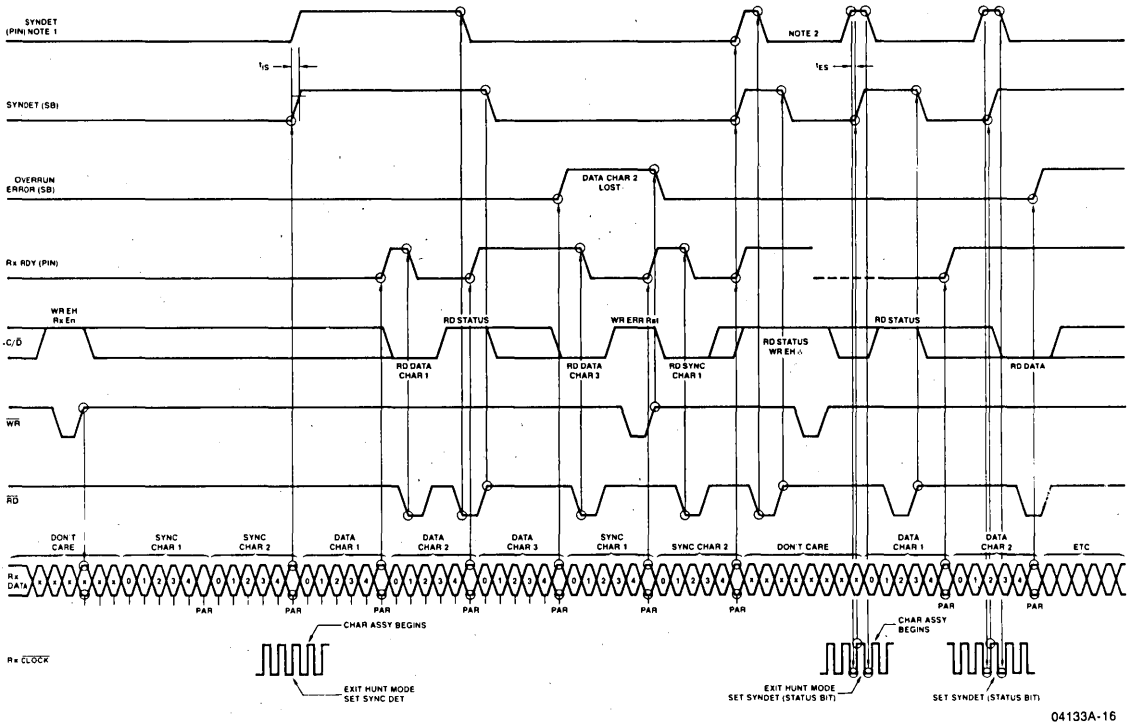
TRANSMITTER CONTROL AND FLAG TIMING (SYNC MODE)



Example Format = 5 bit character with parity, 2 sync characters.

WAVEFORMS (Cont.)

RECEIVER CONTROL AND FLAG TIMING (SYNC MODE)



04133A-16

Notes: 1. Internal sync, 2 sync characters, 5 bits, with parity.
2. External sync, 5 bits, with parity.

8253

Programmable Interval Timer
Advanced MOS/LSI

DISTINCTIVE CHARACTERISTICS

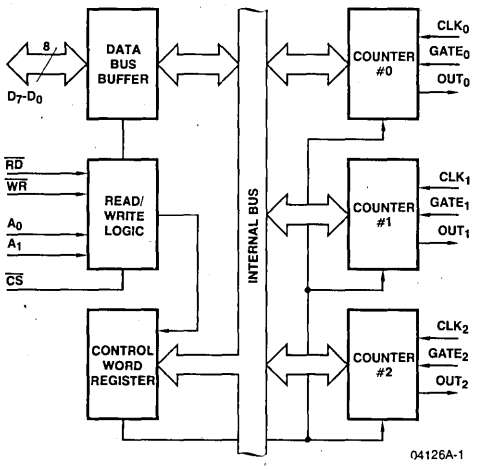
- Both Binary or BCD counting
- Single +5V supply
- Three independent 16-bit counters
- DC to 5MHz
- Programmable counter modes
- Bus oriented I/O

GENERAL DESCRIPTION

The 8253 is a programmable counter/timer chip designed for use with 8080A/8085A microprocessors. They use NMOS technology with a single +5V supply and are direct replacements for Intel's 8253/8253-5.

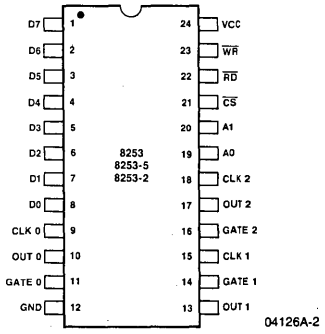
Each device is organized as three independent 16-bit counters, each counter having a rate of up to 5.0MHz. All modes of operation are software programmable. For improved performance devices see **Am9513 System Timing Controller.**

BLOCK DIAGRAM



04126A-1

CONNECTION DIAGRAM – Top View
D-24-1, P-24-1



04126A-2

Note: Pin 1 is marked for orientation.

PIN NAMES

D7-D0	Data bus (8-bit)
CLK N	Counter clock inputs
GATE N	Counter gate inputs
OUT N	Counter outputs
$\overline{RD}$	Read counter
$\overline{WR}$	Write command or data
$\overline{CS}$	Chip select
A0-A1	Counter select
VCC	+5 Volts
GND	Ground

ORDERING INFORMATION

Package Type	Ambient Temperature	Ordering Numbers		
		t_{RD} = 300ns t_{CY} = 380ns	t_{RD} = 200ns t_{CY} = 380ns	t_{RD} = 120ns t_{CY} = 200ns
Molded DIP	0°C ≤ T_A ≤ +70°C	P8253	P8253-5	P8253-2
Hermetic DIP		D8253	D8253-5	D8253-2
Hermetic DIP	−40°C ≤ T_A ≤ +85°C	ID8253	ID8253-5	
Hermetic DIP	−55°C ≤ T_A ≤ +125°C	MD8253B		

04126A-MMP

FUNCTIONAL DESCRIPTION

General

The 8253 is a programmable interval timer/counter specifically designed for use with the 8080A Microcomputer systems. Its function is that of a general-purpose, multitiming element that can be treated as an array of I/O ports in the system software.

The 8253 solves one of the most common problems in any microcomputer system, the generation of accurate time delays under software control. Instead of setting up timing loops in systems software, the programmer configures the 8253 to match his requirements, initializes one of the counters of the 8253 with the desired quantity, then upon command the 8253 will count out the delay and interrupt the CPU when it has completed its tasks. It is easy to see that the software overhead is minimal and that multiple delays can easily be maintained by assignment of priority levels.

Other counter/timer functions that are non-delay in nature, but also common to most microcomputers can be implemented with the 8253.

- Programmable Rate Generator
- Event Counter
- Binary Rate Multiplier
- Real-Time Clock
- Digital One-Shot
- Complex Motor Controller

Data Bus Buffer

This 3-state, bidirectional, 8-bit buffer is used to interface the 8253 to the system data bus. Data is transmitted or received by the buffer upon execution of INput or OUTput CPU instructions. The Data Bus Buffer has three basic functions.

1. Programming the MODES of the 8253
2. Loading the count registers
3. Reading the count values

Read/Write Logic

The Read/Write Logic accepts inputs from the system bus and in turn generates control signals for overall device operation. It is enabled or disabled by CS so that no operation can occur to change the function unless the device has been selected by the system logic.

$\overline{RD}$ (Read)

A "low" on this input informs the 8253 that the CPU is inputting data in the form of a counters value.

$\overline{WR}$ (Write)

A "low" on this input informs the 8253 that the CPU is outputting data in the form of MODE information or loading counters.

A0, A1

These inputs are normally connected to the address bus. Their function is to select one of the three counters to be operated on and to address the control word register for MODE selection.

$\overline{CS}$ (Chip Select)

A "low" on this input enables the 8253. No reading or writing will occur unless the devices is selected. The $\overline{CS}$ input has no effect upon the actual operation of the counter.

Control Word Register

The Control Word Register is selected when A0, A1 are 11. It then accepts information from the data bus buffer and stores it in a register. The information stored in this register controls the operational MODE of each counter, selection of binary or BCD counting and the loading of each count register.

The Control Word Register can only be written into; no read operation of its contents is available.

Counter #0, Counter #1, Counter #2

These three functional blocks are identical in operation so only a single Counter will be described. Each Counter consists of a single, 16-bit, presettable, DOWN counter. The counter can operate in either binary or BCD and its input, gate and output are configured by the selection of MODES stored in the Control Word Register.

The counters are fully independent and each can have separate MODE configuration and counting operation, binary or BCD. Also, there are special features in the control word that handle the loading of the count value so that software overhead can be minimized for these functions.

The reading of the contents of each counter is available to the programmer with simple READ operations for event counting applications, and special commands and logic are included in the 8253 so that the contents of each counter can be read "on the fly" without having to inhibit the clock input.

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A1	A0	
0	1	0	0	0	Load Counter No. 0
0	1	0	0	1	Load Counter No. 1
0	1	0	1	0	Load Counter No. 2
0	1	0	1	1	Write MODE Word
0	0	1	0	0	Read Counter No. 0
0	0	1	0	1	Read Counter No. 1
0	0	1	1	0	Read Counter No. 2
0	0	1	1	1	No-Operation 3-State
1	X	X	X	X	Disable 3-State
0	1	1	X	X	No-Operation 3-State

8253 READ/WRITE PROCEDURE

Write Operations

The systems software must program each counter of the 8253 with the mode and quantity desired. The programmer must write out to the 8253 a MODE control word and the programmed number of count register bytes (1 or 2) prior to actually using the selected counter.

The actual order of the programming is quite flexible. Writing out of the MODE control word can be in any sequence of counter selection, e.g., counter #0 does not have to be first or counter #2 last. Each counter's MODE control word register has a separate address so that its loading is completely sequence independent. (SC0, SC1)

The loading of the Count Register with the actual count value, however, must be done in exactly the sequence programmed in the MODE control word (RL0, RL1). This loading of the counter's count register is still sequence independent like the MODE control word loading, but when a selected count register is to be loaded it **must** be loaded with the number of bytes programmed in the MODE control word (RL0, RL1). The one or two bytes to be loaded in the count register do not have to follow the associated MODE control word. They can be programmed at any time following the MODE control word loading as long as the correct number of bytes is loaded in order.

All counters are down counters. Thus, the value loaded into the count register will actually be decremented. Loading all zeroes into a count register will result in the maximum count (2^{16} for Binary or 10^4 for BCD). In MODE 0 the new count will not restart until the load has been completed. It will accept one of two bytes depending on how the MODE control words (RL0, RL1) are programmed. Then proceed with the restart operation.

Programming Format

MODE Control Word
Counter n

LSB	Count Register byte Counter n
MSB	Count Register byte Counter n

Note: Format shown is a simple example of loading the 8253 and does not imply that it is the only format that can be used.

Alternate Programming Formats

			A1	A0
No. 1		MODE Control Word Counter 0	1	1
No. 2		MODE Control Word Counter 1	1	1
No. 3		MODE Control Word Counter 2	1	1
No. 4	LSB	Count Register Byte Counter 1	0	1
No. 5	MSB	Count Register Byte Counter 1	0	1
No. 6	LSB	Count Register Byte Counter 2	1	0
No. 7	MSB	Count Register Byte Counter 2	1	0
No. 8	LSB	Count Register Byte Counter 0	0	0
No. 9	MSB	Count Register Byte Counter 0	0	0

Note: The exclusive addresses of each counter's count register make the task of programming the 8253 a very simple matter, and maximum effective use of the device will result if this feature is fully utilized.

Read Operations

In most counter applications it becomes necessary to read the value of the count in progress and make a computational decision based on this quantity. Event counters are probably the most common application that uses this function. The 8253 contains logic that will allow the programmer to easily read the contents of any of the three counters without disturbing the actual count in progress.

There are two methods that the programmer can use to read the value of the counters. The first method involves the use of simple I/O read operations of the selected counter. By controlling the A0, A1 inputs to the 8253 the programmer can select the counter to be read (remember that no read operation of the mode register is allowed A0, A1-11). The only requirement with this method is that in order to assure a stable count reading the actual operation of the selected counter **must be inhibited** either by controlling the Gate input or by external logic that inhibits the clock input. The contents of the counter selected will be available as follows:

first I/O Read contains the least significant byte (LSB).

second I/O Read contains the most significant byte (MSB).

Due to the internal logic of the 8253 it is absolutely necessary to complete the entire reading procedure. If two bytes are programmed to be read then two bytes **must** be read before any loading WR command can be sent to the same counter.

Read Operation Chart

A1	A0	RD	
0	0	0	Read Counter No. 0
0	1	0	Read Counter No. 1
1	0	0	Read Counter No. 2
1	1	0	Illegal

Reading While Counting

In order for the programmer to read the contents of any counter without effecting or disturbing the counting operation the 8253 has special internal logic that can be accessed using simple WR commands to the MODE register. Basically, when the programmer wishes to read the contents of a selected counter "on the fly" he loads the MODE register with a special code which latches the present count value into a storage register so that its contents contain an accurate, stable quantity. The programmer then issues a normal read command to the selected counter and the contents of the latched register is available.

MODE Register for Latching Count

A0, A1 = 11

D7	D6	D5	D4	D3	D2	D1	D0
SC1	SC0	0	0	X	X	X	X

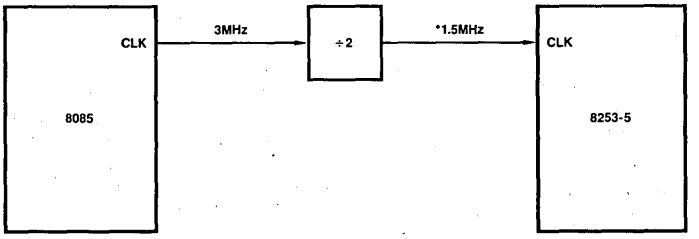
SC1, SC0 – specify counter to be latched.

D5, D4 – 00 designates counter latching operation.

X – don't care.

The same limitation applies to this mode of reading the counter as the previous method. That is, it is mandatory to complete the entire read operation as programmed. This command has no effect on the counter's mode.

Clock Interface*



*If an 8085 clock output is to drive an 8253-5 clock input, it must be reduced to 2MHz or less.

OPERATIONAL DESCRIPTION

General

The complete functional definition of the 8253 is programmed by the systems software. A set of control words **must** be sent out by the CPU to initialize each counter of the 8253 with the desired MODE and quantity information. These control words program the MODE, Loading sequence and selection of binary or BCD counting.

Once programmed, the 8253 is ready to perform whatever timing tasks it is assigned to accomplish.

The actual counting operation of each counter is completely independent and additional logic is provided on-chip so that the usual problems associated with efficient monitoring and management of external, asynchronous events or rates to the microcomputer system have been eliminated.

Programming the 8253

All of the MODES for each counter are programmed by the systems software by simple I/O operations.

Each counter of the 8253 is individually programmed by writing a control word into the Control Word Register (A0, A1 = 11).

Control Word Format

D7	D6	D5	D4	D3	D2	D1	D0
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Definition of Control

SC – Select Counter:

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

RL – Read/Load:

RL1	RL0	
0	0	Counter Latching operation.
1	0	Read/Load most significant byte only.
0	1	Read/Load least significant byte only.
1	1	Read/Load least significant byte first, then most significant byte.

M – MODE:

M2	M1	M0	
0	0	0	MODE 0
0	0	1	MODE 1
X	1	0	MODE 2
X	1	1	MODE 3
1	0	0	MODE 4
1	0	1	MODE 5

BCD:

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

Counter Loading

The count register is not loaded until the count value is written (one or two bytes, depending on the mode selected by the RL bits), followed by a rising edge and a falling edge of the clock. Any read of the counter prior to that falling clock edge may yield invalid data.

MODE DEFINITION

MODE 0: Interrupt on Terminal Count

The output will be initially low after the mode set operation. After the count is loaded into the selected count register, the output will remain low and the counter will count. When terminal count is reached the output will go high and remain high until the selected count register is reloaded with the mode or a new count is loaded. The counter continues to decrement after terminal count has been reached.

Rewriting a counter register during counting results in the following:

1. Write 1st byte stops the current counting.
2. Write 2nd byte starts the new count.

MODE 1: Programmable One-Shot

The output will go low on the count following the rising edge of the gate input.

The output will go high on the terminal count. If a new count value is loaded while the output is low it will not affect the duration of the one-shot pulse until the succeeding trigger. The current count can be read at any time without affecting the one-shot pulse.

The one-shot is retriggerable, hence the output will remain low for the full count after any rising edge of the gate input.

MODE 2: Rate Generator

Divide by N counter. The output will be low for one period of the input clock. The period from one output pulse to the next equals the number of input counts in the count register. If the count register is reloaded between output pulses the present period will not be affected, but the subsequent period will reflect the new value.

The gate input, when low, will force the output high. When the gate input goes high, the counter will start from the initial count. Thus, the gate input can be used to synchronize the counter.

When this mode is set, the output will remain high until after the count register is loaded. The output then can also be synchronized by software.

MODE 3: Square Wave Rate Generator

Similar to MODE 2 except that the output will remain high until one half the count has been completed (for even numbers) and go low for the other half of the count. This is accomplished by decrementing the counter by two on the falling edge of each clock pulse. When the counter reaches terminal count, the state of the output is changed and the counter is reloaded with the full count and the whole process is repeated.

If the count is odd and the output is high, the first clock pulse (after the count is loaded) decrements the count by one. Subsequent clock pulses decrement the clock by two. After timeout, the output goes low and the full count is reloaded. The first clock pulse (following the reload) decrements the counter by three. Subsequent clock pulses decrement the count by two until timeout. Then the whole process is repeated. In this way, if the count is odd, the output will be high for $(N + 1)/2$ counts and low for $(N - 1)/2$ counts.

MODE 4: Software-Triggered Strobe

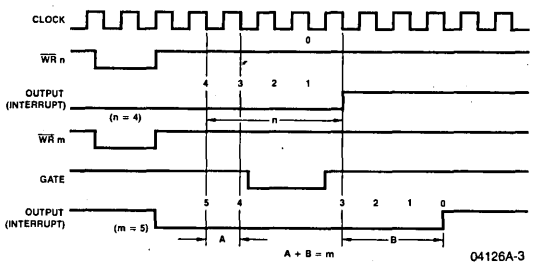
After the mode is set, the output will be high. When the count is loaded, the counter will begin counting. On terminal count, the output will go low for one input clock period, then will go high again.

If the count register is reloaded between output pulses the present period will not be affected, but the subsequent period will reflect the new value. The count will be inhibited while the gate input is low. Reloading the counter register will restart counting beginning with the new number.

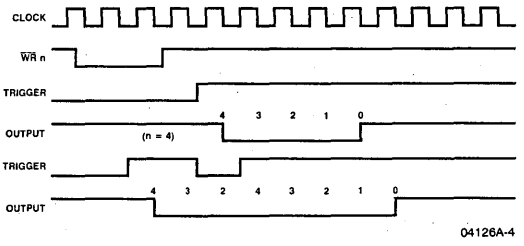
MODE 5: Hardware-Triggered Strobe

The counter will start counting after the rising edge of the trigger input and will go low for one clock period when the terminal count is reached. The counter is retriggerable. The output will not go low until the full count after the rising edge of any trigger.

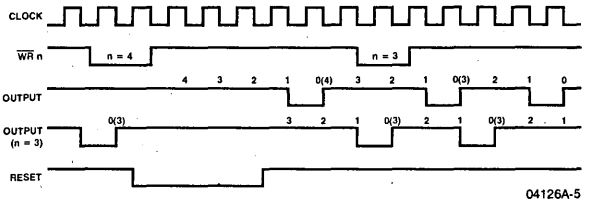
MODE 0. Interrupt on Terminal Count.



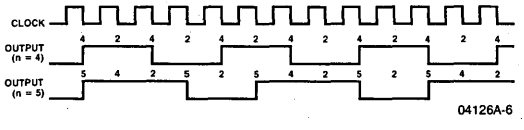
MODE 1. Programmable One-Shot.



MODE 2. Rate Generator.



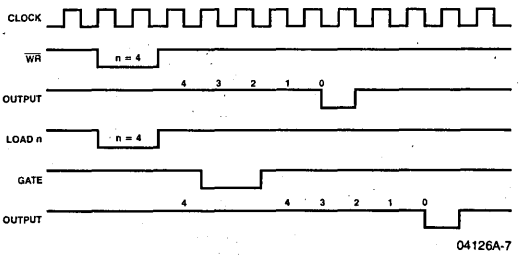
MODE 3. Square Wave Generator.



Gate Pin Operations Summary

Modes \ Signal Status	Low Or Going Low	Rising	High
0	Disables counting	–	Enables counting
1	–	1) Initiates counting 2) Resets output after next clock	–
2	1) Disables counting 2) Sets output immediately high	1) Reloads counter 2) Initiates counting	Enables counting
3	1) Disables counting 2) Sets output immediately high	Initiates counting	Enables counting
4	Disables counting	–	Enables counting
5	–	Initiates counting	–

MODE 4. Software-Triggered Strobe.



MODE 5. Hardware-Triggered Strobe.

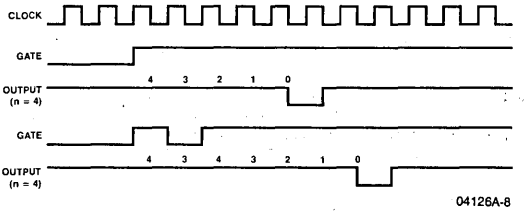


Figure 1. 8253 Timing Diagrams

MAXIMUM RATINGS above which useful life may be impaired

Storage Temperature	−65 to +150°C
Voltage On Any Pin with Respect to Ground	−0.5 to +7.0V
Power Dissipation	1W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	T_A	V_{CC}	V_{SS}
D8253, P8253 D8253-5, P8253-5	0°C ≤ T_A ≤ +70°C	5.0V ± 5%	0V
D8253-2, P8253-2		5.0V ± 10%	0V
ID8253 ID8253-5	−40°C ≤ T_A ≤ +85°C	5.0V ± 10%	0V
MD8253B	−55°C ≤ T_A ≤ +125°C	5.0V ± 10%	0V

DC CHARACTERISTICS

Commercial	8253, 8253-5	T_A = 0 to +70°C, V_{CC} = 5.0V ± 5%;
Commercial	8253-2	T_A = 0 to +70°C, V_{CC} = 5.0V ± 10%;
Industrial		T_A = −40 to +85°C, V_{CC} = 5.0V ± 10%;
Military		T_A = −55 to +125°C, V_{CC} = 5.0V ± 10%

				8253		8253-5		8253-2		
Parameters	**Description**		**Test Conditions**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Units**
V_{IL}	Input Low Voltage			−.5	.8	−.5	.8	−.5	.8	V
V_{IH}	Input High Voltage	Comm, Ind		2.2	V_{CC}+.5V	2.2	V_{CC}+.5V	2.2	V_{CC}+.5V	V
		Mil		2.4						
V_{OL}	Output Low Voltage	8253	I_{OL} = 1.6mA		.45					V
		8253-5	I_{OL} = 2.2mA				.45			
		8253-2	I_{OL} = 2.2mA						.45	
V_{OH}	Output High Voltage	8253	I_{OH} = −150μA	2.4						V
		8253-5	I_{OH} = −400μA			2.4				
		8253-2	I_{OH} = −400μA					2.4		
I_{IL}	Input Load Current	Comm, Ind	V_{IN} = V_{CC} to 0V		±10		±10		±10	μA
		Mil			±20					
I_{OFL}	Output Float Leakage	Comm, Ind	V_{OUT} = V_{CC} to 0V		±10		±10		±10	μA
		Mil			±20					
I_{CC}	V_{CC} Supply Current	Comm			140		140		140	mA
		Ind			160		160			
		Mil			160					

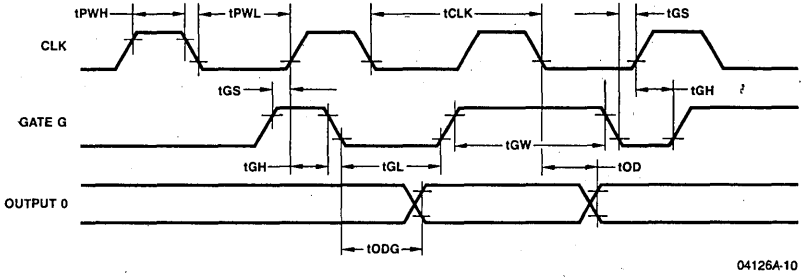
CAPACITANCE T_A = 25°C; V_{CC} = GND = 0V

Parameters	Description	Test Conditions	Min	Typ	Max	Units
C_{IN}	Input Capacitance	f_c = 1MHz			10	pF
$C_{I/O}$	I/O Capacitance	Unmeasured pins returned to V_{SS}			20	pF

CLOCK AND GATE TIMING

Parameters	Description		8253 Min	8253 Max	8253-5 Min	8253-5 Max	8253-2 Min	8253-2 Max	Units
t_{CLK}	Clock Period	Comm, Ind	380	DC	380	DC	200	DC	ns
		Mil	400	DC					
t_{PWH}	High Pulse Width	Comm, Ind	230		230		90		ns
		Mil	250						
t_{PWL}	Low Pulse Width		150		150		90		ns
t_{GW}	Gate Width High		150		150		120		ns
t_{GL}	Gate Width Low		100		100		80		ns
t_{GS}	Gate Setup Time to CLK ↑		100		100		60		ns
t_{GH}	Gate Hold Time After CLK ↑	Comm, Ind	50		50		50		ns
		Mil	100						
t_{OD}	Output Delay from CLK ↓ (Note 1)			400		400		250	ns
t_{ODG}	Output Delay from Gate ↓ (Note 1)			300		300		150	ns

Note: 1. Test Conditions: 8253 C_L = 100pF; 8253-5, 8253-2 C_L = 150pF.



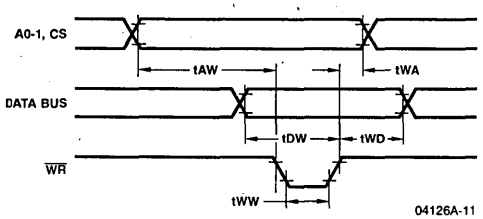
8253
AC CHARACTERISTICS

Commercial 8253, 8253-5 T_A = 0 to +70°C, V_{CC} = 5.0V ± 5%, GND = 0V;
Commercial 8253-2 T_A = 0 to +70°C, V_{CC} = 5.0V ± 10%, GND = 0V;
Industrial T_A = −40 to +85°C, V_{CC} = 5.0V ± 10%, GND = 0V;
Military T_A = −55 to +125°C, V_{CC} = 5.0V ± 10%, GND = 0V

Parameters	Description		8253 Min	8253 Max	8253-5 Min	8253-5 Max	8253-2 Min	8253-2 Max	Units
Read Cycle									
t_{AR}	Address Stable Before $\overline{READ}$		50		30		25		ns
t_{RA}	Address Hold Time for $\overline{READ}$	Comm, Ind	5		5		5		ns
		Mil	10						
t_{RR}	$\overline{READ}$ Pulse Width		400		300		150		ns
t_{RD}	Data Delay from $\overline{READ}$ (Note 2)			300		200		120	ns
t_{DF}	$\overline{READ}$ to Data Floating	Comm, Ind	25	125	25	100	25	100	ns
		Mil	25	175					
t_{RV}	Recovery Time Between $\overline{READ}$ and Any Other Control Signal		1		1		500		μs
Write Cycle									
t_{AW}	Address Stable Before $\overline{WRITE}$		50		30		0		ns
t_{WA}	Address Hold Time for $\overline{WRITE}$		30		30		0		ns
t_{WW}	$\overline{WRITE}$ Pulse Width		400		300		150		ns
t_{DW}	Data Setup Time for $\overline{WRITE}$		300		250		100		ns
t_{WD}	Data Hold Time for $\overline{WRITE}$		40		30		0		ns
t_{RV}	Recovery Time Between $\overline{WRITE}$ and Any Other Control Signal (Note 3)		1		1		500		μs

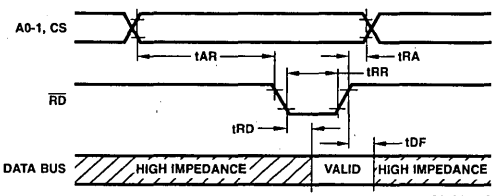
Notes: 1. AC timings measured at V_{OH} = 2.2, V_{OL} = 0.8.
2. Test Conditions: 8253, C_L = 100pF, 8253-5, 8253-2, C_L = 150pF.
3. t_{RV} applies for any READ or WRITE that occurs regardless of the state of the CS input.
4. If the clock occurs less than 100ns after the rising edge of READ or WRITE, the counter selected during the READ or WRITE could be affected.

Write Timing



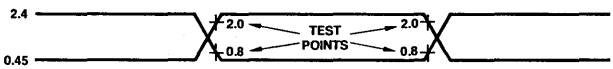
04126A-11

Read Timing



04126A-12

Input Waveforms for AC Tests



04126A-13

8255A

Programmable Peripheral Interface
Advanced MOS/LSI

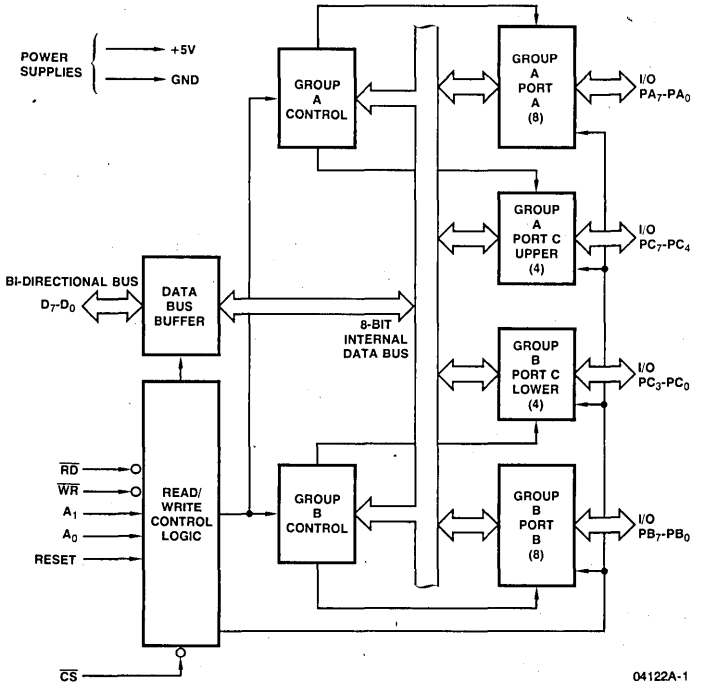
DISTINCTIVE CHARACTERISTICS

- Direct bit set/reset capability easing control application interface
- Reduces system package count
- Improved DC driving capability
- 24 programmable I/O pins
- Completely TTL compatible
- Fully compatible with 8080A and 8085A microprocessor families
- Improved timing characteristics
- Military version available

GENERAL DESCRIPTION

The 8255A is a general purpose programmable I/O device designed for use with 8080A and 8085A microprocessors. It has 24 I/O pins which may be individually programmed in two groups of twelve and used in three major modes of operation. In the first mode, each group of twelve I/O pins may be programmed in sets of 4 and 8 to be input or output. In Mode 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining four pins three are used for handshaking and interrupt control signals. The third mode of operation (Mode 2) is a bidirectional bus mode which uses eight lines for a bidirectional bus, and five lines, borrowing one from the other group, for handshaking.

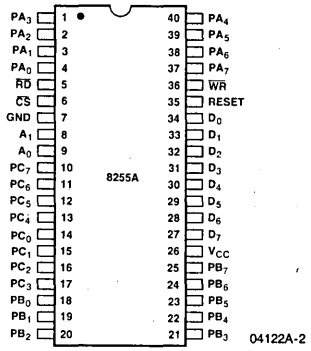
8255A BLOCK DIAGRAM



ORDERING INFORMATION

Package Type	Ambient Temperature Specification	Order Numbers	
		t_{WW} = 400ns	t_{WW} = 300ns
Molded DIP	$0°C \leq T_A \leq 70°C$	P8255A	P8255A-5
Hermetic DIP		D8255A	D8255A-5
Hermetic DIP	$-40°C \leq T_A \leq 85°C$	ID8255A	ID8255A-5
Hermetic DIP	$-55°C \leq T_A \leq 125°C$	MD8255AB	

CONNECTION DIAGRAM – Top View
D-40-1, P-40-1



PIN NAMES

D_7-D_0	Data Bus (Bi-Directional)
Reset	Reset Input
$\overline{CS}$	Chip Select
$\overline{RD}$	Read Input
$\overline{WR}$	Write Input
A_0, A_1	Port Address
PA_7-PA_0	Port A (Bit)
PB_7-PB_0	Port B (Bit)
PC_7-PC_0	Port C (Bit)
V_{CC}	+5 Volts
GND	0 Volts

MAXIMUM RATINGS above which useful life may be impaired

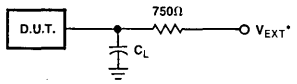
Storage Temperature	−65 to +150°C
V_{CC} with Respect to V_{SS}	−0.5 to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5 to +7.0V
Power Dissipation	1.0W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

CAPACITANCE T_A = 25°C; V_{CC} = GND = 0V

Parameter	Description	Test Conditions	Min.	Typ.	Max.	Unit
C_{IN}	Input Capacitance	fc = 1MHz			10	pF
$C_{I/O}$	I/O Capacitance	Unmeasured pins returned to GND			20	pF

TEST LOAD CIRCUIT (FOR DATA BUS)



*V_{EXT} is set at various voltages during testing to guarantee the specification.

OPERATING RANGE

Part Number	Ambient Temperature	V_{CC}	V_{SS}
D8255A, P8255A D8255A-5, P8255A-5	0°C ≤ T_A ≤ 70°C	5V ±5%	0V
ID8255A ID8255A-5	−40°C ≤ T_A ≤ 85°C	5V ±10%	0V
MD8255AB	−55°C ≤ T_A ≤ 125°C	5V ±10%	0V

DC CHARACTERISTICS over operating range

Parameters	Description	Test Conditions	Min	Max	Units
V_{IL}	Input Low Voltage		−0.5	0.8	Volts
V_{IH}	Input High Voltage		2.0	V_{CC}	Volts
V_{OL}(DB)	Output Low Voltage (Data Bus)	I_{OL} = 2.5mA		0.45	Volts
V_{OL}(PER)	Output Low Voltage (Peripheral Port)	I_{OL} = 1.7mA		0.45	Volts
V_{OH}(DB)	Output High Voltage (Data Bus)	I_{OH} = −400μA	2.4		Volts
V_{OH}(PER)	Output High Voltage (Peripheral Port)	I_{OH} = −200μA	2.4		Volts
I_{DAR} (Note 1)	Darlington Drive Current	R_{EXT} = 750Ω; V_{EXT} = 1.5V	−1.0	−4.0	mA
I_{CC}	Power Supply Current			120	mA
I_{IL}	Input Load Current	V_{IN} = V_{CC} to 0V		±10	μA
I_{OFL}	Output Float Leakage	V_{OUT} = V_{CC} to 0V		±10	μA

Note 1: Available on any 8 pins from Port B and C.

AC CHARACTERISTICS Over Operating Range

BUS PARAMETERS:

Read:

Parameter	Description	8255A ID8255A Min.	8255A ID8255A Max.	8255A-5 ID8255A-5 Min.	8255A-5 ID8255A-5 Max.	MD8255AB Min.	MD8255AB Max.	Unit
t_{AR}	Address Stable Before READ	0		0		0		ns
t_{RA}	Address Stable After READ	0		0		0		ns
t_{RR}	READ Pulse Width	300		300		300		ns
t_{RD}	Data Valid From READ (Note 1)		250		200		250	ns
t_{DF}	Data Float After READ	10	150	10	100	10	150	ns
t_{RV}	Time Between READs and/or WRITEs	850		850		850		ns

Write:

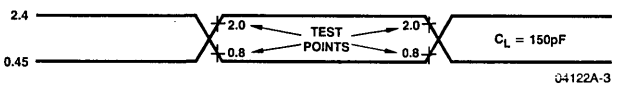
Parameter	Description	8255A ID8255A Min.	8255A ID8255A Max.	8255A-5 ID8255A-5 Min.	8255A-5 ID8255A-5 Max.	MD8255AB Min.	MD8255AB Max.	Unit
t_{AW}	Address Stable Before WRITE	0		0		0		ns
t_{WA}	Address Stable After WRITE	20		20		20		ns
t_{WW}	WRITE Pulse Width	400		300		400		ns
t_{DW}	Data Valid to WRITE (T.E.)	100		100		100		ns
t_{WD}	Data Valid After WRITE	30		30		30		ns

Other Timings:

Parameter	Description	8255A ID8255A Min.	8255A ID8255A Max.	8255A-5 ID8255A-5 Min.	8255A-5 ID8255A-5 Max.	MD8255AB Min.	MD8255AB Max.	Unit
t_{WB}	WR = 1 to Output (Note 1)		350		350		350	ns
t_{IR}	Peripheral Data Before RD	0		0		0		ns
t_{HR}	Peripheral Data After RD	0		0		0		ns
t_{AK}	ACK Pulse Width	300		300		300		ns
t_{ST}	STB Pulse Width	500		500		500		ns
t_{PS}	Per. Data Before T.E. of STB	0		0		0		ns
t_{PH}	Per. Data After T.E. of STB	180		180		180		ns
t_{AD}	ACK = 0 to Output (Note 1)		300		300		300	ns
t_{KD}	ACK = 1 to Output Float	20	250	20	250	20	250	ns
t_{WOB}	WR = 1 to OBF = 0 (Note 1)		650		650		650	ns
t_{AOB}	ACK = 0 to OBF = 1 (Note 1)		350		350		350	ns
t_{SIB}	STB = 0 to IBF = 1 (Note 1)		300		300		300	ns
t_{RIB}	RD = 1 to IBF = 0 (Note 1)		300		300		300	ns
t_{RIT}	RD = 0 to INTR = 0 (Note 1)		400		400		400	ns
t_{SIT}	STB = 1 to INTR = 1 (Note 1)		300		300		300	ns
t_{AIT}	ACK = 1 to INTR = 1 (Note 1)		350		350		350	ns
t_{WIT}	WR = 1 to INTR = 0 (Note 1, 3)		450		450		450	ns

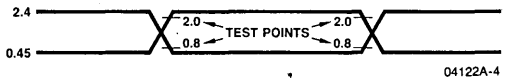
Notes: 1. Test Conditions: 8255A/ID8255A/MD8255AB: C_L = 100pF; 8255A-5: C_L = 150pF.
2. Period of Reset pulse must be at least 50μs during or after power on. Subsequent Reset pulse can be 500ns min.
3. INTR ↑ may occur as early as WR ↓.

AC TESTING INPUT, OUTPUT WAVEFORM

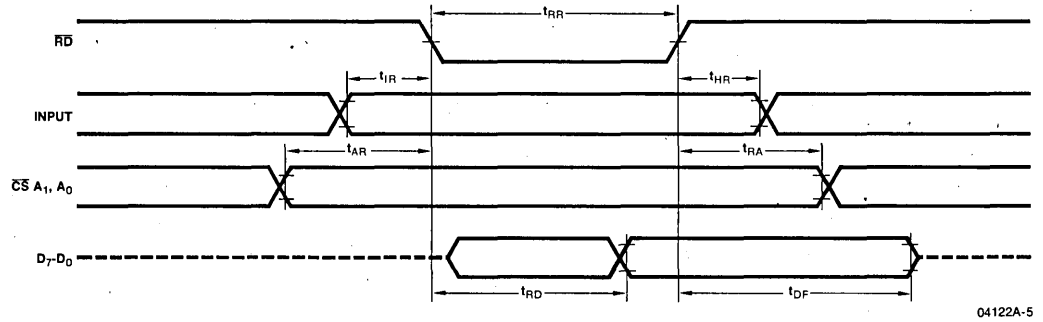


AC testing: Inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0."
Timing measurements are made at 2.0V for a logic "1" and 0.8V for a logic "0."

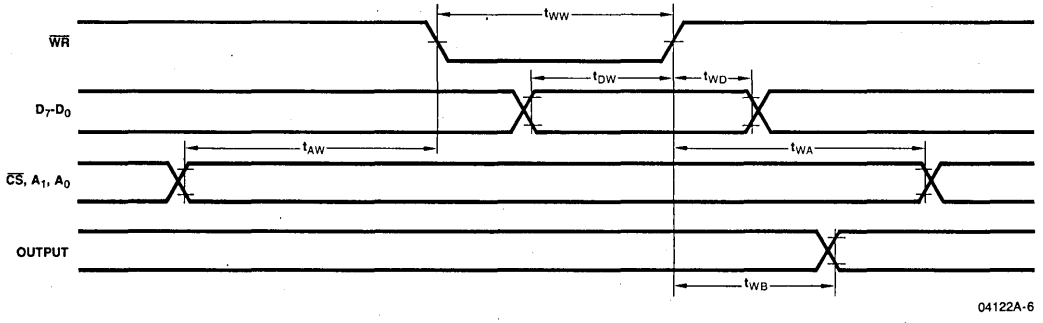
WAVEFORMS



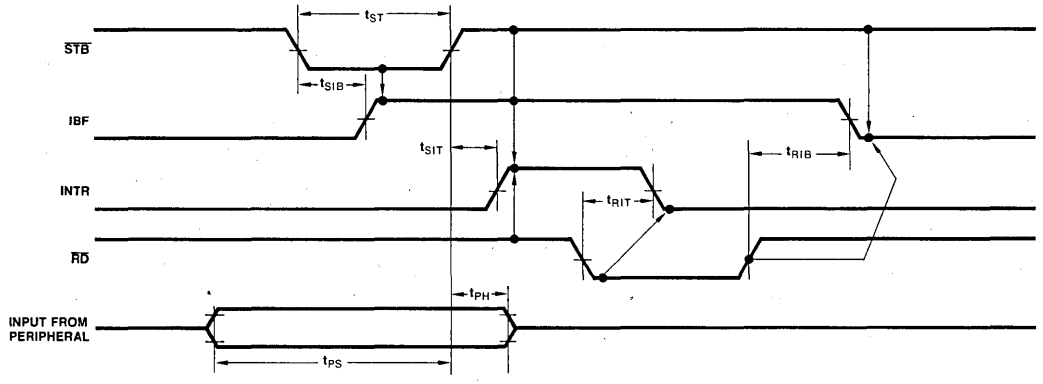
Input Waveforms For A.C. Tests



Mode 0 (Basic Input)

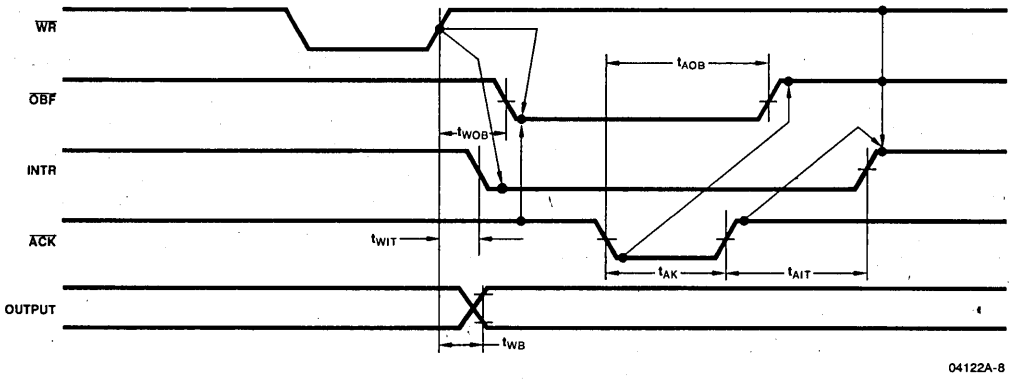


Mode 0 (Basic Output)

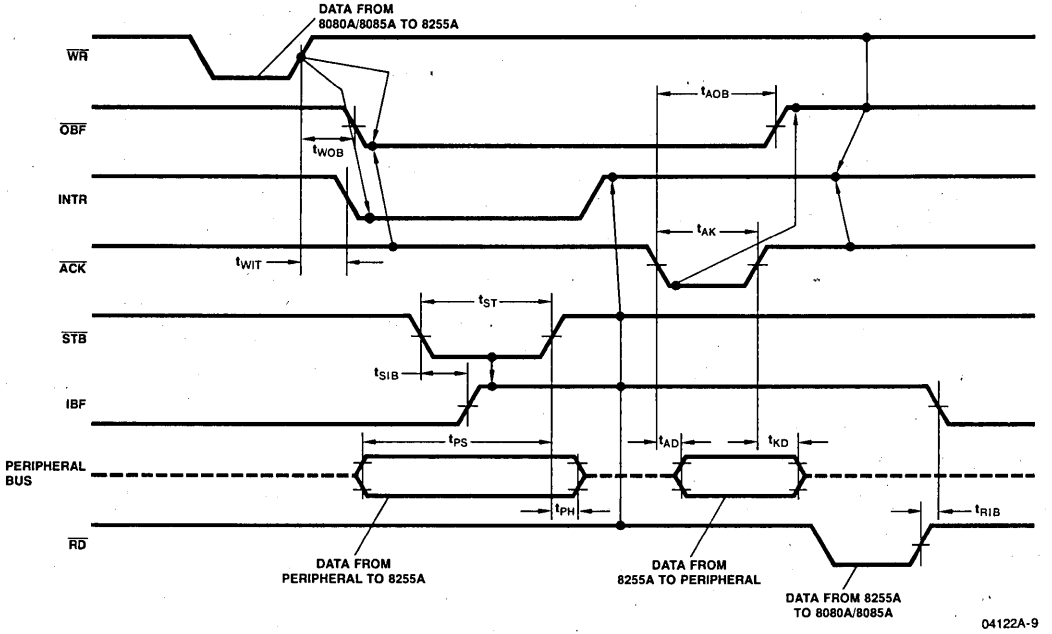


Mode 1 (Strobed Input)

WAVEFORMS (Cont.)



Mode 1 (Strobed Output)



Note: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • $\overline{MASK}$ • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • $\overline{MASK}$ • $\overline{ACK}$ • $\overline{WR}$)

Mode 2 (Bi-directional)

8259A

Programmable Interrupt Controller

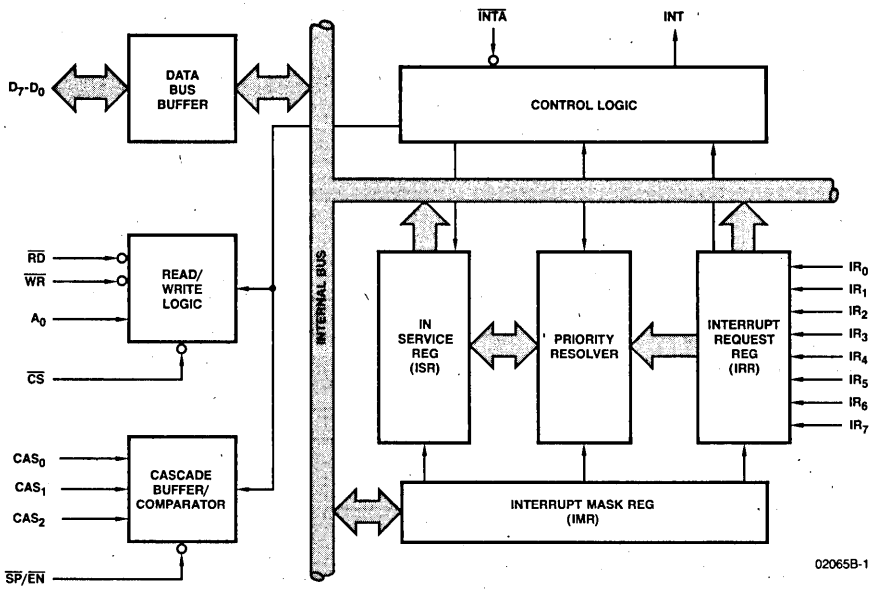
DISTINCTIVE CHARACTERISTICS

- Compatible with 8080/85/86/88 CPUs
- Eight priority levels
- Expandable to 64 levels
- Programmable interrupt modes
- Maskable interrupt priority
- No clock input needed
- Three speed options

GENERAL DESCRIPTION

The 8259A PIC reduces CPU software overhead for handling multi-level priority interrupts. It has several programmable operating modes, so that it can be optimized for different system requirements. The PIC can be expanded from 8 to a maximum of 64 vectored interrupts. It has static circuitry and requires no clock input. The 8259A PIC runs on a single +5V supply, is implemented in NMOS technology, and is packaged in a 28-pin DIP.

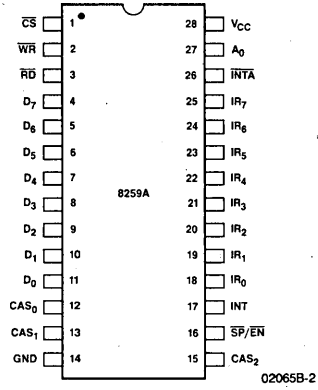
BLOCK DIAGRAM



ORDERING INFORMATION

Package Type	Ambient Temperature Specification	Order Number		
		5MHz	8MHz	2MHz
Molded DIP	$0°C \leq T_A \leq 70°C$	P8259A	P8259A-2	P8259A-8
Hermetic DIP		D8259A	D8259A-2	D8259A-8
Hermetic DIP	$-40°C \leq T_A \leq 85°C$	ID8259A		
Hermetic DIP	$-55°C \leq T_A \leq 125°C$	MD8259AB		

CONNECTION DIAGRAM – Top View
D-28, P-28



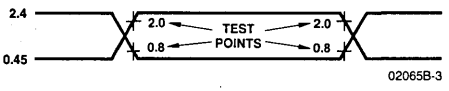
Note: Pin 1 is marked for orientation.

TABLE 1. PIN DESCRIPTION

Symbol	Pin No.	Type	Name and Function
V_{CC}	28	I	**Supply:** +5V Supply.
GND	14	I	**Ground.**
$\overline{CS}$	1	I	**Chip Select:** A low on this pin enables $\overline{RD}$ and $\overline{WR}$ communication between the CPU and the 8259A. INTA functions are independent of $\overline{CS}$.
$\overline{WR}$	2	O	**Write:** A low on this pin when $\overline{CS}$ is low enables the 8259A to accept command words from the CPU.
$\overline{RD}$	3	I	**Read:** A low on this pin when $\overline{CS}$ is low enables the 8259A to release status onto the data bus for the CPU.
D_7-D_0	4-11	I/O	**Bidirectional Data Bus:** Control, status and interrupt-vector information is transferred via this bus.
CAS_0-CAS_2	12, 13, 15	I/O	**Cascade Lines:** The CAS lines form a private 8259A bus to control a multiple 8259A structure. These pins are outputs for a master 8259A and inputs for a slave 8259A.
$\overline{SP}/\overline{EN}$	16	I/O	**Slave Program/Enable Buffer:** This is a dual function pin. When in the Buffered Mode it can be used as an output to control buffer transceivers ($\overline{EN}$). When not in the buffered mode it is used as an input to designate a master ($\overline{SP}$ = 1) or slave ($\overline{SP}$ = 0).
INT	17	O	**Interrupt:** This pin goes high whenever a valid interrupt request is asserted. It is used to interrupt the CPU, thus it is connected to the CPU's interrupt pin.
IR_0-IR_7	18-25	I	**Interrupt Requests:** Asynchronous inputs. An interrupt request is executed by raising an IR input (low to high), and holding it high until it is acknowledged (Edge Triggered Mode), or just by a high level on an IR input (Level Triggered Mode).
$\overline{INTA}$	26	I	**Interrupt Acknowledge:** This pin is used to enable 8259A interrupt-vector data onto the data bus by a sequence of interrupt acknowledge pulses issued by the CPU.
A_0	27	I	**A_0 Address Line:** This pin acts in conjunction with the $\overline{CS}$, $\overline{WR}$, and $\overline{RD}$ pins. It is used by the 8259A to decipher various Command Words the CPU writes and status the CPU wishes to read. It is typically connected to the CPU A_0 address line (A_1 for 8086/88 CPU's).

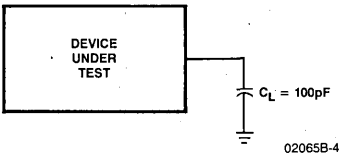
AC TESTING INPUT, OUTPUT WAVEFORM

Input/Output



Note: AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0". Timing measurements are made at 2.0V for a logic "1" and 0.8V for a logic "0".

AC TESTING LOAD CIRCUIT



C_L = 100pF
C_L Includes Jig Capacitance

MAXIMUM RATINGS*

Ambient Temperature Under Bias	−55 to +125°C
Storage Temperature	−65 to +150°C
Voltage on Any Pin with Respect to Ground	−0.5 to +7.0V
Power Dissipation	1.0W

*Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

OPERATING RANGE

Part Number	T_A	V_{CC}	V_{SS}
P8259A D8259A P8259A-2 D8259A-2 P8259A-8 D8259A-8	0°C ≤ T_A ≤ + 70°C	5.0V ± 10%	0V
ID8259A	−40°C ≤ T_A ≤ +85°C	5.0V ± 10%	0V
MD8259AB	−55°C ≤ T_A ≤ +125°C	5.0V ± 10%	0V

DC CHARACTERISTICS

			Commercial 8259A, 8259A-2 8259A-8 T_A = 0 to +70°C V_{CC} = 5V ± 10%		Industrial ID8259A T_A = −40 to +85°C V_{CC} = 5V ± 10%		Military MD8259A T_A = −55 to +125°C V_{CC} = 5V ± 10%		
Parameters	**Description**	**Test Conditions**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Units**
V_{IL}	Input Low Voltage		−.5	.8	−.5	.8	−.5	.8	V
V_{IH}	Input High Voltage		2.0	V_{CC} + .5V	2.3	V_{CC} + .5V	2.3	V_{CC} + .5V	V
V_{OL}	Output Low Voltage	I_{OL} = 2.2mA		.45		.45		.45	V
V_{OH}	Output High Voltage	I_{OH} = −400μA	2.4		2.4		2.4		V
$V_{OH(INT)}$	Interrupt Output High Voltage	I_{OH} = −100μA	3.5		3.5		3.5		V
		I_{OH} = −400μA	2.4		2.4		2.4		
I_{LI}	Input Load Current	0V ≤ V_{IN} ≤ V_{CC}	−10	+10	−10	+10	−10	+10	μA
I_{LOL}	Output Leakage Current	0.45V ≤ V_{OUT} ≤ V_{CC}	−10	+10	−10	+10	−10	+10	μA
I_{LOH}	Output Leakage Current	V_{OUT} = V_{CC}				10		10	μA
I_{CC}	V_{CC} Supply Current			85		125		125	mA
I_{LIR}	IR Input Load Current	V_{IN} = 0		−300					μA
		V_{IN} = V_{CC}		10					

CAPACITANCE (T_A = 25°C, V_{CC} = GND = 0V)

Parameters	Description	Test Conditions	Min	Typ	Max	Units
C_{IN}	Input Capacitance	fc = 1.0MHz			10	pF
$C_{I/O}$	I/O Capacitance	Unmeasured pins returned to V_{SS}			20	pF

AC CHARACTERISTICS

Parameters	Description	Test Conditions	8259A-8 Min	8259A-8 Max	8259A Min	8259A Max	8259A-2 Min	8259A-2 Max	Units
TIMING REQUIREMENTS									
TAHRL	A_0/$\overline{CS}$ Setup to $\overline{RD}$/$\overline{INTA}$ ↓		50		0		0		ns
TRHAX	A_0/$\overline{CS}$ Hold after $\overline{RD}$/$\overline{INTA}$ ↑		5		0		0		ns
TRLRH	$\overline{RD}$ Pulse Width		420		235		160		ns
TAHWL	A_0/$\overline{CS}$ Setup to $\overline{WR}$ ↓		50		0		0		ns
TWHAX	A_0/$\overline{CS}$ Hold after $\overline{WR}$ ↑		20		0		0		ns
TWLWH	$\overline{WR}$ Pulse Width		400		290		190		ns
TDVWH	Data Setup to $\overline{WR}$ ↑		300		240		160		ns
TWHDX	Data Hold after $\overline{WR}$ ↑		40		0		0		ns
TJLJH	Interrupt Request Width (Low)	See Note 1	100		100		100		ns
TCVIAL	Cascade Setup to Second or Third $\overline{INTA}$ ↓ (Slave Only)		55		55		40		ns
TRHRL	End of $\overline{RD}$ to Next $\overline{RD}$ End of $\overline{INTA}$ to Next $\overline{INTA}$ within an $\overline{INTA}$ sequence only	Comm, Ind	160		160		160		ns
	End of $\overline{WR}$ to Next $\overline{WR}$	Military			300				ns
TWHWL	End of $\overline{WR}$ to Next $\overline{WR}$	Comm, Ind	190		190		190		ns
TWHRL	End of $\overline{WR}$ to Next Command	Military			370				ns
TIMING RESPONSES									
TRLDV	Data Valid from $\overline{RD}$/$\overline{INTA}$ ↓	C of Data Bus = 100pF		300		200		120	ns
TRHDZ	Data Float after $\overline{RD}$/$\overline{INTA}$ ↑	C of Data Bus Max test C = 100pF Min test C = 15pF	10	200	10	100	10	85	ns
TJHIH	Interrupt Output Delay			400		350		300	ns
TIALCV	Cascade Valid from First $\overline{INTA}$ ↓ (Master Only)	C_{INT} = 100pF		565		565		360	ns
TRLEL	Enable Active from $\overline{RD}$ ↓ or $\overline{INTA}$ ↓	$C_{Cascade}$ = 100pF		160		125		100	ns
TRHEH	Enable Inactive from $\overline{RD}$ ↑ or $\overline{INTA}$ ↑			325		150		150	ns
TAHDV	Data Valid from Stable Address			350		200		200	ns
TCVDV	Cascade Valid to Valid Data			300		300		200	ns

Note: 1. This is the low time required to clear the input latch in the edge triggered mode.

INDUSTRIAL AND COMMERCIAL ONLY

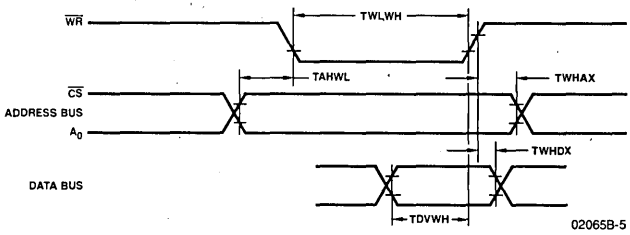
Parameters	Description	Test Conditions	8259A-8 Min	8259A-8 Max	8259A Min	8259A Max	8259A-2 Min	8259A-2 Max	Units
TCHCL*	End of Command to Next Command (Not same command type) End of $\overline{INTA}$ Sequence to Next $\overline{INTA}$ Sequence		500		500		500		ns

*Worst case timing for TCHCL in an actual microprocessor system is typically much greater than 500ns (i.e., 8085a = 1.6μs, 8085A-2 = 1μs, 8086 = 1μs, 8086-2 = 625ns).

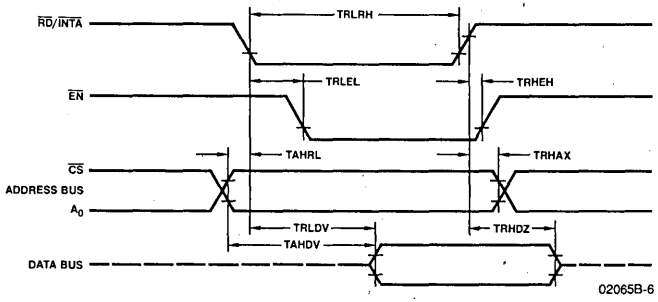
Note: This is the low time required to clear the input latch in the edge triggered mode.

WAVEFORMS

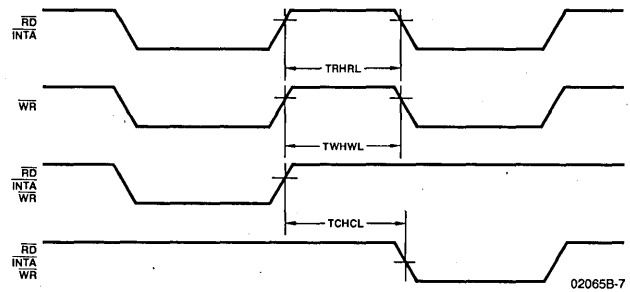
WRITE



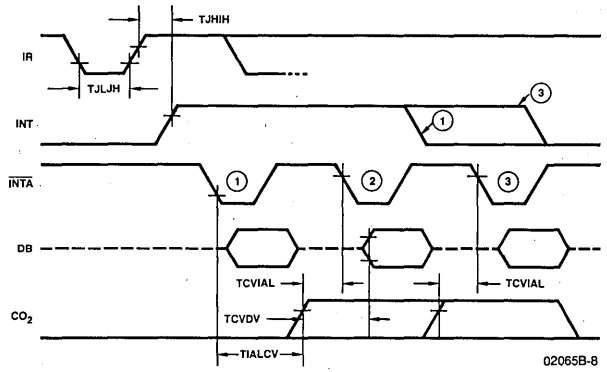
READ/INTA



OTHER TIMING



INTA SEQUENCE



Notes: 1. Interrupt output must remain HIGH at least until leading edge of first $\overline{\text{INTA}}$.
2. Cycle 1 in 8086/88 systems, the Data Bus is not active.

8284A

Clock Generator and Driver for 8086, 8088 Processors

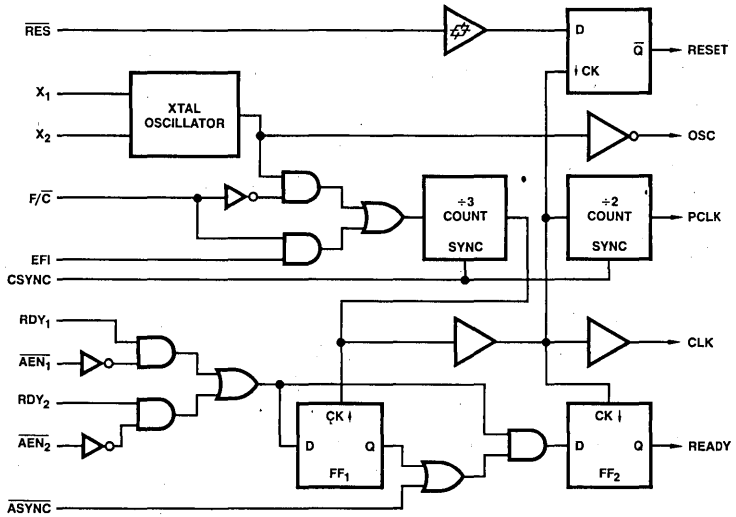
DISTINCTIVE CHARACTERISTICS

- Generates the System Clock for the 8086, 8088 Processors: 5MHz, 8MHz with 8284A
- Uses a crystal or a TTL signal for frequency source
- Provides local READY and Multibus* READY synchronization
- Generates system reset output from Schmitt trigger input
- Capable of clock synchronization with other 8284As

GENERAL DESCRIPTION

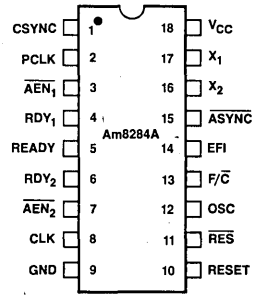
The 8284A is a single chip clock generator/driver for the 8086, 8088 processors. The chip contains a crystal-controlled oscillator, a divide-by-three counter, complete MULTIBUS* "Ready" synchronization and reset logic.

LOGIC DIAGRAM



ABI-070

CONNECTION DIAGRAM
Top View



ABI-071

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

8284 Order Number	Package Type	Temperature Range
D8284	Hermetic DIP	0 to +70°C
8284XC	Dice	0 to +70°C
MD8284	Hermetic DIP	−55 to +125°C
8284XM	Dice	−55 to +125°C

*Multibus is a registered trademark of Intel Corp.

DEFINITION OF FUNCTIONAL TERMS

$\overline{AEN_1}$, $\overline{AEN_2}$ **ADDRESS ENABLE (Input)**
The $\overline{AEN}$ signal is used to qualify the Bus Ready signal (RDY_1 or RDY_2). $\overline{AEN_1}$ validates RDY_1 while $\overline{AEN_2}$ validates RDY_2. It is possible for the processor to access two Multi-Master System Busses if you use both signals. Both signals are tied LOW in non Multi-Master Systems.

RDY_1, RDY_2 **BUS READY (Input)**
These signals are indications from a device located on the system bus that it is available or data has been received. RDY_1 and RDY_2 are qualified by $\overline{AEN_1}$ and $\overline{AEN_2}$ respectively.

$\overline{ASYNC}$ **READY SYNCHRONOUS SELECT (Input)**
The $\overline{ASYNC}$ signal defines the synchronization mode of the READY logic. When $\overline{ASYNC}$ is open (internal pull-up resistor is provided) or pulled HIGH there is one stage of READY Synchronization. When $\overline{ASYNC}$ is LOW there are two stages of READY Synchronization.

READY **READY (Input)**
READY is the synchronized RDY signal input. After the guaranteed hold time to the processor has been met, the READY signal is cleared.

X_1, X_2 **CRYSTAL IN (Inputs)**
These are the input pins for the attached crystal. The crystal frequency is 3 times the desired process clock frequency.

F/$\overline{C}$ **FREQUENCY/$\overline{CRYSTAL\ SELECT}$ (Input)**
When F/$\overline{C}$ is strapped HIGH, CLK is generated from the EFI input. When strapped LOW, the F/$\overline{C}$ allows the processor clock to be generated by the crystal.

EFI **EXTERNAL FREQUENCY (Input)**
Used in conjunction with a HIGH signal on F/$\overline{C}$, CLK is generated from the input frequency appearing on this pin. The input signal is a square wave 3 times the frequency of the desired CLK output.

CLK **PROCESSOR CLOCK (Output)**
CLK is the clock output used by the processor and all devices which directly connect to the processor's local bus (including bipolar support chips and other MOS devices). An output HIGH of 4.5V (V_{CC} = 5V) is provided on this pin to drive MOS devices. The output frequency of CLK is 1/3 of the crystal on EFI input frequency and a 1/3 duty cycle.

PCLK **PERIPHERAL CLOCK (Output)**
This signal is a TTL level peripheral clock signal whose output frequency is 1/2 that of CLK and has a 50% duty cycle.

OSC **OSCILLATOR OUTPUT (Output)**
This signal is the TTL level output of the internal oscillator circuitry. Its frequency is equal to that of the crystal.

$\overline{RES}$ **$\overline{RESET}$ IN (Input)**
This signal is used to generate a RESET. The 8284A provides a Schmitt trigger input so that an RC connection can be used to establish the power-up reset of proper duration.

RESET **RESET (Output)**
This signal is used to reset the 8086 family processors.

CSYNC **CLOCK SYNCHRONIZATION (Input)**
This signal is designed to allow multiple 8284As to be synchronized to provide clocks that are in phase. CSYNC HIGH will reset the internal counters, when CSYNC goes LOW the counters will resume counting. CSYNC needs to be externally synchronized to EFI. When used with the internal oscillator, CSYNC should be hard wired to ground.

FUNCTIONAL DESCRIPTION

OSCILLATOR

The oscillator circuit of the 8284A is designed primarily for use with a fundamental mode, series resonant crystal from which the operating frequency is derived.

The crystal frequency should be selected at three times the required CPU clock. X_1 and X_2 are the two crystal input crystal connections. The output of the oscillator is buffered and brought out on OSC so that other system timing signals can be derived from this stable, crystal-controlled source.

Two 510Ω series resistors are optional for systems which have a V_{CC} ramp time greater than (or equal to) 1V/ms and/or inherent board capacitance between X_1 or X_2 exceeding 10pF. This capacitance value should not include the 8284A's pin capacitance. By limiting the stray capacitance to less than 10pF on X_1 or X_2 the deviation from the desired fundamental frequency is minimized.

CLOCK GENERATOR

The clock generator consists of a synchronous divide-by-three counter with a special clear input that inhibits the counting. This clear input (CSYNC) allows the output clock to be synchronized with an external event (such as another 8284A clock). It is necessary to synchronize the CSYNC input to the EFI clock external to the 8284A (see Figure 1). This is accomplished with two Schottky flip-flops. The counter output is a 33% duty cycle clock at one-third the input frequency.

The F/$\overline{C}$ input is a strapping pin that selects either the EFI input or the crystal oscillator as the clock for the ÷ 3 counter. If the EFI input is selected as the clock source, the oscillator section can be used independently for another clock source. Output is taken from OSC.

CLOCK OUTPUTS

The CLK output is a 33% duty cycle MOS clock driver designed to drive the 8086 or 8088 processors directly. PCLK is a TTL level peripheral clock signal whose output frequency is 1/2 that of CLK. PCLK has a 50% duty cycle.

RESET LOGIC

Reset logic for the 8284A is provided by a Schmitt trigger input ($\overline{RES}$) and a synchronizing flip-flop to generate the reset timing. The reset signal is synchronized to the falling edge of CLK. A simple RC network can be used to provide power-on reset by utilizing this function of the 8284A.

READY SYNCHRONIZATION

Two READY inputs (RDY_1, RDY_2) are provided to accommodate two Multi-Master system busses. Each input has a qualifier ($\overline{AEN}_1$ and $\overline{AEN}_2$, respectively). The $\overline{AEN}$ signals validate their respective RDY signals. If a Multi-Master system is not being used the $\overline{AEN}$ pin should be tied LOW.

To assure RDY setup and hold times are met, synchronization is required for all asynchronous active going edges of either RDY input. Inactive-going edges of RDY in normally ready systems do not require synchronization but must satisfy RDY setup and hold as a matter of proper system design.

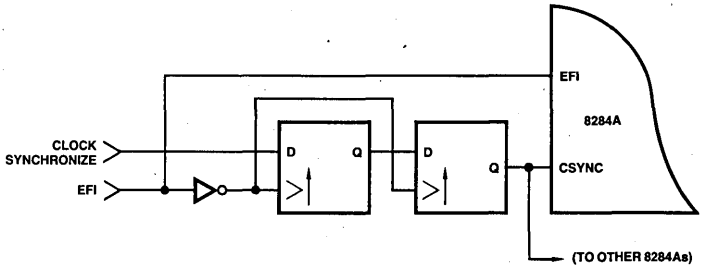
The two modes of READY synchronization operation are defined by the $\overline{ASYNC}$ input.

When $\overline{ASYNC}$ is LOW, two stages of synchronization are provided for active READY input signals. Positive-going asynchronous READY inputs will first be synchronized to flip-flop one at the rising edge of CLK and then synchronized to flip-flop two at the next falling edge of CLK, after which time the READY output will go active (HIGH). Negative-going asynchronous READY inputs will be synchronized directly to flip-flop two at the falling edge of CLK, after which time the READY output will go inactive. This mode of operation is intended for use by asynchronous (normally not ready) devices in the system which cannot be guaranteed by design to meet the required RDY setup timing t_{R1VCL}, on each bus cycle.

When $\overline{ASYNC}$ is high or left open, the first READY flip-flop is bypassed in the READY synchronization logic. READY inputs are synchronized by flip-flop two on the falling edge of CLK before they are presented to the processor. This mode is available for synchronous devices that can be guaranteed to meet the required RDY setup time.

$\overline{ASYNC}$ can be changed on every bus cycle to select the appropriate mode of synchronization for each device in the system.

Figure 1. CSYNC Synchronization



ABI-072

MAXIMUM RATINGS (Above which the useful life may be impaired)

Temperature Under Bias	0 to 70°C
Storage Temperature	−65 to +150°C
All Output and Supply Voltages	−0.5 to +7V
All Input Voltages	−1.0 to +5.5V
Power Dissipation	1W

DC CHARACTERISTICS (T_A = 0 to 70°C, V_{CC} = 5V ± 10%)

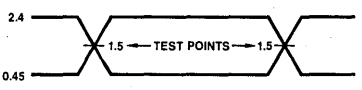
Parameters	Description	Test Conditions	Min	Max	Units
I_F	Forward Input Current ($\overline{ASYNC}$)	V_F = 0.45V		−1.3	mA
	Other Inputs	V_F = 0.45V		−0.5	
I_R	Reverse Input Current ($\overline{ASYNC}$)	$V_R = V_{CC}$		50	μA
	Other Inputs	V_R = 5.25V		50	
V_C	Input Forward Clamp Voltage	I_C = −5mA		−1.0	V
I_{CC}	Power Supply Current			162	mA
V_{IL}	Input LOW Voltage			0.8	V
V_{IH}	Input HIGH Voltage		2.0		V
V_{IHR}	Reset Input HIGH Voltage		2.6		V
V_{OL}	Output LOW Voltage	5mA		0.45	V
V_{OH}	Output HIGH Voltage CLK	−1mA	4		V
	Other Outputs	−1mA	2.4		
$V_{IHR}-V_{ILR}$	$\overline{RES}$ Input Hysteresis		0.25		V

SWITCHING CHARACTERISTICS (T_A = 0 to 70°C, V_{CC} = 5V ± 10%)

Parameter	Description	Test Conditions	Min	Max	Units
TIMING REQUIREMENTS					
t_{EHEL}	External Frequency HIGH Time	90% – 90% V_{IN}	13		ns
t_{ELEH}	External Frequency LOW Time	10% – 10% V_{IN}	13		ns
t_{ELEL}	EFI Period		33		ns
	XTAL Frequency		12	25	MHz
t_{R1VCL}	RDY_1, RDY_2 Active Setup to CLK	$\overline{ASYNC}$ = HIGH	35		ns
t_{R1VCH}	RDY_1, RDY_2 Active Setup to CLK	$\overline{ASYNC}$ = LOW	35		ns
t_{R1VCL}	RDY_1, RDY_2 Inactive Setup to CLK		35		ns
t_{CLR1X}	RDY_1, RDY_2 Hold to CLK		0		ns
t_{AYVCL}	$\overline{ASYNC}$ Setup to CLK		50		ns
t_{CLAYX}	$\overline{ASYNC}$ Hold to CLK		0		ns
t_{A1VR1V}	$\overline{AEN_1}$, $\overline{AEN_2}$ Setup to RDY_1, RDY_2		15		ns
t_{CLA1X}	$\overline{AEN_1}$, $\overline{AEN_2}$ Hold to CLK		0		ns
t_{YHEH}	CSYNC Setup to EFI		20		ns
t_{EHYL}	CSYNC Hold to EFI		10		ns
t_{YHYL}	CSYNC Width		$2 \cdot t_{ELEL}$		ns
t_{I1HCL}	$\overline{RES}$ Setup to CLK	(Note 1)	65		ns
t_{CLI1H}	$\overline{RES}$ Hold to CLK	(Note 1)	20		ns
t_{ILIH}	Input Rise Time	From 0.8 to 2.0V		20	ns
t_{ILIL}	Input Fall Time	From 2.0 to 0.8V		12	ns
TIMING RESPONSES					
t_{CLCL}	CLK Cycle Period		125		ns
t_{CHCL}	CLK HIGH Time		$(1/3\ t_{CLCL}) + 2$		ns
t_{CLCH}	CLK LOW Time		$(2/3\ t_{CLCL}) - 15$		ns
t_{CH1CH2}	CLK Rise or Fall Time	1.0V to 3.5V		10	ns
t_{CL2CL1}					
t_{PHPL}	PCLK HIGH Time		$t_{CLCL} - 20$		ns
t_{PLPH}	PCLK LOW Time		$t_{CLCL} - 20$		ns
t_{RYLCL}	Ready Inactive to CLK (See Note 3)		−8		ns
t_{RYHCH}	Ready Active to CLK (See Note 2)		$(2/3\ t_{CLCL}) - 15$		ns
t_{CLIL}	CLK to Reset Delay			40	ns
t_{CLPH}	CLK to PCLK HIGH Delay			22	ns
t_{CLPL}	CLK to PCLK LOW Delay			22	ns
t_{OLCH}	OSC to CLK HIGH Delay		−5	22	ns
t_{OLCL}	OSC to CLK LOW Delay		2	35	ns
t_{OLOH}	Output Rise Time (except CLK)	From 0.8 to 2.0V		20	ns
t_{OHOL}	Output Fall Time (except CLK)	From 2.0 to 0.8V		12	ns

Notes: 1. Setup and hold necessary only to guarantee recognition at next clock.
2. Applies only to T_3 and T_W states.
3. Applies only to T_2 states.

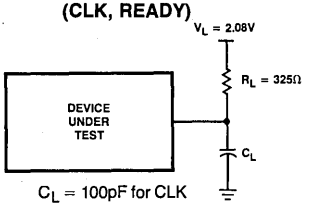
AC TESTING INPUT, OUTPUT WAVEFORM



ABI-073

AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0". Timing measurements are made at 1.5V for both a logic "1" and "0".

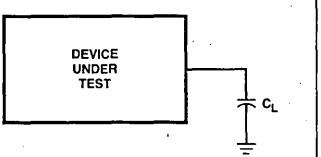
AC TESTING LOAD CIRCUIT (CLK, READY)



C_L = 100pF for CLK
C_L = 30pF for READY

ABI-074

AC TESTING LOAD CIRCUIT (PCLK, OSC, RESET)

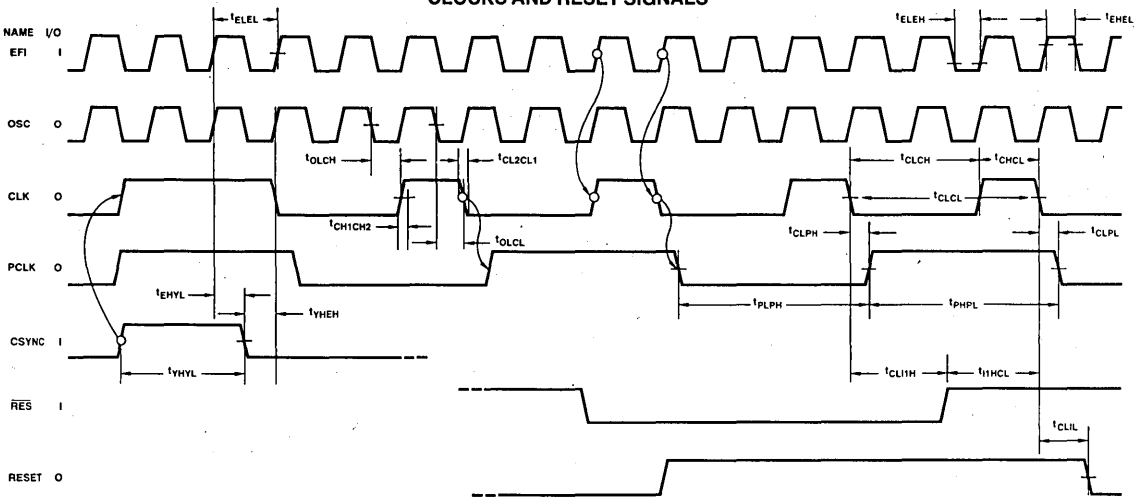


C_L = 100pF

ABI-075

WAVEFORMS

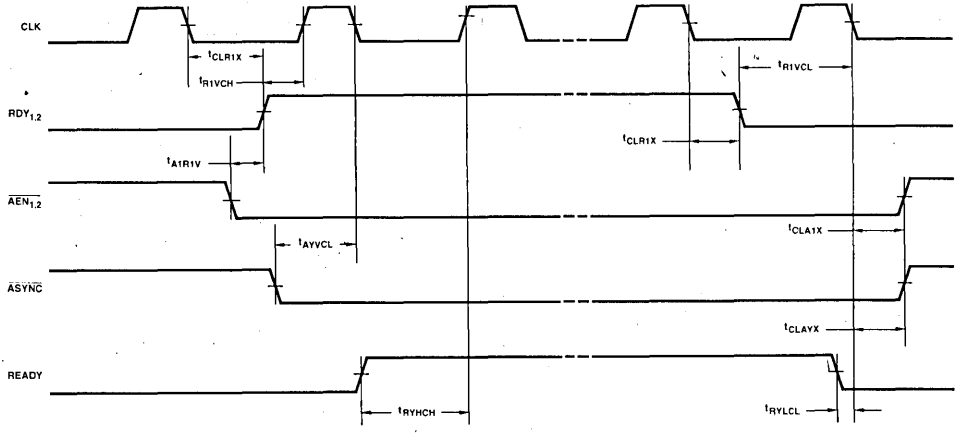
CLOCKS AND RESET SIGNALS



Note: All timing requirements are made at 1.5 volts, unless otherwise noted.

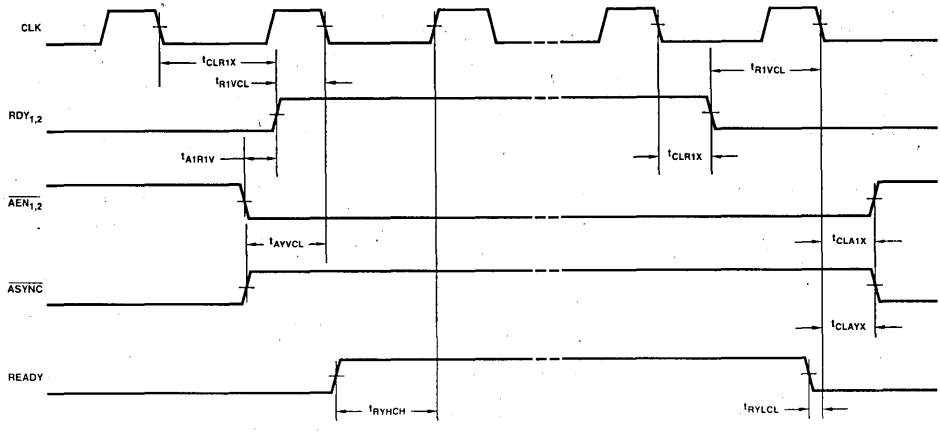
ABI-076

READY SIGNALS (FOR ASYNCHRONOUS DEVICES)



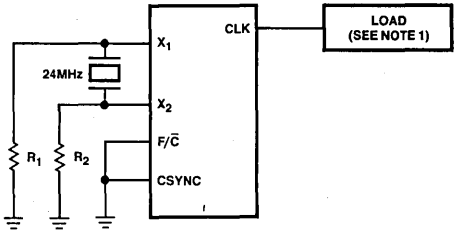
ABI-077

READY SIGNALS (FOR SYNCHRONOUS DEVICES)



ABI-078

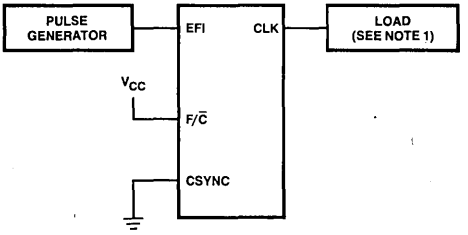
CLOCK HIGH AND LOW TIME (USING X_1, X_2)



$R_1 = R_2 = 510\Omega$.

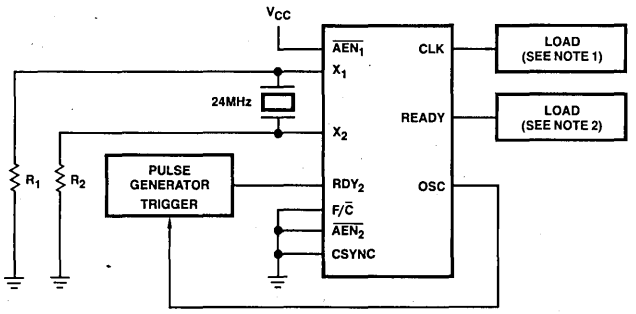
ABI-079

CLOCK HIGH AND LOW TIME (USING EFI)



ABI-080

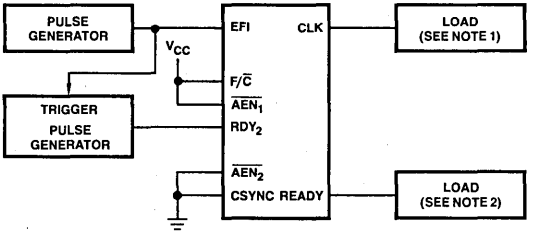
READY TO CLOCK (USING X_1, X_2)



$R_1 = R_2 = 510\Omega$.

ABI-081

READY TO CLOCK (USING EFI)



Notes: 1. $C_L = 100pF$
2. $C_L = 30pF$

ABI-082

8286/8287
Octal Bus Transceivers

DISTINCTIVE CHARACTERISTICS

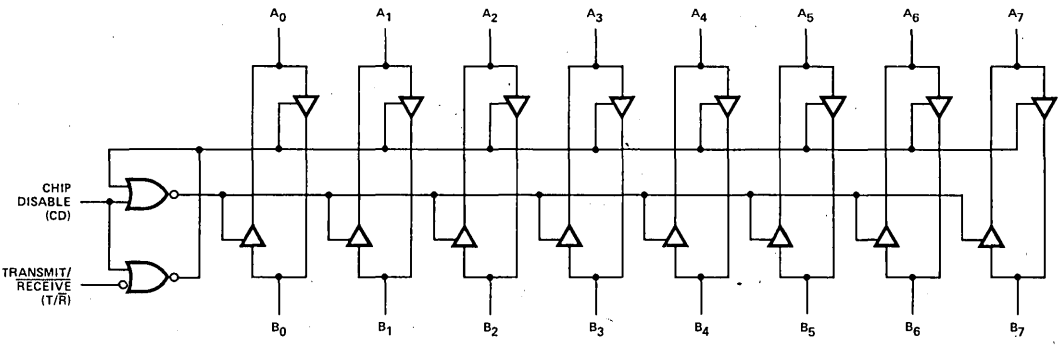
- Data bus buffer/driver for 8086, 8088, 8080A, 8085A, and 8048 processors
- Fully parallel 8-bit transceivers: 8286 is noninverting
  8287 is inverting
- 3-state inputs/outputs for interfacing with bus-oriented systems
- Available in 20-pin, 0.3" center molded DIP or ceramic package
- Advanced bipolar Schottky processing
- Bus port stays in hi-impedance state during power up/down transition

FUNCTIONAL DESCRIPTION

The 8286 and 8287 are 8-bit 3-state bipolar Schottky transceivers. They provide bidirectional drive for bus-oriented microprocessor and digital communications systems. Straight through bidirectional transceivers are featured, with 16mA drive capability on the A ports and 32mA bus drive capability on the B ports. PNP inputs are incorporated to reduce input loading.

One input, Transmit/$\overline{\text{Receive}}$ determines the direction of logic signals through the bidirectional transceiver. The Chip Disable input disables both A and B ports by placing them in a 3-state condition. Chip Disable is functionally the same as an active LOW chip select.

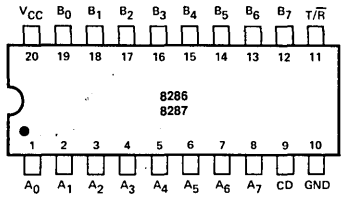
8286
LOGIC DIAGRAM



8287 has inverting transceivers

ABI-001

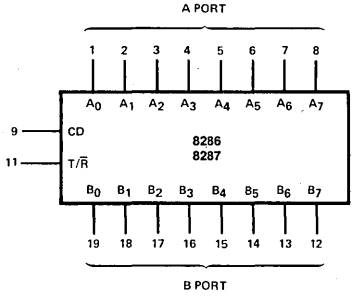
CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

ABI-002

LOGIC SYMBOL



V_{CC} = Pin 20
GND = Pin 10

ABI-003

ABSOLUTE MAXIMUM RATINGS (Above which the useful life may be impaired)

Storage Temperature	−65 to +150°C
Supply Voltage	7.0V
Input Voltage	5.5V
Output Voltage	5.5V
Lead Temperature (Soldering, 10 seconds)	300°C

ELECTRICAL CHARACTERISTICS

The Following Conditions Apply Unless Otherwise Noted:

MIL T_A = −55 to +125°C V_{CC} MIN = 4.5V V_{CC} MAX = 5.5V
COM'L T_A = 0 to +70°C V_{CC} MIN = 4.5V V_{CC} MAX = 5.5V

DC ELECTRICAL CHARACTERISTICS over operating temperature range

Parameters	Description		Test Conditions	Min	Max	Units
V_C	Input Clamp Voltage		I_C = −5mA		−1	V
I_{CC}	Power Supply Current	8287			130	mA
		8286			160	
I_F	Forward Input Current		V_F = 0.45V		−0.2	mA
I_R	Reverse Input Current		V_R = 5.25V		50	μA
V_{OL} (COM'L)	Output Low Voltage	B Outputs	I_{OL} = 32mA		.45	V
		A Outputs	I_{OL} = 16mA		.45	
V_{OL} (MIL)	Output Low Voltage	B Outputs	I_{OL} = 20mA		.45	V
		A Outputs	I_{OL} = 10mA		.45	
V_{OH}	Output High Voltage	B Outputs	I_{OH} = −5mA	2.4		V
		A Outputs	I_{OH} = −1mA	2.4		
I_{OFF}	Output Off Current		V_{OFF} = 0.45V		I_F	
			V_{OFF} = 5.25V		I_R	
V_{IL}	Input Low Voltage	A Port	V_{CC} = 5.0V (See note 1)		0.8	V
		B Port	V_{CC} = 5.0V (See note 1)		0.9	
V_{IH}	Input High Voltage		V_{CC} = 5.0V (See note 1)	2.0		V
C_{IN}	Input Capacitance		F = 1MHz V_{BIAS} = 2.5V, V_{CC} = 5V T_A = 25°C		12	pF

AC ELECTRICAL CHARACTERISTICS (See Note 2)

Parameters	Description		Test Conditions (See Notes)	MIN (COM'L)	MIN (MIL)	Max	Units
TIVOV	Input to Output Delay	Inverting		5		22	ns
		Non-inverting		5		30	
TEHTV	Transmit/Receive Hold Time			5	TENOZ		ns
TTVEL	Transmit/Receive Setup			10	30		ns
TEHOZ	Output Disable Time			3		18	ns
TELOV	Output Enable Time			10	10	30	ns
TILIH, TOLOH	Input, Output Rise Time		From 0.8 to 2.0V			20	ns
TIHIL, TOHOL	Input, Output Fall Time		From 2.0 to 0.8V			12	ns

Notes: 1. COM'L temperature loading conditions
B outputs: I_{OL} = 32mA, I_{OH} = −5mA, C_L = 300pF
A outputs: I_{OL} = 16mA, I_{OH} = −1mA, C_L = 100pF

MIL temperature loading conditions
B outputs: I_{OL} = 20mA, I_{OH} = −5mA, C_L = 300pF
A outputs: I_{OL} = 10mA, I_{OH} = −1mA, C_L = 100pF

2. Refer to waveforms and test load circuits on following pages.

DEFINITION OF FUNCTIONAL TERMS

A_0-A_7 A port inputs/outputs are receiver output drivers when T/$\overline{R}$ is LOW and are transmit inputs when T/$\overline{R}$ is HIGH.

B_0-B_7 B port inputs/outputs are transmit output drivers when T/$\overline{R}$ is HIGH and receiver inputs when T/$\overline{R}$ is LOW.

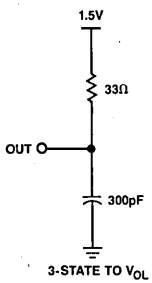
CD Chip Disable forces all output drivers into 3-state when HIGH (same function as active LOW chip select, $\overline{CS}$).

T/$\overline{R}$ Transmit/$\overline{Receive}$ direction control determines whether A port or B port drivers are in 3-state. With T/$\overline{R}$ HIGH A port is the input and B port is the output. With T/$\overline{R}$ LOW A port is the output and B port is the input.

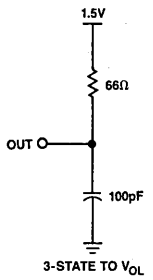
FUNCTION TABLE

Inputs	Conditions		
Chip Disable	0	0	1
Transmit/$\overline{Receive}$	0	1	X
A Port	Out	In	HI-Z
B Port	In	Out	HI-Z

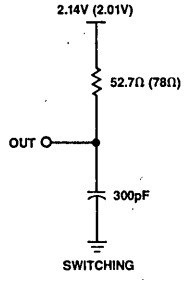
TEST LOAD CIRCUITS



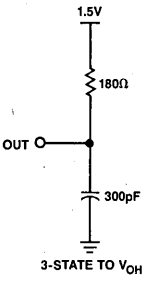
B OUTPUT



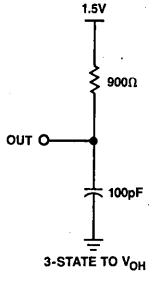
A OUTPUT



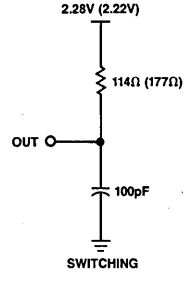
B OUTPUT



B OUTPUT



A OUTPUT

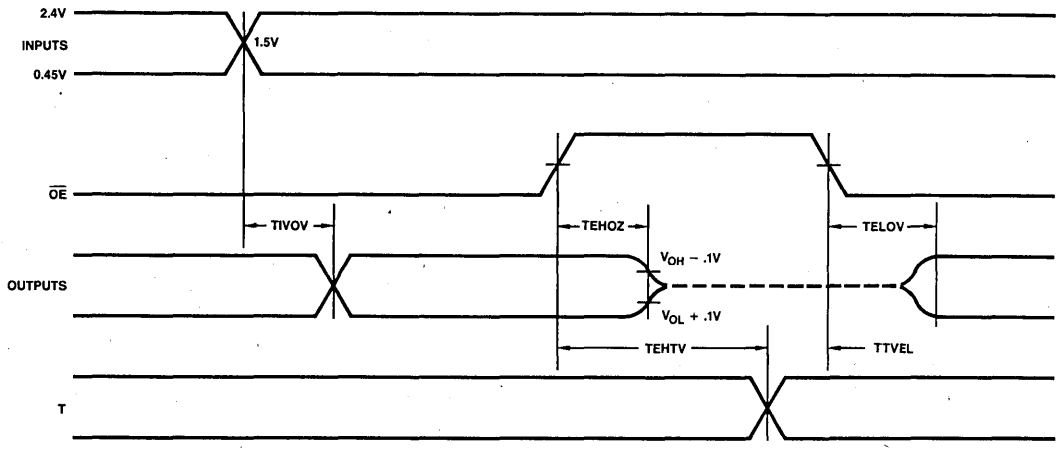


A OUTPUT

Values in parenthesis reflect MIL temp. conditions.

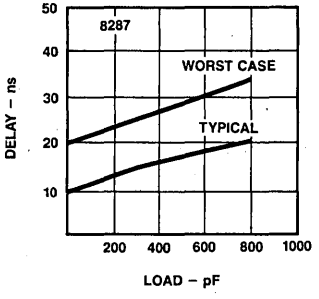
ABI-004

WAVEFORMS

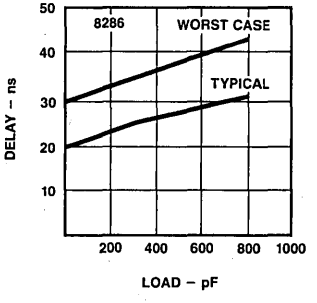


AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0" timing measurements are made at 1.5V for both a logic "1" and "0."

ABI-005



ABI-006

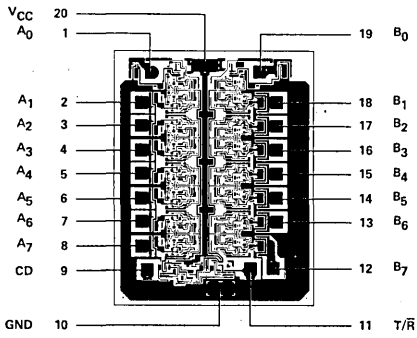


ABI-007

Output Delay versus Capacitance

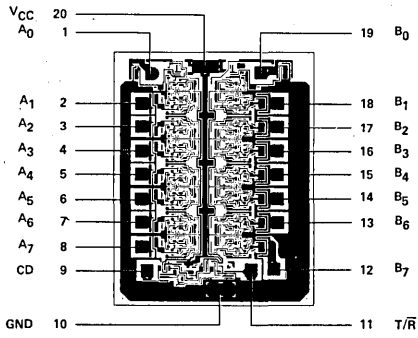
Metallization and Pad Layouts

8286



DIE SIZE .069" X .089"

8287



DIE SIZE .069" X .089"

ORDERING INFORMATION

Order the part number according to the table below to obtain the desired package, temperature range, and screening level.

8286 Order Number	8287 Order Number	Package Type (Note 1)	Operating (Note 2)	Screening Level (Note 3)
MD8286		D-20	M	C-3
MD8286B		D-20	M	B-3
D8286		D-20	C	C-1
D8286B		D-20	C	B-1
P8286	P8287	P-20	C	C-1
P8286B	P8287B	P-20	C	B-1
8286XM	8287XM	Dice	M	Visual inspection to MIL-STD-883 Method 2010B.
8286XC	8287XC	Dice	C	

Notes: 1. P = Molded DIP, D = Hermetic DIP. Number following letter is number of leads.
2. C = 0 to 70°C, V_{CC} = 4.50 to 5.50V; M = −55 to +125°C, V_{CC} = 4.50 to 5.50V.
3. Levels C-1 and C-3 conform to MIL-STD-883, Class C. Level B-3 conforms to MIL-STD-883, Class B.

8288
Bus Controller

DISTINCTIVE CHARACTERISTICS

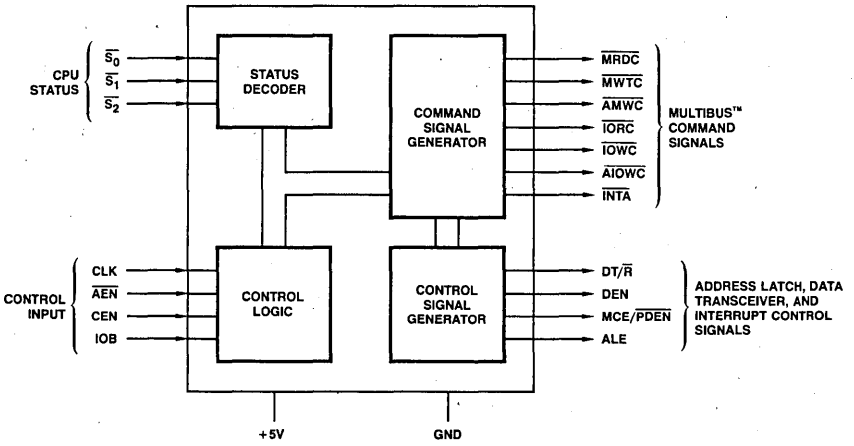
- Bipolar drive capability
- 3-state output drivers
- Multi-master or I/O bus interface
- Flexible system configurations

GENERAL DESCRIPTION

The 8288 optimizes 8086 or 8088 operations by providing command and control timing generation when the CPU is in maximum mode. It provides for highly flexible configurations for larger systems. It also adds powerful bipolar drive capability to the system.

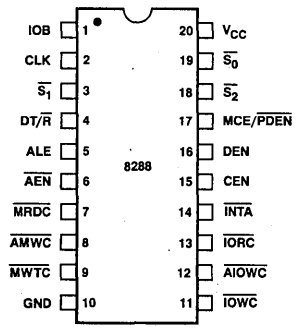
The 8288 is implemented in bipolar technology in a 20-pin DIP.

BLOCK DIAGRAM



ABI-083

CONNECTION DIAGRAM
Top View



ABI-084

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

8288 Order Number	Package Type	Temperature Range
D8288	Hermetic DIP	0 to +70°C
8288XC	Dice	0 to +70°C

DEFINITION OF FUNCTIONAL TERMS

$\overline{S_0}$, $\overline{S_1}$, $\overline{S_2}$ **STATUS (Input)**
These signals are the status input pins from the microprocessor. The 8288 decodes these inputs to generate command and control signals.

CLK **CLOCK (Input)**
Clock signal from the clock generator.

ALE **ADDRESS LATCH ENABLE (Output)**
This signal strobes an address into the address latches. The latching occurs on the falling edge (HIGH to LOW) transition.

DEN **DATA ENABLE (Output)**
This signal enables the data transceivers onto the data bus (local or system).

DT/$\overline{R}$ **DATA TRANSMIT/$\overline{\text{RECEIVE}}$ (Output)**
This signal determines the direction of data flow through the transceivers.

$\overline{AEN}$ **$\overline{\text{ADDRESS ENABLE}}$ (Input)**
This signal enables the 8288 command outputs at least 115ns after it becomes active LOW. When this pin goes inactive, it 3-states the command output drivers.

CEN **COMMAND ENABLE (Input)**
This signal, when LOW, enables all command outputs and the DEN and $\overline{PDEN}$ control outputs are forced to their inactive states.

IOB **INPUT/OUTPUT BUS MODE (Input)**
When strapped HIGH the 8288 functions in the I/O Bus mode. When LOW the 8288 functions in the System Bus mode.

$\overline{AIOWC}$ **$\overline{\text{ADVANCED}}$ I/O WRITE COMMAND (Output)**
The $\overline{AIOWC}$ gives I/O devices early indication of a write instruction by issuing an I/O Write Command earlier in the machine cycle.

$\overline{IOWC}$ **$\overline{\text{I/O WRITE}}$ (Output)**
This signal tells an I/O device to read the data on the data bus.

$\overline{IORC}$ **$\overline{\text{I/O READ}}$ (Output)**
This signal tells an I/O device to drive its data onto the data bus.

$\overline{AMWC}$ **$\overline{\text{ADVANCED MEMORY WRITE}}$ (Output)**
The $\overline{AMWC}$ gives memory devices an early indication of a write instruction by issuing a memory write command earlier in the machine cycle.

$\overline{MWTC}$ **$\overline{\text{MEMORY WRITE}}$ (Output)**
This signal instructs the memory to record the data present on the data bus.

$\overline{MRDC}$ **$\overline{\text{MEMORY READ}}$ (Output)**
This signal instructs the memory to drive its data onto the data bus.

$\overline{INTA}$ **$\overline{\text{INTERRUPT ACKNOWLEDGE}}$ (Output)**
This signal informs the interrupting device that its interrupt has been acknowledged and to drive vectoring information onto the data bus.

MCE/$\overline{PDEN}$ **MASTER CASCADE ENABLE/ $\overline{\text{PERIPHERAL DATA ENABLE}}$ (Output)**
Dual Function pin:
MCE (IOB LOW): This signal occurs during an interrupt sequence. Its function is to read a Cascade Address from a master Priority Interrupt Controller onto the data bus.
$\overline{PDEN}$ (IOB HIGH): This signal enables the data bus transceiver for the I/O Bus during I/O instructions. It performs the same function for the I/O Bus that DEN performs for the system bus.

FUNCTIONAL DESCRIPTION

COMMAND AND CONTROL LOGIC

The command logic decodes the three CPU status lines ($\overline{S_0}$, $\overline{S_1}$, $\overline{S_2}$) to determine what command is to be issued.

This chart shows the meaning of each status "word."

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	Processor State	8288 Command
0	0	0	Interrupt Acknowledge	$\overline{INTA}$
0	0	1	Read I/O Port	$\overline{IORC}$
0	1	0	Write I/O Port	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	Halt	None
1	0	0	Code Access	$\overline{MRDC}$
1	0	1	Read Memory	$\overline{MRDC}$
1	1	0	Write Memory	$\overline{MWTC}$, $\overline{AMWC}$
1	1	1	Passive	None

I/O BUS MODE

The 8288 is put into the I/O Bus mode by strapping the IOB pin HIGH. This mode allows one 8288 Bus Controller to handle two external buses. This allows the CPU to access the I/O Bus with no waiting involved. In the I/O Bus Mode all I/O command lines ($\overline{INTA}$, $\overline{IORC}$, $\overline{IOWC}$, $\overline{AIOWC}$) are always enabled. When the processor initiates an I/O Command, the 8288 immediately activates the command lines using $\overline{PDEN}$ and DT/$\overline{R}$ to control the I/O bus transceiver. There is no arbitration present in this system, so the I/O command lines should not be used to control the system bus. Normal memory access requires a "Bus Ready" signal ($\overline{AEN}$ LOW) before it will proceed. The IOB mode is recommended if I/O or peripherals dedicated to one processor exist in a multiprocessor based system.

SYSTEM BUS MODE

The 8288 is put into the System Bus mode by strapping the IOB pin LOW. This mode is used when only one bus exists. No command is issued until 115ns after the $\overline{AEN}$ line is activated. Bus arbitration is assumed, and this logic will inform the bus controller via the $\overline{AEN}$ line when the bus is free for use. Both I/O commands and memory wait for bus arbitration.

COMMAND OUTPUTS

To prevent the processor from entering unnecessary wait states, the advanced write commands initiate write procedures early in the machine cycle.

The command outputs are:

- $\overline{MRDC}$ – Memory Read Command
- $\overline{MWTC}$ – Memory Write Command
- $\overline{IORC}$ – I/O Read Command
- $\overline{IOWC}$ – I/O Write Command
- $\overline{AMWC}$ – Advanced Memory Write Command
- $\overline{AIOWC}$ – Advanced I/O Write Command
- $\overline{INTA}$ – Interrupt Acknowledge

$\overline{INTA}$ (Interrupt Acknowledge) acts as an I/O read during an interrupt cycle. Its purpose is to inform an interrupting device that its interrupt is being acknowledged and that it should place vectoring information onto the data bus.

CONTROL OUTPUTS

The Data Enable (DEN), Data Transmit/$\overline{Receive}$ (DT/$\overline{R}$) and Master Cascade Enable/Peripheral Data Enable (MCE/$\overline{PDEN}$) are the control outputs of the 8288. The DEN signal determines when the external bus should be enabled onto the local bus while the DT/$\overline{R}$ determines the direction of the data transfer. These two signals usually go to the chip select and direction pins of a transceiver.

The MCE/$\overline{PDEN}$ function is determined by the IOB selection. When IOB is HIGH the $\overline{PDEN}$ serves as a dedicated data enable signal for the I/O or Peripheral System Bus.

INTERRUPT ACKNOWLEDGE AND MCE

The MCE signal is used during an interrupt acknowledge cycle if the 8288 is in the System Bus mode (IOB Low). An interrupt sequence consists of two interrupt acknowledge cycles occurring back to back. No data or address transfers take place during the first cycle. Logic should be provided to mask off MCE during this cycle. Just before the second cycle begins the MCE signal gates a master Priority Interrupt Controller's (PIC) cascade address onto the processor's local bus where ALE (Address Latch Enable) strobes it into the address latches. On the leading edge of the second interrupt cycle the addressed slave PIC gates an interrupt vector onto the system data bus where it is read by the processor.

The MCE signal is not used if the system only contains one PIC. If this is the case the second Interrupt Acknowledge signal gates the interrupt vector onto the processor bus.

ADDRESS LATCH ENABLE AND HALT

Address Latch Enable (ALE) occurs during each machine cycle and serves to strobe the current address into the address latches. ALE also serves to strobe the status ($\overline{S_0}$, $\overline{S_1}$, $\overline{S_2}$) into a latch for halt state decoding.

COMMAND ENABLE

The Command Enable (CEN) input acts as a command qualifier for the 8288. If the CEN pin is HIGH the 8288 functions normally. If the CEN pin is pulled LOW, all command lines are held in their inactive state (not 3-state). This feature can be used to implement memory partitioning and to eliminate address conflicts between system bus devices and resident bus devices.

MAXIMUM RATINGS*

Temperature Under Bias	0 to 70°C
Storage Temperature	−65 to +150°C
All Output and Supply Voltages	−0.5 to +7.0V
All Input Voltages	−1.0 to +5.5V
Power Dissipation	1.5W

*Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS (V_{CC} = 5V ± 10%, T_A = 0 to 70°C)

Parameter	Description	Test Conditions	Min	Max	Units
V_C	Input Clamp Voltage	I_C = −5mA		−1	V
I_{CC}	Power Supply Current			230	mA
I_F	Forward Input Current	V_F = 0.45V		−0.7	mA
I_R	Reverse Input Current	V_R = V_{CC}		50	μA
V_{OL}	Output Low Voltage Command Outputs	I_{OL} = 32mA		0.5	V
	Control Outputs	I_{OL} = 16mA		0.5	V
V_{OH}	Output High Voltage Command Outputs	I_{OH} = −5mA	2.4		V
	Control Outputs	I_{OH} = −1mA	2.4		V
V_{IL}	Input Low Voltage			0.8	V
V_{IH}	Input High Voltage		2.0		V
I_{OFF}	Three-State Leakage	V_{OFF} = 0.4 to 5.25V		100	μA

AC CHARACTERISTICS (V_{CC} = 5V ± 10%, T_A = 0 to 70°C)

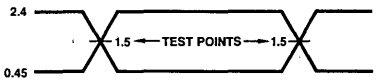
Parameters	Description	Test Conditions	Min	Max	Units
TIMING REQUIREMENTS					
TCLCL	CLK Cycle Period		100		ns
TCLCH	CLK Low Time		50		ns
TCHCL	CLK High Time		30		ns
TSVCH	Status Active Setup Time		35		ns
TCHSV	Status Active Hold Time		10		ns
TSHCL	Status Inactive Setup Time		35		ns
TCLSH	Status Inactive Hold Time		10		ns
TILIH	Input Rise Time	From 0.8V to 2.0V		20	ns
TIHIL	Input Fall Time	From 2.0V to 0.8V		12	ns
TIMING RESPONSES					
TCVNV	Control Active Delay		5.0	45	ns
TCVNX	Control Inactive Delay		10	45	ns
TCLLH, TCLMCH	ALE MCE Active Delay (from CLK)			20	ns
TSVLH, TSVMCH	ALE MCE Active Delay (from Status)			20	ns
TCHLL	ALE Inactive Delay	$\overline{MRDC}$	4.0	15	ns
TCLML	Command Active Delay	$\overline{IORC}$	10	35	ns
TCLMH	Command Inactive Delay	$\overline{MWTC}$	10	35	ns
TCHDTL	Direction Control Active Delay	$\overline{IOWC}$		50	ns
TCHDTH	Direction Control Inactive Delay	$\overline{INTA}$		30	ns
TAELCH	Command Enable Time	$\overline{AMWC}$		40	ns
TAEHCZ	Command Disable Time	$\overline{AIOWC}$		40	ns
TAELCV	Enable Delay Time		115	200	ns
TAEVNV	AEN to DEN	Other		20	ns
TCEVNV	CEN to DEN, PDEN			25	ns
TCELRH	CEN to Command			TCLML	ns
TOLOH	Output Rise Time	From 0.8V to 2.0V		20	ns
TOHOL	Output Fall Time	From 2.0V to 0.8V		12	ns

(I_{OL} = 32mA, I_{OH} = −5mA, C_L = 300pF for $\overline{MRDC}$, $\overline{IORC}$, $\overline{MWTC}$, $\overline{IOWC}$, $\overline{INTA}$, $\overline{AMWC}$, $\overline{AIOWC}$)

(I_{OL} = 16mA, I_{OH} = −1.0mA, C_L = 80pF for Other)

AC TESTING INPUT, OUTPUT WAVEFORM

Input/Output

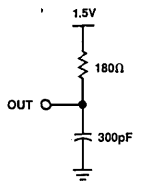


ABI-085

AC Testing: Inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0." The clock is driven at 4.3V and 0.25V. Timing measurements are made at 1.5V for both a logic "1" and "0."

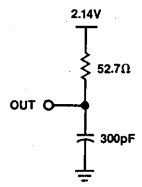
TEST LOAD CIRCUITS

3-State to High



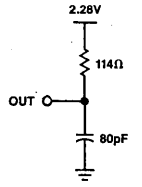
ABI-086

Command Output Test Load



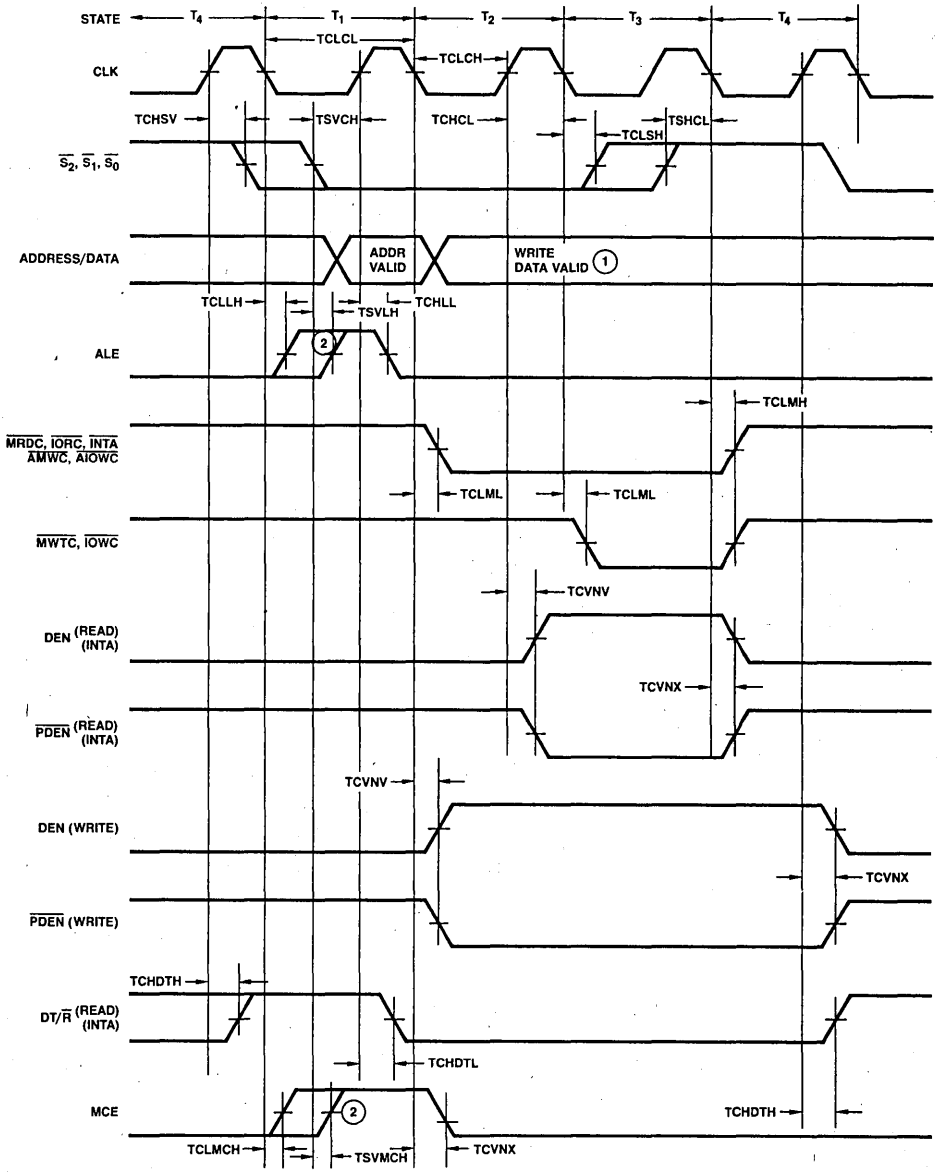
ABI-087

Control Output Test Load



ABI-088

WAVEFORMS

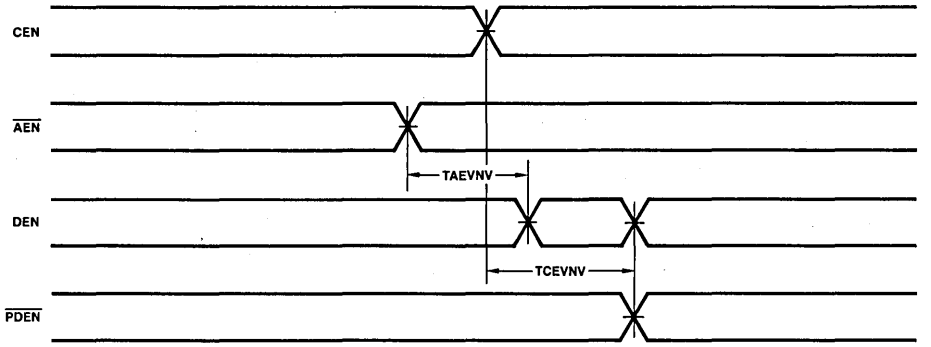


AB1-089

Notes: 1. Address/data bus is shown only for reference purposes.
2. Leading edge of ALE and MCE is determined by the falling edge of CLK or status going active, whichever occurs last.
3. All timing measurements are made at 1.5V unless specified otherwise.

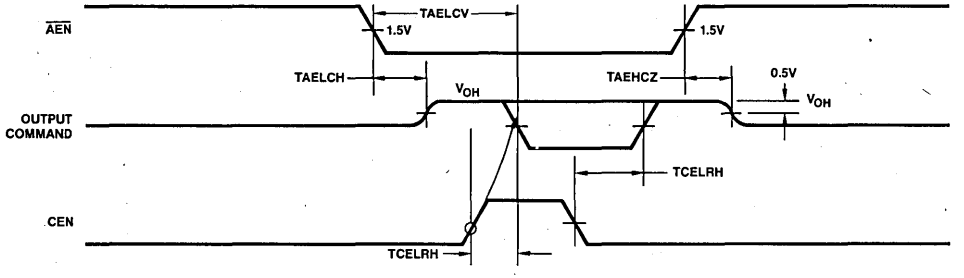
WAVEFORMS (Cont.)

DEN, PDEN QUALIFICATION TIMING



ABI-090

ADDRESS ENABLE (AEN) TIMING (3-STATE ENABLE/DISABLE)



Note: CEN must be low or valid prior to T_2 to prevent the command from being generated.

ABI-091

Z8000 Family

AmZ8001 • AmZ8002

16-Bit Microprocessors

DISTINCTIVE CHARACTERISTICS

- **4 MHz CPU Clock**
  High throughput with low system clock rate for easier system design
- **Powerful General Register Architecture**
  16 general registers provide high throughput in all types of applications
- **Wide Variety of Data Types**
  Instructions operate on bits, bytes, 16 and 32-bit words for efficient programming of a wide variety of functions
- **Partitioned for Operating System Protection**
  Hardware bit protects privileged instructions from execution except by operating system
- **Supports 3 Types of Interrupts**
  Separate pins provided for vectored, non-vectored and non-maskable interrupts
- **Two Compatible CPUs**
  Compact 40-pin 8002 supports 64kB memory
  larger 48-pin 8001 supports 8MB memory

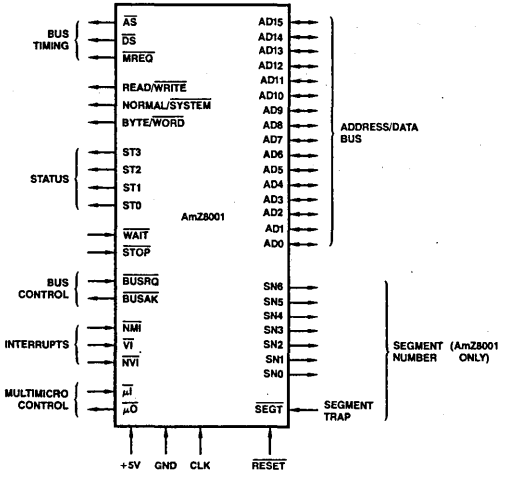
OTHER LITERATURE

- AmZ8001/2 Users Manual
- AmZ8001/2 Processor Interface Manual

GENERAL DESCRIPTION

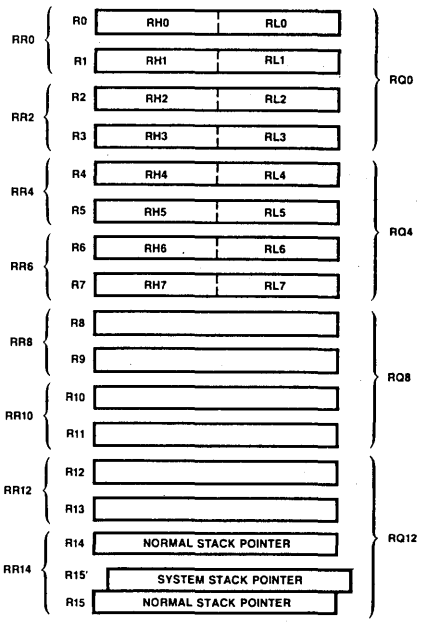
The AmZ8001* is a general-purpose 16-bit CPU belonging to the AmZ8000 family of microprocessors. Its architecture is centered around sixteen 16-bit general registers. The CPU deals with 23-bit address spaces and hence can address directly 8MB of memory. The 23-bit address consists of two components: 7-bit segment number and 16-bit offset. Facilities are provided to maintain three distinct address spaces – code, data and stack. The AmZ8001 implements a powerful instruction set with flexible addressing modes. These instructions operate on several data types – bit, byte, word (16-bit), long word (32-bit), byte string and word string. The CPU can execute instructions in one of two modes – System and Normal. Sometimes these modes are also known as Privileged and Non-Privileged, respectively. The CPU also contains an on-chip memory refresh facility. The AmZ8001 is software compatible with the AmZ8002 microprocessor. The AmZ8001 is fabricated using silicon-gate N-MOS technology and is packaged in a 48-pin DIP. The AmZ8001 requires a single +5 power supply and a single phase clock for its operation.

LOGIC SYMBOL



00971B-1

GENERAL REGISTERS



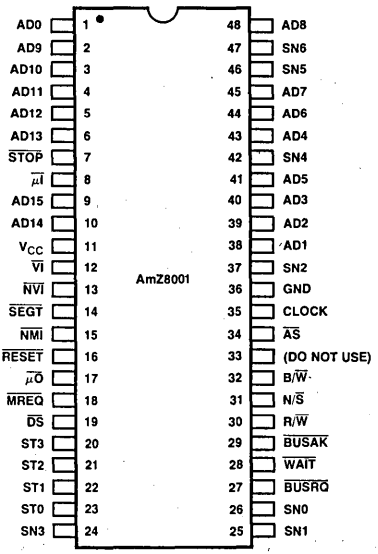
*AmZ8001 only

00971B-2

*Z8000 is a trademark of Zilog, Inc.

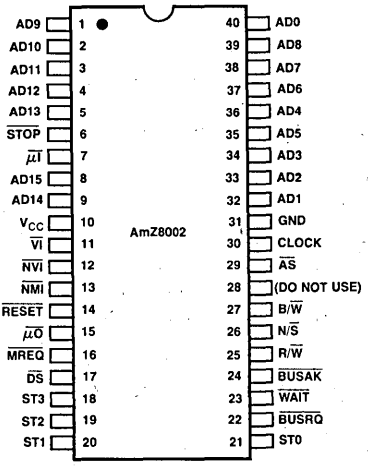
CONNECTION DIAGRAMS – Top Views

D-48-1, P-48-1



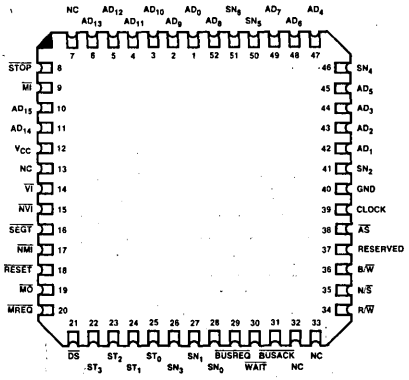
00971B-3

D-40-1, P-40-1



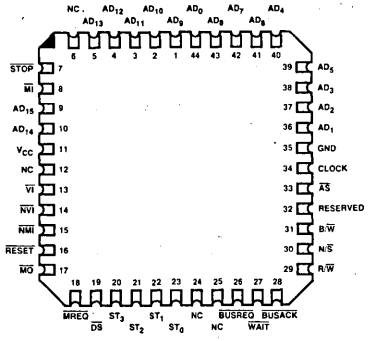
00971B-4

L-52-1



00971B-21

L-44-1



00971B-22

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Example Z800X A D C B

- Basic Device
- Speed
  - Blank = 4MHz
  - A = 6MHz
  - A-8 = 8MHz
- Package Type
  - D = Cerdip
  - P = Plastic
  - L = Leadless Chip Carrier
- Temperature Range
  - C = 0 to +70°C
  - I = −40 to +85°C
  - M = −55 to +125°C
- Burn-in

PROCESSOR ORGANIZATION

The following is a brief discussion of the AmZ8001 and AmZ8002 CPUs. For detailed information, see the AmZ8001/ AmZ8002 User's Manual.

GENERAL PURPOSE REGISTERS

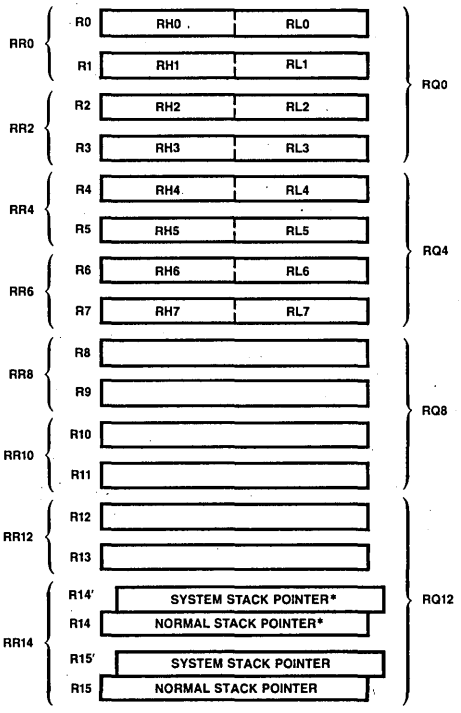
The CPU is organized around sixteen 16-bit general purpose registers R0 through R15 as shown in Figure 1. For byte operations, the first eight registers (R0 through R7) can also be addressed as sixteen 8-bit registers designated as RL0, RH0 and so on to RL7 and RH7. The sixteen registers can also be grouped in pairs RR0, RR2 and so on to RR14 to form eight long word (32-bit) registers. Similarly, the sixteen registers can be grouped in quadruples RQ0, RQ4, RQ8 and RQ12 to form four 64-bit registers.

STACK POINTER (AmZ8001)

The AmZ8001 architecture allows stacks to be maintained in memory. Any general-purpose register pair except RR0 can be used as a stack pointer in stack manipulating instructions such as PUSH and POP. The designated register pair holds a 23-bit segmented address. Certain instructions (such as subroutine call and return) make implicit use of the register pair RR14 as the stack pointer. Two implicit stacks are allowed – normal stack using RR14 as the stack pointer, and system stack using RR14′ as the system stack pointer (see Figure 1). If the CPU is operating in the Normal Mode, RR14 is active, and if the CPU is in System Mode, RR14′ will be used instead of RR14. The implied stack pointer is a part of the general registers and hence can be manipulated using the instructions available for register operations.

STACK POINTER (AmZ8002)

The AmZ8002 architecture allows stacks to be maintained in the memory. Any general purpose register except R0 can be used as a stack pointer in stack manipulating instructions such as PUSH and POP. However, certain instructions such as subroutine call and return make implicit use of the register R15 as the stack pointer. Two implicit stacks are maintained – normal stack using R15 as the stack pointer and system stack using R15′ as the system stack pointer (see Figure 1). If the CPU is operating in the Normal Mode, R15 is active, and if the CPU is in System Mode R15′ will be used instead of R15. The implied stack pointer is a part of the general registers and hence can be manipulated using the instructions available for register operations.



*AmZ8001 only

Figure 1. CPU General Registers.

00971B-5

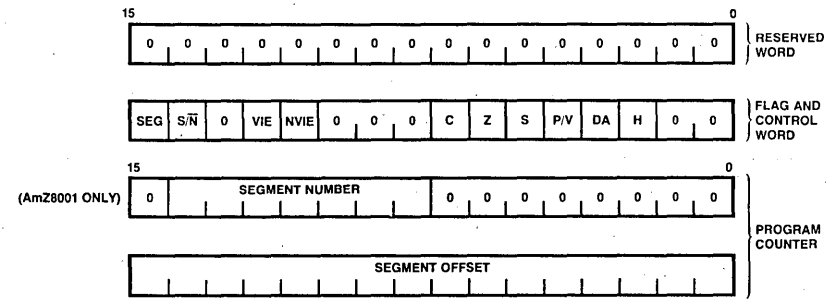


Figure 2. CPU Processor Status.

00971B-6

PROCESSOR STATUS

The CPU status consists of the 16-bit flag and control word (FCW) register, and the 16 or 23-bit program counter (see Figure 2). A reserved word is also included for future expansion. The following is a brief description of the FCW bits.

SEG: Segmented/Non-Segmented Bit. Indicates whether the AmZ8001 is running in segmented or non-segmented mode. 1 indicates segmented, 0 indicates non-segmented. See the section on non-segmented mode, elsewhere in this document. This bit is always 0 in the AmZ8002.

S/$\overline{N}$: System/Normal – 1 indicates System Mode and 0 indicates Normal Mode.

VIE: Vectored Interrupt Enable – 1 indicates that Vectored Interrupt requests will be honored.

NVIE: Non-Vectored Interrupt Enable – 1 indicates that Non-vectored interrupt requests will be honored.

C: Carry – 1 indicates that a carry has occurred from the most significant bit position when performing arithmetic operations.

Z: Zero – 1 indicates that the result of an operation is zero.

S: Sign – 1 indicates that the result of an operation is negative i.e., most significant bit is one.

P/V: Parity/Overflow – 1 indicates that there was an overflow during arithmetic operations. For byte logical operations this bit indicates parity of the result.

DA: Decimal Adjust – Records byte arithmetic operations.

H: Half Carry – 1 indicates that there was a carry from the most significant bit of the lower digit during byte arithmetic.

DATA TYPES

The CPU instructions operate on bits, digits (4 bits), bytes (8 bits), words (16 bits), long words (32 bits), byte strings and word strings type operands. Bits can be set, reset or tested. Digits are used to facilitate BCD arithmetic operations. Bytes are used for characters and small integers. Words are used for integer values and addresses while long words are used for large integer values and addresses. All operands except strings can reside either in memory or general registers. Strings can reside in memory only.

INTERRUPT AND TRAP STRUCTURE

Interrupt is defined as an external asynchronous event requiring program interruption. For example, interruption is caused by a peripheral needing service. Traps are synchronous events resulting from execution of certain instructions under some defined circumstances. Both interrupts and traps are handled in a similar manner.

The CPU supports three types of interrupts in order of descending priority – non-maskable, vectored and non-vectored. The vectored and non-vectored interrupts can be disabled by appropriate control bits in the FCW. The CPU has four traps – system call, segment trap, unimplemented opcode and privileged instruction. The traps have higher priority than interrupts.

When an interrupt or trap occurs, the current program status is automatically pushed on to the system stack. The program status consists of processor status (i.e., PC and FCW) plus a 16-bit identifier. The identifier contains the reason, source and other coded information relating to the interrupt or trap.

After saving the current program status, the new processor status is automatically loaded from the new program status area located in the memory. This area is designated by the New Program Status Area Pointer (NPSAP) register.

SEGMENTED ADDRESSING (AmZ8001 Only)

The AmZ8001 can directly address up to 8MB of memory space, using a 23-bit segmented address. The memory space is divided up into 128 segments, each up to 64KB in size. The upper seven bits of address designate the segment number, and are available on the SN0-SN6 outputs during a memory transaction. See the section on memory transactions for details.

The lower sixteen bits of address designate an offset within the segment, relative to the start of the segment, and are available on AD0-AD15 during part of the memory transaction. See the section on memory transactions for details.

The segmented address may be stored as a long word in memory, or in a register pair. The segment number and offset can be manipulated separately or together, by suitable use of the instruction set.

When the segmented address is contained in code space, a short offset format may be adopted. The segmented address is stored as one word, seven bits of segment number and eight bits of offset. Figure 3 shows the format for segmented addresses.

ADDRESSING MODES

Information contained in the CPU instructions consists of the operation to be performed, the operand type and the location of the operands. Operand locations are designated by general register addresses, memory addresses or I/O addresses. The addressing mode of a given instruction defines the address space referenced and the method to compute the operand address. Addressing modes are explicitly specified or implied in an instruction. Figure 4 illustrates the eight explicit addressing modes: Register (R), Immediate (IM), Indirect Register (IR), Direct Address (DA), Indexed (X), Relative Address (RA), Base Address (BA) and Base Indexed (BX).

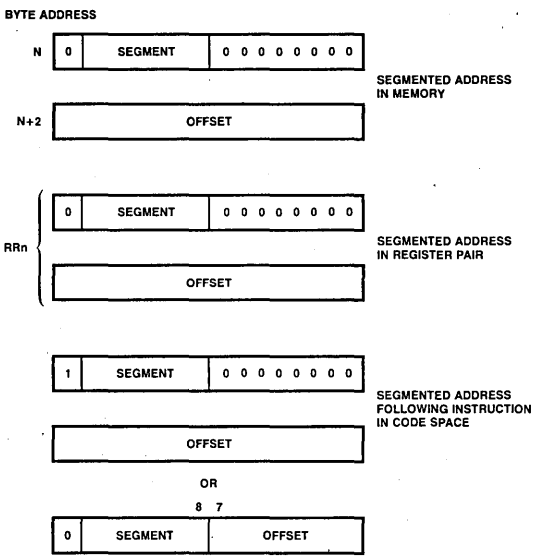


Figure 3. Segmented Address Formats.

00971B-7

When an effective segmented address is being computed according to the designated addressing mode, the segment number is not affected by any carry from the 16-bit offset.

NON-SEGMENTED MODE ON THE AmZ8001

The AmZ8001 can execute code designed to run on the non-segmented AmZ8002. This is achieved by changing the mode of execution of the AmZ8001 from segmented to non-segmented by writing a 0 to the SEG bit in the FCW. (See the section on processor status.) The change to non-segmented mode sets up a suitable environment for running non-segmented code. However, this environment only exists within the code segment that caused the change of mode from segmented to non-segmented.

SN0-SN6 will continue to indicate the code segment until a reset, interruption or return to segmented mode is encountered.

The effects of the non-segmented mode of operation on the AmZ8001 are described below.

a) The AmZ8001 will interpret instruction length as if it was a non-segmented AmZ8002.
b) The AmZ8001 will implement address computation in an identical manner to the AmZ8002.

Other CPU functions, such as interrupt and trap handling, reset and stack pointer manipulation are unaltered. These functions are characterized by the type of CPU, not by the state of the SEG bit in the FCW.

INPUT/OUTPUT

A set of I/O instructions are provided to accomplish byte or word transfers between the CPU and I/O devices. I/O devices are addressed using 16-bit I/O port addresses and I/O address space is not a part of the memory address space. Two types of I/O instructions are provided; each with its own 16-bit address space. I/O instructions include a comprehensive set of In, Out and Block transfers.

CPU TIMING

The CPU accomplishes instruction execution by stepping through a pre-determined sequence of machine cycles, such as memory read, memory write, etc. Each machine cycle requires between three and ten clock cycles. Bus Requests by DMA devices are granted at machine cycle boundaries. No machine cycle is longer than ten clock cycles; thus assuring fast response to a Bus Request (assuming no extra wait states). The start of a machine cycle is always marked by a LOW pulse on the $\overline{AS}$ output. The status output lines ST0-ST3 indicate the nature of the current cycle in a coded form.

STATUS LINE CODES

Status line coding was listed in the table shown under ST0-ST3 outputs in the Interface Signal Description. The following is a detailed description of the status codes.

Internal Operation:

This status code indicates that the CPU is going through a machine cycle for its internal operation. Figure 5 depicts an internal operation cycle. It consists of three clock periods identified as T1, T2 and T3. The $\overline{AS}$ output will be activated with a LOW pulse by the AmZ8001 to mark the start of a machine cycle. The ST0-ST3 will reflect the code for the internal operation. The $\overline{MREQ}$, $\overline{DS}$ and R/$\overline{W}$ outputs will be HIGH. The N/$\overline{S}$ and SN0-SN6 outputs will remain at the same level as in the previous machine cycle. The CPU will ignore the $\overline{WAIT}$ input during the internal operation cycle. The CPU will drive the AD0-AD15 bus with unspecified information during T1. However, the bus will go into high impedance during T2 and remain in that state for the remainder of the cycle. The B/$\overline{W}$ output is also activated by the CPU with unspecified information.

Memory Refresh:

This status code indicates that CPU is accessing the memory to refresh. The refresh cycle consists of three clock periods as depicted in Figure 6. The CPU will activate the $\overline{AS}$ output with a LOW pulse to mark the beginning of a machine cycle and ST0-ST3 outputs will reflect the refresh cycle code. The least significant 9 lines of the AD0-AD15 bus contain the refresh address. Because the memory is word organized, the AD0 will always be LOW. The most significant 7 bus lines are not specified. The $\overline{DS}$ output will remain HIGH for the entire cycle while R/$\overline{W}$, B/$\overline{W}$, SN0-SN6 and N/$\overline{S}$ outputs will remain at the same level as in the machine cycle prior to refresh. The AD0-AD15 bus will go into high impedance state during T2 period and remain there for the remainder of the cycle. The CPU will activate the $\overline{MREQ}$ output LOW during the refresh cycle. It should be noted that $\overline{WAIT}$ input is ignored by the CPU for refresh operations.

I/O Transactions:

There are two status line codes used for I/O transaction cycles. The CPU provides two separate I/O spaces and two types of instructions called Normal I/O and Special I/O. Each I/O space is addressed by a 16-bit address called port address. The timing for both types of I/O transactions is essentially identical. A typical I/O cycle consists of four clock periods T1, T2, TWA and T3 as shown in Figure 7. The TWA is the wait state; insertion of one wait state for an I/O cycle is always automatic. Additional wait cycles can be inserted by LOW on the $\overline{WAIT}$ input. The $\overline{WAIT}$ input is sampled during every TW state. If this input is LOW, one more wait state will be inserted. Insertion of wait states continues until $\overline{WAIT}$ input is HIGH. T3 state will follow the last wait state to complete the I/O cycle.

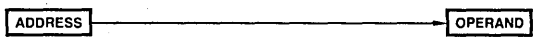
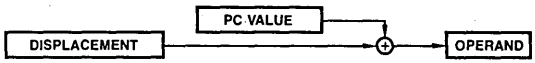
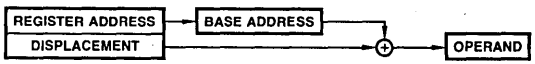
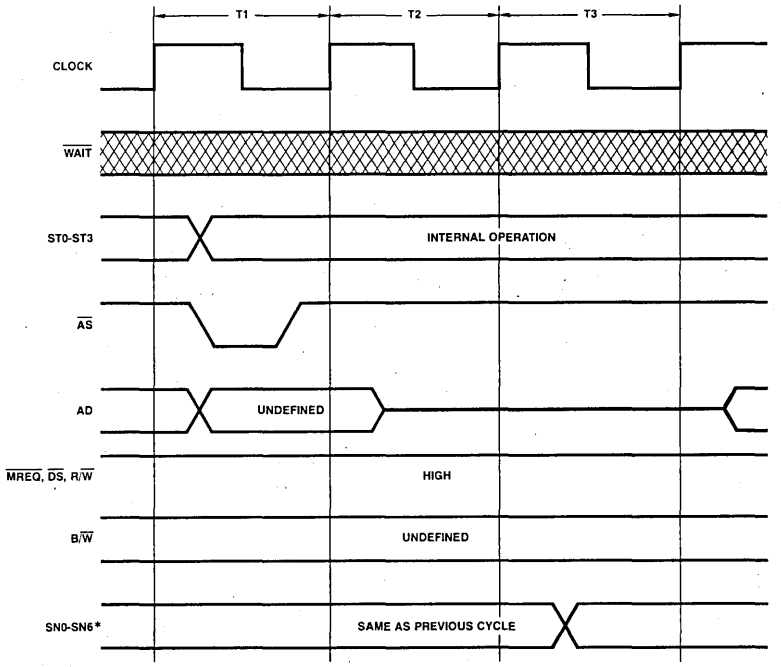
Mode	Operand Addressing			Operand Value
	In the Instruction	In a Register	In Memory	
Register				The content of the register.
Immediate				In the instruction
Indirect Register				The content of the location whose address is in the register.
Direct Address				The content of the location whose address is in the instruction.
Index				The content of the location whose address is the address in the instruction, offset by the content of the working register.
Relative Address				The content of the location whose address is the content of the program counter, offset by the displacement in the instruction.
Base Address				The content of the location whose address is the address in the register, offset by the displacement in the instruction.
Base Index				The content of the location whose address is the address in the register, offset by the displacement in the register.

Figure 4. Addressing Modes.

00971B-8

During I/O cycles the ST0-ST3 outputs will reflect appropriate code depending on the type of instruction being executed (Normal I/O or Special I/O). $\overline{AS}$ output will be pulsed LOW to mark the beginning of the cycle. The CPU drives the AD0-AD15 bus with the 16-bit port address specified by the current instruction. The N/$\overline{S}$ output will be LOW indicating that CPU is operating in the system mode. It should be recalled that the N/$\overline{S}$ output is derived from the appropriate bit in the FCW register. All I/O instructions are privileged instructions and will be allowed to execute only if the FCW specifies system mode operation. The $\overline{MREQ}$ output will be HIGH. The I/O instructions provide both word or byte transactions. The B/$\overline{W}$ output will be HIGH or LOW depending whether the instruction specifies a byte or word transfer. The SN0-SN6 output will remain at the same level as in the machine cycle prior to the I/O cycle.

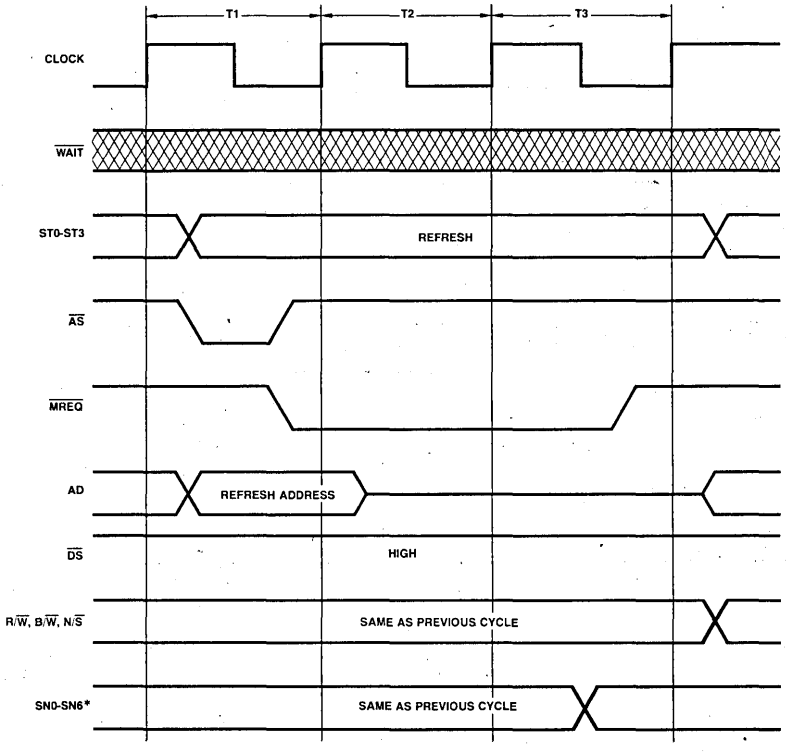
Two kinds of I/O transfers should be considered: Data In means reading from the device and Data Out means writing into the device. For In operations, the R/$\overline{W}$ output will be HIGH. The AD0-AD15 bus will go into high impedance state during T2. During byte input instructions, the CPU reads either the even or odd half of the Data Bus dependent upon the port address. If the port address is even, the most significant half of the Data Bus is read. If the port address is odd, the least significant half of the Data Bus is read. During word input instructions, the CPU reads all 16 bits of the Data Bus. The CPU will drive the $\overline{DS}$



*AmZ8001 only

Figure 5. Internal Operation Cycle.

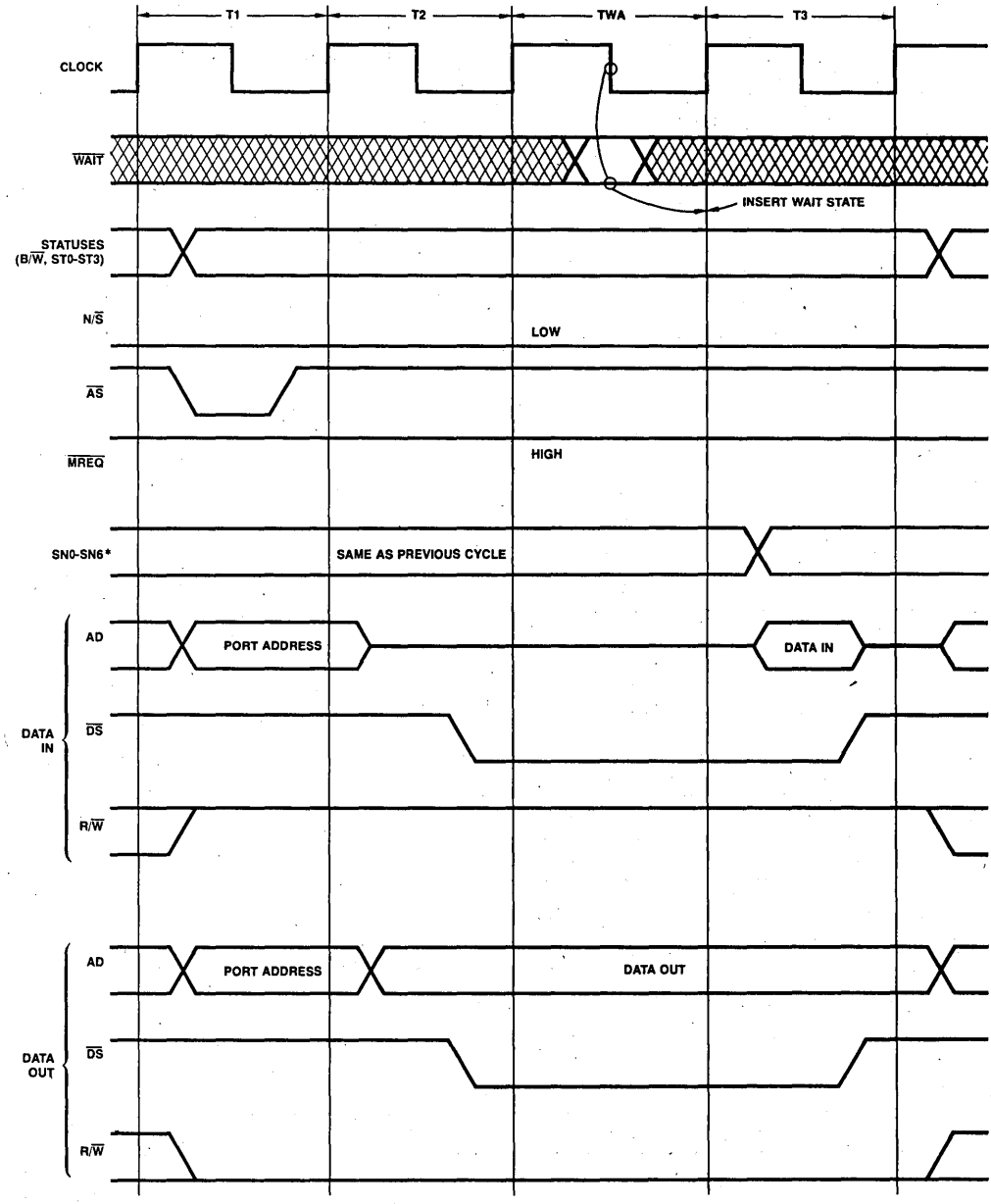
00971B-9



*AmZ8001 only

Figure 6. Refresh Cycle.

00971B-10



*AmZ8001 only

Figure 7. AmZ8001 I/O Cycle.

00971B-11

output LOW to signal to the device that data can be gated on to the bus. The CPU will accept the data during T3 and $\overline{DS}$ output will go HIGH signalling the end of an I/O transaction.

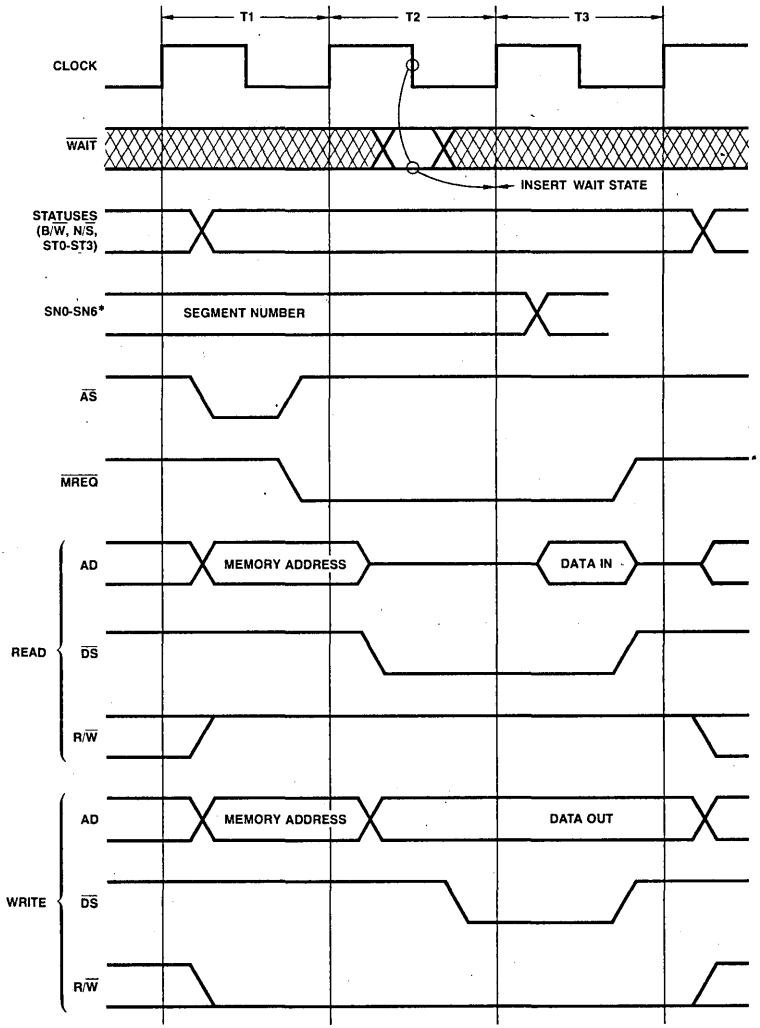
For Data Out, the R/$\overline{W}$ output will be LOW. The CPU will provide data on the AD0-AD15 bus and activates the $\overline{DS}$ output LOW during T2. During byte output instructions, the CPU duplicates the byte data onto both the high and low halves of the Data Bus and external logic, using A0, enables the appropriate byte port. During word output instructions the CPU outputs data onto all 16 bits of the Data Bus. The $\overline{DS}$ output goes HIGH during T3 and the cycle is complete.

Memory Transactions:

There are four status line codes that indicate a memory transaction:

a) Memory transaction to read or write an operand
b) Memory transaction to read from or write into the stack
c) Memory transaction to fetch the first word of an instruction (sometimes called IF1)
d) Memory transaction to fetch the subsequent word of an instruction (sometimes called IFN).

It can be appreciated that all the above transactions essentially fall into two categories: memory read and memory write. In the case of IF1 and IFN cycles, the memory will be read at the address supplied by the program counter. All instructions are multiples of 16-bit words. Words are always addressed by an even address. Thus IF1 and IFN cycles involve performing a memory read for words. On the other hand, a memory transaction for operand and stack operation could be a read or write. Moreover, an operand could be a word or a byte. For stack operation involving the implied stack pointer the address will be in the appropriate stack pointer register (R15, R15', RR14 or RR14'). For operand transactions, the memory address will come from several sources depending on the instruction and the addressing mode. Memory transaction cycle timing is shown in Figure 8. It typically consists of three clock periods



*AmZ8001 only

Figure 8. Memory Transactions.

00971B-12

T1, T2 and T3. Wait states (TW) can be inserted between T2 and T3 by activating the $\overline{WAIT}$ input LOW. The $\overline{WAIT}$ input will be sampled during T2 and during every subsequent TW. The ST0-ST3 outputs will reflect the appropriate code for the current cycle early in T1 and the $\overline{AS}$ output will be pulsed LOW to mark the beginning of the cycle. The N/$\overline{S}$ output will indicate whether the normal or system address space will be used for the current cycle. As shown in the figure the $\overline{MREQ}$ output will go LOW during T1 to indicate a memory operation.

The segment number becomes valid on the segment lines one clock period before the start of the memory operation, and remains valid until the start of T3.

Consider a read operation first. The R/$\overline{W}$ output will be HIGH. The CPU will drive the AD0-AD15 with the appropriate address early in T1. During T2, the bus will go into high-impedance state and $\overline{DS}$ output will be activated LOW by the CPU. The data can be gated on to the bus when $\overline{DS}$ is LOW. During T1 the B/$\overline{W}$ will also be activated to indicate byte or word will be transacted. The memory is word organized and words are addressed by even addresses. However, when addressing bytes, the memory address may be odd or even; an even address for most significant byte of a word and the next odd address for the least significant byte of that word. When reading a byte from the memory, the least significant address bit can be ignored and the whole word containing the desired byte is gated on to the bus. The CPU will pick the appropriate byte automatically and will drive the DS output HIGH indicating data acceptance.

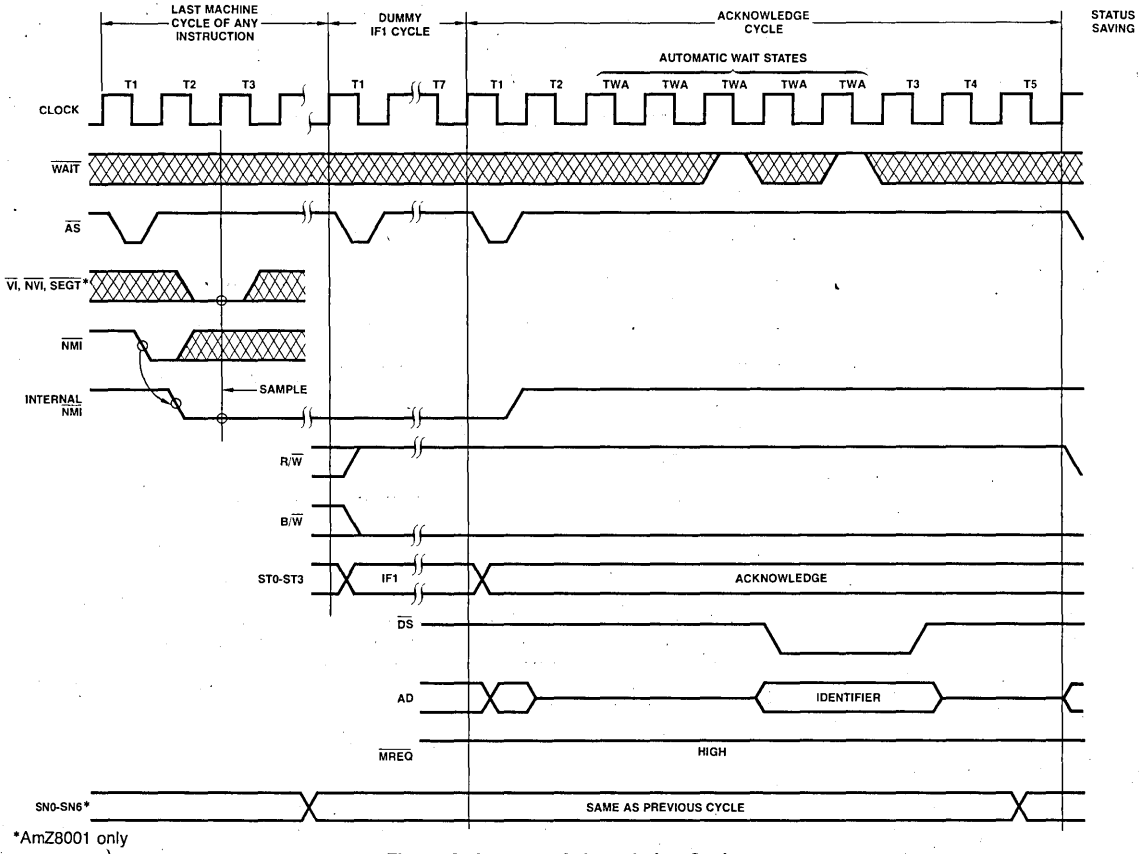
Consider the write operation next. The R/$\overline{W}$ output will be LOW. The CPU removes the address and gates out the data to be written on the bus and activates the $\overline{DS}$ output LOW during T2. If the data to be written is a byte then the same byte will be on both halves of the bus. The $\overline{DS}$ output will go HIGH during T3 signifying completion of the cycle.

Interrupt and Segment Trap Acknowledge:

There are four status line codes devoted to interrupt and trap acknowledgement. These correspond to non-maskable, vectored and non-vectored interrupts, as well as segment trap. The Interrupt Acknowledge cycle is illustrated in Figure 9. The $\overline{NMI}$ input of the AmZ8001 is edge detected i.e., a HIGH to LOW input level change is stored in an internal latch. Similar internal storage is not provided for the $\overline{VI}$, $\overline{NVI}$, and $\overline{SEGT}$ inputs. For $\overline{VI}$ and $\overline{NVI}$ inputs to cause an interruption, the corresponding interrupt enable bits in the FCW must be 1. For the following discussion, both the VIE and NVIE bits in the FCW are assumed to be 1.

As shown in the figure, the $\overline{VI}$, $\overline{NVI}$ and $\overline{SEGT}$ input and the internal $\overline{NMI}$ latch output are sampled during T3 of the last machine cycle of an instruction.

A LOW on these signals triggers the corresponding interrupt acknowledge sequence described on the following page. The CPU executes a dummy IF1 cycle prior to entering the actual acknowledge cycle (see memory transactions for IF1 cycle description).



*AmZ8001 only

Figure 9. Interrupt Acknowledge Cycle.

00971B-13

During this dummy IF1 cycle, the program counter is not updated; instead the implied system stack pointer (RR14') will be decremented. Following the dummy IF1 cycle is the actual interrupt/trap acknowledge cycle.

The interrupt acknowledge cycle typically consists of 10 clock periods; T1 through T5 and five automatic TW (wait states). As usual, the $\overline{AS}$ output will be pulsed LOW during T1 to mark the beginning of a cycle. The ST0-ST3 outputs will reflect the appropriate interrupt acknowledge code, the $\overline{MREQ}$ output will be HIGH, the N/$\overline{S}$ output remains the same as in the preceding cycle, the R/$\overline{W}$ output will be HIGH and the B/$\overline{W}$ output will be LOW. The CPU will drive the Ad0-AD15 bus with unspecified information during T1 and the bus will go into the high impedance state during T2. Three TWA states will automatically follow T2. The $\overline{WAIT}$ input will be sampled during the third TWA state.

If LOW, an extra TW state will be inserted and the $\overline{WAIT}$ will be sampled again during TW. Such insertion of TW states continues until the $\overline{WAIT}$ input is HIGH. After the last TW state, the $\overline{DS}$ output will go LOW and two more automatic wait states (TWA) follow. The interrupting device can gate up to a 16-bit identifier on to the bus when the $\overline{DS}$ output is LOW. The $\overline{WAIT}$ input will be sampled again during the last TWA state. If the $\overline{WAIT}$ input is LOW one TW state will be inserted and the $\overline{WAIT}$ will be sampled during TW. Such TW insertion continues until the $\overline{WAIT}$ input is HIGH. After completing the last TW state T3 will be entered and the $\overline{DS}$ output will go HIGH. The interrupting device should remove the identifier and cease driving the bus. T4 and T5 states will follow T3 to complete the cycle. Following the interrupt acknowledge cycle will be memory transaction cycles to save the status on the stack. Note that the N/$\overline{S}$ output will be automatically LOW during status saving. The SN0-SN6 outputs are undefined during the acknowledge cycle.

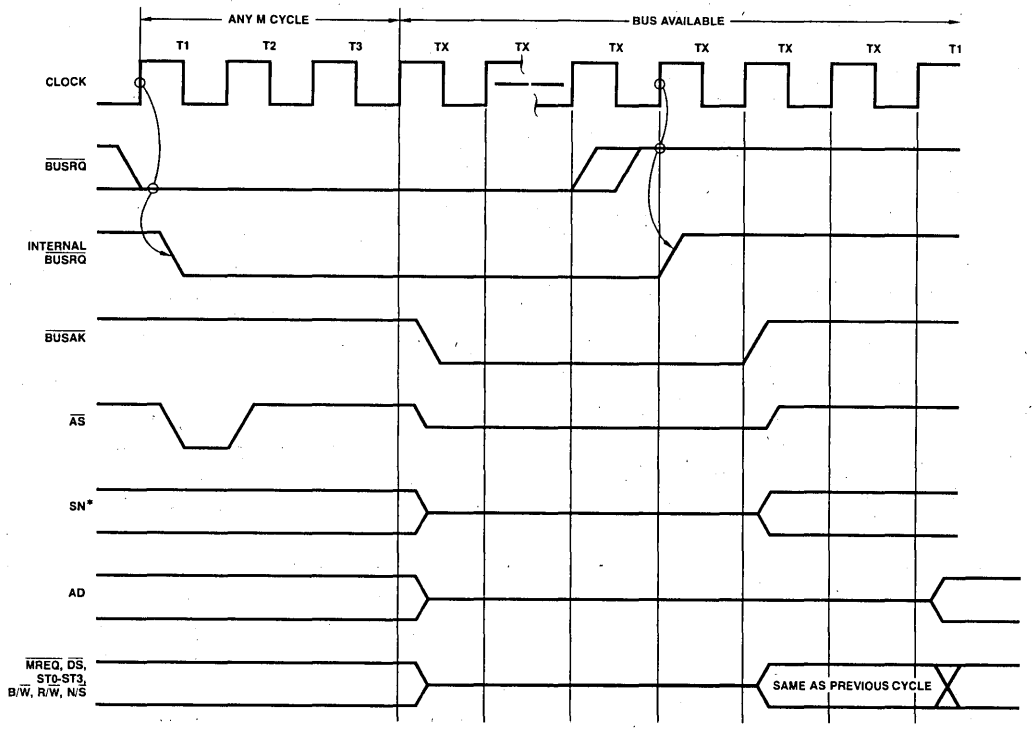
The internal NMI latch will be reset to the initial state at $\overline{AS}$ going HIGH in the interrupt acknowledge cycle. The $\overline{VI}$, $\overline{NVI}$ and $\overline{SEGT}$ input should be kept LOW until this time also.

STATUS SAVING SEQUENCE:

The machine cycles following the interrupt acknowledge cycle push the old status information on the system stack in the following order: program counter, the flag and control word and the interrupt/trap identifier. Subsequent machine cycles fetch the new program status from the new program status area, and then branch to the interrupt/service routine.

BUS REQUEST/BUS ACKNOWLEDGE TIMING:

A LOW on the $\overline{BUSRQ}$ input is an indication to the CPU that another device (such as DMA) is requesting control of the bus. The $\overline{BUSRQ}$ input is synchronized internally at T1 of any machine cycle. (See next paragraph for exception.) The $\overline{BUSAK}$ will go LOW after the last clock period of the machine cycle. The LOW on the $\overline{BUSAK}$ output indicates acknowledgement. When $\overline{BUSAK}$ is LOW the following outputs will go into the high impedance state; AD0-AD15, $\overline{AS}$, $\overline{DS}$, $\overline{MREQ}$, ST0-ST3, B/$\overline{W}$, R/$\overline{W}$, SN0-SN6 and N/S. The BUSRQ must be held LOW until all transactions are completed. When $\overline{BUSRQ}$ goes HIGH, it is synchronized internally, the $\overline{BUSAK}$ output will go HIGH and normal CPU operation will resume. Figure 10 illustrates the $\overline{BUSRQ}$/$\overline{BUSAK}$ timing.



*AmZ8001 only

Figure 10. Bus Request/Acknowledge Cycle.

00971B-14

It was mentioned that $\overline{BUSRQ}$ will be honored during any machine cycle with one exception. This exception is during the execution of TSET/TSETB instructions. $\overline{BUSRQ}$ will not be honored once execution of these instructions has started.

SINGLE STEPPING

The $\overline{STOP}$ input of the CPU facilitates one instruction at a time or single step operation. Figure 11 illustrates $\overline{STOP}$ input timing. The $\overline{STOP}$ input is sampled on the HIGH to LOW transition of the clock input that immediately precedes an IF1 cycle. If the $\overline{STOP}$ is found LOW, AmZ8001 introduces a memory refresh cycle after T3. Moreover, $\overline{STOP}$ input will be sampled again at T3 in the refresh cycle. If $\overline{STOP}$ is LOW one more refresh cycle will follow the previous refresh cycle. The $\overline{STOP}$ will be sampled during T3 of the refresh cycle also. One additional refresh cycle will be added every time $\overline{STOP}$ input is sampled LOW. After completing the last refresh cycle which will occur after $\overline{STOP}$ is HIGH, the CPU will insert two dummy states T4 and T5 to complete the IF1 cycle and resume its normal operations for executing the instruction. See appropriate sections on memory transactions and memory refresh. It should be noted that refresh cycles will occur even if the refresh facility is disabled during single stepping.

MULTIMICROPROCESSOR FACILITIES

The CPU is provided with hardware and software facilities to support multiple microprocessor systems. The $\overline{\mu O}$ and $\overline{\mu I}$ signals of the CPU are used in conjunction with the MBIT, MREQ, MRES and MSET instructions for this purpose. The $\overline{\mu O}$ output can be activated LOW by using appropriate instruction to signal a request from the CPU for a resource. The $\overline{\mu I}$ input is tested by the CPU before activating the $\overline{\mu O}$ output. LOW at the $\overline{\mu I}$ input indicates that the resource is busy. The CPU can examine the $\overline{\mu I}$ input after activating the $\overline{\mu O}$ output LOW. The $\overline{\mu I}$ will be tested again to see if the requested resource became available.

INITIALIZATION

A LOW on the $\overline{Reset}$ input starts the CPU initialization. The initialization sequence is shown in Figure 12. Within five clock periods after the HIGH to LOW level change of the $\overline{Reset}$ input the following will occur:

a) AD0-AD15 bus will be in the HIGH impedance state
b) $\overline{AS}$, $\overline{DS}$, $\overline{MREQ}$, $\overline{BUSAK}$ and $\overline{\mu O}$ outputs will be HIGH
c) ST0-ST3 outputs will be LOW
d) Refresh will be disabled
e) R/$\overline{W}$, B/$\overline{W}$ and N/$\overline{S}$ outputs are not affected. For a power on reset the state of these outputs is not specified.
f) SN0-SN6 outputs will be LOW.

After the $\overline{Reset}$ input returns HIGH and remains HIGH for three clock periods, two (three for the AmZ8001) 16-bit memory read operations will be performed as follows. Note that the N/$\overline{S}$ output will be LOW and ST0-ST3 outputs will reflect IFN code.

a) The contents of the memory location 0002 (segment 0) will be read. This information will be loaded into the FCW of the CPU.
b) The contents of the memory location 0004 (segment 0) will be read. This information will be loaded into the program counter segment number.
c) (AmZ8001 only.) The contents of the memory location 0006 (segment 0) will be read. This information will be loaded into the program counter offset.

This completes initialization sequence and an IF1 cycle will follow to fetch the first instruction to begin program execution. See the section on memory transactions for timing.

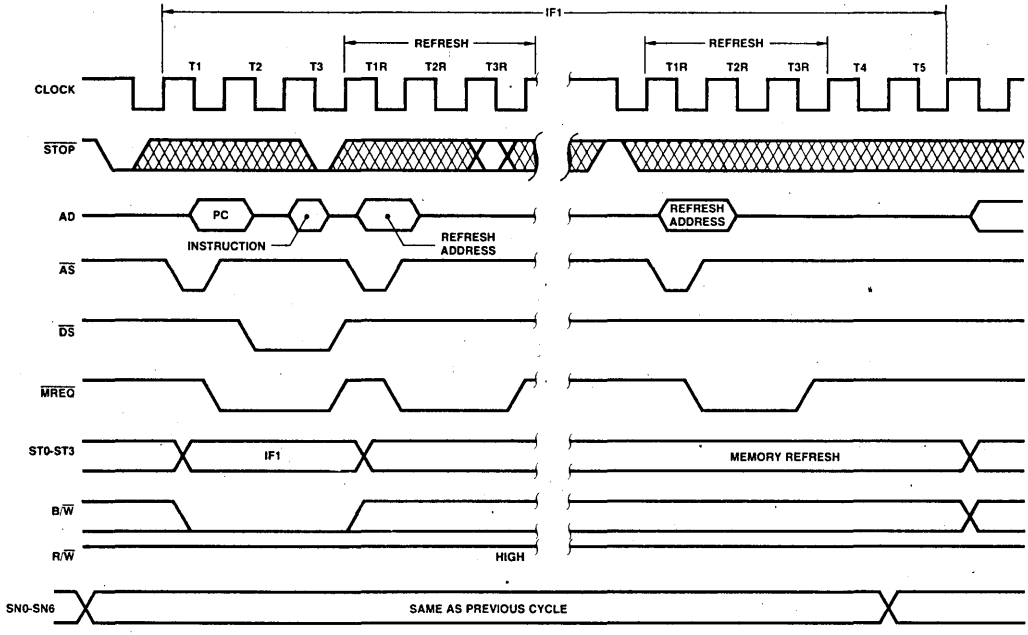
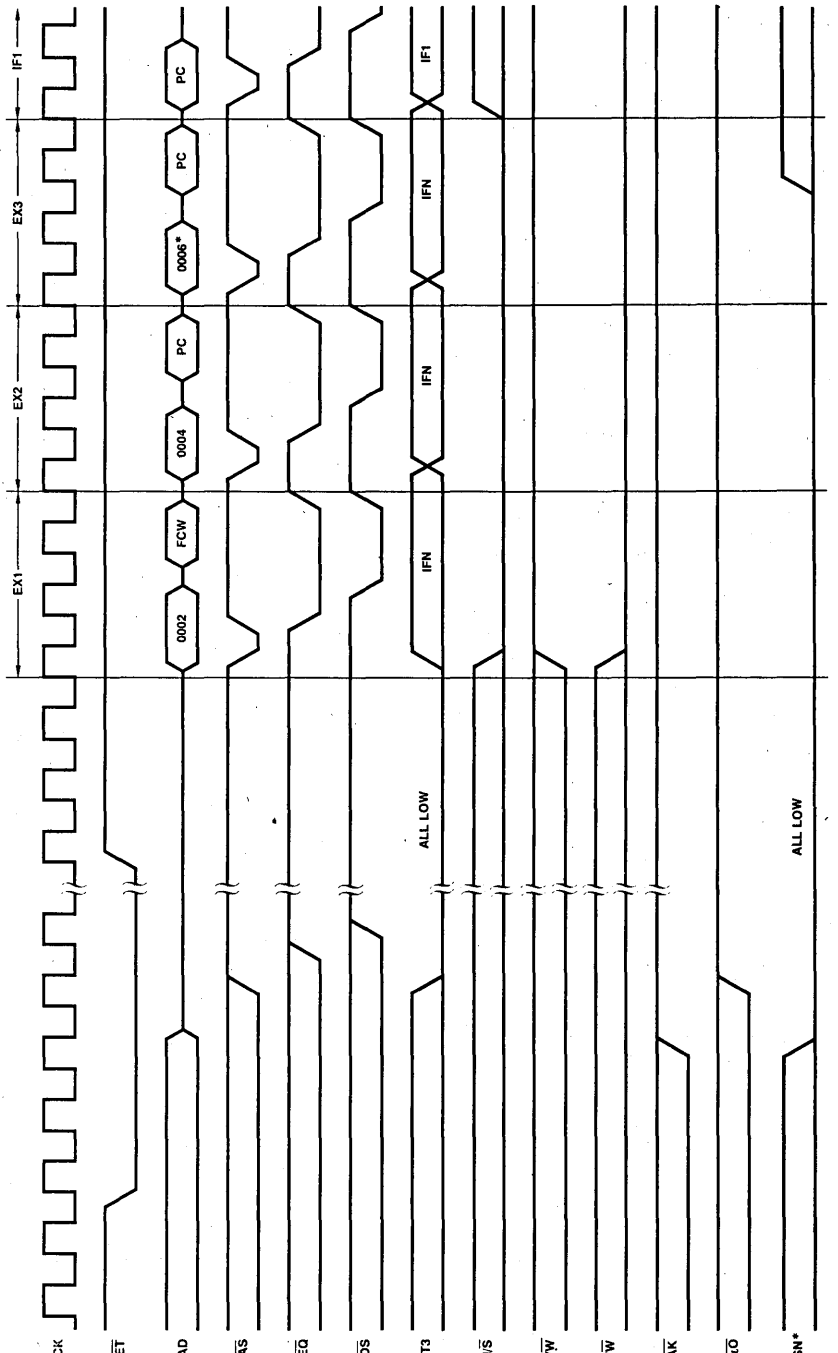


Figure 11. Single Step Timing.

00971B-15



00971B-16

Figure 12. Reset Sequence.

AmZ8001/2 CPU INSTRUCTION SET

ARITHMETIC

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
	ADC ADCB	R, src	R	5	Add with Carry R ← R + src + carry
Add	ADD ADDB ADDL	R, src	R IM IR DA X	4 7 7 9 10	Add R ← R + src
Compare	CP CPB CPL	R, src	R IM IR X	4 7 7 10	Compare with Register R − src
	CP CPB	dst, IM	IR DA X	11 14 15	Compare with Immediate dst − IM
	DAB	dst	R	5	Decimal Adjust
Decrement	DEC DECB	dst, n	R IR DA X	4 11 13 14	Decrement by n dst ← dst − n (n = 1 . . . 16)
Divide	DIV DIVL	R, src	R IM IR DA X	107 107 107 108 109	Divide (signed) Word: $R_n + 1 \leftarrow R_{n,n+1} \div src$ $R_n \leftarrow$ remainder Long Word: $R_{n+2,n+3}$ $\leftarrow R_{n...n+3} \div src$ $R_{n,n+1}$ ← remainder
Extend	EXTS EXTSB EXTSL	dst	R	11	Extend Sign Extend sign of low order half of st through high order half of dst
Increment	INC INCB	dst, n	R IR DA X	4 11 13 14	Increment by n dst ← dst + n (n = 1 . . . 16)
Multiply	MULT MULTL	R, src	R IM IR DA X	70 70 70 71 72	Multiply (signed) Word: $R_{n,n+1} \leftarrow R_n \cdot src$ Long Word: $R_{n...n+3}$ $\leftarrow R_{n+2,n+3} \cdot src$ *Plus seven cycles for each 1 in the multiplicand
Negate	NEG NEGB	dst	R R DA X	7 12 15 16	Negate dst ← 0 − dst
Subtract	SBC SBCB	R, src	R	5	Subtract with Carry R ← R − src − carry
	SUB SUBB SUBL	R, src	R IM IR DA X	4 7 7 9 10	Subtract R ← R − src

LOGICAL

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
AND	AND ANDB	R, src	R IM IR DA X	4 7 7 9 10	AND R ← R AND src
Complement	COM COMB	dst	R IR DA X	7 12 15 16	Complement dst ← NOT dst
OR	OR ORB	R, src	R IM IR DA X	4 7 7 9 10	OR R ← R OR src
Test	TEST TESTB TESTL	dst	R IR DA X	7 8 11 12	TEST dst OR 0
	TCC TCCB	cc, dst	R	5	Test Condition Code Set LSB if cc is true
XOR	XOR XORB	R, src	R IM IR DA X	4 7 7 9 10	Exclusive OR R ← R XOR src

†Clock cycles for byte or word data, non-segmented addresses. Segmented addresses may require 2 to 4 additional cycles. Some long word data require more cycles. See AmZ8000 Users Manual for exact data.

LOAD AND EXCHANGE

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Clear	CLR CLRB	dst	R IR DA X	7 8 11 12	Clear dst ← 0
Exchange	EX EXB	R, src	R IR DA X	6 12 15 16	Exchange R ← src
Load	LD LDB LDL	R, src	R IM IR DA X RA BA BX	3 7 7 9 10 14 14 14	Load into Register R ← src
	LD LDB LDL	dst, R	IR DA X RA BA BX	8 11 12 14 14 14	Load into Memory (Store) dst ← R
	LD LDB	dst, IM	R IR DA X	5 11 14 15	Load Immediate into Memory dst ← IM
	LDA	R, src	DA X RA BA BX	12 13 15 15 15	Load Address R ← source address
	LDAR	R, src	RA	15	Load Address Relative R ← source address
	LDK	R, src	IM	5	Load Constant R ← n (n = 0 . . . 15)
	LDM	R, src, n	IR DA X	11+3n 14+3n 15+3n	Load Multiple R ← src (n consecutive words) (n = 1 . . . 16)
	LDM	dst, R, n	IR DA X	11+3n 14+3n 15+3n	Load Multiple (Store Multiple) dst ← R (n consecutive words) (n = 1 . . . 16)
	LDR LDRB LDRL	R, src	RA	14	Load Relative R ← src (range −32768 . . . +32767)
	LDR LDRB LDRL	dst, R	RA	14	Load Relative (Store Relative) dst ← R (range −32768 . . . +32767)
Pop	POP POPL	dst, R	R IR DA X	8 12 16 16	Pop dst ←IR Autoincrement contents of R
Push	PUSH PUSHL	IR, src	R IM IR DA X	9 12 13 14 14	Push Autodecrement contents of R IR ← src

BIT MANIPULATION

	Mnemonics	Operand	Addr. Modes	Clock Cycles†	Operation
Test	BIT BITB	dst, b	R IR DA X	4 8 10 11	Test Bit Static Z flag ← NOT dst bit specified by b
	BIT BITB	dst, R	R	10	Test Bit Dynamic Z flag ← NOT dst bit specified by contents of R
Reset	RES RESB	dst, b	R IR DA X	4 11 13 14	Reset Bit Static Reset dst bit specified by b
	RES RESB	dst, R	R	10	Reset Bit Dynamic Reset dst bit specified by contents of R
Set	SET SETB	dst, b	R IR DA X	4 11 13 14	Set Bit Static Set dst bit specified by b
	SET SETB	dst, R	R	10	Set Bit Dynamic Set dst bit specified by contents of R
Test and Set	TSET TSETB	dst	R IR DA X	7 11 14 15	Test and Set S flag ← MSB of dst dst ← all 1s

ROTATE AND SHIFT

	Mnemonics	Operand	Addr. Modes	Clock Cycles†	Operation
Rotate	RLDB	R, src	R	9	Rotate Digit Left
	RRDB	R, src	R	9	Rotate Digit Right
	RL RLB	dst, n	R R	6	Rotate Left by n bits (n = 1, 2)
	RLC RLCB	dst, n	R R	6	Rotate Left through Carry by n bits (n = 1, 2)
	RR RRB	dst, n	R R	6	Rotate Right by n bits (n = 1, 2)
	RRC RRCB	dst, n	R R	6	Rotate Right through Carry by n bits (n = 1, 2)
Shift	SDA SDAB SDAL	dst, R	R	15+3n	Shift Dynamic Arithmetic Shift dst left or right by contents of R
	SDL SDLB SDLL	dst, R	R	15+3n	Shift Dynamic Logical Shift dst left or right by contents of R
	SLA SLAB SLAL	dst, n	R	13+3n	Shift Left Arithmetic by n bits
	SLL SLLB SLLL	dst, n	R	13+3n	Shift Left Logical by n bits
	SRA SRAB SRAL	dst, n	R	13+3n	Shift Right Arithmetic by n bits
	SRL SRLB SRLL	dst, n	R	13+3n	Shift Right Logical by n bits

†Clock cycles for byte or word data, non-segmented addresses. Segmented addresses may require 2 to 4 additional cycles. Some long word data require more cycles. See AmZ8000 Users Manual for exact data.

BLOCK TRANSFER AND STRING MANIPULATION (AUTO INCREMENT/DECREMENT AND REPEAT

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Compare	CPD CPDB	Rx, src, Ry, cc	IR	20	Compare and Decrement Rx − src Autodecrement src address Ry ← Ry − 1
	CPDR CPDRB	Rx, src, Ry, cc	IR	11+9n	Compare, Decrement and Repeat Rx − src Autodecrement src address Ry ← Ry − 1 Repeat until cc is true or Ry = 0
	CPI CPIB	Rx, src, Ry, cc	IR	20	Compare and Increment Rx − src Autoincrement src address Ry ← Ry − 1
	CPIR CPIRB	Rx, src, Ry, cc	IR	11+9n	Compare, Increment and Repeat Rx − src Autoincrement src address Ry ← Ry − 1 Repeat until cc is true or Ry = 0
	CPSD CPSDB	dst, src, R, cc	IR	25	Compare String and Decrement dst − src Autodecrement dst and src addresses R ← R − 1
	CPSDR CPSDRB	dst, src, R, cc	IR	11+14n	Compare String, Decr. and Repeat dst − src Autodecrement dst and src addresses R ← R − 1 Repeat until cc is true or R = 0
	CPSI CPSIB	dst, src, R, cc	IR	25	Compare String and Increment dst − src Autoincrement dst and src addresses R ← R − 1
	CPSIR CPSIRB	dst, src, R, cc	IR	11+14n	Compare String, Incr. and Repeat dst − src Autoincrement dst and src addresses R ← R − 1 Repeat until cc is true or R = 0
Load	LDD LDDB	dst, src, R	IR	20	Load and Decrement dst ← src Autodecrement dst and src addresses R ← R − 1
	LDDR LDDRB	dst, src, R	IR	11+9n	Load, Decrement and Repeat dst ← src Autodecrement dst and src addresses R ← R − 1 Repeat until R = 0

BLOCK TRANSFER AND STRING MANIPULATION (Cont.)

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Load	LDI LDIB	dst, src, R	IR	20	Load and Increment dst ← src Autoincrement dst and src addresses R ← R − 1
	LDIR LDIRB	dst, src, R	IR	11+9n	Load, Increment and Repeat dst ← src Autoincrement dst and src addresses R ← R −1 Repeat until R = 0
Translate	TRDB	dst, src, R	IR	25	Translate and Decrement dst ← src (dst) Autodecrement dst address R ← R − 1
	TRDRB	dst, src, R	IR	11+14n	Translate, Decrement and Repeat dst ← src (dst) Autodecrement dst address R ← R −1 Repeat until R = 0
	TRIB	dst, src, R	IR	25	Translate and Increment dst ← src (dst) Autoincrement dst address R ← R −1 0
	TRIRB	dst, src, R	IR	11+14n	Translate, Increment and Repeat dst ← src (dst) Autoincrement dst address R ← R − 1 Repeat until R = 0
Translate and Test	TRTDB	src 1, src 2, R	IR	25	Translate and Test, Decrement RH1 ← src 2 (src 1) Autodecrement src 1 address R ← R − 1
	TRTDRB	src 1, src 2, R	IR	11+14n	Translate and Test, Decrement and Repeat RH1 ← src 2 (src 1) Autodecrement src 1 address R ← R − 1 Repeat until R = 0 or RH1 = 0
	TRTIB	src 1, src 2, R	IR	25	Translate and Test, Increment RH1 ← src 2 (src 1) Autoincrement src 1 address R ← R − 1
	TRTIRB	src 1, src 2, R	IR	11+14n	Translate and Test, Increment and Repeat RH1 ← src 2 (src 1) Autoincrement src 1 address R ← R − 1 Repeat until R = 0 or RH1 = 0

†Clock cycles for byte or word data, non-segmented addresses. Segmented addresses may require 2 to 4 additional cycles. Some long word data require more cycles. See AmZ8000 Users Manual for exact data.

INPUT/OUTPUT

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Input	IN* INB*	R, src	IR DA	10 12	Input R ← src
	IND* INDB*	dst, src, R	IR	21	Input and Decrement dst ← src Autodecrement dst address R ← R − 1
	INDR* INDRB*	dst, src, R	IR	11+10n	Input, Decrement and Repeat dst ← src Autodecrement dst address R ← R − 1 Repeat until R = 0
	INI* INIB*	dst, src, R	IR	21	Input and Increment dst ← src Autoincrement dst address R ← R − 1
	INIR* INIRB*	dst, src, R	IR	11+10n	Input, Increment and Repeat dst ← src Autoincrement dst address R ← R − 1 Repeat until R = 0
Output	OUT* OUTB*	dst, R	IR DA	10 12	Output dst ← R
	OUTD* OUTDB*	dst, src, R	IR	21	Output and Decrement dst ← src Autodecrement src address R ← R − 1
	OTDR* OTDRB*	dst, src, R	IR	11+10n	Output and Decrement dst ← src Autodecrement src address R ← R − 1 Repeat until R = 0
	OUTI* OUTIB*	dst, src, R	IR	21	Output and Increment dst ← src Autoincrement src address R ← R − 1
	OTIR* OTIRB*	dst, src, R	IR	11+10n	Output, Increment and Repeat dst ← src Autoincrement src address R ← R − 1 Repeat until R = 0
Special Input (identical to Input but different states code)	SIN* SINB*	R, src	DA	12	Special Input R ← src
	SIND* SINDB*	dst, src, R	IR	21	Special Input and Decrement dst ← src Autodrecrement dst address R ← R − 1
	SINDR* SINDRB*	dst, src, R	IR	11+10n	Special Input, Decr. and Repeat dst ← src Autodecrement dst address R ← R − 1 Repeat until R = 0
	SINI* SINIB*	dst, src, R	IR	21	Special Input and Increment dst ← src Autoincrement dst address R ← R − 1
	SINIR* SINIRB*	dst, src, R	IR	11+10n	Special Input, Incr. and Repeat dst ← src Autoincrement dst address R ← R − 1 Repeat until R = 0

INPUT/OUTPUT (Cont.)

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Special Output (identical to output, but different status code)	SOUT* SOUTB*	dst, src	DA	12	Special Output dst ← src
	SOUTD* SOUTDB*	dst, src R	IR	21	Special Output and Decrement dst ← src Autodecrement src address R ← R − 1
	SOTDR* SOTDRB*	dst, src, R	IR	11+10n	Special Output, Decr. and Repeat dst ← src Autodecrement src address R ← R − 1 Repeat until R = 0
	SOUTI* SOUTIB*	dst, src, R	IR	21	Special Output and Increment dst ← src Autoincrement src address R ← R − 1
	SOTIR* SOTIRB*	dst, src R	R	11+10n	Special Output, Incr. and Repeat dst ← src Autoincrement src address R ← R − 1 Repeat until R = 0

CPU CONTROL

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Interrupts	DI*	int	–	7	Disable Interrupt (Any combination of NVI, VI)
	EI*	int	–	7	Enable Interrupt (Any combination of NVI, VI)
Halt	HALT*	–	–	8+3n	HALT
Control Words	LDCTL*	CTLR, src	R	7	Load into Control Register CTLR ← src
	LDCTL*	dst, CTLR	R	7	Load into Control Register dst ← CTLR
	LDCTLB	dst, FLGR	R	7	Load into Flag Byte Register FLGR ← src
	LDPS*	src	IR DA X	12 16 17	Load Program Status PS ← src
Multi Micro	MBIT*	–	–	7	Test Multi-Micro Bit Set S if μI is High; reset S if μI is Low
	MREQ*	dst	R	12+7n	Multi-Micro Request
	MRES*	–	–	5	Multi-Micro Reset
	MSET*	–		5	Multi-Micro Set
NOP	NOP			7	No Operation
Flags	RESFLG	flag		7 7	Reset Flag (Any combination of C, Z, S, P/V)
	SETFLG	flag		7	Set Flag (Any combination of C, Z, S, P/V)
	COMFLG	flags	–	7	Complement Flag (Any combination of C, Z, S, P/V)

*Privileged instructions. Executed in system mode only.

†Clock cycles for byte or word data, non-segmented addresses. Segmented addresses may require 2 to 4 additional cycles. Some long word data require more cycles. See AmZ8000 Users Manual for exact data.

PROGRAM CONTROL

	Mnemonics	Operands	Addr. Modes	Clock Cycles†	Operation
Call	CALL	dst	IR DA X	10 12 13	Call Subroutine Autodecrement SP @ SP ← PC PC ← dst
	CALR	dst	RA	10	Call Relative Autodecrement SP @ SP ← PC PC ← PC + dst (range −4094 to +4096)
	SC	src	IM	33	System Call Autodecrement SP @ SP ← old PS Push instruction PS ← System Call PS
Jump	DJNZ DBJNZ	R, dst	RA	11	Decrement and Jump if Non-Zero R ← R − 1 IF R = 0: PC ← PC + dst (range −254 to 0)
	IRET*	–	–	13	Interrupt Return PS ← @ SP Autoincrement SP
	JP	cc, dst	IR – DA X	7 – 7 8	Jump Conditional If cc is true: PC ← dst
	JR	cc, dst	RA	6	Jump Conditional Relative If cc is true: PC ← PC + dst (range −256 to +254
	RET	cc	–	10	Return Conditional If cc is true: PC ← @ SP Autodecrement SP
Return	IRET*	–	–	13	Interrupt Return PS ← @ SP Autoincrement SP

*Privileged instructions. Executed in system mode only.

†Clock cycles for byte or word data, non-segmented addresses. Segmented addresses may require 2 to 4 additional cycles. Some long word data require more cycles. See AmZ8000 Users Manual for exact data.

INTERFACE SIGNAL DESCRIPTION

V_{CC}: +5V Power Supply

V_{SS}: Ground

AD0-AD15: Address/Data Bus (Bidirectional, 3-State)

This 16-bit multiplexed address/data bus is used for all I/O and memory transactions. HIGH on the bus corresponds to 1 and LOW corresponds to 0. AD0 is the least significant bit position with AD15 is most significant. The $\overline{AS}$ output and $\overline{DS}$ output will indicate whether the bus is used for address offset or data. The status output lines ST0-ST3 will indicate the type of transaction; memory or I/O.

$\overline{AS}$: Address Strobe (Output, 3-State)

LOW on this output indicates that the AD0-AD15 bus contains address information. The address information is stable by the time of the LOW-to-HIGH transition of the $\overline{AS}$ output (see timing diagrams). The status outputs ST0-ST3 indicate whether the bus contains a memory address or I/O address.

$\overline{DS}$: Data Strobe (Output, 3-State)

LOW on this output indicates that the AD0-AD15 bus is being used for data transfer. The R/$\overline{W}$ output indicates the direction of data transfer – read (or in) means data into the CPU and write (or out) means data from the CPU. During a read operation, data can be gated on to the bus when $\overline{DS}$ goes LOW. A LOW-to-HIGH transition on the $\overline{DS}$ output indicates that the CPU has accepted the data (see timing diagram). During a write operation, LOW on the $\overline{DS}$ output indicates that data is setup on the bus. Data will be removed sometime after the LOW-to-HIGH transition of the $\overline{DS}$ output (see timing diagram).

R/$\overline{W}$: Read/Write (Output, 3-State)

This output indicates the direction of data flow on the AD0-AD15 bus. HIGH indicates a read operation, i.e., data into the CPU and LOW indicates a write operation, i.e., data from the CPU. This output is activated at the same time as $\overline{AS}$ going LOW and remains stable for the duration of the whole transaction (see timing diagram).

B/$\overline{W}$: Byte/Word (Output, 3-State)

This output indicates the type of data transferred on the AD0-AD15 bus. HIGH indicates byte (8-bit) and LOW indicates word (16-bit) transfer. This output is activated at the same stage as $\overline{AS}$ going LOW and remains valid for the duration of the whole transaction (see timing diagram). The address generated by the CPU is always a byte address. However, the memory is organized as 16-bit words. All instructions and word operands are word aligned and are addressed by even addresses. Thus, for all word transactions with the memory the least significant address bit will be zero. When addressing the memory for byte transactions, the least significant address bit determines which byte of the memory word is needed; even address specifies the most significant byte and odd address specifies the least significant byte. In the case of I/O transactions, the address information on the AD0-AD15 bus refers to an I/O port and B/$\overline{W}$ determines whether a data word or data byte will be transacted. During I/O byte transactions, the least significant address bit A0 determines which half of the AD0-AD15 bus will be used for the I/O transactions. The ST0-ST3 outputs will indicate whether the current transaction is for memory, normal I/O or special I/O.

ST0-ST3: Status (Outputs, 3-State)

These four outputs contain information regarding the current transaction in a coded form. The status line codes are shown in the following table:

ST3	ST2	ST1	ST0	
L	L	L	L	Internal Operation
L	L	L	H	Memory Refresh
L	L	H	L	Normal I/O Transaction
L	L	H	H	Special I/O Transaction
L	H	L	L	Segment Trap Acknowledge
L	H	L	H	Non-Maskable Interrupt Acknowledge
L	H	H	L	Non-Vectored Interrupt Acknowledge
L	H	H	H	Vectored Interrupt Acknowledge
H	L	L	L	Memory Transaction for Operand
H	L	L	H	Memory Transaction for Stack
H	L	H	L	Reserved
H	L	H	H	Reserved
H	H	L	L	Memory Transaction for Instruction Fetch (Subsequent Word)
H	H	L	H	Memory Transaction for Instruction Fetch (First Word)
H	H	H	L	Reserved
H	H	H	H	Reserved

$\overline{WAIT}$: Wait (Input)

LOW on this input indicates to the CPU that memory or I/O is not ready for the data transfer and hence the current transaction should be stretched. The $\overline{WAIT}$ input is sampled by the CPU at certain instances during the transaction (see timing diagram). If $\overline{WAIT}$ input is LOW at these instances, the CPU will go into wait state to prolong the transaction. The wait state will repeat until the $\overline{WAIT}$ input is HIGH at the sampling instant.

N/$\overline{S}$: Normal/System Mode (Output, 3-State)

HIGH on this output indicates that the CPU is operating in Normal Mode and LOW indicates operation in System Mode. This output is derived from the Flag Control Word (FCW) register. The FCW register is described under the program status information section of this document.

$\overline{MREQ}$: Memory Request (Output, 3-State)

LOW on this output indicates that a CPU transaction with memory is taking place.

$\overline{BUSRQ}$: Bus Request (Input)

LOW on this input indicates to the CPU that another device (such as DMA) is requesting to take control of the bus. The $\overline{BUSRQ}$ input can be driven LOW anytime. The CPU synchronizes this input internally. The CPU responds by activating $\overline{BUSAK}$ output LOW to indicate that the bus has been relinquished. Relinquishing the bus means that the AD0-AD15, $\overline{AS}$, $\overline{DS}$, B/$\overline{W}$, R/$\overline{W}$, N/$\overline{S}$, ST0-ST3, SN0-SN6 and $\overline{MREQ}$ outputs will be in the high impedance state. The requesting device should control these lines in an identical fashion to the CPU to accomplish transactions. The $\overline{BUSRQ}$ input must remain LOW as long as needed to perform all the transactions and the CPU will keep the $\overline{BUSAK}$ output LOW. After completing the transactions, the device must disable the AD0-AD15, $\overline{AS}$, $\overline{DS}$, B/$\overline{W}$, R/$\overline{W}$, N/$\overline{S}$, ST0-ST3, SN0-SN6 and $\overline{MREQ}$ into the high impedance state and stop driving the $\overline{BUSRQ}$ input LOW. The CPU will make $\overline{BUSAK}$ output HIGH sometime later and take back the bus control.

$\overline{BUSAK}$: Bus Acknowledge (Output)

LOW on this output indicates that the CPU has relinquished the bus in response to a bus request.

$\overline{NMI}$: Non-Maskable Interrupt (Input)

HIGH to LOW transition on this input constitutes non-maskable interrupt request. The CPU will respond with the Non-maskable Interrupt Acknowledge on the ST0-ST3 outputs and will enter an interrupt sequence. The transition on the $\overline{NMI}$ can occur anytime. Of the three kinds of interrupts available, the non-maskable interrupt has the highest priority.

$\overline{VI}$: Vectored Interrupt (Input)

LOW on this input constitutes vectored interrupt request. Vectored interrupt is next lower to the non-maskable interrupt in priority. The VIE bit in the Flag and Control Word register must be 1 for the vectored interrupt to be honored. The CPU will respond with Vectored Interrupt Acknowledge code on the ST0-ST3 outputs and will begin the interrupt sequence. The $\overline{VI}$ input can be driven LOW anytime and should be held LOW until acknowledged.

$\overline{NVI}$: Non-Vectored Interrupt (Input)

LOW on this input constitutes non-vectored interrupt request. Non-vectored has the lowest priority of the three types of interrupts. The NVIE bit in the Flag and Control Word register must be 1 for this request to be honored. The CPU will respond with Non-Vectored Interrupt Acknowledge code on the ST0-ST3 outputs and will begin the interrupt sequence. The $\overline{NVI}$ input can be driven LOW anytime and should be held LOW until acknowledged.

$\overline{\mu I}$: Micro-In (Input)

This input participates in the resource request daisy chain. See the section on multimicroprocessor support facilities in this document.

$\overline{\mu O}$: Micro-Out (Output)

This output participates in the resource request daisy chain. See the section on multimicroprocessor support facilities in this document.

$\overline{RESET}$: Reset (Input)

LOW on this input initiates a reset sequence in the CPU. See the section on Initialization for details on reset sequence.

CLK: Clock (Input)

All CPU operations are controlled from the signal fed into this input. See DC characteristics for clock voltage level requirements.

$\overline{STOP}$: Stop (Input)

This active LOW input facilitates one instruction at a time operation. See the section on single stepping.

SN0-SN6: Segment Number (Outputs, 3-State) (AmZ8001 Only)

These seven outputs contain the segment number part of a memory address. A HIGH on the output corresponds to 1 and a LOW corresponds to 0. SN0 is the least significant bit position and SN6 is the most significant bit position.

$\overline{SEGT}$: Segment Trap (Input) (AmZ8001 Only)

LOW on this input constitutes a segment trap request. If the line is driven LOW, the CPU will respond with the Segment Trap Acknowledge code on the Status lines, and commence a trap sequence. The $\overline{SEGT}$ input may be driven LOW at any time and is customarily held LOW until acknowledged. This input has priority over the interrupts.

MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65 to +150°C
Voltage at any Pin Relative to V_{SS}	−0.5 to +7.0V
Power Dissipation	2.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE (over which the DC, switching and functional specifications apply)

	4MHz	6MHz	8MHz
	Z800X	**Z800XA**	**Z800XA-8**
Commercial Operating Range T_A[1] = 0 to 70°C V_{CC} = 5V ± 5%	Z800XDC[2] Z800XPC Z800XLC	Z800XADC Z800XAPC Z800XALC	Z800XA-8DC Z800XA-8PC Z800XA-8LC
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ± 10%	Z800XDI Z800XLI	Z800XADI Z800XALI	
Military Operating Range T_A = −55 to +125°C	Z800XDMB Z800XLMB	Z800XADMB Z800XALMB	

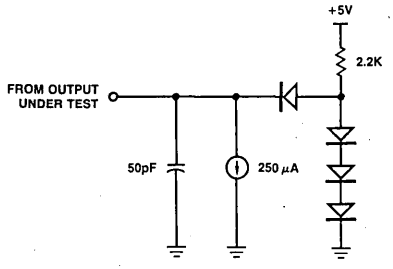
Notes: 1. T_A = Ambient Temperature
2. Add suffix B to indicate burn-in requirement

Standard Test Conditions

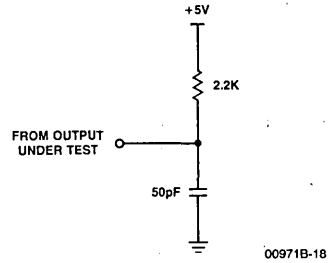
The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

$+4.75V \leq V_{CC} \leq +5.25V$
GND = 0V
$0°C \leq T_A \leq +70°C$

Standard Test Load



Open Drain Test Load



4, 6MHz Devices

ELECTRICAL CHARACTERISTICS over operating range unless otherwise specified

Parameter	Description	Test Conditions		Min	Max	Units
V_{CH}	Clock Input High Voltage	Driven by External Clock Generator		$V_{CC}-0.4$	$V_{CC}+0.3$	Volts
V_{CL}	Clock Input Low Voltage	Driven by External Clock Generator		−0.3	0.45	Volts
V_{IH}	Input High Voltage			2.0	$V_{CC}+0.3$	Volts
V_{IH} $\overline{NMI}$, Reset	Input High Voltage			2.4	$V_{CC}+0.3$	Volts
V_{IL}	Input Low Voltage			−0.3	0.8	Volts
V_{OH}	Output High Voltage	$I_{OH} = -250\mu A$		2.4		Volts
V_{OL}	Output Low Voltage	$I_{OL} = +2.0mA$			0.4	Volts
I_{IL}	Input Leakage	$0.4 \leq V_{IN} \leq +2.4V$			±10	μA
I_{OL}	Output Leakage	$0.4 \leq V_{OUT} \leq +2.4V$			±10	μA
I_{CC}	V_{CC} Supply Current	Commercial			300	mA
		Military	125°C		235	mA
			−55°C		350	mA

8MHz Devices

ELECTRICAL CHARACTERISTICS over operating range unless otherwise specified

Parameter	Description	Test Conditions	Min	Max	Units
V_{CH}	Clock Input High Voltage	Driven by External Clock Generator	$V_{CC}-0.4$	$V_{CC}+0.3$	Volts
V_{CL}	Clock Input Low Voltage	Driven by External Clock Generator	−0.3	0.45	Volts
V_{IH}	Input High Voltage		2.0	$V_{CC}+0.3$	Volts
V_{IH} $\overline{NMI}$, Reset	Input High Voltage		2.4	$V_{CC}+0.3$	Volts
V_{IL}	Input Low Voltage		−0.3	0.8	Volts
V_{OH}	Output High Voltage	$I_{OH} = -250\mu A$	2.4		Volts
V_{OL}	Output Low Voltage	$I_{OL} = +2.0mA$		0.4	Volts
I_{IL}	Input Leakage	$0.4 \leq V_{IN} \leq +2.4V$		±10	μA
I_{OL}	Output Leakage	$0.4 \leq V_{OUT} \leq +2.4V$		±10	μA
I_{CC}	V_{CC} Supply Current	Commercial		325	mA

SWITCHING CHARACTERISTICS over operating range

Number	Parameter	Description	4MHz Devices COML/IND/MIL Min	4MHz Devices COML/IND/MIL Max	6MHz Devices COML/IND/MIL Min	6MHz Devices COML/IND/MIL Max	Units
1	TcC	Clock Cycle Time	250	2000	165	2000	ns
2	TwCh	Clock Width (High)	105	2000	70	2000	ns
3	TwCl	Clock Width (Low)	105	2000	70	2000	ns
4	TfC	Clock Fall Time		20		10	ns
5	TrC	Clock Rise Time		20		10	ns
6†	TdC(SNv)	Clock ↑ to Segment Number Valid (50pF Load)		130		110	ns
7†	TdC(SNn)	Clock ↑ to Segment Number Not Valid	20		10		ns
8	TdC(Bz)	Clock ↑ to Bus Float		65		55	ns
9	TdC(A)	Clock ↑ to Address Valid		100		75	ns
10	TdC(Az)	Clock ↑ to Address Float		65		55	ns
11	TdA(DI)*	Address Valid to Data In Required Valid		475		305	ns
12	TsDI(C)	Data In to Clock ↓ Set-up Time	30		20		ns
13	TdDS(A)*	$\overline{DS}$ ↑ to Address Active	80		70		ns
14	TdC(DO)	Clock ↑ to Data Out Valid		100		75	ns
15	ThDI(DS)	Data In to $\overline{DS}$ ↑ Hold Time	0		0		ns
16	TdDO(DS)*	Data Out Valid to $\overline{DS}$ ↑ Delay	295		195		ns
17	TdA(MR)*	Address Valid to $\overline{MREQ}$ ↓ Delay	55		35		ns
18	TdC(MR)	Clock ↓ to $\overline{MREQ}$ ↓ Delay		80		70	ns
19	TwMRh*	$\overline{MREQ}$ Width (High)	210		135		ns
20	TdMR(A)*	$\overline{MREQ}$ ↓ to Address Not Active	70		35		ns
21	TdDO(DSW)*	Data Out Valid to $\overline{DS}$ ↓ (Write) Delay	55		35		ns
22	TdMR(DI)*	$\overline{MREQ}$ ↓ to Data In Required Valid	375		225		ns
23	TdC(MR)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		80		60	ns
24	TdC(ASf)	Clock ↑ to $\overline{AS}$ ↓ Delay		80		60	ns
25	TdA(AS)*	Address Valid to $\overline{AS}$ ↑ Delay	55		35		ns
26	TdC(ASr)	Clock ↓ to $\overline{AS}$ ↑ Delay		90		80	ns
27	TdAS(DI)*	$\overline{AS}$ ↑ to Data In Required Valid	360		215		ns
28	TdDS(AS)*	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	70		35		ns
29	TwAS*	$\overline{AS}$ Width (Low)	85		55		ns
30	TdAS(A)*	$\overline{AS}$ ↑ to Address Not Active Delay	70		30		ns
31	TdAz(DSR)	Address Float to $\overline{DS}$ (Read) ↓ Delay	0		0		ns
32	TdAS(DSR)*	$\overline{AS}$ ↑ to $\overline{DS}$ (Read) ↓ Delay	80		35		ns
33	TdDSR(DI)*	$\overline{DS}$ (Read) ↓ to Data In Required Valid	205		130		ns
34	TdC(DSr)	Clock ↓ to $\overline{DS}$ ↑ Delay		70		65	ns
35	TdDS(DO)*	$\overline{DS}$ ↑ to Data Out and STATUS Not Valid	75		45		ns
36	TdA(DSR)*	Address Valid to $\overline{DS}$ (Read) ↓ Delay	180		110		ns
37	TdC(DSR)	Clock ↑ to $\overline{DS}$ (Read) ↓ Delay		120		85	ns
38	TwDSR*	$\overline{DS}$ (Read) Width (Low)	275		185		ns
39	TdC(DSW)	Clock ↓ to $\overline{DS}$ (Write) ↓ Delay		95		80	ns
40	TwDSW*	$\overline{DS}$ (Write) Width (Low)	185		110		ns
41	TdDSI(DI)*	$\overline{DS}$ (Input) ↓ to Data In Required Valid	330		200		ns
42	TdC(DSf)	Clock ↓ to $\overline{DS}$ (I/O) ↓ Delay		120		100	ns
43	TwDS*	$\overline{DS}$ (I/O) Width (Low)	410		255		ns
44	TdAS(DSA)*	$\overline{AS}$ ↑ to $\overline{DS}$ (Acknowledge) ↓ Delay	1065		690		ns
45	TdC(DSA)	Clock ↑ to $\overline{DS}$ (Acknowledge) ↓ Delay		120		85	ns
46	TdDSA(DI)*	$\overline{DS}$ (Acknowledge) ↓ to Data In Required Delay	455		295		ns

†Z8001 and Z8001A only.

SWITCHING CHARACTERISTICS (Cont.)

			4MHz Devices		6MHz Devices		
			COML/IND/MIL		COML/IND/MIL		
Number	Parameter	Description	Min	Max	Min	Max	Units
47	TdC(S)	Clock ↑ to Status Valid Delay		110		85	ns
48	TdS(AS)*	Status Valid to AS ↑ Delay	50		30		ns
49	TsR(C)	$\overline{\text{RESET}}$ to Clock ↑ Set-up Time	180		70		ns
50	ThR(C)	$\overline{\text{RESET}}$ to Clock ↑ Hold Time	0		0		ns
51	TwNMI	$\overline{\text{NMI}}$ Width (Low)	100		70		ns
52	TsNMI(C)	$\overline{\text{NMI}}$ to Clock ↑ Set-up Time	140		70		ns
53	TsVI(C)	$\overline{\text{VI}}$, $\overline{\text{NVI}}$ to Clock ↑ Set-up Time	110		50		ns
54	ThVI(C)	$\overline{\text{VI}}$, $\overline{\text{NVI}}$ to Clock ↑ Hold Time	20		20		ns
55†	TsSGT(C)	$\overline{\text{SEGT}}$ to Clock ↑ Set-up Time	70		55		ns
56†	ThSGT(C)	$\overline{\text{SEGT}}$ to Clock ↑ Hold Time	0		0		ns
57	TsMI(C)	$\overline{\text{MI}}$ to Clock ↑ Set-up Time	180		110		ns
58	ThMI(C)	$\overline{\text{MI}}$ to Clock ↑ Hold Time	0		0		ns
59	TdC(MO)	Clock ↑ to $\overline{\text{MO}}$ Delay		120		85	ns
60	TsSTP(C)	$\overline{\text{STOP}}$ to Clock ↓ Set-up Time	140		80		ns
61	ThSTP(C)	$\overline{\text{STOP}}$ to Clock ↓ Hold Time	0		0		ns
62	TsWT(C)	$\overline{\text{WAIT}}$ to Clock ↓ Set-up Time	50		30		ns
63	ThWT(C)	$\overline{\text{WAIT}}$ to Clock ↓ Hold Time	10		10		ns
64	TsBRQ(C)	$\overline{\text{BUSRQ}}$ to Clock ↑ Set-up Time	90		80		ns
65	ThBRQ(C)	$\overline{\text{BUSRQ}}$ to Clock ↑ Hold Time	10		10		ns
66	TdC(BAKr)	Clock ↑ to $\overline{\text{BUSAK}}$ ↑ Delay		100		75	ns
67	TdC(BAKf)	Clock ↑ to $\overline{\text{BUSAK}}$ ↓ Delay		100		75	ns
68	TwA	Address Valid Width	150		95		ns
69	TdDS(S)	$\overline{\text{DS}}$ ↑ to STATUS Not Valid	80		55		ns

*Clock-cycle-time-dependent characteristics. These numbers are computed assuming the clock characteristics are at the limits given in parameters 1 through 5. For other clock frequencies these parameters can be derived from other specs and the clock characteristics. See table on next two pages.

†Z8001 and Z8001A only.

CLOCK-CYCLE-TIME-DEPENDENT CHARACTERISTICS

The parameters listed below are also shown in the switching specification. However they are dependent on the actual values of the clock periods. The equations below define that dependence, so the correct limit for these parameters may be determined for any system, regardless of the actual clock characteristics.

Number	Symbol	4MHz Devices	6MHz Devices
11	TdA(DI)	2TcC + TwCh − 125ns	2TcC + TwCh − 95ns
13	TdDS(C)	TwCl − 25ns	TWCl − 30ns
16	TdDO(DS)	TcC + TwCh − 60ns	TcC + TwCh − 40ns
17	TdA(MR)	TwCh − 50ns	TwCh − 35ns
19	TwMRh	TcC − 40ns	TcC − 30ns
20	TdMR(A)	TwCl − 35ns	TwCl − 35ns
21	TdDO(DSW)	TwCh − 50ns	TwCh − 35ns
22	TdMR(DI)	2TcC − 125ns	2TcC − 105ns
25	TdA(AS)	TwCh − 50ns	TwCh − 35ns
27	TdAS(DI)	2TcC − 140ns	2TcC − 115ns
28	TdDS(AS)	TwCl − 35ns	TwCl − 35ns
29	TwAS	TwCh − 20ns	TwCh − 15ns
30	TdAS(A)	TwCl − 35ns	TwCl − 40ns
32	TdAS(DSR)	TwCl − 25ns	TwCl − 35ns
33	TdDSR(DI)	TcC + TwCh − 150ns	TcC + TwCh − 105ns
35	TdDS(DO)	TwCl − 30ns	TwCl − 25ns
36	TdA(DSR)	TcC − 70ns	TcC − 55ns
38	TwDSR	TcC + TwCh − 80ns	TcC + TwCh − 50ns
40	TwDSW	TcC − 65ns	TcC − 55ns
41	TdDSI(DI)	2TcC − 170ns	2TcC − 130ns
43	TwDS	2TcC − 90ns	2TcC − 75ns
44	TdAS(DSA)	4TcC + TwCl − 40ns	4TcC + TwCl − 40ns
46	TdDSA(DI)	2TcC + TwCh − 150ns	2TcC + TwCh − 105ns
48	TdS(AS)	TwCh − 55ns	TwCh − 40ns
68	TwA	TcC − 90ns	TcC − 70ns
69	TdDS(S)	TwCl − 25ns	TwCl − 15ns

SWITCHING CHARACTERISTICS over operating range

			8MHz Devices		
			COML		
Number	Parameter	Description	Min	Max	Units
1	TcC	Clock Cycle Time	125	2000	ns
2	TwCh	Clock Width (High) (See Note 2)	55	2000	ns
3	TwCl	Clock Width (Low) (See Note 2)	55	2000	ns
4	TfC	Clock Fall Time (See Note 3)		10	ns
5	TrC	Clock Rise Time (See Note 3)		10	ns
6†	TdC(SNv)	Clock ↑ to Segment Number Valid (50pF Load)		100	ns
7†	TdC(SNn)	Clock ↑ to Segment Number Not Valid	10		ns
8	TdC(Bz)	Clock ↑ to Bus Float		50	ns
9	TdC(A)	Clock ↑ to Address Valid		65	ns
10	TdC(Az)	Clock ↑ to Address Float		45	ns
11	TdA(DI)*	Address Valid to Data In Required Valid		225	ns
12	TsDI(C)	Data In to Clock ↓ Set-up Time	15		ns
13	TdDS(A)*	$\overline{DS}$ ↑ to Address Active	40		ns
14	TdC(DO)	Clock ↑ to Data Out Valid		65	ns
15	ThDI(DS)	Data In to $\overline{DS}$ ↑ Hold Time	0		ns
16	TdDO(DS)*	Data Out Valid to $\overline{DS}$ ↑ Delay	150		ns
17	TdA(MR)*	Address Valid to $\overline{MREQ}$ ↓ Delay	30		ns
18	TdC(MR)	Clock ↓ to $\overline{MREQ}$ ↓ Delay		55	ns
19	TwMRh*	$\overline{MREQ}$ Width (High)	105		ns
20	TdMR(A)*	$\overline{MREQ}$ ↓ to Address Not Active	35		ns
21	TdDO(DSW)*	Data Out Valid to $\overline{DS}$ ↓ (Write) Delay	30		ns
22	TdMR(DI)*	$\overline{MREQ}$ ↓ to Data In Required Valid	175		ns
23	TdC(MR)	Clock ↓ to $\overline{MREQ}$ ↑ Delay		55	ns
24	TdC(ASf)	Clock ↑ to $\overline{AS}$ ↓ Delay		55	ns
25	TdA(AS)*	Address Valid to $\overline{AS}$ ↑ Delay	30		ns
26	TdC(ASr)	Clock ↓ to $\overline{AS}$ ↑ Delay		65	ns
27	TdAS(DI)*	$\overline{AS}$ ↑ to Data In Required Valid	170		ns
28	TdDS(AS)*	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	35		ns
29	TwAS*	$\overline{AS}$ Width (Low)	45		ns
30	TdAS(A)*	$\overline{AS}$ ↑ to Address Not Active Delay	30		ns
31	TdAz(DSR)	Address Float to $\overline{DS}$ (Read) ↓ Delay	0		ns
32	TdAS(DSR)*	$\overline{AS}$ ↑ to $\overline{DS}$ (Read) ↓ Delay	30		ns
33	TdDSR(DI)*	$\overline{DS}$ (Read) ↓ to Data In Required Valid	115		ns
34	TdC(DSr)	Clock ↓ to $\overline{DS}$ ↑ Delay		65	ns
35	TdDS(DO)*	$\overline{DS}$ ↑ to Data Out and STATUS Not Valid	40		ns
36	TdA(DSR)*	Address Valid to $\overline{DS}$ (Read) ↓ Delay	85		ns
37	TdC(DSR)	Clock ↑ to $\overline{DS}$ (Read) ↓ Delay		70	ns
38	TwDSR*	$\overline{DS}$ (Read) Width (Low)	140		ns
39	TdC(DSW)	Clock ↓ to $\overline{DS}$ (Write) ↓ Delay		65	ns
40	TwDSW*	$\overline{DS}$ (Write) Width (Low)	85		ns
41	TdDSI(DI)*	$\overline{DS}$ (Input) ↓ to Data In Required Valid	135		ns
42	TdC(DSf)	Clock ↓ to $\overline{DS}$ (I/O) ↓ Delay		85	ns
43	TwDS*	$\overline{DS}$ (I/O) Width (Low)	200		ns
44	TdAS(DSA)*	$\overline{AS}$ ↑ to $\overline{DS}$ (Acknowledge) ↓ Delay	520		ns
45	TdC(DSA)	Clock ↑ to $\overline{DS}$ (Acknowledge) ↓ Delay		65	ns

†Z8001A-8 only.

SWITCHING CHARACTERISTICS (Cont.)

			8MHz Devices		
			COML		
Number	Parameter	Description	Min	Max	Units
46	TdDSA(DI)*	$\overline{DS}$ (Acknowledge) ↓ to Data In Required Delay	235		ns
47	TdC(S)	Clock ↑ to Status Valid Delay		75	ns
48	TdS(AS)*	Status Valid to $\overline{AS}$ ↑ Delay	25		ns
49	TsR(C)	$\overline{RESET}$ to Clock ↑ Set-up Time	70		ns
50	ThR(C)	$\overline{RESET}$ to Clock ↑ Hold Time	0		ns
51	TwNMI	$\overline{NMI}$ Width (Low)	50		ns
52	TsNMI(C)	$\overline{NMI}$ to Clock ↑ Set-up Time	70		ns
53	TsVI(C)	$\overline{VI}$, $\overline{NVI}$ to Clock ↑ Set-up Time	50		ns
54	ThVI(C)	$\overline{VI}$, $\overline{NVI}$ to Clock ↑ Hold Time	20		ns
55†	TsSGT(C)	$\overline{SEGT}$ to Clock ↑ Set-up Time	45		ns
56†	ThSGT(C)	$\overline{SEGT}$ to Clock ↑ Hold Time	0		ns
57	TsMI(C)	$\overline{MI}$ to Clock ↑ Set-up Time	90		ns
58	ThMI(C)	$\overline{MI}$ to Clock ↑ Hold Time	0		ns
59	TdC(MO)	Clock ↑ to $\overline{MO}$ Delay		65	ns
60	TsSTP(C)	$\overline{STOP}$ to Clock ↓ Set-up Time	75		ns
61	ThSTP(C)	$\overline{STOP}$ to Clock ↓ Hold Time	0		ns
62	TsWT(C)	$\overline{WAIT}$ to Clock ↓ Set-up Time	25		ns
63	ThWT(C)	$\overline{WAIT}$ to Clock ↓ Hold Time	10		ns
64	TsBRQ(C)	$\overline{BUSRQ}$ to Clock ↑ Set-up Time	60		ns
65	ThBRQ(C)	$\overline{BUSRQ}$ to Clock ↑ Hold Time	10		ns
66	TdC(BAKr)	Clock ↑ to $\overline{BUSAK}$ ↑ Delay		60	ns
67	TdC(BAKf)	Clock ↑ to $\overline{BUSAK}$ ↓ Delay		60	ns
68	TwA*	Address Valid Width	90		ns
69	TdDS(S)*	$\overline{DS}$ ↑ to STATUS Not Valid	45		ns

*Clock-cycle-time-dependent characteristics. These numbers are computed assuming the clock characteristics are at the limits given in parameters 1 through 5. For other clock frequencies these parameters can be derived from other specs and the clock characteristics. See following table.

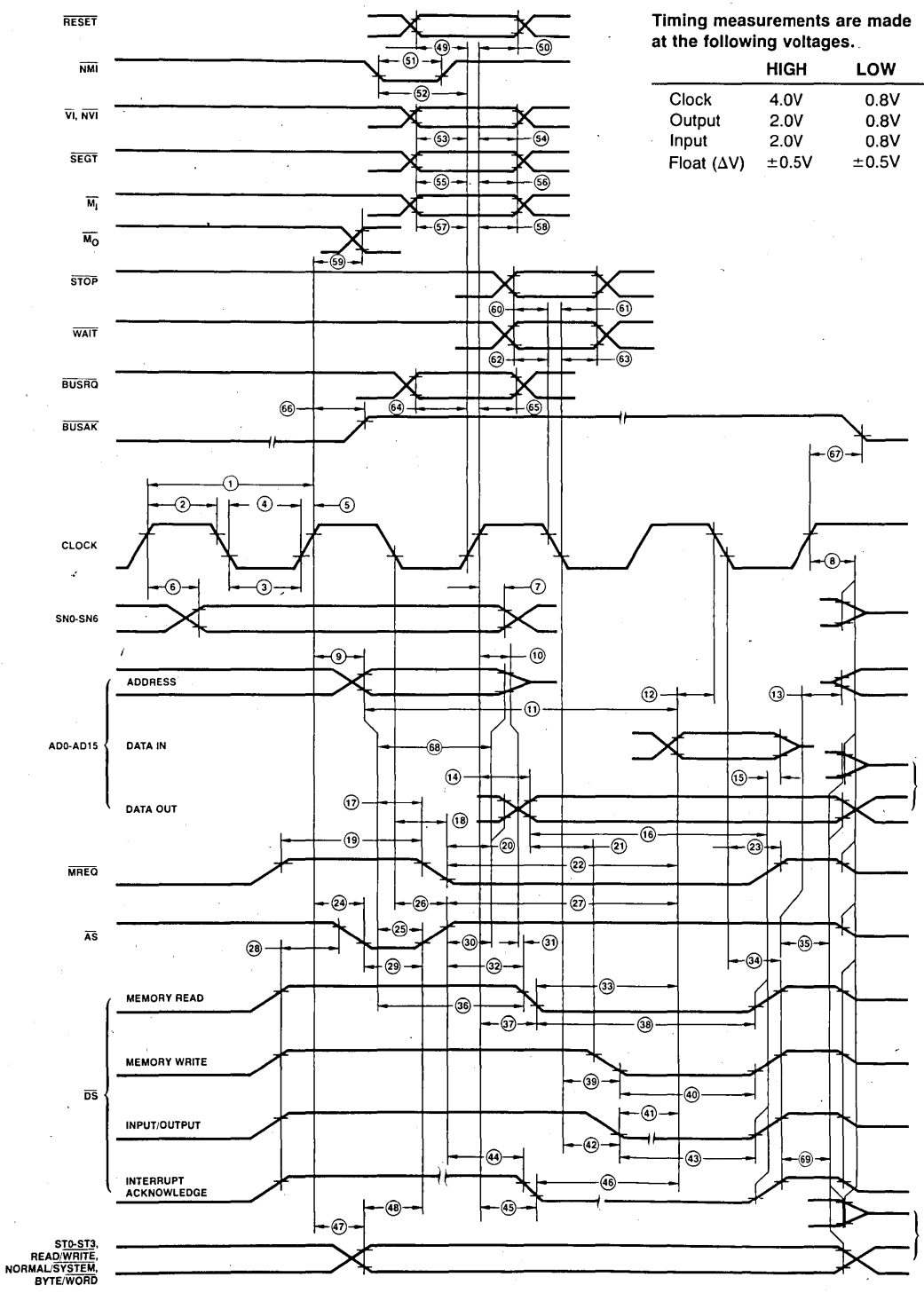
†Z8001A-8 only.

CLOCK-CYCLE-TIME-DEPENDENT CHARACTERISTICS

The parameters listed below are also shown in the switching specification. However they are dependent on the actual values of the clock periods. The equations below define that dependence, so the correct limit for these parameters may be determined for any system, regardless of the actual clock characteristics.

Number	Symbol	8MHz Devices
11	TdA(DI)	2TcC + TwCh – 80ns
13	TdDS(A)	TwCl – 15ns
16	TdDO(DS)	TcC + TwCh – 30ns
17	TdA(MR)	TwCh – 25ns
19	TwMRh	TcC – 20ns
20	TdMR(A)	TwCl – 20ns
21	TdDO(DSW)	TwCh – 25ns
22	TdMR(DI)	2TcC – 75ns
25	TdA(AS)	TwCh – 25ns
27	TdAS(DI)	2TcC – 80ns
28	TdDS(AS)	TwCl – 20ns
29	TwAS	TwCh – 10ns
30	TdAS(A)	TwCl – 25ns
32	TdAS(DSR)	TwCl – 25ns
33	TdDSR(DI)	TcC + TwCh – 65ns
35	TdDS(DO)	TwCl – 15ns
36	TdA(DSR)	TcC – 40ns
38	TwDSR	TcC + TwCh – 40ns
40	TwDSW	TcC – 40ns
41	TdDSI(DI)	2TcC – 115ns
43	TwDS	2TcC – 50ns
44	TdAS(DSA)	4TcC + TwCl – 35ns
46	TdDSA(DI)	2TcC + TwCh – 70ns
48	TdS(AS)	TwCh – 30ns
68	TwA	Tcc – 35ns
69	TdDS(S)	TwCl – 10ns

Z8001 TIMING DIAGRAM



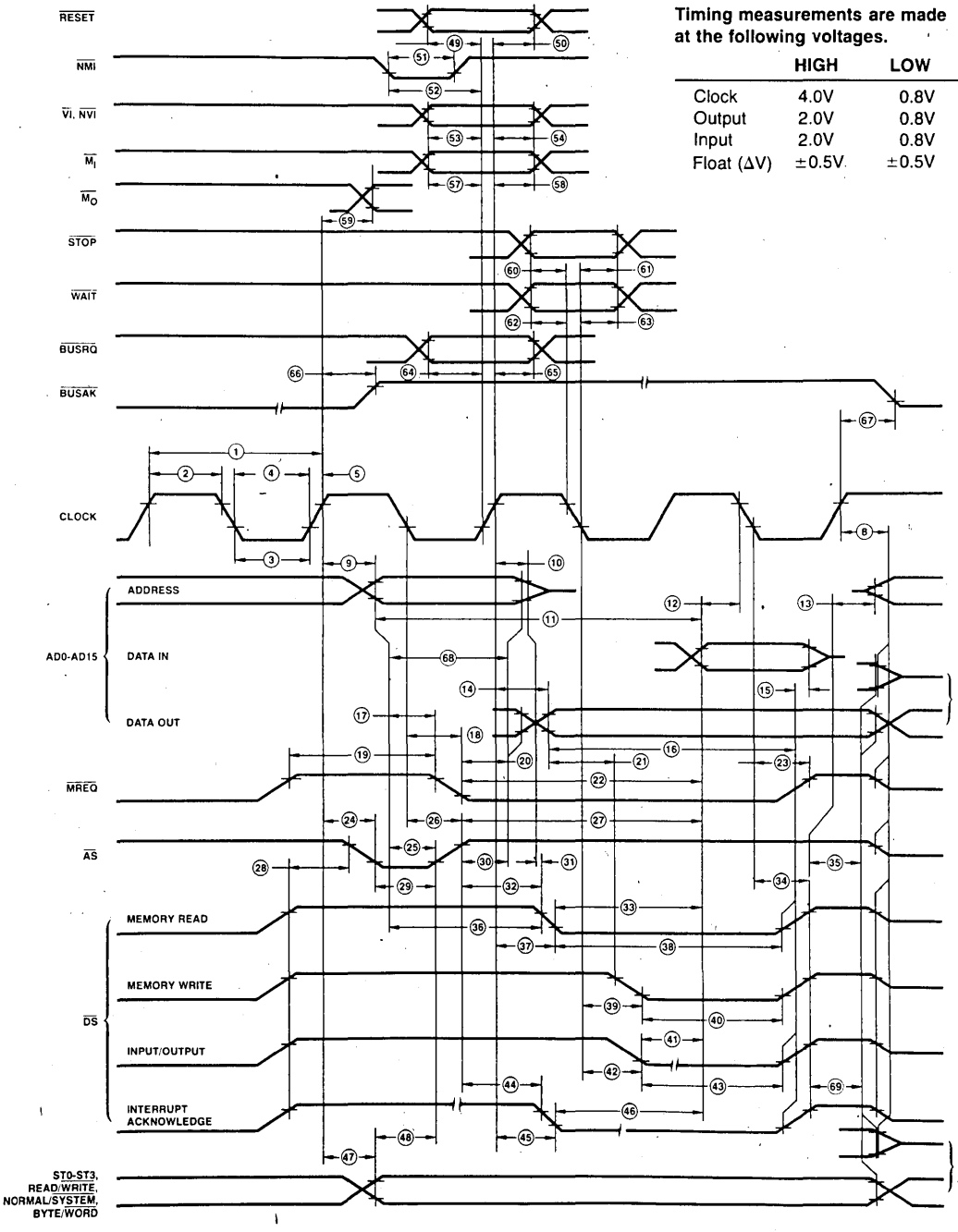
Timing measurements are made at the following voltages.

	HIGH	LOW
Clock	4.0V	0.8V
Output	2.0V	0.8V
Input	2.0V	0.8V
Float (ΔV)	±0.5V	±0.5V

This composite timing diagram does not show actual timing sequences. Refer to this diagram only for the detailed timing relationships of individual edges. Use the preceding illustrations as an explanation of the various timing sequences.

00971B-19

Z8002 TIMING DIAGRAM



Timing measurements are made at the following voltages.

	HIGH	LOW
Clock	4.0V	0.8V
Output	2.0V	0.8V
Input	2.0V	0.8V
Float (ΔV)	±0.5V	±0.5V

This composite timing diagram does not show actual timing sequences. Refer to this diagram only for the detailed timing relationships of individual edges. Use the preceding illustrations as an explanation of the various timing sequences.

00971B-20

AmZ8016
DMA Transfer Controller

DISTINCTIVE CHARACTERISTICS

- Two independent multi-function channels
- Transfer Modes: Single, demand dedicated with bus hold, demand dedicated with bus release, demand interleave
- Data types: Byte-to-Byte, Word-to-Word, Byte/Word funneling
- 8 Megabyte logical addressing range and 16 Megabyte physical addressing range in each address space
- Address increment, decrement or hold
- Automatic loading/reloading of control parameters by each channel
- Optional automatic chaining of operations
- Channel interleave operations
- Interleave operations with system bus
- Base registers for efficient repetitive operations
- Reload word table for efficient channel initialization
- Masked data pattern matching for search operations
- Vectored interrupts on selected transfer conditions
- Software DMA request
- Software or hardware controller wait state insertion
- Memory/peripheral transfer up to 2.66 Megabyte/second at 4MHz and 4 Megabyte/second at 6MHz
- Memory/memory transfer up to 1.33 Megabyte/second at 4MHz and 2 Megabyte/second at 6MHz

AmZ8016 DTC BLOCK DIAGRAM

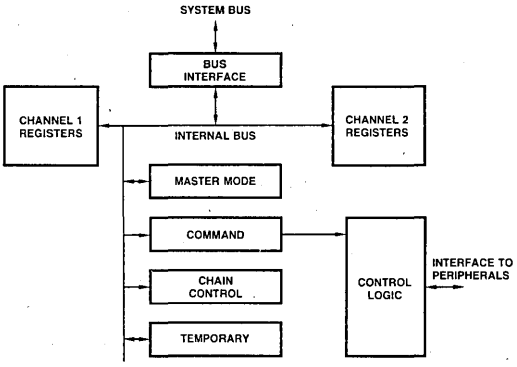


Figure 1.

AIZ-013

GENERAL DESCRIPTION

The AmZ8016* DMA Transfer Controller (DTC) is a high performance peripheral interface circuit for AmZ8000 processor systems. In addition to providing data block transfer capability between memory and peripherals, each of the DTC's two channels can perform peripheral-to-peripheral as well as memory-to-memory transfer. A special Search Mode of Operation compares data read from a memory or peripheral source to the content of a pattern register.

For all DMA operations (search, transfer, and transfer-and-search), the DTC can operate with either byte or word data sizes. In some system configurations it may be necessary to transfer between, word-organized memory and a byte-oriented peripheral. The DTC provides a byte packing/unpacking capability through its byte-word funneling transfer or transfer-and-search option. Some DMA applications may continuously transfer data between the same two locations. These applications may not require the flexibility inherent in reloading registers from memory tables. To service these repetitive DMA operations, base registers are provided on each channel which re-initialize the current source and destination Address and Operation Count registers. To change the data transfer direction under CPU control, provision is made for reassigning the source address as a destination and the destination as a source, eliminating the need for actual reloading of these address registers.

Frequently DMA devices must interface to slow peripherals or slow memory. In addition to providing a hardware $\overline{\text{WAIT}}$ input, the AmZ8016 DTC allows the user to select independently, for both source and destination addresses, automatic insertion of 0, 1, 2 or 4 wait states. The user may even disable the $\overline{\text{WAIT}}$ input pin function altogether and use these software programmed wait states exclusively.

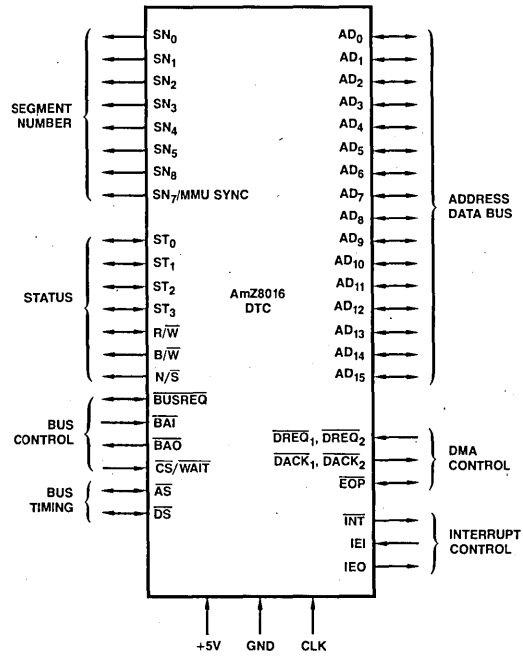
High throughput and powerful transfer options are of limited usefulness if a DMA requires frequent reloading by the host CPU. The AmZ8016 DTC minimizes CPU interactions by allowing each channel to load its control parameters from memory into the channel's control registers. The only action required of the CPU is to load the address of the control parameter table into the channel and issue an instruction to start this register loading operation.

OTHER LITERATURE

- AmZ8016 Application Manual (contains detailed application configuration and software example).

*Z8000 is a trademark of Zilog, Inc.

LOGIC SYMBOL



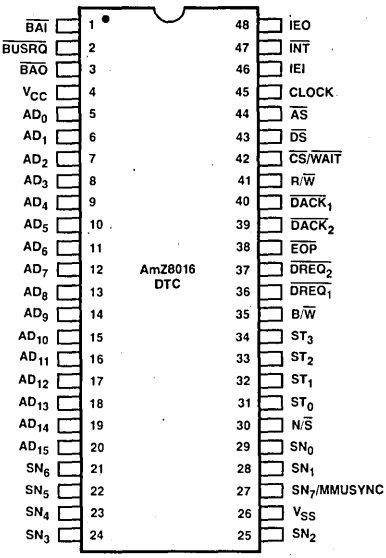
AIZ-014

CONNECTION DIAGRAM

Top View

D-48

P-48



Note: Pin 1 is marked for orientation.

AIZ-015

INTERFACE SIGNAL DESCRIPTION

All inputs to the DTC, except the Clock, are directly TTL compatible;

VCC: **+5V Power Supply**
VSS: **Ground**

CLOCK: (Clock, Input)

The Clock signal controls the internal operations and the rates of data transfers. It is usually derived from a master system clock or the associated CPU clock. The Clock input requires a high voltage input signal. When the DTC is used with an MMU, they must both be driven from the same clock signal. Many DTC input signals can make transitions independent of the DTC clock, these signals can be asynchronous to the DTC clock. On other signals, such as $\overline{WAIT}$ inputs, transitions must meet setup and hold requirements relative to the DTC clock. See the timing diagrams for details.

AD_0–AD_{15} (Address/Data Bus, Input/Output)
The Address/Data Bus is a time-multiplexed, bidirectional, active High, three-state bus used for all I/O and memory transactions. HIGH on the bus corresponds to 1 and LOW corresponds to 0. AD_0 is the least significant bit position and AD_{15} is the most significant. The presence of addresses is defined by the timing edge of $\overline{AS}$ and the asserted or requested presence of data is defined by the $\overline{DS}$ signal. The status output lines ST_0–ST_3 indicate the type of transaction, either memory or I/O. When the DTC is in control of the system bus, it dominates the AD Bus; when the DTC is not in control of the system bus, the CPU or other external devices dominate the AD Bus.

The presence of address or data on the AD_0–AD_{15} bus is defined only by $\overline{AS}$ and $\overline{DS}$. When the DTC is not in control of the bus; there is no required relation between the presence of address or data and the DTC clock. This allows the DTC to be used with a system bus which does not have a bussed clock signal.

$\overline{AS}$ (Address Strobe, Input/Output)
Address Strobe is a bidirectional, active-low, three-state signal. A LOW-to-HIGH transition on this signal while $\overline{DS}$ is high indicates that the AD_0–AD_{15} bus contains address information. During a DMA operation when the DTC is in control of the system, $\overline{AS}$ is an output generated by the DTC to indicate that a valid address is on AD_0–AD_{15}. The address information output by the DTC is stable prior to the LOW-to-HIGH $\overline{AS}$ transition. When the DTC is not in control of the system bus and the external system is transferring information to the DTC or from it, the DTC samples address information from the AD_0–AD_{15} bus on the LOW-to-HIGH $\overline{AS}$ transition. There are no timing requirements between $\overline{AS}$ as an input and the DTC clock; this allows use of the DTC with a system bus which does not have a bussed clock. If $\overline{AS}$ and $\overline{DS}$ are simultaneously low, the DTC is reset.

$\overline{DS}$ (Data Strobe, Input/Output)
Data Strobe is a bidirectional, active-low, three-state signal. A LOW on this signal while $\overline{AS}$ is high indicates that the AD_0–AD_{15} bus is being used for data transfer. When the DTC is not in control of the system bus and the external system is transferring information to or from the DTC, $\overline{DS}$ is a timing input used by the DTC to move data to or from the AD_0–AD_{15} bus. Data is written into the DTC by the external system on the LOW-to-HIGH $\overline{DS}$ transition.

Data is read from the DTC by the external system while $\overline{DS}$ is LOW. There are no timing requirements between $\overline{DS}$ as an input and the DTC clock; this allows use of the DTC with a system bus which does not have a bussed clock. During a DMA operation when the DTC is in control of the system, $\overline{DS}$ is an output generated by the DTC and used by the system to move data to or from the AD_0-AD_{15} bus. When the DTC has bus control, it writes to the external system by placing data on the AD_0-AD_{15} bus before the HIGH-to-LOW $\overline{DS}$ transition and holding the data stable until after the LOW-to-HIGH $\overline{DS}$ transition; while reading from the external system, the LOW-to-HIGH transition of $\overline{DS}$ latches data into the temporary register of the DTC (See timing diagram).

ST_0-ST_3 (Status, Input/Output)
The four Status lines are three-state, bidirectional signals containing coded information regarding the current bus transaction. When the DTC is not in control of the system bus ST_0-ST_3 are inputs and are used to detect interrupt and segment trap acknowledge cycles. There are no timing requirements between transitions on the ST_0-ST_3 input and the DTC clock; input transitions on ST_0-ST_3 are only defined relative to $\overline{AS}$ and $\overline{DS}$. When the DTC is in control of the system bus, the ST_0-ST_3 lines are outputs which indicate the type of memory or I/O transaction being performed. The status codes decoded and generated by the DTC are indicated in Figure 2, by the letters D and G respectively.

R/$\overline{W}$ (Read/$\overline{Write}$, Input/Output)
Read/Write is a bidirectional, three-state signal. Read polarity is HIGH and write polarity is LOW. R/$\overline{W}$ indicates the data direction of the current bus transaction, and is stable starting when $\overline{AS}$ goes LOW until the bus transaction ends (see timing diagram). When the DTC is not in control of the system bus and the external system is transferring information to or from the DTC, R/$\overline{W}$ is a status input used by the DTC to determine if data is entering or leaving on the AD_0-AD_{15} bus during $\overline{DS}$ time. In such a case, Read (HIGH) indicates that the system is requesting data from the DTC and Write (LOW) indicates that the system is presenting data to the DTC. There are no timing requirements between R/$\overline{W}$ as an input and the DTC clock; transitions on R/$\overline{W}$ as an input are only defined relative to $\overline{AS}$ and $\overline{DS}$. When the DTC is in control of the system bus, R/$\overline{W}$ is an output generated by the DTC, 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. Flyby DMA operations are a special case where R/$\overline{W}$ is valid for the normally addressed memory or peripheral locations and must be interpreted in reverse by the "Flyby" peripheral that uses it.

N/$\overline{S}$ (Normal/$\overline{System}$, Output, 3-State)
Normal/System is a three-state output activated only when the DTC is in control of the system bus. This signal is used to indicate which memory space is being accessed. The N/$\overline{S}$ pin is HIGH for normal memory and LOW for system memory. System space is always indicated for I/O cycles.

B/$\overline{W}$ (Byte/$\overline{Word}$, Output, 3-State)
This output indicates the type of data transferred on the AD bus. HIGH indicates a byte (8-bit) and LOW indicates a word (16-bit) transfer. This output is activated when $\overline{AS}$ goes LOW and remains valid for the duration of the whole transaction (see timing diagram). The address generated by the DTC is always a byte address, even though the memory is organized as 16-bit words. All word-sized data are word aligned and must be addressed by even addresses ($A_0 = 0$). When addressing byte transactions, the least significant address bit determines which byte is needed; an even address specifies the most significant byte (AD_8-AD_{15}) and an odd address specifies the least significant byte (AD_0-AD_7). (Note that the higher address specifies the less significant byte!) This addressing mechanism applies to memory accesses as well as I/O and special I/O accesses. When the DTC is a slave, it ignores the B/$\overline{W}$ signal.

ST3	ST2	ST1	ST0	Transaction/Operation	DTC Action (Note)
L	L	L	L	Internal Operation	
L	L	L	H	Memory Refresh	
L	L	H	L	I/O Transaction	G
L	L	H	H	Special I/O Transaction	G
L	H	L	L	Segment Trap Acknowledge	D
L	H	L	H	Non-Maskable Interrupt Acknowledge	D
L	H	H	L	Non-Vectored Interrupt Acknowledge	D
L	H	H	H	Vectored Interrupt Acknowledge	D
H	L	L	L	Memory Transaction for Data/DTC Chaining	G
H	L	L	H	Memory Transaction for Stack	G
H	L	H	L	Reserved	
H	L	H	H	Reserved	
H	H	L	L	Memory Transaction for Program Fetch (Subsequent Word)	G
H	H	L	H	Memory Transaction for Program Fetch (First Word)	
H	H	H	L	Reserved	
H	H	H	H	Reserved	

Note: D = Status code is decoded by DTC when not in control of system bus.
G = Status code is generated by DTC when in control of system bus.

Figure 2. Status Codes

$\overline{CS}/\overline{WAIT}$ (Chip Select/Wait, Input)
When the DTC is not in control of the system bus, this pin serves as an active-low Chip Select ($\overline{CS}$) input. A CPU or other external device uses $\overline{CS}$ to activate the DTC for reading and writing of its internal registers. $\overline{CS}$ may be held low for multiple transfers to and/or from the DTC provided $\overline{AS}$ and $\overline{DS}$ are toggled for each transfer. There are no timing requirements between the $\overline{CS}$ input and the DTC clock; the $\overline{CS}$ input timing requirements are only defined relative to $\overline{AS}$. When the DTC is in control of the system bus, this pin serves as an active-low $\overline{WAIT}$ input. Slow memories and peripheral devices may use $\overline{WAIT}$ to extend $\overline{DS}$ during bus transfers. Unlike the $\overline{CS}$ input, transitions on the $\overline{WAIT}$ input must meet certain timing requirements relative to the DTC clock. See the timing diagram for details. The Wait function may be disabled using a control bit in the Master Mode register, in which case the input is treated as an active LOW Chip Select only and is ignored when the DTC is in control of the system bus.

$\overline{BUSRQ}$ (Bus Request, Input/Output)
Bus Request is an active-low, open-drain, bidirectional signal used by the DTC to obtain control of the bus from the CPU. Before driving $\overline{BUSRQ}$ active, the DTC samples this line to insure that another request is not already being made by another device. $\overline{BUSRQ}$ lines from multiple devices are wire-ORed together externally with a common pull-up resistor of 1.8 kΩ or more. Since the DTC internally synchronizes the sampled $\overline{BUSRQ}$ signal, transitions on $\overline{BUSRQ}$ may be asynchronous to the DTC clock.

$\overline{BAI}$ (Bus Acknowledge In, Input)
$\overline{BAI}$ is an active-low asynchronous input indicating that the CPU has relinquished the bus and that no higher priority device has assumed bus control. Since $\overline{BAI}$ is internally synchronized by the DTC before being used, transitions on $\overline{BAI}$ do not have to be synchronous with the DTC clock. The $\overline{BAI}$ input is usually connected to the $\overline{BUSAK}$ line from the CPU or to the $\overline{BAO}$ output from a higher-priority device in the Bus Request daisy chain. $\overline{AS}$ and $\overline{DS}$ must both be high during the High-to-Low transition of $\overline{BAI}$.

$\overline{BAO}$ (Bus Acknowledge Out, Output)
$\overline{BAO}$ is an active-low output which indicates that $\overline{BAI}$ is active and that the DTC is not currently in control of the bus. This signal is intended for use by lower priority devices on the Bus Request daisy chain.

$\overline{INT}$ (Interrupt, Output)
Interrupt is an active-low, open-drain output used to interrupt the CPU. It may be connected to any of the CPU interrupt inputs, and may be wire-ORed with other sources of interrupts. An external pull-up resistor of 1.8 kΩ or greater is required.

IEI (Interrupt Enable In, Input)
IEI is an active-high input which allows the DTC to activate the $\overline{INT}$ output and to respond to interrupt acknowledge operations. It is used with other signals to implement the interrupt daisy chain. Transitions on IEI do not have to be synchronous with the DTC clock.

IEO (Interrupt Enable Out, Output)
IEO is an active-high output that enables devices lower in the chain when higher priority interrupts are not pending or under service. It is used in conjunction with other signals to implement the interrupt daisy chain. See the Interrupt section of this document for further details on $\overline{INT}$, IEI and IEO.

$\overline{DREQ_1}$, $\overline{DREQ_2}$ (DMA Request, Inputs)
The DMA Request lines are two active-low inputs, one per channel. They may make transitions independent of the DTC clock and are used by external logic to initiate and control DMA operations performed by the DTC.

$\overline{DACK_1}$, $\overline{DACK_2}$ (DMA Acknowledge, Outputs)
The DMA Acknowledge lines are active-low outputs, one per channel, which indicate that the channel is performing a DMA operation. $\overline{DACK}$ is pulsed, held active or held inactive during DMA transfers, as programmed in the Channel Mode register. For Flowthru operations, the peripheral is fully addressed using the conventional I/O addressing protocols and therefore may choose to ignore $\overline{DACK}$. $\overline{DACK}$ is always output as programmed in the Channel Mode register for a DMA operation, even when the operation is initiated by a CPU software request command or as a result of chaining. $\overline{DACK}$ is not output during the actual chaining operations.

$\overline{EOP}$ (End of Process, Input/Output)
$\overline{EOP}$ is an active-low, open-drain, bidirectional signal. It must be pulled up with an external resistor of 1.8 kΩ or more. The DTC emits an output pulse on $\overline{EOP}$ when a TC or MC termination occurs, as defined later. An external source may terminate a DMA operation in progress by driving $\overline{EOP}$ low. $\overline{EOP}$ always applies to the active channel; if no channel is active, $\overline{EOP}$ is ignored. The Suppress output of the MMU may be connected to $\overline{EOP}$ to terminate DMA accesses which violate the MMU protection settings. To provide full access protection, an external $\overline{EOP}$ is accepted even during chaining.

SN_0-SN_6 (Segment Number, Output)
The segment lines are three-state outputs activated only when the DTC is controlling the system bus. SN_0 is the least significant bit of the segment number and SN_6 is the most significant. The AmZ8001 and AmZ8002 CPUs access I/O by outputting a 16-bit I/O address on AD_0-AD_{15}.

When the AmZ8016 DTC is operated in Logical Address space, the I/O address space is increased to 23 bits. The lower 16 bits of I/O address appear on AD_0-AD_{15}. An additional 7 bits of I/O addresses appear on SN_0-SN_6. Users of the DTC in the Logical Address space configuration may choose to disregard the SN_0-SN_6 I/O address information or may use it to increase the DTC's I/O address space beyond that of the CPU.

When the AmZ8016 DTC is configured for Physical Address space, signals SN_0-SN_6 specify the 17th (SN_0) through 23rd (SN_6) bits of a 24-bit linear address. The lower 16 address bits appear on AD_0 through AD_{15} respectively; the 24th address bit is output on SN_7/MMUSync. This 24-bit linear address allows the DTC to access anywhere within 16 Megabytes of memory. Users of the DTC in the physical address space configuration may choose to disregard the extended I/O addressing capability of the DTC by disregarding SN_0-SN_6 and SN_7/MMUSync during I/O operations, or may use the extended addressing to increase the number of I/O ports accessible by the DTC beyond the number of I/O ports accessible by the CPU.

SN_7/MMUSync (Segment Number 7/MMUSync, Output)
When DTC is programmed in Logical Address space, this line outputs an active-HIGH MMUSYNC pulse prior to each machine cycle. The MMU uses this signal to synchronize access to its translation table and to differentiate between CPU and DTC control. The MMU ignores MMUSYNC if ST_0-ST_3 indicate I/O. This output is LOW when DTC is a bus slave and the MM_1 bit is set.

In Physical address space, this line outputs SN_7 which becomes the 24th address bit in a linear address space. This bit can be used to address both memory and I/O – see the SN_0-SN_6 pin description for details. When this output SN_7, a HIGH represents 1 and a LOW represents 0. This pin floats to high impedance state when DTC is a bus slave and the MM_1 bit is cleared.

REGISTER DESCRIPTION

The AmZ8016 block diagram illustrates the internal registers. Figure 3 lists each register along with its size and read/write access restrictions. Registers which can be read by the CPU are either fast (F) or slow (S) readable. Fast registers can be read by an I/O operation without additional wait states. Reading slow registers requires multiple wait states (see Timing Parameters-88 for requirement). It is the responsibility of the user to supply the necessary external logic if slow readable registers are to be read. Registers can be written to by the host CPU (W) and/or can be loaded by the DMA channel itself during chaining (C). All reads or writes must be word accesses since the DTC ignores the B/$\overline{W}$ line in slave mode.

The DTC registers can be categorized into chip-level registers, which control the overall operation and configuration of the DTC, and channel-level registers which are duplicated for each channel. The four chip-level registers are the Master Mode register, the Command register, the Chain Control register and the Temporary register.The Master Mode register selects the way the DTC chip interfaces to the system. The Command register is written to by the host CPU to initiate certain operations within the DTC chip, such as resetting the unit. The Chain Control register is used by a channel while it is reloading its channel-level registers from memory. The Temporary register is used to hold data for Flowthru Transfer/Transfer-and-Searches.

MASTER MODE REGISTER

The 8-bit Master Mode register, shown in Figure 4, controls the chip-level interfaces. It can be read from and written to by the host CPU without wait states through pins AD_0-AD_7 but it is not loadable by chaining. On a reset, the Master Mode register is cleared to all zeroes. The function of each of the Master Mode bits is described in the following paragraphs.

The Chip Enable bit CE = 1 enables the DTC to request the bus. When enabled, the DTC can perform DMA Operations and reload registers. It can always issue interrupts and respond to interrupt acknowledges. When the Chip Enable bit is cleared, the DTC is inhibited from requesting control of the system bus and, therefore, inhibited from performing chaining or DMA operations. The Chip Enable bit (MMO) should not be used as the gating item in starting or stopping the DTC. Channels should be enabled or disabled by using the Set/Clear software request or Hardware Mask Commands. The $\overline{BAO}$ signal follows $\overline{BAI}$ while the Chip Enable bit is cleared.

The Logical/$\overline{Physical}$ Address space bit selects the address space in which the DTC resides. Figure 5 shows the different configuration options. If the DTC outputs addresses which are translated by an MMU, Logical space must be selected. If the addresses output by the DTC pass directly onto the system backplane and if the host CPU is an AmZ8001 using an MMU, the Logical/$\overline{Physical}$ bit must be set to Physical (MM_1 = 0). If

Name	Size	Number	Access Type	Port Address CH-1/CH-2
Master Mode Register	8 bits	1	FW	38
Chain-Control Register	10 bits	1	C	
Temporary Register	16 bits	1	D	
Command Register	8 bits	1	W	2E/2C*
Current Address Register – A:				
Segment/Tag field	15 bits	2	CFW	1A/18
Offset field	16 bits	2	CFW	0A/08
Current Address Register – B:				
Segment/Tag field	15 bits	2	CFW	12/10
Offset field	16 bits	2	CFW	02/00
Base Address Register – A:				
Segment/Tag field	15 bits	2	CFW	1E/1C
Offset field	16 bits	2	CFW	0E/0C
Base Address Register – B:				
Segment/Tag field	15 bits	2	CFW	16/14
Offset field	16 bits	2	CFW	06/04
Current Operation Count	16 bits	2	CFW	32/30
Base Operation Count	16 bits	2	CFW	36/34
Pattern Register	16 bits	2	CSW	4A/48
Mask Register	16 bits	2	CSW	4E/4C
Status Register	16 bits	2	F	2E/2C
Interrupt Save Register	16 bits	2	F	2A/28
Interrupt Vector Register	8 bits	2	CSW	5A/58
Channel Mode Register – High	5 bits	2	CS	56/54
Channel Mode Register – Low	16 bits	2	CSW	52/50
Chain Address Register:				
Segment/Tag field	10 bits	2	CFW	26/24
Offset field	16 bits	2	CFW	22/20

Access Codes: C = Chain Loadable
D = Accessible by DTC channel
F = Fast Readable
S = Slow Readable
W = Writeable

Note: Upper Register Address is determined by user's Chip Select Decode Logic. Only Lower Register Address is shown here.
*The port addresses of Command register can be used alternately for both channels except when issuing a "set/clear IP" command.

Figure 3. DTC Internal Register

AmZ8001 addresses are not translated by an MMU, the DTC must be set to Logical ($MM_1 = 1$). In an AmZ8002 based system, the user may use either the Physical or Logical address space setting.

When set to Logical address space, the segment and offset portions of the Current ARA and ARB registers are viewed as separate portions of the address. Incrementing and decrementing the register affects only the offset portion of the address; no carry or borrow signal is generated into the segment. Only the lower 7 bits of the segment field are used; the setting of the most significant segment bit is disregarded. The 16-bit offset portion of the address appears on pins AD_0-AD_{15} when $\overline{AS}$ is LOW and the 7-bit segment number appears on pins SN_0-SN_6 for the duration of the transaction. The SN_7/MMUSync signal outputs a HIGH pulse prior to each memory transaction, and is never three-stated.

When the Logical/$\overline{\text{Physical}}$ Space bit is set to Physical, the segment and offset portions of the Current ARA and ARB registers are treated as a single linear address. All eight segment bits in the register are used. When an address is incremented or decremented, the carry/borrow signal propagates across the full 24-bit address updating both the segment and offset portions of the address. Both I/O and memory addresses in Physical space are generated by driving the offset portion of the Current Address register onto the AD_0-AD_{15} bus and driving the segment portion of the Current Address register onto the SN_0-SN_7 bus. Timing Diagrams 4, 5 and 6 show how the Logical/$\overline{\text{Physical}}$ Space bit affects the DTC timing. The AD_0-AD_{15} timing is not affected. The SN_0-SN_7 lines are shifted from T_3 to T_1 so that they are

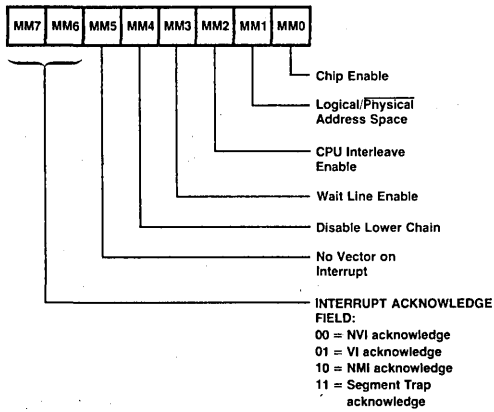
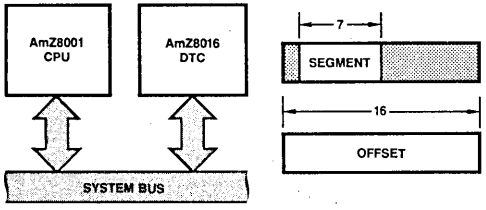
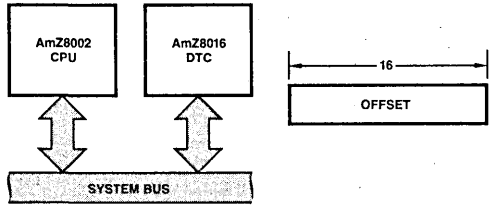


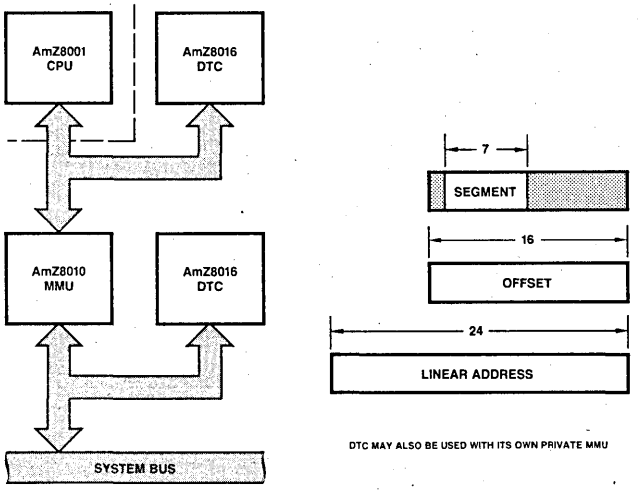
Figure 4. Master Mode Register AIZ-016



a) DTC with AmZ8001 (Segmented) CPU



b) DTC with AmZ8002 (Nonsegmented) CPU



c) DTC with AmZ8001 (Segmented) CPU and AmZ8010 MMU

Figure 5. DTC Configuration Options AIZ-017

valid during the entire transaction. In the Logical Address mode the MMU insures the addresses are valid during the entire transaction.

The CPU Interleave bit enables interleaving between the CPU and the DTC.

The Wait Line Enable bit is used to enable sampling of the $\overline{CS}/\overline{WAIT}$ line during Memory and I/O transactions. Because the DTC provides the ability to insert software programmable wait states, many users may disable sampling of the $\overline{CS}/\overline{WAIT}$ pin to simplify the logic driving this pin. The Wait Line Enable bit provides this flexibility. See the "Wait States" section of this document for details on wait state insertion.

The Disable Lower Chain bit is used to inhibit all lower priority devices on the interrupt daisy chain. When this bit is cleared, the DTC generates LOW and HIGH signals on the IEO output in response to IEI. When the Disable Lower Chain bit is set, IEO is forced LOW, which disables all lower priority interrupts.

The "No Vector on Interrupt" bit selects whether the DTC channel or a peripheral returns a vector during interrupt acknowledge cycles.When this bit is cleared, a channel receiving an interrupt acknowledge will drive the contents of its Interrupt Save register onto the AD_0-AD_{15} data bus while $\overline{DS}$ is LOW. If this bit is set, interrupts are serviced in an identical manner but the AD_0-AD_{15} data bus remains in a high impedance state throughout the acknowledge cycle.

The Vectored/Non-Vectored/Non-Maskable/Segment Trap Field of two bits selects which type of interrupt acknowledge cycle the DTC is to respond to. The DTC decodes from ST_0-ST_3 that an interrupt acknowledge cycle is underway. The setting of this 2-bit field must correspond to the IEI/IEO daisy chain on which the DTC is located to prevent unpredictable results. For example a DTC programmed for vectored interrupts should not be placed on the non-vectored priority chain.

CHAIN CONTROL REGISTER

When a channel starts a chaining operation, it fetches a Reload word from the memory location pointed to by the Chain Address register. This word is then stored in the Chain Control register. The Chain Control register cannot be written to or read from by the CPU. Once a channel starts a chain operation, the channel will not relinquish bus control until all registers specified in the Reload word are reloaded unless an $\overline{EOP}$ signal is issued to the chip. Issuing an $\overline{EOP}$ to a channel during chaining will prevent the chain operation from resuming and the contents of the Reload Word register can be discarded.

TEMPORARY REGISTER

The Temporary register is used to stage data during Flowthru transfers and to hold data being compared during a Search or a Transfer-and-Search. The Temporary register cannot be written to, or read from by the CPU. In byte-word funneling, data may be loaded into or from the Temporary register on a byte-by-byte basis, with bytes sometimes moving between the low byte of the data bus and the high byte of the Temporary register or vice-versa. See the "Transfer" section for details.

COMMAND REGISTER

The DTC Command register is an 8-bit write-only register written to by the host CPU to execute commands. The Command register is loaded from the data on AD_7-AD_0; the data on $AD_{15}-AD_8$ is disregarded. A complete discussion of the commands is given in the "Command Descriptions" section.

CURRENT AND BASE ADDRESS REGISTER A AND B

The Current Address registers A and B (Current ARA and ARB) are used to point to the source and destination addresses for DMA operations. The Base ARA and ARB register contents are transferred into the Current ARA and ARB registers at the end of a DMA operation if the user enables Base-to-Current reloading in the Completion Field of the Channel Mode register. This facilitates DMA operations without reloading of the Current registers. The ARA and ARB registers can be loaded during chaining, can be written to by the host CPU without wait states and can be read by the CPU.

Each of the Base and Current ARA and ARB registers consists of two words organized as a 7-bit Tag Field and an 8-bit segment in one word and a 16-bit offset in the other. See Figure 6. The Segment and Offset contain the actual address driven onto the bus. The Tag Field selects whether the address is to be incremented, decremented or left unchanged, and the status codes associated with the address. The Tag field also allows the user to insert 0, 1, 2 or 4 wait states into memory or I/O accesses addressed by the offset and segment fields.

The Address Reference Select Field in the Tag field selects whether the address pertains to memory space or I/O space. Note that the $N/\overline{S}$ output pin is always LOW (indicating System) for I/O space but may be either HIGH (indicating Normal) or LOW (indicating System) for memory space, as selected in the Address Space field. At the end of each iteration of a DMA Operation, the user may select to leave the address unchanged or to increment it or to decrement it. I/O addresses, if changed, are always incremented/decremented by 2. Memory addresses are changed by 1 if the address points to a byte operand (as programmed in the Channel Mode registers Operation field) and by 2 if the address points to a word operand. Note that if an I/O or memory address is used to point to a word operand, the address must be even to avoid unpredictable results. An address used to point to a byte

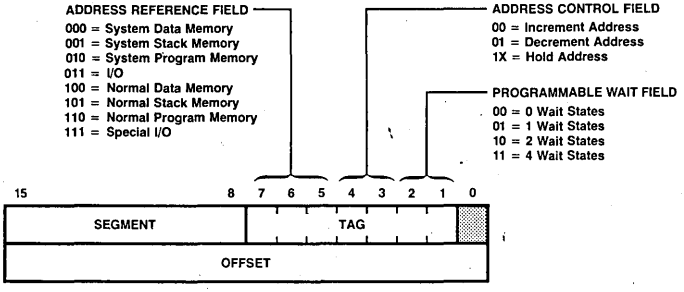


Figure 6. Address Register A and B

AIZ-018

operand may be even or odd. Since memory byte operand addresses will increment/decrement by 1, they will toggle between even and odd values. Since I/O byte operand addresses will increment/decrement by 2, once programmed to an even or odd value, they will remain even or odd, allowing consecutive I/O operations to access the same half of the data bus. High bus is for even address and low bus for odd.

CURRENT AND BASE OPERATION COUNT REGISTERS

Both the Current and Base Operation Count registers may be loaded during chaining, and may be written to, and read from by the host CPU.

The 16-bit Current Operation Count register is used to specify the number of words or bytes to be transferred, searched or transferred-and-searched. For word-to-word operations and byte-word funneling, the Current Operation Count register must be programmed with the number of words to be transferred or searched.

Each time a datum is transferred or searched, the Operation Count register is decremented by 1. Once all of the data is transferred or searched, the transfer or search operation will stop, the Current Operation Count register will contain all zeros, and the TC bit in Status Register will be '1'. If the transfer or search stops before the Current Operation Count register reaches 0, the contents of the register will indicate the number of bytes or words remaining to be transferred or searched. This allows a channel which had been stopped prematurely, to be restarted where it left off without requiring reloading of the Current Operation Count register.

If the DTC is configured for Physical Address Space operation (Master Mode register bit $MM_1 = 0$), the maximum number of words that can be transferred or searched is 64K words.This is specified by setting a word count of 0000. If the DTC is configured for Logical Address Space operation (Master Mode register bit $MM_1 = 1$), the maximum number of words specified to be transferred or searched with either an incrementing or decrementing source or destination address is 32K (8000 hex). This is because in Logical Address Space, offset addresses incremented past FFFF (hex) or decremented below 0000 do not increment/decrement the segment number. Thus, after transferring or searching more than 32K words, the address wraps around within the segment over the same data previously transferred or searched.

For the byte-to-byte operations, the Current Operation Count register should specify the number of bytes to be transferred or searched. The maximum number of bytes which can be specified is 64K bytes; by setting the Current Operation Count register to 0000.

PATTERN AND MASK REGISTERS

The 16-bit Pattern and Mask registers are used in Search and Transfer-and-Search operations. Both the Pattern and Mask registers may be loaded by chaining, may be written to by the host CPU, and may be read from by the host CPU, provided wait states are inserted, since these registers are slow readable. The Pattern register contains the pattern that the read data is compared to. Setting a Mask register bit to '1' specifies that the bit always matches. See the "Search" and "Transfer-and-Search" sections for further details.

STATUS REGISTER

The two 16-bit Status registers, depicted in Figure 7, are read-only registers which can be read by the CPU without wait states. Each of these registers reports on the status of its associated channel.

The Interrupt Status Field in the Status register contains the Channel Interrupt Enable (CIE), Interrupt Pending (IP) and Interrupt Under Service (IUS) bits. These bits are described in detail in the "Interrupt" section of this document.

The DTC status field reports on the current channel state to the CPU. The "channel initialized and waiting for request" status is not explicitly stated – it is reflected by Status register bits ST_{12} through ST_9 being all zero. The "Waiting for Bus" (WFB) status will cause bit ST_{10} to be set and indicates that the channel wants bus control to perform a DMA operation. The channel may or may not actually be asserting $\overline{BUSRQ}$ LOW, depending on the programming of the Master Mode Chip Enable bit and the state of $\overline{BUSRQ}$ and $\overline{BAI}$ when the channel decided it wanted the bus. See the "Bus Request/Grant" section for details. If a channel completes a DMA operation and neither Base-to-Current reloading nor auto-chaining were enabled, the No Auto-Reload or Chaining (NAC) bit will be set. The NAC bit will be reset when the channel receives a "Start Chain Command". If two interrupts are queued, the Second Interrupt Pending bit (SIP) will be set and the channel will be inhibited from further activity until an interrupt acknowledge occurs. See the "Interrupt" section for details. Finally, if the channel is issued an $\overline{EOP}$ during chaining, the Chain Abort (CA), and the NAC will be set. These bits are also set when a "reset" is issued to the DTC. The CA bit holds the NAC bit in the set state. The CA is cleared when a new Chain Address Segment and Tag word or offset word is loaded into the channel.

The Hardware Interface Field provides a Hardware Request (HRQ) bit which provides a means of monitoring the channels $\overline{DREQ}$ input pin. When the $\overline{DREQ}$ pin is LOW, the HRQ bit will be '1' and vice-versa. The Hardware Mask (HM) bit, when set, prevents the DTC from responding to a LOW on $\overline{DREQ}$. Note,

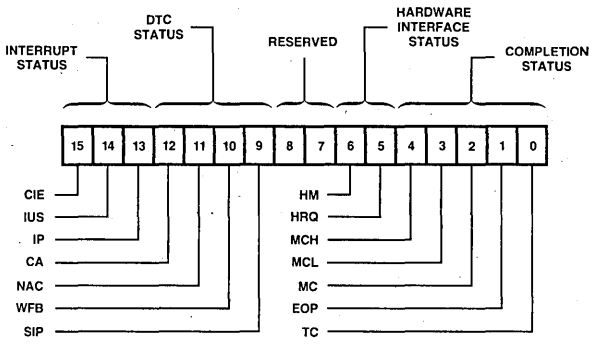


Figure 7. Status Register

AIZ-019

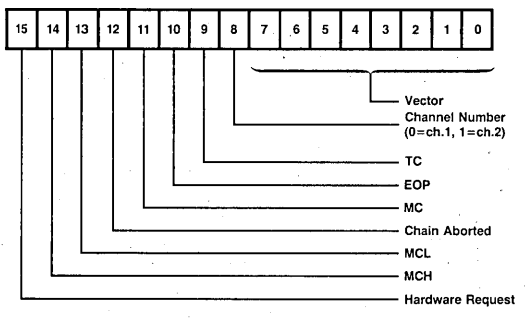
however, that the Hardware Request bit always reports the true (unmasked) status of $\overline{DREQ}$ regardless of the setting of the HM bit. The HM bit can be cleared by software command.

The Completion Field stores data at the end of each DMA operation. This data indicates why the DMA operation ended. When the next DMA operation ends, new data is loaded into these bits overwriting, and thereby erasing the old setting. Three bits indicate whether the DMA operation ended as a result of a TC, MC or EOP termination. The TC bit will be '1' if the Operation Count reaching zero ended the DMA operation. The MC bit will be '1' if an MC termination occurred regardless of whether Stop-on-Match or Stop-on-no-Match was selected. The EOP bit is set only when an $\overline{EOP}$ ends a DMA transfer; it is not set for EOPs issued during chaining. Note that two or even all three of MC, TC and $\overline{EOP}$ may be set if multiple reasons existed for ending the DMA operation. The MCH and MCL bits report on the match state of the upper and lower comparator bytes respectively. These bits are set when the associated comparator byte has a match and are reset otherwise, regardless of whether Stop-on-Match or Stop-on-no-Match is programmed. Regardless of the DMA operation performed, these bits will reflect the comparator status at the end of the DMA operation. These two bits are provided to help determine which byte matched or didn't match when using 8-bit matches with word searches and transfer-and-searches. The two reserved bits return zeroes during reads.

INTERRUPT VECTOR AND INTERRUPT SAVE REGISTERS

Each channel has an Interrupt Vector register and an Interrupt Save register. The Interrupt Vector is 8-bits wide and is written to and read from on AD_0–AD_7. The Interrupt Save register may be read from by the CPU without wait states. The Interrupt Vector register contains the vector or identifier to be output during an Interrupt Acknowledge cycle. When an interrupt occurs (IP = 1), either because a DMA operation terminated or because the $\overline{EOP}$ pin was driven LOW during chaining, the contents of the Interrupt Vector register and part of the Channel Status register are stored in the 16-bit Interrupt Save register (See Figure 8). Because the vector and status are stored, a new vector can be loaded during chaining and a new DMA operation can be performed before an interrupt acknowledge cycle occurs. If another interrupt occurs on the channel before the first is acknowledged, further channel activity is suspended.

As soon as the first clear IP command is issued, the status and vector for the second interrupt is loaded into the Interrupt Save register and channel operation resumes. The DTC can retain only two interrupts for each channel; a third operation cannot be initiated until the first interrupt has been cleared. See the "Interrupt" section for further details.



AIZ-020 **Figure 8. Interrupt Save Register**

CHANNEL MODE REGISTER

Associated with each channel is a Channel Mode registers.There are 21 bits defined in each Channel Mode register the other 11 bits are unused. See Figure 9. The Channel Mode registers may be loaded during chaining and may be read from and written to by the host CPU. CPU reads from the Channel Mode register are slow reads and require insertion of multiple wait states. The Channel Mode register selects what type of DMA operation the channel is to perform, how the operation is to be executed, and what action, if any, is to be taken when the channel finishes.

The Data Operation Field and the Transfer Type field select the type of operation the channel is to perform. It also selects the operand size of bytes or words see Figure 10 for code-definition. The different types of operations are described in detail in the "DMA Operations" section. The Flip bit is used to select whether the Current ARA register points to the source and the Current ARB register points to the destination or vice-versa.

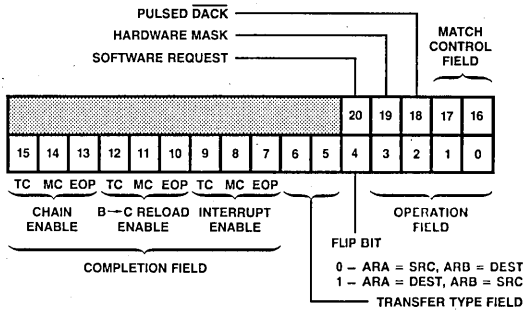


Figure 9. Channel Mode Register AIZ-021

DATA OPERATION FIELD			
Code/Operation	Operand Size ARA	Operand Size ARB	Transaction Type
Transfer			
0001	Byte	Byte	Flowthru
100X	Byte	Word	Flowthru
0000	Word	Word	Flowthru
0011	Byte	Byte	Flyby
0010	Word	Word	Flyby
Transfer-and-Search			
0101	Byte	Byte	Flowthru
110X	Byte	Word	Flowthru
0100	Word	Word	Flowthru
0111	Byte	Byte	Flyby
0110	Word	Word	Flyby
Search			
1111	Byte	Byte	N/A
1110	Word	Word	N/A
101X	Illegal		

TRANSFER FIELD AND MATCH CONTROL FIELD		
Code	Transfer Type	Match Control
00	Single Transfer	Stop on No Match
01	Demand (Bus Hold)	Stop on No Match
10	Demand (Bus Release)	Stop on Word Match
11	Demand Interleave	Stop on Byte Match

Figure 10. Channel Mode Coding

The Completion Field is used to program the action taken by the channel at the end of a DMA operation. This field is discussed in the "Completion Options" section. The 2-bit Match Control field selects whether matches use an 8-bit or 16-bit pattern and whether the channel is to stop-on-match or stop-on-no-match. See Figure 10 and the "Search" section for details. The Software Request bit and Hardware Mask bit can be set and cleared by software command. Only the lower 16 bits can be loaded in parallel with a CPU instruction. These bits are described in detail in the "Initiating DMA Operations" section.

The $\overline{DACK}$ Control bit is used to specify when the $\overline{DACK}$ pin is driven active. When this bit is cleared, the channel's $\overline{DACK}$ pin will be active whenever the channel is performing a DMA Operation, regardless of the type of transaction. Note that the pin will not be active while the channel is chaining. If this bit is set, the $\overline{DACK}$ pin will be inactive during chaining , during both Flowthru Transfers and Flowthru Transfer-and-Searches and during Searches, but $\overline{DACK}$ will be pulsed active during Flyby Transfers and Flyby Transfers-and-Searches at the time necessary to strobe data into or out of the Flyby peripheral. Flyby operations are discussed in detail in the " Flyby Transactions" section.

CHAIN ADDRESS REGISTER

Each channel has a Chain Address register which points to the chain control table in memory containing data to be loaded into the channel's registers. The Chain Address register, as shown in Figure 11, is two words long. The first word consists of the Segment and Tag fields. The second word contains the 16-bit offset portion of the memory address. The highest bit in the segment field is not used when the DTC is configured for Logical Address space (MM1 = 1). The Tag field contains 2 bits used to designate the number of wait states to be inserted during accesses to the Chain Control Table.

The Chain Address register may be loaded during chaining and may be read from and written to by the host CPU without wait states. If an $\overline{EOP}$ is issued to the DTC during chaining, the Chain Address register holds the old address. This is true even if the access failure occurred while new Chain Address data was being loaded, since the old data is restored unless both words of the new data are successfully read. Note, however, that $\overline{EOP}$s that occur when chaining and while loading a new Chain Address cause the new data to be lost.

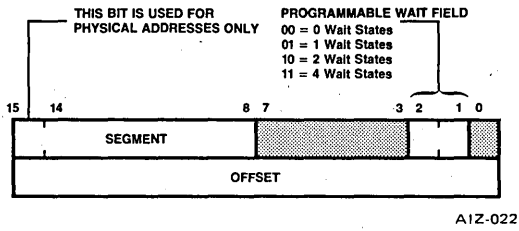


Figure 11. Chain Address Register

FUNCTIONAL DESCRIPTION

Any given DMA operation, be it a transfer, a search or a transfer-and-search operation, consists of three phases. In the first phase, the channel's registers are initialized to specify and control the desired DMA operation. In the second phase, the DMA operation itself is started and performed. The final phase involves terminating the DMA operation and performing any actions selected to occur on termination. Each of these different phases is described in detail in the following sections.

RESET

The DTC can be reset either by hardware or software. The software reset command is described in the "Commands" section. Hardware resets are applied by pulling both $\overline{AS}$ and $\overline{DS}$ LOW. Because the DTC may be in control of the bus when a reset is applied, it is important that $\overline{BAI}$ be driven HIGH when applying a reset to avoid possible bus contention between the applied LOW signals on $\overline{AS}$ and $\overline{DS}$ and the DTC's driving of these pins. As soon as $\overline{BAI}$ goes inactive, the DTC places the AD_0–AD_{15}, SN_0–SN_6, ST_0–ST_3, R/$\overline{W}$, N/$\overline{S}$, B/$\overline{W}$, $\overline{AS}$ and $\overline{DS}$ signals in the high impedance state. If the DTC is programmed for Physical Address Space, SN_7/MMUSync will also be driven into the high impedance state when $\overline{BAI}$ goes HIGH. Figure 22 shows the suggested method of generating hardware resets for the DTC.

Both software and hardware resets clear the Master Mode register, clear CIE, IP, SIP, and WFB and set the CA and NAC in each Channel's Status register. The contents of all other DTC registers will be unchanged for a software reset. Since a hardware reset may have been applied part-way through a DMA operation being performed by a DTC channel, the channel's registers should be assumed to contain indeterminate data following a hardware reset.

Because the CA and NAC bits in the Status register are set by a reset, the channel will be prevented from starting a DMA operation until its Chain Address register's Segment, Tag and offset fields are programmed and the channel is issued a "Start Chain Command".

CHANNEL INITIALIZATION

The philosophy behind the AmZ8016 DTC design is that the DTC should be able to operate with a minimum of interaction with the host CPU. This goal is achieved by having the DTC load its own control parameters from memory into each channel. The CPU has to program only the Master Mode register and each Channel's Chain Address register. All other registers are loaded by the channels themselves from a table located in System Data memory and pointed to by the Chain Address register. This reloading operation is called chaining and the table is called the Chain Control Table.

The offset and segment fields of the Chain Address Register form a 24-bit address or a 23-bit address, which points to a location in system data memory space.Chaining is performed by repetitively reading words from memory. Note that the Chain Address register should always be loaded with an even offset; loading an odd offset will cause unpredictable results. The 2-bit Tag field facilitates interfacing to slow memory by allowing the user to select 0, 1, 2 or 4 programmable wait states. The DTC will automatically insert the programmed number of wait states in each memory access during chaining.

The Chain Address register points to the first word in the Chain Control Table. This word is called the Reload Word. See Figure 12. The purpose of the Reload Word is to specify which registers in the channel are to be reloaded. Reload Word bits 10–15 are undefined and may be 0 or 1. Each of bits 0 through 9 in the Reload Word correspond to either one or two registers in the channel (see Figure 13). When a Reload Word bit is '1', it means that the register or registers corresponding to that bit are to be reloaded. If a Reload Word bit is '0', the register or registers corresponding to that bit are not to be reloaded. The data to be loaded into the selected register(s) follow(s) the Reload Word in memory (i.e., the data are at successively larger memory addresses). The Chain Control Table is a variable length table. Only the data to be loaded are in the table and the data are packed together.

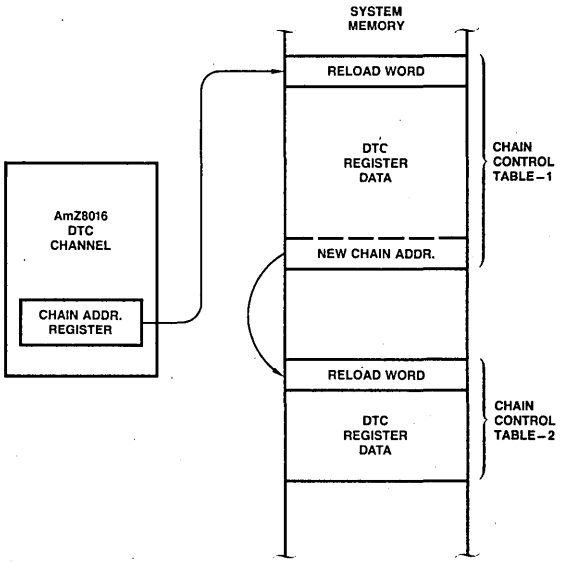


Figure 12. Chaining and Chain Control Tables AIZ-023

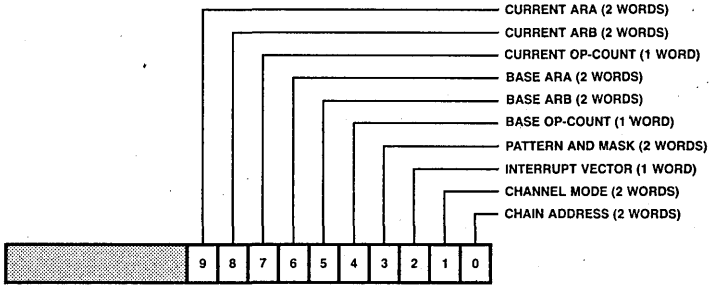


Figure 13. Reload Word AIZ-024

When the channel is to reload itself, it first uses the Chain Address register contents to load the Reload Word into the DTC's Chain Control Register. Next, the Chain Address register contents are incremented by two to point to the next word in memory.The channel then scans to Reload Word register from bit 9 down to bit 0 to see which registers are to be reloaded. If no registers are specified (bits 9–0 are all 0), no registers will be reloaded. If at least one of bits 9–0 are set to '1', the register(s) corresponding to the most significant set bit are reloaded, the bit is cleared and the Chain Address register is incremented by 2. The channel continues this operation of scanning the bits from the most significant to least significant bit position clearing each set bit after reloading its associated registers and incrementing the Chain Address register by 2. If all of bits 9 to 0 are set, all the registers will be reloaded in the order: Current ARA, Current ARB, Current Operation Count, . . . Channel Mode and Chain Address. Figure 14 shows two examples of Chain Control Tables. Example 1 shows the ordering of data when all register are to be reloaded. In example 2 only some registers are reloaded. Once the channel is reloaded, it is ready to perform a DMA operation. Note when loading Address Registers the Segment and Tag Word are loaded first, then the Offset Word.

INITIATING DMA OPERATIONS

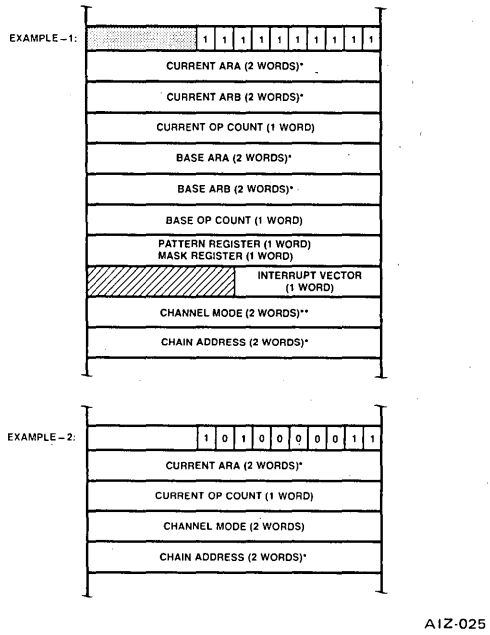
DMA Operations can be initiated in one of three ways – by software request, by hardware request and by loading a set software request bit into the Channel Mode register during Chaining.

Starting After Chaining

If the software request bit of the Channel Mode register is loaded with a '1' during chaining, the channel will perform the programmed DMA operation at the end of chaining. If the channel is programmed for Single Operation or Demand, it will perform the operation immediately. The channel will give up the bus after chaining and before the operation if the CPU Interleave bit in the Master Mode register is set. See the "Channel Response" section for details. Note that once a channel starts a chaining operation by fetching a Reload Word, it retains bus control at least until chaining of the last register's data is performed.

Software Requests

The CPU can issue Software Request commands to start DMA Operations on a channel. This will cause the channel to request the bus and perform transfers. See the description of the software request command for details.



*Note: Load the segment and tag word first, then the offset word.
**Most significant word first, then least significant.

Figure 14. Examples of Chain Control Table

Hardware Requests

DMA operations will often be started by applying a LOW on the channel's $\overline{DREQ}$ input. The "Channel Response" section describes when LOW $\overline{DREQ}$ signals are sampled and when $\overline{DREQ}$ requests can be applied to start the next DMA operation after chaining.

BUS REQUEST/GRANT

Before the DTC can perform a DMA Operation, it must gain control of the system bus. The $\overline{BUSRQ}$, $\overline{BAI}$ and $\overline{BAO}$ interface pins provide connections between the DTC and the host CPU and other DMA devices, to arbitrate which device has control of the system bus. When the DTC wants to gain bus control, it drives $\overline{BUSRQ}$ LOW.

Some period of time after the DTC drives $\overline{BUSRQ}$ LOW, the CPU will relinquish bus control and drive its $\overline{BUSAK}$ signal LOW. This passes down the $\overline{BAI}$, $\overline{BAO}$ daisy chain. When the DTC's $\overline{BAI}$ input goes LOW, it may begin performing operations on the system bus. When the DTC finishes its operation, it stops driving $\overline{BUSRQ}$ LOW and allows $\overline{BAO}$ to follow $\overline{BAI}$.

Listed below are the rules followed by the DTC to request, acquire, and release the system bus. A description of the significance of the steps follows.

1. The DTC requests control of the system bus by driving $\overline{BUSRQ}$ LOW. The DTC may only drive $\overline{BUSRQ}$ LOW if $\overline{BUSRQ}$ is HIGH and $\overline{BAI}$ is HIGH.
2. After driving $\overline{BUSRQ}$ LOW, the DTC waits for its request to be acknowledged on the $\overline{BAI}$ input. When $\overline{BAI}$ goes LOW, the DTC has bus control, performs its operations and continues to drive $\overline{BUSRQ}$ until it completes.
3. When the DTC is finished with the system bus, it stops driving $\overline{BUSRQ}$ LOW and passes the LOW on $\overline{BAI}$ through to $\overline{BAO}$.
4. If the DTC is not requesting use of the bus, $\overline{BAO}$ always follows $\overline{BAI}$. If the DTC receives a LOW on $\overline{BAI}$ and $\overline{BUSRQ}$ is LOW and the DTC is not requesting use of the bus, the DTC drives $\overline{BAO}$ LOW. This situation would occur if some lower priority device was pulling $\overline{BUSRQ}$ LOW. The DTC simply passes the LOW $\overline{BAI}$ grant signal through to the lower priority device.

Note that $\overline{BAO}$ will always be LOW if $\overline{BAI}$ is LOW providing the DTC is not driving $\overline{BUSRQ}$ LOW. If the DTC is driving $\overline{BUSRQ}$ LOW, $\overline{BAO}$ will go low when the DTC finishes using the bus and stops applying a LOW to $\overline{BUSRQ}$. Note also that $\overline{BUSRQ}$ is a bidirectional signal. Since the DTC can only drive $\overline{BUSRQ}$ LOW if $\overline{BUSRQ}$ was previously HIGH, the DTC is able to sample $\overline{BUSRQ}$. Because the DTC may be on a different card than other DTCs and the CPU, some means must be provided to bidirectionally buffer $\overline{BUSRQ}$. Figure 22 shows a representative system with two DTC chips and a CPU. Figure 23 shows the logic used to bidirectionally buffer $\overline{BUSRQ}$. Note that the buffer and gates in the logic are both open collector (o.c.) devices.

It is necessary to ensure that all DTCs will behave identically, regardless of whether they are on the same card or different cards. Also, it is undesirable to require users to provide to Detail A logic on all DTCs, except where the logic is needed to provide buffering to drive a backplane. For this reason, each DTC incorporates identical logic to Detail A inside the chip. Thus, even if no external logic is used, the bus request-grant protocol will follow the above description. Note that when external buffering is used, the design of Detail A is such that when it is placed in series with the replicated Detail A logic inside the chip, the operation of the bus request protocol remains unchanged.

DMA OPERATIONS

There are three types of DMA operations: Transfer, Search and Transfer-and-Search. Transfers move data from a source location to a destination location. Two types of transfers are provided: Flowthru and Flyby. Searches read data from a source and compare the read data to the contents of the Pattern register. A Mask register allows the user to declare "don't care" bits.

The user can program that the search is to stop either when the read data matches the masked pattern or when the read data fails to match the masked pattern. This capability is called Stop-on-Match and Stop-on-no-Match. Transfer-and-Search combines the two functions to facilitate the transferring of variable length data blocks. Like transfer, Transfer-and-Search can be performed in either Flowthru or Flyby mode.

Transfers

Transfers use four of the Channel registers to control the transfer operation: the Current ARA and ARB register; the Current Operation Count register; and the Channel Mode register. Channel Mode register bit CM_4 is called the Flip bit and is used to select whether ARA is to point to the source and ARB is to point to the destination or vice-versa. The Current Operation Count register specifies the number of words or bytes to be transferred.

Bits CM_3–CM_0 in the Channel Mode register program whether Flowthru or Flyby transfer is to be performed. Flowthru transfers are performed in either two or three steps. First, the channel outputs the address of the source and reads the source data into the DTC's Temporary register. In two-step Flowthru Transfer, the channel will then address the destination and write the Temporary register data to the destination location. The three-step Flowthru operation is described later in this section. The source and destination for Flowthru Transfers can both be memory locations or both peripheral devices or one may be a memory location

and the other a peripheral device.The $\overline{DACK}$ output for the transferring channel may be programmed to be inactive throughout the transfer or active during the transfer. This is controlled by bit CM_{18} in the Channel Mode register.

Flyby transfers provide improved transfer throughput over Flowthru but are restricted to transfers between memory and peripherals or between two peripherals. Flyby operations are described in detail in the "Flyby Transactions" section.

Transfers can use both byte- and word-sized data. Flowthru byte-to-byte transfers are performed by reading a byte from the source and writing a byte to the destination. The Current Operation Count register must be loaded with the number of bytes to be transferred. Both the Current ARA and Current ARB registers, if programmed to increment/decrement, will change by ±1 if the register points to memory space and by ±2 if the register points to I/O space.

Flowthru word-to-word transfers require that the Current Operation Count specify the number of words to be transferred. Both the Current ARA and Current ARB registers, if programmed to increment/decrement, will change by ±2 regardless of whether the register points memory or I/O space.

Byte-word funneling provides packing and unpacking of byte data to facilitate high speed transfers between byte and word peripherals and/or memory. This funneling option can only be used in Flowthru mode. Funneled Flowthru transfers are performed in three steps. For transfers from a byte source to a word destination, two consecutive byte reads are performed from the source address. The data read is assembled into the DTC's Temporary register. In the third step, the Temporary register data is written to the destination address in a word transfer. Funneled transfers from a word source to a byte destination are performed by first loading a word from the source into the DTC's Temporary register. The word is then written out to the destination in two byte writes. For funnel operations, the byte-oriented address must be in the Current ARA register and the word-oriented address must be in the Current ARB register. The Flip bit (CM_4) in the Channel Mode register is used to specify which address is the source and which is the destination. When the byte address is to be incremented or decremented, the increment/decrement operation occurs after each of the two reads or writes. The increment/decrement is by ±1.

In byte-to-word funneling operations it is necessary to specify which half of the Temporary register (upper or lower byte) is loaded with the first byte of data. Similarly, for word-to-byte funneling operations it is necessary to define which half of the Temporary register is written out first. Figure 15 summarizes these characteristics for both byte-to-word and word-to-byte funneling operations. The criteria used to determine the packing/unpacking order is based on whether the Current ARB register is programmed for incrementing or decrementing of the address. Note that if the address is to remain unchanged (i.e., if bit TG_4 in the Tag Field of the Current ARB register is 1), the increment/decrement bit (bit TG_3) still specifies the packing order.

Search

Searches use five of the Channel registers to control the transfer operation: either the Current ARA or ARB; the Operation Count; the Pattern and Mask registers; and the Channel Mode register. Channel Mode register bit CM_4 is called the Flip bit and is used to select either Current ARA or ARB as the register specifying the source for the search. Only one of the Current Address registers is used for search operations since there is no destination address required. Channel mode register bit CM_2 is an enable for the output of the comparator and allows the MC (match condition) signal to be generated. The Current Operation Count register specifies the maximum number of words or bytes to be searched.

Search operations involve repetitive reads from the peripheral or memory until the specified match condition is met. The search then stops. This is called a Match Condition or MC termination. Each time a read is performed, the Source address, if so programmed, is incremented or decremented and the Operation Count is decremented by 1. If the match condition has not been met by the time the Operation Count reaches zero, the zero value will force a TC termination, ending the search. Searches can also stop due to a LOW being applied to the $\overline{EOP}$ interface pin. During a search operation, the channel's $\overline{DACK}$ output will be either inactive or active throughout the search. This is controlled by bit CM_{18} in the Channel Mode register. The reads from the peripheral or memory performed during search follow the timing sequences described in the "Flowthru Memory Transactions" and "Flowthru I/O Transactions" sections.

On each read during a Search operation, the DTC's Temporary register is loaded with data and compared to the Pattern register. The user can select that the search is to stop when the Pattern and Temporary register contents match or when they don't match. This Stop-On-Match/Stop-On-No-Match feature is programmed in bit CM_{17} of the Channel Mode register. A Mask register allows the user to exclude or mask selected Temporary register bits from the comparison by setting the corresponding Mask register bit to '1'. The masked bits are defined to always match. Thus, in Stop-On-Match, successful matching of the unmasked bits, in conjunction with the always-matched masked bits, will cause the search to stop. For Stop-On-No-Match, the always-matched masked bits are by definition excluded from not matching and therefore excluded from stopping the search.

For word reads the user may select either 8-bit or 16-bit compares through Channel Mode register bit CM_{16}. In an 8-bit, Stop-On-Match, word-read operation, successful matching of either the

Funneling Direction	Current ARB Tag Field		Increment/Decrement and Packing/Unpacking Rules
	TG4	TG3	
Word-to-Byte (Flip-bit = 1)	0	0	Increment ARB, Write High Byte First
	0	1	Decrement ARB, Write Low Byte First
	1	0	Hold ARB, Write High Byte First
	1	1	Hold ARB, Write Low Byte First
Byte-to-Word (Flip-bit = 0)	0	0	Increment ARB, Read High Half of Word Written First
	0	1	Decrement ARB, Read Low Half of Word Written First
	1	0	Hold ARB, Read High Half of Word Written First
	1	1	Hold ARB, Read Low Half of Word Written First

Figure 15. Byte/Word Funnelling

upper or lower byte of unmasked Pattern and Temporary registers bits will stop the search. Both bytes do not have to match. In 16-bit Stop-On-Match with word reads, all unmasked Pattern and Temporary register bits must match to stop the search. In an 8-bit or 16-bit, Stop-On-No-Match, word-read Search operation, failure of any bit to match will terminate the Search operation.

In an 8-bit Stop-On-Match the byte-reads, the Search will Stop if either the upper or lower byte of unmasked Pattern and Temporary register bits match. For an 8-bit Stop-On-No-Match with byte reads, failure of matching in any unmasked Pattern and Temporary register bit will cause the search to stop.

For 8-bit searches, the upper and lower bytes of the Pattern and Mask register should usually be programmed with the same data. Failure to set the upper and lower bytes of the Pattern and Mask registers to identical values will result in different comparison criteria being used for the upper an lower bytes of the Temporary register. Users failing to program identical values for the upper and lower bytes can predict the results by recognizing that in 8-bit Stop-On-Match, the search will end if all the unmasked bits in either the upper or lower byte matches, and for 8-bit Stop-On-No-Match, the failure of any unmasked bit to match will end the search. For accurate predictions, it is also necessary to know that for word reads the Temporary register high and low bytes are loaded from $AD_{15}-AD_8$ and AD_7-AD_0 respectively. In byte reads, the read byte is duplicated in both halves of the Temporary register except in funneling.

Transfer-and-Search

Transfer-and-Search combines the operations of Transfer and Search functions. The registers used to control Transfer-and-Searches are the Current ARA and ARB register; the Operation Count register; the Pattern and Mask register; and the Channel Mode register.

A Transfer-and-Search operation will end when the data transferred meets the match condition specified in Channel Mode register bits $CM_{17}-CM_{16}$. The Mask and Pattern registers indicate those bits being compared with the Temporary register contents. Like Transfers and Searches, Transfers-and-Searches will also be terminated if the operation count goes to zero or if a LOW is applied to the $\overline{EOP}$ pin. Regardless of whether Transfer-and-Search stops because of a TC, MC or EOP, it will always complete the iteration by writing to the destination address before ending (writing twice for word-to-byte funneling).

In Flowthru mode, Transfer-and-Search the timing is identical to Flowthru Transfer. While the data is in the Temporary register, it is masked by the Mask register and compared to the Pattern register. For word Transfer and Transfer-and-Search, the high and low bytes of the Temporary register are always written to and read from $AD_{15}-AD_8$ and AD_7-AD_0 respectively. For byte Transfer and Transfer-and-Search, the byte read is always loaded into both halves of the Temporary register and the entire register is driven directly out onto the $AD_{15}-AD_0$ bus. Transfer-and-Search can also be used with byte word funneling. In funneling, the match is an 8-bit match or 16-bit match as determined by the setting of bit CM_{16} and CM_{17}.

Flyby Transfer-and-Search can be used to increase throughput for transfer between two peripherals or between memory and a peripheral. In this operation, the operand sizes of the source and destination must be the same. A complete discussion of Flyby timing is given in the "Flyby Transactions" section. During a Flyby Transfer-and-Search, data is loaded into the Temporary register to facilitate the comparison operation and at the same time data is transferred from the source to the destination. When byte operands are used, data is loaded into both bytes of the Temporary register, from the $AD_{15}-AD_8$ bus if the Current ARA register is even and from AD_7-AD_0 line if the Current ARA register is odd. This will alternate for memory bytes so the user must drive both halves of the bus to use the search. When word operands are used, data is loaded directly from $AD_{15}-AD_8$ and AD_7-AD_0 into the Temporary register's high and low bytes respectively.

CHANNEL RESPONSE

Channel Mode register bits CM_6-CM_5 select the channel's response to the request to start a DMA operation. The response falls into either of two types: Single Operation or Demand. There are three subtypes for Demand operations: Demand Dedicated with Bus Hold, Demand Dedicated with Bus Release, and Demand Interleave. To make the discussions clear, it is necessary to define the term "single iteration of a DMA operation". For Search operations, one iteration consists of a single read operation and a comparison of the read data to the unmasked Pattern register bits. The Operation Count will be decremented by 1 and the Current Address register used incremented or decremented if so programmed. For Transfer and Transfer-and-Search operations, a single iteration comprises reading a datum from the source, writing it to the destination, comparing the read datum to the unmasked Pattern register bits (Transfer-and-Search only), decrementing the Operation Count by 1 and incrementing/decrementing the Current ARA and ARB registers if so programmed. In byte-word funneling, a single iteration consists of two reads followed by a write (Byte-to-Word funneling) or one read followed by two writes (Word-to-Byte funneling). In all Transfer and Transfer-and-Search cases the iteration will not stop until the data in the Temporary register is written to the destination.

Single Operation

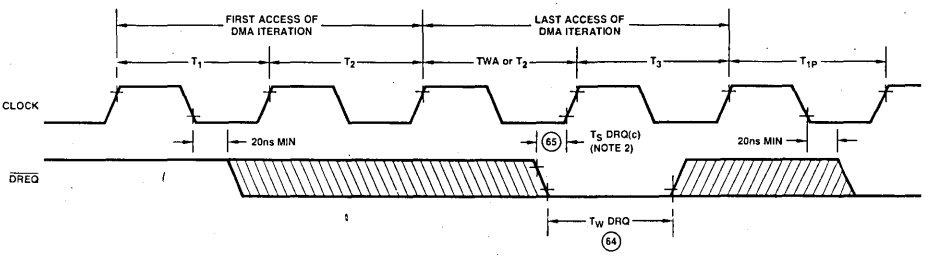
The Single Operation response is intended for use with peripherals which transfer single bytes or words at irregular intervals. Each application of a Software request command will cause the channel to perform a single iteration of the DMA operation. Similarly, if the Software request bit is set by chaining, at the end of chaining the channel will perform a single iteration of the DMA operation. Each application of a HIGH-to-LOW transition on the $\overline{DREQ}$ input will also cause a single iteration of the DMA operation. If the Hardware mask bit is set when the transition is made, the iteration will be performed when the mask is cleared, providing the DMA operation has not terminated. See the Set/Clear Hardware mask bit command for details. Each time a Single Operation ends, the channel will give up control of the bus unless a new transition has occurred on $\overline{DREQ}$. The new transition can occur anytime after the LOW-to-HIGH $\overline{AS}$ transition on the first memory or I/O access of the DMA iteration. Timing Diagram 1 shows the times after which a new transition can be applied and recognized to avoid giving up the bus at the end of the current iteration.

Demand Dedicated with Bus Hold

In Demand Dedicated with Bus Hold (abbreviated Bus Hold), the application of a Software request command or the setting of the software request bit during chaining or applying a LOW level on the $\overline{DREQ}$ input will cause the channel to aquire bus control.

If $\overline{DACK}$ is programmed as a level output ($CM_{18} = 0$), $\overline{DACK}$ will be active from when the channel acquires bus control to when it relinquishes control.

Once the channel gains bus control due to a LOW $\overline{DREQ}$ level, it samples $\overline{DREQ}$ as shown in Timing Diagram 2. If $\overline{DREQ}$ is LOW, an iteration of the DMA operation is performed. If $\overline{DREQ}$ is HIGH, the channel retains bus control and continues to drive all bus control signals active or inactive, but performs no DMA operation. This the user can start or stop execution of DMA operations by modulating $\overline{DREQ}$. Once TC, MC or EOP occurs, the channel will



Notes:
1. HIGH-to-LOW $\overline{DREQ}$ transitions will only be recognized after the HIGH-to-LOW transition of the clock during T_1 of the first access of the DMA iteration.
2. A HIGH-to-LOW $\overline{DREQ}$ transition must meet the conditions in Note 1 and must occur T_SDRQ(c) before state T_3 of the last access of the DMA iteration if the channel is to retain bus control and immediately start the next iteration. $\overline{DREQ}$ may go HIGH before T_SDRQ(c) if it has met the T_WDRQ parameter.
3. Flyby and Search transactions have only a single access; parameter T_SDRQ(c) should be referenced to the start of T_3 of the access. All other operations will always have two or three accesses per iteration.

Timing Diagram 1. Sampling $\overline{DREQ}$ During Single Transfer DMA Operations AIZ-026

either release the bus or, if chaining or Base-to-Current reloading is to occur, perform the desired operation. After chaining or Base-to-Current reloading, if the channel is still in Bus Hold mode and does not have a set software request bit (set either by chaining or command), the channel will relinquish bus control unless a LOW $\overline{DREQ}$ level occurs within the time limits.

Demand Dedicated with Bus Release

In Demand Dedicated with Bus Release (abbreviated Bus Release), the application of a Software Request command will cause the channel to request the bus and perform the programmed DMA operation until TC, MC or EOP. If the channel was programmed for Bus Release, and the software request bit was set during chaining, the channel will start the DMA operation as soon as chaining ends, without releasing the bus, and will continue performing the operation until TC, MC or EOP.

When an active $\overline{DREQ}$ is applied to a channel programmed for Bus Release, the channel will acquire the bus and perform DMA operations until (a) TC, MC or EOP or (b) until $\overline{DREQ}$ goes inactive. Timing Diagram 2 (b) shows when $\overline{DREQ}$ is sampled to determine if the channel should perform another cycle or release the bus. Note that this sampling also occurs on the last cycle of a chaining operation. If a channel has an active $\overline{DREQ}$ at the end of chaining, it will begin performing DMA operations immediately, without releasing the bus. When a TC, MC or EOP occurs, terminating a Bus Release mode operation, the channel, if enabled for chaining and/or Base-to-Current reloading, will perform chaining and/or reloading (assuming the Status register's SIP bit is clear) without releasing the bus.

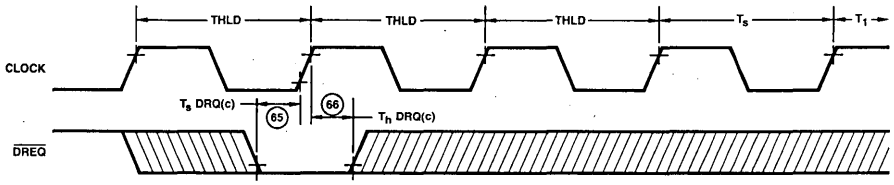
If an active request is not applied and the channel is in Demand Dedicated with Bus Hold, the channel will go into state THLD (see Timing Diagram 2 (a)). If an active request is not applied and the channel is in Demand Dedicated with Bus Release or Demand Interleave mode, it will release the bus. Note that even if an active request is applied in Demand Interleave, the channel may still release the bus. The request for Demand Interleave should continue to be applied to ensure that the channel eventually responds to the request by acquiring the bus (i.e., the request is not latched by the channel).

Demand Interleave

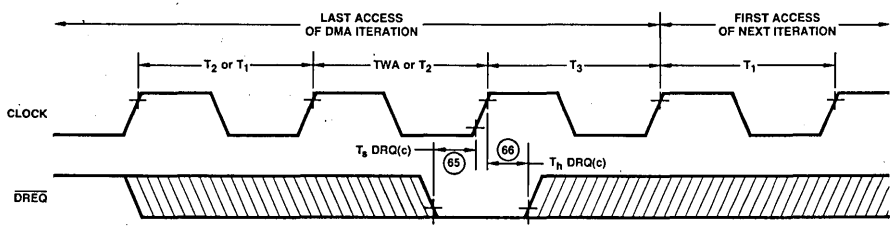
Demand Interleave behaves in different ways depending on the setting of Master Mode register bit MM_2. If MM_2 is set, the DTC will always relinquish bus control and then re-request it after each DMA iteration. This permits the CPU and other devices to gain bus control. For instance, if MM_2 is clear, control can pass from one DTC channel to the other without requiring the DTC to release bus control. If both channels have active requests, control will pass to the channel which did not just have control. If MM2 is clear and both channels have active requests and are in Demand Interleave mode, control will toggle between the channels after each DMA operation iteration and the DTC will retain bus control until both channels are finished with the bus. If MM2 is set and both channels have active requests and are in Demand Interleave mode, each channel will relinquish control to the CPU after each iteration resulting in the following control sequence: channel 1, CPU, channel 2, CPU, etc. Note that if there are other devices on the bus request daisy chain, they may gain control during the part of the sequence labelled CPU.

A software or hardware request will cause a channel programmed for Demand Interleave to perform interleaved DMA operations until TC, MC or EOP. If the Software request bit is set during chaining, the channel will retain the bus after chaining and will immediately start performing DMA iteration and will interleave all DMA iterations after the first. If $\overline{DREQ}$ is LOW on the last cycle during chaining, the channel will perform a single iteration immediately after chaining and interleave thereafter until (a) TC, MC or EOP or (b) $\overline{DREQ}$ goes HIGH. If (b) occurs, the channel will relinquish the bus until $\overline{DREQ}$ goes LOW again and the channel again starts performing interleaved operations. If (a) occurs, the channel will not interleave before first performing chaining and/or Base-to-Current reloading (assuming SIP is cleared).

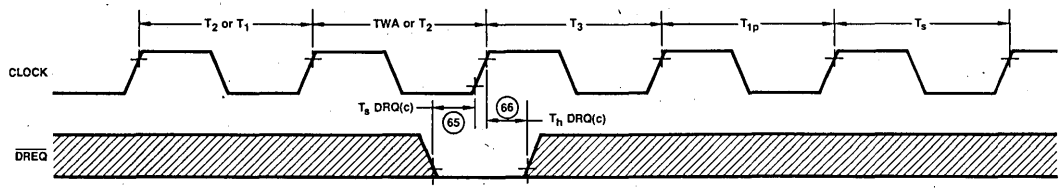
The waveform of $\overline{DACK}$ is programmed in Channel Mode Register (CM_{18}). The Pulsed $\overline{DACK}$ is for flyby transaction only. See Timing Diagram 3. Note: This figure shows a single Search or Flyby iteration. State TWA is optionally inserted if programmed. For more than one iteration, the level $\overline{DACK}$ output would stay active during the time the channel had bus control. When CM_{18} is set, the $\overline{DACK}$ output will be inactive for all non-flyby modes.



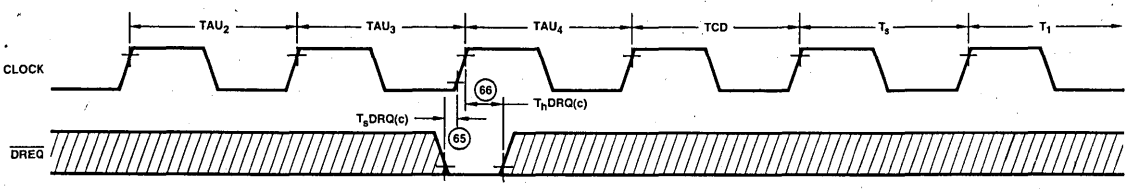
a) Sampling $\overline{\text{DREQ}}$ while in Bus Hold Mode AIZ-027



b) $\overline{\text{DREQ}}$ Sampling in Demand Mode During DMA Operations AIZ-028



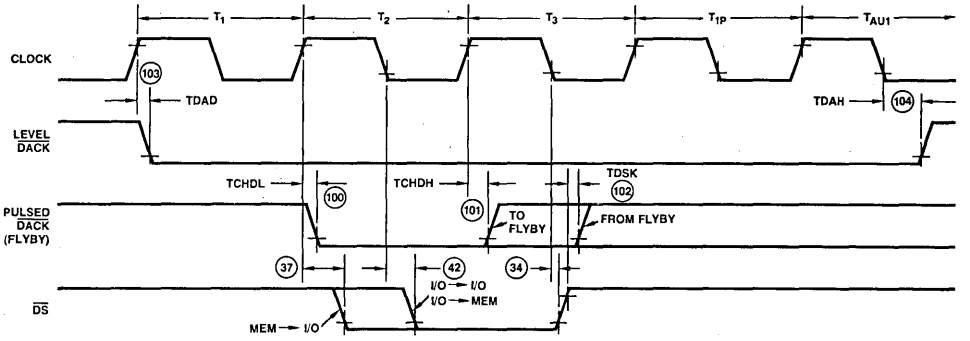
c) Sampling $\overline{\text{DREQ}}$ at the End of Chaining AIZ-029



d) Sampling $\overline{\text{DREQ}}$ at the End of Base-to-Current Reloading AIZ-030

Notes: 1. $\overline{\text{DREQ}}$ must be LOW from the start of $T_sDRQ(c)$ to the end of $T_hDRQ(c)$ to ensure that the request is recognized.
2. Failure to meet this setup time will result in the channel releasing the bus.
3. T_S is a setup state, generated before entering DMA operation cycle.
4. TAU_2 through TAU_4 are auto-reloading states, followed by TCD (chain decision) state.

Timing Diagram 2. $\overline{\text{DREQ}}$ Sampling in Demand Mode



Note: *LEVEL $\overline{\text{DACK}}$ RE occurs as shown if auto-reloading is not programmed. LEVEL $\overline{\text{DACK}}$ stays LOW for three additional clocks for reloading.

Timing Diagram 3. $\overline{\text{DACK}}$ Timing

AIZ-031

WAIT STATES

The DTC has a $\overline{\text{WAIT}}$ input which is multiplexed with Chip Select ($\overline{\text{CS}}$) yielding a $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ input. This pin functions as a $\overline{\text{CS}}$ for the DTC when the DTC is not in control of the bus and as a $\overline{\text{WAIT}}$ input when the DTC is bus master. Because multiplexing $\overline{\text{CS}}$ and $\overline{\text{WAIT}}$ requires external logic, (see Figure 24), the user can select that wait states are automatically inserted when the DTC, as bus master, accesses I/O or memory addresses. The number of wait states to be added to the memory or I/O transfer can be programmed by the user as 0,1, 2 or 4 and can be separately programmed for the Current Address registers A and B and for the Chain Address register. This allows different speed memories and peripheral to be associated with each of these addresses. The Base Address registers A and B also have a Tag Field which is loaded into the Current ARA and ARB registers during Base-to-Current reloading. Because many users utilizing the software programmable wait states will not need the ability to generate hardware wait states through the $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ pin, the wait function can be disabled, yielding a Chip Select input only, by clearing the Wait Line Enable bit (MM_3) in the Master Mode register.

During memory transactions, the $\overline{\text{WAIT}}$ input is sampled in the middle of the T_2 state. If $\overline{\text{WAIT}}$ is HIGH, and if no programmable wait states are selected, the DTC will proceed to state T_3. Otherwise, at least one wait state will be inserted. The flowthru I/O transaction should be programmed to have one wait state inserted (TWA) for Z8000 peripherals, otherwise timing is the same as memory transactions. The $\overline{\text{WAIT}}$ line is then sampled in the middle of state TWA. If $\overline{\text{WAIT}}$ is HIGH the DTC will proceed to state T_3. Otherwise additional wait states will be inserted.

Consider what happens in a transaction when both hardware and software wait states are inserted. Each time the $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ line is sampled, if it is LOW, a hardware wait state will be inserted in the next cycle. The software wait state insertion will be suspended until $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ is sampled and is HIGH. The hardware wait states may be inserted anytime during the software wait state sequence. It is important to note that hardware wait states are served consecutively rather than concurrently with software wait states. For example, assume for a Flowthru I/O Transaction that a user has programmed 4 software wait states. Driving a LOW on the $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ input during T_2 for 2 cycles would insert 2 hardware wait states. Driving $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ HIGH for 3 cycles would allow insertion of three of the four software wait states. Driving $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ LOW for 2 more cycles would insert 2 more hardware wait states. Finally, driving $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ HIGH would allow the final software wait state to be inserted. During this last software wait state, the $\overline{\text{CS}}$/$\overline{\text{WAIT}}$ pin would be sampled for the last time. If it is HIGH, the channel will proceed to state T_3.

If the pin is LOW, the channel will insert hardware wait states until the pin goes HIGH and the channel would then enter state T_3 to complete the I/O transaction.

DMA TRANSACTIONS

There are three types of transactions performed by the AmZ8016 DTC: Flowthru, Flyby, and Search. Figures 16 and 17 show the configurations of Flowthru and Flyby Transactions.

Flowthru I/O Transactions

There are Two types of I/O space on the AmZ8016: I/O and Special I/O. Status lines ST_0–ST_3 specify when an I/O operation

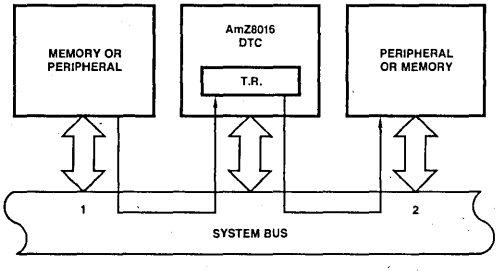


Figure 16. Configuration of Flowthru Transaction

AIZ-032

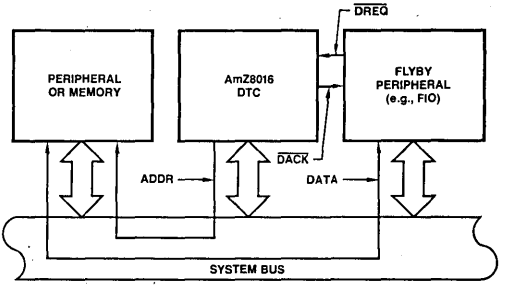


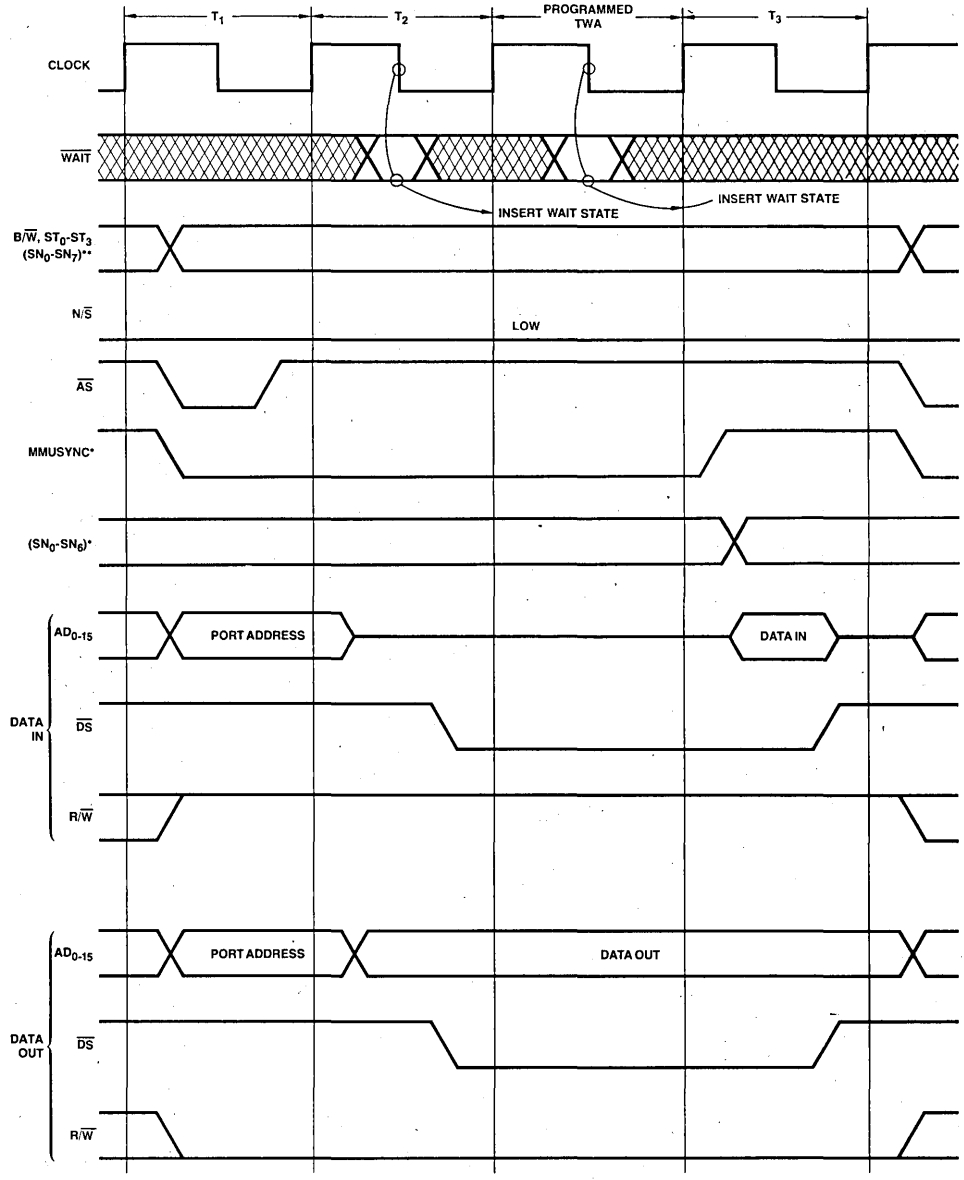
Figure 17. Configuration of Flyby Transaction

AIZ-033

is being performed and which of the two I/O spaces are being accessed, as shown in Figure 2. During an I/O transaction, status signal N/$\overline{S}$ will be LOW to indicate a System Level operation.

Each I/O space is addressed by the host CPU by a 16-bit address. The DTC allows an extended I/O address of 24- or 23-bits to be used at the option of the user. When the Master Mode register bit MM_1 is cleared, the DTC is configured for Physical Address space. If MM_1 is set, the DTC is configured for Logical Address space.

The timing for I/O and Special I/O operations are identical. An I/O cycle consists of three states: T_1, T_2, and T_3 as shown in Timing Diagram 4. The TWA state is a wait state programmed to be inserted by the user. The user may select to insert additional software wait states through the Tag fields of the Current ARA and ARB registers. In addition, if Master Mode register bit MM_3 = 1, hardware wait states may be inserted by driving a LOW signal on the $\overline{CS}/\overline{WAIT}$ pin.



*For logical addressing only
**For physical addressing only

Timing Diagram 4. Flowthru I/O Transactions

AIZ-034

The ST_0-ST_3 lines will reflect the appropriate code for the current cycle (I/O or Special I/O) early in T_1 and the AS ouput will be pulsed LOW to mark the beginning of the cycle. The offset portion of the address for the peripheral being accessed will appear on AD_0-AD_{15} during T_1. The N/$\overline{S}$ line will be set LOW (system) and the R/$\overline{W}$ line and B/$\overline{W}$ line will select a read or write operation for bytes or words. The N/$\overline{S}$, R/$\overline{W}$ and B/$\overline{W}$ lines will become stable during T_1 and will remain stable until after T_3.

I/O address space is byte-address but both 8- and 16-bit data sizes are supported. During I/O transactions the B/$\overline{W}$ ouput signal will be HIGH for byte transactions and LOW for word transactions. For I/O transactions, both even and odd addresses can be output, hence the address bit output on AD_0 may be 0 or 1.

The channel can perform both I/O read and I/O write operations. During an I/O read, the R/$\overline{W}$ output will be HIGH. The AD_0-AD_{15} bus will be placed in the high impedance state by the DTC during T_2. The DTC drive the $\overline{DS}$ output LOW to signal the peripherals that data can be gated onto the bus. The DTC will strobe the data into it's Temporary register during T_3. $\overline{DS}$ will be driven HIGH to signal the end of the I/O transaction.

For byte I/O writes, the channel will drive the same data on data bus lines AD_0-AD_7 and AD_8-AD_{15}. During byte I/O reads when the address bit on AD0 is 0, the DTC will strobe data in from data lines AD_8-AD_{15}. During byte I/O reads when the address bit on AD_0 is 1, the DTC will strobe data in from data lines AD_0-AD_7. Thus, when an 8-bit peripheral is connected to the bus, it's internal registers will typically be mapped at all even or all odd addresses. To simplify access to 8-bit peripherals, byte oriented I/O address are incremented/decremented by 2.

Flowthru Memory Transactions

There are six status codes which can be generated by a DTC channel while it is accessing memory. See Figure 2. Thus, if a user segregates memory into different banks by decoding the NORMAL/$\overline{SYSTEM}$ and ST_0-ST_3 lines, the DTC can be used to move data from space to space.

The timing for all Flowthru memory transactions is the same, regardless of the status code being output. During chaining operations the DTC reads words from an address in System Data memory pointed to by the active channel's Chain Address register. Those chaining operations are performed identically to the Flowthru memory read transactions, except that the data is loaded into an internal DTC channel register rather than the Temporary register. Note that chaining never causes a write or a byte read; thus all memory writes or all byte accesses are due to DMA operations. A typical memory operation consists of three cycles: T_1, T_2 and T_3, as shown in Timing Diagram 5. The user may select to insert 1, 2, or 4 software wait states after state T_2 and before state T_3 by programming the Current Address register Tag field. If the Wait Line Enable bit in the Master Mode register is set, the user may also insert hardware wait states after state T2 and before state T_3 by driving a LOW on the $\overline{CS}$/$\overline{WAIT}$ signal.

The operation of Flowthru memory transaction are performed identical to the Flowthru I/O transactions except for $\overline{DS}$ width. (See Timing Diagrams.)

Flyby Transactions

Flyby transfers and transfer-and-search operations are performed in a single step, providing a transfer rate significantly faster than that available from Flowthrus. In Flyby, operations can only be performed between memory and peripherals or between peripherals and peripherals. Memory-to-memory operations can not be performed in Flyby mode; these must be done using Flowthru.

Flyby Memory-peripheral operations can only be used with peripherals having a special Flyby signal input. This peripheral input is connected to the channel's $\overline{DACK}$ output. For memory-peripheral Flyby, the address of the source memory location, must be programmed in the Current ARA register. The Current ARB register must be programmed with the destination memory location for peripheral-memory Flyby. Flyby peripheral-to-peripheral operations can only be performed if one of the peripherals has a Flyby signal input. This peripheral input must be connected to the channel's $\overline{DACK}$ output. If both peripherals have a Flyby input, only one should be connected to $\overline{DACK}$; the other peripheral's Flyby input should be held high during the Flyby operation. The address of the peripheral not connected to the channel's $\overline{DACK}$ output should be programmed in the Current ARB register. Note that Flip bit is set (CM4 = 1) for I/O to memory write transactions and cleared (CM4 = 0) for memory read to I/O transaction.

A Flyby operation is performed using three states: T_1, T_2, and T_3. During T_1 the channel pulses AS and outputs the address information. See Timing Diagram 6. The R/$\overline{W}$ line is HIGH if the Current ARA specifies the source and Low if the Current ARB specifies the destination.

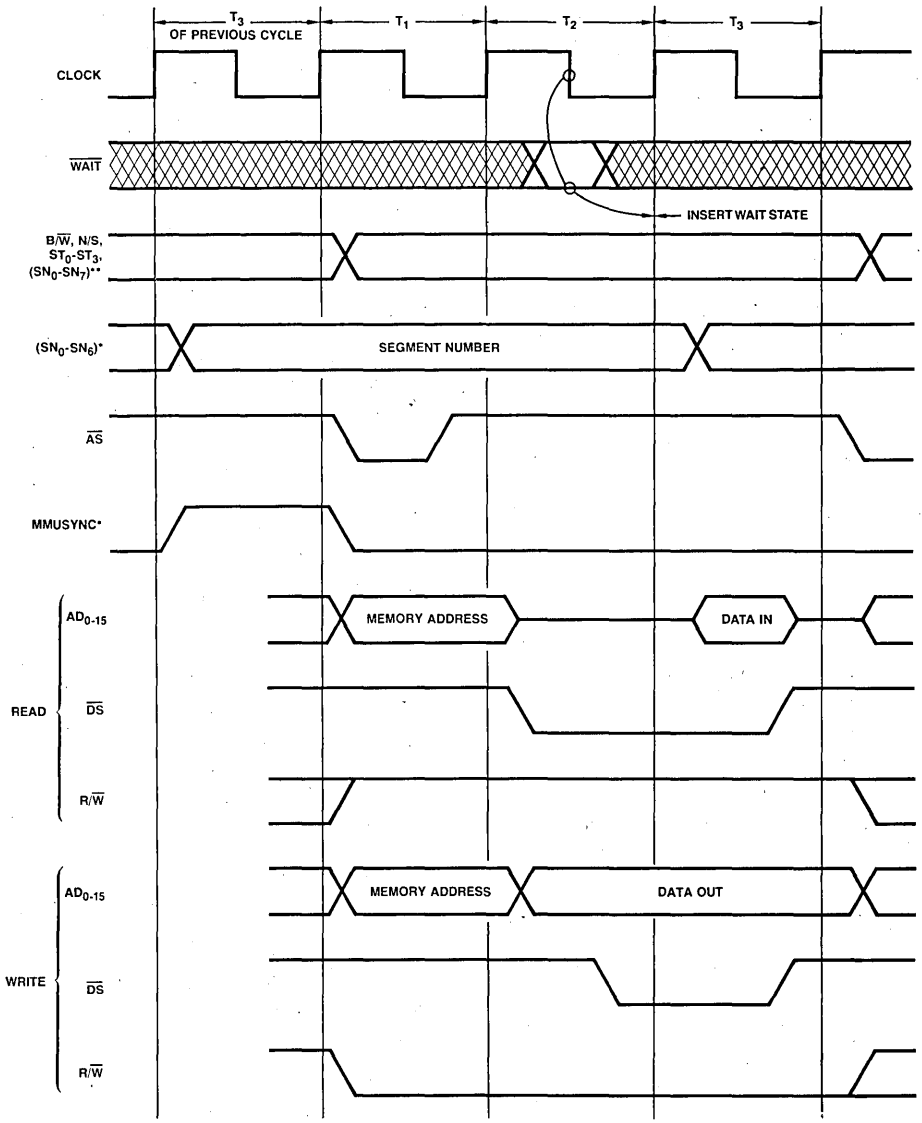
The channel's status lines (ST_0-ST_3) and the N/S line are coded as specified by the Current ARA or ARB Tag field. The B/$\overline{W}$ line indicates the operand size programmed in the Channel Mode registers Operations field. During state T_2 the channel drives both $\overline{DS}$ and $\overline{DACK}$ active. The "Flyby Peripheral" connected to $\overline{DACK}$ inverts R/$\overline{W}$ to determine whether it is being read from or written to.

The $\overline{DACK}$ input serves two purposes: To select the peripheral for the Read/Write, and to provide timing information on when to drive data onto or input data from the AD_0-AD_{15} bus. Note that because the "Flyby Peripheral" never gets explicitly addressed by AD_0-AD_{15}, it must know which internal register to load from or drive onto the AD_0-AD_{15} bus. On state T_3, the $\overline{DS}$ and $\overline{DACK}$ lines are driven inactive to conclude the transfer. In Transfer-and-Search mode, data is loaded into the DTC's Temporary register on the LOW-to-HIGH $\overline{DS}$ transition in order to perform the search function.

To provide adequate data setup time the rising edge of $\overline{DS}$ or $\overline{DACK}$ should be the edge used to perform the write to the transfer destination. To extend the active time of $\overline{DS}$ and $\overline{DACK}$, wait states can be inserted between T_2 and T_3. Software wait states can be inserted by programming the appropriate code in the Tag field of the Current ARA or ARB registers. Hardware wait states can be inserted by pulling $\overline{CS}$/$\overline{WAIT}$ LOW if the Wait Line Enable bit in the Master Mode register is set. The $\overline{CS}$/$\overline{WAIT}$ line is sampled in the middle of the T_2 and TWA states.

TERMINATION

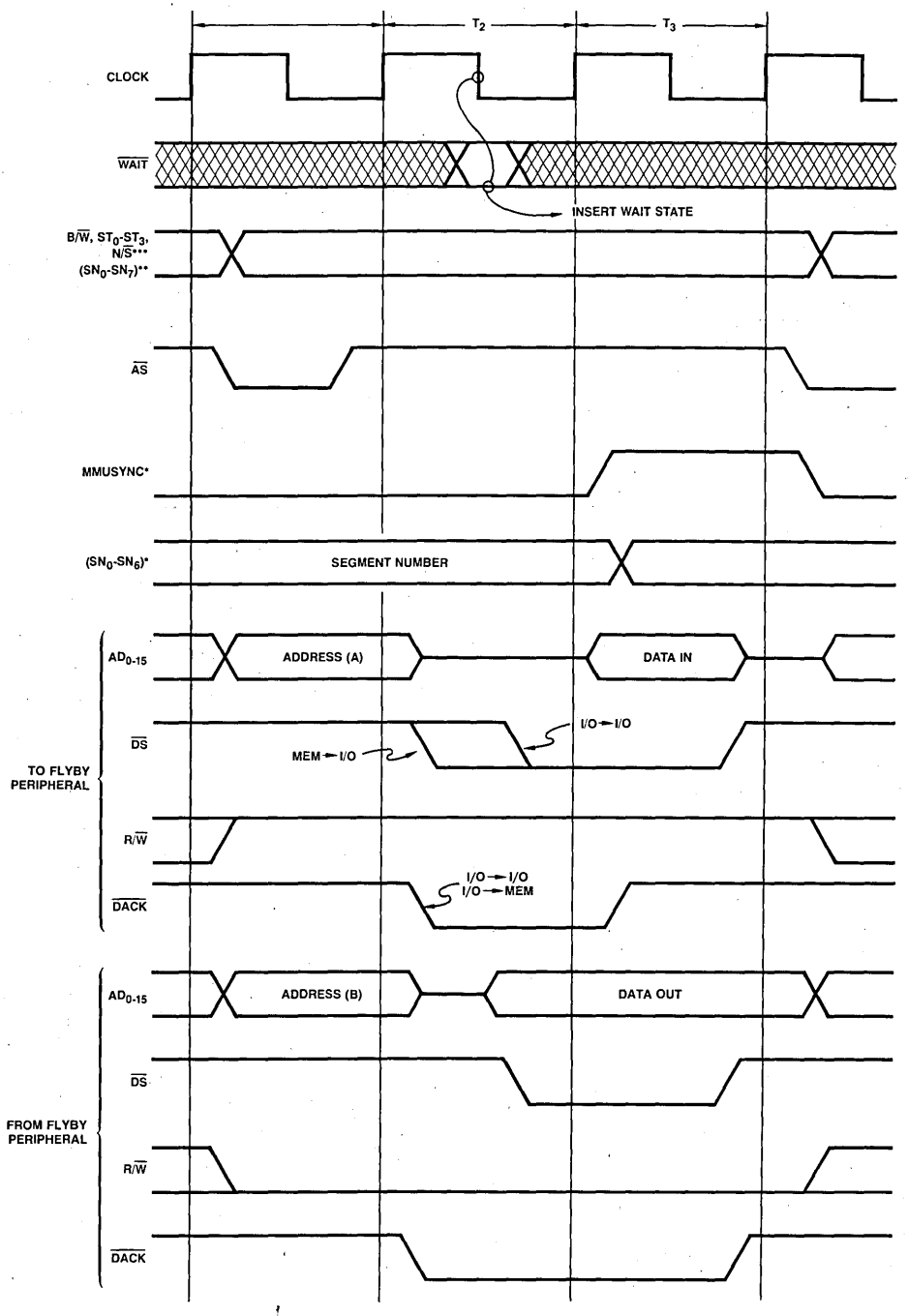
There are three ways a Transfer-and-Search or Search operation can end and two ways a Transfer operation can end. When a channel's Current Operation Count goes to 0, the DMA operation being performed will end. This is called a TC or Terminal Count termination. A DMA operation can also be stopped by driving the $\overline{EOP}$ pin LOW with external logic. This is called an EOP termination. Search and Transfer-and-Search operations have a third method of terminating called Match Condition or MC termination. An MC termination occurs when the data being Transferred-and-Searched or Searched meets the match condition programmed in Channel Mode register bits $CM_{17}-CM_{16}$. These bits allow the user to stop when a match occurs between the unmasked Pattern register bits and the data read from the source, or when a no-match occurs. Both byte and word matches are supported. MC terminations do not apply to Transfer operations since the pattern matching logic is disabled in Transfer mode.



*For logical addressing only
**For physical addressing only

Timing Diagram 5. Flowthru Memory Transactions

AIZ-035



*For logical addressing only

**For physical addressing only

***N/S will be LOW for I/O to I/O Transactions

(A) Address is current ARA

(B) Address is current ARB

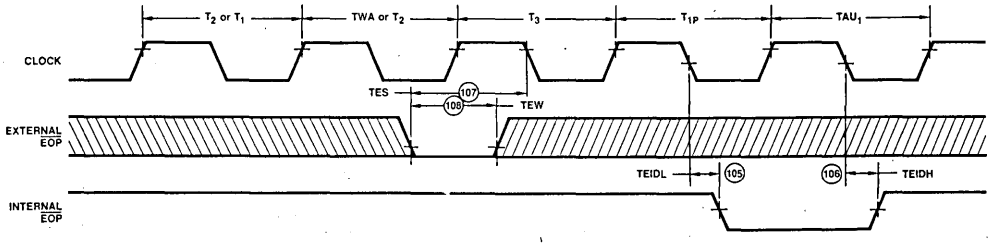
Timing Diagram 6. Flyby Transaction

AIZ-036

End-of-Process

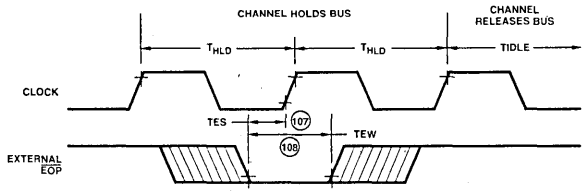
The End-of-Process ($\overline{EOP}$) interface pin is a bi-directional signal. Whenever a TC, MC or EOP termination occurs, the DTC will drive the EOP pin LOW. During DMA operations, the $\overline{EOP}$ pin is sampled by the DTC to determine if $\overline{EOP}$ is being driven LOW by external logic. Timing Diagram 7 shows when internal $\overline{EOP}$s are generated marking termination of all Transfers. These figures also show the point during the DMA iteration when the $\overline{EOP}$ pin is sampled. The generation of internal $\overline{EOP}$s and sampling of external $\overline{EOP}$s for Transfer-and-Searches follows the same timing used for Transfers. Since there is a single $\overline{EOP}$ pin for both channels, $\overline{EOP}$ should only be driven LOW by a channel while that channel is being serviced. This can be accomplished by selecting a level $\overline{DACK}$ output (CMR 18 = 0) and gating each channel's $\overline{EOP}$ request with $\overline{DACK}$, as shown in Figure 18.

Some users may connect the $\overline{EOP}$ pin to the MMU's suppress ($\overline{SUP}$) output in order to detect the illegal memory accesses. To allow this abort feature for all memory accesses, the DTC samples $\overline{EOP}$ during chaining, as shown in Timing Diagram 7 (A). If $\overline{EOP}$ is LOW, the Chain Aborted bit in the active channel's Status register is set, the channel relinguishes bus control and the chan-



a) $\overline{EOP}$ Sampling and Generation During DMA Operations

AIZ-037



b) Sampling of $\overline{EOP}$ During Bus Hold

AIZ-038

Notes: 1. The diagram lists state names for both I/O and memory accesses. Sampling of $\overline{EOP}$ will occur on the falling edge of state T_3.
2. State T_{1P} is a pseudo-T_1 state, without active $\overline{AS}$, generated following termination of any DMA operation.
3. State TAU_1 is an auto-initialization state, generated following the TC, MC or EOP termination.

Timing Diagram 7. $\overline{EOP}$ Timing

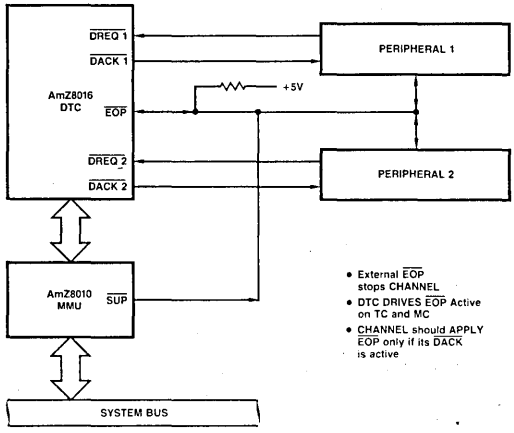


Figure 18. $\overline{EOP}$ Connection

AIZ-039

nel is inhibited from performing a new DMA operation until either a new Chain Address Segment-Tag word or offset word is loaded.

The old Chain Address Offset and Segment may be examined to determine the error-causing address. If an $\overline{EOP}$ is detected while the channel is trying to reload the Chain Address register, the new Chain Address Offset and Segment are discarded and the old address+2 is preserved to allow inspection of the erroneous address.

Programming Completion Options

When a channel ends a DMA operation, the reason for ending is stored in the Completion Status Field of the channel's Status register. See Figure 7. This information is retained until the next DMA operation ends at which time the Status register is updated to reflect the reason(s) for the latest termination. Note that it is conceivable that more than one bit in the Completion Field could be set. As an extreme example, if a channel decremented it's Current Operation Count to zero, causing a TC termination; input data from the source generated a match causing an MC termination; and there was a LOW on $\overline{EOP}$ resulting in an EOP termination, all three of the channel's Status register completion bits would be set.

When a DMA operation ends, the channel can:

(a) Interrupt the host CPU;
(b) Perform Base-to-Current reloading;
(c) Chain reloaded the next DMA operation;
(d) Perform any combination of the above; or
(e) None of the above.

The user selects the action to be performed by the channel in the Completion option field of the Channel Mode register. For each type of termination (TC, MC or EOP) the user can choose which action or actions are to be taken. If no actions are selected for the type of termination that occured, the NAC bit in the Status register will be set.

More than one action can occur when a DMA operation ends. This may arise because more than one action was programmed for the applicable termination. They occur in the following order: interrupt, base-to-current reloading, and then chaining.

Interrupts

In order to allow the DTC to start executing a new DMA operation after issuing an interrupt, but before an interrupt acknowledge is received, a two-deep interrupt queue is implemented on each channel. The following discussion will describe the standard AmZ8000 interrupt structure and then elaborate on the additional interrupt queuing capability of the DTC.

A complete interrupt cycle consists of an interrupt request followed by an interrupt-acknowledge transaction. The request, which consists of $\overline{INT}$ being pulled LOW by a peripheral, notifies the CPU that an interrupt is pending. The interrupt-acknowledge transaction, which is initiated by the CPU as a result of the request, performs two functions: it selects the peripheral whose interrupt is to be acknowledged, and it obtains a vector that identifies the selected device and cause of interrupt.

A peripheral can have one or more sources of interrupt. Each interrupt source has three bits that control how it generates interrupts. These bits are a Channel Interrupt Enable bit (CIE), an Interrupt Pending bit (IP) and an Interrupt Under Service bit (IUS). On the DTC, each channel is an interrupt source. The three interrupt control bits are located in bits $CM_{15}-CM_{13}$ of each channel's Status register.

Each channel has its own vector register for identifying the source of the interrupt during an interrupt acknowledge transaction. There are two bits in the Master Mode register used for controlling interrupt behavior for the whole device. These are a Disable Lower Chain bit (DLC), and a No Vector bit (NV).

Peripherals are connected together via an interrupt daisy chain formed with their IEI and IEO pins (See Figure 19). The interrupt sources within a device are similarly connected into this chain. The overall effect is a daisy chain connecting all the interrupt sources. The daisy chain has two functions: during an interrupt-acknowledged transaction it determines which interrupt source is being acknowledged; at all other times it determines which interrupt sources can initiate an interrupt request.

Figure 20 is a state diagram for interrupt processing for an interrupt source. An interrupt source with an interrupt pending (IP = 1) makes an interrupt request (by pulling $\overline{INT}$ LOW) if, all of the following conditions are met: It is enabled (CIE = 1), is does not have an interrupt under service (IUS = 0), no higher priority interrupt is being serviced (IEI is HIGH), and no interrupt-acknowledge transaction is in progress. IEO is not pulled down by the interrupt source at this time; IEO continues to follow IEI until an interrupt-acknowledge transaction occurs.

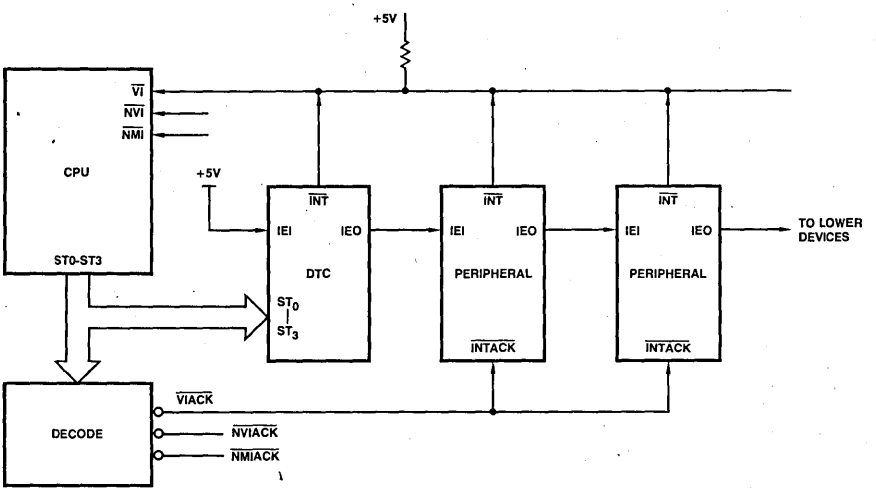


Figure 19. Interrupt Daisy Chain

AIZ-040

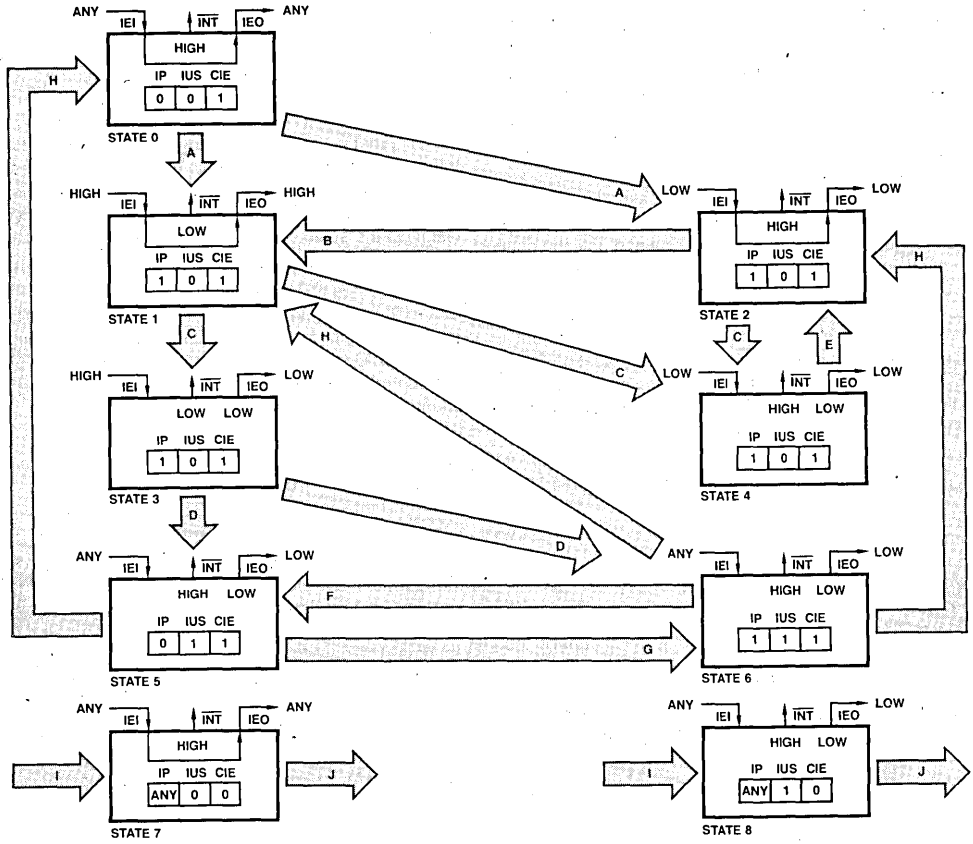


Figure 20. State Diagram for Interrupt Processing AIZ-041

Transition Legend

- A The peripheral detects an interrupt condition and sets Interrupt Pending.
- B All higher priority peripherals finish interrupt service, thus allowing IEI to go High.
- C An interrupt-acknowledge transaction starts and the IEI/IEO daisy chain settles.
- D The interrupt-acknowledge transaction terminates with the peripheral selected. Interrupt Under Service (IUS) is set to 1, and Interrupt Pending (IP) may or may not be reset.
- E The interrupt-acknowledge transaction terminates with a higher priority device having been selected.
- F The Interrupt Pending bit in the peripheral is reset by an I/O operation.
- G A new interrupt condition is detected by the peripheral, causing IP to be set again.
- H Interrupt service is terminated for the peripheral by resetting IUS.
- I CIE is reset to zero, causing interrupts to be disabled (Note 1).
- J CIE is set to one, re-enabling interrupts (Note 2).

State Legend

- 0 No interrupts are pending or under service for this peripheral.
- 1 An interrupt is pending, and an interrupt request has been made by pulling $\overline{INT}$ Low.
- 2 An interrupt is pending, but no interrupt request has been made because a higher priority peripheral has an interrupt under service, and this has forced IEI Low.
- 3 An interrupt-acknowledge sequence is in progress, and no higher priority peripheral has a pending interrupt.
- 4 An interrupt-acknowledge sequence is in progress, but a higher priority peripheral has a pending interrupt, forcing IEI Low.
- 5 The peripheral has an interrupt under service. Service may be temporarily suspended (indicated by IEI going Low) if a higher priority device generates an interrupt.
- 6 This is the same as State 5 except that an interrupt is also pending in the peripheral.
- 7 Interrupts are disabled from this source because CIE = 0.
- 8 Interrupts are disabled from this source and lower priority sources because CIE = 0 and IUS = 1.

Notes: 1. Transition I to state 6 or 7 can occur from any state except 3 or 4 (which only occur during interrupt acknowledge).

2. Transition J from state 6 or 7 can be to any state except 3 or 4, depending on the value of IEI, IP, and IUS.

Most AmZ8000 peripherals have an $\overline{INTACK}$ pin to signal when an interrupt acknowledge cycle is being performed. The DTC uses a different approach of monitoring status lines ST_0-ST_3 to detect interrupt acknowledge cycles. It is important that the Master Mode register bits MM_6 and MM_7 be programmed to select the IEI, IEO daisy chain that the DTC is located on. Some time after $\overline{INT}$ has been pulled LOW, the CPU initiates an interrupt-acknowledge transaction. Between the rising edge of $\overline{AS}$ and the falling edge of $\overline{DS}$, the IEI/IEO daisy chain settles.

Once a channel issues an interrupt, it is desirable to allow the channel to proceed with the next DMA operation before the interrupt is acknowledged. This could lead to problems if the DTC channel attempted to chain reload the Vector register contents. In such a situation, it may not be clear whether the old or new vector would be returned during the acknowledge. This dilemma is resolved in the DTC by providing each channel with an Interrupt Save Register. When the channel sets IP as part of the procedure followed to issue an interrupt, the contents of the Vector register and some of the Status register bits are saved in an Interrupt Save register. See Figure 8. When an Interrupt Acknowledge cycle is performed, the contents of the Interrupt Save register are driven onto the bus. Although the use of an Interrupt Save register allows the channel to proceed with a new task, problems can still potentially arise if a second interrupt is to be issued by the channel before the first interrupt is acknowledged. To avoid conflicts between the first and second interrupt, each channel has a Second Interrupt Pending (SIP) bit in its Status register. When a second interrupt is to be issued before the first interrupt is acknowledged, the SIP bit is set and the channel relinquishes the bus until an acknowledge occurs. For compatibility with polled interrupt schemes, the Interrupt save register can be read by the host CPU without wait states. As an aid to debugging a system's interrupt logic, whenever IP is set, the Interrupt Save register is loaded from the Vector and Status register.

Note that the SIP bit is transferred to the IP bit when IP is cleared by the host CPU. Whenever IEI is HIGH, CIE is set and IUS is cleared, $\overline{INT}$ will go LOW as soon as IP is set. IP can be cleared as soon as the first interrupt is acknowledged. The acknowledge will, as always, automatically set IUS.

The DTC stops driving $\overline{INT}$ low as soon as IUS is set. IUS must be cleared by the CPU before $\overline{INT}$ can be driven low for the second time.

Base-to-Current Reloading

When a channel finishes a DMA operation, the user may select to perform a Base-to-Current Reload. (Base-to-Current reloading is also referred to as Auto-reloading is this document.) In this type of reload, the Current Address Registers A and B are loaded with the data in the Base Address Registers A and B respectively, and the Current Operation Count register is loaded with the data in the Base Operation Count. The Base-to-Current reload operation facilitates repetitive DMA operations without the multiple memory accesses required by chaining. Although the channel must have bus control to perform Base-to-Current reloading, the complete reloading operation occurs in four clock cycles. Note that if the channel had to relinquish the bus because two unacknowledged interrupts were queued, it will have to regain bus control to perform any Base-to-Current reloading (or chaining, for that matter). In this case it acquires the BUS once an interrupt acknowledge is received, even if it immediately afterward will relinquish the bus because no hardware or software request is present.

Chaining

If the channel is programmed to chain at the end of a DMA operation, it will use the Chain Address register to point to a Chain Control Table in memory. The first word in the table is a Reload word, specifying the register(s) to be loaded. Following the Reload word are the data values to be transferred into register(s). Chaining is described in detail in the "Channel Initialization" section.

Because chaining occurs after Base-to-Current reloading, it is possible to reset the Current Address registers A and B and the Current Operation Count register to the values used for previous DMA operations and then chain reload one or two of these registers to some special value to be used perhaps for this DMA operation only. If the Base values are not reloaded during chaining, the channel can revert back to the Base values at a later cycle.

If an all zero Reload word is fetched during chaining, the chain operation will not reload any registers but in all other respects it will perform like any other chaining operation. Thus, the Chain Address will be incremented by 2 point to the next word in memory and at the end of the all Zero-Reload word chain operation the channel will be ready to perform a DMA operation. All zero Reload words are useful as "Stubs" to start or terminate linked lists of DMA operations traversed by chaining. On the other hand, care must be taken in their use since the channel may perform an erroneous operation if it is unintentionally started after the chaining operation.

COMMAND DESCRIPTIONS

Figure 21 shows a list of DTC commands. The commands are executed immediately after being written by the host CPU into the DTC's Command register. A description of each command follows.

Reset

This command causes the DTC to be set to the same state generated by a Hardware Reset. The Master Mode register is set to all zeros and the NAC and CA bits in each channel's Status register are set. The Chain Address should be programmed since its state may be indeterminate after a Reset. The lockout preventing channel activity is cleared by programming the Segment/Tag word or the Offset word and then issuing a "Start Chain" command.

Start Chain Channel 1/Channel 2

This command causes the selected channel to clear the No Auto-Reload or Chain (NAC) bit in the channel's Status register, and to start a chain reload operation of the channel's registers, as described in the "Channel Initialization" section. These effects will take place even if the Reload word fetched is all zeros. This command will only be honored if the Chain Abort (CA) bit and Second Interrupt Pending (SIP) bit in the Channel's Status register are clear. If either the CA or SIP bit is set, this command is disregarded.

Software Request Channel 1/Channel 2

This command sets the software request bit in the selected channel's Mode register. If the Second Interrupt Pending (SIP) bit and No Auto-Reload or Chain (NAC) bit in the channel's Status register are both clear, the channel will start executing the programmed DMA operation. If either the SIP or NAC bit is set, the channel will not start executing a DMA operation until both bits are cleared. The SIP bit will clear when the channel receives an interrupt acknowledge. One way to clear the NAC bit is to issue a Start Chain command to the channel. If the fetched Reload Word is all zeros, the channel's registers will remain unchanged and the

Command	Opcode Bits 7654	Opcode Bits 3210	Example Code (HEX)
Reset	000X	XXXX	00
Start Chain Channel 1	101X	XXX0	A0
Start Chain Channel 2	101X	XXX1	A1
Set Software Request Channel 1	010X	XX10	42
Set Software Request Channel 2	010X	XX11	43
Clear Software Request Channel 1	010X	XX00	40
Clear Software Request Channel 2	010X	XX01	41
Set Hardware Mask Channel 1	100X	XX10	82
Set Hardware Mask Channel 2	100X	XX11	83
Clear Hardware Mask Channel 1	100X	XX00	80
Clear Hardware Mask Channel 2	100X	XX01	81
Set CIE, IUS, IP Channel 1	001E	SP10	32*
Set CIE, IUS, IP Channel 2	001E	SP11	33*
Clear CIE, IUS, IP Channel 1	001E	SP00	30*
Clear CIE, IUS, IP Channel 2	001E	SP01	31*
Set Flip Bit Channel 1	011X	XX10	62
Set Flip Bit Channel 2	011X	XX11	63
Clear Flip Bit Channel 1	011X	XX00	60
Clear Flip Bit Channel 2	011X	XX01	61

*Notes: 1. E = Set to 1 to perform set/clear on CIE, Clear to 0 for no effect on CIE
2. S = Set to 1 to perform set/clear on IUS, Clear to 0 for no effect on IUS
3. P = Set to 1 to perform set/clear on IP, Clear to 0 for no effect on IP
4. X = "don't care" bit. This bit is not decoded and may be 0 or 1

Figure 21. DTC Command Summary

software request bit, if set earlier by command, will cause the programmed DMA operation to start immediately. If during chaining new information is loaded into the Channel Mode register this new information will, of course, overwrite the software request bit.

Set/Clear Hardware Mask 1/Mask 2

This command sets or clears the Hardware Mask bit in the selected channel's Mode register. This command always takes effect. The Hardware Mask bit inhibits recognition of an active signal on the channel's $\overline{DREQ}$ input; this bit does not affect recognition of a software request. If the channel is in single transfer mode, it performs DMA operations upon receipt of a transition on $\overline{DREQ}$ rather than in response to a $\overline{DREQ}$ level, the transition occurring while the Hardware Mask bit is set ($CM_{19} = 1$) will be stored and serviced when the Hardware Mask is cleared (assuming the Channel has not chained in the interim). The DTC will request the system bus 1-1/2 to 2 clocks after the receipt of any $\overline{DREQ}$, after which a minimum of one DMA iteration is unavoidable. $\overline{DREQ}$ transitions are only stored for the current DMA operation. If the channel performs a chain operation of single transfer mode, any $\overline{DREQ}$ transition stored for later servicing is cleared. Timing Diagrams 1 and 2 show the minimum times after each of these events a new $\overline{DREQ}$ edge can be applied if it is to be serviced by the new DMA operation. Note in the diagrams the notation of First iteration and Last iteration. This means for example, $\overline{DREQ}$ may be asserted during the second T_1 of a Flow-through transaction, but may never be asserted during T_1 of a Flyby transaction since Flyby is done in one iteration.

Set/Clear CIE, IUS AND IP Channel 1/Channel 2

This command allows the user to either set or clear any combination of the CIE, IUS and IP bits in the selected channel's Status register. These bits control the operation of the channel's interrupt structure and are described in detail in the "Interrupts" section. Setting the IP bit causes the Interrupt Save register to be loaded with current Vector and Status. The IP and IUS bits can be simultaneously cleared to facilitate an efficient conclusion to the processing of an interrupt.

Set/Clear Flip Bit in Channel 1/Channel 2

The Flip Bit in the selected channel's Mode register can be cleared and set by this command. This allows the user to reverse the source and destination and thereby reverse the data transfer direction without reprogramming the channel. This command will be most useful when repetitive DMA operations are being performed by the channel, using Base-to-Current reloading for channel reinitialization and using this command to control the direction of transfer. Chaining new information into the Channel Mode register will, of course, overwrite the Flip bit.

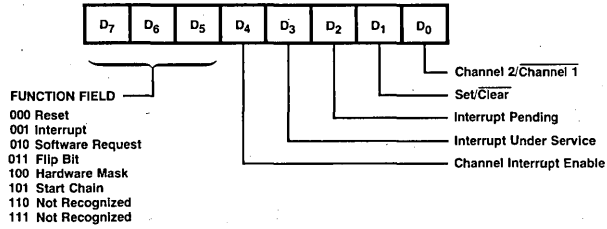


Figure 21B. Command Register

AIZ-042

APPLICATION INFORMATION

Figure 22 shows the configuration of the AmZ8016 DMA Transfer Controller (DTC) and a microprocessor system. The DTC issues a $\overline{BUSRQ}$ active low signal to the CPU to request bus control. When the CPU replies with a Bus Acknowledge $\overline{BUSAK}$ signal through the $\overline{BAI}$-$\overline{BAO}$ daisy chain to the DTC which issued the $\overline{BUSRQ}$, the DTC takes control of the Address-Data bus and the Control bus. In addition to hardware reset the 8016 can be given a software 'reset' command (i.e., loading all zero to Command Register). Two DMA channels are provided per each DTC device. The logic blocks A and B are shown in Figure 23 and 24.

Figure 23 shows the bus request logic used for bidirectionally buffering $\overline{BUSRQ}$. See 'Bus Request/Grant' section for detail. Figure 24 shows the $\overline{CS}/\overline{WAIT}$ logic for the multiplexed $\overline{CS}/\overline{WAIT}$ pin of DTC. See 'Wait States' for detailed description.

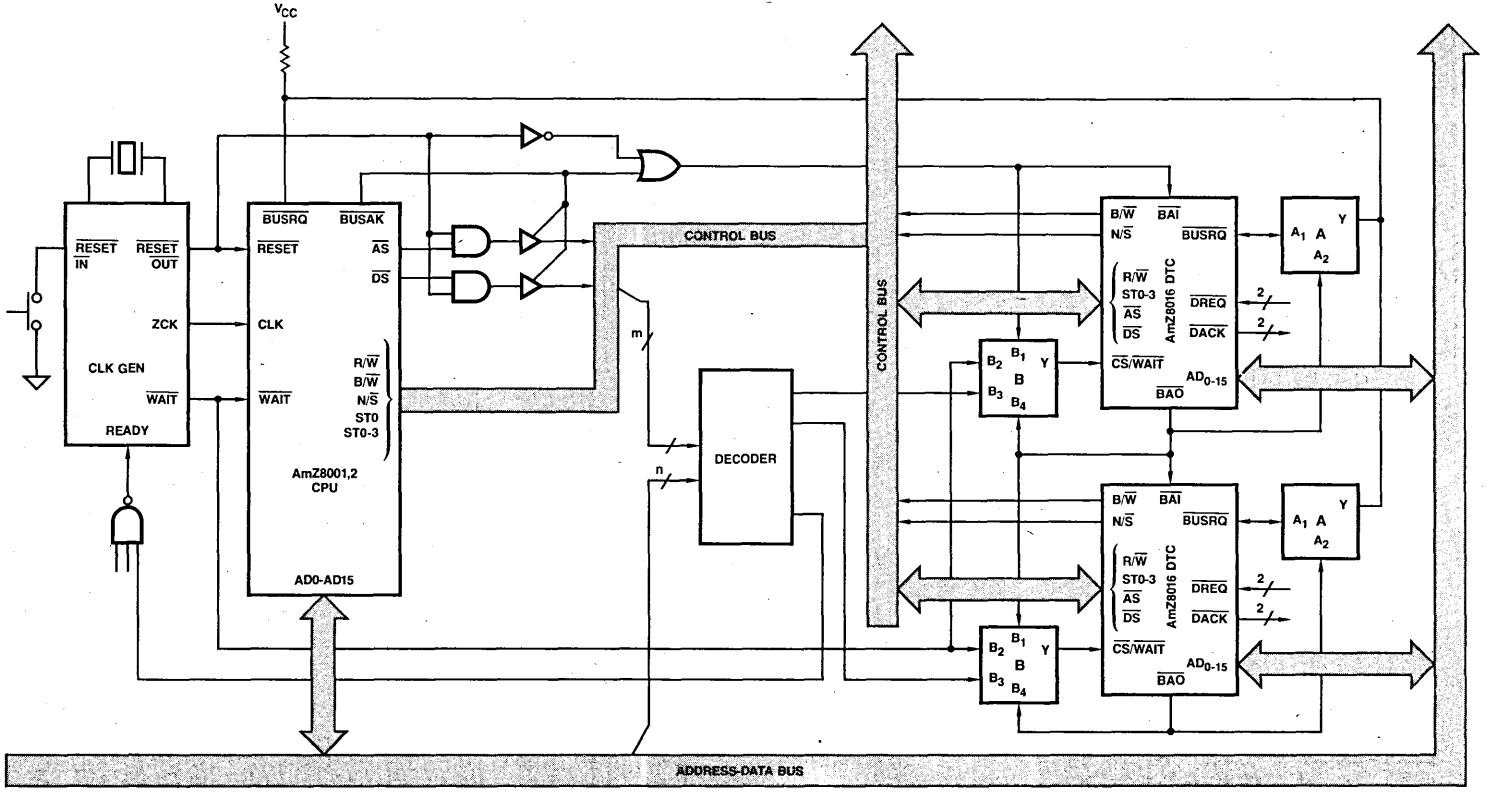
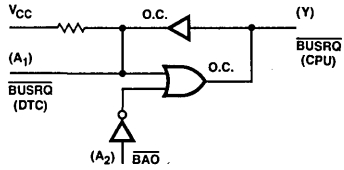
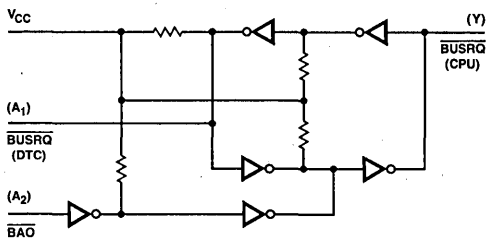


Figure 22. Basic DTC Configuration



AIZ-044

a) Using Three Different Gates



b) Using One TTL Package (7405)

AIZ-045

Figure 23. Logic used to Bidirectionally Buffer $\overline{BUSRQ}$

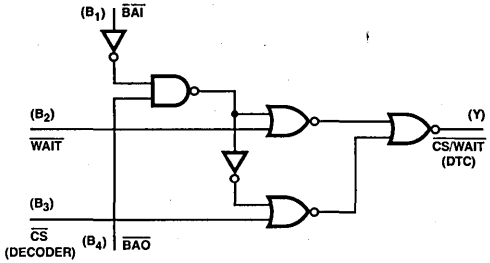
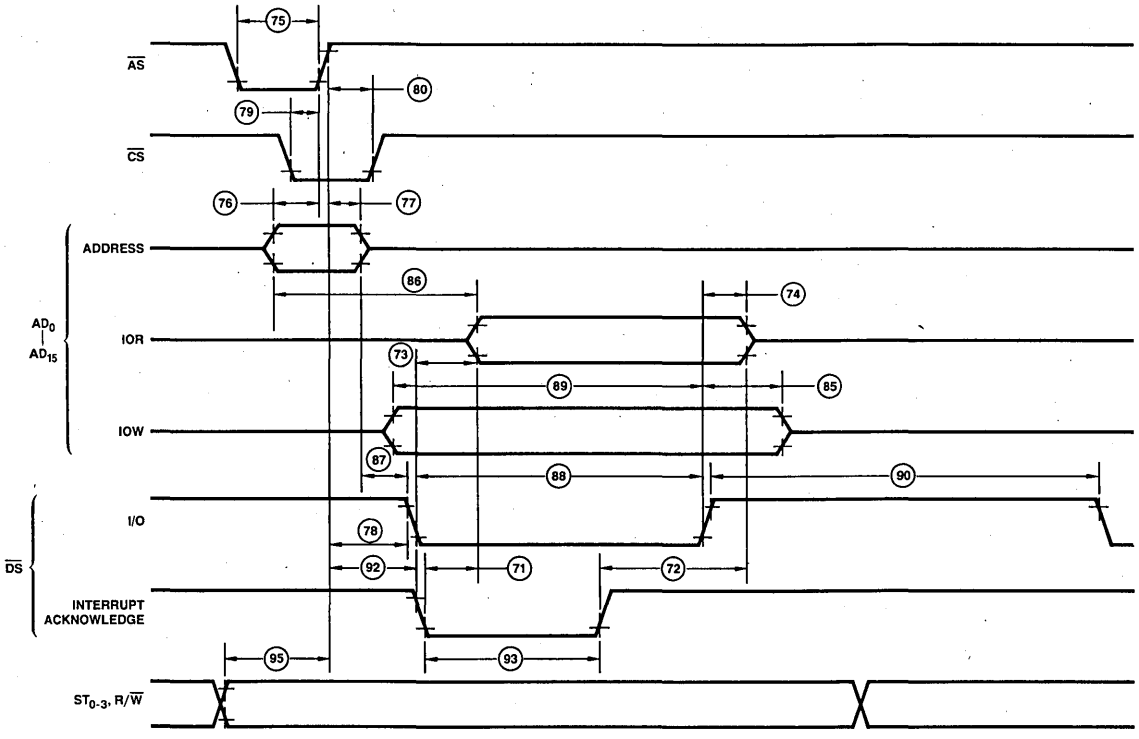


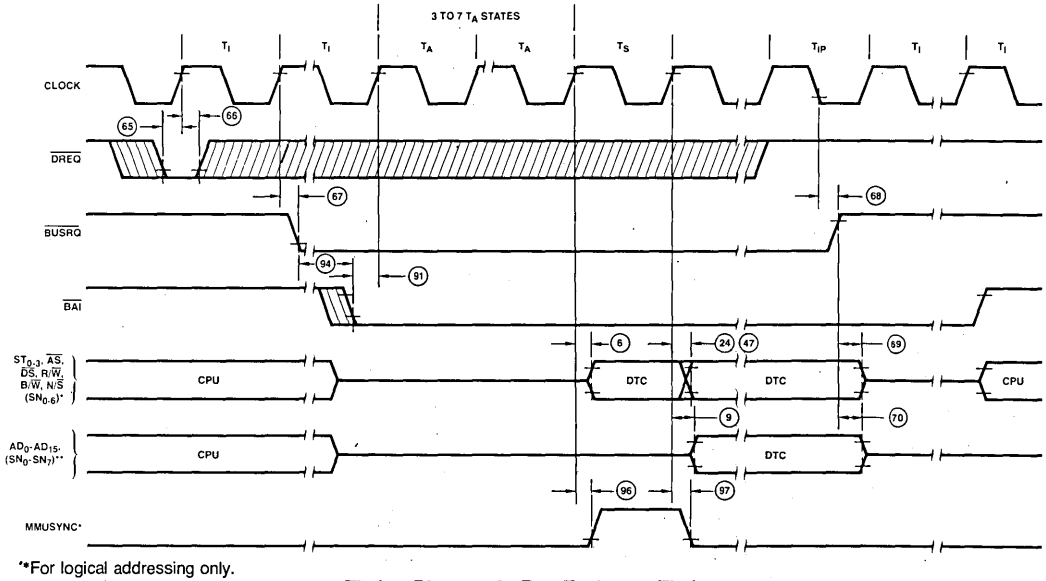
Figure 24. Logic for Multiplexed $\overline{CS/WAIT}$ pin

AIZ-046



Timing Diagram 8. AC Timing when DTC is Bus Slave

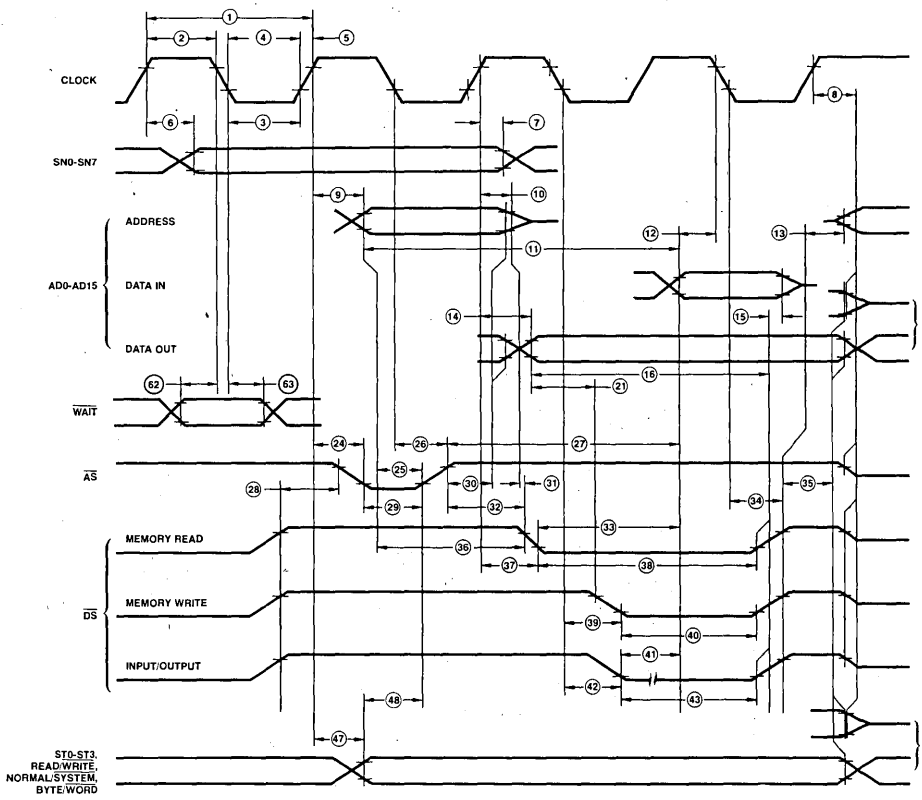
AIZ-047



*For logical addressing only.
**For physical addressing only.

Timing Diagram 9. Bus Exchange Timing

AIZ-048



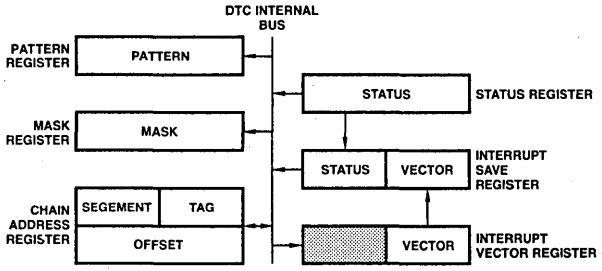
This composite timing diagram does not show actual timing sequences. Refer to this diagram only for the detailed timing relationships of individual edges. Use the preceding illustrations as an explanation of the various timing sequences.

†Timing Diagram 10. AC Timing when DTC is Bus Master

AIZ-049

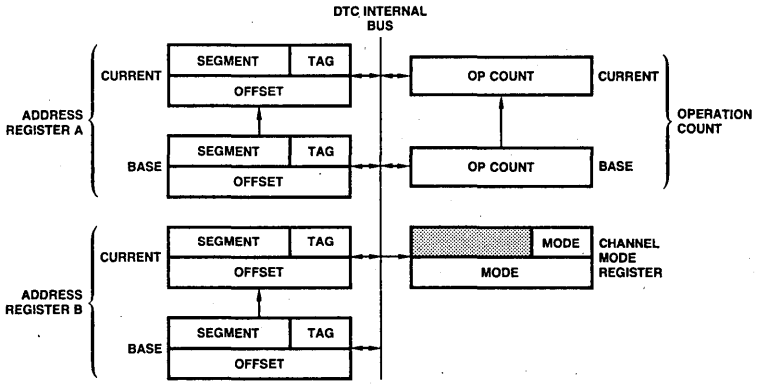
APPENDIX A
8016 REGISTER SUMMARY

Special – Purpose Channel Registers



AIZ-050

General – Purpose Channel Registers

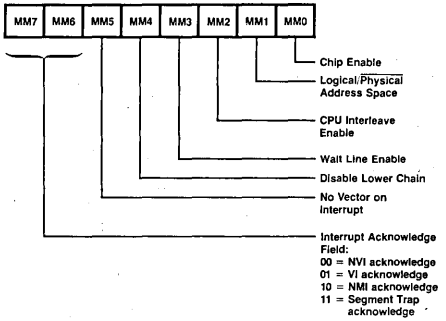


AIZ-051

Master Mode Register

	A7							A0
Address	X	0	1	1	1	0	0	X

Fast Readable
Writeable

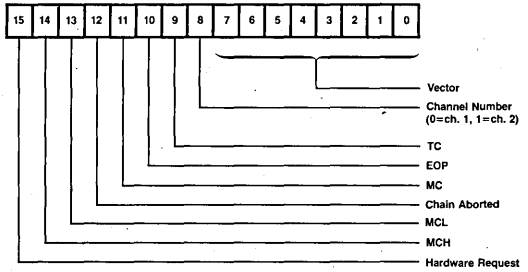


AIZ-052

Interrupt Save Register

	A7							A0	
Address	X	0	1	0	1	0	1	X	CH1
	X	0	1	0	1	0	0	X	CH2

Fast Readable



AIZ-053

Miscellaneous Registers

	A7							A0		
Address	X	0	1	1	0	0	1	X	Current Operation Count	CH1
	X	0	1	1	0	0	0	X	Current Operation Count	CH2
	X	0	1	1	0	1	1	X	Base Operation Count	CH1
	X	0	1	1	0	1	0	X	Base Operation Count	CH2
	X	1	0	0	1	0	1	X	Pattern	CH1
	X	1	0	0	1	0	0	X	Pattern	CH2
	X	1	0	0	1	1	1	X	Mask	CH1
	X	1	0	0	1	1	0	X	Mask	CH2

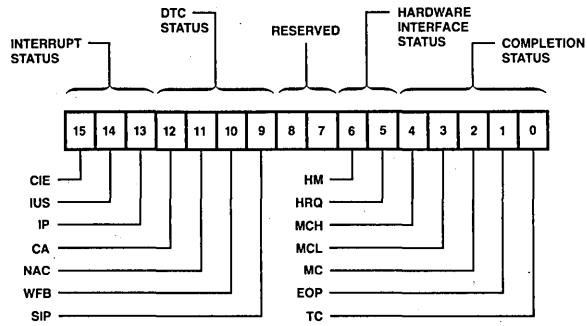
Chain Loadable
Writeable
Pattern and Mask – Slow Readable
Operation Count – Fast Readable

AIZ-054

Status Register

	A7							A0	
Address	X	0	1	0	1	1	1	X	CH1
	X	0	1	0	1	1	0	X	CH2

Fast Readable

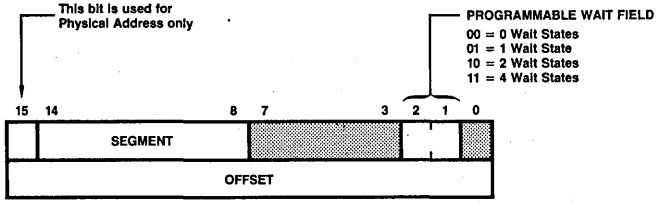


AIZ-055

Chain Address Register

	A7							A0	
Address	X	0	1	0	0	1	1	X	Segment/Tag CH1
	X	0	1	0	0	1	0	X	Segment/Tag CH2
	X	0	1	0	0	0	1	X	Offset CH1
	X	0	1	0	0	0	0	X	Offset CH2

Fast Readable/Writeable
Chain Loadable

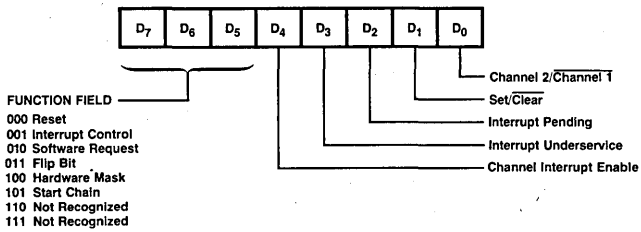


AIZ-056

Command Register

	A7							A0
Address	X	0	1	0	1	1	X	X

Writeable Only

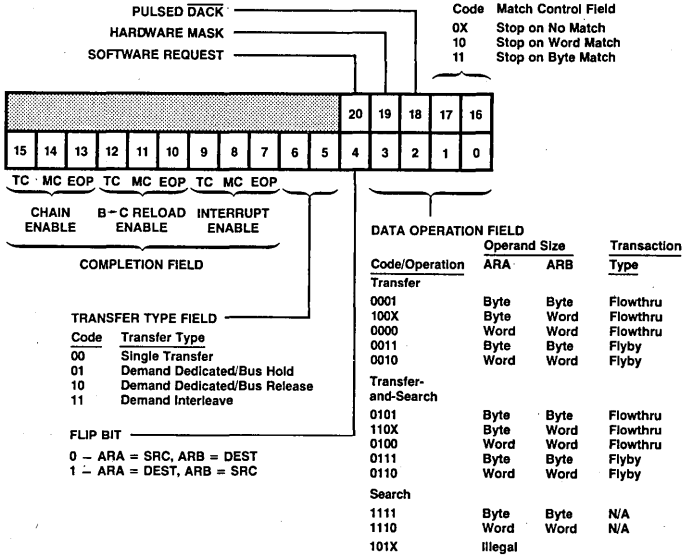


AIZ-057

Channel Mode Register

A7							A0	
X	1	0	1	0	1	1	X	High CH1
X	1	0	1	0	1	0	X	High CH2
X	1	0	1	0	0	1	X	Low CH1
X	1	0	1	0	0	0	X	Low CH2

Chain Loadable
Writable (Lower 16 bits)
Slow Readable



Code	Match Control Field
0X	Stop on No Match
10	Stop on Word Match
11	Stop on Byte Match

Code	Transfer Type
00	Single Transfer
01	Demand Dedicated/Bus Hold
10	Demand Dedicated/Bus Release
11	Demand Interleave

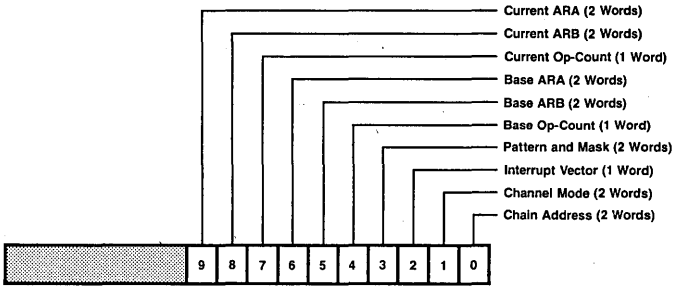
DATA OPERATION FIELD

Code/Operation	Operand Size ARA	Operand Size ARB	Transaction Type
Transfer			
0001	Byte	Byte	Flowthru
100X	Byte	Word	Flowthru
0000	Word	Word	Flowthru
0011	Byte	Byte	Flyby
0010	Word	Word	Flyby
Transfer-and-Search			
0101	Byte	Byte	Flowthru
110X	Byte	Word	Flowthru
0100	Word	Word	Flowthru
0111	Byte	Byte	Flyby
0110	Word	Word	Flyby
Search			
1111	Byte	Byte	N/A
1110	Word	Word	N/A
101X	Illegal		

AIZ-058

Chain Control Register

Chain Loadable Only

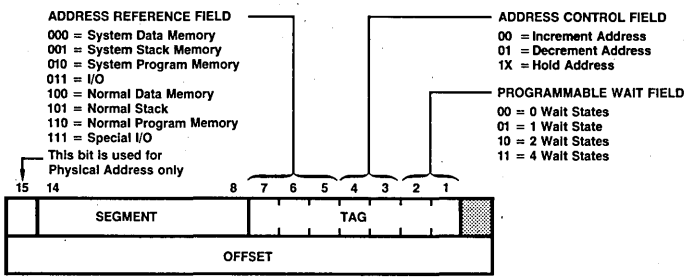


AIZ-059

Address Register

	A7							A0		
Address	X	0	0	1	1	0	1	X	Current ARA Segment/Tag	CH1
	X	0	0	1	1	0	0	X	Current ARA Segment/Tag	CH2
	X	0	0	0	1	0	1	X	Current ARA Offset	CH1
	X	0	0	0	1	0	0	X	Current ARA Offset	CH2
	X	0	0	1	0	0	1	X	Current ARB Segment/Tag	CH1
	X	0	0	1	0	0	0	X	Current ARB Segment/Tag	CH2
	X	0	0	0	0	0	1	X	Current ARB Offset	CH1
	X	0	0	0	0	0	0	X	Current ARB Offset	CH2
	X	0	0	1	1	1	1	X	Base ARA Segment/Tag	CH1
	X	0	0	1	1	1	0	X	Base ARA Segment/Tag	CH2
	X	0	0	0	1	1	1	X	Base ARA Offset	CH1
	X	0	0	0	1	1	0	X	Base ARA Offset	CH2
	X	0	0	1	0	1	1	X	Base ARB Segment/Tag	CH1
	X	0	0	1	0	1	0	X	Base ARB Segment/Tag	CH2
	X	0	0	0	0	1	1	X	Base ARB Offset	CH1
	X	0	0	0	0	1	0	X	Base ARB Offset	CH2

Chain Loadable
Fast Readable and Writeable



AIZ-060

ORDERING INFORMATION

	4MHz	6MHz	Temperature Range	Package Type
	AmZ8016	AmZ8016A		
Commercial Operating Range T_A = 0 to +70°C V_{CC} = 5V ± 5%	AmZ8016DC AmZ8016PC	AmZ8016ADC AmZ8016APC	C = 0 to +70°C I = −40 to +85°C M = −55 to +125°C	D = Hermetic DIP P = Plastic
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ± 10%	AmZ8016DI			
Military Operating Range T_A = −55 to + 125°C	AmZ8016DM			

Notes: T_A denotes ambient temperature.
Add suffix B to indicate burn-in requirement.

ELECTRICAL CHARACTERISTICS over operating range

Parameter	Description	Test Conditions	Min	Max	Units
V_{CH}	Clock Input High Voltage	Driven by External Clock Generator	V_{CC} −0.4	V_{CC} +0.3	Volts
V_{CL}	Clock Input Low Voltage	Driven by External Clock Generator	−0.3	0.45	Volts
V_{IH}	Input High Voltage		2.0	V_{CC} +0.3	Volts
V_{IL}	Input Low Voltage		−0.3	0.8	Volts
V_{OH}	Output High Voltage	I_{OH} = −250μA	2.4		Volts
V_{OL}	Output Low Voltage	I_{OL} = +2.0mA		0.4	Volts
I_{IL}	Input Leakage	$V_{SS} \leq V_{IN} \leq V_{CC}$		±10	μA
I_{OL}	Output Leakage	$V_{SS} \leq V_{OUT} \leq V_{CC}$		±10	μA
I_{CC}	V_{CC} Supply Current	T_A = 0°C		350	mA
		T_A = 75°C		200	mA
C_{IN}	Input Capacitance	Unmeasured pins returned to ground. f = 1MHz over specified temperature range.		10	pF
C_{OUT}	Output Capacitance			15	pF
$C_{I/O}$	Bidirectional Capacitance			20	pF

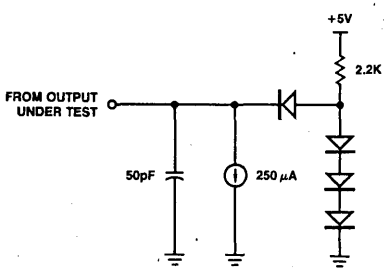
V_{CC} = 5V ±5% unless otherwise specified.

Standard Test Conditions

The characteristics 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:

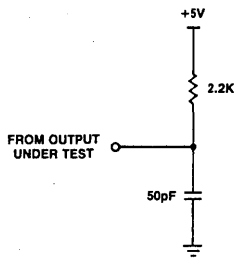
$+4.75V \leq V_{CC} \leq +5.25V$
GND = 0V
$0°C \leq T_A \leq +70°C$

Standard Test Load



AIZ-021

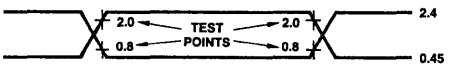
Open-Drain Test Load



AIZ-022

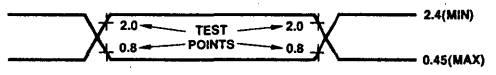
TIMING REFERENCES FOR AC TESTS

Input Waveform



AIZ-023

Output Waveform



AIZ-024

All AC parameters assume a load capacitance of 100pF max, except for parameter 6 TdC(SNv) (50pF max).

SWITCHING CHARACTERISTICS

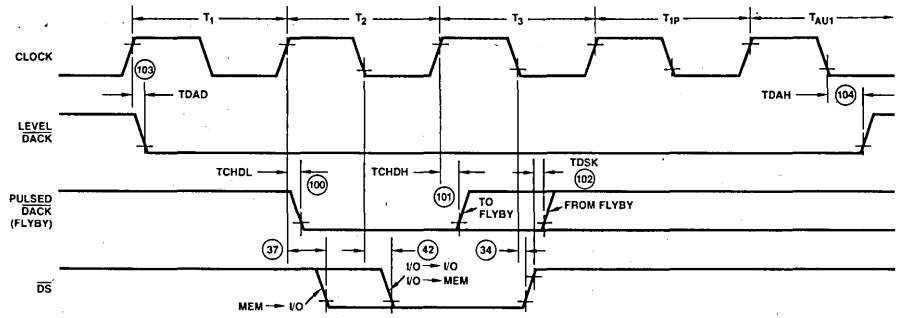
TIMING FOR DTC-PERIPHERAL INTERFACE

Number	Parameter	Description	4MHz Min	4MHz Max	6MHz Min	6MHz Max	Units
100	TCHDL	Clock RE to Pulsed $\overline{DACK}$ FE Delay (Flyby Transactions Only)		100			ns
101	TCHDH	Clock RE to Pulsed $\overline{DACK}$ RE Delay (Transactions TO Flyby Peripheral Only)		100			ns
102	TDSK	$\overline{DS}$ RE to Pulsed $\overline{DACK}$ RE Delay (Transactions FROM Flyby Peripheral Only)	20				ns
103	TDAD	Clock RE to Level $\overline{DACK}$ Valid Delay		100			ns
104	TDAH	Clock FE to Level $\overline{DACK}$ Valid Delay		110			ns
105	TEIDL	Clock FE to Internal $\overline{EOP}$ FE Delay		100			ns
106	TEIDH	Clock FE to Internal $\overline{EOP}$ RE Delay		100			ns
107	TES	External $\overline{EOP}$ Valid to Clock FE Setup Time During Operation	10				ns
108	TEW	External $\overline{EOP}$ Pulse Width Required During Operation	20				ns
109	TES(BH)	External $\overline{EOP}$ Valid to Clock RE Setup Time During Bus Hold	10				ns
110	TEW(BH)	External $\overline{EOP}$ Pulse Width Required During Bus Hold	20				ns

PRELIMINARY

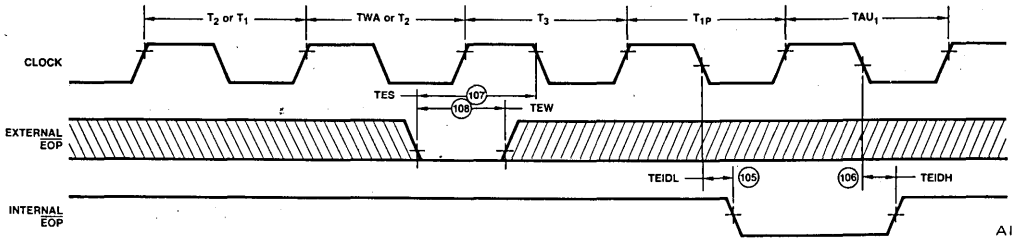
Note: RE = rising edge FE = falling edge

Timing Diagram 11. $\overline{DACK}$ Timing

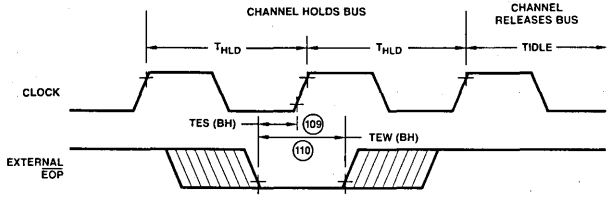


*LEVEL $\overline{DACK}$ RE occurs as shown if auto-reloading is not programmed. LEVEL $\overline{DACK}$ stays LOW for three additional clocks for reloading.

Timing Diagram 12a). $\overline{EOP}$ Sampling and Generation during DMA Operations



Timing Diagram 12b). Sampling of $\overline{EOP}$ during Bus Hold



Notes: 1. The diagram lists state names of both I/O and memory accesses. Sampling of $\overline{EOP}$ will occur on the falling edge of state T_3.
2. State T_{1P} is a pseudo-T_1 state, without an active $\overline{AS}$ generated following termination of any DMA operation.
3. State TAU_1 is an auto-initialization state, generated following the TC, MC or EOP termination.

SWITCHING CHARACTERISTICS

TIMING FOR DTC AS BUS MASTER

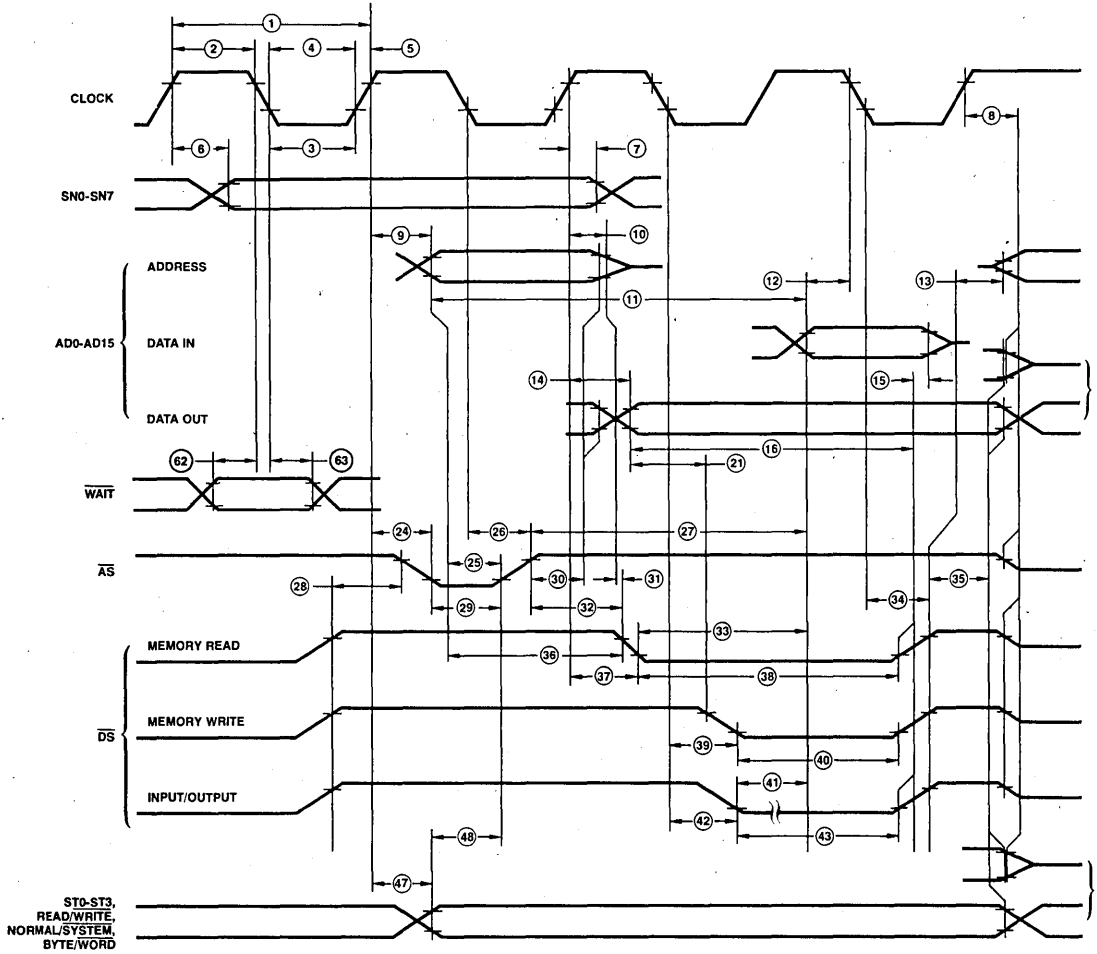
Number	Symbol	Description	4MHz Min	4MHz Max	6MHz Min	6MHz Max	Units
1	TcC	Clock Cycle Time	250	2000			ns
2	TwCh	Clock Width (High)	105				ns
3	TwCl	Clock Width (LOW)	105				ns
4	TfC	Clock Fall Time		20			ns
5	TrC	Clock Rise Time		20			ns
6	TdC(SNv)	Clock RE to Segment Number Valid (50pF Load) Delay***		110			ns
7	TdC(SNn)	Clock RE to Segment Number Valid Delay	20				ns
8	TdC(Bz)	Clock RE to Bus Float Delay		65			ns
9	TdC(A)	Clock RE to Address Valid Delay		90			ns
10	TdC(Az)	Clock RE to Address Float Delay		65			ns
11	TdA(DI)	Address Valid to Data In Required Valid Delay	400				ns
12	TsDI(C)	Data In to Clock FE Setup Time	20				ns
13	TdDS(A)	$\overline{DS}$ RE to Address Active Delay	80				ns
14	TdC(DO)	Clock RE to Data Out Valid Delay		90			ns
15	ThDI(DS)	$\overline{DS}$ RE to Data In Hold Time	0				ns
16	TdDO(DS)	Data Out Valid to $\overline{DS}$ RE Delay	230				ns
21	TdDO(SW)	Data Out Valid to $\overline{DS}$ FE (Write) Delay	55				ns
24	TdC(ASf)	Clock RE to $\overline{AS}$ FE Delay		70			ns
25	TdA(AS)	Address Valid to $\overline{AS}$ RE Delay	50				ns
26	TdC(ASr)	Clock FE to $\overline{AS}$ RE Delay		70			ns
27	TdAS(DI)	$\overline{AS}$ RE to Data In Required Valid Delay	300				ns
28	TdDS(AS)	$\overline{DS}$ RE to $\overline{AS}$ FE Delay	75				ns
29	TwAS	$\overline{AS}$ Width (LOW)	80				ns
30	TdAS(A)	$\overline{AS}$ RE to Address Valid Delay	60				ns
31	TdAz(DSR)	Address Float to $\overline{DS}$ (Read) FE Delay	0				ns
32	TdAS(DSR)	$\overline{AS}$ RE to $\overline{DS}$ FE (Read) Delay	75				ns
33	TdDSR(DI)	$\overline{DS}$ (Read) FE to Data In Required Valid Delay	165				ns
34	TdC(DSr)	Clock FE to $\overline{DS}$ RE Delay		70			ns
35	TdDS(DO)	$\overline{DS}$ RE to Data Out (Write Only) and Status Valid (Read and Write) Delay	85				ns
36	TdA(DSR)	Address Valid to $\overline{DS}$ (Read) FE Delay	120				ns
37	TdC(DSR)	Clock RE to $\overline{DS}$ (Read) FE Delay		60			ns
38	TwDSR	$\overline{DS}$ (Read) Width (LOW)	275				ns
39	TdC(DSW)	Clock FE to $\overline{DS}$ (Write) FE Delay		60			ns
40	TwDSW	$\overline{DS}$ (Write) Width (LOW)	160				ns
41	TdDSI(DI)	$\overline{DS}$ (Input) FE to Data In Required Valid Delay	325				ns
42	TdC(DSf)	Clock FE to $\overline{DS}$ (I/O) FE Delay		60			ns
43	TwDS	$\overline{DS}$ (I/O) Width (LOW)	150*				ns
47	TdC(S)	Clock RE to Status Valid Delay		110			ns
48	TdS(AS)	Status Valid to $\overline{AS}$ RE Delay	60				ns
62	TsWT(C)	$\overline{WAIT}$ to Clock FE Setup Time	20				ns
63	ThWT(C)	$\overline{WAIT}$ to Clock FE Hold Time	30				ns
96	TdC(SNr)	Clock RE to SN7/MMUSync RE Delay**		110			ns
97	TdC(SNf)	Clock RE to SN7/MMUSync FE Delay**	20	110			ns

*Wait states should be inserted by programming a hardware when accessing slow peripherals.
**Logical Addressing only.
***130ns max with Logical Addressing.

Note: RE = rising edge
FE = falling edge

PRELIMINARY

Timing Diagram 13. AC Timing when DTC is Bus Master



This composite timing diagram does not show actual timing sequences. Refer to this diagram only for the detailed timing relationships of individual edges. Use the preceding illustrations as an explanation of the various timing sequences.

AIZ-028

SWITCHING CHARACTERISTICS

TIMING FOR DTC AS BUS SLAVE AND CPU-DTC BUS EXCHANGE

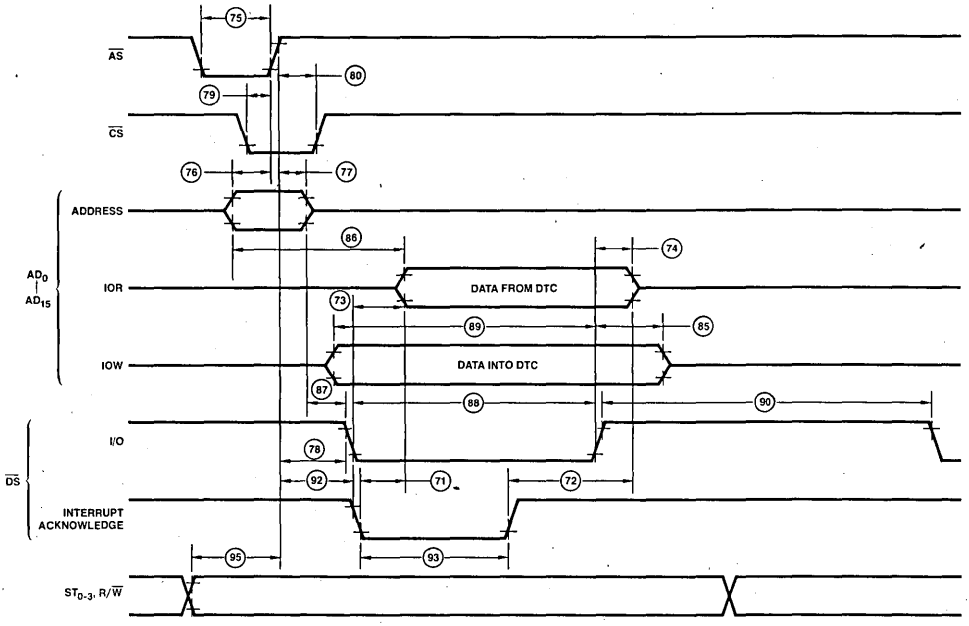
Number	Symbol	Description	4MHz Min	4MHz Max	6MHz Min	6MHz Max	Units
64	TwDRQ	$\overline{DREQ}$ Pulse Width (Single Transfer Mode)	20				ns
65	TsDRQ(c)	$\overline{DREQ}$ Valid to Clock RE Setup Time	50				ns
66	ThDRQ(C)	Clock RE to $\overline{DREQ}$ Valid Hold Time	20				ns
67	TdC(BRQf)	Clock RE to $\overline{BUSRQ}$ FE Delay		150			ns
68	TdC(BRQr)	Clock FE to $\overline{BUSRQ}$ RE Delay		150			ns
69	TdBRQ(BUSc)	$\overline{BUSRQ}$ RE to Control Bus Float Delay		140			ns
70	TdBRQ(BUSd)	$\overline{BUSRQ}$ RE to AD Bus Float Delay		140			ns
71	TdDSA(RDV)	$\overline{DS}$ FE (Acknowledge) to Data Output Valid Delay		135			ns
72	TdDSA(RDZ)	$\overline{DS}$ RE (Acknowledge) to Data Output Float Delay		80			ns
73	TdDSR(DOD)	$\overline{DS}$ FE (IOR) to Data Output Driven Delay		135			ns
74	TdDSR(RDZ)	$\overline{DS}$ RE (IOR) to Data Output Float Delay		80			ns
75	TwAS	$\overline{AS}$ Low Width	70				ns
76	TsA(AS)	Address Valid to $\overline{AS}$ RE Setup Time	30				ns
77	ThAS(Av)	$\overline{AS}$ RE to Address Valid Hold Time	50				ns
78	TdAS(DS)	$\overline{AS}$ RE to $\overline{DS}$ FE Delay (I/O)	50				ns
79	TsCS(AS)	$\overline{CS}$ Valid to $\overline{AS}$ RE Setup Time	0				ns
80	ThCS(AS)	$\overline{AS}$ RE to $\overline{CS}$ Valid Hold Time	40				ns
81	TwAS(DS)	$\overline{AS}$ and $\overline{DS}$ Simultaneously LOW Time (Reset)	3TcC				
82	TdBAI(Az)	$\overline{BAI}$ RE to SN0-SN7, AD0-AD15 Float Delay (Reset)		135			ns
83	TdBAI(ST)	$\overline{BAI}$ RE to ST0-ST3, R/$\overline{W}$, B/$\overline{W}$, N/$\overline{S}$ Float Delay (Reset)		100			
84	TdBAI(DS)	$\overline{BAI}$ RE to $\overline{DS}$, $\overline{AS}$ Float Delay (Reset)		80			ns
85	TdDS(Dn)	$\overline{DS}$ RE (IOW) to Data Valid Hold Time	30				ns
86	TdAC(DRV)	Address Valid to Data (IOR) Required Valid Delay		540			ns
87	TdAZ(DS)	Address Float to $\overline{DS}$ FE (IOR) Delay	0				ns
88	TwDS(IO)	$\overline{DS}$ (IO) LOW Width	150*				ns
89	TsD(DS)	Data (IOW) Valid to $\overline{DS}$ RE Setup Time	40				ns
90	TrDS(W)	$\overline{DS}$ RE (IOW) to $\overline{DS}$ FE (IOW) (Write Recovery Time applies only for issuing Command)	4TcC				ns
91	TsBAK(C)	$\overline{BAI}$ Valid to Clock RE Setup Time	50				ns
92	TdAS(DS)	$\overline{AS}$ RE to $\overline{DS}$ FE (ACK) Delay	100				ns
93	TwDS(AK)	$\overline{DS}$ (ACK) LOW Width	150				ns
94	TdBRQ(BAI)	$\overline{BUSRQ}$ FE to $\overline{BAI}$ FE Required Delay	0				ns
95	TsS(AS)	Status Valid to $\overline{AS}$ RE Setup Time	40				ns
98	TdBAI (BAO)	$\overline{BAI}$ RE, FE to $\overline{BAO}$ RE, FE Delay		80			ns
99	TdIEI(IEO)	IEI RE, FE to IEO RE, FE Delay		80			ns

*2000ns for reading slow-readable registers (worst case)

Note: RE = rising edge
FE = falling edge

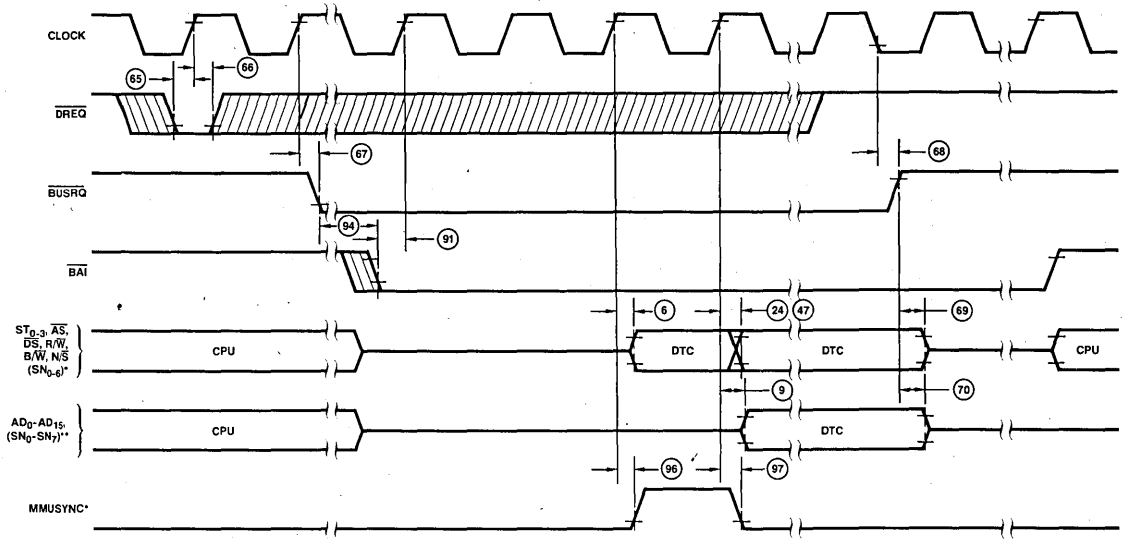
PRELIMINARY

Timing Diagram 14. AC Timing when DTC is Bus Slave



AIZ-029

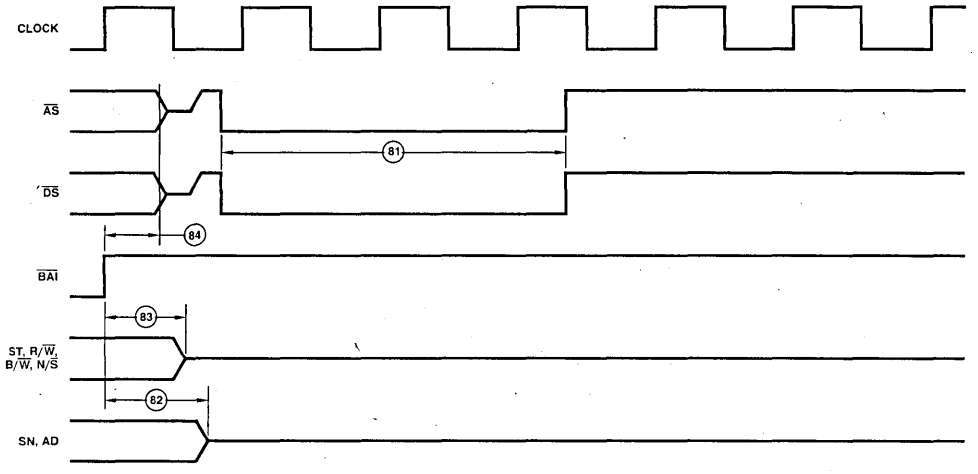
Timing Diagram 15. Bus Exchange Timing



*For logical addressing only.
**For physical addressing only.

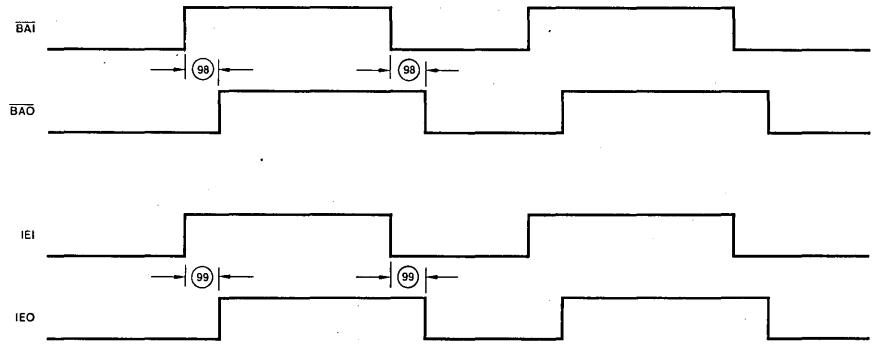
AIZ-030

Timing Diagram 16. Reset Timing



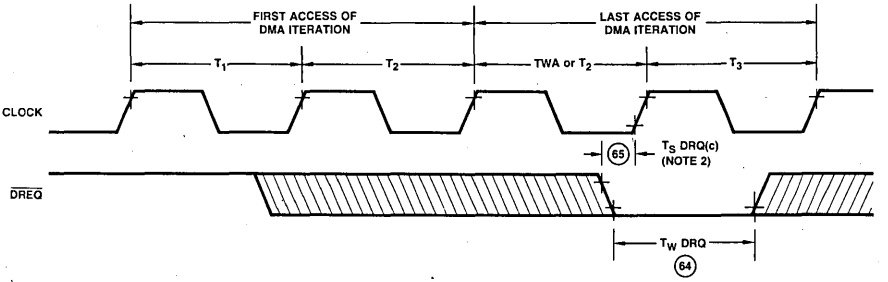
AIZ-031

Timing Diagram 17. Delay Timings



AIZ-032

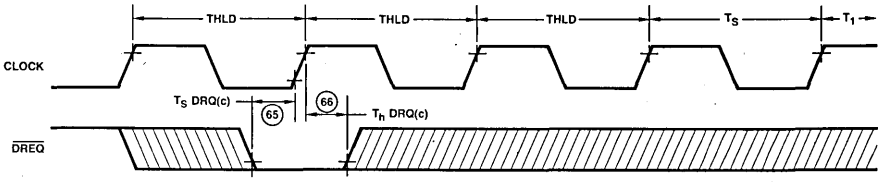
Timing Diagram 18. Sampling DREQ during Single Transfer DMA Operations



AIZ-033

Notes:
1. HIGH-to-LOW $\overline{DREQ}$ transitions will only be recognized after the HIGH-to-LOW transition of the clock during T_1 of the first access of the DMA iteration.
2. A HIGH-to-LOW $\overline{DREQ}$ transition must meet the conditions in Note 1 and must occur T_SDRQ(c) before state T_3 of the last access of the DMA iteration if the channel is to retain bus control and immediately start the next iteration. $\overline{DREQ}$ may go HIGH before T_SDRQ(c) if it has met the T_WDRQ parameter.
3. Flyby and Search transactions have only a single access; parameter T_SDRQ(c) should be referenced to the start of T_3 of the access. All other operations will always have two or three accesses per iteration.

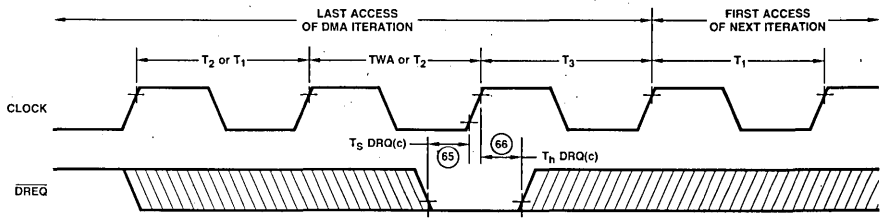
Timing Diagram 19a). Sampling $\overline{DREQ}$ at the End of Chaining



Note: 1. $\overline{DREQ}$ must be LOW from the start of T_SDRQ (c) to the end of T_hDRQ (c) to ensure that the request is recognized.
2. Failure to meet this setup time will result in the channel releasing the bus.

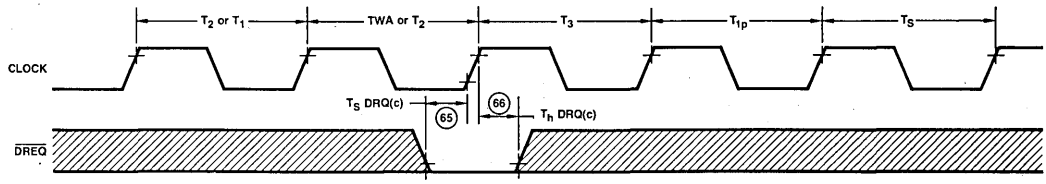
AIZ-034

Timing Diagram 19b). Sampling of $\overline{DREQ}$ While in Bus Hold Mode



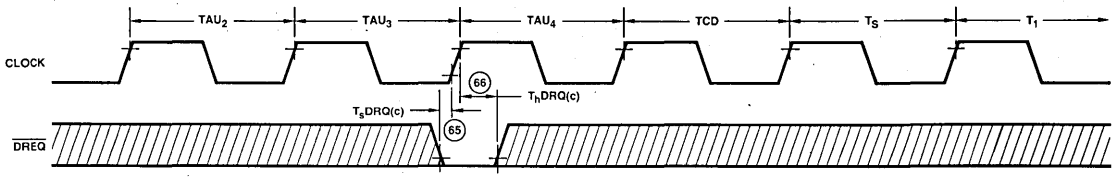
AIZ-035

Timing Diagram 19c). $\overline{DREQ}$ Sampling in Demand Mode during DMA Operations



AIZ-036

Timing Diagram 19d). Sampling $\overline{DREQ}$ at the End of Base-to-Current Reloading



Notes: 1. T_S is a setup state, generated before entering DMA operation cycle.
2. TAU_2 through TAU_4 are auto-reloading states, followed by TCD (chain decision) state.

AIZ-037

Am8127

AmZ8000 Clock Generator

DISTINCTIVE CHARACTERISTICS

- **High-drive high-level clock output**
  Special output provides clock signal matched to requirements of AmZ8000* CPU (4MHz and some 6MHz applications), MMU and DMA devices.
- **Four TTL-level clocks**
  Generates synchronized TTL compatible clocks at 16MHz, 2MHz and 1MHz to drive memory circuits and LSI peripheral devices. An additional TTL clock is synchronized with the CPU high-level clock for registers, latches and other peripherals.
- **Synchronized $\overline{\text{WAIT}}$ state and time-out controls**
  On-chip logic generates $\overline{\text{WAIT}}$ signal under control of Halt, Single-step, Status and Ready signals. Automatic time-out of peripheral wait requests.

ORDERING INFORMATION

Order the part number according to the table below to obtain the desired package, temperature range, and screening level.

Order Number	Package Type (Note 1)	Operating Range (Note 2)	Screening Level (Note 3)
Am8127DC	D-24-SLIM	C	C-1
Am8127DCB	D-24-SLIM	C	B-2 (Note 4)
Am8127DM	D-24-SLIM	M	C-3
Am8127DMB	D-24-SLIM	M	B-3
Am8127LC	L-28-1	C	C-1
Am8127LCB	L-28-1	C	B-2 (Note 4)
Am8127LM	L-28-1	M	C-3
Am8127LBM	L-28-1	M	B-3
Am8127XC	Dice	C	Visual inspection to MIL-STD-883 Method 2010B.
Am8127XM	Dice	M	

Notes: 1. D = Hermetic DIP, L = Chip Pak. Number following letter is number of leads.
2. C = 0 to +70°C, V_{CC} = 4.5 to 5.5V, M = −55 to +125°C, V_{CC} = 4.50 to 5.50V.
3. Levels C-1 and C-3 conform to MIL-STD-883, Class C. Level B-3 conforms to MIL-STD-883, Class B.
4. 160 hour burn-in.

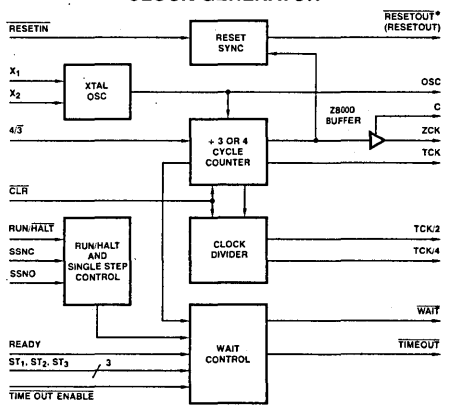
FUNCTIONAL DESCRIPTION

The Am8127 Clock Generator and Controller provides the clock oscillator, frequency dividers and clock drivers for the complete array of AmZ8000 CPUs, peripherals and memory system configurations. In addition to the special 4MHz output driver for the AmZ8001* and AmZ8002* CPUs, a standard buffered TTL 16MHz oscillator output is provided for a dynamic memory timing and control. In addition to 4MHz applications, the Am8127 will also function in some 6MHz Z8000 applications. The Am8127 forms an integral part of the dynamic memory support chip set including the Am8163 EDC and Refresh Controller, Am2964 Dynamic Memory Controller, Am2960 Error Detection and Correction Unit and Am2961/Am2962 EDC Bus Buffers. The oscillator is designed to operate with a 16MHz crystal or with external 16MHz drive. The Am8127 uses an internal divide-by-4 to provide 4MHz clock drive to the AmZ8001/AmZ8002 CPU. Additional dividers generate synchronous buffered 4, 2 and 1MHz clock outputs for use by peripheral devices. The clock divider counters are clearable to allow synchronizing the multiple clock outputs.

The controller functions include $\overline{\text{RESET}}$, RUN/$\overline{\text{HALT}}$, SINGLE-STEP, READY and a READY TIMEOUT counter which limits a peripheral's wait request to 16 clock cycles. The CPU's $\overline{\text{WAIT}}$ input is controlled by RUN/$\overline{\text{HALT}}$, Single-Step, Status and READY. When RUN/$\overline{\text{HALT}}$ is LOW the Am8127 drives the $\overline{\text{WAIT}}$ output LOW causing the CPU to add wait states (TW). The READY input is used by peripherals to request wait states. The active LOW input timeout enable, $\overline{\text{TOEN}}$, is used to force $\overline{\text{TIMEOUT}}$ LOW and WAIT HIGH 16 clock cycles after a peripheral has requested a wait but fails to release the request. The CPU status lines ST_1, ST_2 and ST_3 are decoded in the Am8127 to disable the $\overline{\text{TIMEOUT}}$ counter during CPU "Internal Operations" and during refresh.

The 4/3 input controls the clock duty cycle. An internal pull-up resistor pulls this input high for AmZ8000 CPUs. A LOW input causes the cycle counter to output a 33% duty cycle.

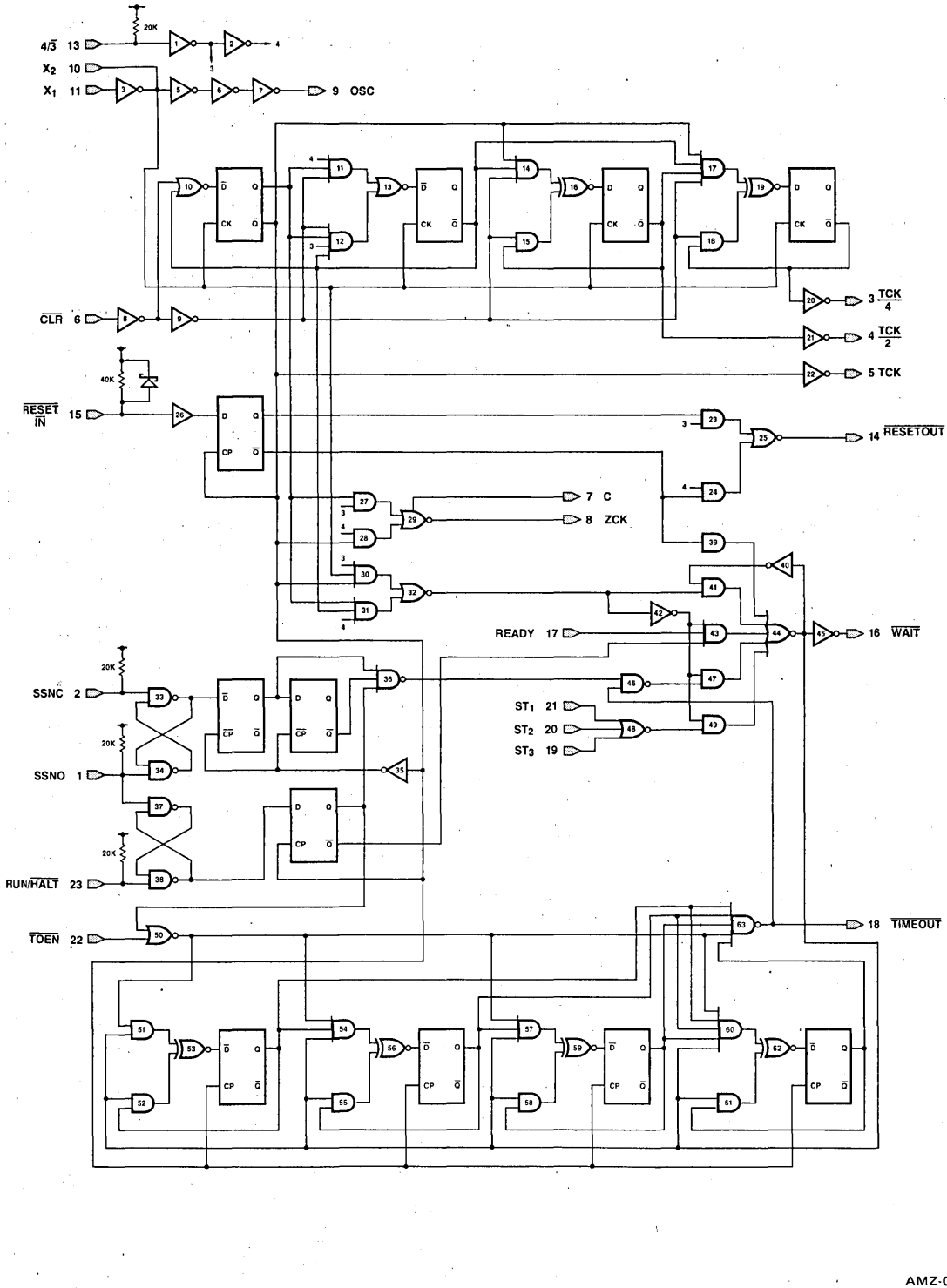
BLOCK DIAGRAM
CLOCK GENERATOR



AMZ-017 *$\overline{\text{RESETOUT}}$ is active LOW when 4/3 = HIGH

*Z8000, Z8001 and Z8002 are trademarks of Zilog, Inc.

LOGIC DIAGRAM



DEFINITION OF FUNCTIONAL TERMS

ZCK Buffered clock output for CPU and peripherals. This output has under/overshoot control and provides the high level output voltage required (V_{CC} – 0.4V). This output is capable of driving multiple CPU clock inputs (or DMA, MMU, etc).

C Bootstrap input. The capacitor C_B is connected from the ZCK clock output to C to provide faster ZCK risetime.

TCK TTL level buffered clock output. TCK is the same frequency as ZCK and is synchronized with ZCK. TCK is in phase with ZCK when the 4/$\overline{3}$ duty cycle control input is HIGH (50% duty cycle) and out of phase with ZCK when 4/$\overline{3}$ is LOW (33% ZCK duty cycle).

TCK/2, TCK/4 TTL buffered clocks for peripherals. TCK/2 and TCK/4 are 1/2 and 1/4 the TCK frequency and are synchronized with the rising edge of TCK.

OSC The clock oscillator TTL buffered output. This output provides a high speed clock for dynamic memory timing (e.g. AmZ8000 uses this output to generate $\overline{RAS}$/MUX-Select/$\overline{CAS}$ timing for dynamic RAMs) or other system application. The ZCK and TCK outputs are synchronized to the OSC rising edge.

4/$\overline{3}$ Clock duty cycle control for ZCK and TCK. A HIGH input (no connection – input has internal pull-up) will result in a 50% duty cycle for AmZ8000 application. A LOW input will cause a 33% duty cycle ZCK output.

$\overline{CLR}$ The clear active LOW input for internal counters. A LOW input meeting set-up and hold time requirements will clear the internal clock counters on the rising edge of OSC.

$\overline{WAIT}$ The $\overline{WAIT}$ output for connection to the CPU $\overline{WAIT}$ input. This latched output controls when the CPU enters wait states in response to the READY, ST_1, ST_2, ST_3, RUN/$\overline{HALT}$ and Single Step inputs.

READY The active HIGH READY input is used by peripherals to request wait states. Ready inputs must meet the wait latch set-up and hold time requirements.

$\overline{TIMEOUT}$ The Timeout Counter active LOW output. The Timeout Counter counts ZCK/TCK clock cycles and is used to force $\overline{WAIT}$ HIGH 15 clock cycles after a peripheral has requested a wait but has failed to release the request. This output is normally used to interrupt the CPU.

$\overline{TOEN}$ The Timeout Enable active LOW input. A LOW input allows the Timeout Counter to count, causes the $\overline{TIMEOUT}$ output to go LOW for one ZCK/TCK clock period after 15 cycles and forces $\overline{WAIT}$ HIGH at the rising edge of the 16th cycle. A HIGH input disables the counter and allows $\overline{WAIT}$ to be controlled by the READY, RUN/$\overline{HALT}$ and Single Step inputs.

$\overline{RESETOUT}$ (RESETOUT) The Reset Output to the CPU. It is active LOW when the 4/$\overline{3}$ input is HIGH and active HIGH when the 4/$\overline{3}$ input is LOW.

$\overline{RESETIN}$ The active LOW Reset Input. A LOW input will cause $\overline{RESETOUT}$ to go LOW synchronous with ZCK ⨍. Pushbutton reset is implemented by momentarily grounding $\overline{RESETIN}$. Power-up reset is implemented by connecting a capacitor from $\overline{RESETIN}$ to ground. Capacitor values from 10μF to 22μF will provide a power-up of less than one second.

RUN/$\overline{HALT}$ A debounced input to allow halt and Single Step control modes. A HIGH input allows the CPU to run. A LOW input forces the $\overline{WAIT}$ output LOW causing the CPU to enter continuous wait states until the ZCK period after RUN/$\overline{HALT}$ is returned to HIGH.

SSNO, SSNC Single Step control inputs. These debounced input allow the CPU to Single Step from one wait state to the next by momentarily disconnecting SSNC from ground and grounding SSNO. RUN/$\overline{HALT}$ must be LOW for Single Step operation.

ST_1, ST_2, ST_3 Status inputs from AmZ8000 CPU's and peripherals. Continuous LOW inputs indicate that the CPU is executing "internal operation" or "refresh." During this time the time out is disabled to avoid signaling an inappropriate interrupt. The status inputs are subject to the set-up and hold time requirements of the $\overline{WAIT}$ latch.

X_1, X_2 External crystal connections (see application section). X_1 may be driven directly by a TTL input.

*$\overline{RESETOUT}$ is active LOW when 4/$\overline{3}$ = HIGH.

ELECTRICAL CHARACTERISTICS

The Following Conditions Apply Unless Otherwise Specified:

COM'L T_A = 0 to 70°C V_{CC} = 5.0V ±5% (MIN. = 4.75V MAX. = 5.25V)
MIL T_A = −55 to +125°C V_{CC} = 5.0V ±10% (MIN. = 4.50V MAX. = 5.50V)

DC CHARACTERISTICS OVER OPERATING RANGE

Parameter	Description	Test Conditions (Note 1)				Min	Typ (Note 2)	Max	Units
V_{OH}	Output HIGH Voltage	V_{CC} = MIN	ZCK	I_{OH} = −0.1mA		V_{CC}−0.4	V_{CC}−0.1		Volts
			TTL Outputs	I_{OH} = −1mA	MIL	2.4	3.4		Volts
				I_{OH} = −2.6mA	COM'L				
V_{OL}	Output LOW Voltage	V_{CC} = MIN		I_{OL} = 0.1mA, ZCK Output				0.4	Volts
				I_{OL} = 16mA, TTL Output				0.5	Volts
V_{IH}	Input HIGH Level	Guaranteed input HIGH Voltage		$\overline{RESETIN}$		2.8	2.25		Volts
				ST_1, ST_2, ST_3, $\overline{CLR}$, $\overline{TOEN}$, X_1, READY		2.0			Volts
V_{IL}	Input LOW Level	Guaranteed input LOW voltage		ST_1, ST_2, ST_3, $\overline{CLR}$, $\overline{TOEN}$, X_1, READY				0.8	Volts
V_I	Input Clamp Voltage	V_{CC} = MIN, I_{IN} = −18mA (Note 3)						−1.5	Volts
$V_{IN}-V_{IL}$	$\overline{RESETIN}$ Hysteresis	V_{CC} = MIN				400	650		mV
I_{IL}	Input LOW Current	V_{CC} = MAX, V_{IN} = 0.4V		SSNO				−1.6	mA
				SSNC, 4/$\overline{3}$, RUN/$\overline{HALT}$, READY				−1.2	mA
				$\overline{TOEN}$, $\overline{CLR}$, X_1				−0.72	mA
				$\overline{RESETIN}$, ST_1, ST_2, ST_3				−0.36	mA
I_{IH}	Input HIGH Current	V_{CC} = MAX, V_{IN} = 2.7V		4/$\overline{3}$, SSNC, SSNO RUN/$\overline{HALT}$			(Note 4)	−300	μA
				$\overline{RESETIN}$			(Note 4)	−200	μA
				$\overline{CLR}$, READY, $\overline{TOEN}$ ST_1, ST_2, ST_3				+50	μA
				X_1				+600	μA
I_I	Input HIGH Current	V_{CC} = MAX, V_{IN} = 5.5V		$\overline{CLR}$, READY, $\overline{TOEN}$ ST_1, ST_2, ST_3				+1.0	mA
I_{SC}	Output Short Circuit Current (Note 5)	V_{CC} = MAX		ZCK Output		−50		−240	mA
				Others		−40		−130	mA
I_{CC}	Power Supply Current	V_{CC} = MAX		X_1 = 2.4V, ZCK = TCK's = LOW			95	140	mA
				Operating, $f_{OSC} \leq$ 24MHz (Note 6)			120	180	

Notes:
1. For conditions shown as MIN or MAX, use the appropriate value specified under Electrical Characteristics for the applicable device type.
2. Typical limits are at V_{CC} = 5.0V, 25°C ambient and maximum loading.
3. Not applicable to X_1.
4. Specification is negative because of internal input pull-up resistors.
5. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.
6. For oscillator frequencies up to 24MHz, outputs open.

STATIC INPUT ELECTRICAL CHARACTERISTICS

The static control inputs, SSNO, SSNC (Single Step), RUN/$\overline{HALT}$ and 4/$\overline{3}$ (clock duty cycle control), are Low-Power Schottky TTL compatible inputs with internal pull-up resistors to the +5V supply. They may be left open for a HIGH input (e.g., 4/$\overline{3}$ is left open for operation with AmZ8001/8002), or grounded for a LOW input.

SSNO, SSNC and RUN/$\overline{HALT}$ are intended to be grounded or opened by switches. 4/$\overline{3}$ is normally left open for AmZ8001/8002. These inputs are specified at 0.4V/2.4V for test convenience.

Parameter	Description	Test Conditions		Min	Typ	Max	Units
V_{IH}	Input HIGH Voltage	Guaranteed HIGH input voltage	RUN/$\overline{HALT}$, SSNO	2.4			Volts
V_{IL}	Input LOW Voltage	Guaranteed LOW input voltage	SSNC, 4/$\overline{3}$			0.4	Volts

MAXIMUM RATINGS (Above which the useful life may be impaired)

Storage Temperature		−65 to +150°C
Temperature (Ambient) Under Bias		−55 to +125°C
Supply Voltage to Ground Potential (Pin 24 to Pin 12) Continuous		−0.5 to +7V
DC Voltage Applied to Outputs for HIGH Output State		−0.5V to +V_{CC} max
DC Input Voltage	X_1, 4/$\overline{3}$, SSNO, SSNC, RUN/$\overline{HALT}$	−0.5V to V_{CC} +0.5V
	Other Inputs	−0.5 to +5.5V
DC Voltage Applied to C		−0.5 to +8V
DC Output Current, Into Outputs		30 mA
DC Input Current		−30 to +5.0mA

SWITCHING CHARACTERISTICS – OSCILLATOR, $\overline{WAIT}$ AND ZCK OUTPUT

(T_A = +25°C, V_{CC} = 5.0V)

Parameters	Description		Min	Typ	Max	Units	Tests Conditions
f_{MAX}	Oscillator Frequency		24			MHz	See Test Circuits (Note 7)
t_{rC}	ZCK Rise Time	C_L = 80pF		9	12	ns	ZCK C_L = 80pF (Note 8)
t_{fC}	ZCK Fall Time			7.6	11	ns	
t_{rC}	ZCK Rise Time	C_L = 200pF		15.4	20	ns	ZCK C_L = 200pF (Note 8)
t_{fC}	ZCK Fall Time			14.0	20	ns	
t_{PLH}	READY to $\overline{WAIT}$			8	14	ns	See Test Circuits
t_{PHL}				11.5	16	ns	
t_{PLH}	Status ST_i to $\overline{WAIT}$			13	17	ns	
t_{PHL}				17.2	21	ns	
t_S	$\overline{CLR}$ to OSC (↓) Set-up Time			15	18	ns	
t_H	$\overline{CLR}$ to OSC (↓) Hold Time			−11	−6	ns	

Notes: 7. Specification is based on fundamental mode crystal. See application section.
8. ZCK rise and fall times are based on a bootstrap capacitor value of 27pF.

SWITCHING CHARACTERISTICS – 4/$\overline{3}$ = HIGH (AmZ8000 Mode)

(T_A = +25°C, V_{CC} = 5.0V)

Parameters	Description		Min	Typ	Max	Units	Test Conditions
t_S	READY to ZCK Set-up Time	(Note 9)	T/4 + 10	T/4 + 4.5		ns	See Test Circuits ZCK C_L = 80pF
t_H	READY to ZCK Hold Time		T/4 + 2	T/4		ns	
t_S	Status ST_i to ZCK Set-up Time		T/4 + 12	T/4 + 9.5		ns	
t_H	Status ST_i to ZCK Hold Time		T/4 − 3	T/4 −7.5		ns	
t_S	$\overline{TOEN}$ to ZCK Set-up Time		30	22		ns	
t_H	$\overline{TOEN}$ to ZCK Hold Time		−10	−16		ns	
t_{SKEW}	ZCK to OSC		3	6	10	ns	
t_{SKEW}	ZCK to TCK		0	4.0	7	ns	
t_{PLH}	ZCK to $\overline{RESETOUT}$ Propagation Delay			9.0	13	ns	
t_{PHL}				4	8	ns	

Note: 9. T = ZCK period.

Am8127
SWITCHING CHARACTERISTICS OVER OPERATING RANGE – OSCILLATOR, $\overline{WAIT}$ AND ZCK OUTPUTS*

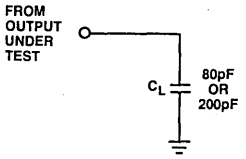
Parameters	Description		Test Conditions	Am8127 COM'L T_A = 0 to +70°C, V_{CC} = 5.0V ±5%			Am8127 MIL T_A = −55 to +125°C, V_{CC} = 5.0V ±10%			Units
				Min	Max 0°C	Max 70°C	Min	Max −55°C	Max 125°C	
t_{MAX}	Oscillator Frequency		(Note 7)	24			24			MHz
t_{rC}	ZCK Rise Time	C_L = 80pF	C_L = 80pF (Note 8)		15	15		20	15	ns
t_{fC}	ZCK Fall Time				14	14		20	14	ns
t_{rC}	ZCK Rise Time	C_L = 200pF	C_L = 200pF (Note 8)		25	20		32	20	ns
t_{fC}	ZCK Fall Time				25	20		32	20	ns
t_{PLH}	READY to WAIT Propagation Delay		See Test Circuits		17	17		19	19	ns
t_{PHL}					19	19		19	19	ns
t_{PLH}	Status ST_i to $\overline{WAIT}$ Propagation Delay				20	20		22	22	ns
t_{PHL}					25	25		25	25	ns
t_S	$\overline{CLR}$ to OSC ($\int$) Setup Time			21			30			ns
t_H	$\overline{CLR}$ to OSC ($\int$) Hold Time			−3			0			ns

*AC performance over the operating temperature range is guaranteed by testing defined in Group A, Subgroup 9.

SWITCHING CHARACTERISTICS OVER OPERATING RANGE – 4/$\overline{3}$ = HIGH (AmZ8000 Mode)

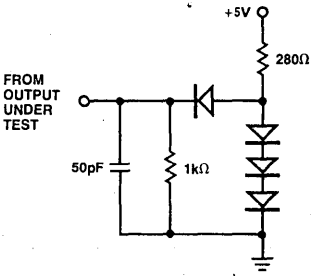
Parameters	Description	Test Conditions	Am8127 COM'L T_A = 0 to +70°C, V_{CC} = 5.0V ±5%		Am8127 MIL T_A = −55 to +125°C, V_{CC} = 5.0V ±10%		Units
			Min	Max	Min	Max	
t_S	READY to ZCK Setup Time	See Test Circuits ZCK C_L = 80pF	T/4 + 14		T/4 + 17		ns
t_H	READY to ZCK Hold Time		T/4 + 5		T/4 + 5		ns
t_S	Status ST_i to ZCK Setup Time		T/4 + 15		T/4 + 20		ns
t_H	Status ST_i to ZCK Hold Time		T/4		T/4 + 5		ns
t_S	$\overline{TOEN}$ to ZCK Setup Time		35		40		ns
t_H	$\overline{TOEN}$ to ZCK Hold Time		−5		0		ns
t_{SKEW}	ZCK to OSC Skew		2	14	2	17	ns
t_{SKEW}	ZCK to TCK Skew		−2	10	−2	14	ns
t_{PLH}	ZCK to $\overline{RESETOUT}$ Propagation Delay			16		20	ns
t_{PHL}				16		20	ns

SWITCHING TEST CIRCUITS



AMZ-018

ZCK Output

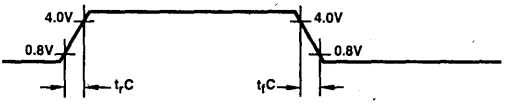


AMZ-019

TTL Outputs

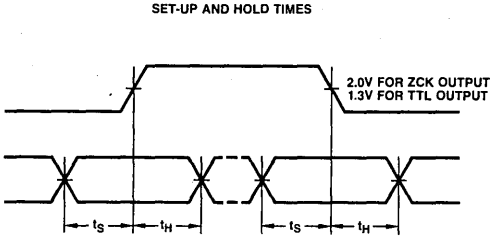
SWITCHING TEST WAVEFORMS

ZCK RISE AND FALL TIMES



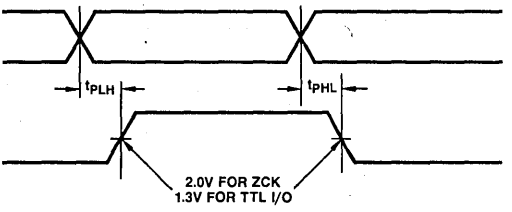
AMZ-020

SET-UP AND HOLD TIMES



AMZ-021

PROPAGATION DELAY TIMES



AMZ-022

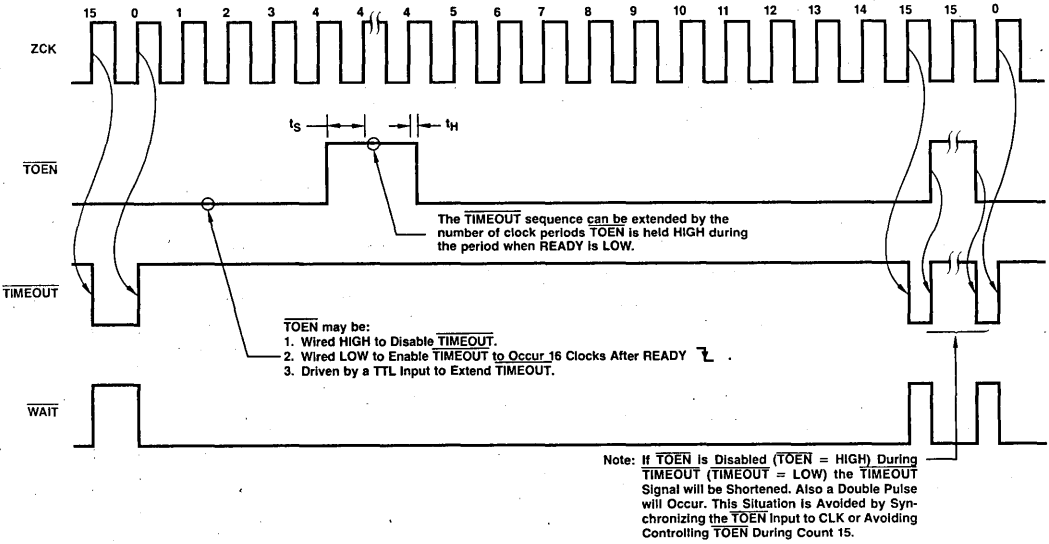
TYPICAL CRYSTAL SPEC

Mode	Fundamental AT cut
Resonance	Parallel or Series
Load	32pF (Net of 56pF C's shown + stray C)
Stability	±0.01% (or to user requirement)

$\overline{\text{WAIT}}$, $\overline{\text{TIMEOUT}}$ FUNCTION TABLE

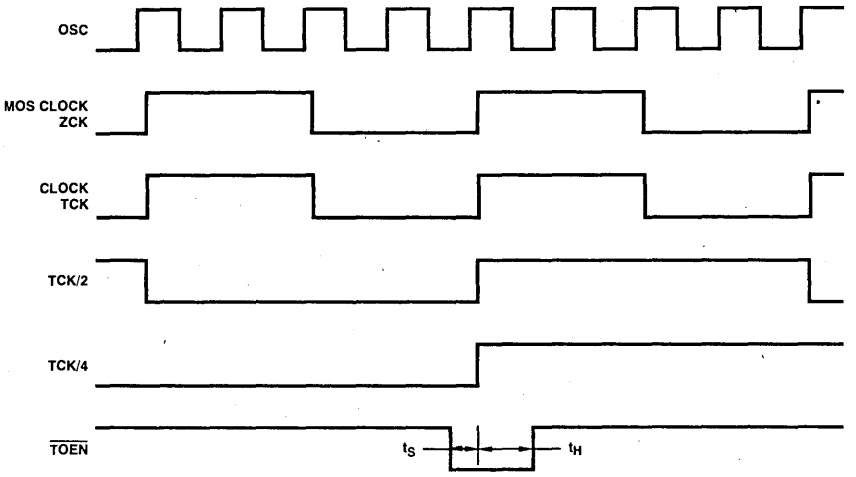
RUN/$\overline{\text{HALT}}$	SSNC	ST_3 ST_2 ST_1	READY	$\overline{\text{TOEN}}$	$\overline{\text{TIMEOUT}}$ COUNTER	$\overline{\text{TIMEOUT}}$	$\overline{\text{WAIT}}$
H	X	L L L	H	X	Cleared	H	H
		L L L	L	X	Cleared	H	H
		Any ST_i = H	H	L	Cleared	H	H
			L	H	Hold	H	L
			L	H	Count + 1 on ZCK ↓	H until 16 clocks after ready ↓, then LOW one ZCK period	L until 16 clocks after ready ↓, then LOW one ZCK period
L	L	X	X	X	Hold	H	L
	H						HIGH one ZCK period

$\overline{\text{TIMEOUT}}$ COUNTER TIMING



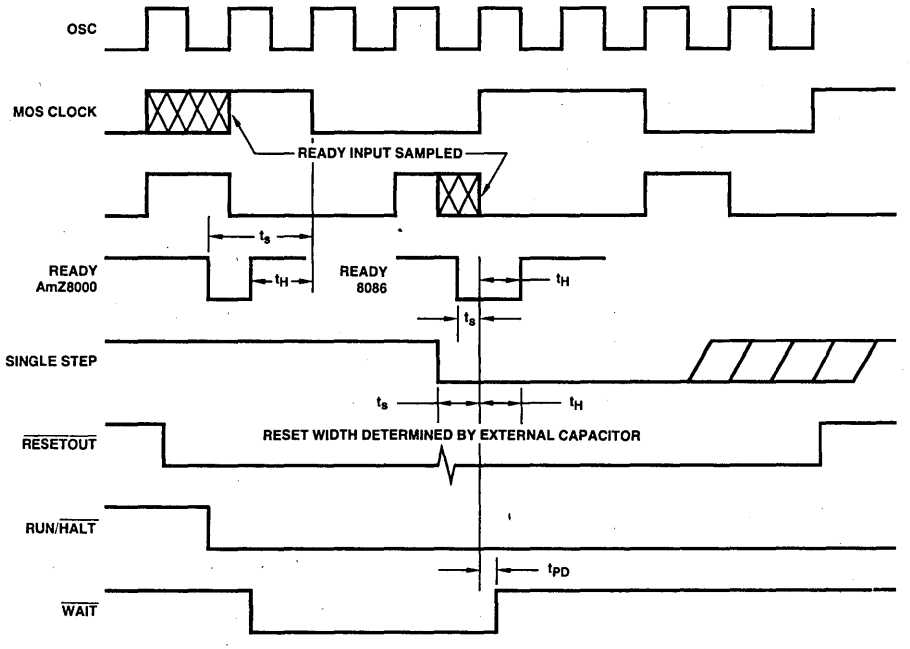
AMZ-027

Am8127 CLOCK OUTPUTS
DIVIDE BY 4 MODE (AmZ8000)



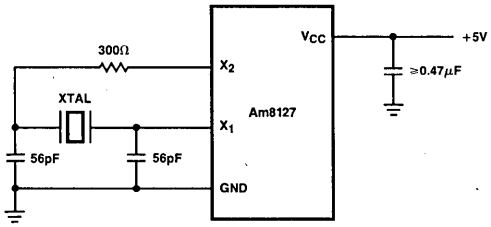
AMZ-024

Am8127 READY, WAIT, RESET, AND SINGLE STEP



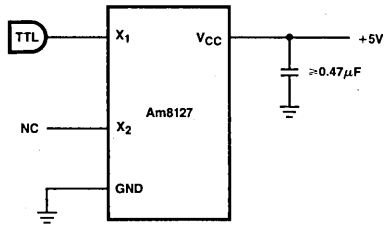
AMZ-026

CRYSTAL CONTROLLED OSCILLATOR



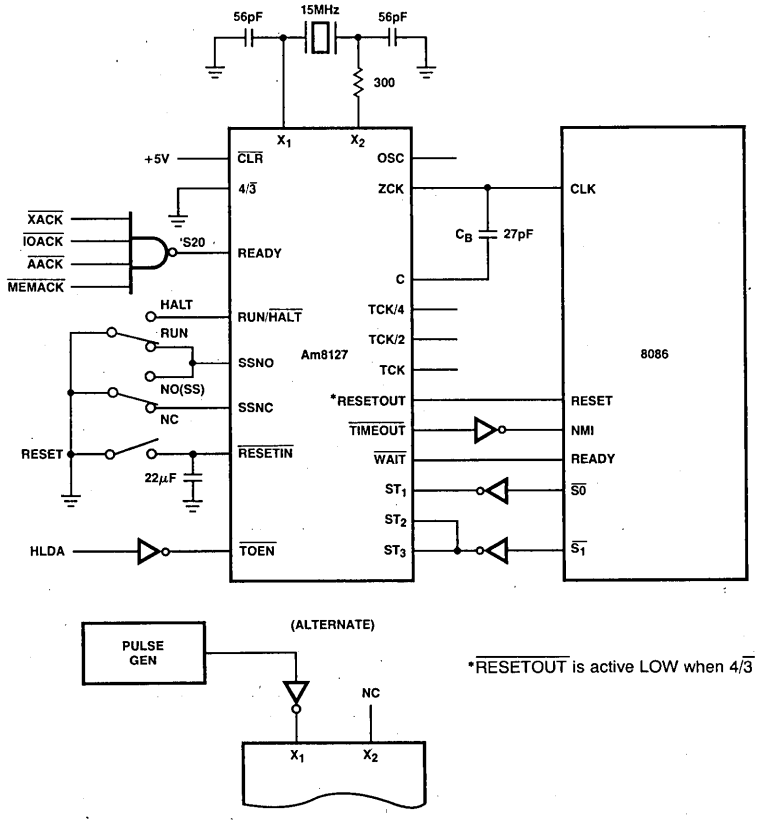
AMZ-028

EXTERNAL CLOCK DRIVE



AMZ-029

AmZ8000 APPLICATION
(50% Duty Cycle ZCK)

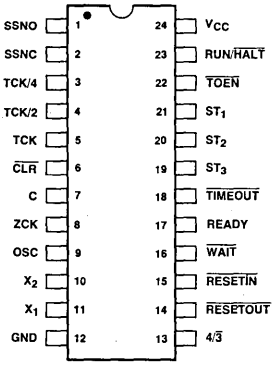


*$\overline{\text{RESETOUT}}$ is active LOW when $4/\overline{3}$ = HIGH

AMZ-030

The typical operating configuration for Am8127 is shown above. The component values shown provide a 4MHz clock output for the AmZ8002 CPU. The 27pF capacitor from C to ZCK is a bootstrap to ensure clock rise to $V_{CC}-0.4V$ within the specified rise time. The 22μF reset capacitor is chosen to guarantee reset, plus adequate delay for reset during power-up with a slowly rising V_{CC} supply voltage. Ground SSNO if RUN/$\overline{\text{HALT}}$ or S-S isn't used.

CONNECTION DIAGRAM
Top View

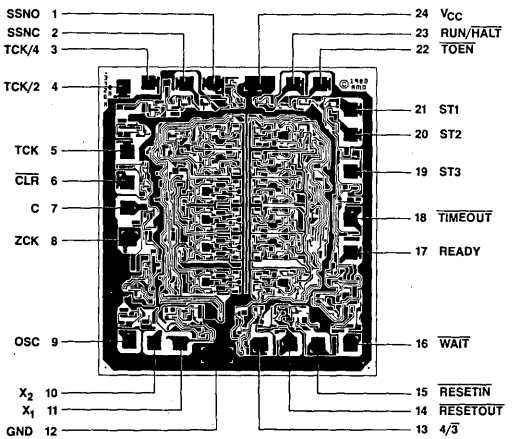


24 Pin 0.3" wide

Note: Pin 1 is marked for orientation.

AMZ-032

METALLIZATION AND PAD LAYOUT



DIE SIZE 0.098" X 0.088"

Single-Chip Microcomputers

8051/8031

Single-Chip 8-Bit Microcomputer

8031 – Control oriented CPU with RAM and IO

8051 – An 8031 with factory mask-programmable ROM

DISTINCTIVE CHARACTERISTICS

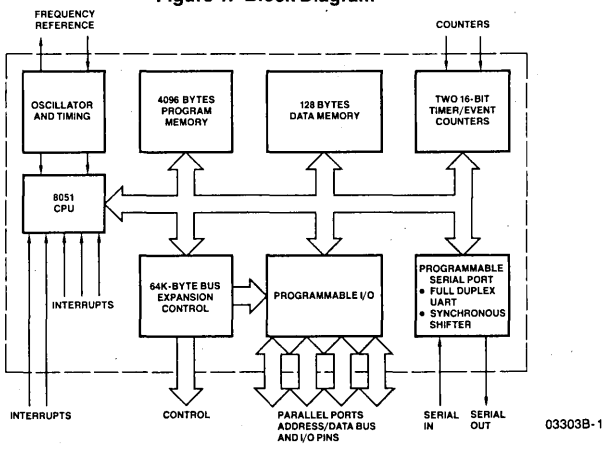
- 4K x 8 ROM
- 128 x 8 RAM
- Four 8-bit ports, 32 I/O lines
- Two 16-bit timer/event counters
- High-performance full-duplex serial channel
- External memory expandable to 128K
- Compatible with 8080 and 8085 peripherals
- Boolean processor
- 8048 architecture enchanced with:
  - Non-paged jumps
  - Direct addressing
  - Four 8-register banks
  - Stack depth up to 128-bytes
  - Multiply, divide, subtract, compare
- Most instructions execute in 1μs
- 4μs multiply and divide

GENERAL DESCRIPTION

The 8051/8031 are members of a family of advanced single-chip microcomputers. The 8051 contains 4K x 8 read-only program memory; 128 x 8 RAM; 32 I/O lines; two 16-bit timer/counters; a five-source, two-priority-level, nested interrupt structure; a serial I/O port for either multi-processor communications, I/O expansion, or full duplex UART; and on-chip oscillator and clock circuits. The 8031 is identical, except that it lacks the program memory. For systems that require extra capability, the 8051 can be expanded using standard TTL compatible memories and the byte oriented 8080 and 8085 peripherals.

The 8051 microcomputer, like its 8048 predecessor, is efficient both as a controller and as a boolean processor. The 8051 has extensive facilities for binary and BCD arithmetic and excels in bit-handling capabilities. Efficient use of program memory results from an instruction set consisting of 44% one-byte, 41% two-byte, and 15% three-byte instructions. With a 12MHz crystal, 58% of the instructions execute in 1μs, 40% in 2μs and multiply and divide require only 4μs. Among the many instructions added to the standard 8048 instruction set are multiply, divide, subtract, and compare.

Figure 1. Block Diagram



ORDERING INFORMATION

Operating Range	Package	12MHz w/ROM	12MHz w/o ROM
Commercial $0°C \leq T_A \leq 70°C$ $V_{CC} = +5V \pm 5\%$ $V_{SS} = 0V$	Hermetic DIP	D8051-XXXXX	D8031
	Molded DIP	P8051-XXXXX	P8031
Industrial $-40°C \leq T_A \leq 85°C$ $V_{CC} = +5V \pm 10\%$ $V_{SS} = 0V$	Hermetic DIP	ID8051-XXXXX	ID8031
Military $-55°C \leq T_A \leq 125°C$ $V_{CC} = +5V \pm 10\%$ $V_{SS} = 0V$	Hermetic DIP	MD8051B-XXXXX	MD8031B

*XXXXX is a five digit ROM code identifier assigned by factory.

LOGIC SYMBOL

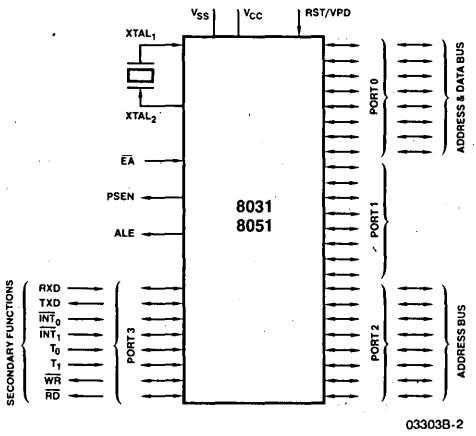


Figure 2.

CONNECTION DIAGRAM
Top View
D-40
P-40

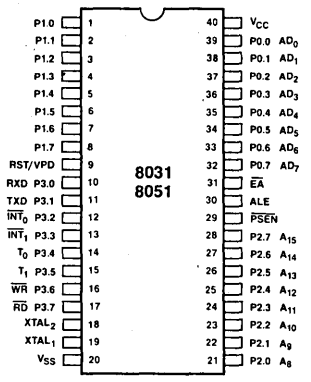


Figure 3.

8051 FAMILY PIN DESCRIPTION

V_{SS}

Circuit ground potential.

V_{CC}

+5V power supply during operation.

PORT 0

Port 0 is an 8-bit open drain bidirectional I/O port. It is also the multiplexed low-order address and data bus when using external memory. It is used for data output during program verification. Port 0 can sink/source eight LS TTL loads.

PORT 1

Port 1 is an 8-bit quasi-bidirectional I/O port. It is used for the low-order address byte during program verification. Port 1 can sink/source four LS TTL loads.

PORT 2

Port 2 is an 8-bit quasi-bidirectional I/O port. It also emits the high-order address byte when accessing external memory. It is used for the high-order address and the control signals during program verification. Port 2 can sink/source four LS TTL loads.

PORT 3

Port 3 is an 8-bit quasi-bidirectional I/O port. It also contains the interrupt, timer, serial port, and $\overline{RD}$ and $\overline{WR}$ pins that are used by various options. The output latch corresponding to a secondary function must be programmed to a one (1) for that function to operate. Port 3 can sink/source four LS TTL loads. The secondary functions are assigned to the pins of Port 3, as follows:

- RXD/data (P3.0). Serial port's receiver data input (asynchronous) or data input/output (synchronous).
- TXD/clock (P3.1). Serial port's transmitter data output (asynchronous) or clock output (synchronous).
- $\overline{INT_0}$ (P3.2). Interrupt 0 input or gate control input for counter 0.
- $\overline{INT_1}$ (P3.3). Interrupt 1 input or gate control input for counter 1.
- T_0 (P3.4). Input to counter 0.
- T_1 (P3.5). Input to counter 1.
- $\overline{WR}$ (P3.6). The write control signal latches the data byte from Port 0 into the External Data Memory.
- $\overline{RD}$ (P3.7). The read control signal enables External Data Memory to Port 0.

RST/V_{PD}

A high level on this pin resets the 8051. If V_{PD} is held within its spec (approximately +5V), while V_{CC} drops below spec, V_{PD} will provide standby power to the RAM. When V_{PD} is low, the RAM's current is drawn from V_{CC}. A small internal resistor permits power-on reset using only a capacitor connected to V_{CC}.

ALE

Provides Address Latch Enable output used for latching the address into external memory during normal operation. It is activated every six oscillator periods, except during an external data memory access.

$\overline{PSEN}$

The Program Store Enable output is a control signal that enables the external Program Memory to the bus during external fetch operations. It is activated every six oscillator periods, except during external data memory accesses. Remains high during internal program execution.

$\overline{EA}$

When held at a TTL high level, the 8051 executes instructions from the internal ROM when the PC is less than 4096. When held at a TTL low level, the 8051 fetches all instructions from external Program Memory.

$XTAL_1$

Input to the oscillator's high gain amplifier. Required when a crystal is used. Connect to V_{SS} when external source is used on $XTAL_2$.

$XTAL_2$

Output from the oscillator's amplifier. Input to the internal timing circuitry. A crystal or external source can be used.

THE 8051 FAMILY

The 8051 is a stand-alone high-performance single-chip computer intended for use in sophisticated real-time applications such as instrumentation, industrial control, and intelligent computer peripherals. It provides the hardware features, architectural enhancements, and new instructions that make it a powerful and cost effective controller for applications requiring up to 64K bytes of program memory and/or up to 64K bytes of data storage. A Block Diagram is shown in Figure 1.

The 8031 is a control-oriented CPU without on-chip program memory. It can address 64K-bytes of External Program Memory in addition to 64K-bytes of External Data Memory. For systems requiring extra capability, each member of the 8051 Family can be expanded using standard memories and the byte oriented 8080 and 8085 peripherals. The 8051 is an 8031 with the lower 4K-bytes of Program Memory filled with on-chip mask programmable ROM.

The two pin-compatible versions of this component reduce development problems to a minimum and provide maximum flexibility. The 8051 is suited for low-cost, high volume production; and the 8031 for applications desiring the flexibility of External Program Memory which can be easily modified and updated in the field.

MACRO-VIEW OF THE 8051

On a single die the 8051 microcomputer combines CPU; 4K x 8 read-only program memory; 128 x 8 RAM; 32 I/O lines; two 16-bit timer/event counters; a five-source, two-priority-level, nested interrupt structure; serial I/O port for either multi-processor communications, I/O expansion or full duplex UART; and on-chip oscillator and clock circuits. This section will provide an overview of the 8051 by providing a high-level description of its major elements: the CPU architecture and the on-chip functions peripheral to the CPU. The generic term "8051" is used to refer collectively to the 8031 and 8051.

8051 CPU ARCHITECTURE

The 8051 CPU manipulates operands in four memory spaces. These are the 64K-byte Program Memory, 64K-byte External Data Memory, 256-byte Internal Data Memory and 16-bit Program Counter spaces. The Internal Data Memory address space is further divided into the 128-byte Internal Data RAM and 128-byte Special Function Register (SFR) address spaces shown in Figure 4. Four Register Banks (each with eight registers), 128 addressable bits and the stack reside in the Internal Data RAM. The stack depth is limited only by the available Internal Data RAM and its location is determined by the 8-bit stack pointer. All registers except the Program Counter and the four 8-Register Banks reside in the Special Function Register address space. These memory mapped registers include arithmetic registers, pointers, I/O ports, interrupt system registers, timers, and a serial port. 128 bit locations in the SFR address space are addressable as bits. The 8051 contains 128 bytes of Internal Data RAM and 20 SFRs.

The 8051 provides a non-paged Program Memory address space to accommodate relocatable code. Conditional branches are performed relative to the Program Counter. The register-indirect jump permits branching relative to a 16-bit base register with an offset provided by an 8-bit index register. Sixteen-bit jumps and calls permit branching to any location in the contiguous 64K Program Memory address space.

The 8051 has five methods for addressing source operands: Register, Direct, Register-Indirect, Immediate, and Base-Register-plus-Index-Register-Indirect Addressing. The first three methods can be used for addressing destination operands. Most instructions have a "destination, source" field that specifies the data type, addressing methods, and operands involved. For operations other than moves, the destination operand is also a source operand.

Registers in the four 8-Register Banks can be accessed through Register, Direct, or Register-Indirect Addressing; the 128 bytes of Internal Data RAM through Direct or Register-Indirect Addressing; and the Special Function Registers through Direct Addressing. External Data Memory is accessed through Register-Indirect Addressing. Look-Up-Tables resident in Program Memory can be accessed though Base-Register-plus-Index-Register-Indirect Addressing.

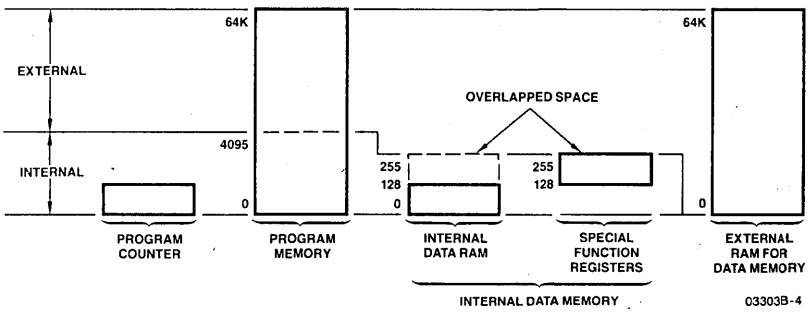
The 8051 is classified as an 8-bit machine since the internal ROM, RAM, Special Function Registers, Arithmetic/Logic Unit, and external data bus are each 8-bits wide. The 8051 performs operations on bit, nibble, byte, and double-byte data types.

The 8051 has extensive facilities for byte transfer, logic, and integer arithmetic operations. It excels at bit handling since data transfer, logic, and conditional branch operations can be performed directly on Boolean variables.

8051 INSTRUCTION SET

The 8051's instruction set is an enhancement of the instruction set familiar to 8048 users. It is enhanced to allow expansion of on-chip CPU peripherals and to optimize byte efficiency and execution speed. Op codes were reassigned to add new high-power operations and to permit new addressing modes which make the old operations more orthogonal. Efficient use of program memory results from an instruction set consisting of 49 single-byte, 45 two-byte, and 17 three-byte instructions. When using a 12MHz oscillator, 64 instructions execute in 1μs and 45

Figure 4. 8051 Family Memory Organization



instructions execute in 2μs. The remaining instructions (multiply and divide) execute in only 4μs. The number of bytes in each instruction and the number of cycles required for execution are listed in Table 1 on pages 14 and 15.

ON-CHIP PERIPHERAL FUNCTIONS

Thus far only the CPU and memory spaces of the 8051 have been described. In addition to the CPU and memories, an interrupt system, extensive I/O facilities, and several peripheral functions are integrated on-chip to relieve the CPU of repetitious, complicated, or time-critical tasks and to permit stringent real-time control of external system interfaces. The extensive I/O facilities include the I/O pins, parallel I/O ports, bidirectional address/data bus, and the serial port for I/O expansion. The CPU peripheral functions integrated on-chip are the two 16-bit counters and the serial port. All of these work together to greatly boost system performance.

INTERRUPT SYSTEM

External events and the real-time-driven on-chip peripherals require service by the CPU asynchronous to the execution of any particular section of code. To tie the asynchronous activities of these functions to normal program execution, a sophisticated multiple-source, two-priority-level, nested interrupt system is provided. Interrupt response latency ranges from 3μs to 7μs when using a 12MHz crystal.

The 8051 acknowledges interrupt requests from five sources: Two from external sources via the $\overline{INT_0}$ and $\overline{INT_1}$ pins, one from each of the two internal counters and one from the serial I/O port. Each interrupt vectors to a separate location in Program Memory for its service program. Each of the five sources can be assigned to either of two priority levels and can be independently enabled and disabled. Additionally all enabled sources can be globally disabled or enabled. Each external interrupt is programmable as either level- or transition-activated and is active-low to allow the "wire or-ing" of several interrupt sources to the input pin. The interrupt system is shown diagrammatically in Figure 5.

I/O FACILITIES

The 8051 has instructions that treat its 32 I/O lines as 32 individually addressable bits and as four parallel 8-bit ports addressable as Ports 0, 1, 2, and 3. Ports 0, 2, and 3 can also assume other functions. Port 0 provides the multiplexed low-order address and data bus used for expanding the 8051 with standard memories and peripherals. Port 2 provides the high-order address bus when expanding the 8051 with external Program Memory or more than 256 bytes of External Data Memory. The pins of Port 3 can be configured individually to provide external interrupt request inputs, counter inputs, the serial port's receiver input and transmitter output, and to generate the control signals used for reading and writing External Data Memory. The generation or use of an alternate function on a Port 3 pin is done automatically by the 8051 as long as the pin is configured as an input. The configuration of the ports is shown on the 8051 Family Logic Symbol of Figure 2.

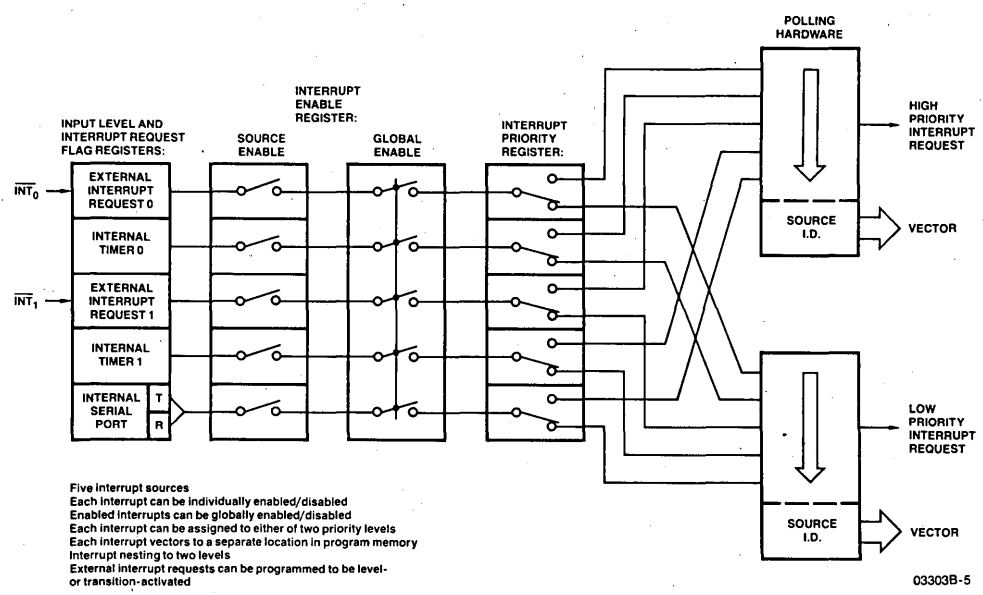
Open Drain I/O Pins

Each pin of Port 0 can be configured as an open drain output or as a high-impedance input. Resetting the microcomputer programs each pin as an input by writing a one (1) to the pin. If a zero (0) is later written to the pin it becomes configured as an output and will continuously sink current. Rewriting the pin to a one (1) will place its output driver in a high-impedance state and configure the pin as an input. Each I/O pin of Port 0 can sink/source eight LS TTL loads.

Quasi-Bidirectional I/O Pins

Ports 1, 2, and 3 are quasi-bidirectional buffers. Resetting the microcomputer programs each pin as an input by writing a one (1) to the pin. If a zero (0) is later written to the pin it becomes configured as an output and will continuously sink current. Any pin that is configured as an output will be reconfigured as an input when a one (1) is written to the pin. Simultaneous to this reconfiguration, the output driver of the quasi-bidirectional port will source current for two oscillator periods. Since current is sourced only when a bit previously written to a zero (0) is up-

Figure 5. 8051 Interrupt System



dated to a one (1), a pin programmed as an input will not source current into the TTL gate that is driving it if the pin is later written with another one (1). Since the quasi-bidirectional output driver sources current for only two oscillator periods, an internal pull-up resistor of approximately 20K- to 40K-ohms is provided to hold the external driver's loading at a TTL high level. Ports 1, 2, and 3 can sink/source four LS TTL loads.

Microprocessor Bus

A microprocessor bus is provided to permit the 8051 to solve a wide range of problems and to allow the upward growth of user products. This multiplexed address and data bus provides an interface compatible with standard memories, 8080 peripherals, and the 8085 compatible memories that include on-chip programmable I/O ports and timing functions. These are summarized in the 8051 Microcomputer Expansion Components chart of Figure 6.

When accessing external memory the high-order address is emitted on Port 2 and the low-order address on Port 0. The ALE signal is provided for strobing the address into an external latch. The program store enable ($\overline{PSEN}$) signal is provided for enabling an external memory device to Port 0 during a read from the Program Memory address space. When the MOVX instruction is executed Port 3 automatically generates the read ($\overline{RD}$) signal for enabling an External Data Memory device to Port 0 or generates the write ($\overline{WR}$) signal for strobing the external memory device with the data emitted by Port 0. Port 0 emits the address and data to the external memory through a push/pull driver that can sink/source eight LS TTL loads. At the end of the read/write bus cycle, Port 0 is automatically reprogrammed to its high impedance state and Port 2 is returned to the state it had prior to the bus cycle. The 8051 generates the address, data, and control signals needed by memory and I/O devices in a manner that minimizes the requirements placed on external program and data memories. At 12MHz, the Program Memory cycle time is 500ns and the access times required from stable address and $\overline{PSEN}$ are approximately 320ns and 150ns respectively. The External Data Memory cycle time is 1μs and the access times required for stable address from read ($\overline{RD}$) or write ($\overline{WR}$) command are approximately 600ns and 250ns respectively.

TIMER/EVENT COUNTERS

The 8051 contains two 16-bit counters for measuring timing events and pulse widths, for counting events, as well as for generating precise, periodic interrupt requests. Each can be programmed independently to one of the following three modes:

Mode 0 – similar to an 8048 8-bit timer or counter with divide by 32 prescaler.

Mode 1 – 16-bit time-interval or event counter.

Mode 2 – 8-bit time-interval or event counter with automatic reload upon overflow.

Additionally, counter 0 can be programmed to a mode that divides it into one 8-bit time-interval or event counter and one 8-bit time-interval counter (Mode 3). When counter 0 is in Mode 3, counter 1 can be programmed to any of the three aforementioned modes, although it cannot set an interrupt request flag or generate an interrupt. This mode is useful because counter 1's overflow can be used to pulse the serial port's transmission-rate generator. Along with their multiple operating modes and 16-bit precision, the counters can also handle very high input frequencies. These range from 0.1MHz to 1.0MHz (from 1.2MHz to 12MHz crystal) when programmed for an input that is a division by 12 of the oscillator frequency and from 0Hz to an upper limit of 50KHz to 0.5MHz (from 1.2MHz to 12MHz crystal) when programmed for external inputs. Both internal and external inputs can be gated to the counter by a second external source for directly measuring pulse widths.

Figure 6. 8051 Microcomputer Expansion Components

Category	AMD Part No.	Description	Comments	Program or Data Memory	Crystal Frequency MHz (Max)
Standard EPROMs	2708	1K x 8 450ns Light Erasable	User programmable and erasable.	P	7
	2716-1	2K x 8 350ns Light Erasable		P	8
	2732	4K x 8 450ns Light Erasable		P	8
	2732A-2	4K x 8 200ns Light Erasable		P	12
Standard RAMs	2114A	1K x 4 100ns RAM	Data memory can be easily expanded using standard NMOS RAMs.	D	12
	2148	1K x 4 70ns RAM		D	12
Standard I/O	8212	8-Bit I/O Port	Serves as Address Latch or I/O port.	D	12
	8255A	Programmable Peripheral Interface	Three 8-bit programmable I/O ports.	D	12
	8251A	Programmable Communications Interface	Serial Communications Receiver/Transmitter.	D	12
Standard Peripherals	8286	Bi-directional Bus Driver	8080 and 8085 peripheral devices are compatible with the 8051 allowing easy addition of specialized interfaces.	D	12
	8287	Bi-directional Bus Driver (Inverting)		D	12
	8253A	Programmable Interval Timer		D	12
	8279	Programmable Keyboard/Display Interface (128 Keys)		D	12
Universal Peripheral Interfaces	8041A	ROM Program Memory	Mask programmable to perform custom I/O and control functions	D/P	12/11.7
Memories with on-chip I/O and Peripheral Functions.	8155-2	256 x 8 330ns RAM		D	12

The counters are started and stopped under software control. Each counter sets its interrupt request flag when it overflows from all ones to all zeroes (or auto-reload value). The operating modes and input sources are summarized in Figures 7 and 8. The effects of the configuration flags and the status flags are shown in Figures 9 and 10.

SERIAL COMMUNICATIONS

The 8051's serial I/O port is useful for serially linking peripheral devices as well as multiple 8051s through standard asynchronous protocols with full-duplex operation. The serial port also has a synchronous mode for expansion of I/O lines using CMOS and TTL shift registers. This hardware serial communications interface saves ROM code and permits a much higher transmission rate than could be achieved through software. In response to a serial port interrupt request, the CPU has only to read/write the serial port's buffer to service the serial link. A block diagram of the serial port is shown in Figures 11 and 12. Methods for linking UART (universal asynchronous receiver/transmitter) devices are shown in Figure 13 and a method for I/O expansion is shown in Figure 14.

The full-duplex serial I/O port provides asynchronous modes to facilitate communications with standard UART devices, such as printers and CRT terminals, or communications with other 8051s in multi-processor systems. The receiver is double buffered to eliminate the overrun that would occur if the CPU failed to respond to the receiver's interrupt before the beginning of the next frame. The 8051 can generally maintain the serial link at its maximum rate so double buffering of the transmitter is not needed. A minor degradation in transmission rate can occur in rare events such as when the servicing of the transmitter has to wait for a lengthy interrupt service program to complete. In asynchronous modes, false start-bit rejection is provided on received frames. For noise rejection a best two-out-of-three vote is taken on three samples near the center of each received bit.

When interfacing with standard UART devices, the serial channel can be programmed to Mode 1 which transmits/receives a ten-bit frame or programmed to Mode 2 or 3 which transmits/receives an eleven-bit frame as shown in Figure 15. The frame consists of a start bit, eight or nine data bits, and one stop bit. In Modes 1 and 3, the transmission-rate timing circuitry receives a pulse from counter 1 each time the counter overflows. The input to counter 1 can be an external source or a division by 12 of the oscillator frequency. The auto-reload mode of the counter provides communication rates of 122 to 31,250 bits per second (including start and stop bits) for a 12MHz crystal. In Mode 2 the communication rate is a division by 64 of the oscillator frequency yielding a transmission rate of 187,500 bits per second (including start and stop bits) for a 12MHz crystal.

Figure 7. Timer/Event Counter Modes 0, 1 and 2

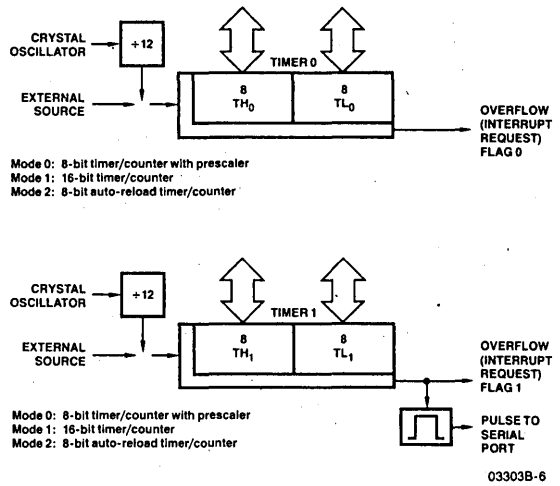


Figure 8. Timer/Event Counter 0 In Mode 3

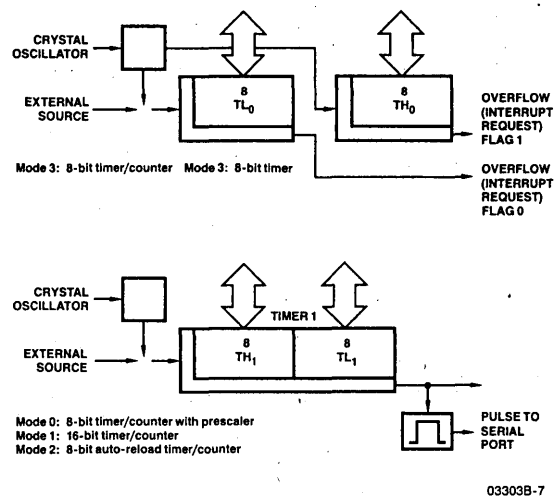


Figure 9. Timer/Counter 0 Control and Status Flag Circuitry

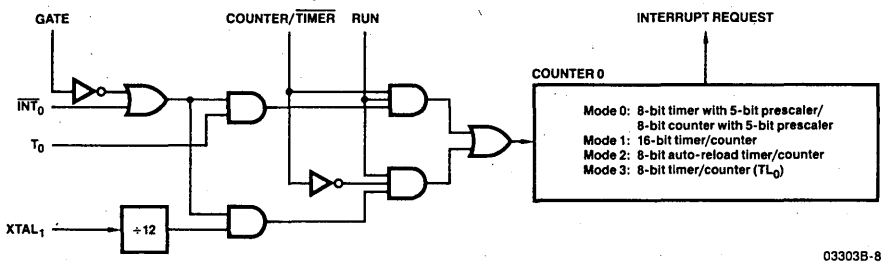


Figure 10. Timer/Counter 1 Control and Status Flag Circuitry

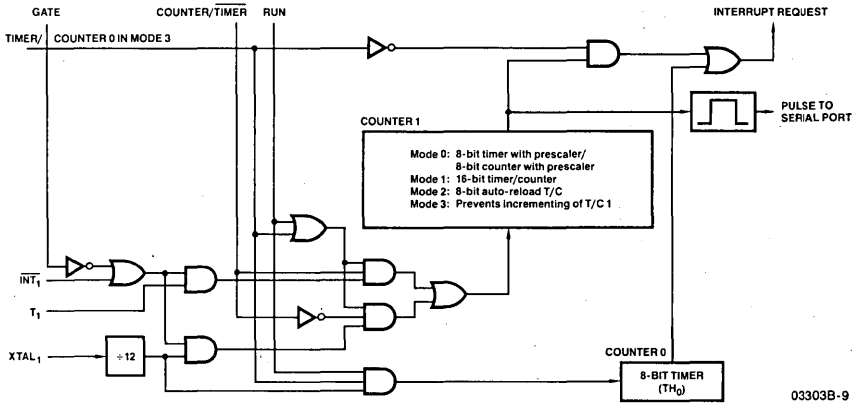


Figure 11. Serial Port – Synchronous Mode 0

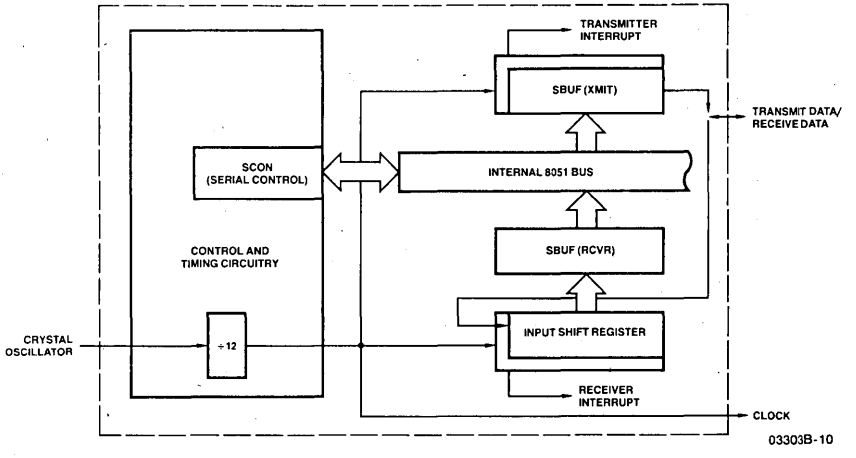


Figure 12. Serial Port – UART Modes 1, 2 and 3

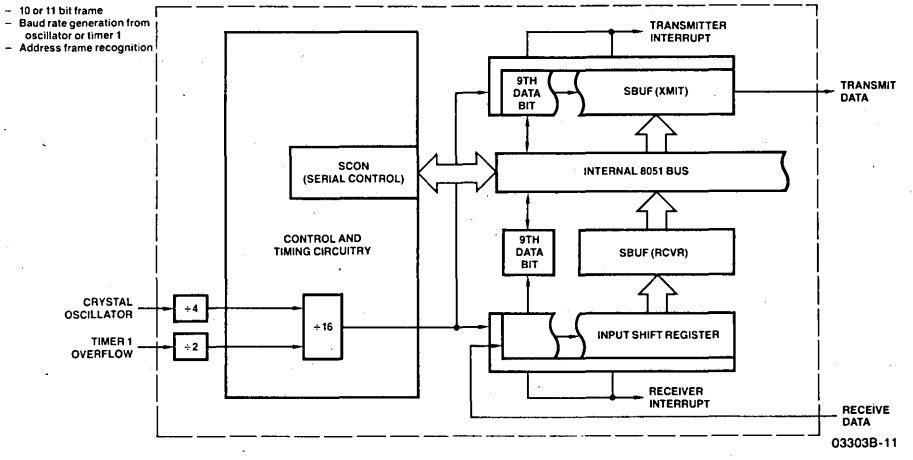


Figure 13. UART Interfacing Schemes

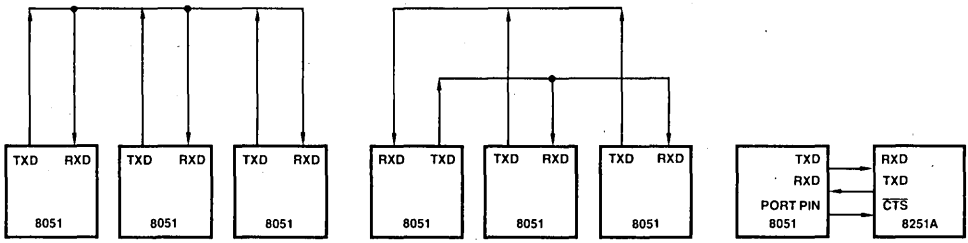


Figure 14. I/O Expansion Technique

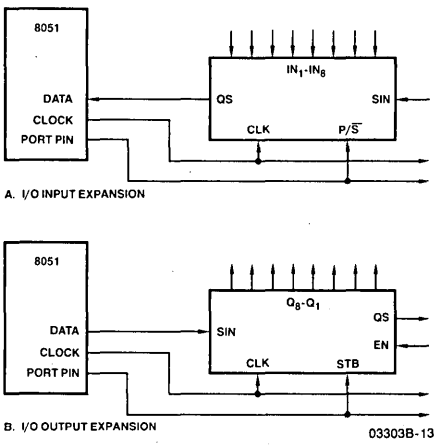
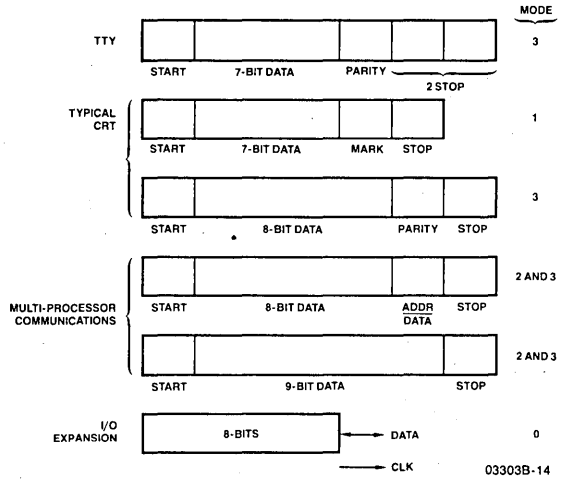


Figure 15. Typical Frame Formats



Distributed processing offers a faster, more powerful system than a single CPU can provide. This results from a hierarchy of interconnected processors, each with its own memories and I/O. In a multiprocessing environment, a single host 8051 controls other slave 8051s configured to operate simultaneously on separate portions of a program. The interconnected 8051s reduce the load on the host processor and result in a lower-cost system of data transmission. This form of distributed processing is especially effective in a complex process where controls are required at physically separated locations.

In Modes 2 and 3 interprocessor communication is facilitated by the automatic wake-up of slave processors through interrupt driven address-frame recognition. The protocol for interprocessor communications is shown in Figure 16. In synchronous mode (Mode 0) the high speed serial port provides an efficient, low-cost method of expanding I/O lines using standard TTL and CMOS shift registers. The serial channel provides a clock output for synchronizing the shifting of bits to/from an external register. The data rate is a division by 12 of the oscillator frequency and hence is 1M bits per second at 12MHz.

Figure 16. Protocol for Multi-Processor Communications

1. Slaves – Configure serial port to interrupt CPU if the received ninth data bit is a one (1).
2. Master – Transmit frame containing address in first 8 data bits and set ninth data bit (i.e., ninth data bit designates address frame).
3. Slaves – Serial port interrupts CPU when address frame is received. Interrupt service program compares received address to its address. The slave which has been addressed reconfigures its serial port to interrupt the CPU on all subsequent transmissions.
4. Master – Transmit control frames and data frames (these will be accepted only by the previously addressed slave).

ABSOLUTE MAXIMUM RATINGS above which useful life may be impaired*

Storage Temperature	−65 to +150°C
Voltage on Any Pin with Respect to Ground	−0.5 to +7.0V
Power Dissipation	2W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling, and use in order to avoid exposure to excessive voltages.

*NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

8051/8031 DC CHARACTERISTICS

Parameters	Description	Test Conditions	Min	Typ	Max	Units
V_{IL}	Input Low Voltage		−0.5		0.8	V
V_{IH}	Input High Voltage (Except RST/V_{PD} and $XTAL_2$)		2.0		V_{CC} +0.5	V
V_{IH1}	Input High Voltage to RST/V_{PD} for Reset, $XTAL_2$	$XTAL_1$ to V_{SS}	2.5			V
V_{PD}	Power Down Voltage to RST/V_{PD}	V_{CC} = 0V	4.5		5.5	V
V_{OL}	Output Low Voltage, Ports 1, 2, 3 (Note 1)	I_{OL} = 1.6mA			0.45	V
V_{OL1}	Output Low Voltage, Port 0, ALE, $\overline{PSEN}$ (Note 1)	I_{OL} = 3.2mA			0.45	V
V_{OH}	Output High Voltage, Ports 1, 2, 3	I_{OH} = −80μA	2.4			V
V_{OH1}	Output High Voltage, Port 0, ALE, $\overline{PSEN}$	I_{OH} = −400μA	2.4			V
I_{IL}	Logical 0 Input Current, $XTAL_2$, Ports 1, 2, 3	$XTAL_1$ at V_{SS} V_{IL} = 0.45V			−800	μA
I_{IH1}	Input High Current to RST/V_{PD} for Reset	V_{IN} = V_{CC} − 1.5V			500	μA
I_{LI}	Input Leakage Current to Port 0, $\overline{EA}$	0 < V_{IN} < V_{CC}			10	μA
I_{CC}	Power Supply Current			125	160	mA
I_{PD}	Power Down Current			10	20	mA
C_{IO}	Capacitance of I/O Buffer	fc = 1MHz			10	pF

Note 1. V_{OL} is degraded when the 8051 rapidly discharges external capacitance. This AC noise is most pronounced during emission of address data. When using external memory, locate the latch or buffer as close to the 8051 as possible.

8051 PROGRAM VERIFICATION

To ensure correct factory masked ROM, the following procedure may be followed. The address of the location to be verified is input on Port 1 (pins 1 through 8) while Port 2 (pins 21 through 28) and PSEN (pin 29) are held low. RST/VPD (pin 9) and ALE (pin 30) are held high. The data to be verified is read out through Port 0 (pins 32 through 39).

Datum	Emitting Ports	Time Interval	Degraded I/O Lines	V_{OL} (peak) (max)
Address	P2, P0	T3, T9	P1, P3	0.8V
Write Data	P0	T6	P1, P3, ALE	0.8V

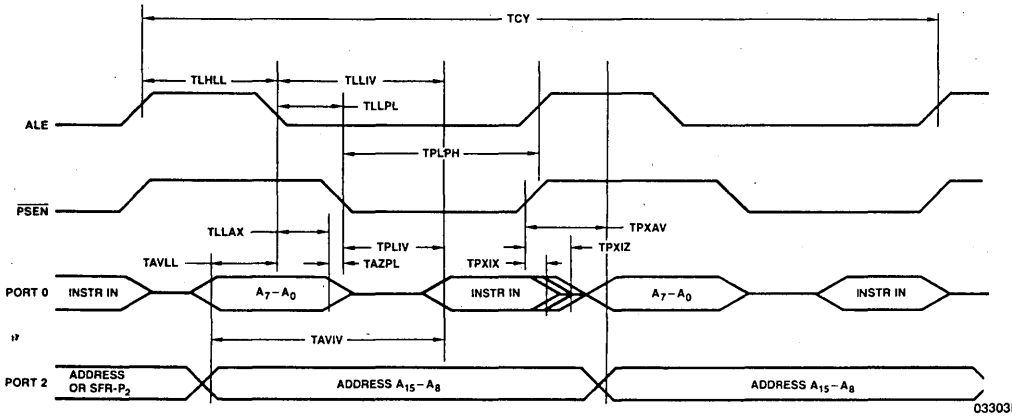
8051/8031 AC CHARACTERISTICS (C_L for Port 0, ALE and $\overline{PSEN}$ Outputs = 100pF; C_L for All Other Outputs = 80pF)

Parameter	Description	12MHz Clock		Variable Clock 1/TCLCL = 1.2MHz to 12MHz		Units
		Min	Max	Min	Max	
PROGRAM MEMORY						
TCY	Min Instruction Cycle Time (Note 3)	1.0		12TCLCL	12TCLCL	ns
TLHLL	ALE Pulse Width	127		2TCLCL−40		ns
TAVLL	Address Setup to ALE	53		TCLCL−30		ns
TLLAX	Address Hold After ALE (Note 1)	48		TCLCL−35		ns
TLLIV	ALE to Valid Instruction In		233		4TCLCL−100	ns
TLLPL	ALE to $\overline{PSEN}$	58		TCLCL−25		ns
TPLPH	$\overline{PSEN}$ Pulse Width	215		3TCLCL−35		ns
TPLIV	$\overline{PSEN}$ to Valid Instruction In		125		3TCLCL−125	ns
TPXIX	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
TPXIZ	Input Instruction Float After $\overline{PSEN}$ (Note 2)		63		TCLCL−20	ns
TPXAV	Address Valid After $\overline{PSEN}$ (Note 2)	75		TCLCL−8		ns
TAVIV	Address to Valid Instruction In		302		5TCLCL−115	ns
TAZPL	Address Float to $\overline{PSEN}$	0		0		ns
EXTERNAL DATA MEMORY						
TRLRH	$\overline{RD}$ Pulse Width	400		6TCLCL−100		ns
TWLWH	$\overline{WR}$ Pulse Width	400		6TCLCL−100		ns
TLLAX	Address Hold After ALE (Note 1)	132		2TCLCL−35		ns
TRLDV	$\overline{RD}$ to Valid Data In		250		5TCLCL−165	ns
TRHDX	Data Hold After $\overline{RD}$	0		0		ns
TRHDZ	Data Float After $\overline{RD}$		97		2TCLCL−70	ns
TLLDV	ALE to Valid Data In		517		8TCLCL−150	ns
TAVDV	Address to Valid Data In		585		9TCLCL−165	ns
TLLWL	ALE to $\overline{WR}$ or $\overline{RD}$	200	300	3TCLCL−50	3TCLCL+50	ns
TAVWL	Address to $\overline{WR}$ or $\overline{RD}$	203		4TCLCL−130		ns
TWHLH	$\overline{WR}$ or $\overline{RD}$ High to ALE High	43	123	TCLCL−40	TCLCL+40	ns
TDVWX	Data Valid to $\overline{WR}$ Transition	33		TCLCL−50		ns
TQVWH	Data Setup Before $\overline{WR}$	433		7TCLCL−150		ns
TWHQX	Data Hold After $\overline{WR}$	33		TCLCL−50		ns
TRLAZ	Address Float After $\overline{RD}$		0		0	ns

Notes: 1. TLLAX for access to Program Memory is different from TLLAX for access to Data Memory.
2. Interfacing the 8051 to devices with float times up to 75ns is permissible. This limited bus contention will not cause any damage to Port 0 drivers.
3. TCY is the minimum instruction cycle time which consists of 12 oscillator clocks or two ALE cycles.

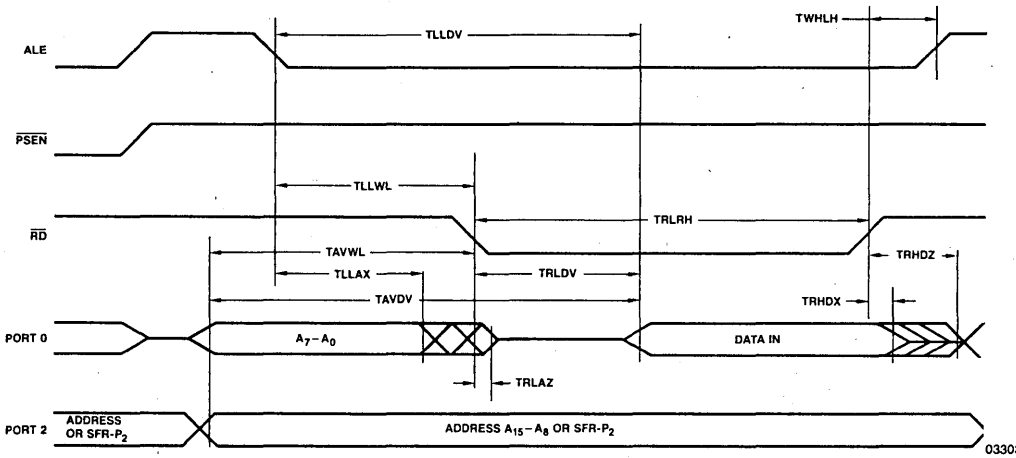
WAVEFORMS

PROGRAM MEMORY READ CYCLE



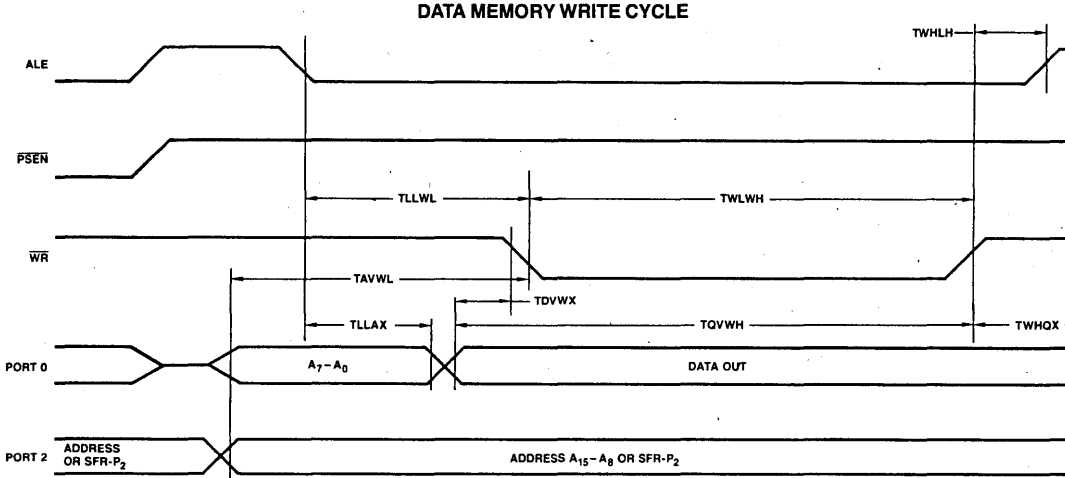
03303B-15

DATA MEMORY READ CYCLE



03303B-16

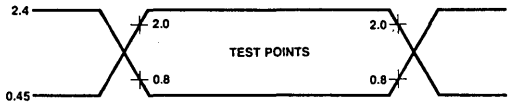
DATA MEMORY WRITE CYCLE



03303B-17

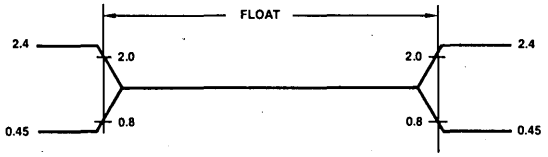
AC TESTING INPUT, OUTPUT, FLOAT WAVEFORMS

INPUT/OUTPUT



03303B-18

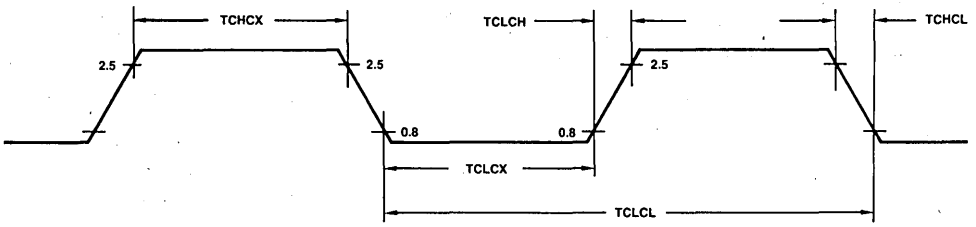
FLOAT



03303B-19

AC testing inputs are driven at 2.4V for a logic "1" and 0.45V for a logic "0."
Timing measurements are made at 2.0V for a logic "1" and 0.8V for a logic "0."
For timing purposes, the float state is defined as the point at which a P_0 pin sinks 3.2mA or sources 400μA at the voltage test levels.

EXTERNAL CLOCK DRIVE $XTAL_2$

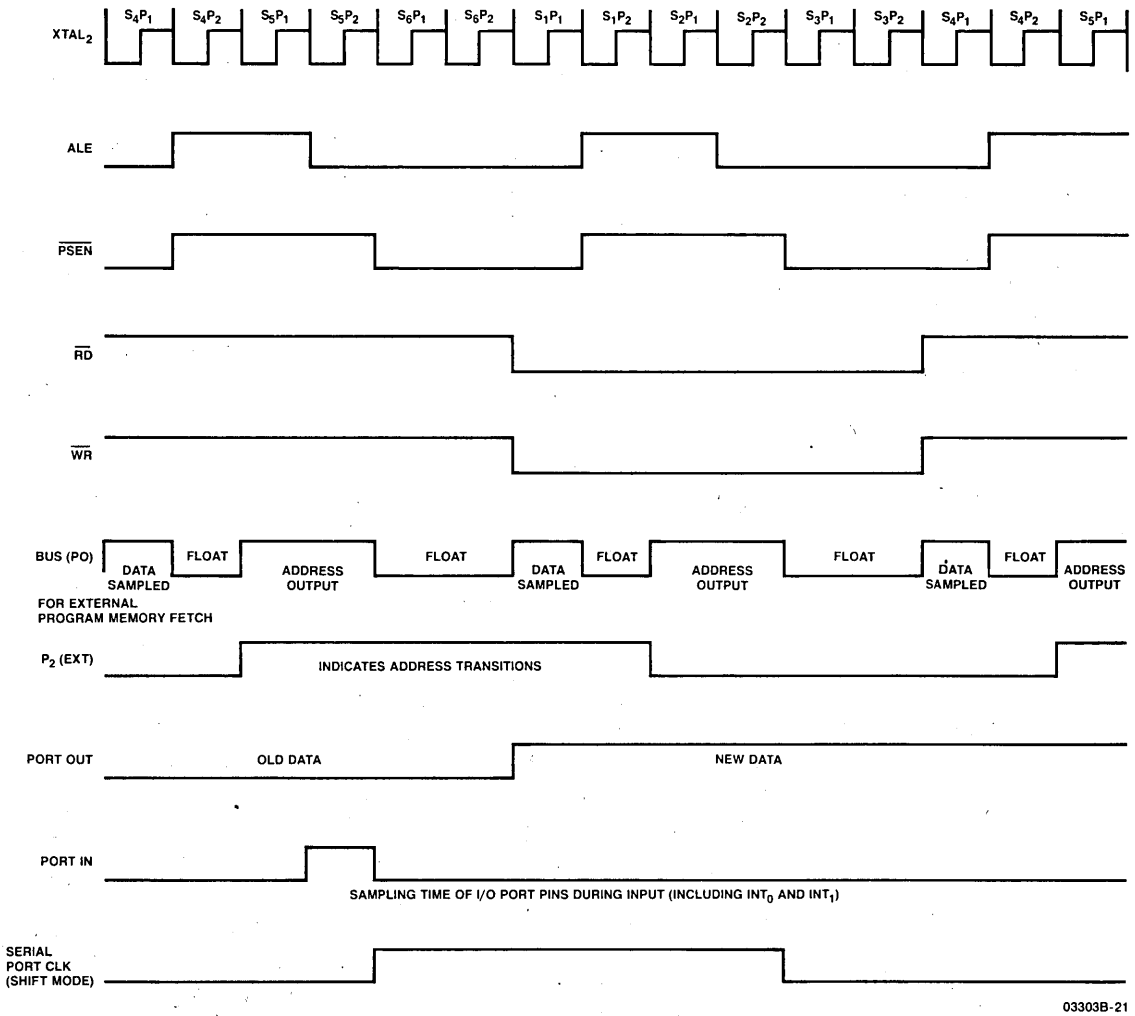


03303B-20

CLOCKING DETAILS

Symbol	Parameters	Variable Clock Freq = 1.2MHz to 12MHz		Unit
		Min	Max	
TCLCL	Oscillator Period	83.3	833.3	ns
TCHCX	High Time	20	TCLCL-TCLCX	ns
TCLCX	Low Time	20	TCLCL-TCHCX	ns
TCLCH	Rise Time		20	ns
TCHCL	Fall Time		20	ns

WAVEFORMS



All internal timing is referenced to the internal time states shown at the top of the page. This waveform represents the signal on the X_2 input of the oscillator. This diagram represents when these signals are actually clocked within the chip. However, the time it takes a signal to propagate to the pins is in the range of 50 – 150ns. Prop delays are dependent on many variables, such as temperature, pin loading. Even the different signals vary. Typically though, /RD and /WR have prop delays of approximately 50ns and the other timing signals approximately 85ns, at room temperature, fully loaded. These differences in prop delays between signals have been integrated into the timing specs.

TABLE 1. 8051/8031 INSTRUCTION SET

INSTRUCTIONS THAT AFFECT FLAG SETTINGS*

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	O		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C, bit	X		
MUL	O	X		ANL C,/bit	X		
DIV	O	X		ORL C, bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C, bit	X		
RLC	X			CJNE	X		
SETB C	1						

Interrupt Response Time: To finish execution of current instruction, respond to the interrupt request, push the PC and to vector to the first instruction of the interrupt service program requires 38 to 81 oscillator periods (3 to 7μs @ 12MHz).

*Note that operations on SFR byte address 208 or bit addresses 209-215 (i.e., the PSW or bits in the PSW) will also affect flag settings.

DATA TRANSFER (Note 1)

Mnemonic		Description	Bytes	Cycle
MOV	A,Rn	Move register to Accumulator	1	1
*MOV	A,direct	Move direct byte to Accumulator	2	1
MOV	A,@Ri	Move indirect RAM to Accumulator	1	1
MOV	A,#data	Move immediate data to Accumulator	2	1
MOV	Rn,A	Move Accumulator to register	1	1
*MOV	Rn,direct	Move direct byte to register	2	2
MOV	Rn,#data	Move immediate data to register	2	1
*MOV	direct,A	Move Accumulator to direct byte	2	1
*MOV	direct,Rn	Move register to direct byte	2	2
*MOV	direct,direct	Move direct byte to direct byte	3	2
*MOV	direct,@Ri	Move indirect RAM to direct byte	2	2
*MOV	direct,#data	Move immediate data to direct byte	3	2
MOV	@Ri,A	Move Accumulator to indirect RAM	1	1
*MOV	@Ri,direct	Move direct byte to indirect RAM	2	2
MOV	@Ri,#data	Move immediate data to indirect RAM	2	1
*MOV	DPTR, #data16	Move 16-bit constant to Data Pointer	3	2
*MOVC	A,@A+ DPTR	Move Code byte relative to DPTR to Accumulator	1	2
*MOVC	A,@A+PC	Move Code byte relative to PC to Accumulator	1	2
MOVX	A,@Ri	Move External RAM (8-bit address) to Accumulator	1	2
*MOVX	A,@DPTR	Move External RAM (16-bit address) to Accumulator	1	2
MOVX	@Ri,A	Move Accumulator to External RAM (8-bit address)	1	2
*MOVX	@DPTR,A	Move Accumulator to External RAM (16-bit address)	1	2
*PUSH	direct	Push direct byte onto stack	2	2
*POP	direct	Pop direct byte off of stack	2	2
XCH	A,Rn	Exchange register with Accumulator	1	1
*XCH	A,direct	Exchange direct byte with Accumulator	2	1
XCH	A,@Ri	Exchange indirect RAM with Accumulator	1	1
XCHD	A,@Ri	Exchange indirect RAM's least sig nibble with A's LSN	1	1

*New operation not provided by 8048/8049 Family

All mnemonics copyrighted© Intel Corporation 1980.

BOOLEAN VARIABLE MANIPULATION

Mnemonic		Description	Byte	Cycle
CLR	C	Clear Carry Flag	1	1
*CLR	bit	Clear direct bit	2	1
*SETB	C	Set Carry Flag	1	1
*SETB	bit	Set direct bit	2	1
CPL	C	Complement Carry Flag	1	1
*CPL	bit	Complement direct bit	2	1
*ANL	C,bit	AND direct bit to Carry Flag	2	2
*ANL	C,/bit	AND complement of direct bit to Carry	2	2
*ORL	C,bit	OR direct bit to Carry Flag		
*ORL	C,/bit	OR complement of direct bit to Carry	2	2
*MOV	C,bit	Move direct bit to Carry Flag	2	1
*MOV	bit,C	Move Carry Flag to direct bit	2	2

LOGIC

Mnemonic		Description	Bytes	Cycle
ANL	A,Rn	AND register to Accumulator	1	1
*ANL	A,direct	AND direct byte to Accumulator	2	1
ANL	A,@Ri	AND indirect RAM to Accumulator	1	1
ANL	A,#data	AND immediate data to Accumulator	2	1
*ANL	direct,A	AND Accumulator to direct byte	2	1
*ANL	direct,#data	AND immediate data to direct byte	3	2
ORL	A,Rn	OR register to Accumulator	1	1
*ORL	A,direct	OR direct byte to Accumulator	2	1
ORL	A,@Ri	OR indirect RAM to Accumulator	1	1
ORL	A,#data	OR immediate data to Accumulator	2	1
*ORL	direct,A	OR Accumulator to direct byte	2	1
*ORL	direct,#data	OR immediate data to direct byte	3	2
XRL	A,Rn	Exclusive-OR register to Accumulator	1	1
*XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	1
XRL	A,@Ri	Exclusive-OR indirect RAM to Accumulator	1	1
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	1
*XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	1

Note 1: Special care should be taken (particularly with the 8031) when using the MOV instruction. The MOV instruction should *not* be used to move data on port 0 nor 2 when these ports are used to address external memory.

TABLE 1. 8051/8031 INSTRUCTION SET (Cont.)

LOGIC (Cont.)

Mnemonic		Description	Bytes	Cycle
*XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	2
CLR	A	Clear Accumulator	1	1
CPL	A	Complement Accumulator	1	1
RL	A	Rotate Accumulator Left	1	1
RLC	A	Rotate Accumulator Left through Carry Flag	1	1
RR	A	Rotate Accumulator Right	1	1
RRC	A	Rotate Accumulator Right through Carry Flag	1	1
SWAP	A	Exchange nibbles within Accumulator	1	1

ARITHMETIC

Mnemonic		Description	Bytes	Cycle
ADD	A,Rn	Add register to Accumulator	1	1
*ADD	A,direct	Add direct byte to Accumulator	2	1
ADD	A,@Ri	Add indirect RAM to Accumulator	1	1
ADD	A,#data	Add immediate data to Accumulator	2	1
ADDC	A,Rn	Add register to Accumulator with carry	1	1
*ADDC	A,direct	Add direct byte to Accumulator with Carry Flag	2	1
ADDC	A,@Ri	Add indirect RAM and Carry Flag to Accumulator	1	1
ADDC	A,#data	Add immediate data and Carry Flag to Accumulator	2	1
*SUBB	A,Rn	Subtract register from Accumulator with Borrow	1	1
*SUBB	A,direct	Subtract direct byte from Accumulator with Borrow	2	1
*SUBB	A,@Ri	Subtract indirect RAM from Accumulator with Borrow	1	1
*SUBB	A,#data	Subtract immediate data from Accumulator with Borrow	2	1
INC	A	Increment Accumulator	1	1
INC	Rn	Increment register	1	1
*INC	direct	Increment direct byte	2	1
INC	@Ri	Increment indirect RAM	1	1
DEC	A	Decrement Accumulator	1	1
DEC	Rn	Decrement register	1	1
*DEC	direct	Decrement direct byte	2	1
*DEC	@Ri	Decrement indirect RAM	1	1
*INC	DPTR	Increment Data Pointer	1	2
*MUL	AB	Multiply Accumulator times B	1	4
*DIV	AB	Divide Accumulator by B	1	4
DA	A	Decimal Adjust Accumulator	1	1

OTHER

Mnemonic	Description	Bytes	Cycle
NOP	No Operation	1	1

CONTROL TRANSFER (BRANCH)

Mnemonic		Description	Bytes	Cycle
AJMP	addr11	Absolute Jump	2	2
*LJMP	addr16	Long Jump	3	2
*SJMP	rel	Short Jump (relative addr)	2	2
*JMP	@A+DPTR	Jump indirect relative to the DPTR	1	2
JZ	rel	Jump if Accumulator is zero	2	2
JNZ	rel	Jump if Accumulator is not zero	2	2
JC	rel	Jump if Carry Flag is set	2	2
JNC	rel	Jump if carry is not set	2	2
*JB	bit,rel	Jump relative if direct bit is set	3	2
*JNB	bit,rel	Jump relative if direct bit is not set	3	2
*JBC	bit,rel	Jump relative if direct bit is set, then clear bit	3	2
*CJNE	A,direct,rel	Compare direct byte to Accumulator and Jump if not Equal	3	2
*CJNE	A,#data,rel	Compare immediate to Accumulator and Jump if not Equal	3	2
*CJNE	Rn,#data, rel	Compare immediate to reg and Jump if not Equal	3	2
*CJNE	@Ri, #data,rel	Compare immed. to indirect RAM and Jump if not Equal	3	2
DJNZ	Rn,rel	Decrement register and Jump if not zero	2	2
*DJNZ	direct,rel	Decrement direct byte and Jump if not zero	3	2

CONTROL TRANSFER (SUBROUTINE)

Mnemonic		Description	Bytes	Cycle
ACALL	addr11	Absolute Subroutine Call	2	2
LCALL	addr16	Long Subroutine Call	3	2
RET		Return from Subroutine Call	1	2
RETI		Return from Interrupt Call	1	2

Notes on data addressing modes:

- Rn – Working register R0 – R7 of the currently selected Register bank.
- direct – 128 internal RAM locations, any I/O port, control, or status register.
- @Ri – Indirect internal RAM location addressed by register R0 or R1.
- #data – 8-bit constant included in instruction.
- #data16 – 16-bit constant included as bytes 2 and 3 of instruction.
- bit – 128 software flags, any I/O pin, control, or status bit.

Notes on program addressing modes:

- addr16 – Destination address for LCALL and LJMP may be anywhere within the 64-Kilobyte program memory address space.
- addr11 – Destination address for ACALL and AJMP will be within the same 2-Kilobyte page of program memory as the first byte of the following instruction.
- rel – SJMP and all conditional jumps include an 8-bit offset byte. Range is +127, −128 bytes relative to first byte of the following instruction.

*New operation not provided by 8048/8049 Family

TABLE 2. INSTRUCTION OPCODES IN HEXADECIMAL ORDER

Hex Code	Bytes	Mnemonic	Operands	Hex Code	Bytes	Mnemonic	Operands
00	1	NOP		42	2	ORL	Data addr,A
01	2	AJMP	Code addr	43	3	ORL	Data addr,#data
02	3	LJMP	Code addr	44	2	ORL	A,#data
03	1	RR	A	45	2	ORL	A,data addr
04	1	INC	A	46	1	ORL	A,@R0
05	2	INC	Data addr	47	1	ORL	A,@R1
06	1	INC	@R0	48	1	ORL	A,R0
07	1	INC	@R1	49	1	ORL	A,R1
08	1	INC	R0	4A	1	ORL	A,R2
09	1	INC	R1	4B	1	ORL	A,R3
0A	1	INC	R2	4C	1	ORL	A,R4
0B	1	INC	R3	4D	1	ORL	A,R5
0C	1	INC	R4	4E	1	ORL	A,R6
0D	1	INC	R5	4F	1	ORL	A,R7
0E	1	INC	R6	50	2	JNC	Code addr
0F	1	INC	R7	51	2	ACALL	Code addr
10	3	JBC	Bit addr,code addr	52	2	ANL	Data addr,A
11	2	ACALL	Code addr	53	3	ANL	Data addr,#data
12	3	LCALL	Code addr	54	2	ANL	A,#data
13	1	RRC	A	55	2	ANL	A,data addr
14	1	DEC	A	56	1	ANL	A,@R0
15	2	DEC	Data addr	57	1	ANL	A,@R1
16	1	DEC	@R0	58	1	ANL	A,R0
17	1	DEC	@R1	59	1	ANL	A,R1
18	1	DEC	R0	5A	1	ANL	A,R2
19	1	DEC	R1	5B	1	ANL	A,R3
1A	1	DEC	R2	5C	1	ANL	A,R4
1B	1	DEC	R3	5D	1	ANL	A,R5
1C	1	DEC	R4	5E	1	ANL	A,R6
1D	1	DEC	R5	5F	1	ANL	A,R7
1E	1	DEC	R6	60	2	JZ	Code addr
1F	1	DEC	R7	61	2	AJMP	Code addr
20	3	JB	Bit addr,code addr	62	2	XRL	Data addr,A
21	2	AJMP	Code addr	63	3	XRL	Data addr,#data
22	1	RET		64	2	XRL	A, #data
23	1	RL	A	65	2	XRL	A,data addr
24	2	ADD	A,#data	66	1	XRL	A,@R0
25	2	ADD	A,data addr	67	1	XRL	A,@R1
26	1	ADD	A,@R0	68	1	XRL	A,R0
27	1	ADD	A,@R1	69	1	XRL	A,R1
28	1	ADD	A,R0	6A	1	XRL	A,R2
29	1	ADD	A,R1	6B	1	XRL	A,R3
2A	1	ADD	A,R2	6C	1	XRL	A,R4
2B	1	ADD	A,R3	6D	1	XRL	A,R5
2C	1	ADD	A,R4	6E	1	XRL	A,R6
2D	1	ADD	A,R5	6F	1	XRL	A,R7
2E	1	ADD	A,R6	70	2	JNZ	Code addr
2F	1	ADD	A,R7	71	2	ACALL	Code addr
30	3	JNB	Bit addr,code addr	72	2	ORL	C,bit addr
31	2	ACALL	Code addr	73	1	JMP	@A+DPTR
32	1	RETI		74	2	MOV	A,#data
33	1	RLC	A	75	3	MOV	Data addr,#data
34	2	ADDC	A,#data	76	2	MOV	@R0,#data
35	2	ADDC	A,data addr	77	2	MOV	@R1,#data
36	1	ADDC	A,@R0	78	2	MOV	R0,#data
37	1	ADDC	A,@R1	79	2	MOV	R1,#data
38	1	ADDC	A,R0	7A	2	MOV	R2,#data
39	1	ADDC	A,R1	7B	2	MOV	R3,#data
3A	1	ADDC	A,R2	7C	2	MOV	R4,#data
3B	1	ADDC	A,R3	7D	2	MOV	R5,#data
3C	1	ADDC	A,R4	7E	2	MOV	R6,#data
3D	1	ADDC	A,R5	7F	2	MOV	R7,#data
3E	1	ADDC	A,R6	80	2	SJMP	Code addr
3F	1	ADDC	A,R7	81	2	AJMP	Code addr
40	2	JC	Code addr	82	2	ANL	C,bit addr
41	2	AJMP	Code addr	83	1	MOVC	A,@A+PC

TABLE 2. INSTRUCTION OPCODES IN HEXADECIMAL ORDER (Cont.)

Hex Code	Bytes	Mnemonic	Operands	Hex Code	Bytes	Mnemonic	Operands
84	1	DIV	AB	C2	2	CLR	Bit addr
85	3	MOV	Data addr,data addr	C3	1	CLR	C
86	2	MOV	Data addr,@R0	C4	1	SWAP	A
87	2	MOV	Data addr,@R1	C5	2	XCH	A,data addr
88	2	MOV	Data addr,R0	C6	1	XCH	A,@R0
89	2	MOV	Data addr,R1	C7	1	XCH	A,@R1
8A	2	MOV	Data addr,R2	C8	1	XCH	A,R0
8B	2	MOV	Data addr,R3	C9	1	XCH	A,R1
8C	2	MOV	Data addr,R4	CA	1	XCH	A,R2
8D	2	MOV	Data addr,R5	CB	1	XCH	A,R3
8E	2	MOV	Data addr,R6	CC	1	XCH	A,R4
8F	2	MOV	Data addr,R7	CD	1	XCH	A,R5
90	3	MOV	DPTR,#data	CE	1	XCH	A,R6
91	2	ACALL	Code addr	CF	1	XCH	A,R7
92	2	MOV	Bit addr,C	D0	2	POP	Data addr
93	1	MOVC	A,@A+DPTR	D1	2	ACALL	Code addr
94	2	SUBB	A,#data	D2	2	SETB	Bit addr
95	2	SUBB	A,data addr	D3	1	SETB	C
96	1	SUBB	A,@R0	D4	1	DA	A
97	1	SUBB	A,@R1	D5	3	DJNZ	Data addr,code addr
98	1	SUBB	A,R0	D6	1	XCHD	A,@R0
99	1	SUBB	A,R1	D7	1	XCHD	A,@R1
9A	1	SUBB	A,R2	D8	2	DJNZ	R0,code addr
9B	1	SUBB	A,R3	D9	2	DJNZ	R1,code addr
9C	1	SUBB	A,R4	DA	2	DJNZ	R2,code addr
9D	1	SUBB	A,R5	DB	2	DJNZ	R3,code addr
9E	1	SUBB	A,R6	DC	2	DJNZ	R4,code addr
9F	1	SUBB	A,R7	DD	2	DJNZ	R5,code addr
A0	2	ORL	C,/bit addr	DE	2	DJNZ	R6,code addr
A1	2	AJMP	Code addr	DF	2	DJNZ	R7,code addr
A2	2	MOV	C,bit addr	E0	1	MOVX	A,@DPTR
A3	1	INC	DPTR	E1	2	AJMP	Code addr
A4	1	MUL	AB	E2	1	MOVX	A,@R0
A5		Reserved		E3	1	MOVX	A,@R1
A6	2	MOV	@R0,data addr	E4	1	CLR	A
A7	2	MOV	@R1,data addr	E5	2	MOV	A,data addr
A8	2	MOV	R0,data addr	E6	1	MOV	A,@R0
A9	2	MOV	R1,data addr	E7	1	MOV	A,@R1
AA	2	MOV	R2,data addr	E8	1	MOV	A,R0
AB	2	MOV	R3,data addr	E9	1	MOV	A,R1
AC	2	MOV	R4,data addr	EA	1	MOV	A,R2
AD	2	MOV	R5,data addr	EB	1	MOV	A,R3
AE	2	MOV	R6,data addr	EC	1	MOV	A,R4
AF	2	MOV	R7,data addr	ED	1	MOV	A,R5
B0	2	ANL	C,/bit addr	EE	1	MOV	A,R6
B1	2	ACALL	Code addr	EF	1	MOV	A,R7
B2	2	CPL	Bit addr	F0	1	MOVX	@DPTR,A
B3	1	CPL	C	F1	2	ACALL	Code addr
B4	3	CJNE	A,#data,code addr	F2	1	MOVX	@R0,A
B5	3	CJNE	A,data addr,code addr	F3	1	MOVX	@R1,A
B6	3	CJNE	@R0,#data,code addr	F4	1	CPL	A
B7	3	CJNE	@R1,#data,code addr	F5	2	MOV	Data addr,A
B8	3	CJNE	R0,#data,code addr	F6	1	MOV	@R0,A
B9	3	CJNE	R1,#data,code addr	F7	1	MOV	@R1,A
BA	3	CJNE	R2,#data,code addr	F8	1	MOV	R0,A
BB	3	CJNE	R3,#data,code addr	F9	1	MOV	R1,A
BC	3	CJNE	R4,#data,code addr	FA	1	MOV	R2,A
BD	3	CJNE	R5,#data,code addr	FB	1	MOV	R3,A
BE	3	CJNE	R6,#data,code addr	FC	1	MOV	R4,A
BF	3	CJNE	R7,#data,code addr	FD	1	MOV	R5,A
C0	2	PUSH	Data addr	FE	1	MOV	R6,A
C1	2	AJMP	Code addr	FF	1	MOV	R7,A

8-Bit CPUs and Microcomputers

8080A/9080A

8-Bit Microprocessor

DISTINCTIVE CHARACTERISTICS

- Plug-in replacements for 8080A, 8080A-1, 8080A-2
- High-speed version with 1μsec instruction cycle
- Military temperature range operation to 1.5μsec
- Ion-implanted, n-channel, silicon-gate MOS technology
- 3.2mA of output drive at 0.4V (two full TTL loads)
- 700mV of high, 400mV of low level noise immunity
- 820mW maximum power dissipation at ±5% power

GENERAL DESCRIPTION

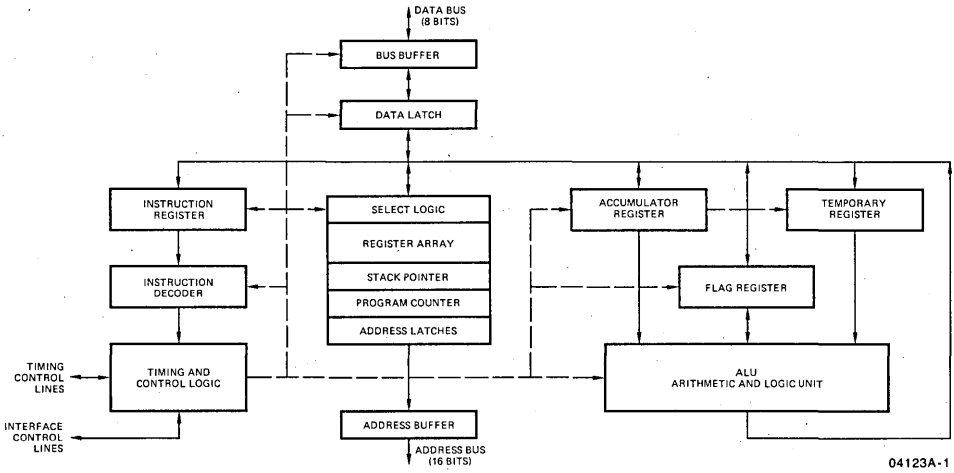
The 8080A products are complete, general-purpose, single-chip digital processors. They are fixed instruction set, parallel, 8-bit units fabricated with Advanced N-Channel Silicon Gate MOS technology. When combined with external memory and peripheral devices, powerful microcomputer systems are formed. The 8080A may be used to perform a wide variety of operations, ranging from complex arithmetic calculations to character handling to bit control. Several versions are available offering a range of performance options.

The processor has a 16-bit address bus that may be used to directly address up to 64K bytes of memory. The memory may be any combination of read/write and read-only. Data are transferred into or out of the processor on a bi-directional 8-bit data bus that is separate from the address lines. The data bus transfers instructions, data and status information between system devices. All transfers are handled using asynchronous handshaking controls so that any speed memory or I/O device is easily accommodated.

An accumulator plus six general registers are available to the programmer. The six registers are each 8 bits long and may be used singly or in pairs for both 8 and 16-bit operations. The accumulator forms the primary working register and is the destination for many of the arithmetic and logic operations.

A general purpose push-down stack is an important part of the processor architecture. The contents of the stack reside in R/W memory and the control logic, including a 16-bit stack pointer, is located on the processor chip. Subroutine call and return instructions automatically use the stack to store and retrieve the contents of the program counter. Push and Pop instructions allow direct use of the stack for storing operands, passing parameters and saving the machine state.

BLOCK DIAGRAM



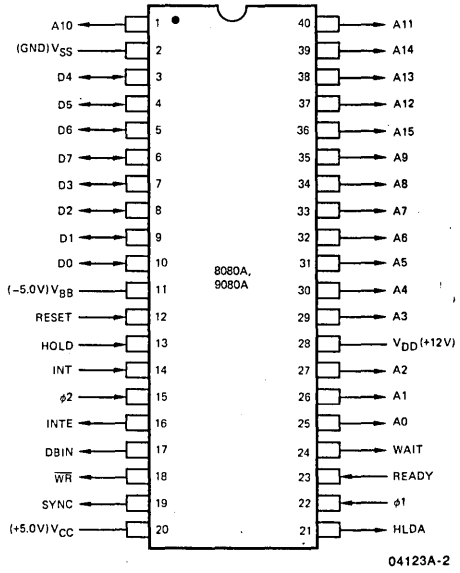
04123A-1

ORDERING INFORMATION

Package Type	Ambient Temperature Specification	Minimum Clock Period		
		320ns	380ns	480ns
Molded DIP	$0°C \leq T_A \leq 70°C$	P8080A-1	P8080A-2	P8080A
Hermetic DIP		D8080A-1	D8080A-2	D8080A
Hermetic DIP	$-40°C \leq T_A \leq 85°C$	ID8080A-1	ID8080A-2	ID8080A
Hermetic DIP	$-55°C \leq T_A \leq 125°C$		MD8080A-2B	MD8080AB
			Am9080A-2DMB	Am9080ADMB

04123A-MMP

CONNECTION DIAGRAM – Top View
D-40, P-40



04123A-2

Note: Pin 1 is marked for orientation.

INTERFACE SIGNAL SUMMARY

TYPE	PINS	ABBREVIATION	SIGNAL
INPUT	1	V_{SS}	Ground
INPUT	3	V_{DD}, V_{CC}, V_{BB}	+12V, +5V, –5V Supplies
INPUT	2	$\phi 1$, $\phi 2$	Clocks
INPUT	1	RESET	Reset
INPUT	1	HOLD	Hold
INPUT	1	INT	Interrupt
INPUT	1	READY	Ready
IN/OUT	8	D_0–D_7	Data Bus
OUTPUT	16	A_0–A_{15}	Address
OUTPUT	1	INTE	Interrupt Enable
OUTPUT	1	DBIN	Data Bus In Control
OUTPUT	1	$\overline{WR}$	Write Not
OUTPUT	1	SYNC	Cycle Synchronization
OUTPUT	1	HLDA	Hold Acknowledge
OUTPUT	1	WAIT	Wait

INTERFACE SIGNAL DESCRIPTION

$\phi 1$, $\phi 2$ The Clock inputs provide basic timing generation for all internal operations. They are non-overlapping two phase, high level signals. All other inputs to the processor are TTL compatible.

RESET The Reset input initializes the processor by clearing the program counter, the instruction register, the interrupt enable flip-flop and the hold acknowledge flip-flop. The Reset signal should be active for at least three clock periods. The general registers are not cleared.

HOLD The Hold input allows an external signal to cause the processor to relinquish control over the address lines and the data bus. When Hold goes active, the processor completes its current operation, activates the Hlda output, and puts the 3-state address and data lines into their high-impedance state. The Holding device can then utilize the address and data busses without interference.

READY The Ready input synchronizes the processor with external units. When Ready is absent, indicating the external operation is not complete, the processor will enter the Wait state. It will remain in the Wait state until the clock cycle following the appearance of Ready.

INT The Interrupt input signal provides a mechanism for external devices to modify the instruction flow of the program in progress. Interrupt requests are handled efficiently with the vectored interrupt procedure and the general purpose stack. Interrupt processing is described in more detail on the next page.

D_0-D_7 The Data Bus is comprised of 8 bidirectional signal lines for transferring data, instructions and status information between the processor and all external units.

A_0-A_{15} The Address Bus is comprised of 16 output signal lines used to address memory and peripheral devices.

SYNC The Sync output indicates the start of each processor cycle and the presence of processor status information on the data bus.

DBIN The Data Bus In output signal indicates that the bidirectional data bus is in the input mode and incoming data may be gated onto the Data Bus.

WAIT The Wait output indicates that the processor has entered the Wait state and is prepared to accept a Ready from the current external operation.

$\overline{WR}$ The Write output indicates the validity of output on the data bus during a write operation.

HLDA The Hold Acknowledge output signal is a response to a Hold input. It indicates that processor activity has been suspended and the Address and Data Bus signals will enter their high impedance state.

INTE The Interrupt Enable output signal shows the status of the interrupt enable flip-flop, indicating whether or not the processor will accept interrupts.

INSTRUCTION SET INTRODUCTION

The instructions executed by the 8080A are variable length and may be one, two or three bytes long. The length is determined by the nature of the operation being performed and the addressing mode being used.

The instruction summary shows the number of successive memory bytes occupied by each instruction, the number of clock cycles required for the execution of the instruction, the binary coding of the first byte of each instruction, the mnemonic coding used by assemblers and a brief description of each operation. Some branch-type instructions have two execution times depending on whether the conditional branch is taken or not. Some fields in the binary code are labeled with alphabetic abbreviations. That shown as **vvv** is the address pointer used in the one-byte Call instruction (RST). Those shown as **ddd** or **sss** designate destination and source register fields that may be filled as follows:

111 A register
000 B register
001 C register
010 D register
011 E register
100 H register
101 L register
110 Memory

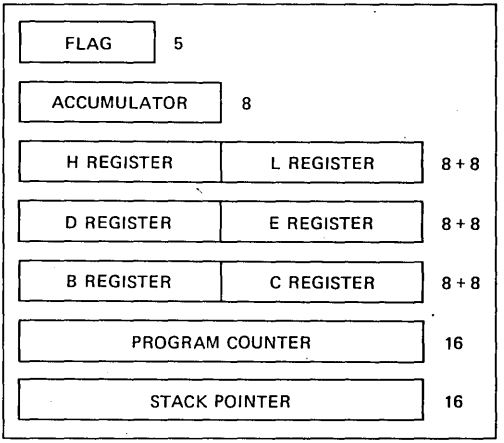
The register diagram shows the internal registers that are directly available to the programmer. The accumulator is the primary working register for the processor and is a specified or implied operand in many instructions. All I/O operations take place via the accumulator. Registers H, L, D, E, B and C may be used singly or in the indicated pairs. The H and L pair is the implied address pointer for many instructions.

The Flag register stores the program status bits used by the conditional branch instructions: carry, zero, sign and parity. The fifth flag bit is the intermediate carry bit. The flags and the accumulator can be stored on or retrieved from the stack with a single instruction. Bit positions in the flag register when pushed onto the stack (PUSH PSW) are:

7	6	5	4	3	2	1	0
S	Z	0	CY1	0	P	1	CY2

where S = sign, Z = zero, CY1 = intermediate carry, P = parity, CY2 = carry.

REGISTER DIAGRAM



During Sync time at the beginning of each instruction cycle the data bus contains operation status information that describes the machine cycle being executed. Positions for the status bits are:

7	6	5	4	3	2	1	0
MEMR	INP	M1	OUT	HLTA	STK	$\overline{WO}$	INTA

STATUS DEFINITION:

INTA Interrupt Acknowledge. Occurs in response to an Interrupt input and indicates that the processor will be ready for an interrupt instruction on the data bus when DBIN goes true.

$\overline{WO}$ Write or Output indicated when signal is low. When high, a Read or Input will occur.

STK Stack indicates that the content of the stack pointer is on the address bus.

HLTA Halt Acknowledge.

OUT Output instruction is being executed.

M1 First instruction byte is being fetched.

INP Input instruction is being executed.

MEMR Memory Read operation.

INTERRUPT PROCESSING

When the processor interrupt mechanism is enabled (INTE=1), interrupt signals from external devices will be recognized unless the processor is in the Hold State. In handling an interrupt, the processor will complete the execution of the current instruction, disable further interrupts and respond with INTA status instead of executing the next sequential instruction in the interrupted program.

The interrupting device should supply an instruction opcode to the processor during the next DBIN time after INTA status appears.

Any opcode may be used except XTHL. If the instruction supplied is a single byte instruction, it will be executed. (The usual single byte instruction utilized is RST.) If the interrupt instruction is two or three bytes long, the next one or two processor cycles, as indicated by the DBIN signal, should be used by the external device to supply the succeeding byte(s) of the interrupt instruction. Note that INTA status from the processor is not present during these operations.

If the interrupt instruction is not some form of CALL, it is executed normally by the processor except that the Program Counter is not incremented. The next instruction in the interrupted program is then fetched and executed. Notice that the interrupt mechanism must be re-enabled by the processor before another interrupt can occur.

If the interrupt instruction is some form of CALL, it is executed normally. The Program Counter is stored and control transferred to the interrupt service subroutine. This routine has responsibility for saving and restoring the machine state and for re-enabling interrupts if desired. When the interrupt service is complete, a RETURN instruction will transfer control back to the interrupted program.

INSTRUCTION SET SUMMARY

DATA TRANSFER

Op Code |7|6|5|4|3|2|1|0	No. of Bytes	Clock Cycles	Assembly Mnemonic	Instruction Description
01dddsss	1	5	MOVr, r	Move register to register
01110sss	1	7	MOVm, r	Move register to memory
01ddd110	1	7	MOVr, m	Move memory to register
00ddd110	2	7	MVI, r	Move to register, immediate
00110110	2	10	MVI, m	Move to memory, immediate
00111010	3	13	LDA	Load Acc, direct
00001010	1	7	LDAX B	Load Acc, indirect via B & C
00011010	1	7	LDAX D	Load Acc, indirect via D & E
00101010	3	16	LHLD	Load H & L, direct
00100001	3	10	LXI H	Load H & L, immediate
00010001	3	10	LXI D	Load D & E, immediate
00000001	3	10	LXI B	Load B & C, immediate
00110001	3	10	LXI SP	Load stack pointer, immediate
00100010	3	16	SHLD	Store H & L, direct
00110010	3	13	STA	Store Acc, direct
00000010	1	7	STAX B	Store Acc, indirect via B & C
00010010	1	7	STAX D	Store Acc, indirect via D & E
11111001	1	5	SPHL	Transfer H & L to stack pointer
11101011	1	4	XCHG	Exchange D & E with H & L
11100011	1	18	XTHL	Exchange top of stack with H & L
11011011	2	10	IN	Input to Acc
11010011	2	10	OUT	Output from Acc

CONTROL

01110110	1	7	HLT	Halt and enter wait state
00110111	1	4	STC	Set carry flag
00111111	1	4	CMC	Compliment carry flag
11111011	1	4	EI	Enable interrupts
11110011	1	4	DI	Disable interrupts
00000000	1	4	NOP	No operation

BRANCH

11000011	3	10	JMP	Jump unconditionally
11011010	3	10	JC	Jump on carry
11010010	3	10	JNC	Jump on no carry
11001010	3	10	JZ	Jump on zero
11000010	3	10	JNZ	Jump on not zero
11110010	3	10	JP	Jump on positive
11111010	3	10	JM	Jump on minus
11101010	3	10	JPE	Jump on parity even
11100010	3	10	JPO	Jump on parity odd
11001101	3	17	CALL	Call unconditionally
11011100	3	17-11	CC	Call on carry
11010100	3	17-11	CNC	Call on no carry
11001100	3	17-11	CZ	Call on zero
11000100	3	17-11	CNZ	Call on not zero
11110100	3	17-11	CP	Call on positive
11111100	3	17-11	CM	Call on minus
11101100	3	17-11	CPE	Call on parity even
11100100	3	17-11	CPO	Call on parity odd
11001001	1	10	RET	Return unconditionally
11011000	1	11-5	RC	Return on carry
11010000	1	11-5	RNC	Return on no carry
11001000	1	11-5	RZ	Return on zero
11000000	1	11-5	RNZ	Return on not zero
11110000	1	11-5	RP	Return on positive
11111000	1	11-5	RM	Return on minus
11101000	1	11-5	RPE	Return on parity even
11100000	1	11-5	RPO	Return on parity odd
11101001	1	5	PCHL	Jump unconditionally, indirect via H & L
11VVV111	1	11	RST	Restart

ARITHMETIC

10000sss	1	4	ADDr	Add register to Acc
10001sss	1	4	ADCr	Add with carry register to Acc
10000110	1	7	ADDm	Add memory to Acc
10001110	1	7	ADCm	Add with carry memory to Acc
11000110	2	7	ADI	Add to Acc, immediate
11001110	2	7	ACI	Add with carry to Acc, immediate
00001001	1	10	DAD B	Double add B & C to H & L
00011001	1	10	DAD D	Double add D & E to H & L
00101001	1	10	DAD H	Double add H & L to H & L
00111001	1	10	DAD SP	Double add stack pointer to H & L
10010sss	1	4	SUBr	Subtract register from Acc
10011sss	1	4	SBBr	Subtract with borrow register from Acc
10010110	1	7	SUBm	Subtract memory from Acc
10011110	1	7	SBBm	Subtract with borrow memory from Acc
11010110	2	7	SUI	Subtract from Acc, immediate
11011110	2	7	SBI	Subtract with borrow from Acc, immediate
00100111	1	4	DAA	Decimal adjust Acc

STACK OPERATIONS

11000101	1	11	PUSH B	Push registers B & C on stack
11010101	1	11	PUSH D	Push registers D & E on stack
11100101	1	11	PUSH H	Push registers H & L on stack
11110101	1	11	PUSH PSW	Push Acc and flags on stack
11000001	1	10	POP B	Pop registers B & C off stack
11010001	1	10	POP D	Pop registers D & E off stack
11100001	1	10	POP H	Pop registers H & L off stack
11110001	1	10	POP PSW	Pop Acc and flags off stack

LOGICAL

10100sss	1	4	ANA r	And register with Acc
10100110	1	7	ANA m	And memory with Acc
11100110	2	7	ANI	And with Acc, immediate
10101sss	1	4	XRA r	Exclusive or register with Acc
10101110	1	7	XRA m	Exclusive Or memory with Acc
11101110	2	7	XRI	Exclusive Or with Acc, immediate
10110sss	1	4	ORA r	Inclusive Or register with Acc
10110110	1	7	ORA m	Inclusive Or memory with Acc
11110110	2	7	ORI	Inclusive Or with Acc, immediate
10111sss	1	4	CMP r	Compare register with Acc
10111110	1	7	CMP m	Compare memory with Acc
11111110	2	7	CPI	Compare with Acc, immediate
00101111	1	4	CMA	Compliment Acc
00000111	1	4	RLC	Rotate Acc left
00001111	1	4	RRC	Rotate Acc right
00010111	1	4	RAL	Rotate Acc left through carry
00011111	1	4	RAR	Rotate Acc right through carry

INCREMENT/DECREMENT

00ddd100	1	5	INR r	Increment register
00110100	1	10	INR m	Increment memory
00000011	1	5	INX B	Increment extended B & C
00010011	1	5	INX D	Increment extended D & E
00100011	1	5	INX H	Increment extended H & L
00110011	1	5	INX SP	Increment stack pointer
00ddd101	1	5	DCR r	Decrement register
00110101	1	10	DCR m	Decrement memory
00001011	1	5	DCX B	Decrement extended B & C
00011011	1	5	DCX D	Decrement extended D & E
00101011	1	5	DCX H	Decrement extended H & L
00111011	1	5	DCX SP	Decrement stack pointer

MAXIMUM RATINGS (Above which useful life may be impaired)

Storage Temperature	−65°C to +150°C
Ambient Temperature Under Bias	−55°C to +125°C
All Signal Voltages With Respect to V_{BB}	−0.3V to +20V
All Supply Voltages With Respect to V_{BB}	−0.3V to +20V
Power Dissipation	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	Ambient Temperature	V_{DD}	V_{CC}	V_{BB}	V_{SS}
D8080A, P8080A D8080A-1, P8080A-1 D8080A-2, P8080A-2	0 to 70°C	12V ±5%	5.0V ±5%	−5.0V ±5%	0V
ID8080A ID8080A-1 ID8080A-2	−40 to 85°C	12V ±10%	5.0V ±10%	−5.0 ±10%	0V
MD8080AB, Am9080ADMB MD8080A-2B, Am9080A-2DMB	−55 to 125°C	12V ±10%	5.0V ±10%	−5.0V ±10%	0V

ELECTRICAL CHARACTERISTICS

over operating range (note 1)

Parameters	Description	Test Conditions		8080A, -X Min.	8080A, -X Typ.	8080A, -X Max.	Am9080A, -X Min.	Am9080A, -X Typ.	Am9080A, -X Max.	Am9080A, -XDMB MD8080A, -XB Min.	Am9080A, -XDMB MD8080A, -XB Typ.	Am9080A, -XDMB MD8080A, -XB Max.	Units
V_{IL}	Input LOW Voltage			−1.0		0.8	−1.0		0.8	−1.0		0.8	Volts
V_{IH}	Input HIGH Voltage			**3.3**		V_{CC}+1	**3.0**		V_{CC}+1	**3.0**		V_{CC}+1	Volts
V_{ILC}	Input LOW Voltage, Clock			−1.0		0.8	−1.0		0.8	−1.0		0.8	Volts
V_{IHC}	Input HIGH Voltage, Clock	A-1		9.0		V_{DD}+1	9.0		V_{DD}+1				Volts
		A-2		9.0		V_{DD}+1	9.0		V_{DD}+1	V_{DD}−2		V_{DD}+1	
		A		9.0		V_{DD}+1	9.0		V_{DD}+1	V_{DD}−2		V_{DD}+1	
V_{OL}	Output LOW Voltage	I_{OL} = **3.2mA**							**0.40**			**0.40**	Volts
		I_{OL} = **1.9mA**				**0.45**							
V_{OH}	Output HIGH Voltage	I_{OH} = −200μA					3.7			3.7			Volts
		I_{OH} = −150μA		3.7									
I_{DD}(AV)	V_{DD} Supply Current, Average	Operating, Minimum Clock Period	−55°C								50	80	mA
			0°C			70		40	70		45	75	
			25°C		40			35	65		40	70	
			70°C					30	55		35	60	
			125°C								30	50	
I_{CC}(AV)	V_{CC} Supply Current, Average	Operating, Minimum Clock Period	−55°C								45	60	mA
			0°C			80		35	50		40	55	
			25°C		60			30	45		35	50	
			70°C					25	40		30	45	
			125°C								25	40	
I_{BB}(AV)	V_{BB} Supply Current, Average	Operating, Minimum Clock Period				1.0			1.0			1.0	mA
I_{IL}	Input Leakage Current	(Note 4)				±10			±10			±10	μA
I_{CL}	Clock Leakage Current	$V_{SS} \leq V_{\phi} \leq V_{DD}$				±10			±10			±10	μA
I_{DL}	Data Bus Current,	$V_{IN} \leq V_{SS}$ + 0.8V				−100			−100			−100	μA
	Input Mode (Note 2)	$V_{IN} \geq V_{SS}$ + 0.8V				−2.0			−2.0			−2.0	mA
I_{FL}	Address and Data Bus	$V_{A/D} = V_{CC}$				10			10			10	μA
	Leakage in OFF State	$V_{A/D} = V_{SS}$				−100			−100			−100	μA

CAPACITANCE

f = 1.0MHz, Inputs = 0V, T_A = 25°C
$V_{DD} = V_{CC} = V_{SS} = 0V$, $V_{BB} = -5.0V$

Parameters	Description	Typ.	Max.	Units
C_{ϕ}	Clock Input Capacitance	12	25	pF
C_I	Input Capacitance	4.0	10	pF
C_O	Output Capacitance	8.0	20	pF
$C_{I/O}$	I/O Capacitance	10	20	pF

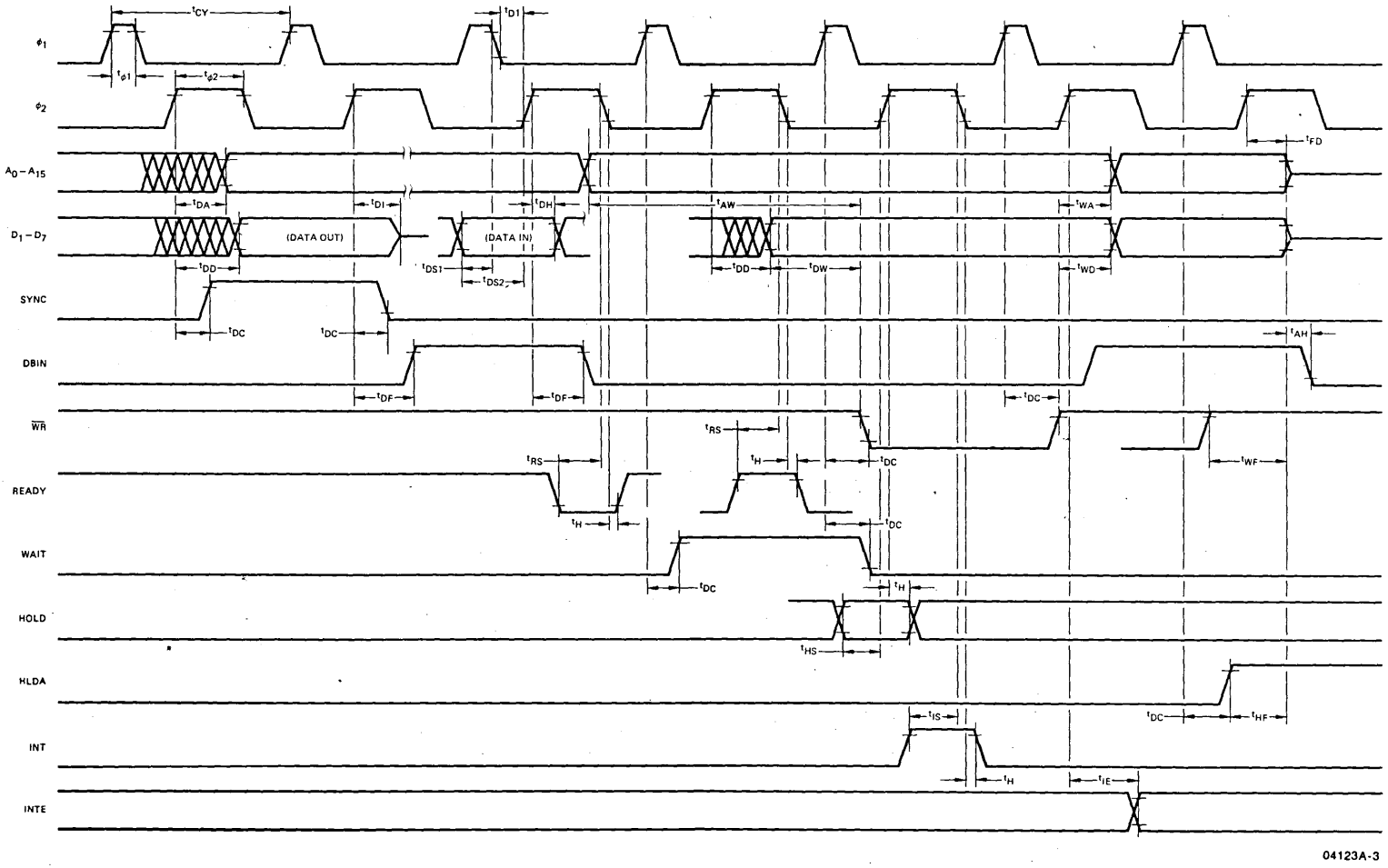
SWITCHING CHARACTERISTICS over operating range (Note 9)

Parameters	Description	Test Conditions	9080A-1, 8080A-1		9080A-2, 8080A-2		9080A, 8080A		Units
			Min	Max	Min	Max	Min	Max	
t_{DA}	Clock $\phi 2$ to Address Out Delay	Load Capacitance =100pF		150		175		200	ns
t_{DD}	Clock $\phi 2$ to Data Out Delay			180		200		220	ns
t_{DI}	Clock $\phi 2$ to Data Bus Input Mode Delay	(Note 5)		t_{DF}		t_{DF}		t_{DF}	ns
t_{DS1}	Data In to Clock $\phi 1$ Setup Time	Both t_{DS1} and t_{DS2} Must be Satisified	10		20		30		ns
t_{DS2}	Data In to Clock $\phi 2$ Setup Time		120		130		150		ns
t_{DC}	Clock to Control Output Delay	Load Capacitance = 50pF		110		120		120	ns
t_{RS}	Ready to Clock $\phi 2$ Setup Time		90		90		120		ns
t_H	Clock $\phi 2$ to Control Signal Hold Time		0		0		0		ns
t_{IS}	Interrupt to Clock $\phi 2$ Setup Time		100		100		120		ns
t_{HS}	Hold to Clock $\phi 2$ Setup Time		120		120		140		ns
t_{IE}	Clock $\phi 2$ to INTE Delay	Load Capacitance = 50pF		200		200		200	ns
t_{FD}	Clock $\phi 2$ to Address/Data OFF Delay			120		120		120	ns
t_{DF}	Clock $\phi 2$ to DBIN Delay	Load Capacitance = 50pF	25	130	25	140	25	140	ns
t_{DH}	Clock $\phi 2$ to Data In Hold Time	(Note 5)							ns
t_{AW}	Address Valid to Write Delay	(Note 8)							ns
t_{DW}	Output Data Valid to Write Delay								ns
t_{KA}	Address Valid to Write Increment			110		130		140	ns
t_{KD}	Output Data Valid to Write Increment			150		170		170	ns
t_{WA}	Write to Address Invalid Delay								ns
t_{WD}	Write to Output Data Invalid Delay								ns
t_{HF}	HLDA to Address/Data OFF Delay								ns
t_{WF}	Write to Address/Data OFF Delay								ns
t_{KH}	HLDA to Address/Data OFF Increment			50		50		50	ns
t_{AH}	DBIN to Address Hold Time		−20		−20		−20		ns

NOTES:

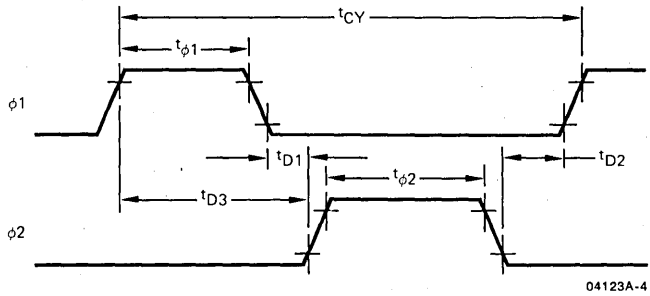
1. Typical values are at T_A = 25°C, nominal supply voltages and nominal processing parameters.
2. Pull-up devices are connected to the Data Bus lines when the input signal is high during DBIN time. When switching the input from HIGH-to-LOW a transient current must be absorbed by the driving device until the input reaches a LOW level.
3. Timing reference levels –
   Clocks: HIGH = 9.0V, LOW = 1.0V
   Inputs: HIGH = 3.3V, LOW = 0.8V
   Outputs: HIGH = 2.0V, LOW = 0.8V
4. Control inputs impress currents on the driving signal during HIGH-to-LOW transitions. Values shown are for logic high or logic low levels. Peak current during transition is as much as 2.0mA.
5. Bus contention cannot occur and data hold times are adequate when DBIN is used to enable Data In. t_{DH} is the smaller of 50ns or t_{DF}.
6. RESET should remain active for at least three clock periods.
7. With interrupts enabled, the interrupted instruction will be one with an interrupt input stable during the indicated interval of the last clock period of the preceding instruction. Additional synchronization not necessary.
8. $t_{AW} = 2\,t_{CY} - t_{D3} - t_r - t_{KA}$
   $t_{DW} = t_{CY} - t_{D3} - t_r - t_{KD}$
   For HLDA Off: $t_{WD} = t_{WA} = t_{D3} + t_r + 10ns$
   For HLDA On: $t_{WD} = t_{WA} = t_{WF}$
   $t_{HF} = t_{D3} + t_r - t_{KH}$
   $t_{WF} = t_{D3} + t_r - 10ns$
   $t_r = \phi 2$ rise time
9. The switching specifications listed for the 9080A, 9080A-2, 9080A-1 meet or exceed the corresponding specifications for the 8080A, 8080A-2, 8080A-1.

SWITCHING WAVEFORMS SUMMARY



This chart presents relative timing waveform relationships and does not show actual processor operating cycles.

CLOCK WAVEFORM DETAIL

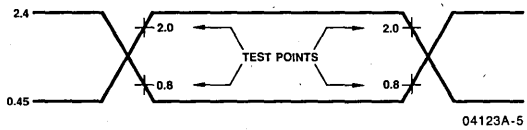


$t_{CY} = t_{D3} + t_{r\phi2} + t_{\phi2} + t_{f\phi2} + t_{D2} + t_{r\phi1}$

CLOCK SWITCHING CHARACTERISTICS over operating range

Parameters	Description	9080A-1, 8080A-1 Min	9080A-1, 8080A-1 Max	9080A-2, 8080A-2 Min	9080A-2, 8080A-2 Max	9080A, 8080A Min	9080A, 8080A Max	Units
t_{CY}	Clock Period	320	2000	380	2000	480	2000	ns
t_r, t_f	Clock Transition Times	0	25	0	50	0	50	ns
$t_{\phi1}$	Clock $\phi1$ Pulse Width	50		60		60		ns
$t_{\phi2}$	Clock $\phi2$ Pulse Width	145		175		220		ns
t_{D1}	$\phi1$ to $\phi2$ Offset	0		0		0		ns
t_{D2}	$\phi2$ to $\phi1$ Offset	60		70		70		ns
t_{D3}	$\phi1$ to $\phi2$ Delay	60		70		80		ns

Input, Output Waveforms for A.C. Tests



8085A/AH

Single Chip 8-Bit N-Channel Microprocessor

DISTINCTIVE CHARACTERISTICS

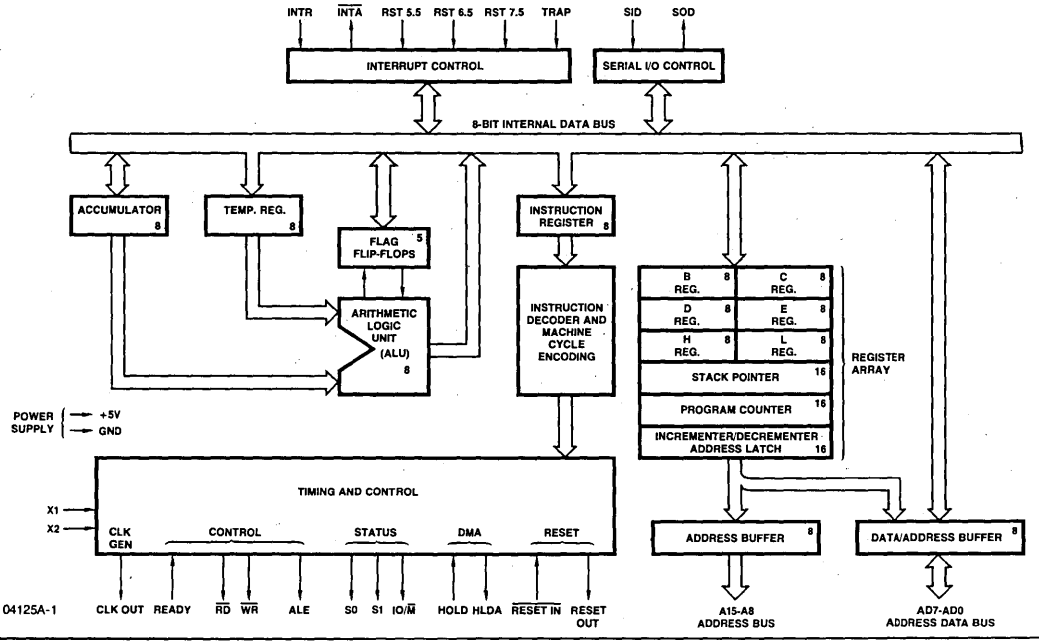
- Complete 8-bit parallel CPU
- On-chip system controller; advanced cycle status information available for large system control
- Four vectored interrupts (one is non-maskable)
- On-chip clock generator (with external crystal, LC or R/C network)
- Serial in/serial out port
- Decimal, binary and double precision arithmetic
- Direct addressing capability to 64K bytes of memory
- 1.3μs instruction cycle (8085A)
- 0.8μs instruction cycle (8085A-2)
- 100% software compatible with 8080A
- Single +5V power supply

GENERAL DESCRIPTION

The 8085A is a new generation, complete 8-bit parallel central processing unit (CPU). Its instruction set is 100% software compatible with the 8080A microprocessor. Specifically, the 9080A incorporates all of the features that the 8224 (clock generator) and 8228 (system controller) provided for the 9080A. The 8085A-2 is a faster version of the 8085A. The 8085AH is a 3MHz CPU with 10% supply tolerances and lower power consumption.

The 8085A uses a multiplexed Data Bus. The address is split between the 8-bit address bus and the 8-bit data bus. The on-chip address latches of 8155/8355 memory products allow a direct interface with 8085A. The 8085A components, including various timing compatible support chips, allow system speed optimization.

BLOCK DIAGRAM

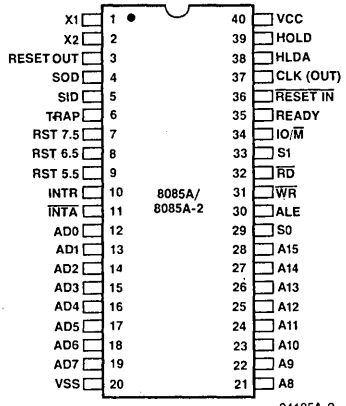


ORDERING INFORMATION

Package Type	Temperature Range	Maximum Clock Frequency	
		3MHz	5MHz
Molded DIP	$0°C \leq T_A \leq +70°C$	P8085A	P8085A-2
		P8085AH	
Hermetic DIP		D8085A	D8085A-2
		D8085AH	
Hermetic DIP	$-40°C \leq T_A \leq +85°C$	ID8085A	
Hermetic DIP	$-55°C \leq T_A \leq +125°C$	MD8085AB	

04125A-MMP

CONNECTION DIAGRAM
Top View
D-40, P-40



Note: Pin 1 is marked for orientation.

Figure 1.

8085A FUNCTIONAL PIN DEFINITION

The following describes the function of each pin:

A8-A15 (Output 3-State)

Address Bus – the most significant eight bits of the memory address or the eight bits of the I/O address, 3-stated during Hold and Halt modes and during RESET.

AD0-AD7 (Input/Output 3-State)

Multiplexed Address/Data Bus – lower eight bits of the memory address (or I/O address) appear on the bus during the first clock cycle of a machine cycle. It then becomes the data bus during the second and third clock cycles.

Three-stated during Hold and Halt modes.

ALE (Output)

Address Latch Enable – it occurs during the first clock cycle of a machine cycle and enables the address to get latched into the on-chip latch of peripherals. The falling edge of ALE is set to guarantee setup and hold times for the address information. The falling edge ALE can also be used to strobe the status information. ALE in never 3-stated.

S0, S1 (Output)

Data Bus Status. Encoded status of the bus cycle.

S1	S0	
0	0	HALT
0	1	WRITE
1	0	READ
1	1	FETCH

S1 can be used as an advanced R/$\overline{W}$ status.

$\overline{RD}$ (Output 3-State)

READ – A low level on $\overline{RD}$ indicates the selected memory or I/O device is to be read and that the Data Bus is available for the data transfer. Three-stated during Hold and Halt and during RESET.

$\overline{WR}$ (Output 3-State)

WRITE – A low level on $\overline{WR}$ indicates the data on the Data Bus is to be written into the selected memory or I/O location. Data is set up at the trailing edge of $\overline{WR}$. Three-stated during Hold and Halt modes.

READY (Input)

If READY is high during a read or write cycle, it indicates that the memory or peripheral is ready to send or receive data. If READY is low, the CPU will wait an integral number of clock cycles for READY to go high before completing the read or write cycle.

HOLD (Input)

HOLD – indicates that another Master is requesting the use of the Address and Data Buses. The CPU, upon receiving the Hold request, will relinquish the use of buses as soon as the completion of the current machine cycle. Internal processing can continue. The processor can regain the buses only after the Hold is removed. When the Hold is acknowledged, the Address, Data, $\overline{RD}$, $\overline{WR}$ and IO/$\overline{M}$ lines are three-stated.

HLDA (Output)

HOLD ACKNOWLEDGE – indicates that the CPU has received the Hold request and that it will relinquish the buses in the next clock cycle. HLDA goes low after the Hold request is removed. The CPU takes the buses one half clock cycle after HLDA goes LOW.

INTR (Input)

INTERRUPT REQUEST – is used as a general purpose interrupt. It is sampled only during the next to the last clock cycle of the instruction. If it is active, the Program Counter (PC) will be inhibited from incrementing and an $\overline{INTA}$ will be issued. During this cycle a RESTART or CALL instruction can be inserted to jump to the interrupt service routine. The INTR is enabled and disabled by software. It is disabled by Reset and immediately after an interrupt is accepted.

$\overline{INTA}$ (Output)

INTERRUPT ACKNOWLEDGE – is used instead of (and has the same timing as) $\overline{RD}$ during the Instruction cycle after an INTR is accepted. It can be used to activate the Am9519 Interrupt chip or some other interrupt port.

RST 5.5
RST 6.5
RST 7.5
} (Inputs)

RESTART INTERRUPTS – these three inputs have the same timing as INTR except they cause an internal RESTART to be automatically inserted.

RST 7.5 ——→ Highest Priority
RST 6.5
RST 5.5 ——→ Lowest Priority

The priority of these interrupts is ordered as shown above. These interrupts have a higher priority than the INTR. However they may be individually masked out using the SIM instructions.

TRAP (Input)

Trap interrupt is a non-maskable restart interrupt. It is recognized at the same time as INTR. It is unaffected by any mask or Interrupt Enable. It has the highest priority of any interrupt.

$\overline{\text{RESET IN}}$ (Input)

Reset sets the Program Counter to zero and resets the Interrupt Enable and HLDA flip-flops. None of the other flags or registers (except the instruction register) are affected. The CPU is held in the reset condition as long as RESET is applied.

RESET OUT (Output)

Indicates CPU is being reset. Can be used as a system RESET. The signal is synchronized to the processor clock.

X1, X2 (Input)

Crystal, LC or R/C network connections to set the internal clock generator. X1 can also be an external clock input instead of a crystal. The input frequency is divided by 2 to give the internal operating frequency.

CLK (Output)

Clock Output for use as a system clock when a crystal or R/C network is used as an input to the CPU. The period of CLK is twice the X1, X2 input period.

IO/$\overline{\text{M}}$ (Output)

IO/$\overline{\text{M}}$ indicates whether the Read/Write is to memory or I/O. 3-stated during Hold and Halt modes.

SID (Input)

Serial input data line. The data on this line is loaded into accumulator bit 7 whenever a RIM instruction is executed.

SOD (Output)

Serial output data line. The output SOD is set or reset as specified by the SIM instruction.

VCC

+5 volt supply.

VSS

Ground reference.

FUNCTIONAL DESCRIPTION

The 8085A is a complete 8-bit parallel central processor. It is designed with N-channel depletion loads and requires a single +5 volt supply. Its basic clock speed is 3MHz (5MHz: 8085A-2) thus improving on the present 9080's performance with higher system speed. Also it is designed to fit into a minimum system of three ICs: The CPU, a RAM/IO, and a ROM or PROM/IO chip.

The 8085A uses a multiplexed Data Bus. The address is split between the higher 8-bit Address Bus and the lower 8-bit Address/Data Bus. During the first cycle the address is sent out. The lower eight bits are latched into the peripherals by the Address Latch Enable (ALE). During the rest of the machine cycle the Data Bus is used for memory or I/O data.

The 8085A provides $\overline{\text{RD}}$, $\overline{\text{WR}}$ and IO/$\overline{\text{Memory}}$ signals for bus control. An Interrupt Acknowledge signal ($\overline{\text{INTA}}$) is also provided. Hold, Ready and all Interrupts are synchronized. The 8085A also provides serial input data (SID) and serial output data (SOD) lines for simple serial interface.

In addition to these features, the 8085A has three maskable, restart interrupts and one non-maskable trap interrupt.

8085A vs. 8080A

The 8085A includes the following features on-chip in addition to all of the 9080A functions.

a. Internal clock generator
b. Clock output
c. Fully synchronized Ready
d. Schmitt action on $\overline{\text{RESET IN}}$
e. RESET OUT pin
f. $\overline{\text{RD}}$, $\overline{\text{WR}}$ and IO/$\overline{\text{M}}$ Bus Control Signals
g. Encoded Status information
h. Multiplexed Address and Data
i. Direct Restarts and non-maskable Interrupt
j. Serial Input/Output lines

The internal clock generator requires an external crystal or R/C network. It will oscillate at twice the basic CPU operating frequency. A 50% duty cycle, two phase, non-overlapping clock is generated from this oscillator internally and one phase of the clock (ϕ2) is available as an external clock. The 8085A directly provides the external RDY synchronization previously provided by the 8224. The $\overline{\text{RESET IN}}$ input is provided with a Schmitt action input so that power-on reset only requires a resistor and capacitor. RESET OUT is provided for System RESET.

The 8085A provides $\overline{\text{RD}}$, $\overline{\text{WR}}$ and IO/$\overline{\text{M}}$ signals for Bus control. An $\overline{\text{INTA}}$ which was previously provided by the 8228 in 9080A systems is also included in 8085A.

STATUS INFORMATION

Status information is directly available from the 8085A. ALE serves as a status strobe. The status is partially encoded and provides the user with advanced timing of the type of bus transfer being done. IO/$\overline{\text{M}}$ cycle status signal is provided directly also. Decoded S0, S1 carries the following status information:

MACHINE CYCLE STATUS

IO/$\overline{\text{M}}$	S1	S0	Status
0	0	1	Memory write
0	1	0	Memory read
1	0	1	I/O write
1	1	0	I/O read
0	1	1	Opcode fetch
1	1	1	Interrupt Acknowledge
•	0	0	Halt
•	X	X	Hold
•	X	X	Reset

• = 3-state (high impedance)
X = unspecified

S1 can be interpreted as R/$\overline{\text{W}}$ in all bus transfers.

In the 8085A the eight LSB of address are multiplexed with the data instead of status. The ALE line is used as a strobe to enter the lower half of the address into the memory or peripheral address latch. This also frees extra pins for expanded interrupt capability.

INTERRUPT AND SERIAL I/O

The 8085A/8085A-2 has 5 interrupt inputs: INTR, RST 5.5, RST 6.5, RST 7.5 and TRAP. INTR is identical in function to the 8080A INT. Each of three RESTART inputs, 5.5, 6.5, 7.5, has programmable mask. TRAP is also a RESTART interrupt except it is non-maskable.

The three RESTART interrupts cause the internal execution of RST (saving the program counter in the stack and branching to the RESTART address) if the interrupts are enabled and if the interrupt mask is not set. The nonmaskable TRAP causes the internal execution of a RST independent of the state of the interrupt enable or masks.

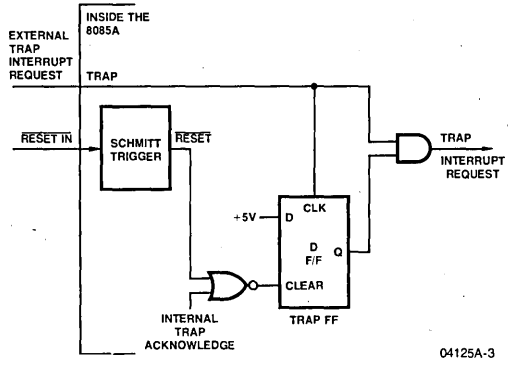
Name	RESTART Address (Hex)
TRAP	24_{16}
RST 5.5	$2C_{16}$
RST 6.5	34_{16}
RST 7.5	$3C_{16}$

There are two different types of inputs in the restart interrupts. RST 5.5 and RST 6.5 are high level-sensitive like INTR (and INT on the 8080A) and are recognized with the same timing as INTR. RST 7.5 is rising edge-sensitive. For RST 7.5, only a pulse is required to set an internal flip-flop which generates the internal interrupt request. The RST 7.5 request flip-flop remains set until the request is serviced. Then it is reset automatically. This flip-flop may also be reset by using the SIM instruction or by issuing a RESET IN to the 8085A. The RST 7.5 internal flip-flop will be set by a pulse on the RST 7.5 pin even when the RST 7.5 interrupt is masked out.

The status of the three RST interrupt masks can only be affected by the SIM instruction and RESET IN.

The interrupts are arranged in a fixed priority that determines which interrupt is to be recognized if more than one is pending as follows: TRAP – highest priority, RST 7.5, RST 6.5, RST 5.5, INTR – lowest priority. This priority scheme does not take into account the priority of a routine that was started by a higher priority interrupt. RST 5.5 can interrupt a RST 7.5 routine if the interrupts were re-enabled before the end of the RST 7.5 routine.

The TRAP interrupt is useful for catastrophic errors such as power failure or bus error. The TRAP input is recognized just as any other interrupt but has the highest priority. It is not affected by any flag or mask. The TRAP input is both edge and level sensitive. The TRAP input must go high and remain high to be acknowledged, but will not be recognized again until it goes low, then high again. This avoids any false triggering due to noise or logic glitches. The following diagram illustrates the TRAP interrupt request circuitry within the 8085A.



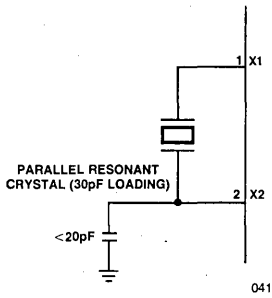
Note that the servicing of any interrupt (TRAP, RST 7.5, RST 6.5, RST 5.5, INTR) disables all future interrupts (except TRAPs) until an EI instruction is executed.

The TRAP interrupt is special in that it preserves the previous interrupt enable status. Performing the first RIM instruction following a TRAP interrupt allows you to determine whether interrupts were enabled or disabled prior to the TRAP. All subsequent RIM instructions provide current interrupt enable status.

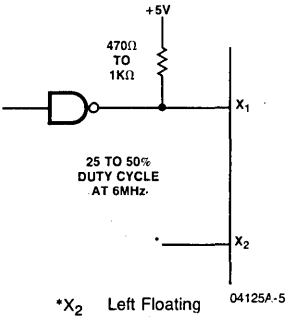
The serial I/O system is also controlled by the RIM and SIM instructions. SID is read by RIM, and SIM sets the SOD data.

DRIVING THE X1 AND X2 INPUTS

The user may drive the X1 and X2 inputs of the 8085A or 8085A-2 with a crystal, an external clock source or an R/C network as shown below. The driving frequency must be twice the desired internal operating frequency (the 8085A would require a 6MHz crystal for 3MHz internal operation).

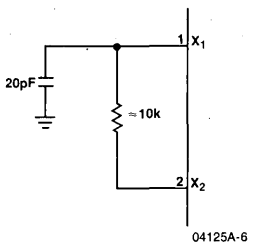


1-6 MHz Input Frequency

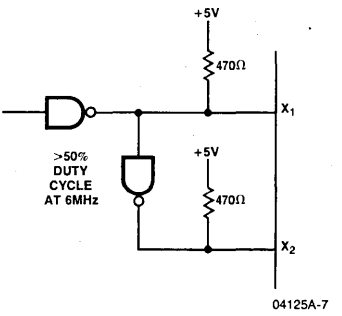


1-6 MHz Input Frequency

The 20pF capacitor is required to guarantee oscillation at the proper frequency during system startup. Capacitance from X_2 to Ground should not exceed 20pF.



≈3 MHz Input Frequency



≈6 MHz Input Frequency

RC Mode causes a large drift in clock frequency because of the variation in on-chip timing generation parameters. Use of RC Mode should be limited to an application which can tolerate a wide frequency variation.

Note: Duty cycle refers to the percentage of the clock input cycle when X_1 is high.

Figure 2. Driving the Clock Inputs (X1 and X2) of 8085A.

GENERATING 8085A WAIT STATE

The following circuit may be used to insert one WAIT state in each 8085A machine cycle.

The D flip-flops should be chosen such that

- CLK is rising edge triggered
- CLEAR is low-level active.

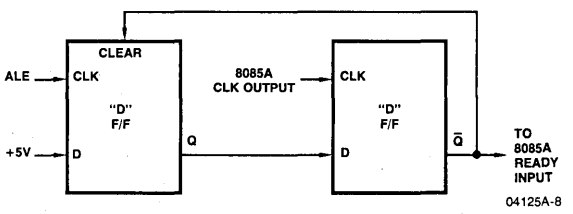


Figure 3. Generation of a Wait State for 8085A CPU.

BASIC SYSTEM TIMING

The 8085A has a multiplexed Data Bus. ALE is used as a strobe to sample the lower 8 bits of address on the Data Bus. Figure 2 shows an instruction fetch, memory read and I/O write cycle (OUT). Note that during the I/O write and read cycle that the I/O port address is copied on both the upper and lower half of the address.

As in the 9080A, the READY line is used to extend the read and write pulse lengths so that the 8085A can be used with slow memory. Hold causes the CPU to relinquish the bus when it is through with it by floating the Address and Data Buses.

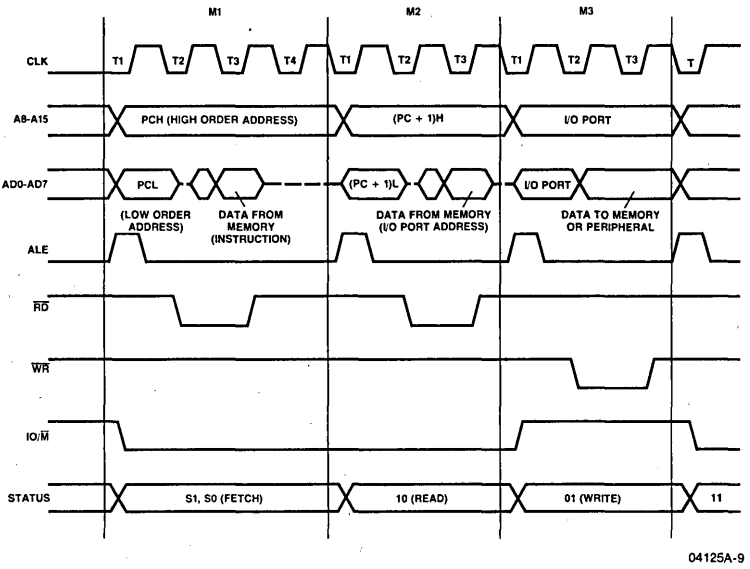


Figure 4. 8085A Basic System Timing.

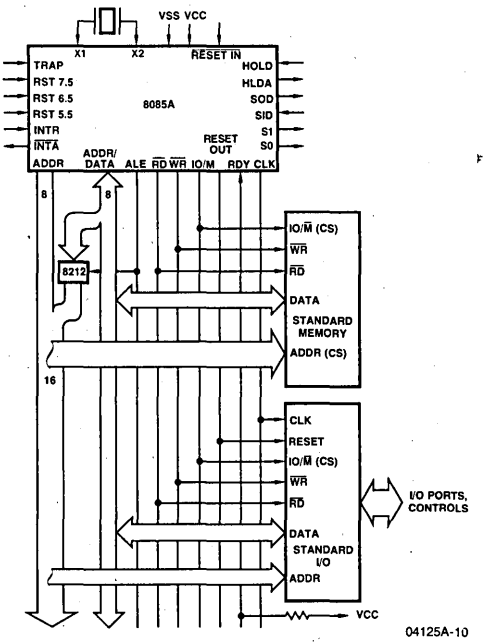


Figure 5. System Using Standard Memories.

MAXIMUM RATINGS above which useful life may be impaired

Storage Temperature	−65 to +150°C
Ambient Temperature Under Bias	−55 to +125°C
V_{CC} with Respect to V_{SS}	−0.5 to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5 to +7.0V
Power Dissipation	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	Ambient Temperature	V_{CC}	V_{SS}
D8085A, P8085A D8085A-2, P8085A-2	$0°C \leq T_A \leq +70°C$	5V ± 5%	0V
D8085AH, P8085AH		5V ± 10%	0V
ID8085A	$-40°C \leq T_A \leq +85°C$	5V ± 10%	0V
MD8085AB	$-55°C \leq T_A \leq +125°C$	5V ± 10%	0V

DC CHARACTERISTICS

				8085A, 8085AH 8085A-2		ID8085A, MD8085AB		
Parameters	Description	Test Conditions		Min	Max	Min	Max	Units
VIL	Input Low Voltage			−0.5	+0.8	−0.5	+0.8	Volts
VIH	Input High Voltage			2.0	VCC+0.5	2.2	VCC+0.5	Volts
VOL	Output Low Voltage	IOL = 2.0mA			0.45		0.45*	Volts
VOH	Output High Voltage	IOH = −400μA		2.4		2.4		Volts
ICC	Power Supply Current	VIN = VCC	8085A, 8085A-2		170		200	mA
			8085AH		135**		200	mA
IIL†	Input Leakage	VIN = VCC			±10		±10	μA
ILO	Output Leakage	0.45V ≤ VOUT ≤ VCC			±10		±10	μA
VILR	Input Low Level, RESET			−0.5	+0.8	−0.5	+0.8	Volts
VIHR	Input High Level, RESET			2.4	VCC+0.5	2.4	VCC+0.5	Volts
VHY	Hysteresis, RESET			0.25		0.25		Volts

*IOL = 1.6mA
**Typical value at 25°C is 150mA max.
†Except Pin 1 and Pin 2.

AC CHARACTERISTICS

Parameters	Description		8085A		8085A-2		ID8085A, MD8085AB		Units
			Min	Max	Min	Max	Min	Max	
tCYC	CLK Cycle Period		320	2000	200	2000	320	2000	ns
tr, tf	CLK Rise and Fall Time			30		30		30	ns
tAL	A8-A15 Valid before Trailing Edge of ALE (Note 1)		115		50		115		ns
tACL	A0-A7 Valid to Leading Edge of Control		240		115		240		ns
tXKR	X1 Rising to CLK Rising		25	120	30	100	30	120	ns
tXKF	X1 Rising to CLK Falling		30	150	30	110	30	150	ns
t1	CLK Low Time	Standard 150pF Loading	80		40		80		ns
		Lightly Loaded (Note 8)	100				100		
t2	CLK High Time	Standard 150pF Loading	120		70		120		ns
		Lightly Loaded (Note 8)	150				150		
tALL	A0-A7 Valid to Leading Edge of Control		90		50		90		ns
tLRY	ALE to READY Stable			110		30		110	ns
tLA	Address Hold Time after ALE		100		50		100		ns
tLL	ALE Width		140		80		140		ns
tLCK	ALE Low During CLK High		100		50		100		ns
tLC	Trailing Edge of ALE to Leading Edge of Control		130		60		130		ns
tAFR	Address Float after Leading Edge of READ (INTA)			0		0		0	ns
tAD	Valid Address to Valid Data In			575		350		575	ns
tRD	$\overline{\text{READ}}$ (or $\overline{\text{INTA}}$) to Valid Data			300		150		300	ns
tRDH	Data Hold Time after $\overline{\text{READ}}$ ($\overline{\text{INTA}}$) (Note 7)		0		0		0		ns
tRAE	Trailing Edge of $\overline{\text{READ}}$ to Re-Enabling of Address		150		90		150		ns
tCA	Address (A8-A15) Valid after Control		120		60		120		ns
tDW	Data Valid to Trailing Edge of $\overline{\text{WRITE}}$		420		230		420		ns
tWD	Data Valid after Trailing Edge of $\overline{\text{WRITE}}$		100		60		100		ns
tCC	Width of Control Low ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{INTA}}$)		400		230		400		ns
tCL	Trailing Edge of Control to Leading Edge of ALE		50		25		50		ns
tARY	READY Valid from Address Valid			220		100		220	ns
tRYS	READY Setup Time to Leading Edge of CLK		110		100		110		ns
tRYH	READY Hold Time		0		0		0		ns
tHACK	HLDA Valid to Trailing Edge of CLK		110		40		110		ns
tHABF	Bus Float after HLDA			210		150		210	ns
tHABE	HLDA to Bus Enable			210		150		210	ns
tLDR	ALE to Valid Data In			460		270		460	ns
tRV	Control Trailing Edge to Leading Edge of Next Control		400		220		400		ns
tAC	A8-A15 Valid to Leading Edge of Control (Note 1)		270		115		270		ns
tHDS	HOLD Setup Time to Trailing Edge of CLK		170		120		170		ns
tHDH	HOLD Hold Time		0		0		0		ns
tINS	INTR Setup Time to Falling Edge of CLK, also RST and TRAP		160		150		160		ns
tINH	INTR Hold Time		0		0		0		ns
tLDW	ALE to Valid Data during $\overline{\text{WRITE}}$			200		120		200	ns
tWDL	Leading Edge of $\overline{\text{WRITE}}$ to Data Valid			40		20		40	ns

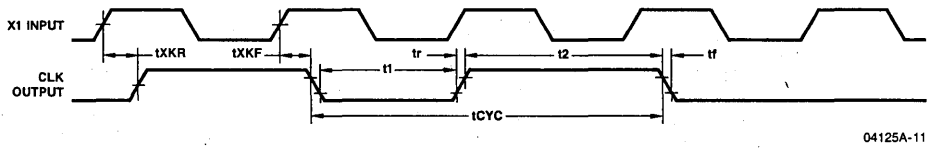
Notes:
1. A8-A15 Address Specs apply to IO/$\overline{\text{M}}$, S0 and S1. Except A8-A15 are undefined during T4-T6 of cycle whereas IO/$\overline{\text{M}}$, S0 and S1 are stable.
2. Test Conditions: tCYC = 320ns (Am8085A)/200ns (Am8085A-2); CL = 150pF
3. For all output timing where CL = 150pF use the following correction factors.
   25pF ≤ CL < 150pF: −.10ns/pF
   150pF < CL ≤ 300pF: +.30ns/pF
4. Output timings are measured with purely capacitive load.
5. All timings are measured at output voltage VL = 0.8V, VH = 2.0V and 1.5V with 20ns rise and fall time on inputs.
6. To calculate timing specifications at other values of tCYC use the table in Table 2.
7. Data Hold Time is guaranteed under all loading conditions.
8. Loading equivalent to 50pF +1 TTL input.

BUS TIMING SPECIFICATION AS A TCYC DEPENDENT

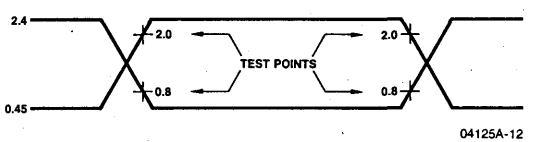
Parameters	Description	8085AH, ID8085A, 8085A, MD8085AB Min	8085AH, ID8085A, 8085A, MD8085AB Max	8085A-2 Min	8085A-2 Max	Units
tAL	Address Valid before Trailing Edge of ALE	(1/2)T−45		(1/2)T−50		ns
tLA	Address Hold Time after ALE	(1/2)T−60		(1/2)T−50		ns
tLL	ALE Width	(1/2)T−20		(1/2)T−20		ns
tLCK	ALE Low During CLK High	(1/2)T−60		(1/2)T−50		ns
tLC	Trailing Edge of ALE to Leading Edge of Control	(1/2)T−30		(1/2)T−40		ns
tAD	Valid Address to Valid Data In		(5/2+N)T−225		(5/2+N)T−150	ns
tRD	READ (or INTA) to Valid Data		(3/2+N)T−180		(3/2+N)T−150	ns
tRAE	Trailing Edge of READ to Re-Enabling of Address	(1/2)T−10		(1/2)T−10		ns
tCA	Address (A8-A15) Valid after Control	(1/2)T−40		(1/2)T−40		ns
tDW	Data Valid to Trailing Edge of WRITE	(3/2+N)T−60		(3/2+N)T−70		ns
tWD	Data Valid after Trailing Edge of WRITE	(1/2)T−60		(1/2)T−40		ns
tWDL	Leading Edge of WRITE to Data Valid		40		40	ns
tCC	Width of Control LOW (RD, WR, INTA)	(3/2+N)T−80		(3/2+N)T−70		ns
tCL	Trailing Edge of Control to Leading Edge of ALE	(1/2)T−110		(1/2)T−75		ns
tARY	READY Valid from Address Valid		(3/2)T−260		(3/2)T−200	ns
tHACK	HLDA Valid to Trailing Edge of CLK	(1/2)T−50		(1/2)T−60		ns
tHABF	Bus Float after HLDA		(1/2)T+50		(1/2)T+50	ns
tHABE	HLDA to Bus Enable		(1/2)T+50		(1/2)T+50	ns
tAC	Address Valid to Leading Edge of Control	(2/2)T−50		(2/2)T−85		ns
t1	CLK Low Time	(1/2)T−80		(1/2)T−60		ns
t2	CLK High Time	(1/2)T−40		(1/2)T−30		ns
tRV	Control Trailing Edge to Leading Edge of Next Control	(3/2)T−80		(3/2)T−80		ns
tLDR			(4/2)T−180		(4/2)T−130	ns
tLDW	Trailing Edge of ALE to Valid Data During WRITE		200		200	ns

Note: N is equal to the total WAIT states.
T = tCYC.

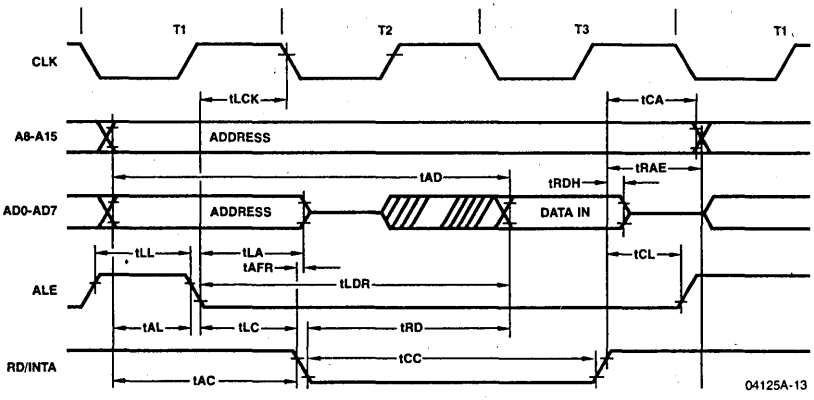
CLOCK TIMING WAVEFORM



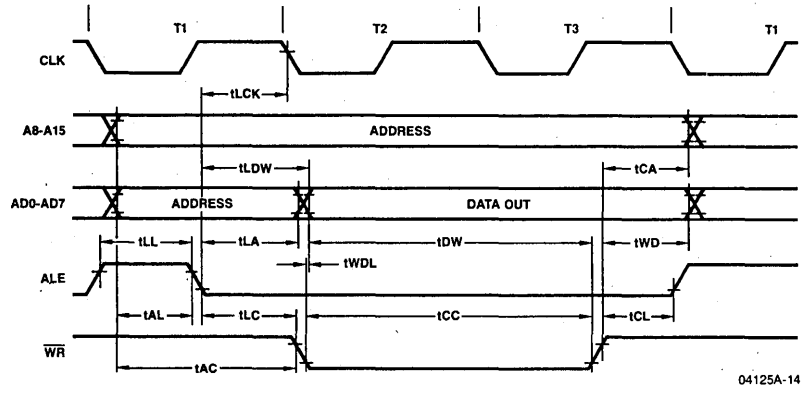
INPUT, OUTPUT WAVEFORMS FOR AC TESTS.



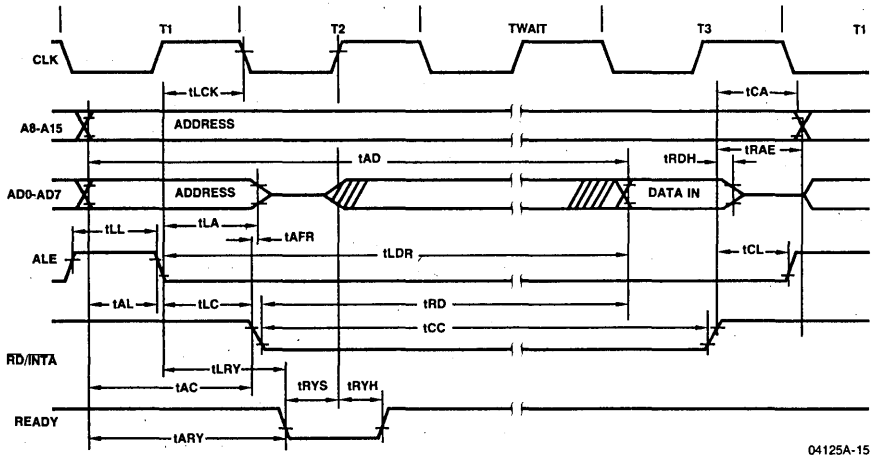
READ OPERATION



WRITE OPERATION



TYPICAL READ OPERATION WITH WAIT CYCLE



Same READY timing applies to WRITE operation.

Figure 6. 8085A/8085A-2 Bus Timing

HOLD OPERATION

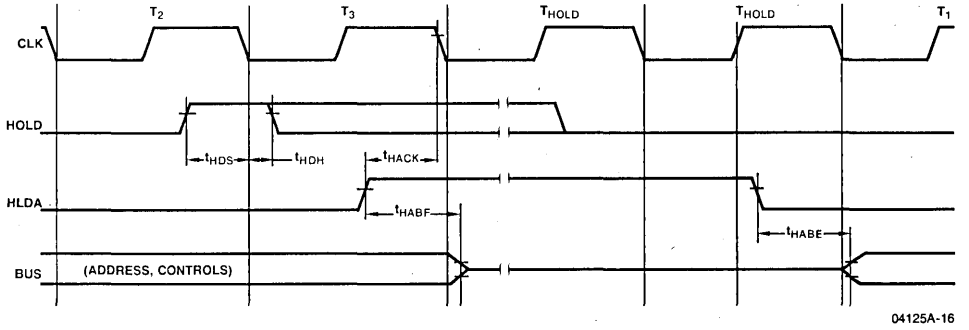


Figure 7. 8085A Hold Timing.

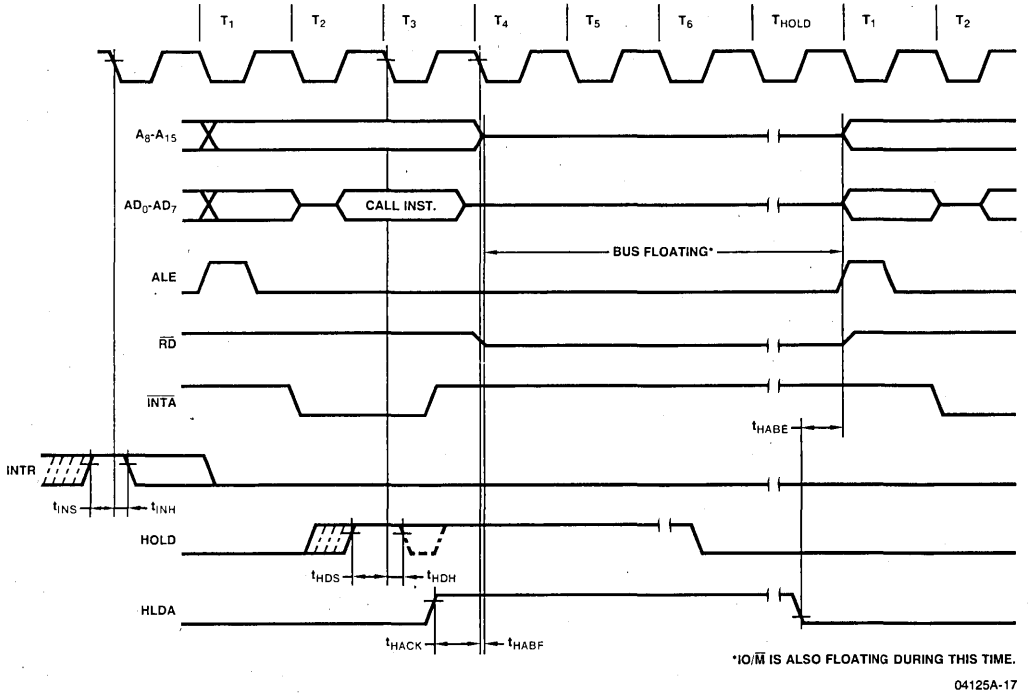


Figure 8. 8085A Interrupt and Hold Timing.

INSTRUCTION SET SUMMARY

Mnemonic*	Description	Instruction Code (Note 1) D7	D6	D5	D4	D3	D2	D1	D0	Clock Cycles (Note 2)
MOVE, LOAD AND STORE										
MOVr1r2	Move register to register	0	1	D	D	D	S	S	S	4
MOV Mr	Move register to memory	0	1	1	1	0	S	S	S	7
MOV rM	Move memory to register	0	1	D	D	D	1	1	0	7
MVI r	Move immediate register	0	0	D	D	D	1	1	0	7
MVI M	Move immediate memory	0	0	1	1	0	1	1	0	10
LXI B	Load immediate register Pair B & C	0	0	0	0	0	0	0	1	10
LXI D	Load immediate register Pair D & E	0	0	0	1	0	0	0	1	10
LXI H	Load immediate register Pair H & L	0	0	1	0	0	0	0	1	10
LXI SP	Load immediate stack pointer	0	0	1	1	0	0	0	1	10
STAX B	Store A indirect	0	0	0	0	0	0	1	0	7
STAX D	Store A indirect	0	0	0	1	0	0	1	0	7
LDAX B	Load A indirect	0	0	0	0	1	0	1	0	7
LDAX D	Load A indirect	0	0	0	1	1	0	1	0	7
STA	Store A direct	0	0	1	1	0	0	1	0	13
LDA	Load A direct	0	0	1	1	1	0	1	0	13
SHLD	Store H & L direct	0	0	1	0	0	0	1	0	16
LHLD	Load H & L direct	0	0	1	0	1	0	1	0	16
XCHG	Exchange D & E, H & L Registers	1	1	1	0	1	0	1	1	4
STACK OPS										
PUSH B	Push register Pair B & C on stack	1	1	0	0	0	1	0	1	12
PUSH D	Push register Pair D & E on stack	1	1	0	1	0	1	0	1	12
PUSH H	Push register Pair H & L on stack	1	1	1	0	0	1	0	1	12
PUSH PSW	Push A and Flags on stack	1	1	1	1	0	1	0	1	12
POP B	Pop register Pair B & C off stack	1	1	0	0	0	0	0	1	10
POP D	Pop register Pair D & E off stack	1	1	0	1	0	0	0	1	10
POP H	Pop register Pair H & L off stack	1	1	1	0	0	0	0	1	10
POP PSW	Pop A and Flags off stack	1	1	1	1	0	0	0	1	10
XTHL	Exchange top of stack H & L	1	1	1	0	0	0	1	1	16
SPHL	H & L to stack pointer	1	1	1	1	1	0	0	1	6
JUMP										
JMP	Jump unconditional	1	1	0	0	0	0	1	1	10
JC	Jump on carry	1	1	0	1	1	0	1	0	7/10
JNC	Jump on no carry	1	1	0	1	0	0	1	0	7/10
JZ	Jump on zero	1	1	0	0	1	0	1	0	7/10
JNZ	Jump on no zero	1	1	0	0	0	0	1	0	7/10
JP	Jump on positive	1	1	1	1	0	0	1	0	7/10
JM	Jump on minus	1	1	1	1	1	0	1	0	7/10
JPE	Jump on parity even	1	1	1	0	1	0	1	0	7/10
JPO	Jump on parity odd	1	1	1	0	0	0	1	0	7/10
PCHL	H & L to program counter	1	1	1	0	1	0	0	1	6
CALL										
CALL	Call unconditional	1	1	0	0	1	1	0	1	18
CC	Call on carry	1	1	0	1	1	1	0	0	9/18
CNC	Call on no carry	1	1	0	1	0	1	0	0	9/18
CZ	Call on zero	1	1	0	0	1	1	0	0	9/18
CNZ	Call on no zero	1	1	0	0	0	1	0	0	9/18
CP	Call on positive	1	1	1	1	0	1	0	0	9/18
CM	Call on minus	1	1	1	1	1	1	0	0	9/18
CPE	Call on parity even	1	1	1	0	1	1	0	0	9/18
CPO	Call on parity odd	1	1	1	0	0	1	0	0	9/18
RETURN										
RET	Return	1	1	0	0	1	0	0	1	10
RC	Return on carry	1	1	0	1	1	0	0	0	6/12
RNC	Return on no carry	1	1	0	1	0	0	0	0	6/12
RZ	Return on zero	1	1	0	0	1	0	0	0	6/12
RNZ	Return on no zero	1	1	0	0	0	0	0	0	6/12
RP	Return on positive	1	1	1	1	0	0	0	0	6/12
RM	Return on minus	1	1	1	1	1	0	0	0	6/12
RPE	Return on parity even	1	1	1	0	1	0	0	0	6/12
RPO	Return on parity odd	1	1	1	0	0	0	0	0	6/12
RESTART										
RST	Restart	1	1	A	A	A	1	1	1	12
INPUT/OUTPUT										
IN	Input	1	1	0	1	1	0	1	1	10
OUT	Output	1	1	0	1	0	0	1	1	10

INSTRUCTION SET SUMMARY (Cont.)

Mnemonic*	Description	Instruction Code (Note 1) D7	D6	D5	D4	D3	D2	D1	D0	Clock Cycles (Note 2)
INCREMENT AND DECREMENT										
INR r	Increment register	0	0	D	D	D	1	0	0	4
DCR r	Decrement register	0	0	D	D	D	1	0	1	4
INR M	Increment memory	0	0	1	1	0	1	0	0	10
DCR M	Decrement memory	0	0	1	1	0	1	0	1	10
INX B	Increment B & C registers	0	0	0	0	0	0	1	1	6
INX D	Increment D & E registers	0	0	0	1	0	0	1	1	6
INX H	Increment H & L registers	0	0	1	0	0	0	1	1	6
INX SP	Increment stack pointer	0	0	1	1	0	0	1	1	6
DCX B	Decrement B & C	0	0	0	0	1	0	1	1	6
DCX D	Decrement D & E	0	0	0	1	1	0	1	1	6
DCX H	Decrement H & L	0	0	1	0	1	0	1	1	6
DCX SP	Decrement stack pointer	0	0	1	1	1	0	1	1	6
ADD										
ADD r	Add register to A	1	0	0	0	0	S	S	S	4
ADC r	Add register to A with carry	1	0	0	0	1	S	S	S	4
ADD M	Add memory to A	1	0	0	0	0	1	1	0	7
ADC M	Add memory to A with carry	1	0	0	0	1	1	1	0	7
ADI	Add immediate to A	1	1	0	0	0	1	1	0	7
ACI	Add immediate to A with carry	1	1	0	0	1	1	1	0	7
DAD B	Add B & C to H & L	0	0	0	0	1	0	0	1	10
DAD D	Add D & E to H & L	0	0	0	1	1	0	0	1	10
DAD H	Add H & L to H & L	0	0	1	0	1	0	0	1	10
DAD SP	Add stack pointer to H & L	0	0	1	1	1	0	0	1	10
SUBTRACT										
SUB r	Subtract register from A	1	0	0	1	0	S	S	S	4
SBB r	Subtract register from A with borrow	1	0	0	1	1	S	S	S	4
SUB M	Subtract memory from A	1	0	0	1	0	1	1	0	7
SBB M	Subtract memory from A with borrow	1	0	0	1	1	1	1	0	7
SUI	Subtract immediate from A	1	1	0	1	0	1	1	0	7
SBI	Subtract immediate from A with borrow	1	1	0	1	1	1	1	0	7
LOGICAL										
ANA r	And register with A	1	0	1	0	0	S	S	S	4
XRA r	Exclusive Or register with A	1	0	1	0	1	S	S	S	4
ORA r	Or register with A	1	0	1	1	0	S	S	S	4
CMP r	Compare register with A	1	0	1	1	1	S	S	S	4
ANA M	And memory with A	1	0	1	0	0	1	1	0	7
XRA M	Exclusive Or memory with A	1	0	1	0	1	1	1	0	7
ORA M	Or memory with A	1	0	1	1	0	1	1	0	7
CMP M	Compare memory with A	1	0	1	1	1	1	1	0	7
ANI	And immediate with A	1	1	1	0	0	1	1	0	7
XRI	Exclusive Or immediate with A	1	1	1	0	1	1	1	0	7
ORI	Or immediate with A	1	1	1	1	0	1	1	0	7
CPI	Compare immediate with A	1	1	1	1	1	1	1	0	7
ROTATE										
RLC	Rotate A left	0	0	0	0	0	1	1	1	4
RRC	Rotate A right	0	0	0	0	1	1	1	1	4
RAL	Rotate A left through carry	0	0	0	1	0	1	1	1	4
RAR	Rotate A right through carry	0	0	0	1	1	1	1	1	4
SPECIALS										
CMA	Complement A	0	0	1	0	1	1	1	1	4
STC	Set carry	0	0	1	1	0	1	1	1	4
CMC	Complement carry	0	0	1	1	1	1	1	1	4
DAA	Decimal adjust A	0	0	1	0	0	1	1	1	4
CONTROL										
EI	Enable Interrupts	1	1	1	1	1	0	1	1	4
DI	Disable Interrupts	1	1	1	1	0	0	1	1	4
NOP	No operation	0	0	0	0	0	0	0	0	4
HLT	Halt	0	1	1	1	0	1	1	0	5
NEW Am8085A INSTRUCTIONS										
RIM	Read Interrupt Mask	0	0	1	0	0	0	0	0	4
SIM	Set Interrupt Mask	0	0	1	1	0	0	0	0	4

Notes: 1. DOD or SSS: 8=000, C=001, D=010, E=011, H=100, L=101, Memory=110, A=111.
2. Two possible cycle times (6/12) indicate instruction cycles dependent on condition flags.

*All mnemonics copyright © Intel Corporation 1977

8155/8156

2048-Bit Static MOS RAM With I/O Ports and Timer

DISTINCTIVE CHARACTERISTICS

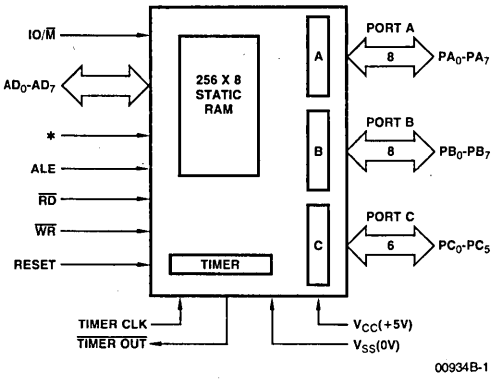
- 256 word x 8-bits
- Single +5V power supply
- Completely static operation
- Internal address latch
- 2 programmable 8-bit I/O ports
- 1 programmable 6-bit I/O port
- Programmable 14-bit binary counter/timer
- Multiplexed address and data bus

GENERAL DESCRIPTION

The 8155 and 8156 are RAM and I/O chips to be used in the 8085A MPU system. The RAM portion is designed with 2K bit static cells organized as 256 x 8. They have a maximum access time of 400ns to permit use with no wait states in 8085A CPU. The 8155-2 and 8156-2 have maximum access times of 330ns for use with the 8085A. The I/O portion consists of three general purpose I/O ports. One of the three ports can be programmed to be status pins, thus allowing the other two ports to operate in handshake mode.

A 14-bit programmable counter/timer is also included on chip to provide either a square wave or terminal count pulse for the CPU system depending on timer mode.

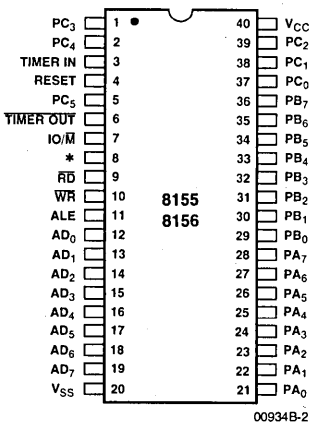
BLOCK DIAGRAM



*8155 = $\overline{CE}$, 8156 = CE

CONNECTION DIAGRAM

Top View
D-40-1
P-40-1



Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Package Type	Temperature Range	Order Numbers			
		8155		8156	
		400ns	330ns	400ns	330ns
Molded DIP	0°C ≤ T_A ≤ 70°C	P8155	P8155-2	P8156	P8156-2
		P8155H	P8155H-2	P8156H	P8156H-2
Hermetic DIP		D8155	D8155-2	D8156	D8156-2
		D8155H	D8155H-2	D8156H	D8156H-2
Hermetic DIP	−40°C ≤ T_A ≤ 85°C	ID8155	ID8155-2	ID8156	ID8156-2
		ID8155H	ID8155H-2	ID8156H	ID8156H-2
Hermetic DIP	−55°C ≤ T_A ≤ 125°C	MD8155B		MD8156B	

FUNCTIONAL PIN DEFINITION

The following describes the functions of all of the 8155/8156 pins.

RESET

The Reset signal is a pulse provided by the 8085A to initialize the system. Input high on this line resets the chip and initializes the three I/O ports to input mode. The width of RESET pulse should typically by 600ns. (Two 8085A clock cycle times).

AD_0-AD_7

These are 3-state Address/Data lines that interface with the CPU lower 8-bit Address/Data Bus. The 8-bit address is latched into the address latch on the falling edge of the ALE. The address can be either for the memory section or the I/O section depending on the polarity of the IO/$\overline{M}$ input signal. The 8-bit data is either written into the chip or read from the chip depending on the status of $\overline{WRITE}$ or $\overline{READ}$ input signal.

CE OR $\overline{CE}$

Chip Enable: On the 8155, this pin is $\overline{CE}$ and is ACTIVE LOW. On the 8156, this pin is CE and is ACTIVE HIGH.

$\overline{RD}$

Input low on this line with the Chip Enable active enables the AD_{0-7} buffers. If IO/$\overline{M}$ pin is low, the RAM content will be read out to the AD bus. Otherwise the content of the selected I/O port will be read to the AD bus.

$\overline{WR}$

Input low on this line with the Chip Enable active causes the data on the AD lines to be written to the RAM or I/O ports depending on the polarity of IO/$\overline{M}$.

ALE

Address Latch Enable: This control signal latches both the address on the AD_{0-7} lines and the state of the Chip Enable and IO/$\overline{M}$ into the chip at the falling edge of ALE.

IO/$\overline{M}$

IO/$\overline{Memory}$ Select: This line selects the memory if low and selects the IO if high.

PA_0-PA_7

These 8 pins are general purpose I/O pins. The in/out direction is selected by programming the Command/Status Register.

PB_0-PB_7

These 8 pins are general purpose I/O pins. The in/out direction is selected by programming the Command/Status Register.

PC_0-PC_5

These 6 pins can function as either input port, output port, or as control signals for PA and PB. Programming is done through the C/S Register. When PC_{0-5} are used as control signals, they will provide the following:

- PC_0 – A INTR (Port A Interrupt)
- PC_1 – A BF (Port A Buffer full)
- PC_2 – $\overline{A\text{-}STB}$ (Port A Strobe)
- PC_3 – B INTR (Port B Interrupt)
- PC_4 – B BF (Port B Buffer Full)
- PC_5 – $\overline{B\ STB}$ (Port B Strobe)

TIMER IN

This is the input to the counter timer.

$\overline{TIMER\ OUT}$

This pin is the timer output. This output can be either a square wave or a pulse depending on the timer mode.

V_{CC}

+5 volt supply.

V_{SS}

Ground reference.

MAXIMUM RATINGS above which useful life may be impaired

Storage Temperature	−65°C to +150°C
V_{CC} with Respect to V_{SS}	−0.5V to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5V to +7.0V
Power Dissipation	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

DC CHARACTERISTICS

Parameters	Description		Test Conditions	Min	Typ	Max	Units
V_{IL}	Input Low Voltage			−0.5		0.8	Volts
V_{IH}	Input High Voltage			2.0		V_{CC} +0.5	Volts
V_{OL}	Output Low Voltage		*I_{OL} = 2mA			0.45	
V_{OH}	Output High Voltage		I_{OH} = −400μA	2.4			Volts
I_{IL}	Input Leakage		V_{IN} = V_{CC} to 0V			±10	μA
I_{LO}	Output Leakage Current		0.45V ≤ V_{OUT} ≤ V_{CC}			±10	μA
I_{CC}	V_{CC} Supply Current	8155, 8156	0.45V ≤ V_{OUT} ≤ V_{CC}			180	mA
		8155H, 8156H				125	mA
I_{IL} (CE)	Chip Enable Leakage	8155	V_{IN} = V_{CC} to 0V		+100		μA
		8156			−100		μA

*I_{OL} = 1.6mA for MD8155B and MD8156B

OPERATING RANGE

Part Number	Ambient Temperature	V_{CC}	V_{SS}
D8155, P8155 D8155-2, P8155-2 D8156, P8156 D8156-2, P8156-2	0 to 70°C	5.0V ±5%	0V
D8155H, P8155H D8155H-2, P8155H-2 D8156H, P8156H D8156H-2, P8156H-2	0 to 70°C	5.0V ±10%	0V
ID8155, ID8156 ID8155H, ID8156H ID8155-2, ID8156-2 ID8155H-2, ID8156H-2	−40 to 85°C	5.0V ±10%	0V
MD8155B, MD8156B	−55 to +125°C	5.0V ±10%	0V

AC CHARACTERISTICS

			8155/56/55H/56H ID8155, ID8156 ID8155H, ID8156H MD8155B, MD8156B		8155-2, 8156-2 8155H-2, 8156H-2		
Parameters	**Description**		**Min**	**Max**	**Min**	**Max**	**Units**
t_{AL}	Address to Latch Setup Time		50		30		ns
t_{LA}	Address Hold Time after Latch		80		30		ns
t_{LC}	Latch to READ/WRITE Control		100		40		ns
t_{RD}	Valid Data Out Delay from READ Control			170		140	ns
t_{AD}	Address Stable to Data Out Valid			400		330	ns
t_{LL}	Latch Enable Width		100		70		ns
t_{RDF}	Data Bus Float After READ		0	100	0	80	ns
t_{CL}	READ/WRITE Control to Latch Enable		20		10		ns
t_{CC}	READ/WRITE Control Width		250		200		ns
t_{DW}	Data In to WRITE Setup Time		150		100		ns
t_{WD}	Data In Hold Time After WRITE	COM'L, IND	25		25		ns
		MIL	0				ns
t_{RV}	Recovery Time Between Controls		300		200		ns
t_{WP}	WRITE to Port Output			400		300	ns
t_{PR}	Port Input Setup Time		70		50		ns
t_{RP}	Port Input Hold Time		50		10		ns
t_{SBF}	Strobe to Buffer Full			400		300	ns
t_{SS}	Strobe Width		200		150		ns
t_{RBE}	READ to Buffer Empty			400		300	ns
t_{SI}	Strobe to INTR On			400		300	ns
t_{RDI}	READ to INTR Off			400		300	ns
t_{PSS}	Port Setup Time to Strobe		50		0		ns
t_{PHS}	Port Hold Time After Strobe		120		100		ns
t_{SBE}	Strobe to Buffer Empty			400		300	ns
t_{WBF}	WRITE to Buffer Full			400		300	ns
t_{WI}	WRITE to INTR Off			400		300	ns
t_{TL}	TIMER-IN to TIMER-OUT Low			400		300	ns
t_{TH}	TIMER-IN to TIMER-OUT High			400		300	ns
t_{RDE}	Data Bus Enable from READ Control		10		10		ns
t_1	TIMER-IN Low Time		80		40		ns
t_2	TIMER-IN High Time		120		70		ns

Note: Test Condition; 150pF Load.

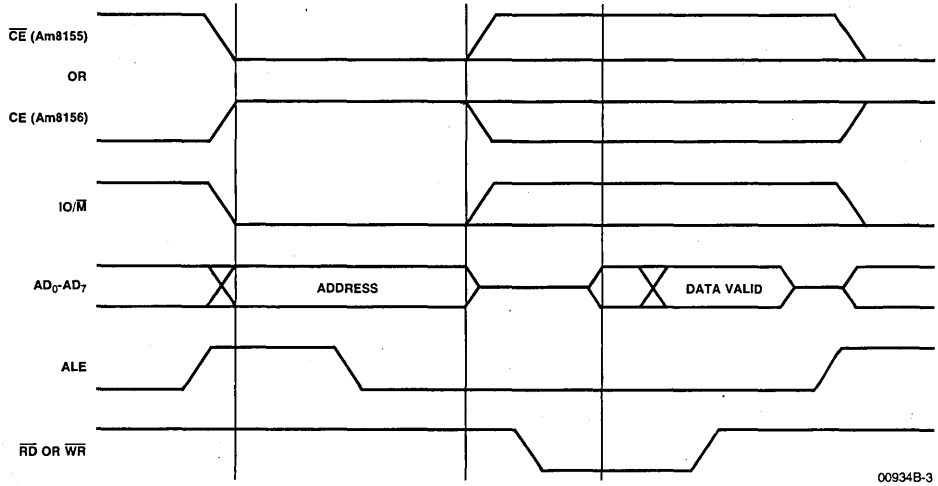
OPERATIONAL DESCRIPTION

The 8155-8156 includes the following operational features:

- 2K Bit Static RAM organized as 256 x 8
- Two 8-bit I/O ports (PA and PB) and one 6-bit I/O port (PC)
- 14-bit down counter

The I/O portion contains four registers (Command/Status, PA_{0-7}, PB_{0-7}, PC_{0-5}). The IO/$\overline{M}$ (IO/$\overline{\text{Memory}}$ Select) pin selects the I/O or the memory (RAM) portion. Detailed descriptions of memory, I/O ports and timer functions will follow.

The 8-bit address on the AD lines, the Chip Enable input, and IO/$\overline{M}$ are all latched on chip at the falling edge of ALE. A low on the IO/$\overline{M}$ must be provided to select the memory section.



Note: For detailed timing diagram information, see Figure 7 and AC Characteristics.

Figure 1. Memory Read/Write Cycle.

PROGRAMMING OF THE COMMAND/STATUS REGISTER

The command register consists of eight latches, one for each bit. Four bits (0-3) define the mode of the ports. Two bits (4-5) enable or disable the interrupt from port C when it acts as control port, and the last two bits (6-7) are for the timer.

The C/S register contents can be altered at any time by using the I/O address XXXXX000 during a WRITE operation. The meaning of each bit of the command byte is defined as follows:

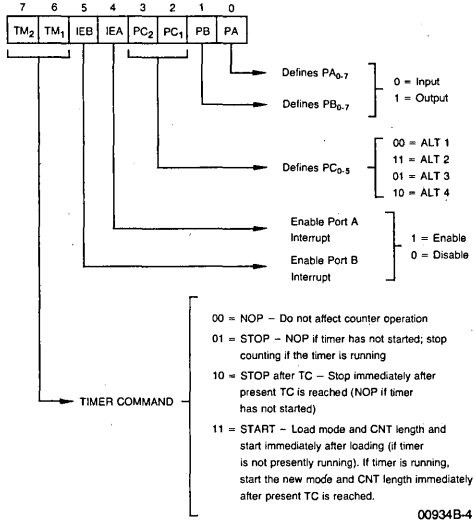


Figure 2. Command/Status Register Bit Assignment.

READING THE COMMAND/STATUS REGISTER

The status register consists of seven latches, one for each bit: six (0-5) for the status of the ports and one (6) for the status of the timer.

The status of the timer and the I/O section can be polled by reading the C/S Register (Address XXXXX000). Status word format is shown below:

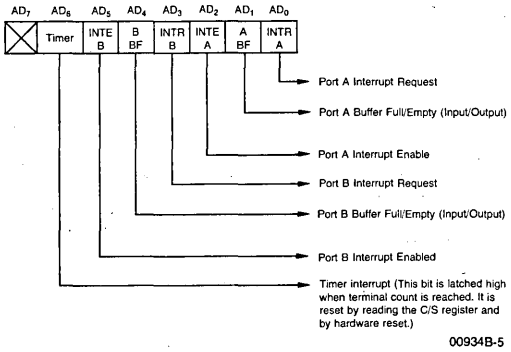


Figure 3. Command/Status Register Status Word Format.

INPUT/OUTPUT SECTION

The I/O section of the 8155/56 consists of four registers as described below.

- Command/Status Register (C/S) – This register is assigned the address XXXXX000. The C/S address serves the dual purpose.

  When the C/S register is selected during WRITE operation, a command is written into the command register. The contents of this register are not accessible through the pins.

  When the C/S (XXXXX000) is selected during a READ operation, the status information of the I/O ports and the timer become available on the AD_{0-7} lines.

- PA Register – This register can be programmed to be either input or output ports depending on the status of the contents of the C/S Register. Also depending on the command, this port can operate in either the basic mode or the strobed mode (see timing diagram). The I/O pins assigned in relation to this register are PA_{0-7}. The address of this register is XXXXX001.

- PB Register – This register functions the same as PA Register. The I/O pins assigned are PB_{0-7}. The address of this register is XXXXX010.

- PC Register – This register has the address XXXXX011 and contains only 6 bits. The 6 bits can be programmed to be either input ports, output ports or as control signals for PA and PB by properly programming the AD_2 and AD_3 bits of the C/S register.

  When PC_{0-5} is used as a control port, 3 bits are assigned for Port A and 3 for Port B. The first bit is an interrupt that the 8155 sends out. The second is an output signal indicating whether the buffer is full or empty, and the third is an input pin to accept a strobe for the strobed input mode. See Table 1.

When the 'C' port is programmed to either ALT3 or ALT4, the control signals for PA and PB are initialized as follows:

Control	Input Mode	Output Mode
BF	Low	Low
INTR	Low	High
$\overline{STB}$	Input Control	Input Control

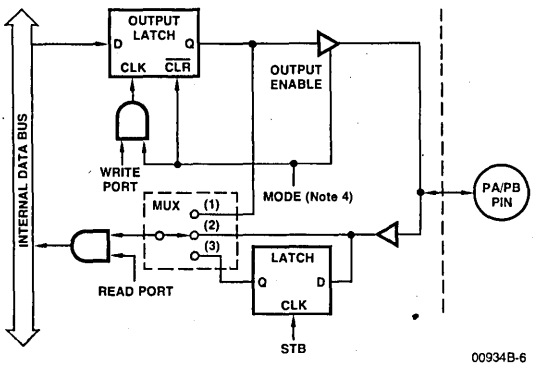
The set and reset of INTR and BF with respect to $\overline{STB}$, $\overline{WR}$ and $\overline{RD}$ timing is shown in Figure 8.

To summarize, the register's assignments are:

Address	Pinouts	Functions	No. of Bits
XXXXX000	Internal	Command/Status Register	8
XXXXX001	PA_{0-7}	General Purpose I/O Port	8
XXXXX010	PB_{0-7}	General Purpose I/O Port	8
XXXXX011	PC_{0-5}	General Purpose I/O Port or Control Lines	6

The following diagram shows how I/O Ports A and B are structured within the 8155 and 8156:

8155/8156
One Bit of Port A or Port B



Notes: 1. Output Mode
2. Simple Input
3. Strobed Input
} Multiplexer Control
4. = 1 for output mode
= 0 for input mode.

Read Port = (IO/$\overline{M}$ = 1) • ($\overline{RD}$ = 0) • (CE active) • (Port address selected)

Write Port = (IO/$\overline{M}$ = 1) • ($\overline{WR}$ = 0) • (CE active) • (Port address selected)

Note in the diagram that when the I/O ports are programmed to be output ports, the contents of the output ports can still be read by a READ operation when appropriately addressed.

Note also that the output latch is cleared when the port enters the input mode. The output latch cannot be loaded by writing to the port if the port is in the input mode. The result is that each time a port mode is changed from input to ouput, the output pins will go low. When the 8155/8156 is RESET, the output latches are all cleared and all 3 ports enter the input mode.

When in the ALT 1 or ALT 2 modes, the bits of Port C are structured like the diagram above in the simple input or output mode, respectively.

Reading from an input port with nothing connected to the pins will provide unpredictable results.

Table 1. Table of Port Control Assignment.

Pin	ALT 1	ALT 2	ALT 3	ALT 4
PC0	Input Port	Output Port	A INTR (Port A Interrupt)	A INTR (Port A Interrupt)
PC1	Input Port	Output Port	A BF (Port A Buffer Full)	A BF (Port A Buffer Full)
PC2	Input Port	Output Port	A $\overline{STB}$ (Port A Strobe)	A $\overline{STB}$ (Port A Strobe)
PC3	Input Port	Output Port	Output Port	B INTR (Port B Interrupt)
PC4	Input Port	Output Port	Output Port	B BF (Port B Buffer Full)
PC5	Input Port	Output Port	Output Port	B $\overline{STB}$ (Port B Strobe)

TIMER SECTION

The timer is a 14-bit down counter that counts the 'timer input' pulses and provides either a square wave or pulse when terminal count (TC) is reached.

The timer has the I/O address XXXXX100 for the low order byte of the register and the I/O address XXXXX101 for the high order byte of the register.

To program the timer, the COUNT LENGTH REG is loaded first, one byte at a time, by selecting the timer addresses. Bits 0-13 will specify the length of the next count and bits 14-15 will specify the timer output mode. The value loaded into the count length register can have any value from 2_H through $3FFF_H$ in bits 0-13.

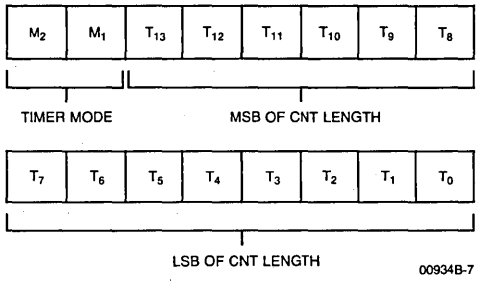


Figure 4. Timer Format.

There are four modes to choose from:

0 – Puts out low during second half of count
1 – Square wave
2 – Single pulse upon TC being reached
3 – Repetitive single pulse everytime TC is readied and automatic reload of counter upon TC being reached, until instructed to stop by a new command loaded into C/S.

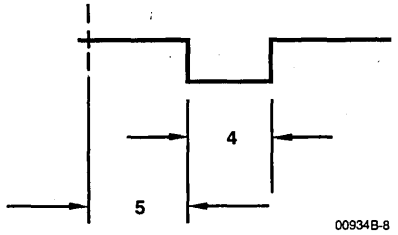
Bits 6-7 of Command/Status Register Contents are used to start and stop the counter. There are four commands to choose from (See the further description on Command/Status Register.).

C/S7	C/S6	
0	0	NOP – Do not affect counter operation.
0	1	STOP – NOP if timer has not started; stop counting if the timer is running.
1	0	STOP AFTER TC – Stop immediately after present TC is reached (NOP if timer has not started).
1	1	START – Load mode and CNT length and start immediately after loading (if timer is not presently running). If timer is running, start the new mode and CNT length immediately after present TC is reached.

M2 and M1 define the timer mode as follows:

M2	M1	
0	0	Puts out low during second half of count.
0	1	Square wave, i.e., the period of the square wave equals the count length programmed with automatic reload at terminal count.
1	0	Single pulse upon TC being reached.
1	1	Automatic reload, i.e., single pulse everytime TC is reached.

Note: In case of an asymmetric count, i.e., 9, larger half of the count will be high, the larger count will stay active as shown in Figure 5.



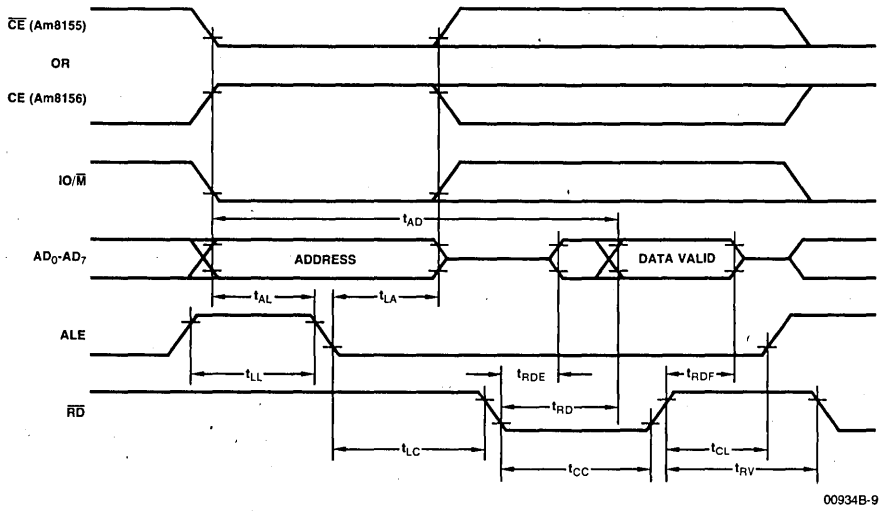
Note: 5 and 4 refer to the number of clock cycles in that time period.

Figure 5. Asymmetric Count.

The counter in the 8155 is not initialized to any particular mode or count when hardware RESET occurs, but RESET does stop the counting. Therefore, counting cannot begin following RESET until a START command is issued via the C/S register.

WAVEFORMS

A. READ CYCLE.



B. WRITE CYCLE.

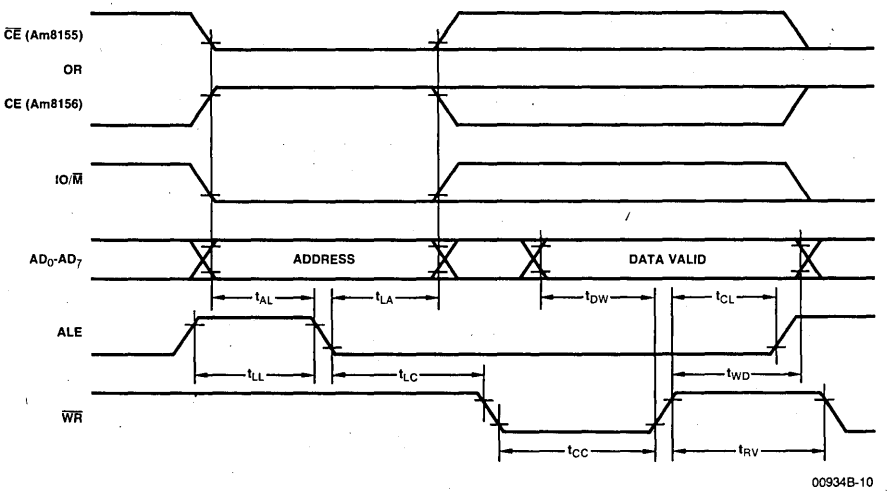
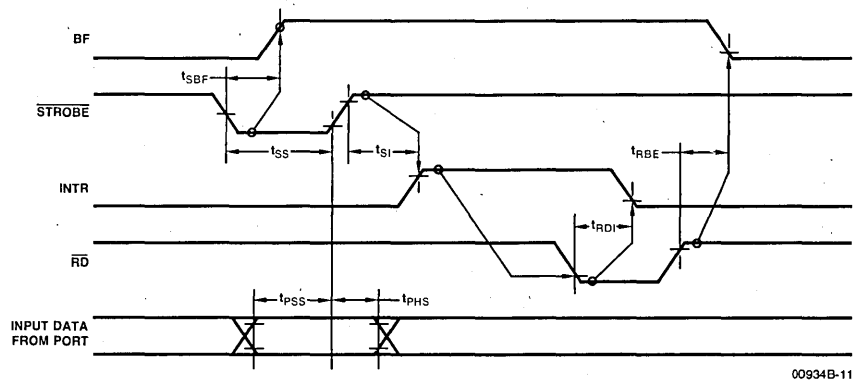


Figure 7. 8155/8156 Read/Write Timing Diagrams.

WAVEFORMS (Cont.)

A. STROBED INPUT MODE.



B. STROBED OUTPUT MODE.

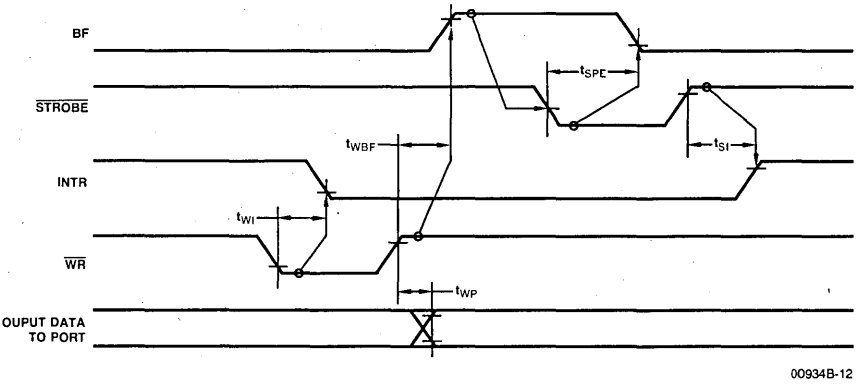
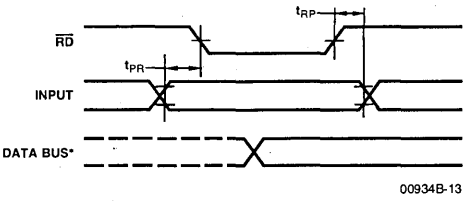
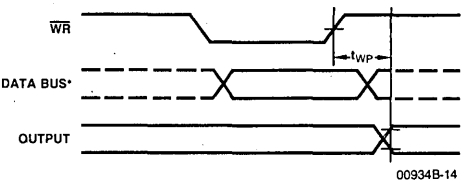


Figure 8. Strobed I/O Timing.

BASIC INPUT MODE.



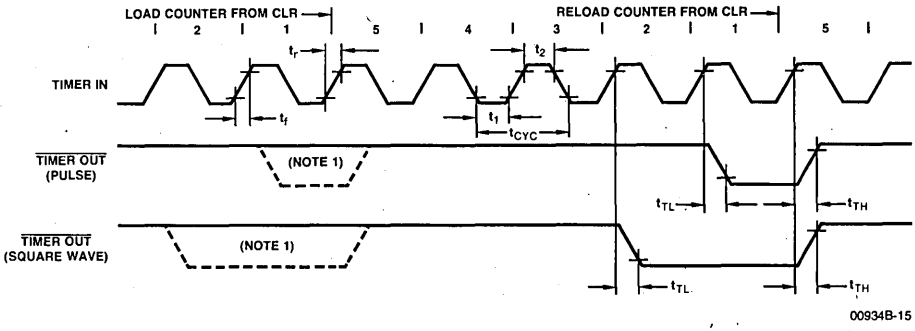
BASIC OUTPUT MODE.



*Data bus timing is shown in Figure 7.

Figure 9. Basic I/O Timing Waveform.

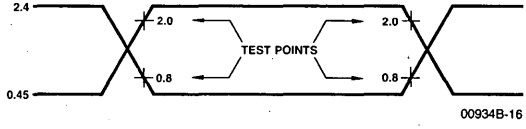
WAVEFORMS (Cont.)



Note 1: The timer output is periodic if in an automatic reload mode (M_1 mode bit = 1).

Figure 10. Timer Output Waveform Countdown from 5 to 1.

INPUT/OUTPUT WAVEFORMS FOR AC TESTS



Interface Support Products

Note: This section included for reference only. For complete data sheets, refer to "Bipolar Microprocessor Logic and Interface" data book, 1983 edition. For PAL information see AMD Programmable Array Logic Handbook, 1983 edition.

Am25LS2521

Eight-Bit Equal-to Comparator

DISTINCTIVE CHARACTERISTICS

- 8-bit byte oriented equal comparator
- Cascadable using $\overline{E}_{IN}$
- High-speed, Low-Power Schottky technology
- t_{pd} A • B to $\overline{E}_{OUT}$ in 9ns
- Standard 20-pin package

RELATED PRODUCTS

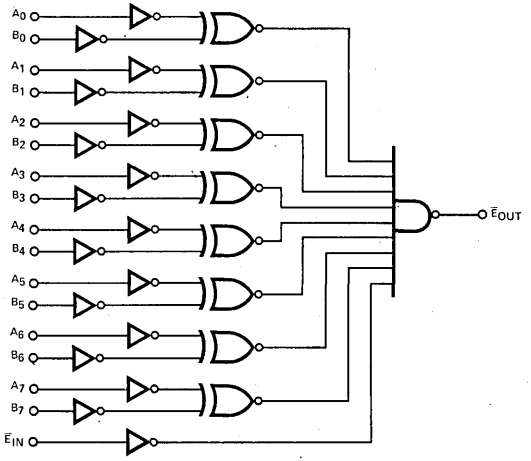
Part No.	Description
Am29806	Chip Select Decoder
Am29809	9-Bit Comparator

FUNCTIONAL DESCRIPTION

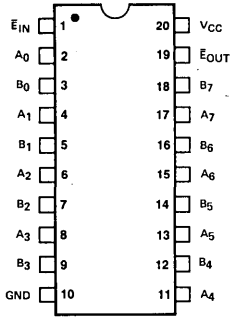
The Am25LS2521 is an 8-bit "equal to" comparator capable of comparing two 8-bit words for "equal to" with provision for expansion or external enabling. The matching of the two 8-bit inputs plus a logic LOW on the $\overline{E}_{IN}$ produces an active LOW on the output $\overline{E}_{OUT}$.

The logic expression for the device can be expressed as: $\overline{E}_{OUT} = \overline{(A_0 \odot B_0)\ (A_1 \odot B_1)\ (A_2 \odot B_2)\ (A_3 \odot B_3)\ (A_4 \odot B_4)\ (A_5 \odot B_5)\ (A_7 \odot B_7)\ E_{IN}}$. It is obvious that the expression is valid where $A_0 - A_7$ and $B_0 - B_7$ are expressed as either assertions or negations. This is also true for pair of terms i.e. A_0 can be compared with B_0 at the same time $\overline{A}_1$ is compared with $\overline{B}_1$. It is only essential that the polarity of the paired terms be maintained.

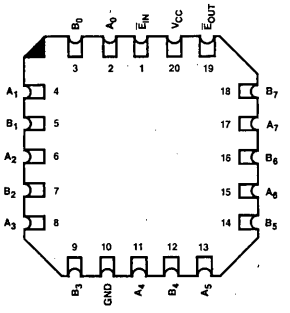
LOGIC DIAGRAM



CONNECTION DIAGRAMS – Top Views

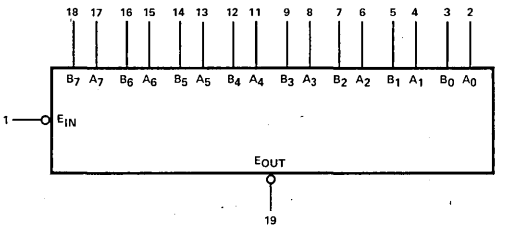


Leadless Chip Carrier
L-20-1



Note: Pin 1 is marked for orientation.

LOGIC SYMBOL



V_{CC} = Pin 20
GND = Pin 10

Am25LS2535

Eight Input Multiplexer with Control Register

DISTINCTIVE CHARACTERISTICS

- High speed eight-input multiplexer
- On-chip Multiplexer Select and Polarity Control Register
- Output polarity control for inverting or non-inverting output
- Common register enable
- Asynchronous register clear
- Three-state output for expansion
- Am25LS features improved noise margin, higher drive, and faster operation

FUNCTIONAL DESCRIPTION

The Am25LS2535 is an eight-input Multiplexer with Control Register. The device features high speed from clock to output and is intended for use in high speed computer control units or structured state machine designs.

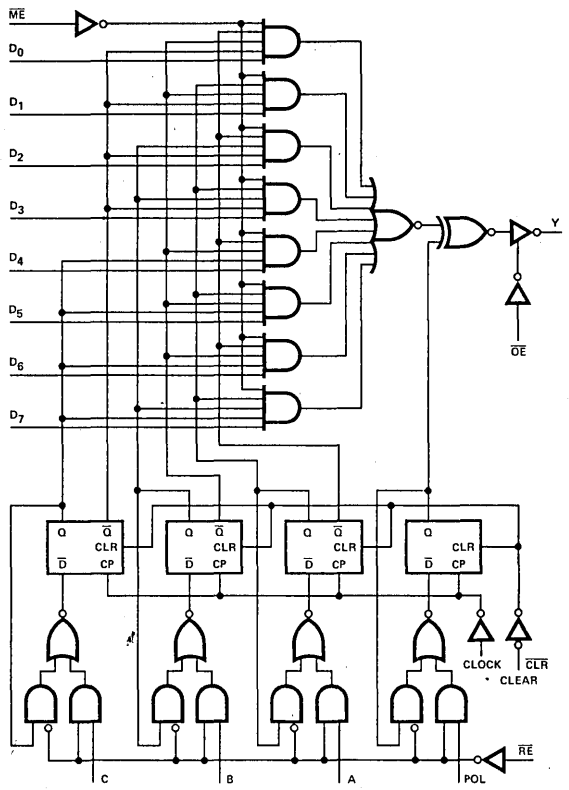
The Am25LS2535 contains an internal register which holds the A, B and C multiplexer select lines as well as the POL (polarity) control bit. When the Register Enable input ($\overline{RE}$) is LOW, new data is entered into the register on the LOW-to-HIGH, transition of the clock. When $\overline{RE}$ is HIGH, the register retains its current data. An asynchronous clear input ($\overline{CLR}$) is used to reset the register to a logic LOW level.

The A, B and C register outputs select one of eight multiplexer data inputs. A HIGH on the Polarity Control flip-flop output causes a true (non-inverting) multiplexer output, and a LOW causes the output to be inverted. In a computer control unit, this allows testing of either true or complemented flag data at the microprogram sequencer test input.

An active LOW Multiplexer Enable input ($\overline{ME}$) allows the selected multiplexer input to be passed to the output. When $\overline{ME}$ is HIGH, the output is determined only by the Polarity Control bit.

The Am25LS2535 also features a three-state Output Enable control ($\overline{OE}$) for expansion. When $\overline{OE}$ is LOW, the output is enabled. When $\overline{OE}$ is HIGH, the output is in the high impedance state.

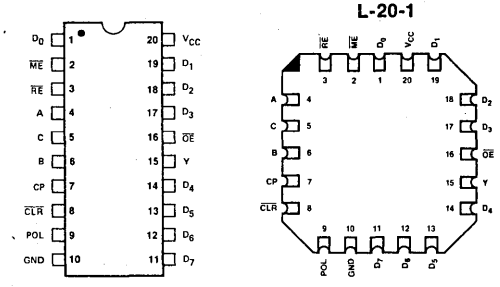
LOGIC DIAGRAM



RELATED PRODUCTS

Part No.	Description
Am2922	8 Input Multiplexer
Am2923	8 Input Multiplexer

CONNECTION DIAGRAMS – Top Views



Leadless Chip Carrier
L-20-1

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Package Type	Temperature Range	Order Number
Molded DIP	0 to +70°C	AM25LS2535PC
Hermetic DIP	0 to +70°C	AM25LS2535DC
Chip-Pak	0 to +70°C	AM25LS2535LC
Dice	0 to +70°C	AM25LS2535XC
Hermetic DIP	−55 to +125°C	AM25LS2535DM
Hermetic Flat Pack	−55 to +125°C	AM25LS2535FM
Chip-Pak	−55 to +125°C	AM25LS2535LM
Dice	−55 to +125°C	AM25LS2535XM

Am25LS2536

Eight-Bit Decoder with Control Storage

DISTINCTIVE CHARACTERISTICS

- 8-bit decoder/demultiplexer with control storage
- 3-state outputs
- Common clock enable
- Common clear
- Polarity control
- Advanced Low Power Schottky Process

RELATED PRODUCTS

Part No.	Description
Am25LS2537	1 of 10 Decoder
Am25LS2538	1 of 8 Decoder
Am25LS2539	Dual 1 of 4 Decoder
Am25LS2548	Chip Select Address Decoder
Am2921	1 of 8 Decoder
Am2924	3 to 8 Line Decoder/Demultiplexer

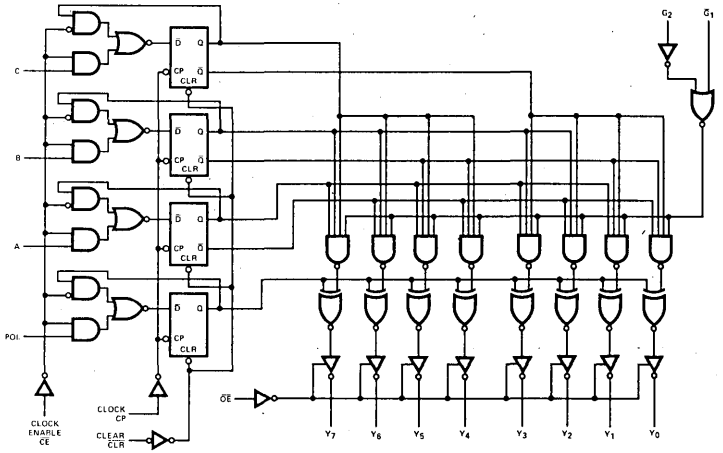
FUNCTIONAL DESCRIPTION

The Am25LS2536 is an eight-bit decoder with control storage. It provides a conventional 8-bit decoder function with two enable inputs which may also be used for data input. This can be used to implement a demultiplexer function. In addition, the exclusive "OR" gate allows for polarity control of the selected output. The 3-state outputs are enabled by a LOW on the ($\overline{OE}$) output enable.

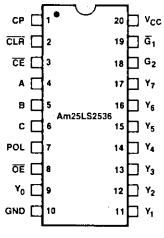
The three control bits representing the output selection and the single bit polarity control are stored in "D" type flip-flops. These flip-flops have both Clear, Clock, and Clock Enable functions provided. The $\overline{G_1}$ and G_2 input provide either polarity for input control or data.

LOGIC DIAGRAM

8-Bit Decoder/Demultiplexer with Control Storage

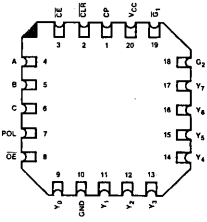


CONNECTION DIAGRAMS — Top Views



Note: Pin 1 is marked for orientation.

Leadless Chip Carrier
L-20-1



ORDERING INFORMATION

Package Type	Temperature Range	Order Number
Molded DIP	0 to +70°C	AM25LS2536PC
Hermetic DIP	0 to +70°C	AM25LS2536DC
Chip-Pak	0 to +70°C	AM25LS2536LC
Dice	0 to +70°C	AM25LS2536XC
Hermetic DIP	−55 to +125°C	AM25LS2536DM
Hermetic Flat-Pak	−55 to +125°C	AM25LS2536FM
Chip-Pak	−55 to +125°C	AM25LS2536LM
Dice	−55 to +125°C	AM25LS2536XM

Am25LS2548

Chip Select Address Decoder with Acknowledge

DISTINCTIVE CHARACTERISTICS

- One-of-Eight Decoder provides eight chip select outputs
- Acknowledge output responds to enables and read or write command
- Open-collector Acknowledge output for wired-OR application
- Inverting and non-inverting enable inputs for upper address decoding

RELATED PRODUCTS

Part No.	Description
Am25LS2536	8-Bit Decoder
Am25LS2537	1 of 10 Decoder
Am25LS2538	1 of 8 Decoder
Am25LS2539	Dual 1 of 4 Decoder
Am2921	1 of 8 Decoder
Am2924	3 to 8 Line Decoder/Demultiplexer

FUNCTIONAL DESCRIPTION

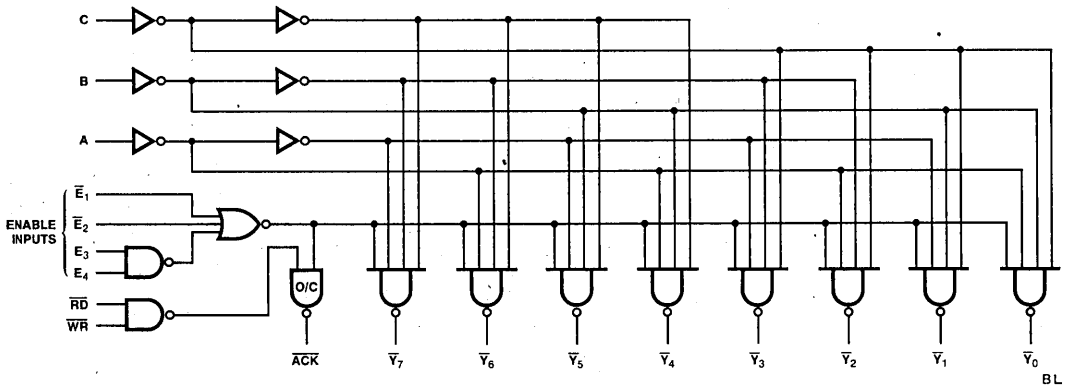
The Am25LS2548 Address Decoder combines a three-line to eight-line decoder with four qualifying enable inputs (two active HIGH and two active LOW) and the acknowledge output required for "ready" or "wait state" control of all popular MOS microprocessors.

The acknowledge output, $\overline{ACK}$, is active LOW and responds to the combination of all enables active and a read ($\overline{RD}$) or write ($\overline{WR}$) input command.

The eight chip select outputs are individually active LOW in response to the combination of all enables active and the corresponding 3-bit input code at inputs A, B, and C.

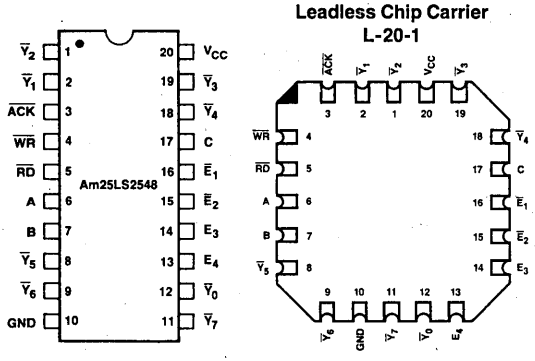
The Am25LS2548 is intended for chip select decoding in small, medium or large systems where multiple chip selects must be generated and address space must be allocated conservatively.

LOGIC DIAGRAM



BLI-045

CONNECTION DIAGRAMS – Top Views

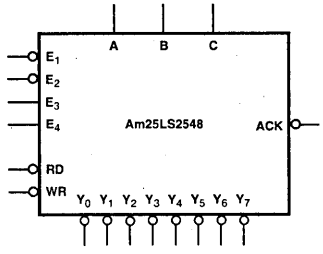


Leadless Chip Carrier
L-20-1

Note: Pin 1 is marked for orientation.

BLI-046

LOGIC SYMBOL



BLI-047

Am2960 • Am2960-1
Am2960A

Cascadable 16-Bit Error Detection and Correction Unit

DISTINCTIVE CHARACTERISTICS

- **Boosts Memory Reliability**
  Corrects all single-bit errors. Detects all double and some triple-bit errors. Reliability of dynamic RAM systems is increased more than 60-fold.
- **Very High Speed**
  Perfect for MOS microprocessor, minicomputer, and mainframe systems.
  - Data in to error detect: 32ns worst case.
  - Data in to corrected data out: 65ns worst case.

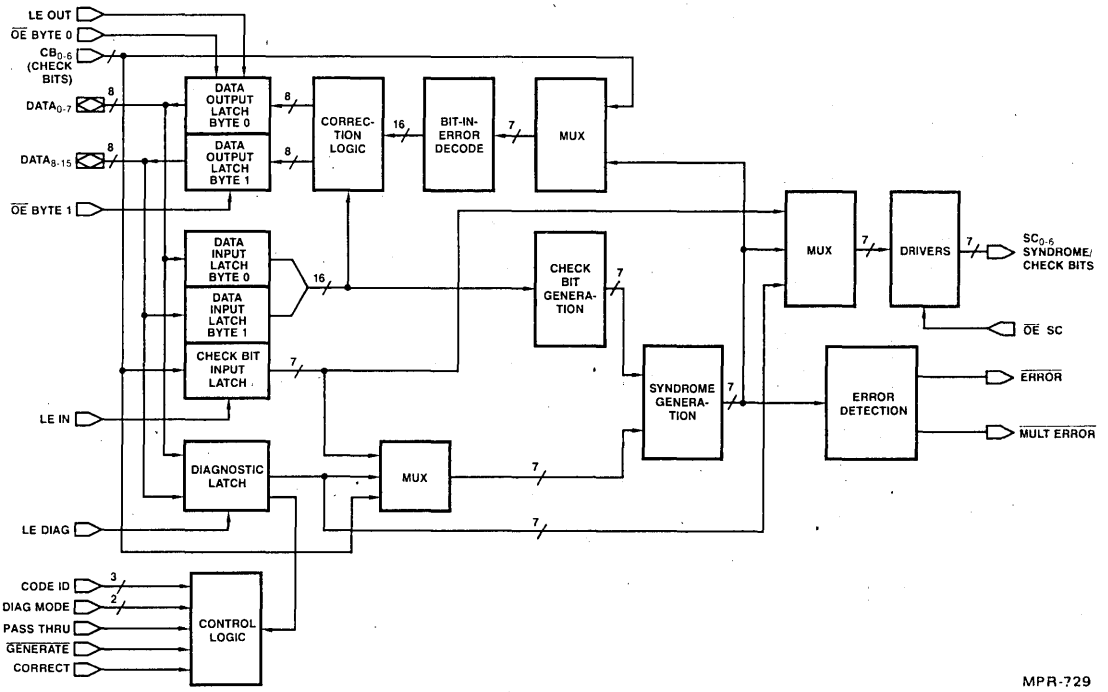
  High performance systems can use the Am2960 EDC in check-only mode to avoid memory system slowdown.
- **Replaces 25 to 50 MSI chips**
  All necessary features are built-in to the Am2960 EDC, including diagnostics, data in, data out, and check bit latches.
- **Handles Data Words From 8 to 64 Bits**
  The Am2960 EDC cascades: 1 EDC for 8 or 16 bits, 2 for 32 bits, 4 for 64 bits.
- **Easy Byte Operations**
  Separate byte enables on the data out latch simplify the steps and cuts the time required for byte writes.
- **Diagnostics Built-In**
  The processor may completely exercise the EDC under software control to check for proper operation of the EDC.

GENERAL DESCRIPTION

The Am2960 Error Detection and Correction Unit (EDC) contains the logic necessary to generate check bits on a 16-bit data field according to a modified Hamming Code, and to correct the data word when check bits are supplied. Operating on data read from memory, the Am2960 will correct any single bit error and will detect all double and some triple bit errors. For 16-bit words, 6 check bits are used. The Am2960 is expandable to operate on 32-bit words (7 check bits) and 64-bit words (8 check bits). In all configurations, the device makes the error syndrome available on separate outputs for data logging.

The Am2960 also features two diagnostic modes, in which diagnostic data can be forced into portions of the chip to simplify device testing and to execute system diagnostic functions. The product is supplied in a 48 lead hermetic DIP package.

BLOCK DIAGRAM



These devices are also characterized as:
AmZ8161
AmZ8162

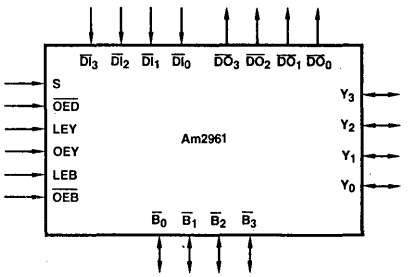
Am2961 • Am2962

4-Bit Error Correction Multiple Bus Buffers

DISTINCTIVE CHARACTERISTICS

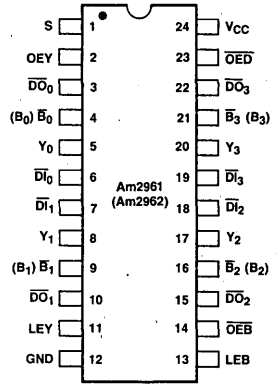
- Quad high-speed LSI bus-transceiver
- Provides complete data path interface between the Am2960 Error Detection and Correction Unit, the system data bus and dynamic RAM memory
- Three-state 24mA output to data bus
- Three-state data output to memory
- Inverting data bus for Am2961 and noninverting for Am2962
- Data bus latches allow operation with multiplexed buses
- Space saving 24-pin 0.3" package

LOGIC SYMBOL



B-Bus is noninverting for Am2962.

CONNECTION DIAGRAM
Top View



24 pin slim (0.3")

Note: Pin 1 is marked for orientation. BLI-122

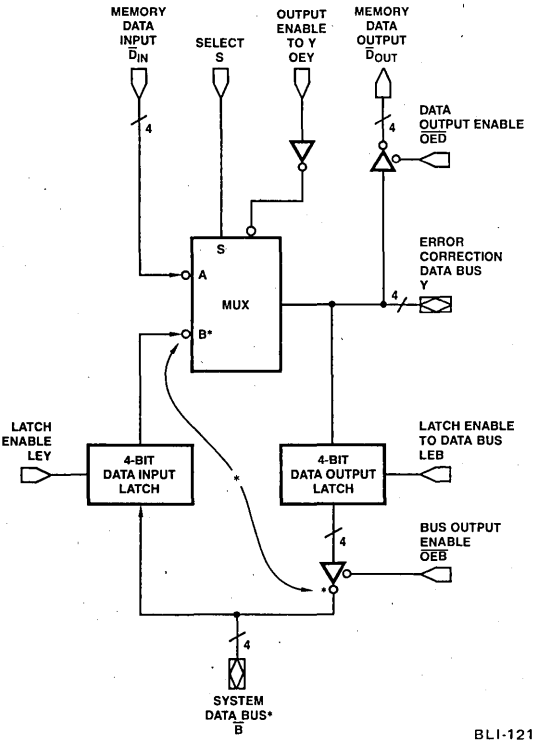
FUNCTIONAL DESCRIPTION

The Am2961 and Am2962 are high-performance, low-power Schottky multiple bus buffers that provide the complete data path interface between the Am2960 Error Detection and Correction Unit, dynamic RAM memory and the system data bus. The Am2961 provides an inverting data path between the data bus (B_i) and the Am2960 error correction data input (Y_i) and the Am2962 provides a noninverting configuration (B_i to Y_i). Both devices provide inverting data paths between the Am2960 and memory data bus thereby optimizing internal data path speeds.

The Am2961 and Am2962 are 4-bit devices. Four devices are used to interface each 16-bit Am2960 Error Detection and Correction Unit with dynamic memory. The system can easily be expanded to 32 or more bits for wider memory applications. The 4-bit configuration allows enabling the appropriate devices two-at-a-time for intermixed word or byte, read and write in 16-bit systems with error correction.

Data latches between the error correction data bus and the system data bus facilitate byte writing in memory systems wider than 8-bits. They also provide a data holding capability during single-step system operation.

LOGIC DIAGRAM



BLI-121

*Am2962 is the same function but noninverting to the system data bus, B.

Am2964B

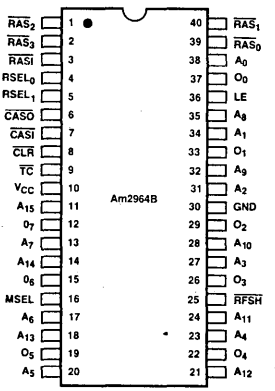
Dynamic Memory Controller

DISTINCTIVE CHARACTERISTICS

- Dynamic Memory Controller for 16K and 64K MOS dynamic RAMs
- 8-Bit Refresh Counter for refresh address generation, has clear input and terminal count output
- Refresh Counter terminal count selectable at 256 or 128
- Latch input $\overline{RAS}$ Decoder provides 4 $\overline{RAS}$ outputs, all active during refresh
- Dual 8-Bit Address Latches plus separate $\overline{RAS}$ Decoder Latches
- Grouping functions on a common chip minimizes speed differential or skew between address, $\overline{RAS}$ and $\overline{CAS}$ outputs
- 3-Port, 8-Bit Address Multiplexer with Schottky speed
- Burst mode, distributed refresh or transparent refresh mode determined by user
- Noninverting address, $\overline{RAS}$ and $\overline{CAS}$ paths

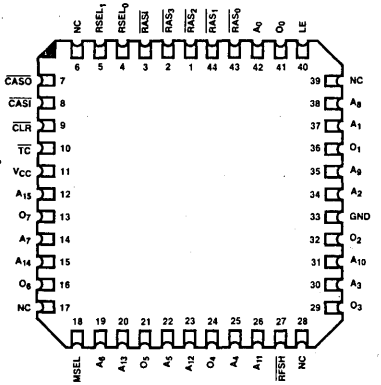
CONNECTION DIAGRAMS
Top Views

DIP



BLI-124

Chip-Pak



Note: Pin 1 is marked for orientation.

FUNCTIONAL DESCRIPTION

The Am2964B Dynamic Memory Controller (DMC) replaces a dozen MSI devices by grouping several unique functions. Two 8-bit latches capture and hold the memory address. These latches and a clearable, 8-bit refresh counter feed into an 8-bit, 3-input, Schottky speed MUX, for output to the dynamic RAM address lines.

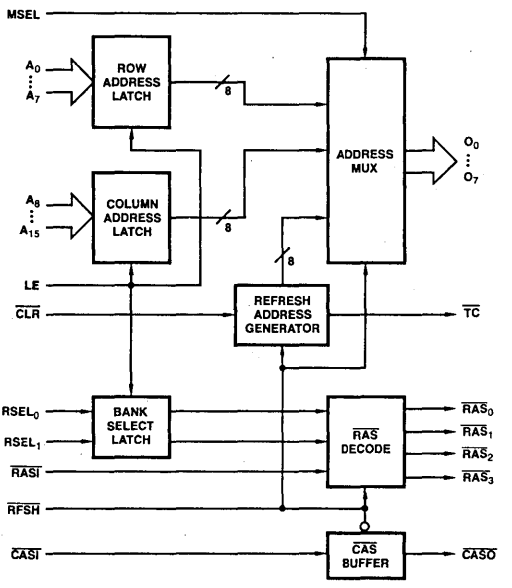
The same silicon chip also includes a special $\overline{RAS}$ decoder and $\overline{CAS}$ buffer. Placing these functions on the same chip minimizes the time skew between output functions which would otherwise be separate MSI chips, and therefore, allows a faster memory cycle time by the amount of skew eliminated.

The $\overline{RAS}$ Decoder allows upper addresses to select one-of-four banks of RAM by determining which bank receives a $\overline{RAS}$ input. During refresh ($\overline{RFSH}$ = LOW) the decoder mode is changed to four-of-four and all banks of memory receive a $\overline{RAS}$ input for refresh in response to a $\overline{RASI}$ active LOW input. $\overline{CAS}$ is inhibited during refresh.

Burst mode refresh is accomplished by holding $\overline{RFSH}$ LOW and toggling $\overline{RASI}$.

A_{15} is a dual function input which controls the refresh counter's range. For 64K RAMs it is an address input. For 16K RAMs it can be pulled to +12V through 1KΩ to terminate the refresh count at 128 instead of 256.

LOGIC DIAGRAM



BLI-123

This device is also characterized as:
AmZ8165
AmZ8166

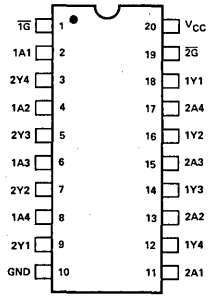
Am2965 • Am2966

Octal Dynamic Memory Drivers with Three-State Outputs

DISTINCTIVE CHARACTERISTICS

- **Controlled rise and fall characteristics**
  Internal resistors provide symmetrical drive to HIGH and LOW states, eliminating need for external series resistor.
- **Output swings designed to drive 16K and 64K RAMs**
  V_{OH} guaranteed at V_{CC} −1.15V. Undershoot going LOW guaranteed at less than 0.5V.
- **Large capacitive drive capability**
  35mA min source or sink current at 2.0V. Propagation delays specified for 50pF and 500pF loads.
- **Pin-compatible with 'S240 and 'S244**
  Non-inverting Am2966 replaces 74S244; inverting Am2965 replaces 74S240. Faster than 'S240/244 under equivalent load.
- **No-glitch outputs**
  Outputs forced into OFF state during power up and down. No glitch coming out of three-state.

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

BLI-125

FUNCTIONAL DESCRIPTION

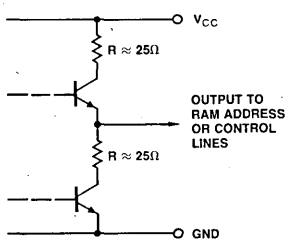
The Am2965 and Am2966 are designed and specified to drive the capacitive input characteristics of the address and control lines of MOS dynamic RAMs. The unique design of the lower output driver includes a collector resistor to control undershoot on the HIGH-to-LOW transition. The upper output driver pulls up to V_{CC} − 1.15V to be compatible with MOS memory and is designed to have a rise time symmetrical with the lower output's controlled fall time. This allows optimization of Dynamic RAM performance.

The Am2965 and Am2966 are pin-compatible with the popular 'S240 and 'S244 with identical 3-state output enable controls. The Am2965 has inverting drivers and the Am2966 has non-inverting drivers.

The inclusion of an internal resistor in the lower output driver eliminates the requirement for an external series resistor, therefore reducing package count and the board area required. The internal resistor controls the output fall and undershoot without slowing the output rise.

These devices are designed for use with the Am2964 Dynamic Memory Controller where large dynamic memories with highly capacitive input lines require additional buffering. Driving eight address lines or four $\overline{RAS}$ and four $\overline{CAS}$ lines with drivers on the same silicon chip also provides a significant performance advantage by minimizing skew between drivers. Each device has specified skew between drivers to improve the memory access worst case timing over the min and max t_{PD} difference of unspecified devices.

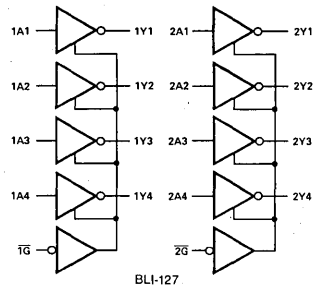
TYPICAL OUTPUT DRIVER



BLI-126

LOGIC DIAGRAMS

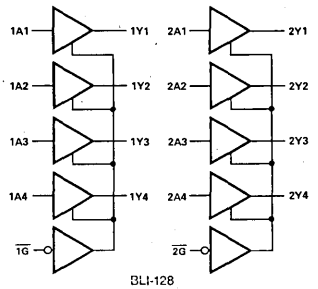
Am2965



BLI-127

Inputs		Outputs
$\overline{G}$	A	Y
H	X	Z
L	H	L
L	L	H

Am2966



BLI-128

Inputs		Outputs
$\overline{G}$	A	Y
H	X	Z
L	L	L
L	H	H

Am29806/Am29809

6-Bit Chip Select Decoder
9-Bit Equal-to Comparator

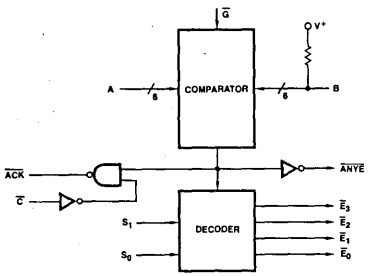
PRELIMINARY

DISTINCTIVE CHARACTERISTICS

- High-speed, expandable, 9-bit "equal-to" comparator (Am29809)
- High-speed comparator with chip select decoder (Am29806)
- Multibus™ compatible, open-collector acknowledge output
- Internal pull-up resistors on all B inputs
- Acknowledge timing control input
- 24-pin, 0.3" space-saving package
- Fully TTL-compatible inputs and outputs
- IMOX™ high-performance IMplanted OXide isolated process

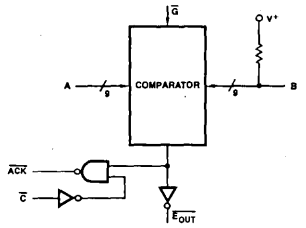
BLOCK DIAGRAM

Am29806 (6-Bit)



03424B-1

Am29809 (9-Bit)



03424B-2

GENERAL DESCRIPTION

Am29809 9-Bit Comparator

The Am29809 is a 9-bit "equal-to" comparator. Its combinatorial, active LOW output, $\overline{E}_{OUT}$, responds to the combination of a LOW input on the enable input $\overline{G}$ and a match between input words A and B.

Am29806 Chip Select Decoder

The Am29806 combines a 6-bit "equal-to" comparator with a 2- to 4-line decoder to select one-of-four active LOW chip select outputs. The selected output becomes active in response to the select inputs S_0, S_1 and is enabled by an active LOW input on the enable input $\overline{G}$ and a match between comparator inputs A and B. The active LOW output, Any Enable ($\overline{ANYE}$), responds to a valid comparison of A and B and is intended for use as an output enable control for data path buffers associated with the selected peripheral or board.

Both devices have open collector, active LOW acknowledge outputs with a conditional timing input $\overline{C}$ that may be driven by a timing circuit or wait state generator. The acknowledge output responds to a valid comparison, $\overline{G}$ = LOW and $\overline{C}$ = LOW.

Both devices have internal pull-up resistors on the comparator B-inputs for easy connection to SPST switches to ground selected input lines. The comparator function is described by:

$$\overline{E}_{OUT} = \overline{(A_0 \cdot B_0)\,(A_1 \cdot B_1)\,(A_2 \cdot B_2) \ldots (A_i \cdot B_i)\,G}$$

ORDERING INFORMATION

Order the part number according to the table below to obtain the desired package, temperature range and screening level.

Order Number	Order Number	Package Type (Note 1)	Operating Range (Note 2)	Screening Level (Note 3)
Am29806PC	Am29809PC	D-24-SLIM	C	C-1
Am29806PC-B	Am29809PC-B	P-24-SLIM	C	B-2 (Note 4)
Am29806DC	Am29809DC	D-24-SLIM	C	C-1
Am29806DC-B	Am29809DC-B	D-24-SLIM	C	B-2 (Note 4)
Am29806DM	Am29809DM	D-24-SLIM	M	C-3
Am29806DM-B	Am29809DM-B	D-24-SLIM	M	B-3
Am29806LC	Am29809LC	L-28-1	C	C-1
Am29806LC-B	Am29809LC-B	L-28-1 to be	C	B-2 (Note 4)
Am29806LM	Am29809LM	L-28-1 announced	M	C-3
Am29806LM-B	Am29809LM-B	L-28-1	M	B-3
Am29806XC	Am29809XC	Dice	C	Visual Insp to MIL-STD-883 Method 20108
Am29806XL	Am29809XL	Dice	M	

Notes: 1. D = Hermetic DIP, L = Chip-Pak, Number following letter is number of leads.
2. C = 0 to 70°C, V_{CC} = 4.5 to 5.5V, M = −55 to 125°C, V_{CC} = 4.50 to 5.50V.
3. Levels C-1 and C-3 conform to MIL-STD-883, Class C. Level B-3 conforms to MIL-STD-883, Class B.
4. 160 hour burn-in.

Multibus is a trademark of Intel Corporation. IMOX is a trademark of Advanced Micro Devices, Inc.

Am29818

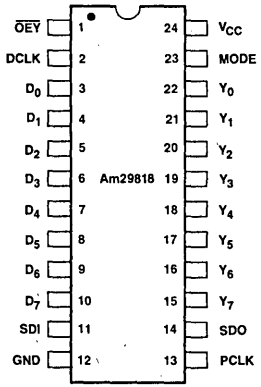
SSR™ Diagnostics/WCS Pipeline Register

ADVANCED INFORMATION

DISTINCTIVE CHARACTERISTICS

- High-speed noninverting 8-bit parallel register for any data path or pipelining application
- High-speed 8-bit "shadow register" with serial shift mode for Serial Shadow Register (SSR) Diagnostics
  - Controllability: serial scan in new machine state
  - Observability: serial scan out diagnostics routine results
- WCS (Writable Control Store) pipeline register
  - Load WCS from serial register
  - Read WCS via serial scan
- IMOX™ high performance IMplanted OXide isolated process
- 24-pin, 0.3" space saving package
- Alternate sourced as SN54/74S818

CONNECTION DIAGRAM



ABL-013

Leadless Chip Carrier
L-28-1

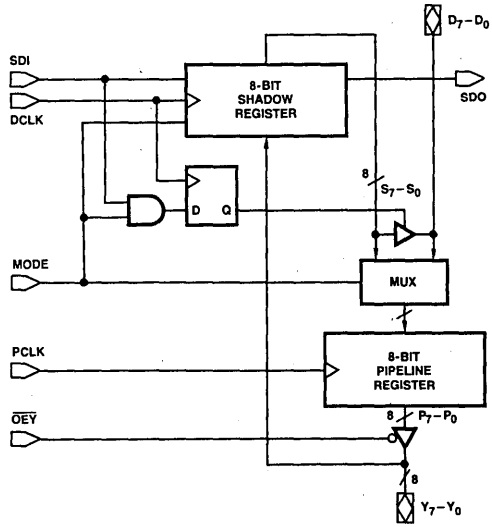
GENERAL DESCRIPTION

The Am29818 is a high-speed, general-purpose pipeline register with an on-board shadow register for performing Serial Shadow Register (SSR) Diagnostics and/or Writable Control Store loading.

The D-to-Y path provides an 8-bit parallel data path pipeline register for *normal* system operation. The shadow register can load parallel data to or from the pipeline register and can output data through the D input port (as in WCS loading).

The 8-bit shadow register has multiplexer inputs that select parallel inputs from the Y-port or adjacent bits in the shadow register to operate as a right-shift-only shift register. This register can then participate in a serial loop throughout the system where normal data, address, status and control registers are replaced with Am29818 Diagnostic Pipeline Registers. The loop can be used to scan in a complete test routine starting point (data, address, etc.). Then after a specified number of machine cycles scan out the results to be inspected for the expected results. WCS loading can be accomplished using the same technique. An instruction word can be serially shifted into the shadow register and written into the WCS RAM by enabling the D output.

BLOCK DIAGRAM



ABL-014

IMOX and SSR are trademarks of Advanced Micro Devices, Inc.

This document contains information on a product under development at Advanced Micro Devices, Inc. The information is intended to help you to evaluate this product. AMD reserves the right to change or discontinue work on this proposed product without notice.

Am29821/822 • Am29823/824 • Am29825/826

High Performance Bus Interface Registers

DISTINCTIVE CHARACTERISTICS

- High-speed parallel registers with positive edge-triggered D-type flip-flops
  - Noninverting CP-Y t_{PD} = 7.5ns typ
  - Inverting CP-Y t_{PD} = 7.5ns typ
- Buffered common Clock Enable ($\overline{EN}$)
- Buffered common asynchronous Clear input ($\overline{CLR}$)
- Three-state outputs glitch free during power-up and down
- Outputs have Schottky clamp to ground
- 48mA Commercial I_{OL}, 32mA MIL I_{OL}
- Low input/output capacitance
  - 6pF inputs (typical)
  - 8pF outputs (typical)
- Metastable "Hardened" Registers
- I_{OH} specified at 2.0V and 2.4V
- 24-pin 0.3" space saving package
- IMOX™ high performance IMplanted OXide isolated process

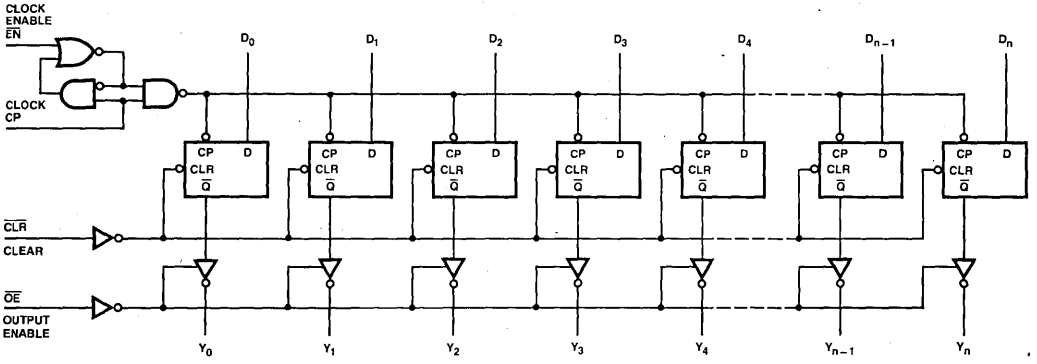
FUNCTIONAL DESCRIPTION

The Am29820 Series bus interface registers are designed to eliminate the extra packages required to buffer existing registers and provide extra data width for wider address/data paths or buses carrying parity. The Am29821 and Am29822 are buffered, 10-bit wide versions of the popular '374/'534 functions. The Am29823 and Am29824 are 9-bit wide buffered registers with Clock Enable ($\overline{EN}$) and Clear ($\overline{CLR}$) – ideal for parity bus interfacing in high performance microprogrammed systems. The Am29825 and Am29826 are 8-bit buffered registers with all the '823/4 controls plus multiple enables ($\overline{OE_1}$, $\overline{OE_2}$, $\overline{OE_3}$) to allow multiuser control of the interface, e.g., $\overline{CS}$, DMA, and RD/$\overline{WR}$. They are ideal for use as an output port requiring high I_{OL}/I_{OH}.

All of the Am29800 high performance interface family are designed for high capacitance load drive capability while providing low capacitance bus loading at both inputs and outputs. All inputs are Schottky diode inputs, and all outputs are designed for low capacitance bus loading in the high impedance state.

	Device		
	10-Bit	**9-Bit**	**8-Bit**
Noninverting	Am29821	Am29823	Am29825
Inverting	Am29822	Am29824	Am29826

LOGIC DIAGRAM



LBI-001

IMOX is a trademark of Advanced Micro Devices, Inc.

Am29827/Am29828

High Performance Buffers

ADVANCED INFORMATION

DISTINCTIVE CHARACTERISTICS

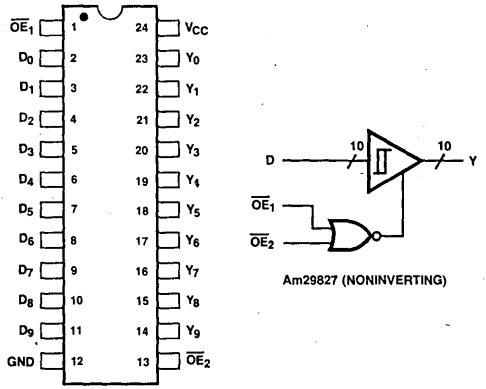
- High-speed buffers and inverters
  - Noninverting t_{PD} = 4.5ns typ
  - Inverting t_{PD} = 4.0ns typ
- 200mV minimum input hysteresis on input data ports
- Three-state outputs glitch-free during power-up and -down
- Outputs have Schottky clamp to ground
- 48mA commercial I_{OL}, 32mA military I_{OL}
- High capacitance load capability
- Low capacitance inputs and outputs
- I_{OH} specified 2.0V and 2.4V
- 24-pin 0.3" space saving package
- Fully TTL compatible inputs and outputs
- IMOX™ high performance IMplanted OXide isolated process

FUNCTIONAL DESCRIPTION

The Am29827 and Am29828 10-bit bus buffers provide high performance bus interface buffering for wide data/address paths or buses carrying parity. The 10-bit buffers have NOR-ed output enables for maximum control flexibility. All buffer data inputs have 200mV minimum input hysteresis to provide improved noise rejection.

All of the Am29800 high performance interface family are designed for high capacitance load drive capability while providing low capacitance bus loading at both inputs and outputs. All inputs are Schottky diode inputs, and all outputs are designed for low capacitance bus loading in the high impedance state.

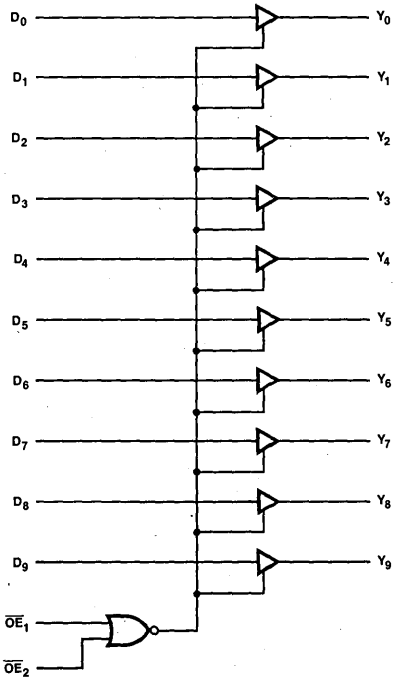
Am29827/Am29828
10-BIT BUS DRIVERS



Am29827 (NONINVERTING)

ABI-126

Am29827/Am29828
10-BIT BUFFERS



ABI-127

IMOX is a trademark of Advanced Micro Devices, Inc.

Am29833/34 • Am29853/54

Parity Bus Transceivers

PRELIMINARY

DISTINCTIVE CHARACTERISTICS

- High-speed bidirectional bus transceiver for processor organized devices
- Error flag with open-collector output
- Generates odd parity for all-zero protection
- Buffered direction three-state control
- Separate Transmit and Receive enable controls
- Output short-circuit protected to V_{CC} limits
- 200mV minimum input hysteresis on input data ports
- High-capacitance drive capability
  - 48mA commercial I_{OL}
  - 32mA military I_{OL}
- 24-pin 0.3″ slim DIP package

FUNCTIONAL DESCRIPTION

The Am29833/34/53/54 are high-performance bus transceivers designed for two-way communications. They each contain an 8-bit data path from the A (port) to the B (port), a 9-bit data path from the B (port) to the A (port), and a 9-bit parity checker/generator. Two options are available. The Am29833/34 register option, and the Am29853/54 latch option. With the register option, the error flag can be clocked and stored in a register and read at the open-collector $\overline{ERR}$ output. The clear ($\overline{CLR}$) input is used to clear the error flag register. With the latch option, the error can be either passed, stored, sampled or cleared at the error flag output by using the $\overline{EN}$ and $\overline{CLR}$ controls.

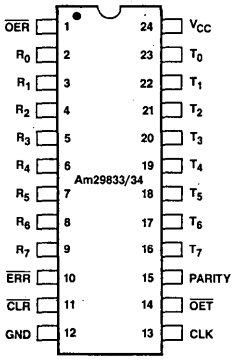
The output enables $\overline{OET}$ and $\overline{OER}$ are used to force the port outputs to the high-impedance state so that the device can drive bus lines directly. In addition, the $\overline{OER}$ and $\overline{OET}$ can be used to force a parity error by enabling both lines simultaneously. This transmission of inverted parity gives the designer more system diagnostic capability. The Am29833 and Am29853 are noninverting, while the Am29834 and Am29854 present inverting data at the outputs. The devices are specified at 48mA output sink current over the commercial range and 32mA over the military range.

CONNECTION DIAGRAMS – Top Views

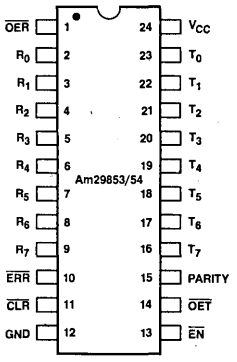
Leadless Chip Carrier
L-28-1

ABI-105

8-BIT TO 9-BIT PARITY TRANSCEIVERS



ABI-106



ABI-107

ORDERING INFORMATION

Order the part number according to the table below to obtain the desired package, temperature range, and screening level.

Order Number	Package Type (Note 1)	Operating Range (Note 2)	Screening Level (Note 3)
AM2983_/5_DC	D-24-SLIM	C	C-1
AM2983_/5_DCB	D-24-SLIM	C	B-2 (Note 4)
AM2983_/5_DM	D-24-SLIM	M	C-3
AM2983_/5_DMB	D-24-SLIM	M	B-3
AM2983_/5_LC	L-28-1	C	C-1
AM2983_/5_LCB	L-28-1	C	B-2 (Note 4)
AM2983_/5_LM	L-28-1	M	C-3
AM2983_/5_LMB	L-28-1	M	B-3
AM2983_/5_XC	Dice	C	Visual inspection to MIL-STD-883 Method 2010B.
AM2983_/5_XM	Dice	M	

Notes:
1. D = Hermetic DIP, L = Chip-Pak. Number following letter is number of leads.
2. C = 0 to +70°C, V_{CC} = 4.75 to 5.25V, M = −55 to +125°C, V_{CC} = 4.50 to 5.50V.
3. Levels C-1 and C-3 conform to MIL-STD-883, Class C. Level B-3 conforms to MIL-STD-883, Class B.
4. 160 hour burn-in.

Am29841/842 • Am29843/844 • Am29845/846

High Performance Bus Interface Latches

DISTINCTIVE CHARACTERISTICS

- High-speed parallel latches
  - Noninverting transparent t_{PD} = 5.25ns typ
  - Inverting transparent t_{PD} = 6.0ns typ
- Buffered common latch enable, clear and preset input
- Three-state outputs glitch-free during power-up and down
- Outputs have Schottky clamp to ground
- 48mA Commercial I_{OL}, 32mA MIL I_{OL}
- Low input/output capacitance
  - 6pF inputs (typical)
  - 8pF outputs (typical)
- I_{OH} specified 2.0V and 2.4V
- 24-pin 0.3" space saving package
- Fully TTL compatible inputs and outputs
- IMOX™ high performance IMplanted OXide isolated process

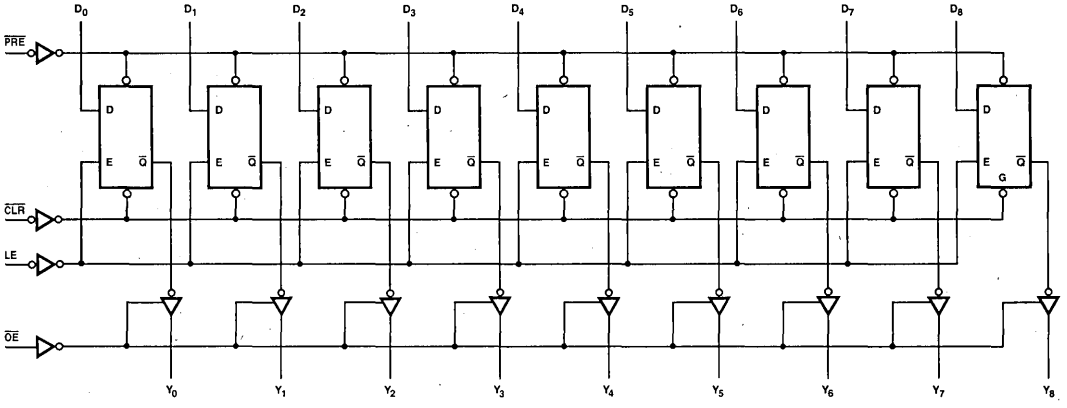
FUNCTIONAL DESCRIPTION

The Am29840 Series bus interface latches are designed to eliminate the extra packages required to buffer existing latches and provide extra data width for wider address/data paths or buses carrying parity. The Am29841 and Am29842 are buffered, 10-bit wide versions of the popular '373 function. The Am29843 and Am29844 are 9-bit wide buffered latches with Preset ($\overline{PRE}$) and Clear ($\overline{CLR}$) – ideal for parity bus interfacing in high performance systems. The Am29845 and Am29846 are 8-bit buffered latches with all the '843/4 controls plus multiple enables ($\overline{OE}_1$, $\overline{OE}_2$, $\overline{OE}_3$) to allow multiuser control of the interface, e.g., $\overline{CS}$, DMA, and RD/$\overline{WR}$. They are ideal for use as an output port requiring high I_{OL}/I_{OH}.

All of the Am29800 high performance interface family are designed for high capacitance load drive capability while providing low capacitance bus loading at both inputs and outputs. All inputs are Schottky diode inputs, and all outputs are designed for low capacitance bus loading in the high impedance state.

	Device		
	10-Bit	**9-Bit**	**8-Bit**
Noninverting	Am29841	Am29843	Am29845
Inverting	Am29842	Am29844	Am29846

LOGIC DIAGRAM
Am29843



ABI-010

IMOX is a trademark of Advanced Micro Devices, Inc.

Am29861 • Am29862
Am29863 • Am29864

High Performance Bus Transceivers
ADVANCED INFORMATION

DISTINCTIVE CHARACTERISTICS

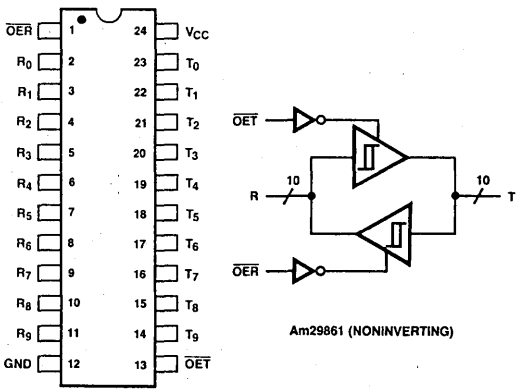
- High-speed symmetrical bidirectional transceivers
  - Noninverting t_{PD} = 4.5ns typ
  - Inverting t_{PD} = 4.0ns typ
- 200mV minimum input hysteresis on input data ports
- Three-state outputs glitch-free during power-up and -down
- Outputs have Schottky clamp to ground
- 48mA commercial I_{OL}, 32mA military I_{OL}
- Low input/output capacitance
  - 6pF inputs (typical)
  - 8pF outputs (typical)
- I_{OH} specified 2.0V and 2.4V
- 24-pin 0.3″ space saving package
- Fully TTL compatible inputs and outputs
- IMOX™ high performance IMplanted OXide isolated process

FUNCTIONAL DESCRIPTION

The Am29860 Series bus transceivers provide high performance bus interface buffering for wide data/address paths or buses carrying parity. The Am29863/64 9-bit transceivers have NOR-ed output enables for maximum control flexibility. All transceiver data inputs have 200mV minimum input hysteresis to provide improved noise rejection.

All of the Am29800 high performance interface family are designed for high capacitance load drive capability while providing low capacitance bus loading at both inputs and outputs. All inputs are Schottky diode inputs, and all outputs are designed for low capacitance bus loading in the high impedance state.

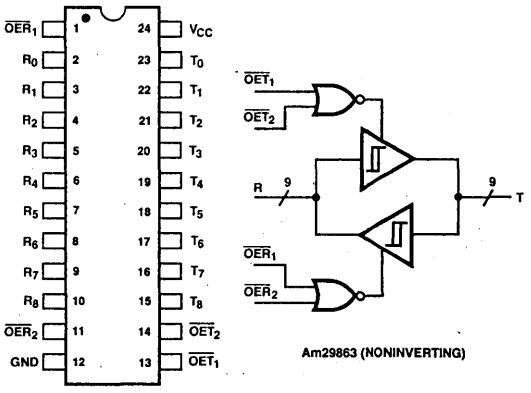
Am29861/Am29862
10-BIT TRANSCEIVERS



Am29861 (NONINVERTING)

ABI-092

Am29863/Am29864
9-BIT TRANSCEIVERS



Am29863 (NONINVERTING)

ABI-093

IMOX is a trademark of Advanced Micro Devices, Inc.

This document contains information on a product under development at Advanced Micro Devices, Inc. The information is intended to help you to evaluate this product. AMD reserves the right to change or discontinue work on this proposed product without notice.

Am8120

Octal D-Type Flip-Flop with Clear, Clock Enable and Three-State Control

DISTINCTIVE CHARACTERISTICS

- Buffered common clock enable input
- Buffered common asynchronous clear input
- Three-state outputs
- 8-bit, high-speed parallel register with positive edge-triggered, D-type flip-flops

RELATED PRODUCTS

Part No.	Description
Am25S18	Quad D Register
Am2920	Octal D Type Flip-Flop
Am2954/5	Octal D Registers

FUNCTIONAL DESCRIPTION

The Am8120 is an 8-bit register built using advanced Low-Power Schottky technology. The register consists of eight D-type flip-flops with a buffered common clock, a buffered common clock enable, a buffered asynchronous clear input, and three-state outputs.

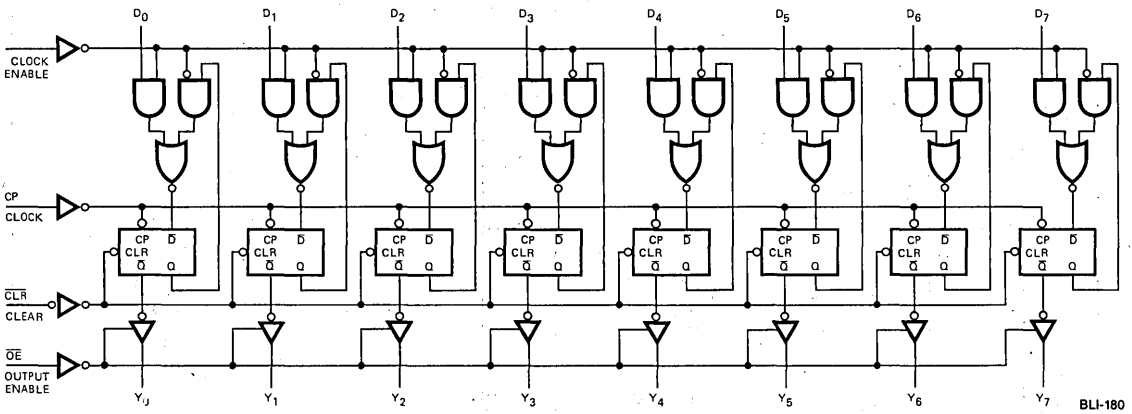
When the clear input is LOW, the internal flip-flops of the register are reset to logic 0 (LOW), independent of all other inputs. When the clear input is HIGH, the register operates in the normal fashion.

When the three-state output enable ($\overline{OE}$) input is LOW, the Y outputs are enabled and appear as normal TTL outputs. When the output enable ($\overline{OE}$) input is HIGH, the Y outputs are in the high impedance (three-state) condition. This does not affect the internal state of the flip-flop Q output.

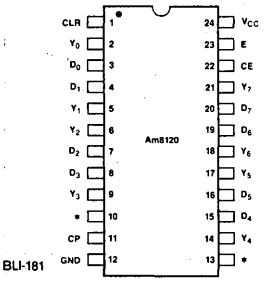
The clock enable input ($\overline{E}$) is used to selectively load data into the register. When the $\overline{E}$ input is HIGH, the register will retain its current data. When the $\overline{E}$ is LOW, new data is entered into the register on the LOW-to-HIGH transition of the clock input.

This device is packaged in a slim 24-pin package (0.3 inch row spacing).

LOGIC DIAGRAM

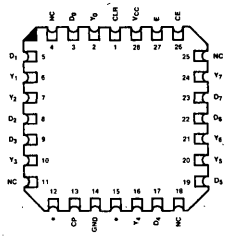


CONNECTION DIAGRAMS — Top Views



Note: Pin 1 is marked for orientation.

Leadless Chip Carrier
L-28-1



*Reserved — do not use.

ORDERING INFORMATION

Package Type	Temperature Range	Order Number
Molded DIP	0 to +70°C	AM8120PC
Hermetic DIP	0 to +70°C	AM8120DC
Chip-Pak	0 to +70°C	AM8120LC
Dice	0 to +70°C	AM8120XC
Hermetic DIP	−55 to +125°C	AM8120DM
Hermetic Flat Pack	−55 to +125°C	AM8120FM
Chip-Pak	−55 to +125°C	AM8120LM
Dice	−55 to +125°C	AM8120XM

Am8163 • Am8167

Dynamic Memory Timing, Refresh and EDC Controllers

DISTINCTIVE CHARACTERISTICS

- Complete CPU to dynamic RAM control interface
- $\overline{RAS}$/MSEL/$\overline{CAS}$ Sequencer to eliminate delay lines
- Memory request/refresh arbitration
- Complete EDC/data path controls for Word/Byte read or write
- Automatic write-back of corrected data and check bits when single errors are detected on any read cycle
- Refresh interval timer independent of CPU
- Refresh control during Single-Step or Halt modes
- EDC error flag latches for error logging under software control
- Two timing configurations support a broad range of processors (Z80, Z8000, 8086, 8088, MC68000)

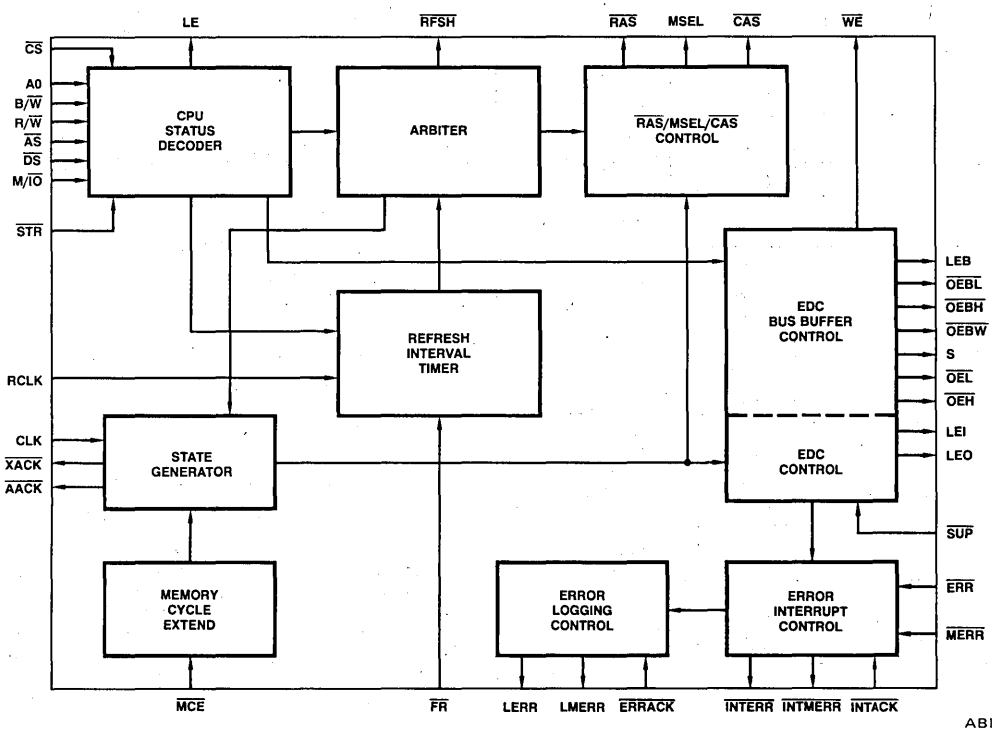
GENERAL DESCRIPTION

The Am8163 and Am8167 are high speed bus interface controllers forming an integral part of the 8086 and AmZ8000* memory support chip set using dynamic MOS RAMs with Error Detection and Correction (EDC). The complete chip set includes the Am8284A and AmZ8127 Clock Generators, the Am2964B Dynamic Memory Controller, the Am2961/62 EDC Bus Buffers, the Am2960 EDC Unit and Am2965/66 RAM Drivers.

The Am8163 and Am8167 provide all of the control interface functions including $\overline{RAS}$/Address-MUX/$\overline{CAS}$ timing (without delay lines), refresh timing, memory request/refresh arbitration and all EDC enables and controls. The enable controls are configured for both word and byte operations including the data controls for byte write with error correction. The Am8163/7 generates bus and operating mode controls for the Am8160 EDC Unit.

The Am8163/7 uses the AmZ8127 oscillator output to generate $\overline{RAS}$/Address MUX/$\overline{CAS}$ timing. An internal refresh interval timer generates the memory refresh request independent of the CPU to guarantee the proper refresh timing under all combinations of CPU and DMA memory requests.

BLOCK DIAGRAM



ABI-055

*Z8000 is a trademark of Zilog.

Am8212

Eight-Bit Input/Output Port

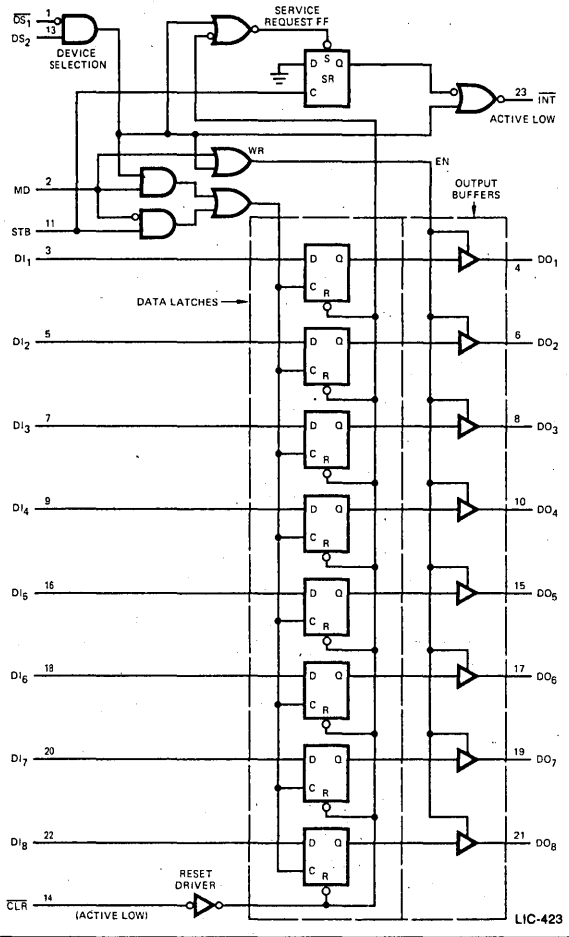
Distinctive Characteristics

- Fully parallel, 8-bit data register and buffer replacing latches, multiplexers and buffers needed in microprocessor systems.
- 4.0V output high voltage for direct interface to MOS microprocessors, such as the Am9080A family.
- Input load current 250μA max.
- Reduces system package count
- Available for operation over both commercial and military temperature ranges.
- Service request flip-flop for interrupt generation
- Three-state outputs sink 15mA
- Asynchronous register clear with clock over-ride

FUNCTIONAL DESCRIPTION

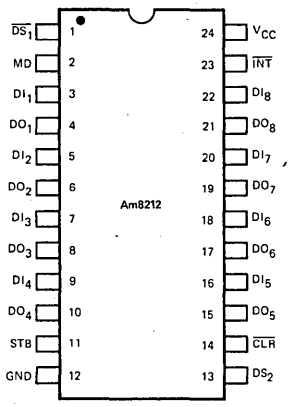
All of the principal peripheral and input/output functions of a Microcomputer System can be implemented with the Am8212. The Am8212 input/output port consists of an 8-latch with 3-state output buffers along with control and device selection logic, which can be used to implement latches, gated buffers or multiplexers.

LOGIC DIAGRAM



LIC-423

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

LIC-424

PIN DEFINITION

$DI_1 - DI_8$	DATA IN
$DO_1 - DO_8$	DATA OUT
$\overline{DS_1} - DS_2$	DEVICE SELECT
MD	MODE
STB	STROBE
$\overline{INT}$	INTERRUPT (ACTIVE LOW)
$\overline{CLR}$	CLEAR (ACTIVE LOW)

ORDERING INFORMATION

Package Type	Temperature Range	Order Number
Hermetic DIP	–55°C to +125°C	AM8212DM
Hermetic DIP	0°C to +70°C	D8212
Molded DIP	0°C to +70°C	P8212
Dice	0°C to +70°C	AM8212XC

Am8216 • Am8226

Four-Bit Parallel Bidirectional Bus Driver

Distinctive Characteristics

- Data bus buffer driver for 8080 type CPU's
- Low input load current – 0.25mA maximum
- High output drive capability for driving system data bus – 50mA at 0.5V
- Am8216 has non-inverting outputs
- Output high voltage compatible with direct interface to MOS
- Three-state outputs
- Advanced Schottky processing
- Available in military and commercial temperature range
- Am8226 has inverting outputs

FUNCTIONAL DESCRIPTION

The Am8216 and Am8226 are four-bit, bi-directional bus drivers for use in bus oriented applications. The non-inverting Am8216, and inverting Am8226 drivers are provided for flexibility in system design.

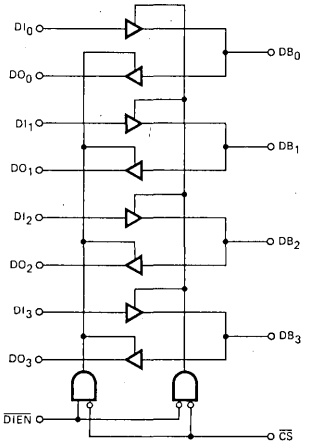
Each buffered line of the four bit driver consists of two separate buffers that are three-state to achieve direct bus interface and bi-directional capability. On one side of the driver the output of one buffer and the input of another are tied together (DB), this side is used to interface to the system side components such as memories, I/O, etc., because its interface is TTL compatible and it has high drive (50mA). On the other side of the driver the inputs and outputs are separated to provide maximum flexibility. Of course, they can be tied together so that the driver can be used to buffer a true bi-directional bus. The DO outputs on this side of the driver have a special high voltage output drive capability so that direct interface to the 8080 type CPUs is achieved with an adequate amount of noise immunity.

The $\overline{CS}$ input is a device enable. When it is "high" the output drivers are all forced to their high-impedance state. When it is a "LOW" the device is enabled and the direction of the data flow is determined by the $\overline{DIEN}$ input.

The $\overline{DIEN}$ input controls the direction of data flow which is accomplished by forcing one of the pair of buffers into its high impedance state and allowing the other to transmit its data. A simple two gate circuit is used for this function.

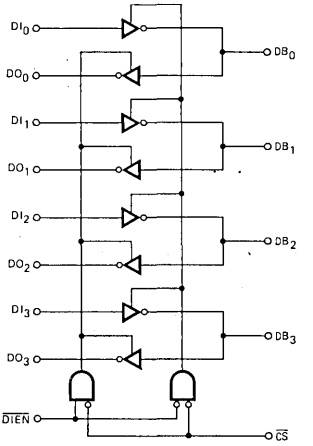
LOGIC DIAGRAMS

Am8216



LIC-439

Am8226



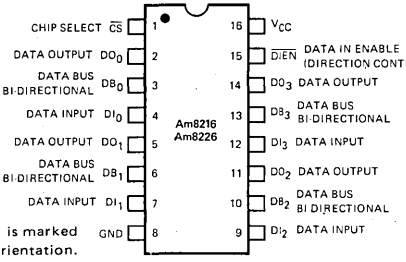
LIC-440

ORDERING INFORMATION

Package Type	Temperature Range	Am8216 Order Number	Am8226 Order Number
Hermetic DIP	–55°C to +125°C	MD8216	MD8226
Hermetic DIP	0°C to +70°C	D8216	D8226
Molded DIP	0°C to +70°C	P8216	P8226
Dice	0°C to +70°C	AM8216XC	AM8226XC

CONNECTION DIAGRAM

Top View



Note: Pin 1 is marked for orientation.

LIC-441

Am8224

Clock Generator and Driver for 8080A Compatible Microprocessors

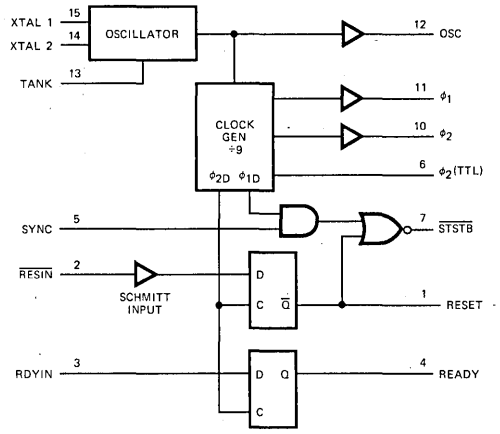
Distinctive Characteristics

- Single chip clock generator/driver for 8080A compatible CPU
- Power-up reset for CPU
- Ready synchronizing flip-flop
- Status strobe signal
- Oscillator output for external system timing
- Am8224-4 version available for use with 1μsec instruction cycle of Am9080A-4
- Available for operation over both commercial and military temperature ranges
- Crystal controlled for stable system operation
- Reduces system package count
- Advanced Schottky processing

FUNCTIONAL DESCRIPTION

The Am8224 is a single chip Clock Generator/Driver for the Am9080A and 8080A CPU. It contains a crystal-controlled oscillator, a "divide by nine" counter, two high-level drivers and several auxiliary logic functions, including a power-up reset, status strobe and synchronization of ready. Also provided are TTL compatible oscillator and ϕ_2 outputs for external system timing. The Am8224 provides the designer with a significant reduction of packages used to generate clocks and timing for the Am9080A or 8080A for both commercial and military temperature range applications. A high speed version, the Am8224-4, is available for use with the high speed Am9080A-4.

LOGIC DIAGRAM



LIC-619

ORDERING INFORMATION

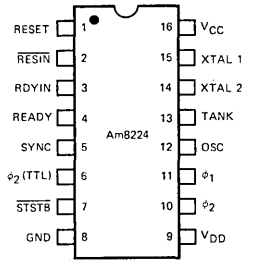
Package Type	Temperature Range	Order Number
Hermetic DIP	−55°C to +125°C	AM8224DM
Hermetic DIP	0°C to +70°C	D8224
Molded DIP	0°C to +70°C	AM8224PC
Dice	0°C to +70°C	AM8224XC
Hermetic DIP	0°C to +70°C	AM8224-4DC*

*For use with Am9080A-4 with clock period between 250ns and 320ns.

PIN DEFINITION

XTAL 1	CONNECTIONS FOR CRYSTAL
XTAL 2	
TANK	USED WITH OVERTONE XTAL
OSC	OSCILLATOR OUTPUT
ϕ_2(TTL)	ϕ_2 CLK (TTL LEVEL)
V_{CC}	+5.0V
V_{DD}	+12V
GND	0V
$\overline{RESIN}$	RESET INPUT
RESET	RESET OUTPUT
RDYIN	READY INPUT
READY	READY OUTPUT
SYNC	SYNC INPUT
$\overline{STSTB}$	STATUS STB (ACTIVE LOW)
ϕ_1	Am9080A/8080A CLOCKS
ϕ_2	

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

LIC-620

Am8228 • Am8238

System Controller and Bus Driver for 8080A Compatible Microprocessors

Distinctive Characteristics

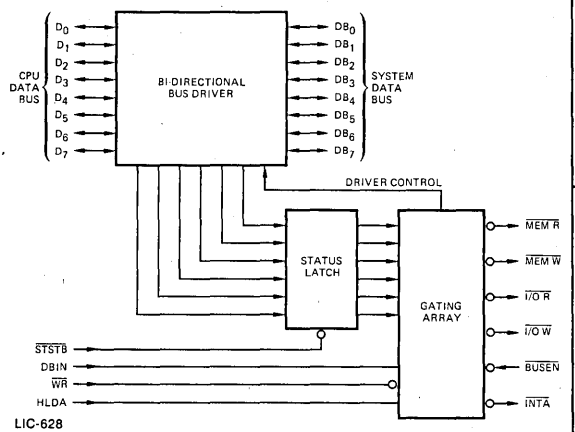
- Multi-byte instruction interrupt acknowledge
- Selectable single level vectored interrupt (RST-7)
- 28-pin molded or hermetic DIP package
- Single chip system controller and data bus driver for Am9080/8080A systems
- Am8238-4 high speed version available for use with 1μsec instruction cycle of Am9080A-4
- Bi-directional three-state bus driver for CPU independent operation
- Advanced low-power Schottky processing
- Available in military and commercial temperature range
- Am8238 has extended $\overline{IOW}/\overline{MEMW}$ pulse width

FUNCTIONAL DESCRIPTION

The Am8228 and Am8238 are single chip System Controller Data Bus drivers for the Am9080A Microcomputer System. They generate all control signals required to directly interface Am9080A/8080A compatible system circuits (memory and I/O) to the CPU.

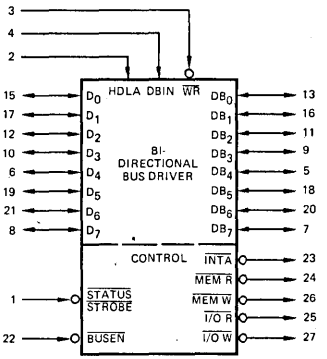
Bi-directional bus drivers with three-state outputs are provided for the system data bus, facilitating CPU independent bus operations such as direct memory access. Interrupt processing is accommodated by means of a single vectored interrupt or by means of the standard 8080A multiple byte interrupt vector operation.

LOGIC DIAGRAM



LIC-628

LOGIC SYMBOL



V_{CC} = Pin 28
GND = Pin 14

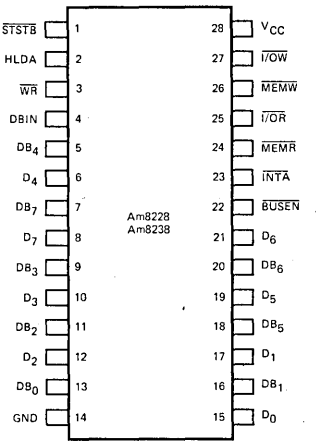
LIC-629

ORDERING INFORMATION

Package Type	Temperature Range	Am8228 Order Number	Am8238 Order Number
Molded DIP	0°C to +70°C	AM8228PC	AM8238PC
Hermetic DIP	0°C to +70°C	D8228	D8238
Hermetic DIP	−55°C to +125°C	AM8228DM	AM8238DM
Dice	0°C to +70°C	AM8228XC	AM8238XC
Hermetic DIP	0°C to +70°C		AM8238-4DC*
Molded DIP	0°C to +70°C		AM8238-4PC*

*For use with Am9080A-4 with minimum clock period of 250ns.

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation.

LIC-630

AMD 20-Pin PAL* Family

20-Pin IMOX™ Programmable Array Logic Elements

(See AMD Programmable Array Logic Handbook – 1983 Edition for details)

DISTINCTIVE CHARACTERISTICS

- **Fast**
  - High speed "A" versions (t_{pd} = 25ns, t_s = 20ns, t_{co} = 15ns, max)
  - Standard speed versions (t_{pd} = 35ns, t_s = 30ns, t_{co} = 25ns, max)
- **Flexible**
  - User programmability allows customized designs
  - Eases design updates in prototype or product
- **Low Cost**
  - Reduces board space/chip count
  - Reduces design time
  - Reduces inventory cost
- **Reliable**
  - Proven Platinum-Silicide fuse technology
  - Fully AC and DC tested
  - Preload of output registers allows full logical testing

FUNCTIONAL DESCRIPTION

AMD PALs are high speed electrically programmable array logic elements. They utilize the familiar sum-of-products (AND-OR) structure allowing users to program custom logic functions to fit most applications precisely.

Initially the AND gates are connected, via fuses, to both the true and complement of every input. By selective programming of fuses the AND gates may be "connected" to only the true input (by blowing the complement fuse), to only the complement input (by blowing the true fuse), or to neither type of input (by blowing both fuses) establishing a logical "don't care." When both the true and complement fuses are left intact a logical false results on the output of the AND gate. An AND gate with all fuses blown will assume the logical true state. The outputs of the AND gates are connected to fixed OR gates. The only limitations imposed are the number of inputs to the AND gates (up to 16) and the number of AND gates per OR (up to 8).

The part types in the AMD PAL family are differentiated by the allocation of registered (with internal feedback) and combinatorial (bi-directional and dedicated) outputs. All combinatorial AMD PALs are available in both active HIGH (AND-OR) and active LOW (AND-OR-INVERT) versions.

AMD PAL FAMILY CHARACTERISTICS

All members of the AMD PAL family have common electrical characteristics and programming procedures. All parts in this family are produced with a fusible link at each input to the AND gate array. Connections may be selectively removed by applying appropriate voltages to the circuit.

All parts are fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are pre-programmed during manufacturing to insure extremely high field programming yields (>98%), and provide extra test paths to achieve excellent parametric correlation.

Platinum-Silicide was selected as the fuse link material to achieve a well controlled melt rate resulting in large non-conductive gaps that ensure very stable, long term reliability. Extensive operating testing has proven that this low-field, large-gap technology offers the best reliability for fusible link programmable logic.

The AMD PAL family is manufactured using Advanced Micro Devices' selective oxidation process, IMOX. This advanced process permits an increase in density and a decrease in internal capacitance resulting in the fastest possible programmable logic devices.

The AMD PAL family also incorporates the unique capability of preloading the output registers during testing to any desired value. Preload is invaluable when testing the logical functionality of a programmed AMD PAL.

AMD PAL FAMILY TABLE

Part Number	Array Inputs	Logic	OE	Outputs	t_{pd} (MAX) STD	t_{pd} (MAX) A	t_s (MAX) STD	t_s (MAX) A	t_{co} (MAX) STD	t_{co} (MAX) A	
AmPAL16R8	(8) Dedicated (8) Feedback	(8) 8-Wide AND-OR	Dedicated	Registered Inverting	–	–	30	20	25	15	ns
AmPAL16R6	(8) Dedicated (6) Feedback (2) Bidirectional	(6) 8-Wide AND-OR	Dedicated	Registered Inverting	35	25	30	20	25	15	ns
		(2) 7-Wide AND-OR-INVERT	Programmable	Bidirectional							
AmPAL16R4	(8) Dedicated (4) Feedback (4) Bidirectional	(4) 8-Wide AND-OR	Dedicated	Registered Inverting	35	25	30	20	25	15	ns
		(4) 7-Wide AND-OR-INVERT	Programmable	Bidirectional							
AmPAL16L8	(10) Dedicated (6) Bidirectional	(8) 7-Wide AND-OR-INVERT	Programmable	(6) Bidirectional (2) Dedicated	35	25	–	–	–	–	ns
AmPAL16H8	(10) Dedicated (6) Bidirectional	(8) 7-Wide AND-OR	Programmable	(6) Bidirectional (2) Dedicated	35	25	–	–	–	–	ns
AmPAL16LD8	(10) Dedicated (6) Bidirectional	(8) 8-Wide AND-OR-INVERT	–	Dedicated	35	25	–	–	–	–	ns
AmPAL16HD8	(10) Dedicated (6) Bidirectional	(8) 8-Wide AND-OR	–	Dedicated	35	25	–	–	–	–	ns

*PAL is a registered trademark of Monolithic Memories, Inc.
IMOX is a trademark of Advanced Micro Devices, Inc.

AMD Half Power PAL* Family

Half Power 20-Pin IMOX™ Programmable Array Logic Elements

(See AMD Programmable Array Logic Handbook – 1983 Edition for details)

DISTINCTIVE CHARACTERISTICS

- **Low Power Dissipation**
  - 1/2 the power of standard PALs (I_{CC} = 60mA typ) Reduces power supply requirements and improves system reliability
- **High Performance**
  - Meets all standard power PAL AC specs (t_{pd} = 35ns max, t_s = 30ns max, t_{co} = 25ns max)
- **High Output Drive**
  - I_{OL} = 24mA Same as standard power devices
  - Drives buses directly (no special driver chips needed)
- **Plug-in Replacement for Standard PALs**
  - Meets ALL standard PAL specs at 1/2 the power
- **Excellent Programming Yield** (typ 98%)
  - Platinum Silicide fuses ensure high programming yield, fast programming and unsurpassed reliability
- **Improved Testability**
  - Preload feature permits full logical verification
- **Superior Quality**
  - Full AC and DC testing done at the factory utilizing special designed-in test features

GENERAL DESCRIPTION

These low power devices represent a breakthrough in PAL technology by meeting all the specifications of the standard power parts with only half the current requirements. They can directly replace the standard power PALs with no loss of performance, and they, like the standard devices, incorporate the PRELOAD feature essential for full logic verification during testing.

AMD Half Power PALs are high-speed Programmable Array Logic elements which utilize the familiar sum-of-products (AND-OR) logic structure which allows users to program custom logic functions to precisely fit their applications. They are a replacement for Low Power Schottky SSI/MSI logic circuits, reduce the chip count by more than 5 to 1 and greatly simplify prototyping and board layout.

Seven different device types are available, including both registered and combinatorial devices. In addition, the combinatorial devices are offered in both active HIGH (AND-OR) and active LOW (AND-OR-INVERT) versions. The low power dissipation means reduced power supply requirements and improved reliability, while the use of AMD's IMOX process permits extremely high operating speeds.

AMD HALF POWER PAL FAMILY TABLE

Part Number	Array Inputs	Logic	OE	Outputs	t_{pd} (Max) ns	t_s (Max) ns	t_{co} (Max) ns	I_{CC} (Max) mA
AmPAL16R8L	Eight Dedicated Eight Feedback	Eight 8-Wide AND-OR	Dedicated	Registered Inverting	–	30	25	90
AmPAL16R6L	Eight Dedicated Six Feedback	Six 8-Wide AND-OR	Dedicated	Registered Inverting	35	30	25	90
	Two Bidirectional	Two 7-Wide AND-OR-INVERT	Programmable	Bidirectional				
AmPAL16R4L	Eight Dedicated Four Feedback	Four 8-Wide AND-OR	Dedicated	Registered Inverting	35	30	25	90
	Four Bidirectional	Four 7-Wide AND-OR-INVERT	Programmable	Bidirectional				
AmPAL16L8L	Ten Dedicated Six Bidirectional	Eight 7-Wide AND-OR-INVERT	Programmable	Six Bidirectional Two Dedicated	35	–	–	80
AmPAL16H8L	Ten Dedicated Six Bidirectional	Eight 7-Wide AND-OR	Programmable	Six Bidirectional Two Dedicated	35	–	–	80
AmPAL16LD8L	Ten Dedicated Six Bidirectional	Eight 8-Wide AND-OR-INVERT	–	Dedicated	35	–	–	80
AmPAL16HD8L	Ten Dedicated Six Bidirectional	Eight 8-Wide AND-OR	–	Dedicated	35	–	–	80

*PAL is a registered trademark of Monolithic Memories, Inc.

*PAL is a registered trademark of Monolithic Memories, Inc. IMOX is a trademark of Advanced Micro Devices, Inc.

AMD PAL FAMILY CHARACTERISTICS

All members of the AMD PAL family have common electrical characteristics and programming procedures. All parts are produced with a fusible link at each input to the AND gate array, and connections may be selectively removed by applying appropriate voltages to the circuit.

Initially the AND gates are connected, via fuses, to both the true and complement of each input. By selective programming of fuses the AND gates may be "connected" to only the true input (by blowing the complement fuse), to only the complement input (by blowing the true fuse), or to neither type of input (by blowing both fuses) establishing a logical "don't care." When both the true and complement fuses are left intact a logical false results on the output of the AND gate, while all fuses blown results in a logical true state. The outputs of the AND gates are connected to fixed OR gates. The only limitations imposed are the number of inputs to the AND gates (up to 16) and the number of AND gates per OR (up to 8).

All parts are fabricated with AMD's fast programming, highly reliable Platinum-Silicide Fuse technology. Utilizing an easily implemented programming algorithm, these products can be rapidly programmed to any customized pattern. Extra test words are pre-programmed during manufacturing to insure extremely high field programming yields (>98%), and provide extra test paths to achieve excellent parametric correlation.

POWER-UP RESET

The registered devices in the AMD PAL family have been designed to reset during system power-up. Following power-up, all registers will be initialized to zero, setting all the outputs to a logic 1. This feature provides extra flexibility to the designer and is especially valuable in simplifying state machine initialization.

PRELOAD

AMD Low Power PALs are designed with unique PRELOAD circuitry that provides an easy method of testing registered devices for logical functionality. PRELOAD allows any arbitrary state value to be loaded into the PAL's output registers.

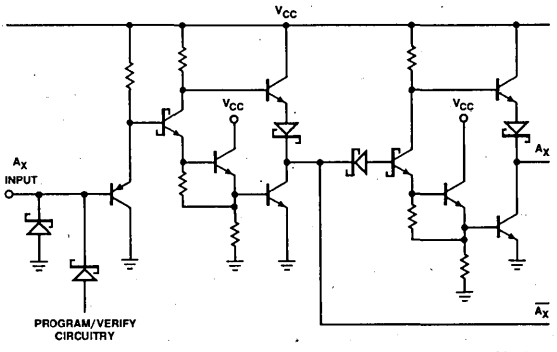
A typical functional test sequence would be to verify all possible state transitions for the device being tested. This requires the ability to set the state registers into an arbitrary "present state" value and to set the device inputs to any arbitrary "present input" value. Once this is done, the state machine is clocked into a new state, or "next state." The next state is then checked to validate the transition from the present state. In this way any state transition can be checked.

Without PRELOAD, it is difficult and in some cases impossible to load an arbitrary present state value. This can lead to logic verification sequences that are either incomplete or excessively long. Long test sequences result when the feedback from the state register "interferes" with the inputs, forcing the machine to go through many transitions before it can reach an arbitrary state value. Therefore the test sequence will be mostly state initialization and not actual testing. The test sequence becomes excessively long when a state must be reentered many times to test a wide variety of input combinations.

In addition, complete logic verification may become impossible when states that need to be tested can not be entered with normal state transitions. For example, even though necessary, the state entered when machine powers up can not be tested, because it can not be entered from the main sequence. Similarly, "forbidden" or don't care states that are not normaly entered need to be tested to ensure that they return to the main sequence.

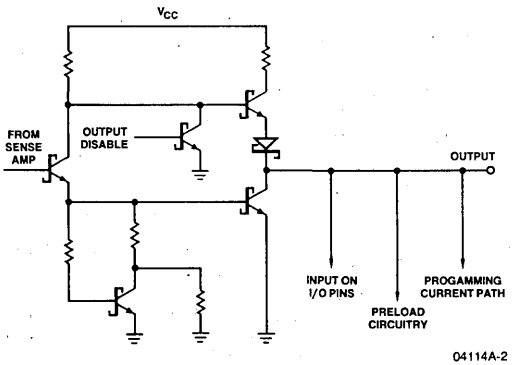
PRELOAD eliminates these problems by providing the capability to go directly to any desired arbitrary state. Thus test sequences may be greatly shortened, and all possible states can be tested, greatly reducing test time and development costs, and guaranteeing proper in-system operation.

INPUT CIRCUITRY



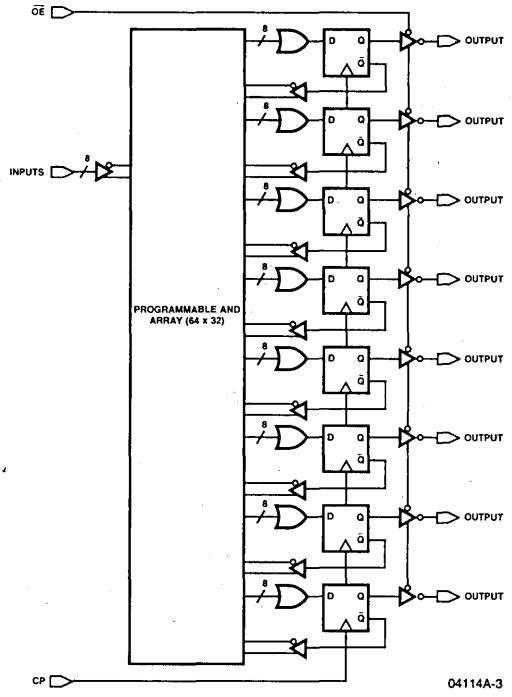
04114A-1

OUTPUT CIRCUITRY

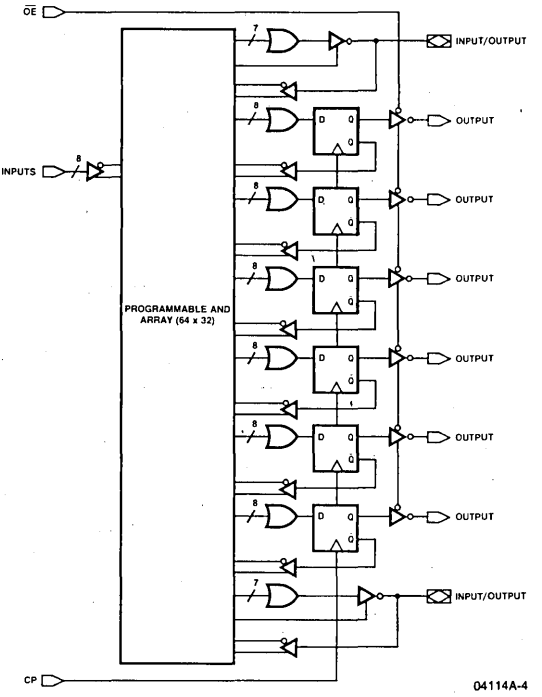


04114A-2

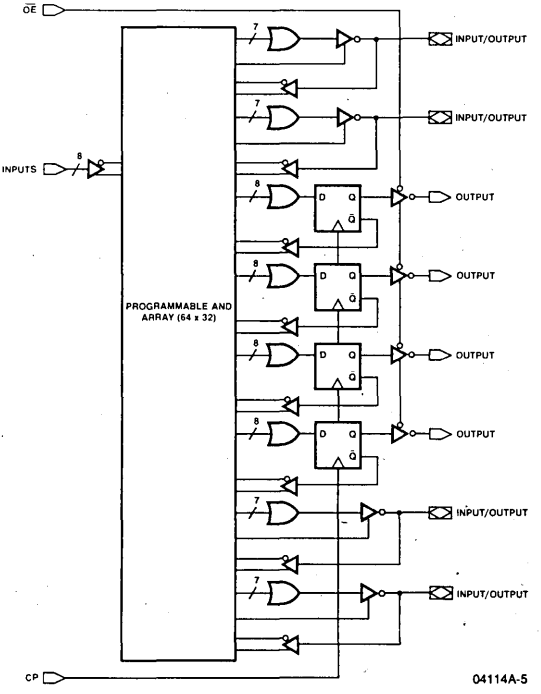
AmPAL16R8L
BLOCK DIAGRAM



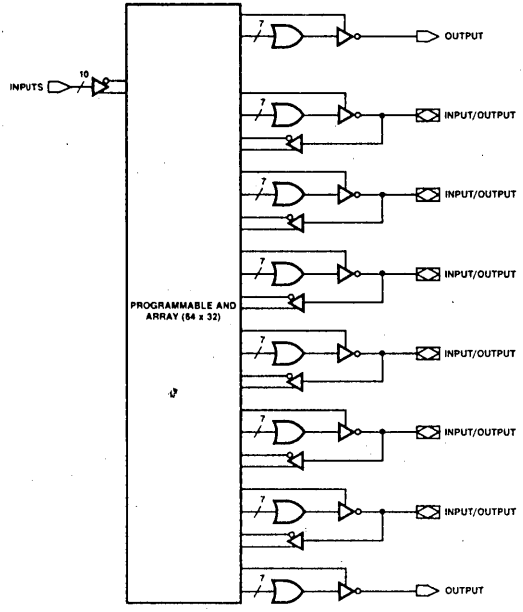
AmPAL16R6L
BLOCK DIAGRAM



AmPAL16R4L
BLOCK DIAGRAM

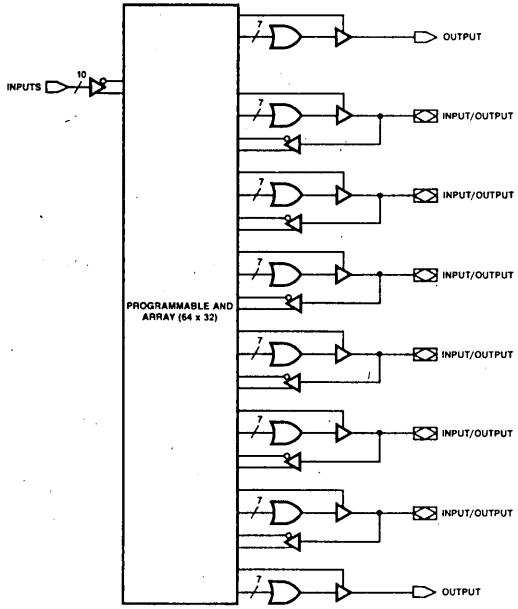


AmPAL16L8L
BLOCK DIAGRAM



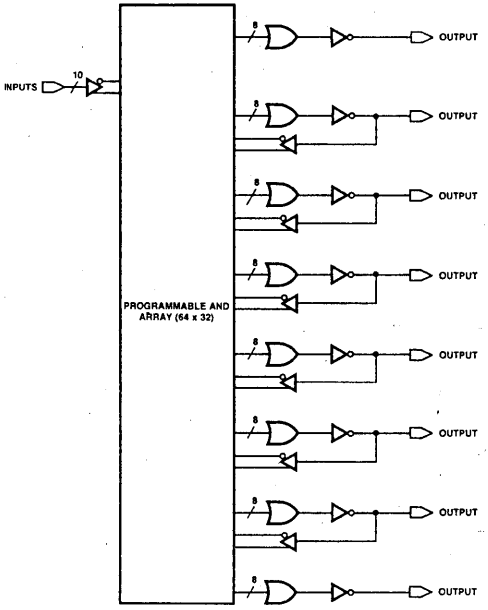
04114A-6

AmPAL16H8L
BLOCK DIAGRAM



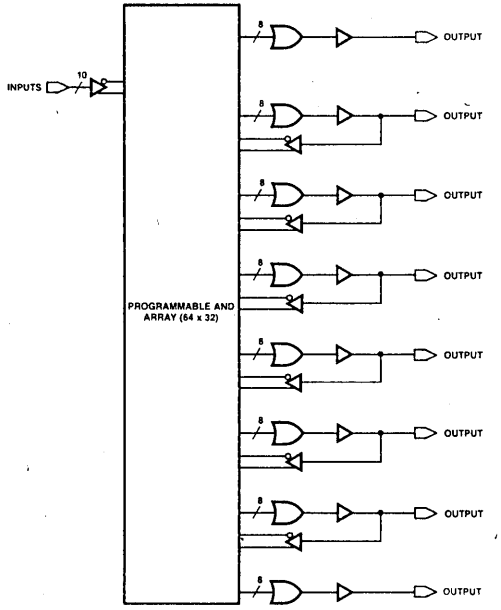
04114A-7

AmPAL16LD8L
BLOCK DIAGRAM



04114A-8

AmPAL16HD8L
BLOCK DIAGRAM



04114A-9

Advanced General Purpose Peripherals

Overview

In today's world of increasing demands placed on microcomputer – leased system applications, the "smart" high performance peripheral can offer significant advantages. In many cases these products can both expand the capabilities of the system and extend its life cycle without the extensive hardware and software changes required by a full system redesign.

Advanced Micro Devices' proprietary Am9500* Family of intelligent MOS/LSI peripheral products represent the perfect complement to an 8- or 16-bit microprocessor based system by providing maximum performance at lower cost. The AmZ8530 Serial Communications Controller, AmZ8038 FIFO I/O, Am9513 Programmable Interval Timer, Am9516 Universal DMA Controller and Am9519 Programmable Interrupt Controller have all been designed to interface easily with the major 8- and 16-bit CPU's, and are Z-bus, Multibus and Q-bus compatible with minimal external logic. AMD has included guides for each of these devices, which appear at the end of the relevant data sheets and which demonstrate methods for interfacing with the leading 16-bit microprocessors. For interfacing these devices to the Z8000, 8085A, Z80 and the 6800, refer to the AMD Am9500 Peripheral Products Interface Guide.

For broader guidelines on interfacing these devices to the 8086, Z8000 and the 6800, refer to the appendix at the end of this book.

*Included in the Am9500 Family are:

- The 9511A Arithmetic Processor.
- The 9512 Floating Point Processor.
- The 9513 System Timing Controller.
- The 9516 Direct Memory Access Controller (16-bit).
- The 9517A Direct Memory Access Controller.
- The 9518 Data Cyphering Processor.
- The 9519A Universal Interrupt Controller.
- The 9520 Burst Error Processor.
- The 9521 Burst Error Processor.
- The Am8052/8152 Advanced CRT Chip Set.

AmZ8030 • AmZ8530 (SCC)

Serial Communications Controller

DISTINCTIVE CHARACTERISTICS

- **Two 1M.bps full duplex serial channels**
  Each channel has independent oscillator, baud-rate generator, and PLL for clock recovery, dramatically reducing external components.
- **Programmable protocols**
  NRZ, NRZI, and FM data encoding supported under program control.
- **Programmable Asynchronous Modes**
  5 to 8 bit characters with prorammable stop bits, clock, break detect, and error conditions.
- **Programmable Synchronous Modes**
  SDLC and HDLC and SDLC loop supported with frame control, zero insertion and deletion, abort, and residue handling. CRC-16 and CCITT generators and checkers.
- **Z8000* compatible**
  The Z8030 interfaces directly to the Z8000 CPU bus and to the Z8000 interrupt structure.
- **Compatible with non-multiplexed bus**
  The Z8530 interfaces easily to most other CPUs.

See the "AmZ8030/AmZ8530 Serial Communications Controller Technical Manual" – 1982 edition (A1Z-2135) for detailed technical information.

GENERAL DESCRIPTION

The SCC Serial Communications Controller is a dual-channel, multi-protocol data communications peripheral designed for use with 8- and 16-bit microprocessors. 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 phase-locked loops, and crystal oscillators, which dramatically reduce the need for external logic.

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 drivers, 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 SCC is offered in two versions. The AmZ8030 is directly compatible with the Z8000 and 8086 CPUs. The AmZ8530 is designed for non-multiplexed buses and is easily interfaced to most other CPUs, such as 8080, Z80, 6800 68000, and †Multibus.

BLOCK DIAGRAM

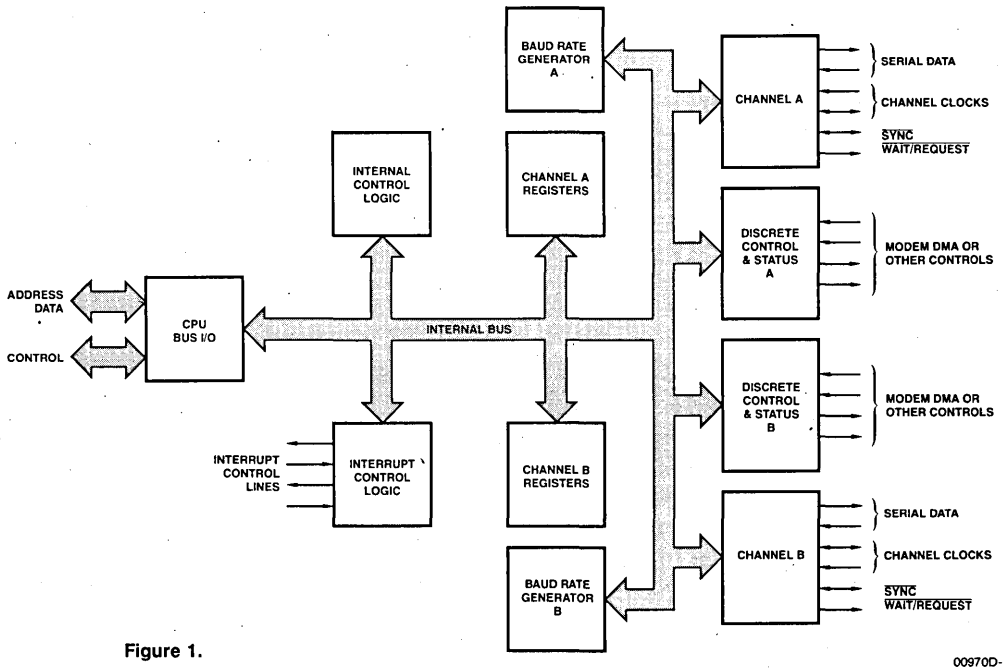


Figure 1.

00970D-1

*Z8000, Z8030 and Z8530 are trademarks of Zilog, Inc. †MULTIBUS is a trademark of Intel.

ARCHITECTURE

The SCC internal structure includes two full-duplex channels, two baud rate generators, internal control and interrupt logic, and a bus interface to the Z8000 CPU (AmZ8030) or to a non-multiplexed CPU bus (AmZ8530). 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 1).

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 synchronous 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 (A only), and one containing the Interrupt Pending bits (A only).

The registers for each channel are designated as follows:

WR0-WR15 – Write Registers 0 through 15.
RR0-RR3, RR10, RR12, RR13, RR15 – Read Registers 0 through 3, 10, 12, 13, 15.

The following table 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 2 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 sync-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).

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, shift right/shift left command
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

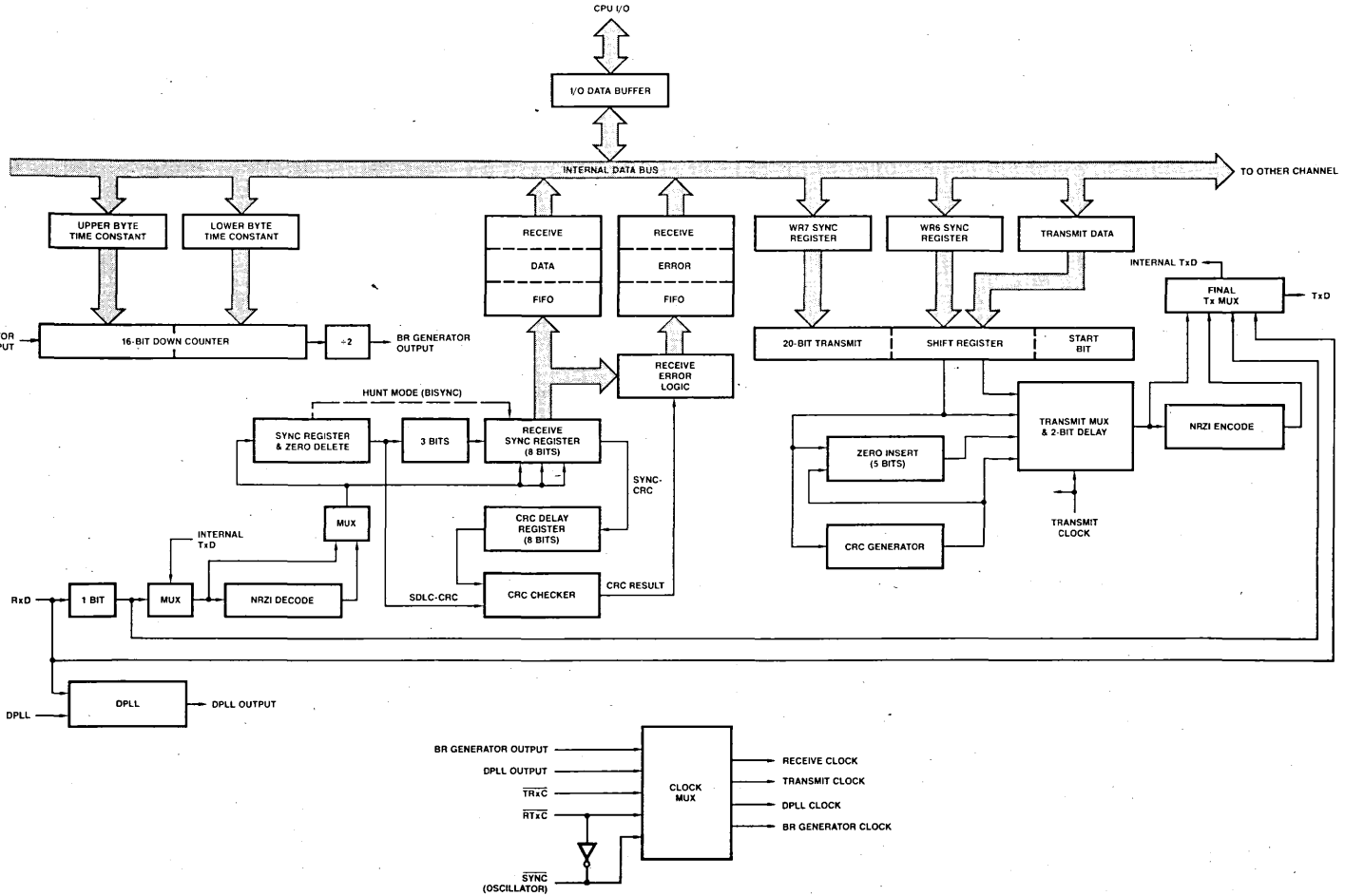


Figure 2. Data Path

00970D-2

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, it interacts with the CPU and provides vectored interrupts and handshaking signals.

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-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 18). 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 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 synchronous pattern (Bisync), or with an external synchronous signal. Leading synchronous 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.

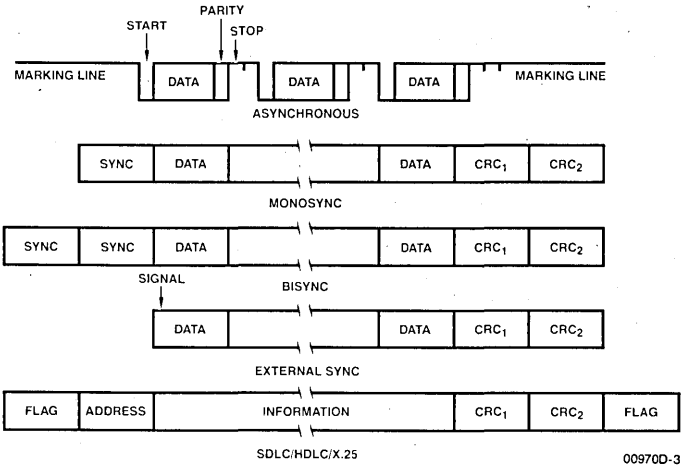


Figure 3. SCC Protocols

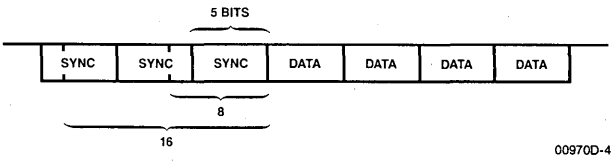


Figure 4. Detecting 5- or 7-Bit Synchronous Characters

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 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 $\overline{\text{SYNC}}$ 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.

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 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).

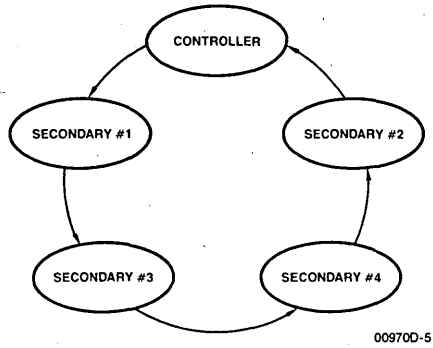


Figure 5. An SDLC Loop

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.

When a secondary station has a message to transmit and recognizes an EOP on the line, it changes the last binary one of the EOP to a zero 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 zero, 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 $\overline{TRxC}$ pin, the output of the baud rate generator may be echoed out via the $\overline{TRxC}$ pin.

The following formula relates the time constant to the baud rate. (The baud rate is in bits/second and the BR clock period is in seconds.)

$$\text{baud rate} = \frac{1}{2\,(\text{time constant} + 2) \times (\text{BR clock period})}$$

Time-Constant Values for Standard Baud Rates at BR Clock = 3.9936MHz

Rate (Baud)	Time Constant (decimal notation)	Error
19200	102	–
9600	206	–
7200	275	0.12%
4800	414	–
3600	553	0.06%
2400	830	–
2000	996	0.04%
1800	1107	0.03%
1200	1662	–
600	3326	–
300	6654	–
150	13310	–
134.5	14844	0.0007%
110	18151	0.0015%
75	26622	–
50	39934	–

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 1/0 or 0/1). 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 15/16 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 $\overline{TRxC}$ 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, as 1 is represented by no change in level and a 0 is represented by a change in level. In FM_1 (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 FM_0 (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,

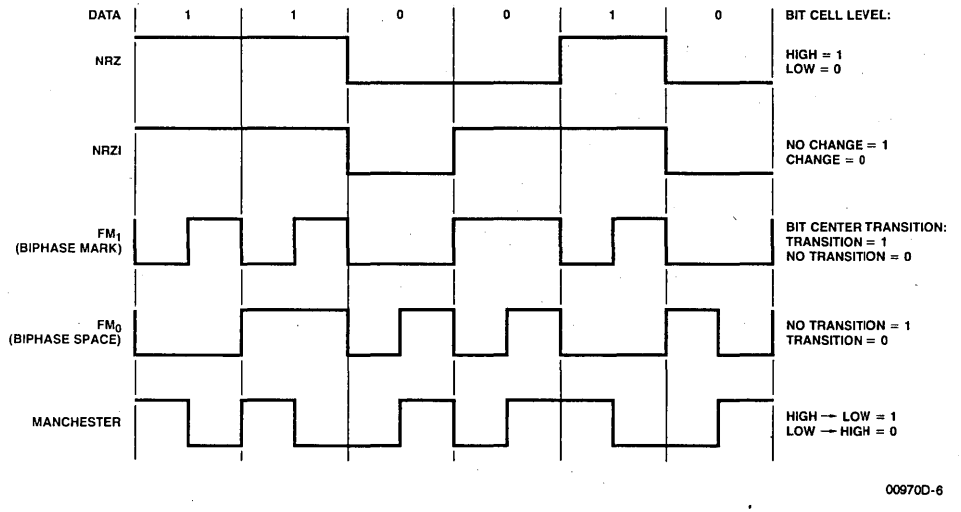


Figure 6. Data Encoding Methods

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 0/1 the bit is a 0. If the transition is 1/0 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 $\overline{CTS}$ 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 $\overline{WAIT}/\overline{REQUEST}$ 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 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 a 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 8 and 9).

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 Z-Bus interrupt priority chain (Figure 7). As a Z-Bus 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 A/D 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 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 interrupts. 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

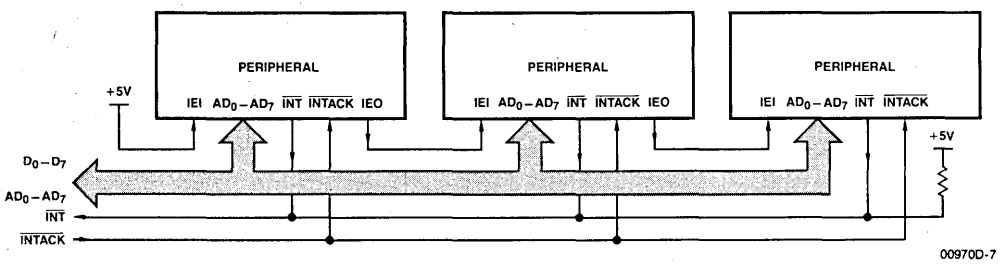


Figure 7. Z-Bus Interrupt Schedule

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 $\overline{CTS}$, $\overline{DCD}$, and $\overline{SYNC}$ 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 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 $\overline{WAIT/REQUEST}$ output in conjunction with the Wait/Request bits in WR1. The $\overline{WAIT/REQUEST}$ output can be defined under software control as a $\overline{WAIT}$ line in the CPU Block Transfer mode or as a $\overline{REQUEST}$ line in the DMA Block Transfer mode.

To a DMA controller, the SCC $\overline{REQUEST}$ output indicates that the SCC is ready to transfer data to or from memory. To the CPU, the $\overline{WAIT}$ line indicates that the SCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The $\overline{DTR/REQUEST}$ line allows full-duplex operation under DMA control.

PROGRAMMING

Each channel has fifteen Write registers that are individually programmed from the system bus to configure the functional personality of each channel. Each channel also has eight Read registers from which the system can read Status, Baud rate, or Interrupt information.

The AmZ8030 and AmZ8530 differ in the way the system accesses these registers:

In the AmZ8030 all registers are directly addressable from the multiplexed Address Data bus. See Figure 10 and Figure 11 for timing. The AmZ8030 can operate in either of two modes: when bit 0 in Write Register 0 is reset (or after initialization with a hardware reset) Address lines AD_1 through AD_5 select the register to be read from or written into during Data Stroke $\overline{DS}$. (This is called left shift and is the natural AmZ8000 mode). When bit 0 in Write Register 0 is set, Address lines AD_0 through AD_4 select the register to be read from or written into. (This is called right shift and is more natural for interfacing with other microprocessors.)

Table 2 describes the register addressing for both modes.

Channel A/Channel B selection is made either by AD_0 or by AD_5.

If Bit D_0 in WR0 is reset (or after hardware reset):

AD_5 selects the channel (0 = B, 1 = A)
(this is called "Select Shift Left Mode")

If Bits D_0 and D_1 in WR0 are set, AD_0 selects the channel (0 = B, 1 = A) (this is called "Select Shift Right Mode)

In the AmZ8530 only the four data registers (Read, Write for channels A and B) are directly selected by a High on the D/$\overline{C}$ input and the appropriate levels on the $\overline{RD}$, $\overline{WR}$ and A/$\overline{B}$ pins. All other registers are addressed indirectly by the content of Write Register 0 in conjunction with a Low on the D/$\overline{C}$ input and the appropriate levels on the $\overline{RD}$, $\overline{WR}$ and A/$\overline{B}$ pins. If bit 4 in WW0 is 1 and bits 5 and 6 are 0 then bits 0, 1, 2 address the higher registers 8 through 15. If bits 4, 5, 6 contain a different code, bits 0, 1, 2 address the lower registers 0 through 7 as shown on Table 3.

TABLE 2. REGISTER ADDRESSING (AmZ8030 ONLY)

AD_4	AD_3	AD_2	AD_1	Write Register	Read Register
0	0	0	0	0	0
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	(0)
0	1	0	1	5	(1)
0	1	1	0	6	(2)
0	1	1	1	7	(3)
1	0	0	0	Data	Data
1	0	0	1	9	–
1	0	1	0	10	10
1	0	1	1	11	(15)
1	1	0	0	12	12
1	1	0	1	13	13
1	1	1	0	14	(10)
1	1	1	1	15	15

Writing to or reading from any register except RR0, WR0 and the Data Registers thus involves two operations:

First write the appropriate code into WR0, then follow this by a write or read operation on the register thus specified. Bits 0 through 4 in WW0 are automatically cleared after this operation, so that WW0 then points to WR0 or RR0 again.

Channel A/Channel B selection is made by the A/$\overline{B}$ input (High = A, Low = B)

In both AmZ8030 and AmZ8530 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.

TABLE 3. REGISTER ADDRESSING (AmZ8530 ONLY)

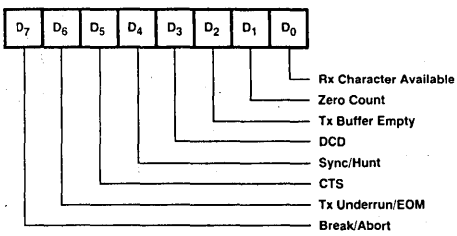
D/C̄ "Point High" Code in WR0:		D_2 in WR0:	D_1 in WR0:	D_0	Write Register	Read Register
High	Either way	X	X	X	Data	Data
Low	Not true	0	0	0	0	0
Low	Not true	0	0	1	1	1
Low	Not true	0	1	0	2	2
Low	Not true	0	1	1	3	3
Low	Not true	1	0	0	4	(0)
Low	Not true	1	0	1	5	(1)
Low	Not true	1	1	0	6	(2)
Low	Not true	1	1	1	7	(3)
Low	True	0	0	0	Data	Data
Low	True	0	0	1	9	–
Low	True	0	1	0	10	10
Low	True	0	1	1	11	(15)
Low	True	1	0	0	12	12
Low	True	1	0	1	13	13
Low	True	1	1	0	14	(10)
Low	True	1	1	1	15	15

READ REGISTERS

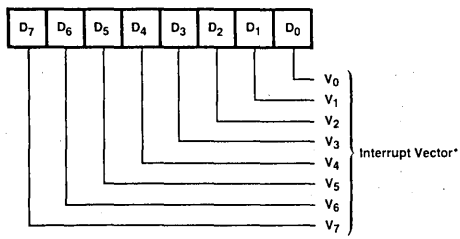
The SCC contains eight read registers (actually nine, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10, and RR15). Two registers (RR12 and RR13) may be read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A). Figure 8 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

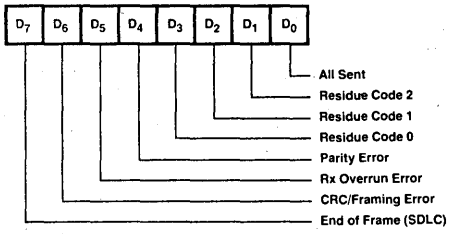


Read Register 2

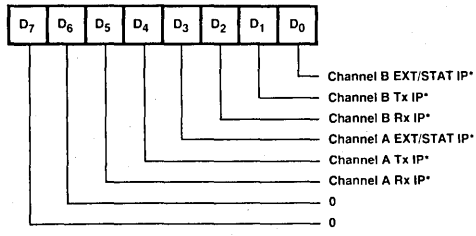


*Modified In B Channel

Read Register 1



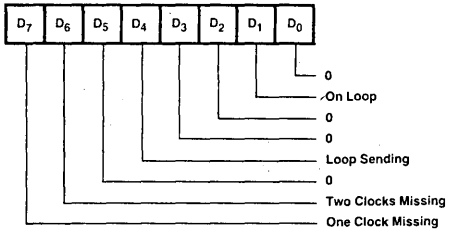
Read Register 3



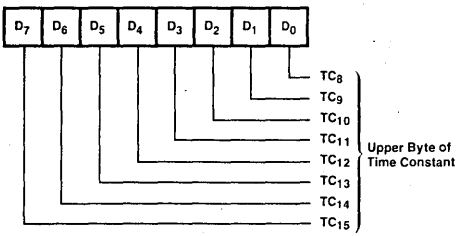
*Always 0 in B Channel

Figure 8. Read Register Bit Functions

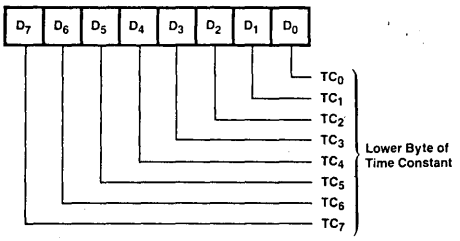
Read Register 10



Read Register 13



Read Register 12



Read Register 15

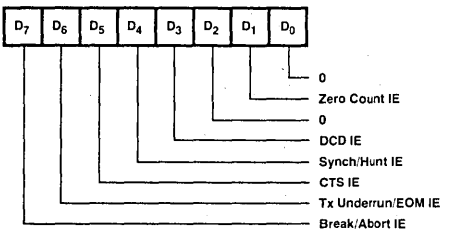
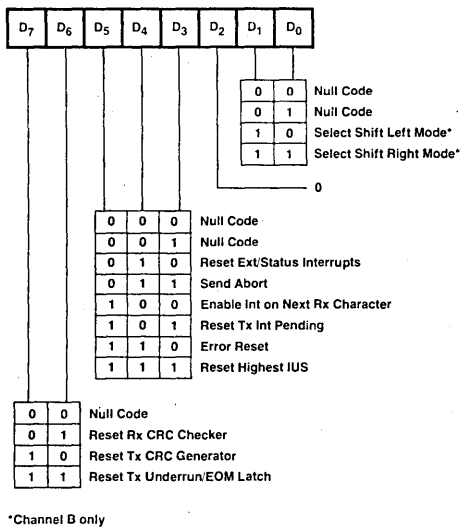


Figure 8. Read Register Bit Functions (Cont.)

WRITE REGISTERS

The SCC contains 15 write registers (16 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 9 shows the format of each write register.

Write Register 0 (AmZ8030)



*Channel B only

Write Register 0 (AmZ8530)

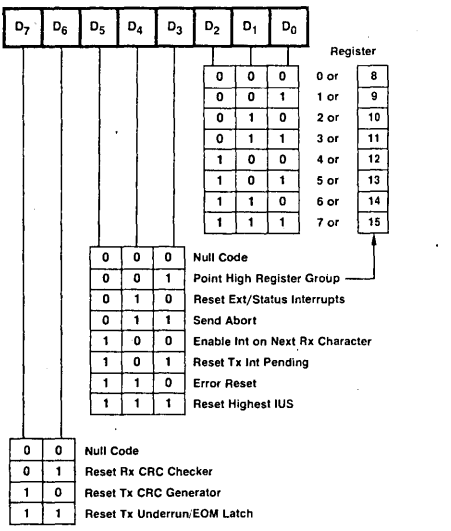
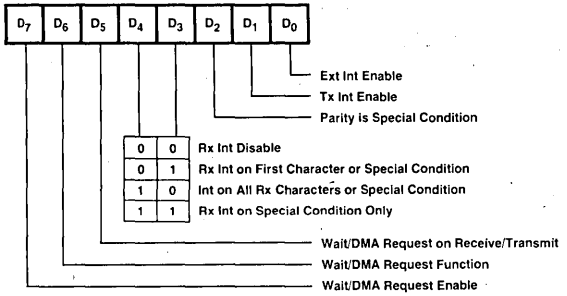
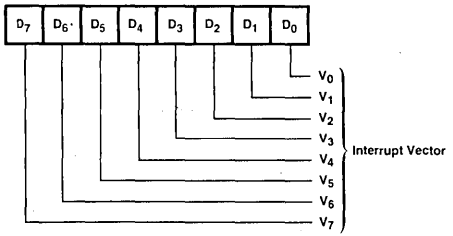


Figure 9. Write Register Bit Functions

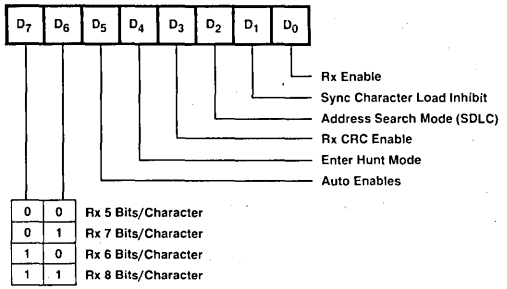
Write Register 1



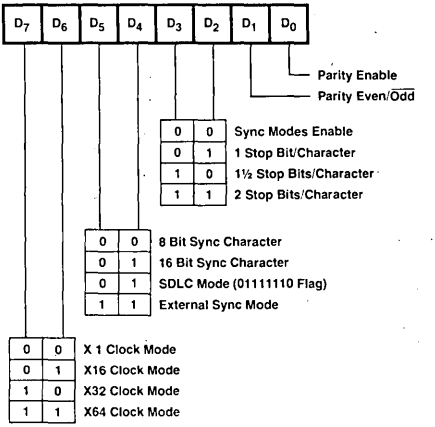
Write Register 2



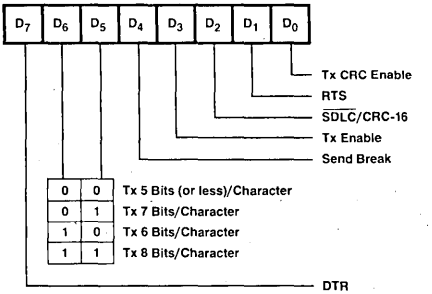
Write Register 3



Write Register 4



Write Register 5



Write Register 6

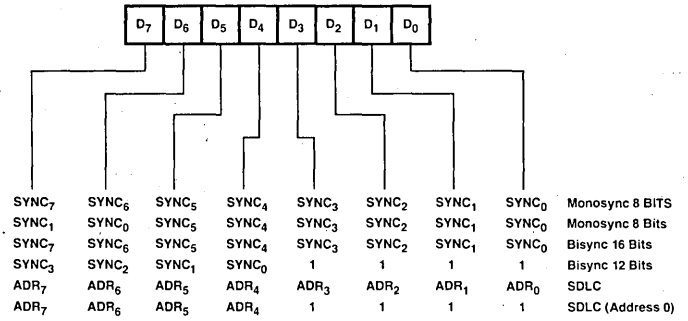
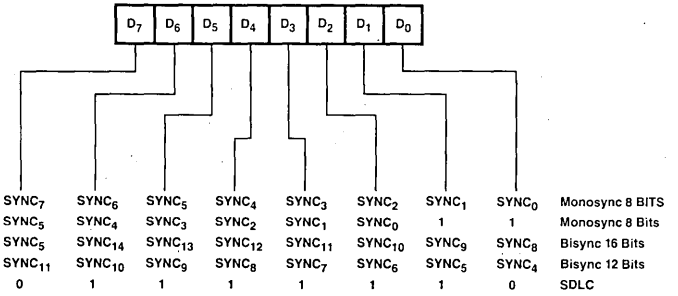
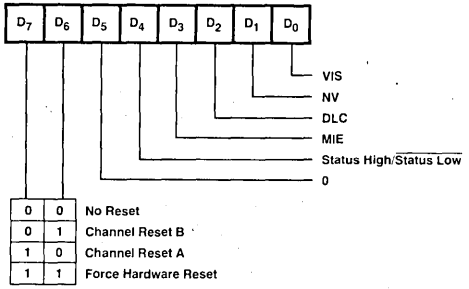


Figure 9. Write Register Bit Functions (Cont.)

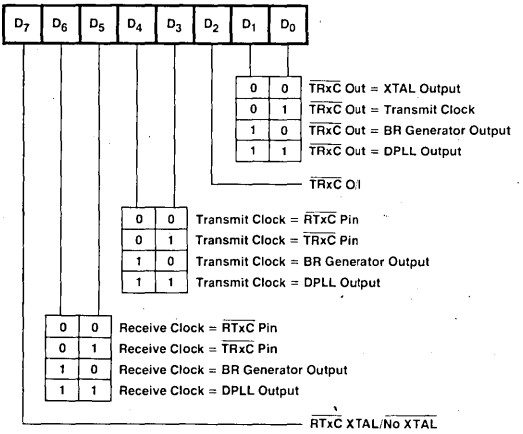
Write Register 7



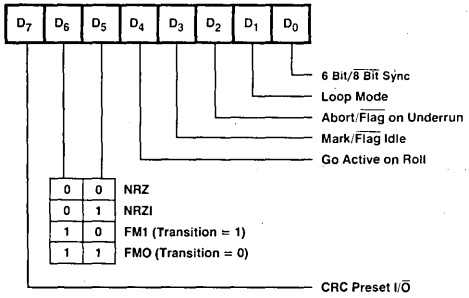
Write Register 9



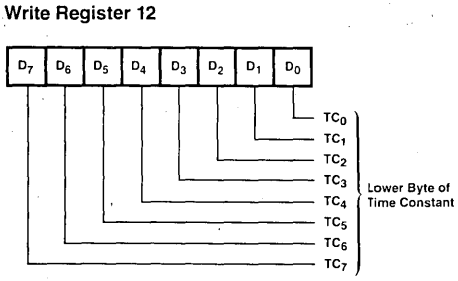
Write Register 11



Write Register 10



Write Register 12



Write Register 13

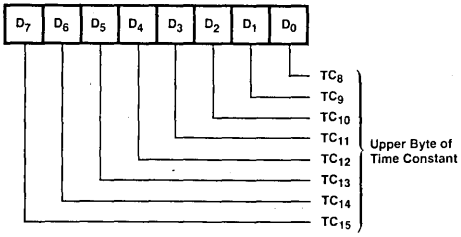


Figure 9. Write Register Bit Functions (Cont.)

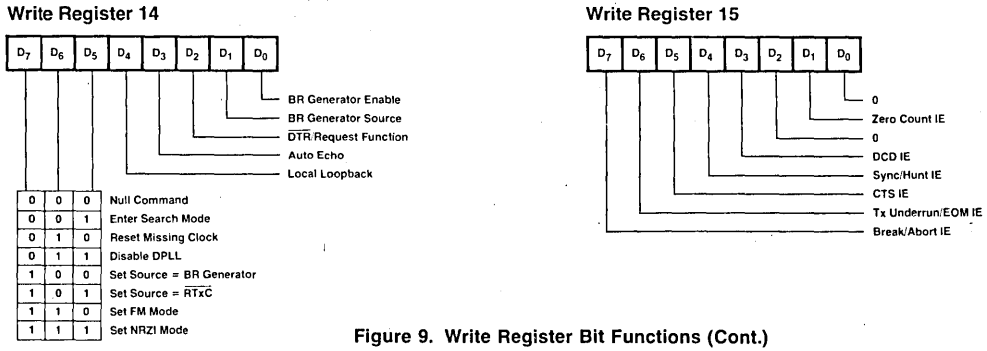


Figure 9. Write Register Bit Functions (Cont.)

AmZ8030 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 to the falling edge of $\overline{DS}$ in the second transaction involving the SCC. This time must be at least 6 PCLK cycles plus 200 ns.

READ CYCLE TIMING

Figure 10 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. 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.

WRITE CYCLE TIMING

Figure 11 illustrates write cycle timing. The address on AD_0 – AD_7 and the state if $\overline{CS_0}$ and $\overline{INTACK}$ are latched by the rising edge of $\overline{AS}$. R/$\overline{W}$ must be Low to indicate a write cycle. CS_1 must be High for the write cycle to occur. $\overline{DS}$ Low strobes the data into the SCC.

INTERRUPT ACKNOWLEDGE CYCLE TIMING

Figure 12 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 R/$\overline{W}$ 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 $\overline{DS}$ Low by placing its interrupt vector on AD_0 – AD_7. It then sets the appropriate interrupt-under-service latch internally.

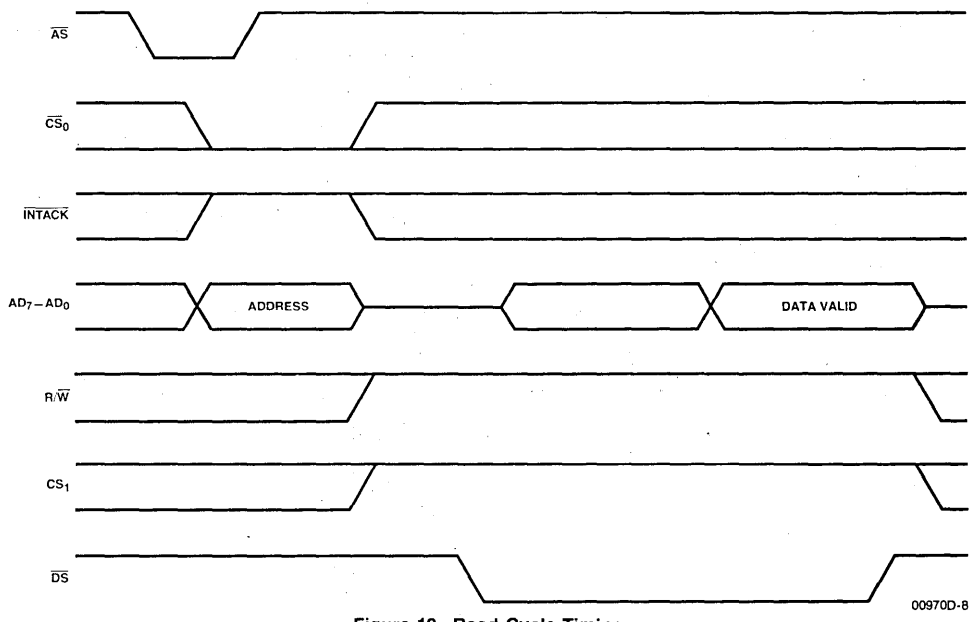


Figure 10. Read Cycle Timing

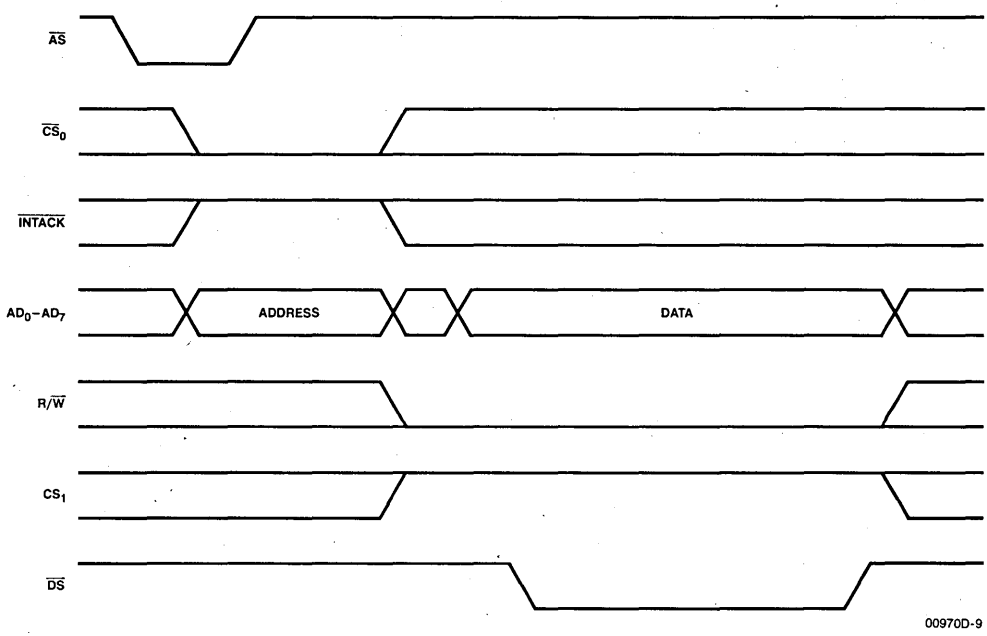


Figure 11. Write Cycle Timing

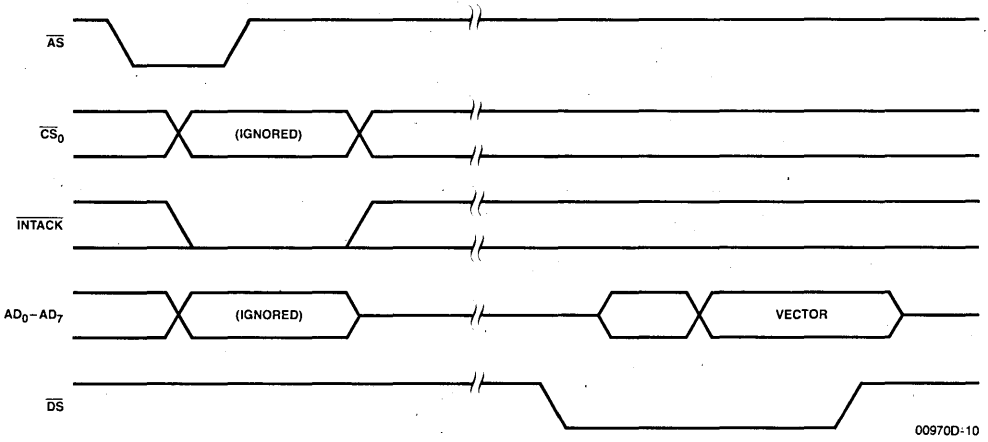


Figure 12. Interrupt Acknowledge Cycle Timing

AmZ8530 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 invloving the SCC. The recovery time required for proper operation is specified from the rising edge of $\overline{WR}$ or $\overline{RD}$ in the first transaction invloving 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 6 PCLK cycles plus 200ns.

Read Cycle Timing

Figure 13 illustrates Read cyle 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.

Write Cycle Timing

Figure 14 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.

Interrupt Acknowledge Cycle Timing

Figure 15 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-Under-Service internally.

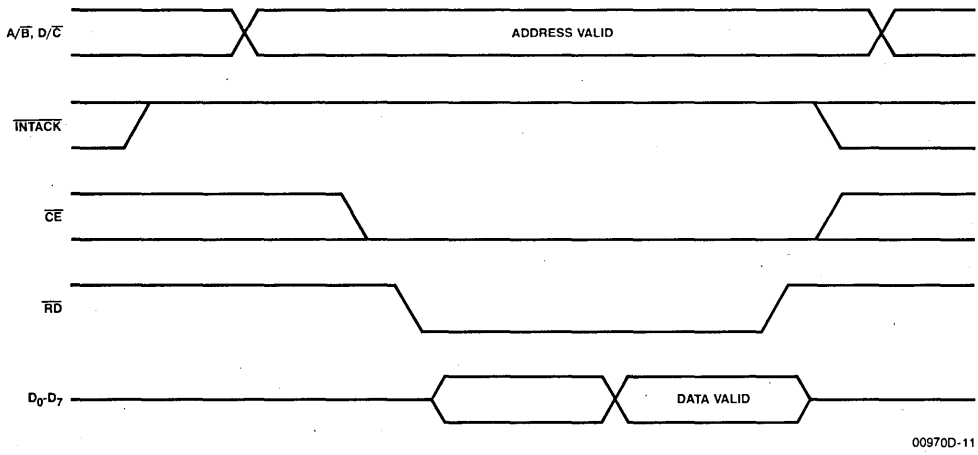


Figure 13. Read Cycle Timing

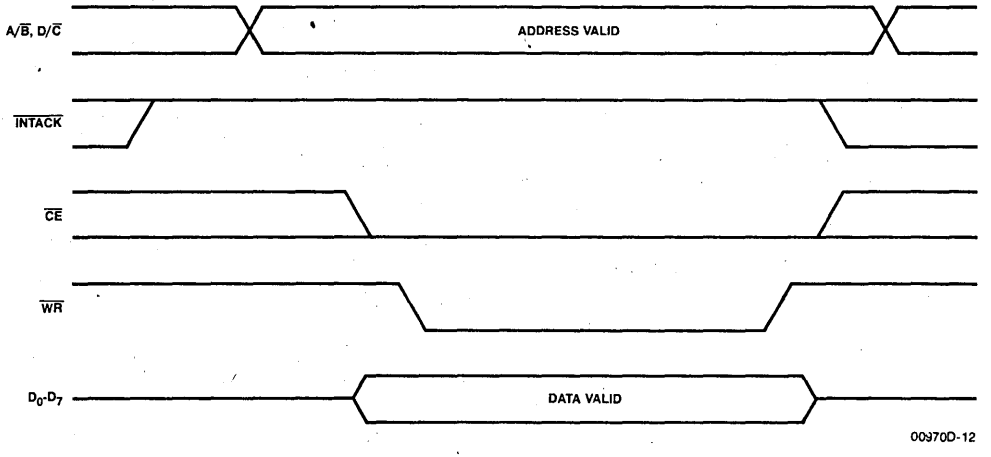


Figure 14. Write Cycle Timing

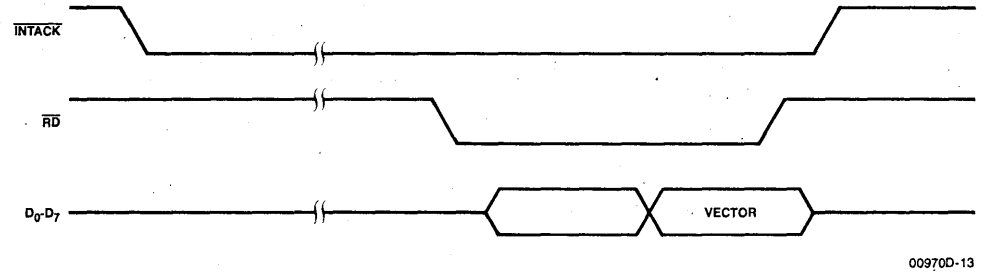


Figure 15. Interrupt Acknowledge Cycle Timing

PIN DESCRIPTIONS FOR AmZ8030

The following section describes the pin functions of the SCC. Figures 16 and 17 detail the respective pin functions and pin assignments.

V_{CC}: +5V Power Supply

GND: Ground

AD_0 – AD_7: Address/Data Bus (bidirectional, active High, 3-state).

These multiplexed lines carry register addresses to the SCC as well as data or control information to and from the SCC.

$\overline{AS}$: Address Strobe (input, active Low).

Addresses on AD_0 – AD_7 are latched by the rising edge of this signal.

$\overline{CS_0}$: Chip Select 0 (input, active Low).

This signal is latched concurrently with the addresses on AD_0 – AD_7 and must be active for the intended bus transaction to occur.

CS_1: Chip Select 1 (input, active High).

This second select signal must also be active before the intended bus transaction can occur. CS_1 must remain active throughout the transaction.

$\overline{CTSA}$, $\overline{CTSB}$: Clear to Send (Inputs, active Low).

If these pins are programmed as Auto Enables, a Low on these inputs enables their respective transmitter. 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.

$\overline{DCDA}$, $\overline{DCDB}$: Data Carrier Detect (Inputs, 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.

$\overline{DS}$: Data Strobe (input, active Low).

This signal provides timing for the transfer of data into and out of the SCC. If $\overline{AS}$ and $\overline{DS}$ coincide, this is interpreted as a reset.

$\overline{DTR}$/$\overline{REQA}$, $\overline{DTR}$/$\overline{REQB}$: 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 a 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.

$\overline{INT}$: Interrupt Request (output, open-drain, active Low).

This signal is activated when the SCC requests an interrupt.

$\overline{INTACK}$: Interrupt Acknowledge (input, active Low).

This signal indicates an active interrupt acknowledge cycle. During this cycle, the SCC interrupt daisy chain settles. When $\overline{DS}$ 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 $\overline{AS}$.

PCLK: Clock (input)

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, although the frequency of this clock must be at least 90% of the CPU clock frequency for a Z8000. PCLK is a TTL level signal. Maximum transmit rate is 1/4 PCLK.

RxDA, RxDB: Receive Data (inputs, active High)

These input signals receive serial data at standard TTL levels.

$\overline{RTxCA}$, $\overline{TRxCB}$: 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 of 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.

$\overline{RTSA}$, $\overline{RTSB}$: Request to Send (outputs active Low)

When the Request to Send RTS bit in Write Register 5 (Figure 6) 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 $\overline{RTS}$ pins strictly follow the state of the RTS bit. Both pins can be used as general-purpose outputs.

R/$\overline{W}$: Read/Write (input)

This signal specifies whether the operation to be performed is read or a write.

$\overline{SYNCA}$, $\overline{SYNCB}$: Synchronization (inputs or outputs, active Low)

These pins can act either as inputs, ouputs, or part of the crystal oscillator circuit.

In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to $\overline{CTS}$ and $\overline{DCD}$. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 (Figure 5) 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 out-puts 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 synchronous 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.

$\overline{TRxCA}$, $\overline{TRxCB}$: Transmit/Receive Clocks (inputs or outputs, active Low)

These pins can be programmed in several different modes of operation. $\overline{TRxC}$ 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.

$\overline{W}$/$\overline{REQA}$, $\overline{W}$/$\overline{REQB}$: 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.

PIN DESCRIPTIONS FOR AmZ8530

The following section describes the pin functions of the SCC. Figures 18 and 19 detail the respective pin functions and pin assignments.

V_{CC}: +5V Power Supply

GND: Ground

A/$\overline{B}$. *Channel A/Channel B Select* (input). This signal selects the channel in which the read or write operation occurs.

$\overline{CE}$. *Chip Enable* (input, active Low). This signal selects the SCC for a read or write operation.

$\overline{CTSA}$, $\overline{CTSB}$. *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.

D/$\overline{C}$. *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.

$\overline{DCDA}$, $\overline{DCDB}$. *Data Carrier Detect* (inputs, 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 accomodate slow rise time signals. The SCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

D_0-D_7. *Data Bus* (bidirectional, 3-state). These lines carry data and commands to and from the SCC.

$\overline{DTR/REQA}$, $\overline{DTR/REQB}$. *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.

$\overline{INT}$. *Interrupt Request* (output, open-drain, active Low). This signal is activated when the SCC requests an interrupt.

$\overline{INTACK}$. *Interrupt Acknowledge* (input, active Low). This signal indicates an active Interrupt Acknowledge cycle. During this cycle, the SCC interrupt daisy chain settles. When $\overline{RD}$ 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.

$\overline{RD}$. *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.

RxDA, RxDB. *Receive Data* (inputs, active High). These input signals receive serial data at standard TTL levels.

$\overline{RTxCA}$, $\overline{RTxCB}$. *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.

$\overline{RTSA}$, $\overline{RTSB}$. *Request To Send* (outputs active Low). When the Request to Send (RTS) bit in Write Register 5 (Figure 6) 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 $\overline{RTS}$ pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

$\overline{SYNCA}$, $\overline{SYNCB}$. *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 $\overline{CTS}$ and $\overline{DCD}$. In this mode, transitions on these lines affect the state of the Synchronous/Hunt status bits in Read Register 0 (Figure 5) 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 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.

$\overline{TRxCA}$, $\overline{TRxCB}$. *Transmit/Receive Clocks* (inputs or outputs, active Low). These pins can be programmed in several different modes of operation. $\overline{TRxC}$ 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.

$\overline{WR}$. *Write* (input, active Low). When the SCC is selected, this signal indicates a write operation. The coincidence of $\overline{RD}$ and $\overline{WR}$ is interpreted as a reset.

$\overline{W/REQA}$, $\overline{W/REQB}$. *Wait/Request* (outputs, open-drain when programmed for a Wait function, driven High or Low when progammed 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.

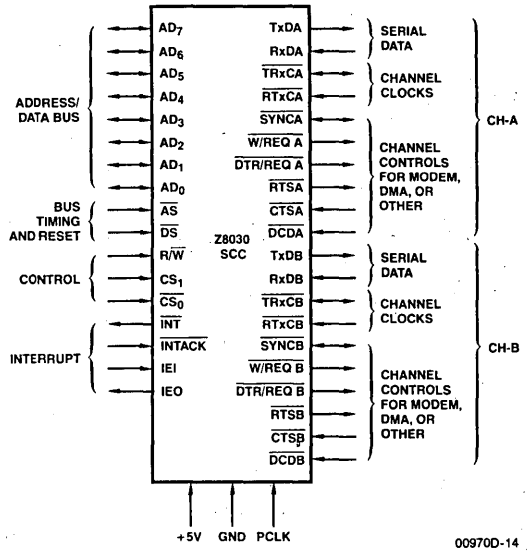
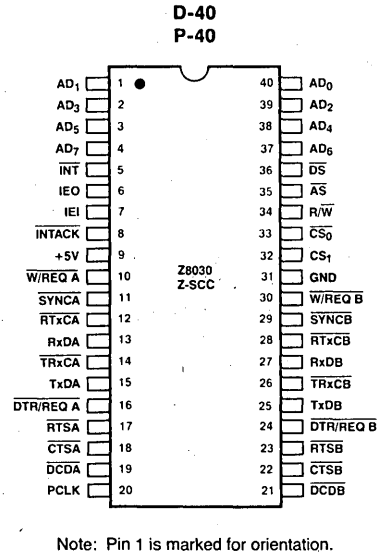


Figure 16. Logic Symbol



Note: Pin 1 is marked for orientation.

00970D-15

Figure 17. Connection Diagram

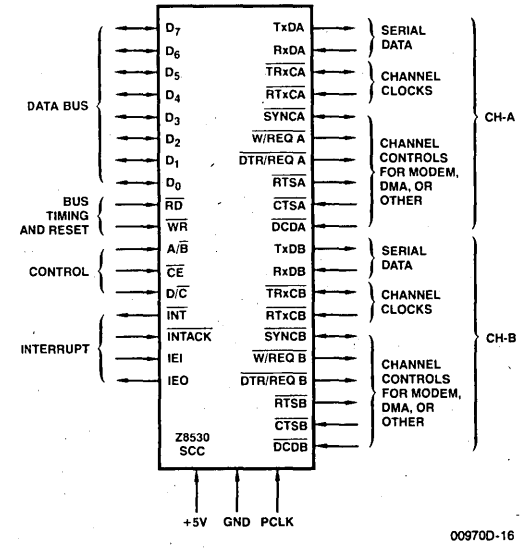
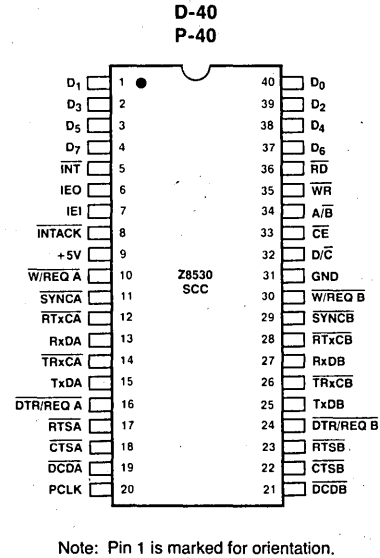


Figure 18. Logic Symbol

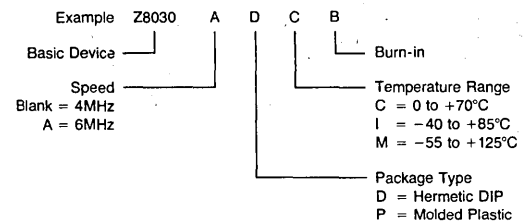


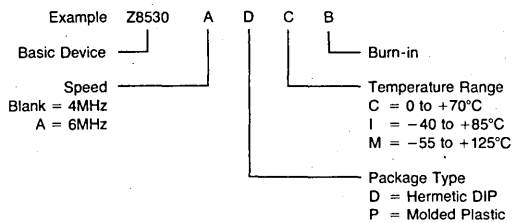
Note: Pin 1 is marked for orientation.

00970D-17

Figure 19. Connection Diagram

ORDERING INFORMATION





MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65 to +150°C
Voltage at any Pin Relative to V_{SS}	−0.5 to +7.0V
Power Dissipation	1.8W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

(over which the DC, switching and functional specifications apply)

	4MHz	6MHz
Commercial Operating Range T_A = 0 to +70°C V_{CC} = 5V ± 5%	Z8030DC Z8030PC Z8530DC Z8530PC	Z8030ADC Z8030APC Z8530ADC Z8530APC
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ± 5%	Z8030DI Z8530DI	
Military Operating Range T_A = −55 to +125°C V_{CC} = 5V ± 10%	Z8030DMB Z8530DMB	

Notes: T_A denotes ambient temperature.
Add suffix B to indicate burn-in requirement.

ELECTRICAL CHARACTERISTICS

Parameter	Description	Test Conditions	Min	Typ	Max	Units
V_{IH}	Input HIGH Voltage	Standard	2.0		V_{CC} + 0.3	V
		Military	2.2			
V_{IL}	Input LOW Voltage		−0.3		0.8	V
V_{OH}	Output HIGH Voltage	I_{OH} = −250μA	2.4			V
V_{OL}	Output LOW Voltage	I_{OL} = +2.0mA			0.4	V
I_{IL}	Input Leakage	0.4 ≤ V_{IN} ≤ +2.4V			±10.0	μA
I_{OL}	Output Leakage	0.4 ≤ V_{OUT} ≤ +2.4V			±10.0	μA
I_{CC}	V_{CC} Supply Current				250	mA
C_{IN}	Input Capacitance	Unmeasured pins returned to ground. f = 1MHz over specified temperature range.			10	pF
C_{OUT}	Output Capacitance				15	pF
$C_{I/O}$	Bidirectional Capacitance				20	pF

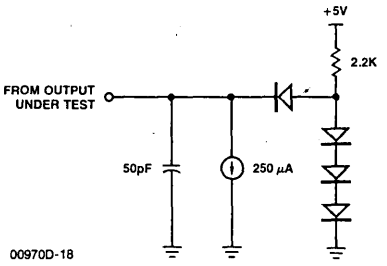
V_{CC} = 5V ± 5% unless otherwise specified, over specified temperature range.

Standard Test Conditions

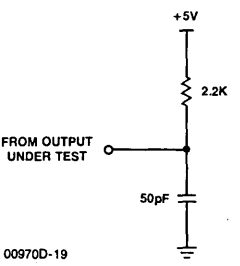
The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

+4.75V ≤ V_{CC} ≤ +5.25V
GND = 0V
0°C ≤ T_A ≤ +70°C

Standard Test Load



Open Drain Test Load



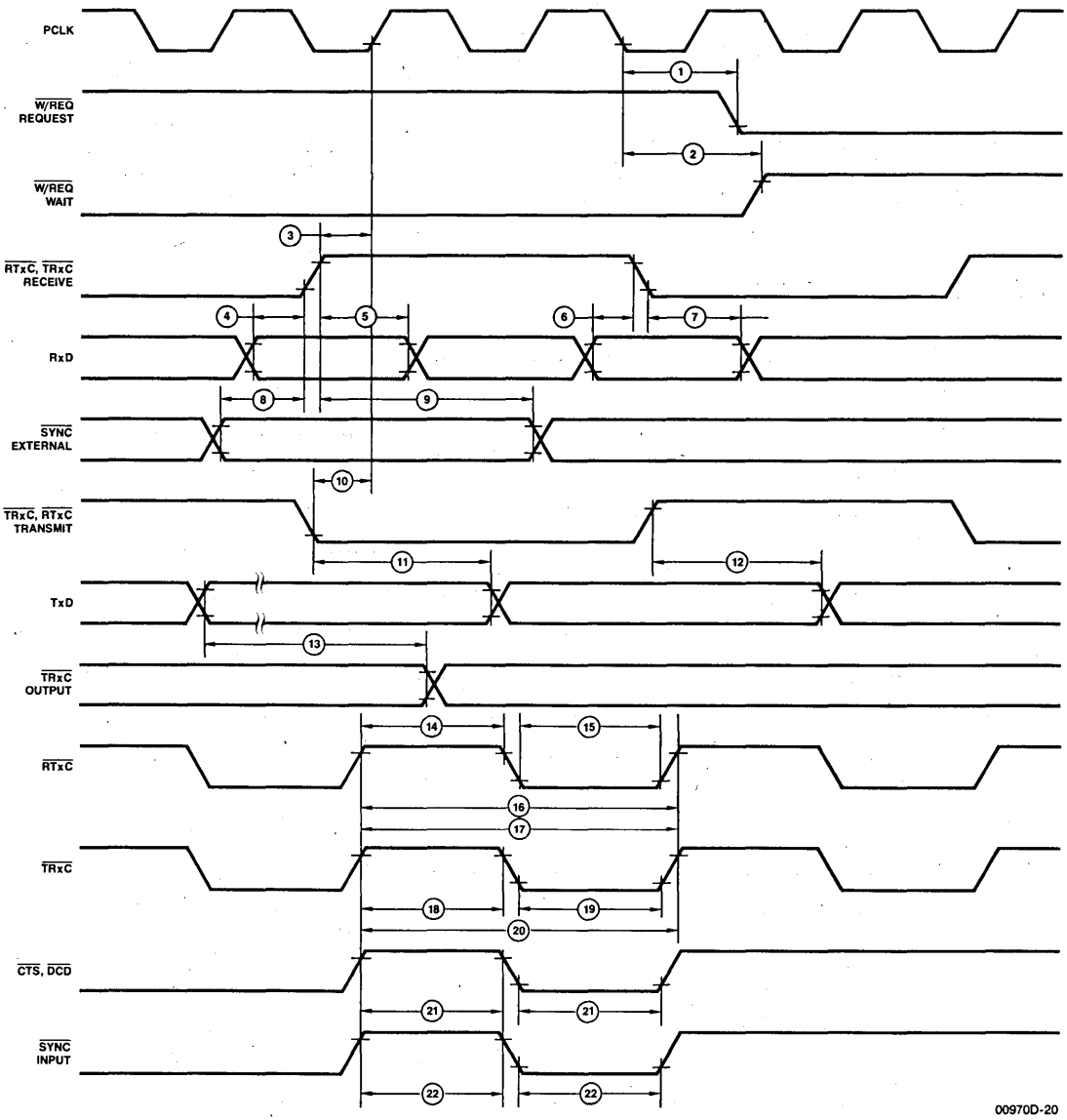
GENERAL TIMING

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdPC(REQ)	PCLK ↓ to W/$\overline{REQ}$ Valid Delay		250		250	ns
2	TdPC(W)	PCLK ↓ to Wait Inactive Delay		350		350	ns
3	TsRXC(PC)	$\overline{RxC}$ ↑ to PCLK ↑ Setup Time (Notes 1, 4)	50		50		ns
4	TsRXD(RXCr)	RxD to $\overline{RxC}$ ↑ Setup Time (XI Mode) (Note 1)	0		0		ns
5	ThRXD(RXCr)	RxD to $\overline{RxC}$ ↑ Hold Time (XI Mode) (Note 1)	150		150		ns
6	TsRXD(RXCf)	RxD to $\overline{RxC}$ ↓ Setup Time (XI Mode) (Notes 1, 5)	0		0		ns
7	ThRXD(RXCf)	RxD to $\overline{RxC}$ ↓ Hold Time (XI Mode) (Notes 1, 5)	150		150		ns
8	TsSY(RXC)	$\overline{SYNC}$ to $\overline{RxC}$ ↑ Setup Time (Note 1)	−200		−200		ns
9	ThSY(RXC)	$\overline{SYNC}$ to $\overline{RxC}$ ↑ Hold Time (Note 1)	3TcPC +200		3TcPC +200		ns
10	TsTXC(PC)	$\overline{TxC}$ ↓ to PCLK ↑ Setup Time (Notes 2, 4)	0		0		ns
11	TdTXCf(TXD)	$\overline{TxC}$ ↓ to TxD Delay (XI Mode) (Note 2)		300		300	ns
12	TdTXCr(TXD)	$\overline{TxC}$ ↑ to TxD Delay (XI Mode) (Notes 2, 5)		300		300	ns
13	TdTXD(TRX)	TxD to $\overline{TRxC}$ Delay (Send Clock Echo)					ns
14	TwRTXh	$\overline{RTxC}$ High Width	180		180		ns
15	TwRTXl	$\overline{RTxC}$ Low Width	180		180		ns
16	TcRTX	$\overline{RTxC}$ Cycle Time	400		400		ns
17	TcRTXX	Crystal Oscillator Period (Note 3)	250	1000	250	1000	ns
18	TwTRXh	$\overline{TRxC}$ High Width	180		180		ns
19	TwTRXl	$\overline{TRxC}$ Low Width	180		180		ns
20	TcTRX	$\overline{TRxC}$ Cycle Time	400		400		ns
21	TwEXT	$\overline{DCD}$ or $\overline{CTS}$ Pulse Width	200		200		ns
22	TwSY	$\overline{SYNC}$ Pulse Width	200		200		ns

Notes:
1. RxC is $\overline{RTxC}$ or $\overline{TRxC}$, whichever is supplying the receive clock.
2. TxC is $\overline{TRxC}$ or $\overline{RTxC}$, whichever is supplying the transmit clock.
3. Both $\overline{RTxC}$ and $\overline{SYNC}$ have 30pF 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 $\overline{RxC}$ and PCLK or $\overline{TxC}$ and PCLK is required.
5. Parameter applies only to FM encoding/decoding.
6. Parameter applies only if $\overline{RTxC}$ is used directly as Tx or Rx clock. If used as a DPLL or BR source, specifications are the same as PCLK (4MHz).

*Timings are preliminary and subject to change.

Figure 20. General Timing

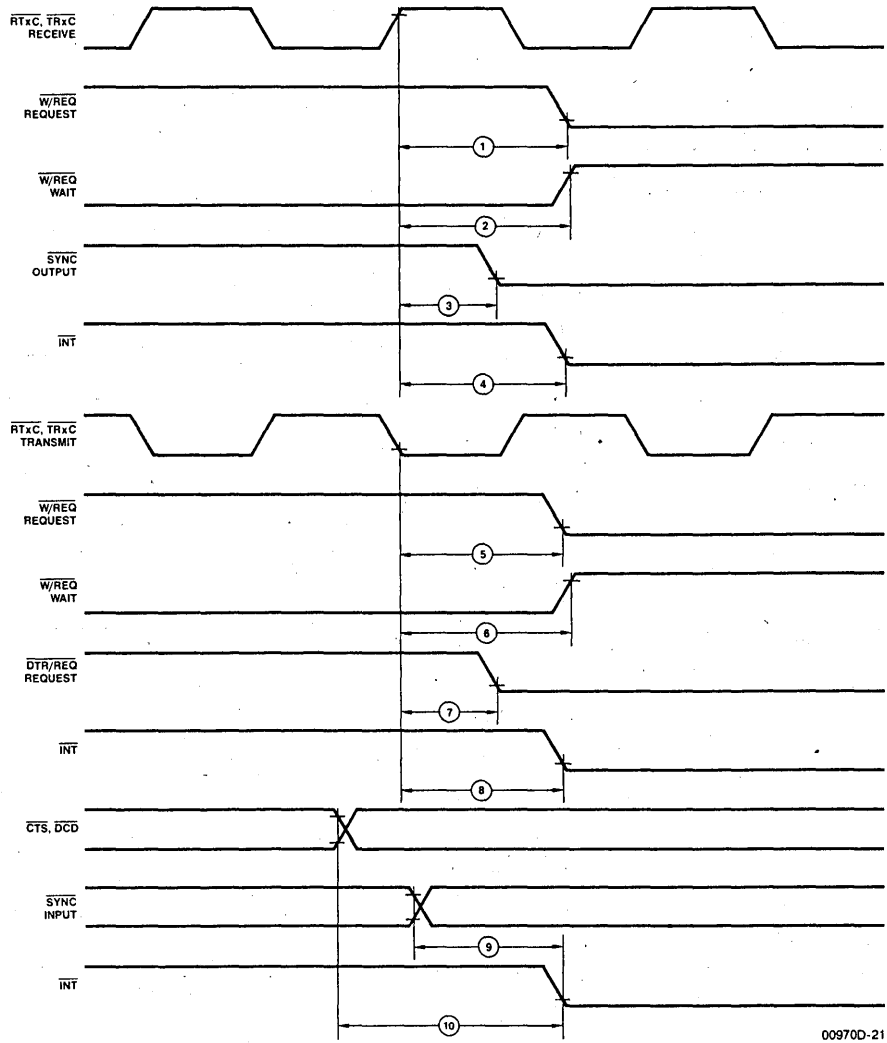


SYSTEM TIMING

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdRXC(REQ)	$\overline{RxC}$ ↑ to $\overline{W/REQ}$ Valid Delay (Note 2)	8	12	8	12	TcPC
2	TdRXC(W)	$\overline{RxC}$ ↑ to Wait Inactive Delay (Notes 1, 2)	8	12	8	12	TcPC
3	TdRXC(SY)	$\overline{RxC}$ ↑ to $\overline{SYNC}$ Valid Delay (Note 2)	4	7	4	7	TcPC
4	TdRXC(INT)	$\overline{RxC}$ ↑ to $\overline{INT}$ Valid Delay (Notes 1, 2)	10	16	10	16	TcPC
5	TdTXC(REQ)	$\overline{TxC}$ ↓ to $\overline{W/REQ}$ Valid Delay (Note 3)	5	8	5	8	TcPC
6	TdTXC(W)	$\overline{TxC}$ ↓ to Wait Inactive Delay (Notes 1, 3)	5	8	5	8	TcPC
7	TdTXC(DRQ)	$\overline{TxC}$ ↓ to $\overline{DTR/REQ}$ Valid Delay (Note 3)	4	7	4	7	TcPC
8	TdTXC(INT)	$\overline{TxC}$ ↓ to $\overline{INT}$ Valid Delay (Notes 1, 3)	6	10	6	10	TcPC
9	TdSY(INT)	$\overline{SYNC}$ Transition to $\overline{INT}$ Valid Delay (Note 1)	2	6	2	6	TcPC
10	TdEXT(INT)	$\overline{DCD}$ or $\overline{CTS}$ Transition to $\overline{INT}$ Valid Delay (Note 1)	2	6	2	6	TcPC

Notes: 1. Open-drain output, measured with open-drain test load.
2. $\overline{RxC}$ is $\overline{RTxC}$ or $\overline{TRxC}$, whichever is supplying the receive clock.
3. $\overline{TxC}$ is $\overline{TRxC}$ or $\overline{RTxC}$, whichever is supplying the transmit clock.
*Timings are preliminary and subject to change.

Figure 21. System Timing

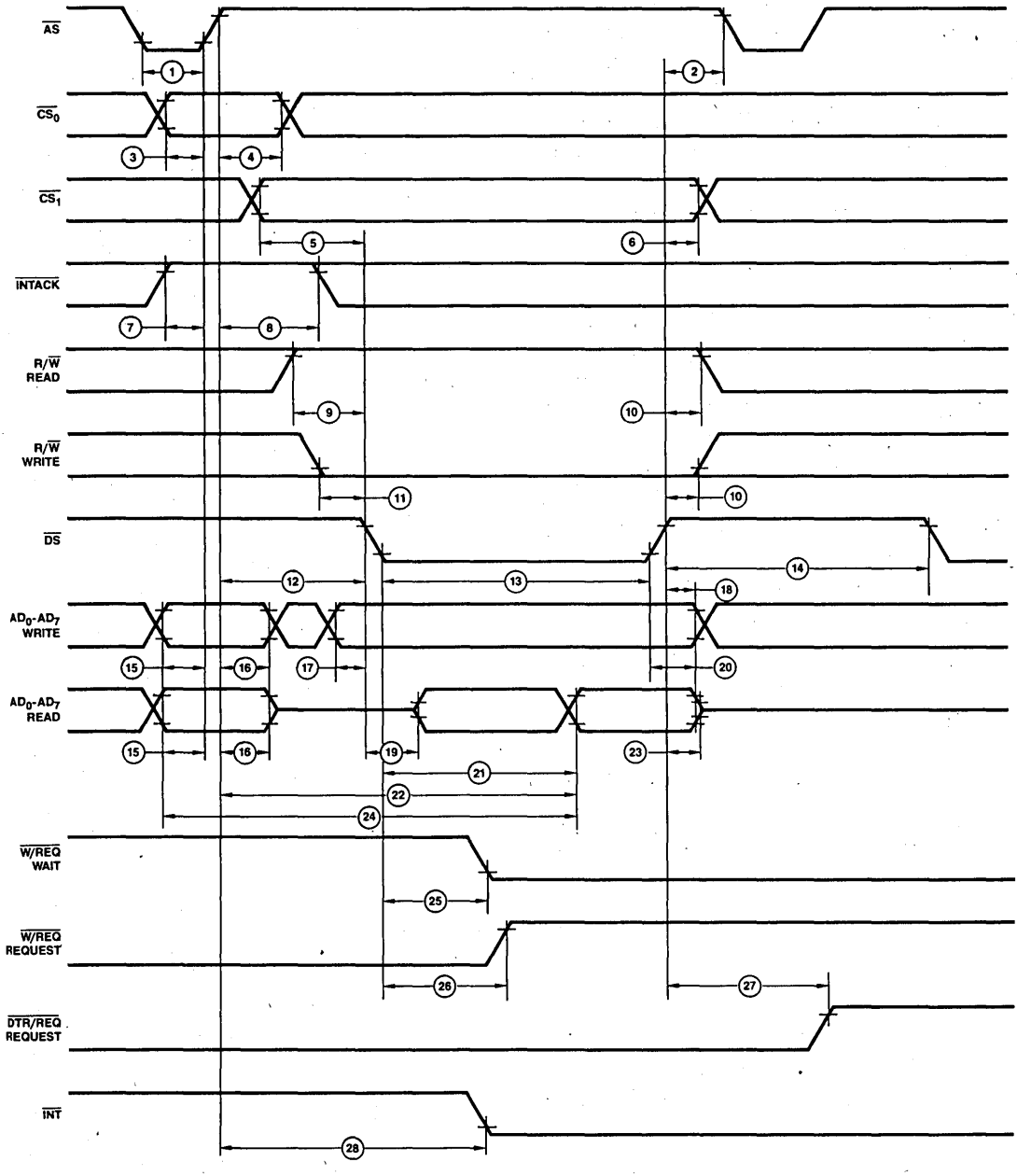


READ AND WRITE TIMING (AmZ8030)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TwAS	$\overline{AS}$ LOW Width	70		50		ns
2	TdDS(AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	50		25		ns
3	TsCS0(AS)	$\overline{CS_0}$ to $\overline{AS}$ ↑ Setup Time (Note 1)	0		0		ns
4	ThCS0(AS)	$\overline{CS_0}$ to $\overline{AS}$ ↑ Hold Time (Note 1)	60		40		ns
5	TsCS1(DS)	CS_1 to $\overline{DS}$ ↓ Setup Time (Note 1)	100		80		ns
6	ThCS1(DS)	CS_1 to $\overline{DS}$ ↑ Hold Time (Note 1)	55		40		ns
7	TsIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Setup Time	0		0		ns
8	ThIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Hold Time	250		250		ns
9	TsRWR(DS)	R/$\overline{W}$ (Read) to $\overline{DS}$ ↓ Setup Time	100		80		ns
10	ThRW(DS)	R/$\overline{W}$ to $\overline{DS}$ ↑ Hold Time	55		40		ns
11	TsRWW(DS)	R/$\overline{W}$ (Write) to $\overline{DS}$ ↓ Setup Time	0		0		ns
12	TdAS(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay	60		40		ns
13	TwDSI	$\overline{DS}$ LOW Width	390		250		ns
14	TrC	Valid Access Recovery Time (Note 2)	6TcPC +200		6TcPC +130		ns
15	TsA(AS)	Address to $\overline{AS}$ ↑ Setup Time (Note 1)	30		10		ns
16	ThA(AS)	Address to $\overline{AS}$ ↑ Hold Time (Note 1)	50		30		ns
17	TsDW(DS)	Write Data to $\overline{DS}$ ↓ Setup Time	30		20		ns
18	ThDW(DS)	Write Data to $\overline{DS}$ ↑ Hold Time	30		20		ns
19	TdDS(DA)	$\overline{DS}$ ↓ to Data Active Delay	0		0		ns
20	TdDSr(DR)	$\overline{DS}$ ↑ to Read Data Not Valid Delay	0		0		ns
21	TdDSf(DR)	$\overline{DS}$ ↓ to Read Data Valid Delay		250		180	ns
22	TdAS(DR)	$\overline{AS}$ ↑ to Read Data Valid Delay		520		335	ns

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Parameter applies only between transactions involving the 8030.

Figure 22. Read and Write Timing (AmZ8030)



INTERRUPT ACKNOWLEDGE TIMING, RESET TIMING, CYCLE TIMING (AmZ8030)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
23	TdDS(DRz)	$\overline{DS}$ ↑ to Read Data Float Delay (Note 3)		70		45	ns
24	TdA(DR)	Address Required Valid to Read Data Valid Delay		570		420	ns
25	TdDS(W)	$\overline{DS}$ ↓ to Wait Valid Delay (Note 4)		240		200	ns
26	TdDSf(REQ)	$\overline{DS}$ ↓ to $\overline{W/REQ}$ Not Valid Delay		240		200	ns
27	TdDSr(REQ)	$\overline{DS}$ ↓ to $\overline{DTR/REQ}$ Not Valid Delay		5TcPC +300		5TcPC +250	ns
28	TdAS(INT)	$\overline{AS}$ ↑ to $\overline{INT}$ Valid Delay (Note 4)		500		500	ns
29	TdAS(DSA)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ (Acknowledge) Delay (Note 5)					ns
30	TwDSA	$\overline{DS}$ (Acknowledge) Low Width	390		250		ns
31	TdDSA(DR)	$\overline{DS}$ ↓ (Acknowledge) to Read Data Valid Delay		250		180	ns
32	TsIEI(DSA)	IEI to $\overline{DS}$ ↓ (Acknowledge) Setup Time	120		100		ns
33	ThIEI(DSA)	IEI to $\overline{DS}$ ↑ (Acknowledge) Hold Time	0		0		ns
34	TdIEI(IEO)	IEI to IEO Delay		120		100	ns
35	TdAS(IEO)	$\overline{AS}$ ↑ to IEO Delay (Note 6)		250		250	ns
36	TdDSA(INT)	$\overline{DS}$ ↓ (Acknowledge) to $\overline{INT}$ Inactive Delay (Note 4)		500		500	ns
37	TdDS(ASQ)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay for No Reset	30		15		ns
38	TdASQ(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay for No Reset	30		30		ns
39	TwRES	$\overline{AS}$ and $\overline{DS}$ Coincident Low for Reset (Note 7)	250		250		ns
40	TwPCl	PCLK Low Width	105	2000	70	1000	ns
41	TwPCh	PCLK High Width	105	2000	70	1000	ns
42	TcPC	PCLK Cycle Time	250	4000	165	2000	ns
43	TrPC	PCLK Rise Time		20		15	ns
44	TfPC	PCLK Fall Time		20		10	ns

Notes: 3. Float delay is defined as the time required for a ±0.5V change in the output with a maximum dc load and minimum ac load.
4. Open-drain output, measured with open-drain test load.
5. Parameter is system dependent. For any 8030 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 8030, and TdIEIf(IEO) for each device separating them in the daisy chain.
6. Parameter applies only to a 8030 pulling $\overline{INT}$ 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 8030.

All timing references assume 2.0V for a logic "1" and 0.8V for a logic.

Figure 23. Interrupt Acknowledge Timing (AmZ8030)

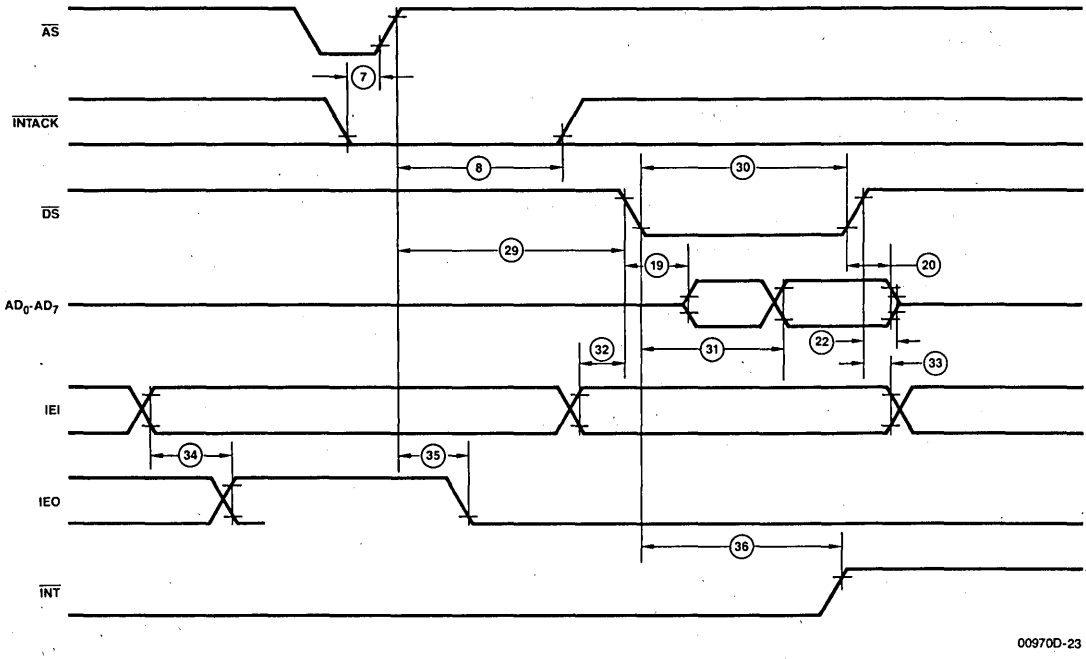


Figure 24. Reset Timing (AmZ8030)

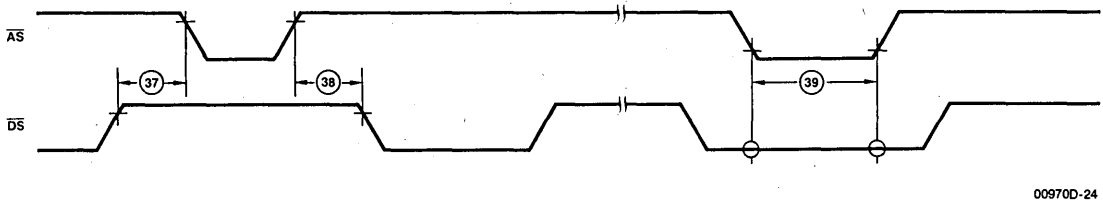
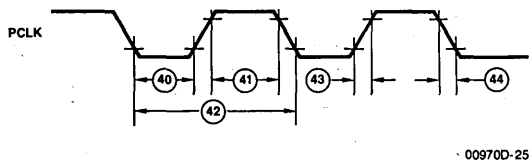


Figure 25. Cycle Timing (AmZ8030)

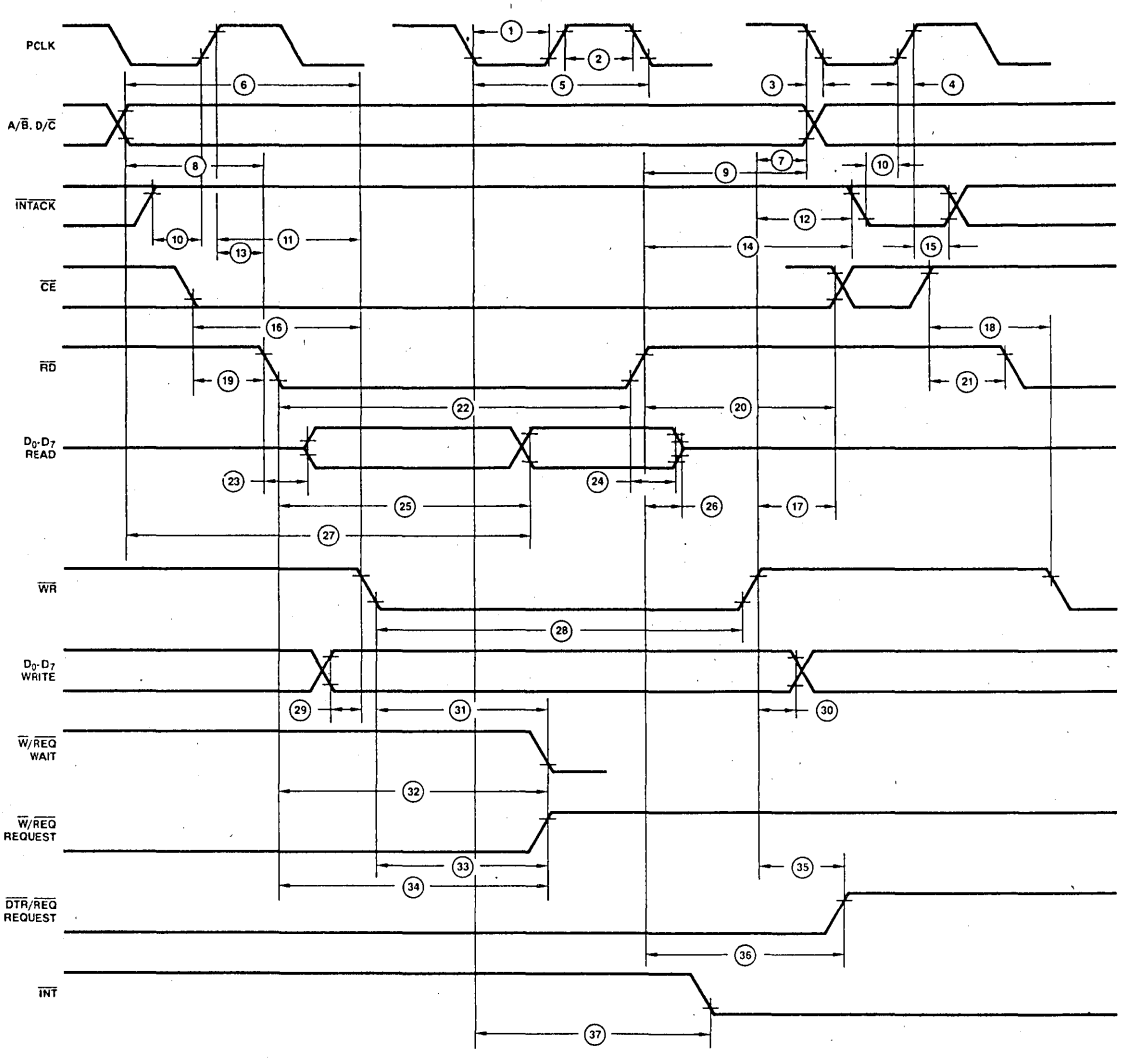


READ AND WRITE TIMING (AmZ8530)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TwPCl	PCLK Low Width	105	2000	70	1000	ns
2	TwPCh	PCLK High Width	105	2000	70	1000	ns
3	TfPC	PCLK Fall Time		20		10	ns
4	TrPC	PCLK Rise Time		20		15	ns
5	TcPC	PCLK Cycle Time	250	4000	165	2000	ns
6	TsA(WR)	Address to $\overline{WR}$ ↓ Setup Time	80		80		ns
7	ThA(WR)	Address to $\overline{WR}$ ↑ Hold Time	0		0		ns
8	TsA(RD)	Address to $\overline{RD}$ ↓ Setup Time	80		80		ns
9	ThA(RD)	Address to $\overline{RD}$ ↑ Hold Time	0		0		ns
10	TsIA(PC)	$\overline{INTACK}$ to PCLK ↑ Setup Time	0		0		ns
11	TsIAi(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↓ Setup Time (Note 1)	200		200		ns
12	ThIA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
13	TsIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ Setup Time (Note 1)	200		200		ns
14	ThIA(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↑ Hold Time	0		0		ns
15	ThIA(PC)	$\overline{INTACK}$ to PCLK ↑ Hold Time	100		100		ns
16	TsCEl(WR)	$\overline{CE}$ Low to $\overline{WR}$ ↓ Setup Time	0		0		ns
17	ThCE(WR)	$\overline{CE}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
18	TsCEh(WR)	$\overline{CE}$ High to $\overline{WR}$ ↓ Setup Time	100		70		ns
19	TsCEl(RD)	$\overline{CE}$ Low to $\overline{RD}$ ↓ Setup Time (Note 1)	0		0		ns
20	ThCE(RD)	$\overline{CE}$ to $\overline{RD}$ ↑ Hold Time (Note 1)	0		0		ns
21	TsCEh(RD)	$\overline{CE}$ High to $\overline{RD}$ ↓ Setup Time (Note 1)	100		70		ns
22	TwRDl	$\overline{RD}$ Low Width (Note 1)	390		250		ns
23	TdRD(DRA)	$\overline{RD}$ ↓ to Read Data Active Delay	0		0		ns
24	TdRDr(DR)	$\overline{RD}$ ↑ to Read Data Not Valid Delay	0		0		ns
25	TdRDf(DR)	$\overline{RD}$ ↓ to Read Data Valid Delay		250		180	ns
26	TdRD(DRz)	$\overline{RD}$ ↑ to Read Data Float Delay (Note 2)		70		45	ns

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float delay is defined as the time required for a ±0.5V change in the output with a maximum dc load and minimum ac load.

Figure 26. Read and Write Timing (AmZ8530)



00970D-26

INTERRUPT ACKNOWLEDGE TIMING, RESET TIMING, CYCLE TIMING (AmZ8530)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
27	TdA(DR)	Address Required Valid to Read Data Valid Delay		590		420	ns
28	TwWRI	$\overline{WR}$ Low Width	390		250		ns
29	TsDW(WR)	Write Data to $\overline{WR}$ ↓ Setup Time	0		0		ns
30	ThDW(WR)	Write Data to $\overline{WR}$ ↑ Hold Time	0		0		ns
31	TdWR(W)	$\overline{WR}$ ↓ to Wait Valid Delay (Note 4)		240		200	ns
32	TdRD(W)	$\overline{RD}$ ↓ to Wait Valid Delay (Note 4)		240		200	ns
33	TdWRf(REQ)	$\overline{WR}$ ↓ to $\overline{W/REQ}$ Not Valid Delay		240		200	ns
34	TdRDf(REQ)	$\overline{RD}$ ↓ to $\overline{W/REQ}$ Not Valid Delay		240		200	ns
35	TdWRr(REQ)	$\overline{WR}$ ↑ to $\overline{DTR/REQ}$ Not Valid Delay		5TcPC +300		5TcPC +250	ns
36	TdRDr(REQ)	$\overline{RD}$ ↑ to $\overline{DTR/REQ}$ Not Valid Delay		5TcPC +300		5TcPC +250	ns
37	TdPC(INT)	PCLK ↓ to $\overline{INT}$ Valid Delay (Note 4)		500		500	ns
38	TdIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ (Acknowledge) Delay (Note 5)					ns
39	TwRDA	$\overline{RD}$ (Acknowledge) Width	285		250		ns
40	TdRDA(DR)	$\overline{RD}$ ↓ (Acknowledge) to Read Data Valid Delay		190		180	ns
41	TsIEI(RDA)	IEI to $\overline{RD}$ ↓ (Acknowledge) Setup Time	120		100		ns
42	ThIEI(RDA)	IEI to $\overline{RD}$ ↑ (Acknowledge) Hold Time	0		0		ns
43	TdIEI(IEO)	IEI to IEO Delay Time		120		100	ns
44	TdPC(IEO)	PCLK ↑ to IEO Delay		250		250	ns
45	TdRDA(INT)	$\overline{RD}$ ↓ to $\overline{INT}$ Inactive Delay (Note 4)		500		500	ns
46	TdRD(WRQ)	$\overline{RD}$ ↑ to $\overline{WR}$ ↓ Delay for No Reset	30		15		ns
47	TdWRQ(RD)	$\overline{WR}$ ↑ to $\overline{RD}$ ↓ Delay for No Reset	30		30		ns
48	TwRES	$\overline{WR}$ and $\overline{RD}$ Coincident Low for Reset	250		250		ns
49	Trc	Valid Access Recovery Time (Note 3)	6TcPC +200		6TcPC +130		ns

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 TdIEIf(IEO) for each device separating them in the daisy chain.

•Timings are preliminary and subject to change.

Figure 27. Interrupt Acknowledge Timing (AmZ8530)

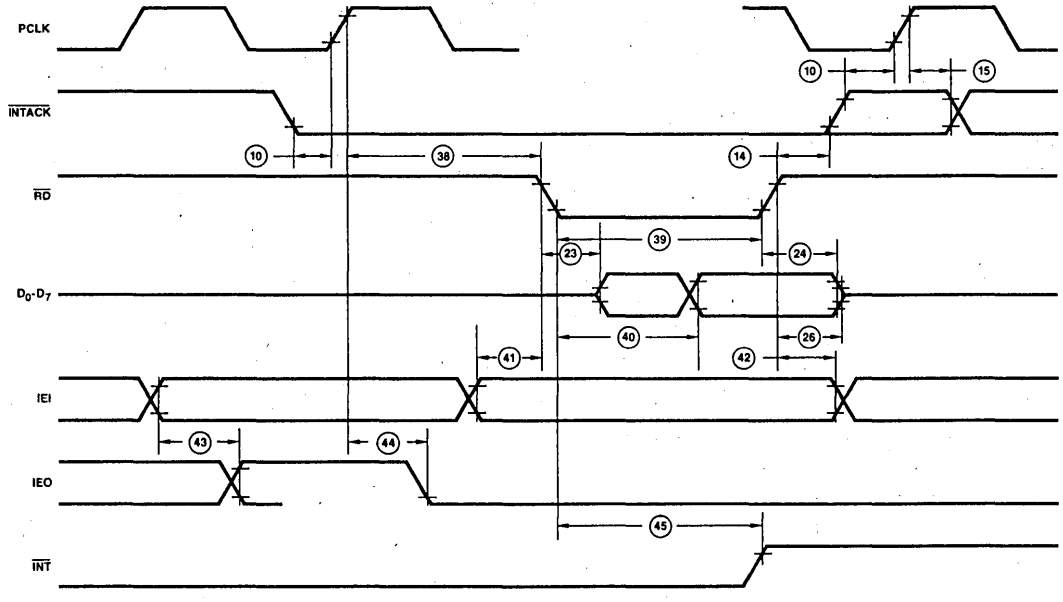


Figure 28. Reset Timing (AmZ8530)

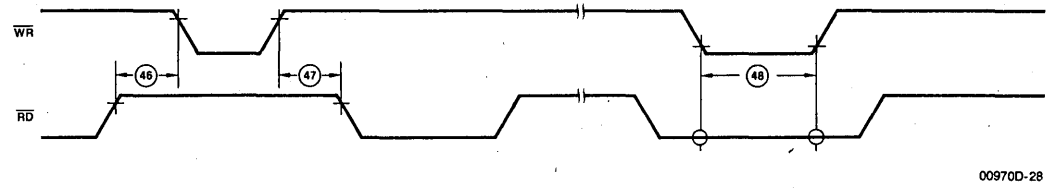
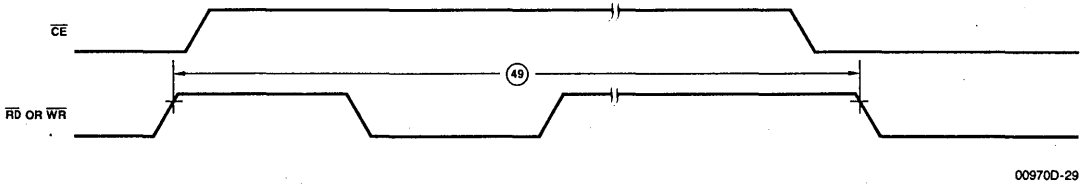


Figure 29. Cycle Timing (AmZ8530)



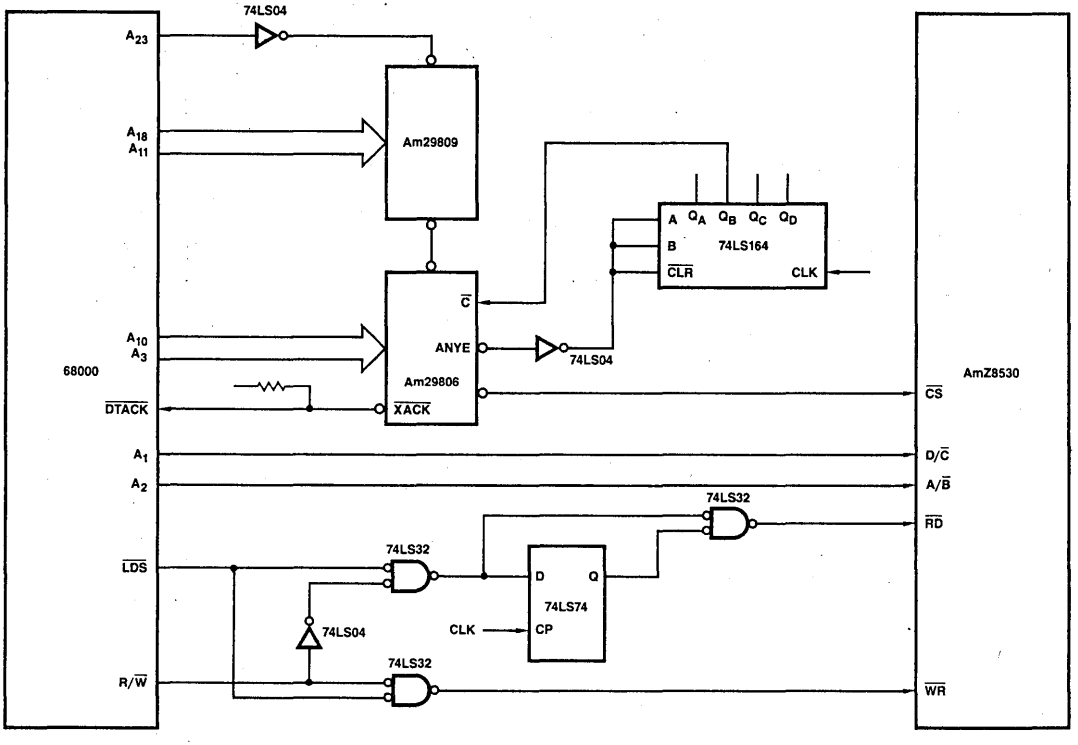
AmZ8530 SCC with Interrupt

With the addition of external hardware, the interrupt structure of the AmZ8530 could be used to enhance system performance. Only the circuitry that generates the appropriate interrupt control signals will be described here.

1. The 74LS138 3-to-8 decoder receives the FC_0, FC_1 and FC_2 processor status signals and decodes them into interrupt acknowledge INTA input to the AmZ8530. A_1, A_2 and A_3 designate the different interrupt levels that are decoded by the 74LS138. Note that INTA inhibits RD from the processor.
2. The 74LS164 shift register delays the RD signal sufficiently allowing sufficient time for the interrupt daisy chain to settle. In addition, the shift register produces a RD signal to get the vector on the bus prior to generating DTACK input to the 68000 CPU.
3. The 74LS148 encoder, encodes the interrupt request signals from the AmZ8530 and inputs the desired interrupt level to the 68000.

INTERFACE DEVICES

Address Decoder	29806	1
Address Comparator	29809	1
Inverters	74LS04	1
ORs	74LS32	1
Shift Registers	74LS74	2
Flip/Flop	74LS74	1
NOR Gates	74LS02	1
3-8 Decoder	74LS148	1
3-8 Encoder	74LS148	1

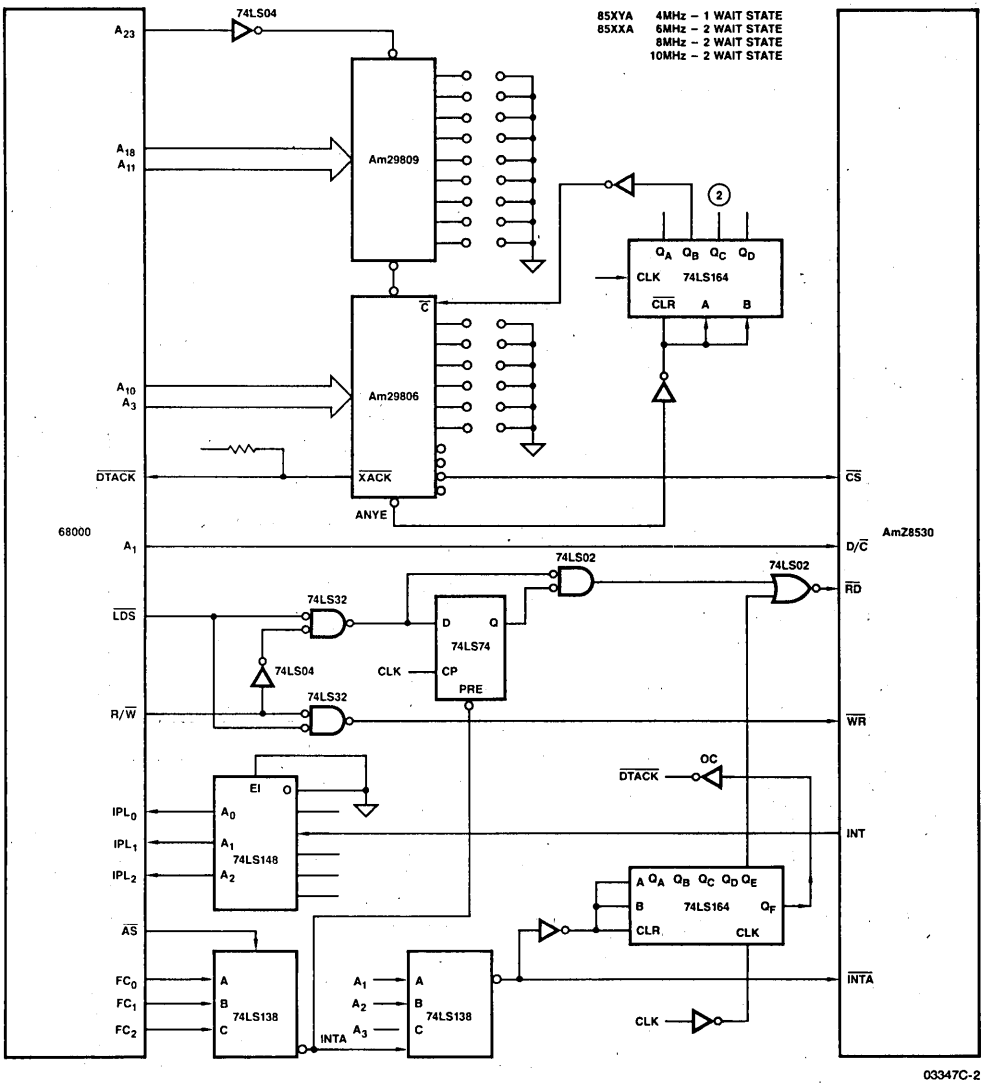


03347C-1

Why Design-In the 8530 SCC?

The 8530 Serial Communications Controller will probably compete against the MC6850, 6852 or 6854 in current 68000 system design. Motorola eventually plans to introduce more sophisticated serial I/Os some time in late 83. Until then, the AmZ8530 is the only SCC that can offer user programmable features for high-end applications.

- **Supports all advanced protocols:** HDLC, SDLC, IBM Bisync
- **Speed:** The AmZ8530 is the fastest available SCC with transmit rates of up to 1.5M bps.
- **Baud-Rate Generator:** The 8530 has it and the competition doesn't. This reduces chip count and cuts down on board real estate. Estimated cost savings of at least $10.00 per system. The baud rate is programmable and can be set to match the peripheral.
- **Built-In Digital Phase Lock Loop:** Another $10.00 savings per system. Otherwise, it must be implemented in 5-6 MSI or a PLA. The 8530 has the DPLL on-board, (while none of Motorola's devices do) which facilitates separation of data from clock in self clocking encoding schemes.
- **Auto Echo and Local Loopback:** Improves diagnostics by easy monitoring of transmitted data stream. None of the Motorola devices has this ability.



COMMUNICATIONS CONTROLLER COMPARISON

Features	AmZ8030 SCC	Z-80 SIO	Z-80 DART	MC6850 AIA
Supply	+5V,0	+5V,0	+5V,0	+5V,0
Package	40-Pin DIP	40-Pin DIP	40-Pin DIP	24-Pin DIP
Technology	N-Channel MOS	N-Channel MOS	N-Channel MOS	N-Channel MOS
Clocks	One	One	One	Two
Channels	Two	Two	Two	One
Transmission Rates	1.5M Bits/sec	800K Bits/sec	800K Bits/sec	1M Bits/sec
Min Clock Cycle Time	165nsec	250nsec	250nsec	500nsec
Error Protection	CRC, CCITT	CRCC, CCITT	CRC, CCITT	Parity
Interrupt Capability	Programmable	Programmable	Programmable Vector Interrupt	Non-Programmable
B. R. Generator	On-Chip, Programmable	No	No	No
Digital PLL	On-Chip	No	No	No
Modem Control	Handshake Interface	Handshake Interface	Handshake Interface	Handshake Interface
Abort Signals	Programmable	No	No	No
Protocols	Asynchronous, Synchronous: Byte Oriented (IBM BiSync) Bit Oriented (HDLC, SDLC)	Asynchronous	Asynchronous	Asynchronous
Auto Echo	Yes	No	No	No
Local Loopback	Yes, for Local or Remote Maintenance	No	No	No

COMMUNICATIONS CONTROLLER COMPARISON

Features	MC6852 SSD	MC6854 ADL	8251 PCI	8273 PPC
Supply	+5V,0	+5V,0	+5V,0	+5V,0
Package	24-Pin DIP	28-Pin DIP	28-Pin DIP	40-Pin DIP
Technology	N-Channel MOS	N-Channel MOS	N-Channel MOS	N-Channel MOS
Clocks	Two	Two	One	One
Channels	One	One	One	One
Transmission Rates	?	?	64K Bits/sec	64K Bits/sec
Min Clock Cycle Time	280nsec	500nsec	250nsec	250nsec
Error Protection	Parity	CRC	Parity	CRC
Interrupt Capability	Non-Programmable	Non-Programmable	No	Programmable
B. R. Generator	No	No	No	No
Digital PLL	No	No	No	On Chip
Modem Control	Handshake Interface	Handshake Interface	Handshake Interface	Handshake Interface
Abort Signals	No	Programmable	No	No
Protocols	Synchronous: Bit Oriented (HDLC, SDLC)	Synchronous: Bit Oriented (HDLC, SDLC)	Asynchronous, Synchronous: Byte Oriented (IBM BiSync)	Asynchronous, Synchronous: Bit Oriented (HDLC, SDLC)
Auto Echo	No	No	No	No
Local Loopback	No	No	No	No

AmZ8031 • AmZ8531 (ASCC)

Asynchronous Serial Communications Controller

DISTINCTIVE CHARACTERISTICS

- **Two 1M.bps full duplex serial channels**
  Each channel has independent oscillator, generator, and PLL for clock recovery, dramatically reducing the need for external components.
- **Programmable protocols**
  NRZ, NRZI, and FM data encoding supported under program control.
- **Programmable Asynchronous Modes**
  5 to 8 bit characters with programmable stop bits clock break detect, and error conditions.
- **Z8000* compatible**
  The Z8031 interfaces directly to the Z8000 CPU bus and to the Z8000 interrupt structure.
- **Compatible with non-multiplexed bus**
  The Z8531 interfaces easily to most other CPUs.

GENERAL DESCRIPTION

Asynchronous Serial Communications Controllers are dual-channel communications peripherals designed for use with 8- and 16-bit microprocessors. They function as serial-to-parallel, and parallel-to-serial converter/controllers, and contain a variety of new, sophisticated internal functions, including on-chip baud rate generators, digital phase-locked loops and crystal oscillators, to dramatically reduce the need for external circuitry.

Both channels have facilities for modem control; in cases where these controls aren't needed, they can be used for general purpose I/O.

The AmZ8031 is directly compatible with the Z8000 and 8086 CPUs, while the AmZ8531 is designed for non-multiplexed buses, and is easily interfaced with most other CPUs such as 8080, Z80, 6800, 68000 and Multibus.**

BLOCK DIAGRAM

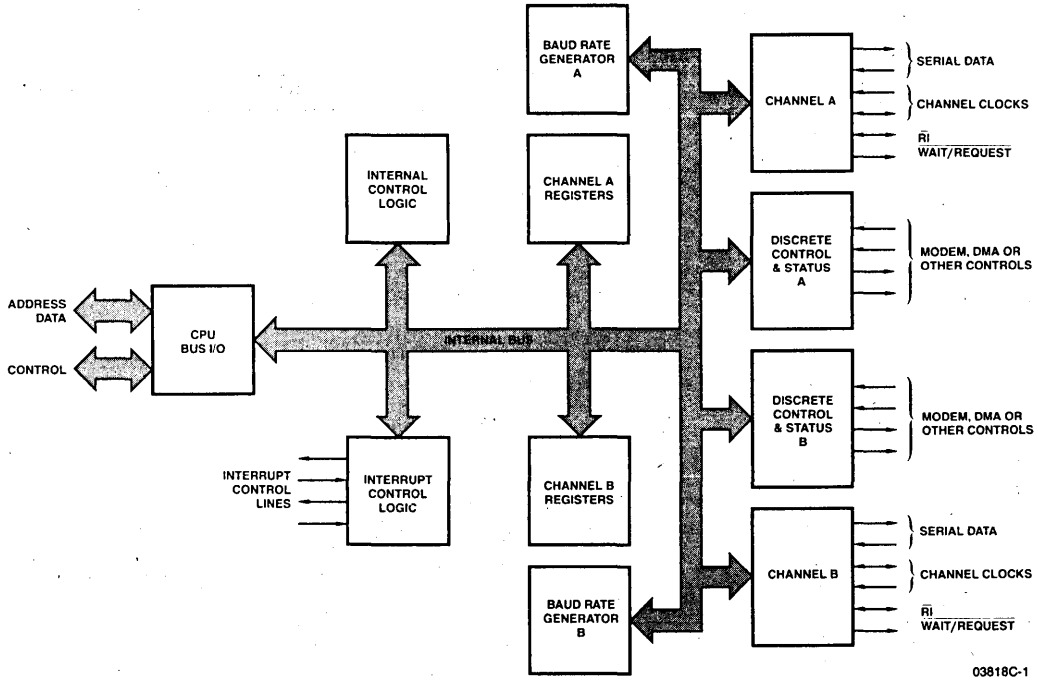
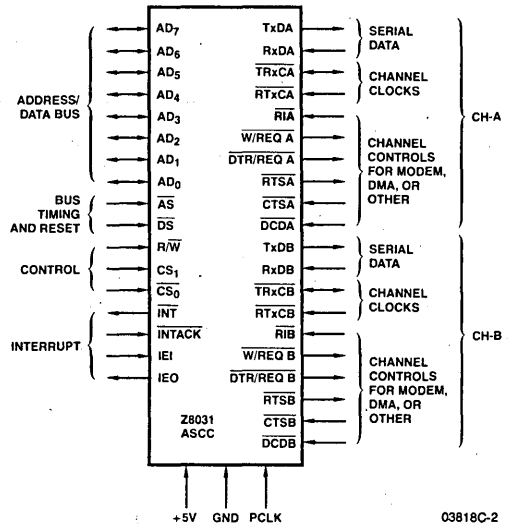


Figure 1.

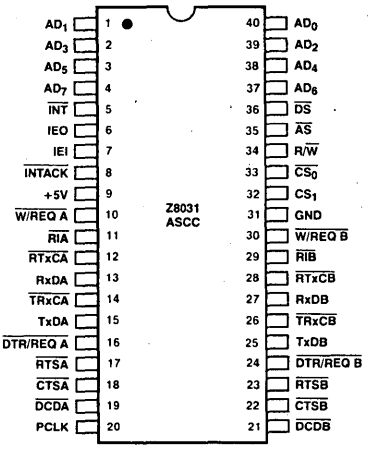
*Z8000 is a trademark of Zilog. **Multibus is a trademark of Intel.

Z8031 Logic Symbol



03818C-2

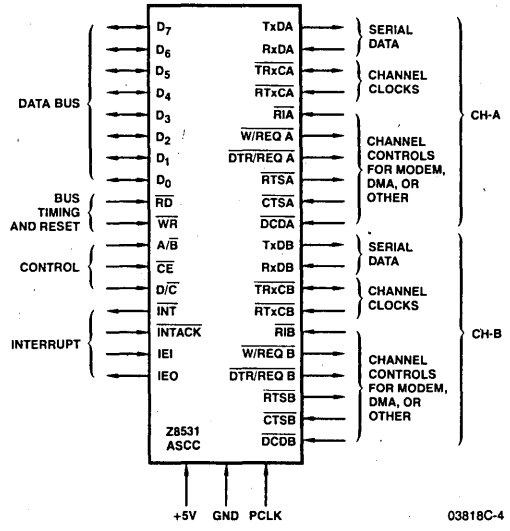
Z8031 Connection Diagram – Top View
D-40-2, P-40-1



03818C-3

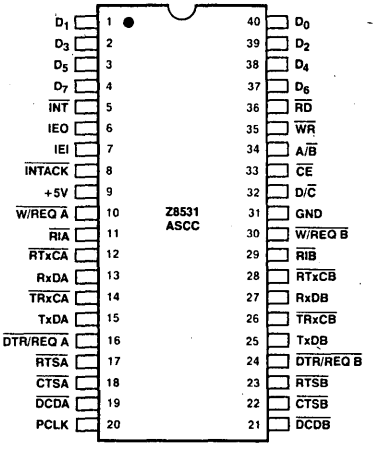
Note: Pin 1 is marked for orientation.

Z8531 Logic Symbol



03818C-4

Z8531 Connection Diagram – Top View
D-40-2, P-40-1



03818C-5

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Example Z8031 A D C B

- Basic Device
- Speed: Blank = 4MHz, A = 6MHz
- Package Type: D = Hermetic DIP, P = Molded Plastic
- Temperature Range: C = 0 to +70°C, I = −40 to +85°C, M = −55 to +125°C
- Burn-in

Example Z8531 A D C B

- Basic Device
- Speed: Blank = 4MHz, A = 6MHz
- Package Type: D = Hermetic DIP, P = Molded Plastic
- Temperature Range: C = 0 to +70°C, I = −40 to +85°C, M = −55 to +125°C
- Burn-in

ARCHITECTURE

The ASCC internal structure includes two full-duplex channels, two baud rate generators, internal control and interrupt logic, and a bus interface to the Z8000 CPU (AmZ8031) or to a non-multiplexed CPU bus (AmZ8531). 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 1).

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 synchronous 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 (A only), and one containing the Interrupt Pending bits (A only).

The registers for each channel are designated as follows:

WR0-WR15 – Write Registers 0 through 15.

RR0-RR3, RR10, RR12, RR13, RR15 – Read Registers 0 through 3, 10, 12, 13, 15.

The following table lists the functions assigned to each read or write register. The ASCC 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 2 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 an 11-bit transmit shift register that can be loaded 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).

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, shift right/shift left command
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

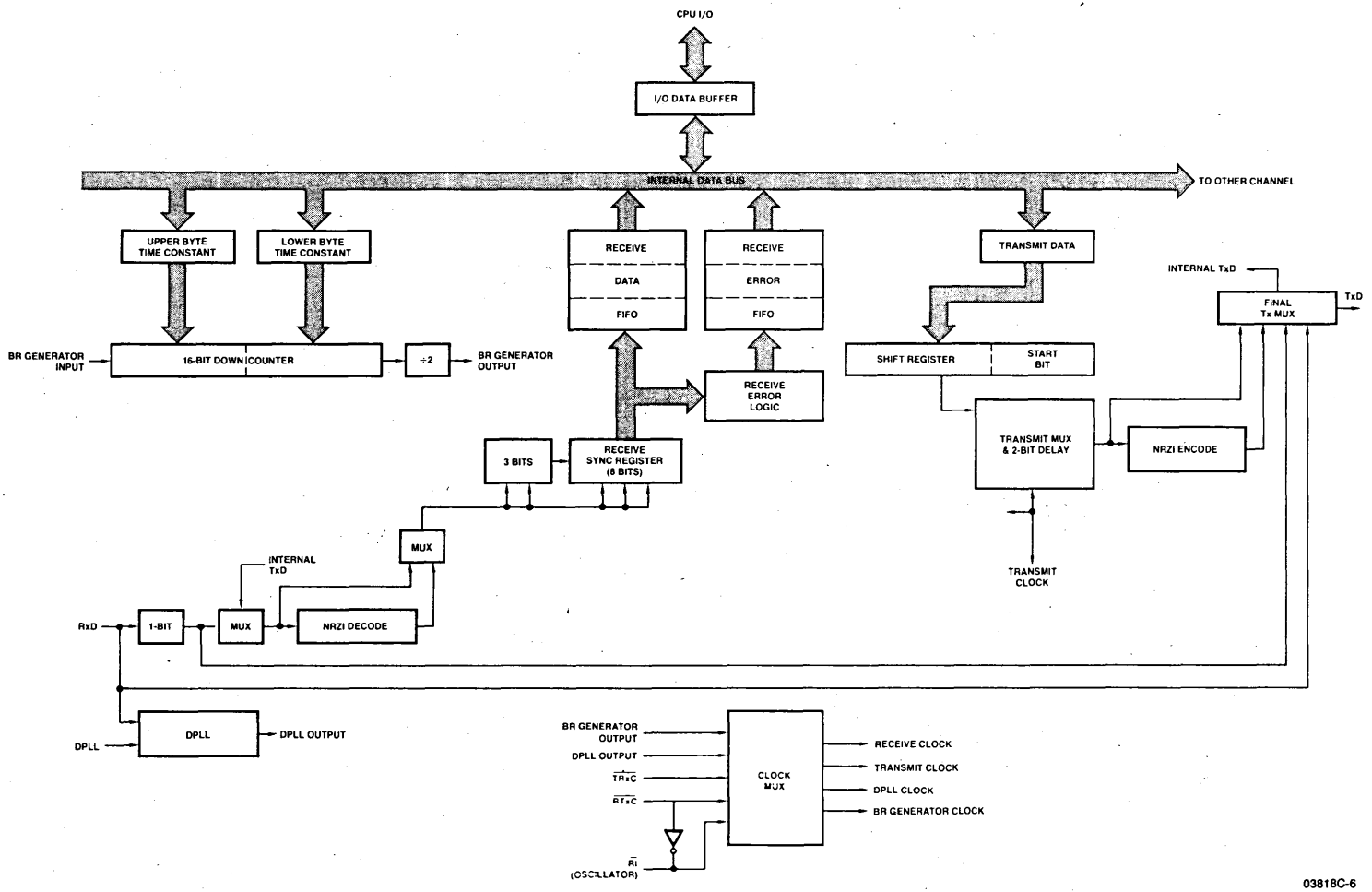


Figure 2. Data Path

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, it interacts with the CPU and provides vectored interrupts and handshaking signals.

DATA COMMUNICATIONS CAPABILITIES

The ASCC provides two independent full-duplex channels programmable for use in any common asynchronous data-communication protocol. Figure 3 and the following description briefly detail this protocol.

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 14). 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 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 ASCC 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.

BAUD RATE GENERATOR

Each channel in the ASCC 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 zero, 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 $\overline{TRxC}$ pin, the output of the baud rate generator may be echoed out via the $\overline{TRxC}$ pin.

The following formula relates the time constant to the baud rate. (The baud rate is in bits/second and the BR clock period is in seconds.)

$$\text{baud rate} = \frac{1}{2\ (\text{time constant} + 2) \times (\text{BR clock period})}$$

Time-Constant Values for Standard Baud Rates at BR Clock = 3.9936MHz

Rate (Baud)	Time Constant (decimal notation)	Error
19200	102	–
9600	206	–
7200	275	0.12%
4800	414	–
3600	553	0.06%
2400	830	–
2000	996	0.04%
1800	1107	0.03%
1200	1662	–
600	3326	–
300	6654	–
150	13310	–
134.5	14844	0.0007%
110	18151	0.0015%
75	26622	–
50	39934	–

DIGITAL PHASE-LOCKED LOOP

The ASCC 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 ASCC 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 1/0 or 0/1). 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

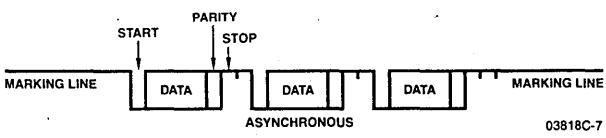


Figure 3. ASCC Protocols

DPLL looks for edges only during a time centered on the 15/16 counting transition.

The 32X clock for the DPLL can be programmed to come from either the $\overline{RTxC}$ input or the output of the baud rate generator. The DPLL output may be programmed to be echoed out of the ASCC via the $\overline{TRxC}$ pin (if this pin is not being used as an input).

DATA ENCODING

The ASCC may be programmed to encode and decode the serial data in four different ways. 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 FM_1 (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 FM_0 (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 ASCC 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 0/1 the bit is a 0. If the transition is 1/0 the bit is a 1.

AUTO ECHO AND LOCAL LOOPBACK

The ASCC 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 $\overline{CTS}$ 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 $\overline{WAIT/REQUEST}$ on transmit.

The ASCC 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 with NRZ, NRZI or FM coding of the data stream.

I/O INTERFACE CAPABILITIES

The ASCC 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 ASCC are automatically updated whenever any function is performed. 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 ASCC 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 6 and 7).

To speed interrupt response time, the ASCC 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 ASCC (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

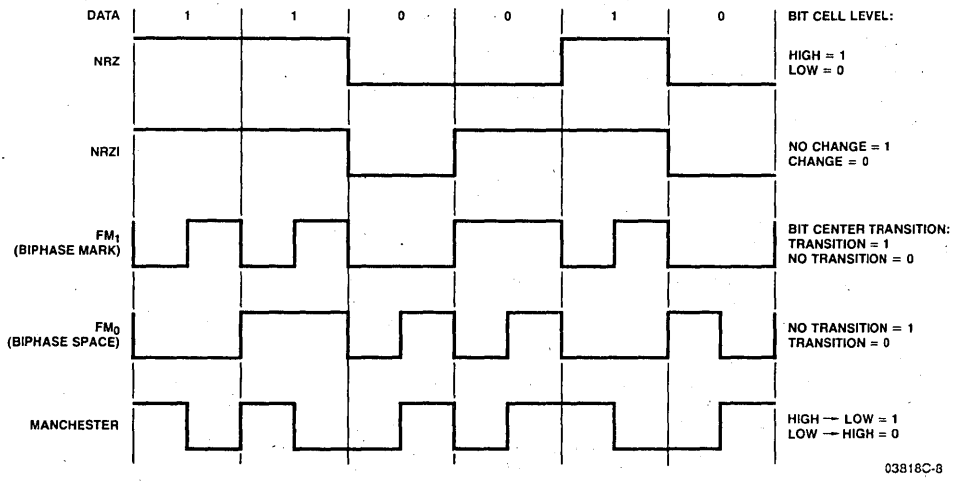


Figure 4. Data Encoding Methods

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 Z-Bus interrupt priority chain (Figure 5). As a Z-Bus peripheral, the ASCC 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 A/D bus.

In the ASCC, the 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 ASCC, 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 ASCC and external to the ASCC 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 ASCC 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 interrupts. 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 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 $\overline{CTS}$, $\overline{DCD}$, and $\overline{RI}$ 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).

CPU/DMA BLOCK TRANSFER

The ASCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the $\overline{WAIT}/\overline{REQUEST}$ output in conjunction with the Wait/Request bits in WR1. The $\overline{WAIT}/\overline{REQUEST}$ output can be defined under software control as a $\overline{WAIT}$ line in the CPU Block Transfer mode or as a $\overline{REQUEST}$ line in the DMA Block Transfer mode.

To a DMA controller, the ASCC $\overline{REQUEST}$ output indicates that the ASCC is ready to transfer data to or from memory. To the CPU, the $\overline{WAIT}$ line indicates that the ASCC is not ready to transfer data, thereby requesting that the CPU extend the I/O cycle. The $\overline{DTR}/\overline{REQUEST}$ line allows full-duplex operation under DMA control.

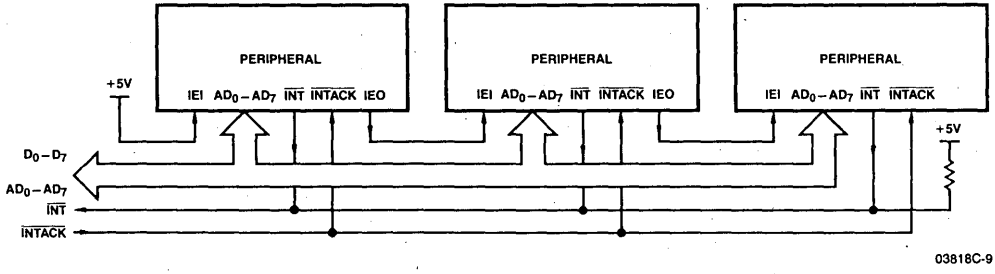


Figure 5. Z-Bus Interrupt Schedule

PROGRAMMING (AmZ8031)

The AmZ8031 contains 11 write registers in each channel that are programmed by the system separately to configure the functional personality of the channels.

All of the registers in the AmZ8031 are directly addressable. How the AmZ8031 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 $\overline{A/B}$ is taken from AD_0 and the state of AD_5 is ignored. In the shift left mode, $\overline{A/B}$ is taken from AD_5 and the state of AD_0 is ignored. AD_7 and AD_6 are always ignored as address bits and the register address itself occupies $AD_4 - AD_1$.

The system program first issues a series of commands to initialize the basic mode of operation. For example, the 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 (AmZ8531)

The AmZ8531, register addressing is direct for the data registers only, which are selected by a High on the D/$\overline{C}$ pin. In all other cases (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 registers in the AmZ8531, 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.

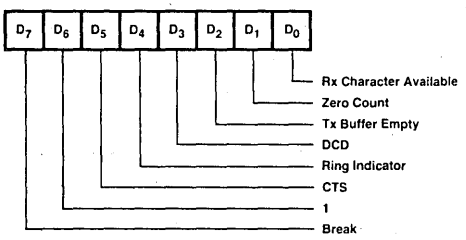
The system program first issues a series of commands to initialize the basic mode of operation. For example, the character length, clock rate, number of stop bits, even or odd parity might be set first. Then the interrupt mode would be set, and finally, receiver or transmitter enable.

READ REGISTERS

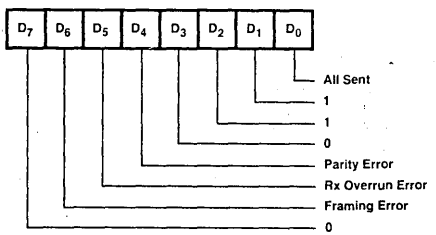
The ASCC contains 8 read registers (actually 9, counting the receive buffer (RR8) in each channel). Four of these may be read to obtain status information (RR0, RR1, RR10 and RR15). Two registers (RR12 and RR13) may be read to learn the baud rate generator time constant. RR2 contains either the unmodified interrupt vector (Channel A) or the vector modified by status information (Channel B). RR3 contains the Interrupt Pending (IP) bits (Channel A). Figure 6 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).

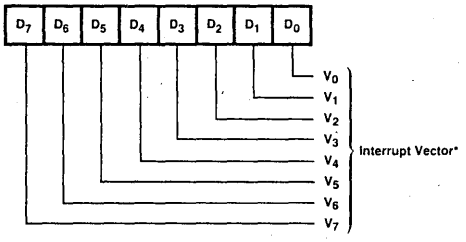
Read Register 0



Read Register 1

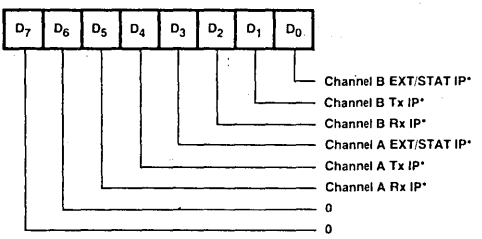


Read Register 2



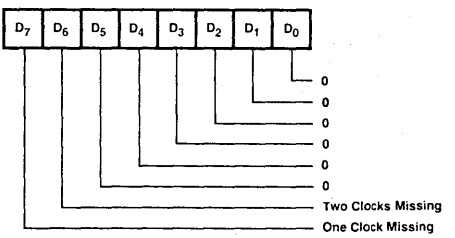
*Modified In B Channel

Read Register 3



*Always 0 in B Channel

Read Register 10



Read Register 12

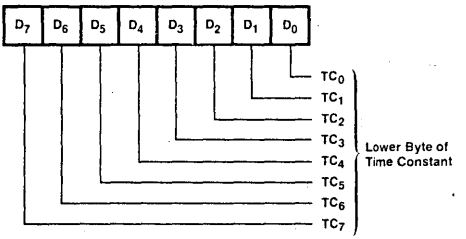
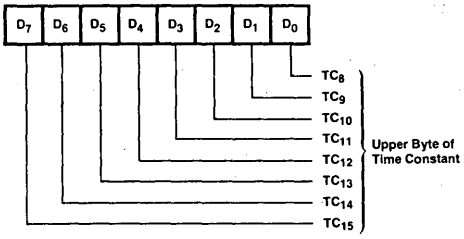


Figure 6. Read Register Bit Functions

Read Register 13



Read Register 15

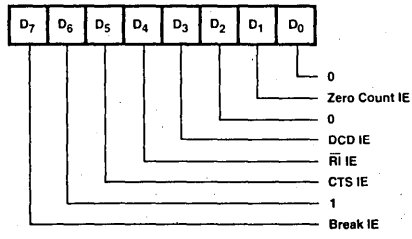
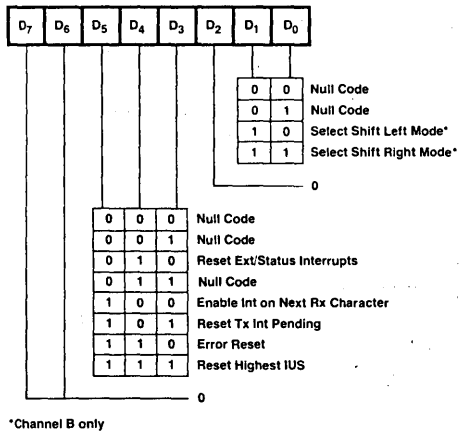


Figure 6. Read Register Bit Functions (Cont.)

WRITE REGISTERS

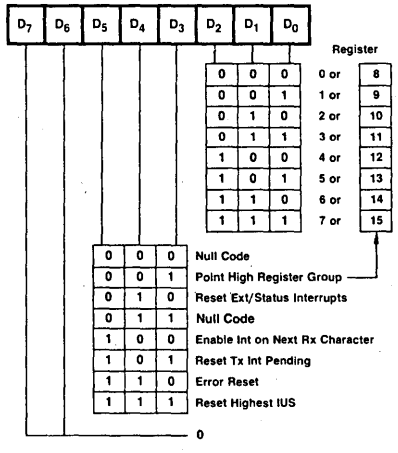
The ASCC contains 11 write registers (12 counting WR8, the transmit buffer) in each channel. These write registers are programmed separately to configure the functional "personality" of the channels. In addition, there are two registers (WR2 and WR9) shared by the two channels that may be accessed through either of them. WR2 contains the interrupt vector for both channels, while WR9 contains the interrupt control bits. Figure 7 shows the format of each write register.

Write Register 0 (AmZ8031)

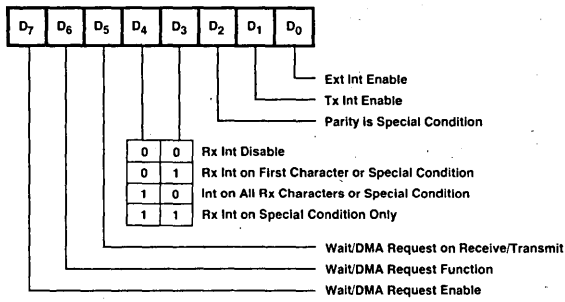


*Channel B only

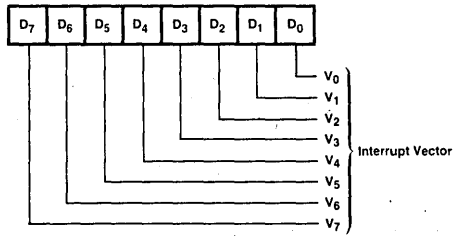
Write Register 0 (AmZ8531)



Write Register 1



Write Register 2



Write Register 3

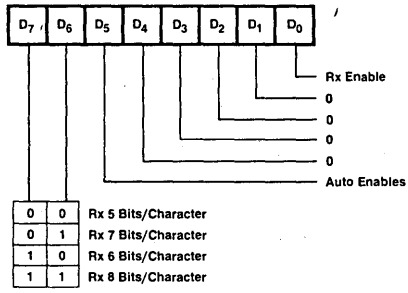
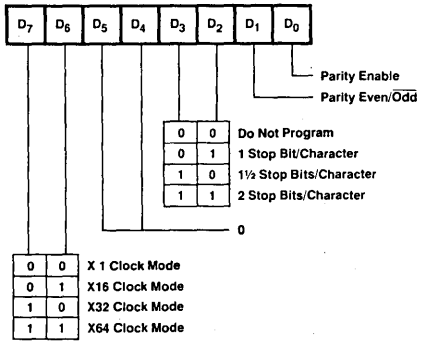
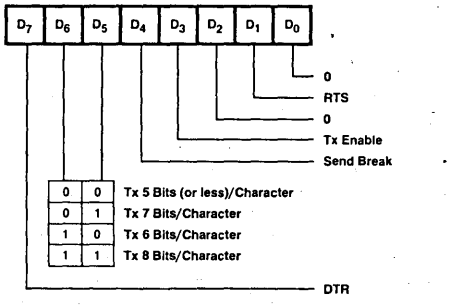


Figure 7. Write Register Bit Functions

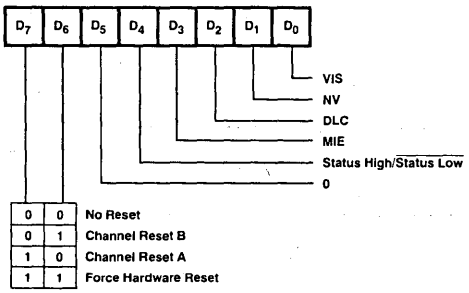
Write Register 4



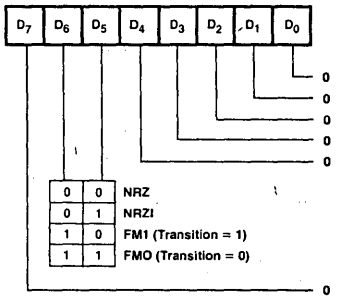
Write Register 5



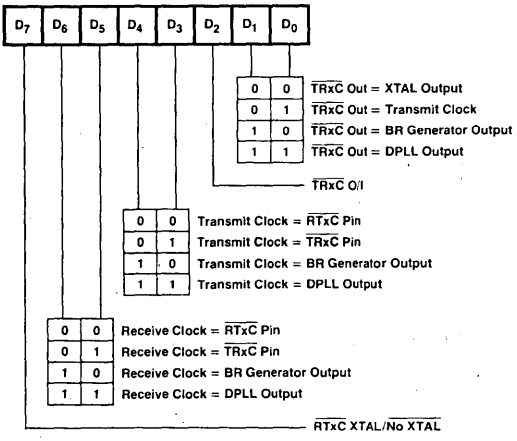
Write Register 9



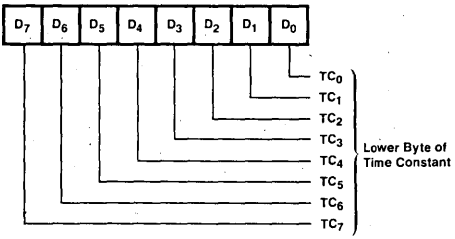
Write Register 10



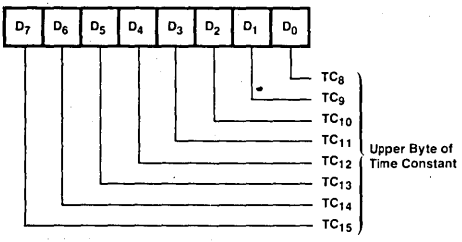
Write Register 11



Write Register 12



Write Register 13



Write Register 14

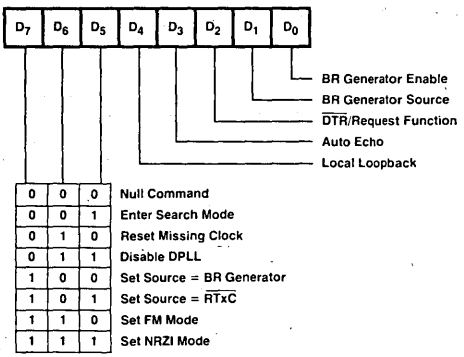


Figure 7. Write Register Bit Functions (Cont.)

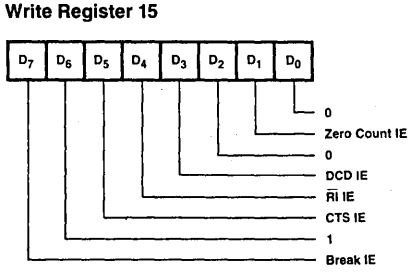


Figure 7. Write Register Bit Functions (Cont.)

AmZ8031 TIMING

The ASCC 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 ASCC to the falling edge of $\overline{DS}$ in the second transaction involving the ASCC. This time must be at least 6 PCLK cycles plus 200ns.

READ CYCLE TIMING

Figure 8 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/W must be High to indicate a read cycle. CS_1 must also be High for the read cycle to occur. The data bus drivers in the ASCC are then enabled while $\overline{DS}$ is Low.

WRITE CYCLE TIMING

Figure 9 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. CS_1 must be High for the write cycle to occur. $\overline{DS}$ Low strobes the data into the ASCC.

INTERRUPT ACKNOWLEDGE CYCLE TIMING

Figure 10 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 addess and $\overline{CS_0}$ are ignored. The state of R/$\overline{W}$ 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 ASCC and IEI is High when DS falls, the acknowledge cycle was intended for the ASCC. In this case, the ASCC may be programmed to respond to $\overline{DS}$ Low by placing its interrupt vector on AD_0-AD_7. It then sets the appropriate interrupt-under-service latch internally.

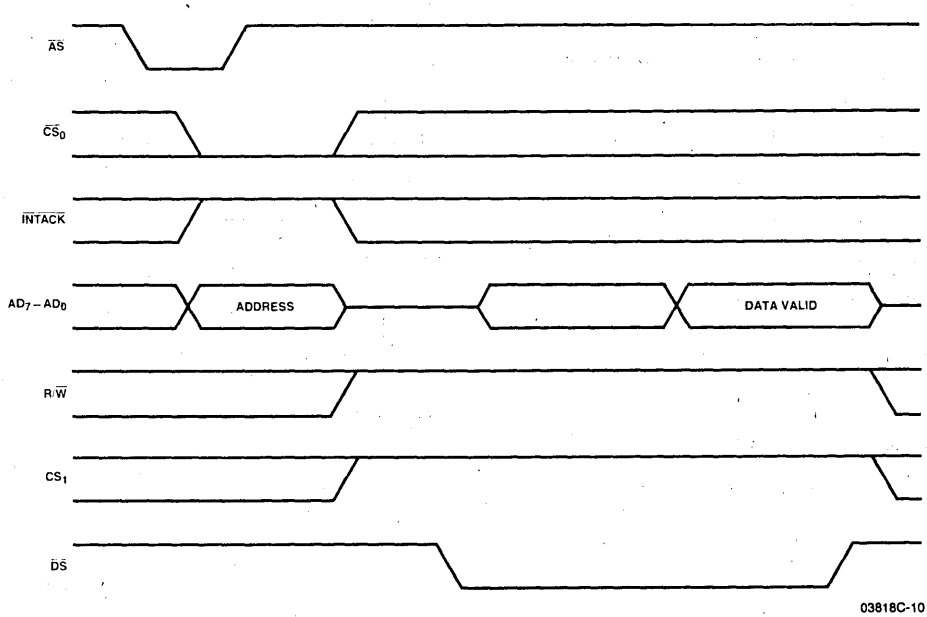


Figure 8. Z8031 Read Cycle Timing

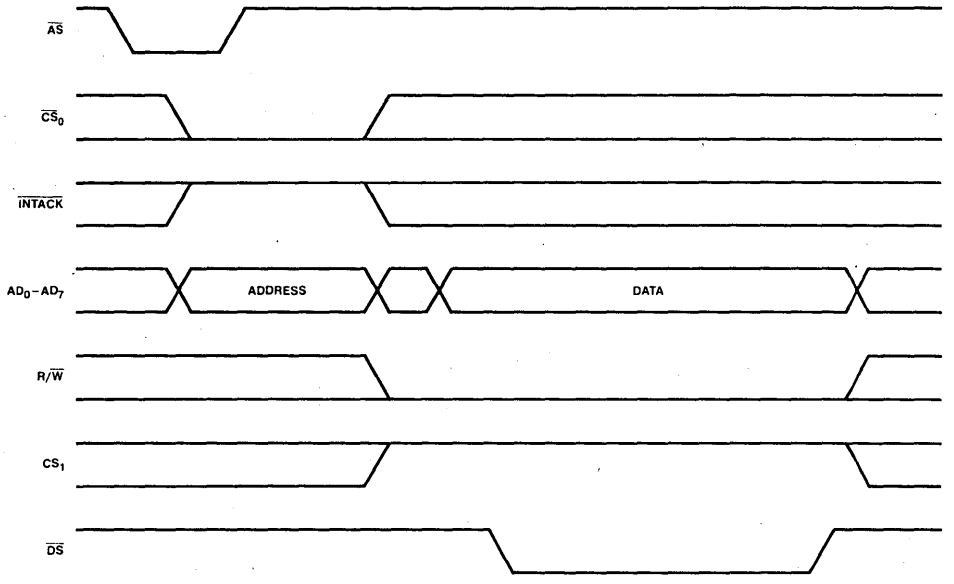


Figure 9. Z8031 Write Cycle Timing

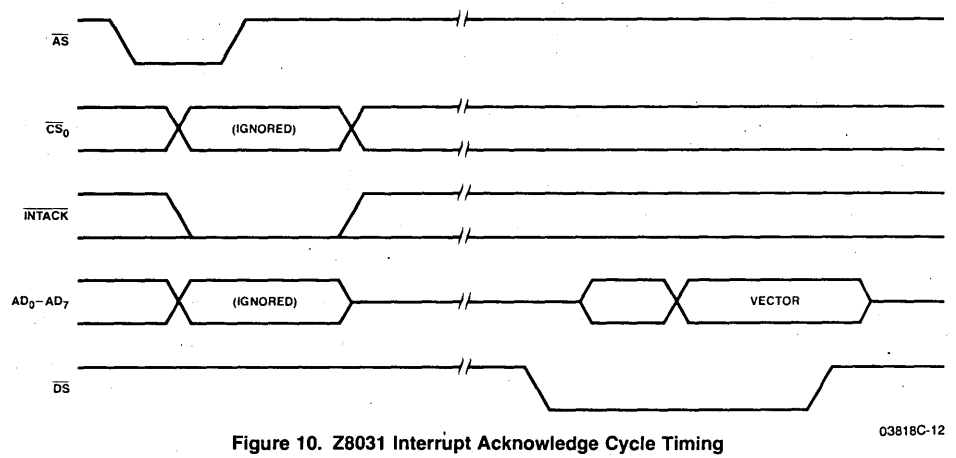


Figure 10. Z8031 Interrupt Acknowledge Cycle Timing

AmZ8531 TIMING

The ASCC 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 ASCC. The recovery time required for proper operation is specified from the rising edge of $\overline{WR}$ or $\overline{RD}$ in the first transaction involving the ASCC to the falling edge of $\overline{WR}$ or $\overline{RD}$ in the second transaction involving the ASCC. This time must be at least 6 PCLK cycles plus 200ns.

READ CYCLE TIMING

Figure 11 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.

WRITE CYCLE TIMING

Figure 12 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.

INTERRUPT ACKNOWLEDGE CYCLE TIMING

Figure 13 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 ASCC and IEI is High when $\overline{RD}$ falls, the acknowledge cycle is intended for the ASCC. In this case, the ASCC may be programmed to respond to $\overline{RD}$ Low by placing its interrupt vector on D_0-D_7 it then sets the appropriate interrupt-under-service latch internally.

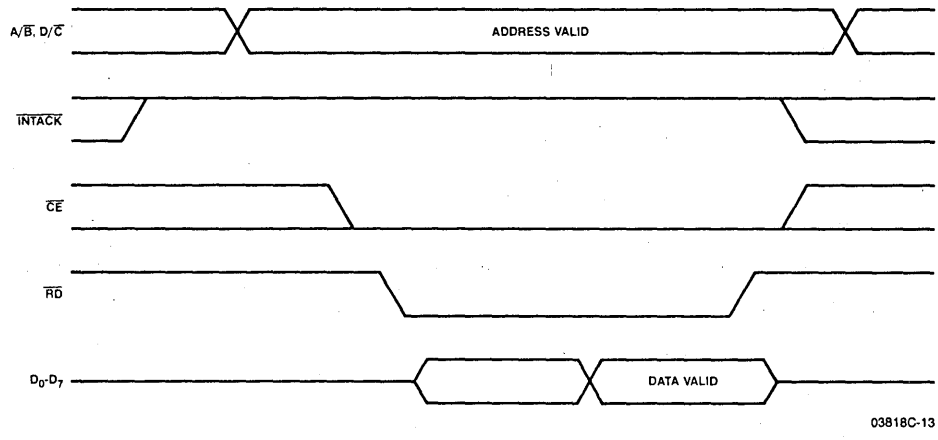


Figure 11. Z8531 Read Cycle Timing

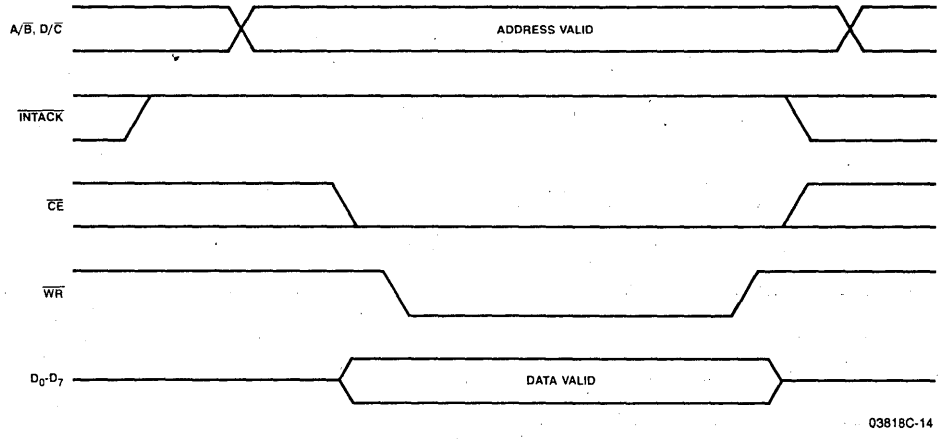


Figure 12. Z8531 Write Cycle Timing

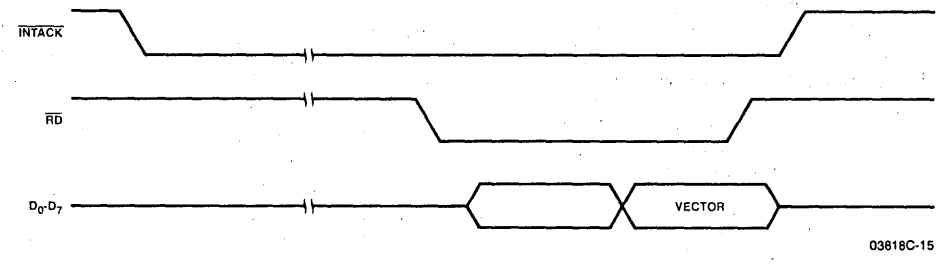


Figure 13. Z8531 Interrupt Acknowledge Cycle Timing

PIN DESCRIPTION FOR AmZ8031

The following section describes the pin functions of the ASCC. Figures 14 and 15 detail the respective pin functions and pin assignments.

V_{CC}: +5V Power Supply

GND: Ground

$AD_0 - AD_7$: Address/Data Bus (bidirectional, active High, three-state).
These multiplexed lines carry register addresses to the ASCC as well as data or control information to and from the ASCC.

$\overline{AS}$: Address Strobe (input, active Low).
Addresses on $AD_0 - AD_7$ are latched by the rising edge of this signal.

$\overline{CS_0}$: Chip Select 0 (input, active Low).
This signal is latched concurrently with the addresses on $AD_0 - AD_7$ and must be active for the intended bus transaction to occur.

CS_1: Chip Select 1 (input, active High).
This second select signal must also be active before the intended bus transaction can occur. CS_1 must remain active throughout the transaction.

$\overline{CTSA}$, $\overline{CTSB}$: Clear to Send (inputs, active Low).
If these pins are programmed as Auto Enables, a Low on these inputs enables their respective transmitter. 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 ASCC detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

$\overline{DCDA}$, $\overline{DCDB}$: Data Carrier Detect (inputs, 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 ASCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

$\overline{DS}$: Data Strobe (input, active Low).
This signal provides timing for the transfer of data into and out of the ASCC. If AS and DS coincide, this is interpreted as a reset.

$\overline{DTR}/\overline{REQA}$, $\overline{DTR}/\overline{REQB}$: 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 ASCC interrupt or the ASCC 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.

$\overline{INT}$: Interrupt Request (output, open-drain, active Low).
This signal is activated when the ASCC requests an interrupt.

$\overline{INTACK}$: Interrupt Acknowledge (input, active Low).
This signal indicates an active interrupt acknowledge cycle. During this cycle, the ASCC interrupt daisy chain settles. When $\overline{DS}$ becomes active, the ASCC places an interrupt vector on the data bus (if IEI is High). $\overline{INTACK}$ is latched by the rising edge of $\overline{AS}$.

PCLK: Clock (input).
This is the master ASCC clock used to synchronize internal signals. PCLK is not required to have any phase relationship with the master system clock, although the frequency of this clock must be at least 90% of the CPU clock frequency for a Z8000. PCLK is a TTL level signal.

RxDA, RxDB: Receive Data (inputs, active High).
These input signals receive serial data at standard TTL levels.

$\overline{RTxCA}$, $\overline{TRxCB}$: 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 of the digital phase-locked loop. These pins can also be programmed for use with the respective $\overline{RI}$ pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

$\overline{RTSA}$, $\overline{RTSB}$: Request to Send (outputs, active Low).
When the Request to Send (RTS) bit in Write Register 5 (Figure 7) is set, the RTS signal goes Low. When the $\overline{RTS}$ bit is reset and Auto Enable is on, the signal goes High after the transmitter is empty. With Auto Enable off, the $\overline{RTS}$ pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

R/$\overline{W}$: Read/Write (input).
This signal specifies whether the operation to be performed is read or a write.

$\overline{RIA}$, $\overline{RIB}$: Ring Indicator (inputs, active Low).
These pins can act either as inputs or as part of the crystal oscillator circuit.

In normal operation (crystal oscillator option not selected), these pins are inputs similar to $\overline{CTS}$ and $\overline{DCD}$. In this mode, transitions on these lines affect the state of the Ring Indicator status bits in Read Register 0 (Figure 6) but have no other function.

TxDA, TxDB: Transmit Data (outputs, active High).
These output signals transmit serial data at standard TTL levels.

$\overline{TRxCA}$, $\overline{TRxCB}$: Transmit/Receive Clocks (inputs or outputs, active Low).
These pins can be programmed in several different modes of operation. $\overline{TRxC}$ 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.

$\overline{W}/\overline{REQA}$, $\overline{W}/\overline{REQB}$: 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 ASCC data rate. The reset state is Wait.

PIN DESCRIPTION FOR AmZ8531

The following section describes the pin functions of the ASCC. Figures 16 and 17 detail the respective pin functions and pin assignments.

V_{CC}: +5V Power Supply.

GND: Ground.

A/$\overline{B}$: Channel A/Channel B Select (input).
This signal selects the channel in which the read or write operation occurs.

$\overline{CE}$: Chip Enable (input, active Low).
This signal selects the ASCC for a read or write operation.

$\overline{CTSA}$, $\overline{CTSB}$: 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 ASCC detects pulses on these inputs and can interrupt the CPU on both logic level transitions.

D/$\overline{C}$: Data/Control Select (input).
This signal defines the type of information transferred to or from the ASCC. A High means data is transferred; a Low indicates a command.

$\overline{DCDA}$, $\overline{DCDB}$: Data Carrier Detect (inputs, 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 accomodate slow rise time signals. The ASCC detects pulses on these pins and can interrupt the CPU on both logic level transitions.

$D_0 - D_7$: Data Bus (bidirectional, three-state).
These lines carry data and commands to and from the ASCC.

$\overline{DTR}$/$\overline{REQA}$, $\overline{DTR}$/$\overline{REQB}$: 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 ASCC interrupt or the ASCC 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.

$\overline{INT}$: Interrupt Request (output, open-drain, active Low).
This signal is activated when the ASCC requests an interrupt.

$\overline{INTACK}$: Interrupt Acknowledge (input, active Low).
This signal indicates an active interrupt acknowledge cycle. During this cycle, the ASCC interrupt daisy chain settles. When $\overline{RD}$ becomes active, the ASCC 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 ASCC clock used to synchronize internal signals. PCLK is a TTL level signal.

$\overline{RD}$: Read (input, active Low).
This signal indicates a read operation and when the ASCC is selected, enables the ASCC's bus drivers. During the interrupt acknowledge cycle, this signal gates the interrupt vector onto the bus if the ASCC is the highest priority device requesting an interrupt.

RxDA, RxDB: Receive Data (inputs, active High).
These input signals receive serial data at standard TTL levels.

$\overline{RTxCA}$, $\overline{RTxCB}$: 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 $\overline{RI}$ pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

$\overline{RTSA}$, $\overline{RTSB}$: Request To Send (outputs, active Low).
When the Request To Send (RTS) bit in Write Register 5 (Figure 7) 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. With Auto Enable off, the $\overline{RTS}$ pin strictly follows the state of the RTS bit. Both pins can be used as general-purpose outputs.

TxDA, TxDB: Transmit Data (outputs, active High).
These output signals transmit serial data at standard TTL levels.

$\overline{TRxCA}$, $\overline{TRxCB}$: Transmit/Receive Clocks (inputs or outputs, active Low).
These pins can be programmed in several different modes of operation. $\overline{TRxC}$ 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.

$\overline{WR}$: Write (input, active Low).
When the ASCC is selected, this signal indicates a write operation. The coincidence of $\overline{RD}$ and $\overline{WR}$ is interpreted as a reset.

$\overline{W}$/$\overline{REQA}$, $\overline{W}$/$\overline{REQB}$: Wait/Request (outputs, open-drain when programmed for a Wait function, driven High or Low when progammed 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 ASCC data rate. The reset state is Wait.

ABSOLUTE MAXIMUM RATINGS above which useful life may be impared

Storage Temperature	−65 to +150°C
Voltage at any Pin Relative to V_{SS}	−0.5V to +7.0V
Power Dissipation	1.8W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

(over which the DC, switching and functional specifications apply)

	4MHz	6MHz
Commercial Operating Range T_A = 0 to 70°C V_{CC} = 5V ±5%	Z8031DC Z8031PC Z8531DC Z8531PC	Z8031ADC Z8031APC Z8531ADC Z8531APC
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ±5%	Z8031DI Z8531DI	
Military Operating Range T_A = −55 to +125°C V_{CC} = 5V ±10%	Z8031DMB Z8531DMB	

Notes: T_A denotes ambient temperature.
Add suffix B to indicate burn-in requirement.

ELECTRICAL CHARACTERISTICS

Parameter	Description	Test Conditions	Min	Typ	Max	Units
V_{IH}	Input HIGH Voltage		2.0		V_{CC} + 0.3	V
V_{IL}	Input LOW Voltage		−0.3		0.8	V
V_{OH}	Output HIGH Voltage	I_{OH} = −250μA	2.4			V
V_{OL}	Output LOW Voltage	I_{OL} = +2.0mA			0.4	V
I_{IL}	Input Leakage	0.4 ≤ V_{IN} ≤ +2.4V			±10.0	μA
I_{OL}	Output Leakage	0.4 ≤ V_{OUT} ≤ +2.4V			±10.0	μA
I_{CC}	V_{CC} Supply Current				250	mA
C_{IN}	Input Capacitance	Unmeasured pins returned to ground. f = 1MHz over specified temperature range.			10	pF
C_{OUT}	Output Capacitance				15	pF
$C_{I/O}$	Bidirectional Capacitance				20	pF

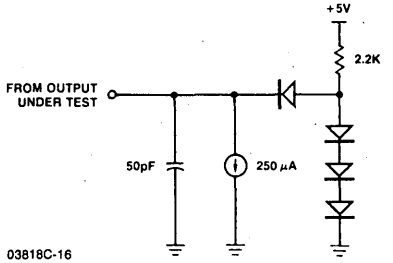
V_{CC} = 5V ± 5% unless otherwise specified, over specified temperature range.

Standard Test Conditions

The characteristics 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:

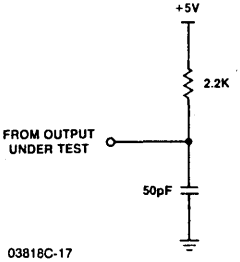
+4.75V ≤ V_{CC} ≤ +5.25V
GND = 0V
0°C ≤ T_A ≤ +70°C

Standard Test Load



03818C-16

Open Drain Test Load



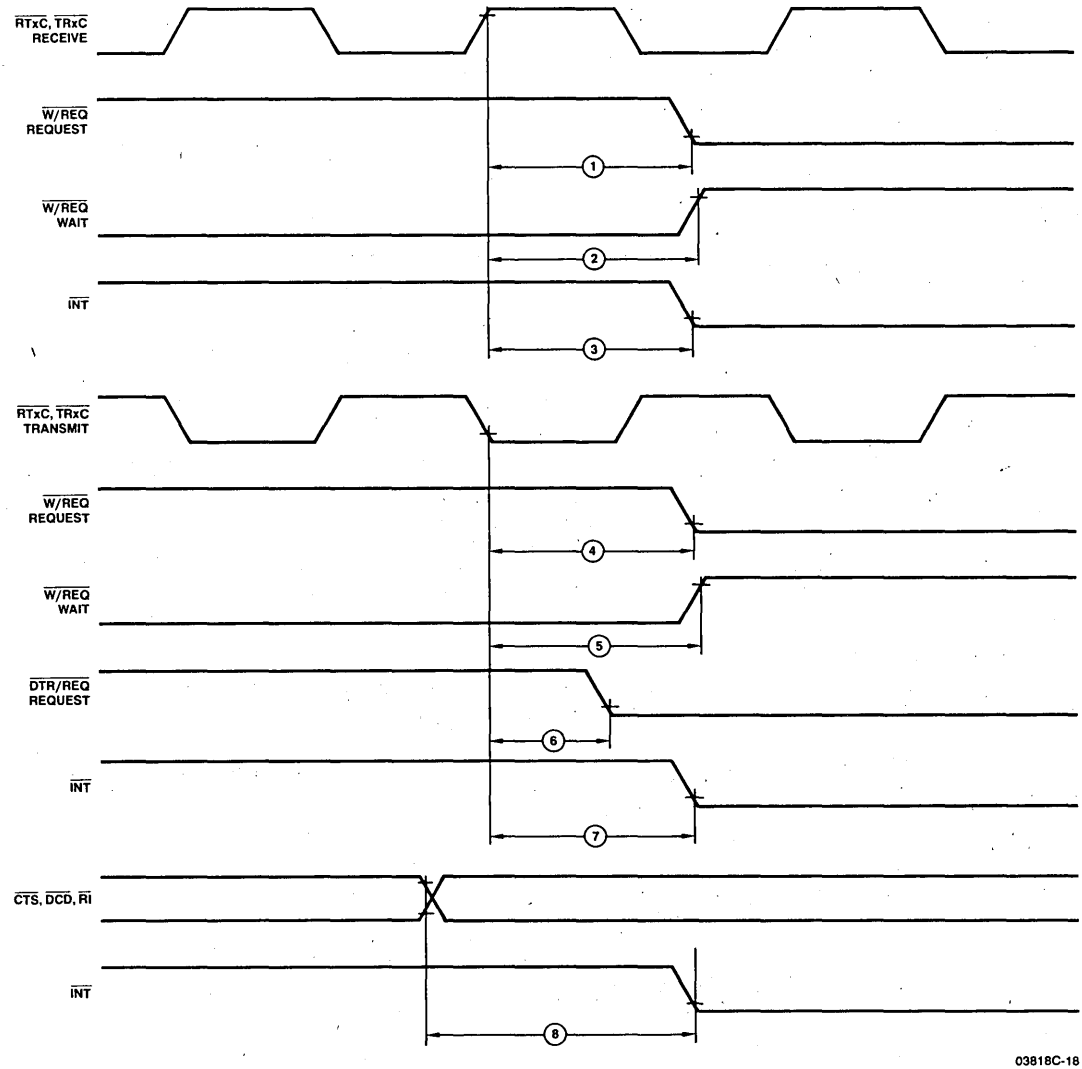
03818C-17

SYSTEM TIMING

Number	Parameters	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdRXC(REQ)	$\overline{RxC}\uparrow$ to $\overline{W/REQ}$ Valid Delay (Note 2)	8	12	8	12	TcPC
2	TdRXC(W)	$\overline{RxC}\uparrow$ to Wait Inactive Delay (Notes 1, 2)	8	12	8	12	TcPC
3	TdRXC(INT)	$\overline{RxC}\uparrow$ to $\overline{INT}$ Valid Delay (Notes 1, 2)	10	16	10	16	TcPC
4	TdTXC(REQ)	$\overline{TxC}\downarrow$ to $\overline{W/REQ}$ Valid Delay (Note 3)	5	8	5	8	TcPC
5	TdTXC(W)	$\overline{TxC}\downarrow$ to Wait Inactive Delay (Notes 1, 3)	5	8	5	8	TcPC
6	TdTXC(DRQ)	$\overline{TxC}\downarrow$ $\overline{DTR/REQ}$ Valid Delay (Note 3)	4	7	4	7	TcPC
7	TdTXC(INT)	$\overline{TxC}\downarrow$ to $\overline{INT}$ Valid Delay (Notes 1, 3)	6	10	6	10	TcPC
8	TdEXT(INT)	$\overline{DCD}$ or $\overline{CTS}$ Transition to $\overline{INT}$ Valid Delay (Note 1)	2	6	2	6	TcPC

Notes: 1. Open-drain output, measured with open-drain test load.
2. $\overline{RxC}$ is $\overline{RTxC}$ or $\overline{TRxC}$, whichever is supplying the receive clock.
3. $\overline{TxC}$ is $\overline{TRxC}$ or $\overline{RTxC}$, whichever is supplying the transmit clock.
*Timings are preliminary and subject to change.

Figure 14. System Timing

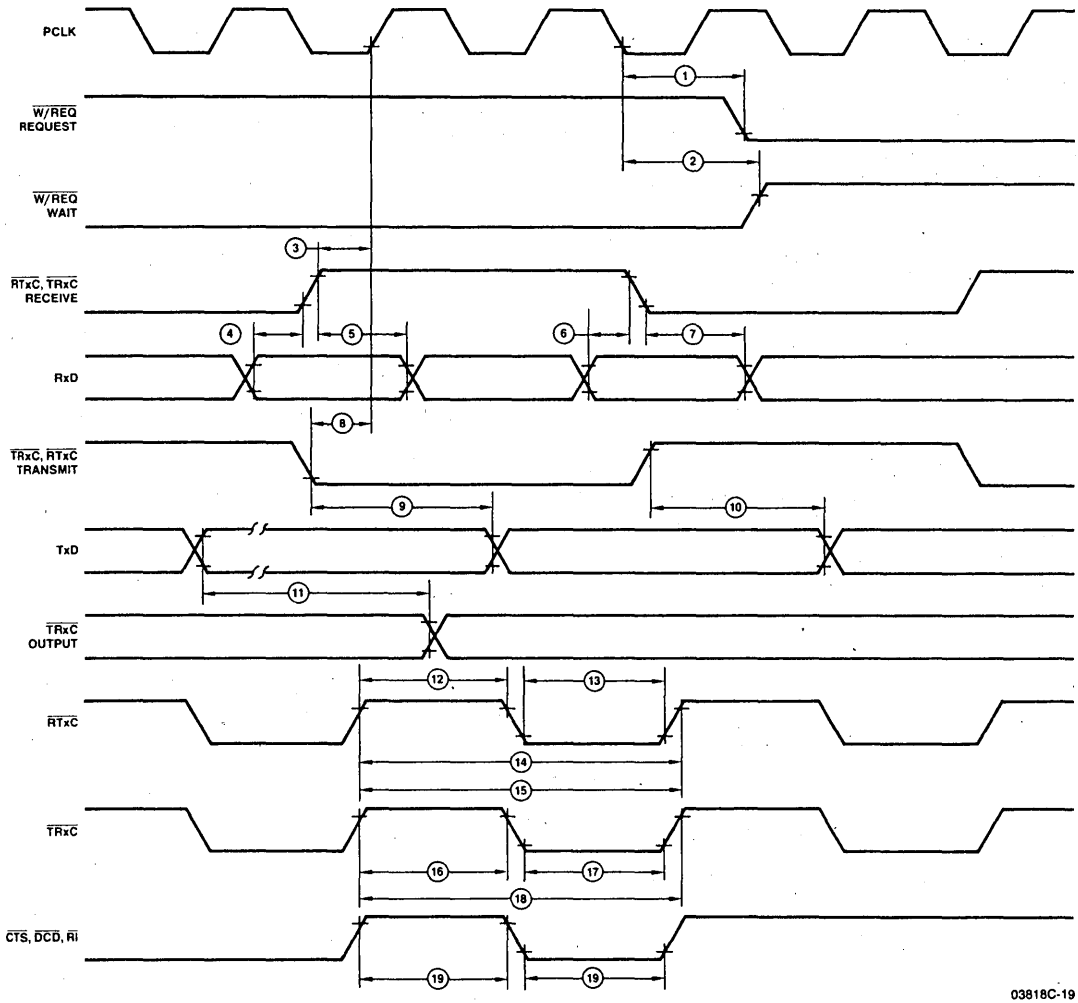


GENERAL TIMING (See Figure 15)

Number	Parameters	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdPC(REQ)	PCLK↓ to W/$\overline{REQ}$ Valid Delay		250		250	ns
2	TdPC(W)	PCLK↓ to Wait Inactive Delay		350		350	ns
3	TsRXC(PC)	$\overline{RxC}$↑ to PCLK↑ Setup Time (Notes 1, 4)	50		50		ns
4	TsRXD(RXCr)	RxD to $\overline{RxC}$↑ Setup Time (X1 Mode) (Note 1)	0		0		ns
5	ThRXD(RXCr)	RxD to $\overline{RxC}$↑ Hold Time (X1 Mode) (Note 1)	150		150		ns
6	TsRXD(RXCf)	RxD to $\overline{RxC}$↓ Setup Time (X1 Mode) (Notes 1, 5)	0		0		ns
7	ThRXD(RXCf)	RxD to $\overline{RxC}$↓ Hold Time (X1 Mode) (Notes 1, 5)	150		150		ns
8	TsTXC(PC)	$\overline{TxC}$↓ to PCLK↑ Setup Time (Notes 2, 4)	0		0		ns
9	TdTXCf(TXD)	$\overline{TxC}$↓ to TxD Delay (X1 Mode) (Note 2)		300		300	ns
10	TdTXCr(TXD)	$\overline{TxC}$↑ to TxD Delay (X1 Mode) (Notes 2, 5)		300		300	ns
11	TdTXD(TRX)	TxD to $\overline{TRxC}$ Delay (Send Clock Echo)					ns
12	TwRTXh	$\overline{RTxC}$ High Width	180		180		ns
13	TwRTXl	$\overline{RTxC}$ Low Width	180		180		ns
14	TcRTX	$\overline{RTxC}$ Cycle Time	400		400		ns
15	TcRTXX	Crystal Oscillator Period (Note 3)	250	1000	250	1000	ns
16	TwTRXh	$\overline{TRxC}$ High Width	180		180		ns
17	TwTRXl	$\overline{TRxC}$ Low Width	180		180		ns
18	TcTRX	$\overline{TRxC}$ Cycle Time	400		400		ns
19	TwEXT	$\overline{DCD}$ or $\overline{CTS}$ Pulse Width	200		200		ns

Notes:
1. RxC is $\overline{RTxC}$ or $\overline{TRxC}$, whichever is supplying the receive clock.
2. TxC is $\overline{TRxC}$ or $\overline{RTxC}$, whichever is supplying the transmit clock.
3. Both $\overline{RTxC}$ and $\overline{RI}$ have 30pF capacitors to the 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 $\overline{RxC}$ and $\overline{PCLK}$ or $\overline{TxC}$ and $\overline{PCLK}$ is required.
5. Parameter applies only to FM encoding/decoding.

Figure 15. General Timing

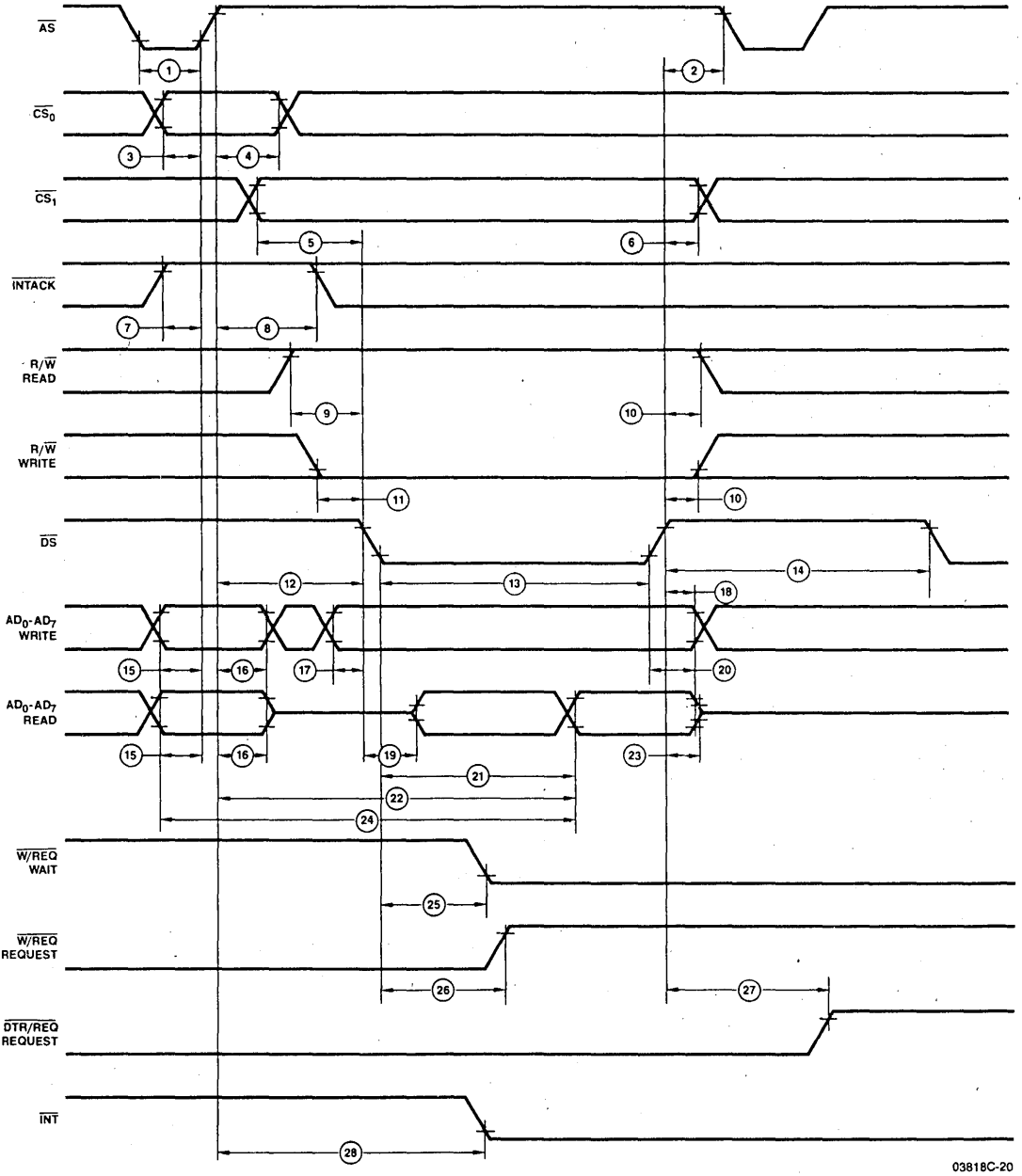


Z8031 READ AND WRITE TIMING (see Figure 16)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TwAS	$\overline{AS}$ LOW Width	70		50		ns
2	TdDS(AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	50		25		ns
3	TsCS0(AS)	$\overline{CS_0}$ to $\overline{AS}$ ↑ Setup Time (Note 1)	0		0		ns
4	ThCS0(AS)	$\overline{CS_0}$ to $\overline{AS}$ ↑ Hold Time (Note 1)	60		40		ns
5	TsCS1(DS)	CS_1 to $\overline{DS}$ ↓ Setup Time (Note 1)	100		80		ns
6	ThCS1(DS)	CS_1 to $\overline{DS}$ ↑ Hold Time (Note 1)	55		40		ns
7	TsIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Setup Time	0		0		ns
8	ThIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Hold Time	250		250		ns
9	TsRWR(DS)	R/$\overline{W}$ (Read) to $\overline{DS}$ ↓ Setup Time	100		80		ns
10	ThRW(DS)	R/$\overline{W}$ to $\overline{DS}$ ↑ Hold Time	55		40		ns
11	TsRWW(DS)	R/$\overline{W}$ (Write) to $\overline{DS}$ ↓ Setup Time	0		0		ns
12	TdAS(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay	60		40		ns
13	TwDSI	$\overline{DS}$ LOW Width	390		250		ns
14	TrC	Valid Access Recovery Time (Note 2)	6TcPC +200		6TcPC +130		ns
15	TsA(AS)	Address to $\overline{AS}$ ↑ Setup Time (Note 1)	30		10		ns
16	ThA(AS)	Address to $\overline{AS}$ ↑ Hold Time (Note 1)	50		30		ns
17	TsDW(DS)	Write Data to $\overline{DS}$ ↓ Setup Time	30		20		ns
18	ThDW(DS)	Write Data to $\overline{DS}$ ↑ Hold Time	30		20		ns
19	TdDS(DA)	$\overline{DS}$ ↓ to Data Active Delay	0		0		ns
20	TdDSr(DR)	$\overline{DS}$ ↑ to Read Data Not Valid Delay	0		0		ns
21	TdDSf(DR)	$\overline{DS}$ ↓ to Read Data Valid Delay		250		180	ns
22	TdAS(DR)	$\overline{AS}$ ↑ to Read Data Valid Delay		520		335	ns

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Parameter applies only between transactions involving the 8030.

Figure 16. Z8031 Read and Write Timing



Z8031 INTERRUPT ACKNOWLEDGE TIMING, RESET TIMING, CYCLE TIMING (see Figures 17, 18, 19)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
23	TdDS(DRz)	$\overline{DS}$ ↑ to Read Data Float Delay (Note 3)		70		45	ns
24	TdA(DR)	Address Required Valid to Read Data Valid Delay		570		420	ns
25	TdDS(W)	$\overline{DS}$ ↓ to Wait Valid Delay (Note 4)		240		200	ns
26	TdDSf(REQ)	$\overline{DS}$ ↓ to $\overline{W/REQ}$ Not Valid Delay		240		200	ns
27	TdDSr(REQ)	$\overline{DS}$ ↓ to $\overline{DTR/REQ}$ Not Valid Delay		5TcPC +300		5TcPC +250	ns
28	TdAS(INT)	$\overline{AS}$ ↑ to $\overline{INT}$ Valid Delay (Note 4)		500		500	ns
29	TdAS(DSA)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ (Acknowledge) Delay (Note 5)					ns
30	TwDSA	$\overline{DS}$ (Acknowledge) Low Width	390		250		ns
31	TdDSA(DR)	$\overline{DS}$ ↓ (Acknowledge) to Read Data Valid Delay		250		180	ns
32	TsIEI(DSA)	IEI to $\overline{DS}$ ↓ (Acknowledge) Setup Time	120		100		ns
33	ThIEI(DSA)	IEI to $\overline{DS}$ ↑ (Acknowledge) Hold Time	0		0		ns
34	TdIEI(IEO)	IEI to IEO Delay		120		100	ns
35	TdAS(IEO)	$\overline{AS}$ ↑ to IEO Delay (Note 6)		250		250	ns
36	TdDSA(INT)	$\overline{DS}$ ↓ (Acknowledge) to $\overline{INT}$ Inactive Delay (Note 4)		500		500	ns
37	TdDS(ASQ)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay for No Reset	30		15		ns
38	TdASQ(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay for No Reset	30		30		ns
39	TwRES	$\overline{AS}$ and $\overline{DS}$ Coincident Low for Reset (Note 7)	250		250		ns
40	TwPCl	PCLK Low Width	105	2000	70	1000	ns
41	TwPCh	PCLK High Width	105	2000	70	1000	ns
42	TcPC	PCLK Cycle Time	250	4000	165	2000	ns
43	TrPC	PCLK Rise Time		20		15	ns
44	TfPC	PCLK Fall Time		20		10	ns

Notes:
3. Float delay is defined as the time required for a ±0.5V change in the output with a maximum dc load and minimum ac load.
4. Open-drain output, measured with open-drain test load.
5. Parameter is system dependent. For any 8031 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 8031, and TdIEIf(IEO) for each device separating them in the daisy chain.
6. Parameter applies only to a 8031 pulling $\overline{INT}$ 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 8031.

*Timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic.

Figure 17. Z8031 Interrupt Acknowledge Timing

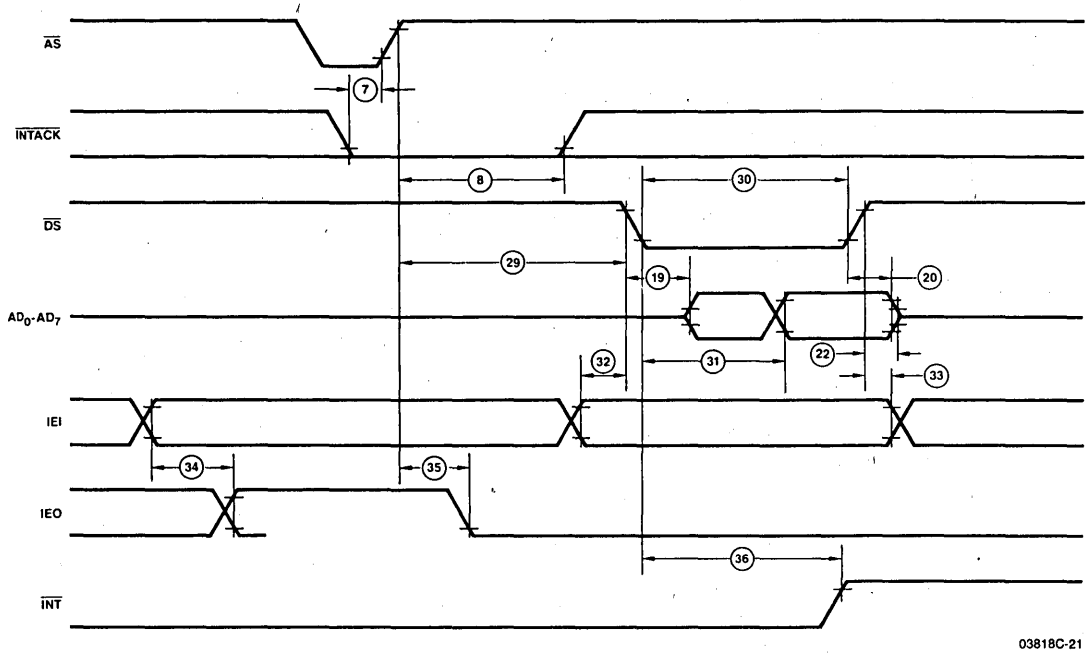


Figure 18. Z8031 Reset Timing

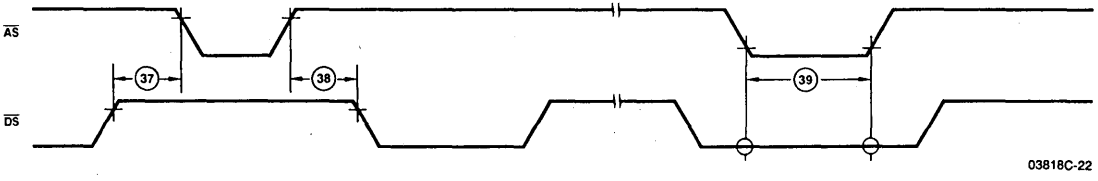
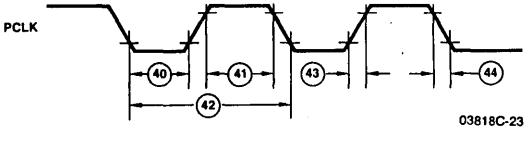


Figure 19. Z8031 Cycle Timing

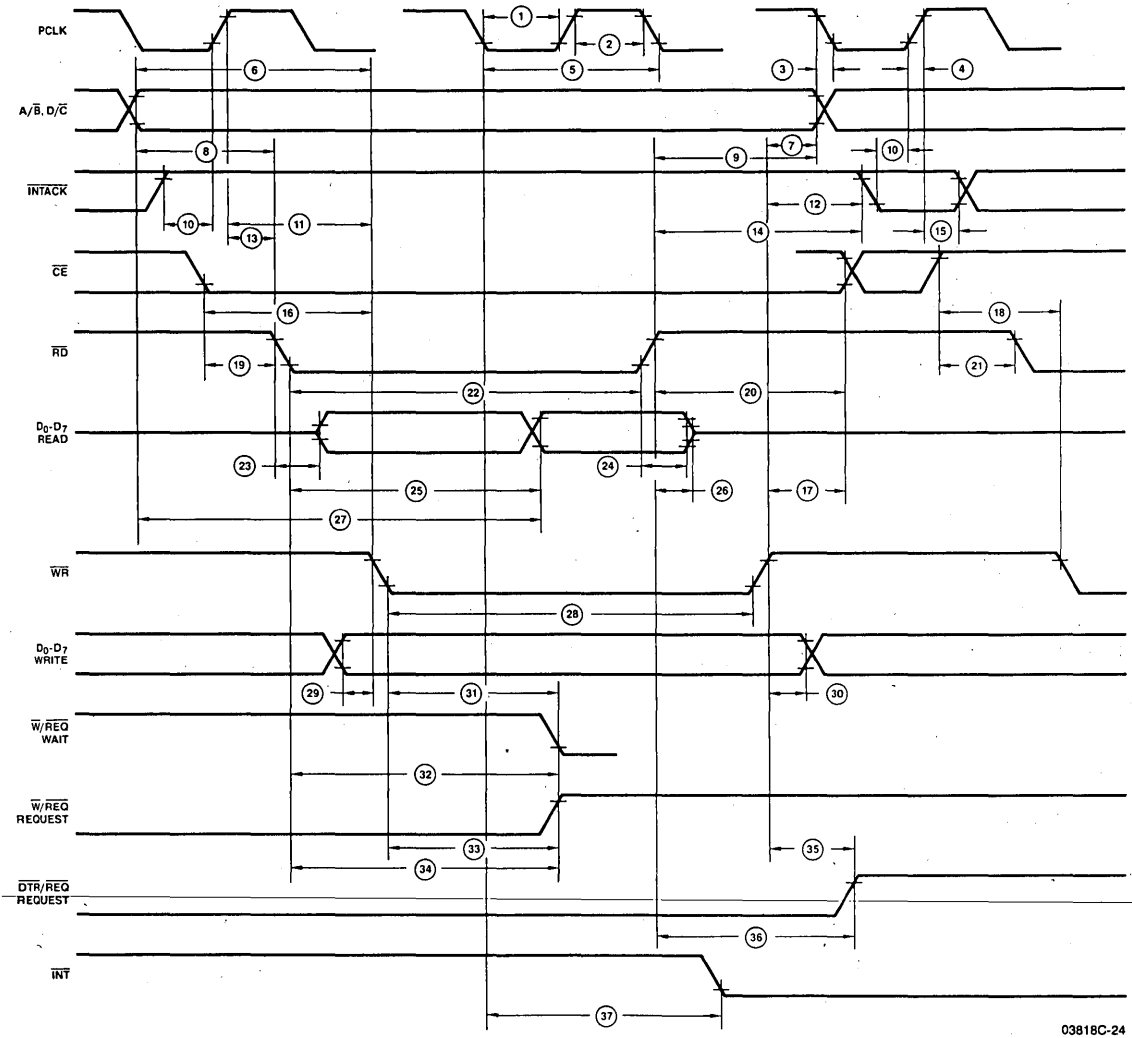


Z8531 READ AND WRITE TIMING (see Figure 20)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TwPCl	PCLK Low Width	105	2000	70	1000	ns
2	TwPCh	PCLK High Width	105	2000	70	1000	ns
3	TfPC	PCLK Fall Time		20		10	ns
4	TrPC	PCLK Rise Time		20		15	ns
5	TcPC	PCLK Cycle Time	250	4000	165	2000	ns
6	TsA(WR)	Address to $\overline{WR}$ ↓ Setup Time	80		80		ns
7	ThA(WR)	Address to $\overline{WR}$ ↑ Hold Time	0		0		ns
8	TsA(RD)	Address to $\overline{RD}$ ↓ Setup Time	80		80		ns
9	ThA(RD)	Address to $\overline{RD}$ ↑ Hold Time	0		0		ns
10	TslA(PC)	$\overline{INTACK}$ to PCLK ↑ Setup Time	0		0		ns
11	TslAi(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↓ Setup Time (Note 1)	200		200		ns
12	ThlA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
13	TslAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ Setup Time (Note 1)	200		200		ns
14	ThlA(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↑ Hold Time	0		0		ns
15	ThlA(PC)	$\overline{INTACK}$ to PCLK ↑ Hold Time	100		100		ns
16	TsCEl(WR)	$\overline{CE}$ Low to $\overline{WR}$ ↓ Setup Time	0		0		ns
17	ThCE(WR)	$\overline{CE}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
18	TsCEh(WR)	$\overline{CE}$ High to $\overline{WR}$ ↓ Setup Time	100		70		ns
19	TsCEl(RD)	$\overline{CE}$ Low to $\overline{RD}$ ↓ Setup Time (Note 1)	0		0		ns
20	ThCE(RD)	$\overline{CE}$ to $\overline{RD}$ ↑ Hold Time (Note 1)	0		0		ns
21	TsCEh(RD)	$\overline{CE}$ High to $\overline{RD}$ ↓ Setup Time (Note 1)	100		70		ns
22	TwRDl	$\overline{RD}$ Low Width (Note 1)	390		250		ns
23	TdRD(DRA)	$\overline{RD}$ ↓ to Read Data Active Delay	0		0		ns
24	TdRDr(DR)	$\overline{RD}$ ↑ to Read Data Not Valid Delay	0		0		ns
25	TdRDf(DR)	$\overline{RD}$ ↓ to Read Data Valid Delay		250		180	ns
26	TdRD(DRz)	$\overline{RD}$ ↑ to Read Data Float Delay (Note 2)		70		45	ns

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float delay is defined as the time required for a ±0.5V change in the output with a maximum dc load and minimum ac load.

Figure 20. Z8531 Read and Write Timing



Z8531 INTERRUPT ACKNOWLEDGE TIMING, RESET TIMING, CYCLE TIMING (see Figures 21, 22, 23)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
27	TdA(DR)	Address Required Valid to Read Data Valid Delay		590		420	ns
28	TwWRI	$\overline{WR}$ Low Width	390		250		ns
29	TsDW(WR)	Write Data to $\overline{WR}$ ↓ Setup Time	0		0		ns
30	ThDW(WR)	Write Data to $\overline{WR}$ ↑ Hold Time	0		0		ns
31	TdWR(W)	$\overline{WR}$ ↓ to Wait Valid Delay (Note 4)		240		200	ns
32	TdRD(W)	$\overline{RD}$ ↓ to Wait Valid Delay (Note 4)		240		200	ns
33	TdWRf(REQ)	$\overline{WR}$ ↓ to $\overline{W}/\overline{REQ}$ Not Valid Delay		240		200	ns
34	TdRDf(REQ)	$\overline{RD}$ ↓ to $\overline{W}/\overline{REQ}$ Not Valid Delay		240		200	ns
35	TdWRr(REQ)	$\overline{WR}$ ↑ to $\overline{DTR}/\overline{REQ}$ Not Valid Delay		5TcPC +300		5TcPC +250	ns
36	TdRDr(REQ)	$\overline{RD}$ ↑ to $\overline{DTR}/\overline{REQ}$ Not Valid Delay		5TcPC +300		5TcPC +250	ns
37	TdPC(INT)	PCLK ↓ to $\overline{INT}$ Valid Delay (Note 4)		500		500	ns
38	TdIAi(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ (Acknowledge) Delay (Note 5)					ns
39	TwRDA	$\overline{RD}$ (Acknowledge) Width	285		250		ns
40	TdRDA(DR)	$\overline{RD}$ ↓ (Acknowledge) to Read Data Valid Delay		190		180	ns
41	TsIEI(RDA)	IEI to $\overline{RD}$ ↓ (Acknowledge) Setup Time	120		100		ns
42	ThIEI(RDA)	IEI to $\overline{RD}$ ↑ (Acknowledge) Hold Time	0		0		ns
43	TdIEI(IEO)	IEI to IEO Delay Time		120		100	ns
44	TdPC(IEO)	PCLK ↑ to IEO Delay		250		250	ns
45	TdRDA(INT)	$\overline{RD}$ ↓ to $\overline{INT}$ Inactive Delay (Note 4)		500		500	ns
46	TdRD(WRQ)	$\overline{RD}$ ↑ to $\overline{WR}$ ↓ Delay for No Reset	30		15		ns
47	TdWRQ(RD)	$\overline{WR}$ ↑ to $\overline{RD}$ ↓ Delay for No Reset	30		30		ns
48	TwRES	$\overline{WR}$ and $\overline{RD}$ Coincident Low for Reset	250		250		ns
49	Trc	Valid Access Recovery Time (Note 3)	6TcPC +200		6TcPC +130		ns

Notes: 3. Parameter applies only between transactions involving the ASCC.
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 TdIEIf(IEO) for each device separating them in the daisy chain.

Figure 21. Z8531 Interrupt Acknowledge Timing

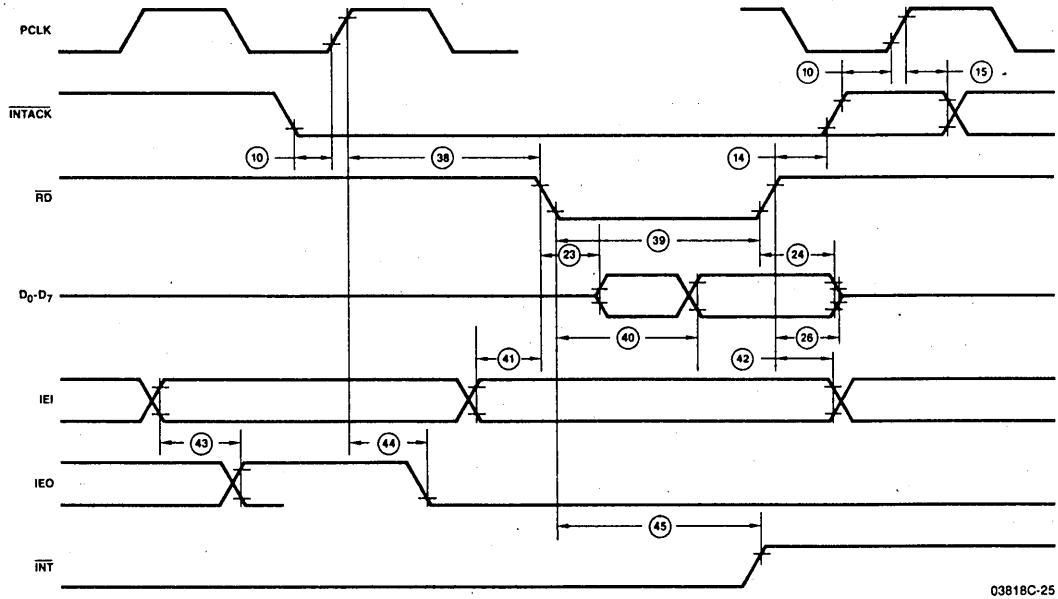


Figure 22. Z8531 Reset Timing

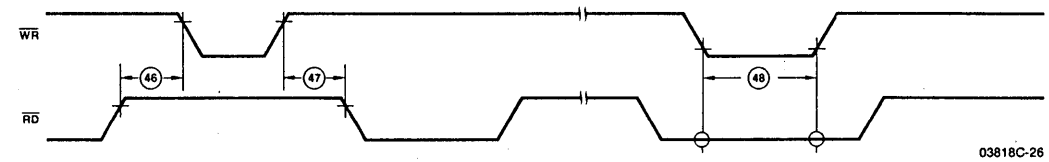
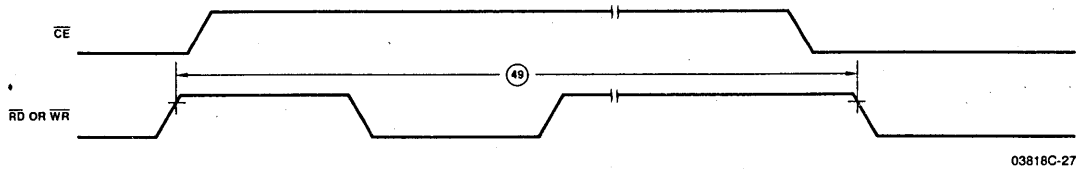


Figure 23. Z8531 Cycle Timing



AmZ8036/8536

Counter/Timer and Parallel I/O Unit

DISTINCTIVE CHARACTERISTICS

- Two independent 8-bit, double-buffered, bidirectional I/O ports plus a 4-bit special purpose I/O port. I/O ports feature programmable polarity, programmable direction (Bit mode), "pulse catchers" and programmable open-drain outputs.
- Four handshake modes, including 3-wire (like the IEEE-488).
- REQUEST/$\overline{\text{WAIT}}$ signal for high-speed data transfer.
- Flexible pattern-recognition logic, programmable as a 16-vector interrupt controller.
- Three independent 16-bit counter/timers with up to four external access lines per counter/timer (count input, output, gate, and trigger), and three output duty cycles (pulsed, one-shot, and square-wave), programmable as retriggerable or nonretriggerable.
- Easy to use since all registers are read/write and directly addressable.

GENERAL DESCRIPTION

The AmZ8036* CIO Counter/Timers and Parallel I/O elements are general-purpose peripheral circuits, satisfying most counter/timer and parallel I/O needs encountered in system designs. These versatile devices contain three I/O ports and three counter/timers. Many programmable options tailor their configuration to specific applications.

The use of these devices is simplified by making all internal registers (command, status, and data) readable and (except for status bits) writable. Each register is given its own unique address so that it can be accessed directly on the AmZ8036. The AmZ8036 is directly Z-Bus compatible.

See "AmZ8036/AmZ8536 Counter Timer, Parallel I/O Technical Manual" – 1982 edition (A1Z-2090) for detailed technical information and software examples.

CIO BLOCK DIAGRAM

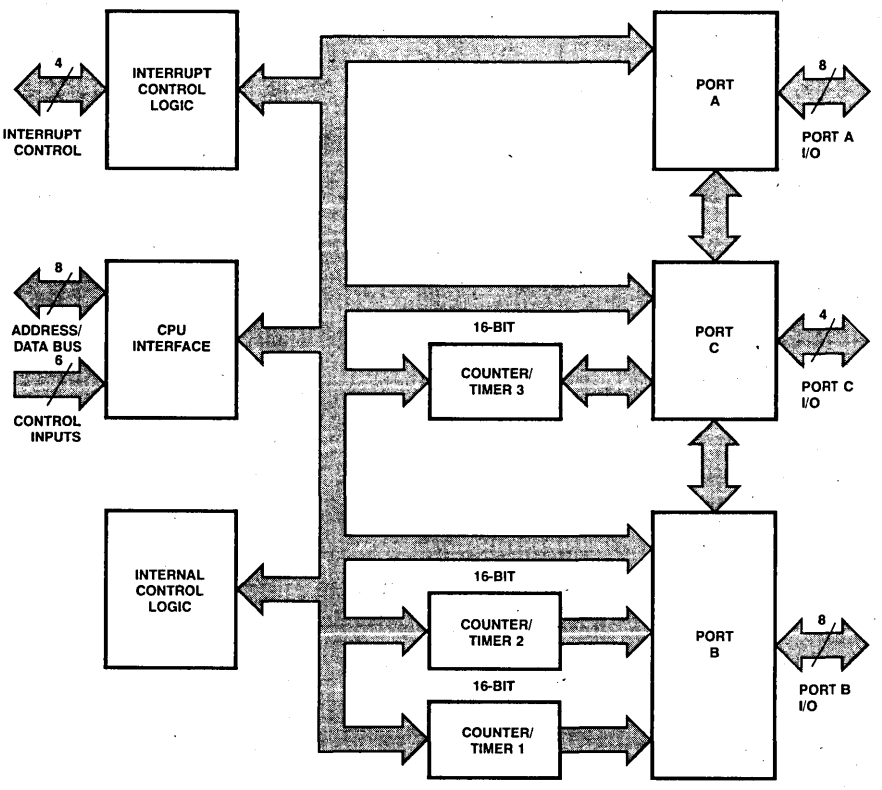
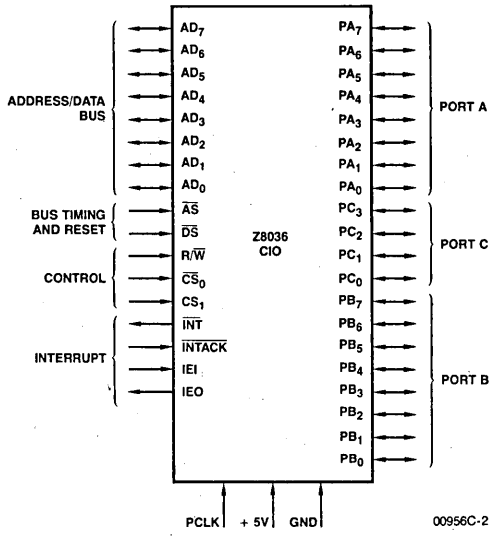


Figure 1.

00956C-1

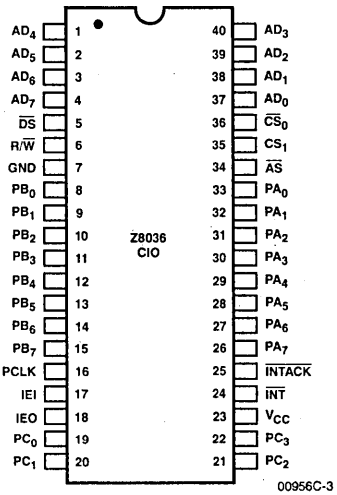
*Z8000 is a trademark of Zilog, Inc.

LOGIC SYMBOL



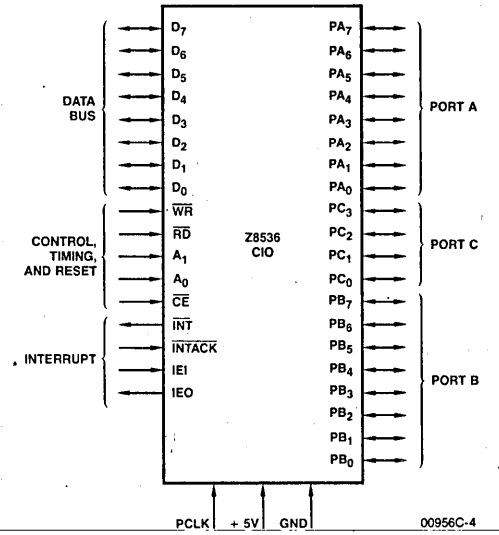
CONNECTION DIAGRAM – Top View

D-40-1, P-40-1



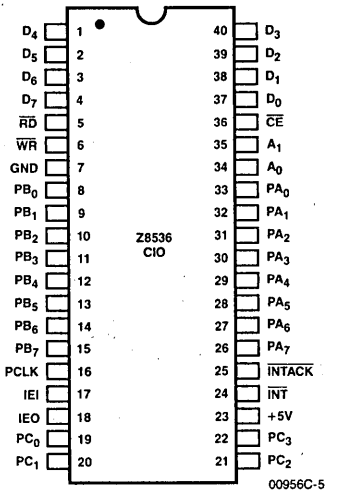
Note: Pin 1 is marked for orientation.

LOGIC SYMBOL



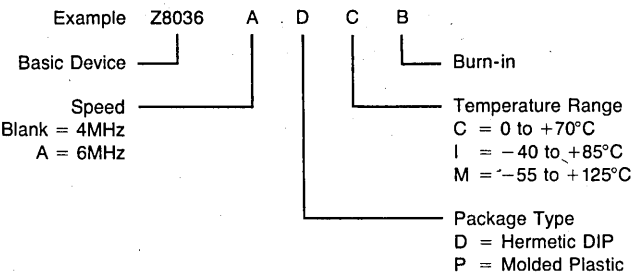
CONNECTION DIAGRAM – Top View

D-40-1, P-40-1



Note: Pin 1 is marked for orientation.

ORDERING INFORMATION



INTERFACE SIGNAL DESCRIPTION

V_{CC}: +5V Power Supply

GND: Ground

IEI: Interrupt Enable In (Input)

IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt-driven device on the system bus. A HIGH IEI indicates to the CIO that no other higher priority device has an interrupt under service or is requesting an interrupt.

IEO: Interrupt Enable Out (Output)

IEO is normally connected to the next lower priority device's IEI input and inhibits interrupts from lower priority devices. IEO is HIGH only if IEI is HIGH and the CIO has not requested an interrupt.

$\overline{INT}$: Interrupt Request (Output, Open Drain)

This signal is active LOW when the CIO is requesting an interrupt and stays active until the end of the interrupt acknowledge sequence.

$\overline{INTACK}$: Interrupt Acknowledge (Input)

This signal, when active LOW, indicates to the CIO that an interrupt acknowledge cycle is in progress. $\overline{INTACK}$ is sampled while AS is LOW.

PA_0-PA_7: Port A I/O lines (Bidirectional, 3-state or Open Drain)

These eight I/O lines transfer information between Port A and external devices.

PB_0-PB_7: Port B I/O lines (Bidirectional, 3-state, or Open Drain)

These eight I/O lines transfer information between Port B and external devices. The lines can also be programmed to provide external access to Counters/Timers 1 and 2.

PC_0-PC_3: Port C I/O lines (Bidirectional, 3-state or Open Drain)

These four I/O lines provide Handshake, WAIT and REQUEST lines for Ports A and B, or provide external access to Counter/Timer 3 or access to Port C.

PCLK: Peripheral Clock (Input)

PCLK may be synchronous or asynchronous to the CPU's clock and may be of lower frequency than the CPU's clock. It is used with timers and Request/$\overline{Wait}$ logic. The input is TTL compatible.

Z8036 Only

AD_0-AD_7: Address/Data Bus (Bidirectional, 3-state)

Multiplexed address/data lines for transfers between the CPU and CIO.

$\overline{AS}$*: Address Strobe (Input)

Register addresses on AD_0-AD_7 lines, $\overline{INTACK}$, and $\overline{CS_0}$ are sampled while AS is LOW, and latched while $\overline{AS}$ is HIGH.

$\overline{CS_0}$ and $\overline{CS_1}$: Chip Select 0 and Chip Select 1 (Inputs)

Chip Select 0 and Chip Select 1 must be LOW and HIGH, respectively, in order to select the device. $\overline{CS_0}$ is latched by $\overline{AS}$.

$\overline{DS}$*: Data Strobe (Input)

An active LOW $\overline{DS}$ provides the timing for transfer of data to or from the CIO. The R/$\overline{W}$ input indicates the direction of data transfer.

R/$\overline{W}$: Read/Write (Input)

R/$\overline{W}$ is active HIGH when the CPU is reading from the CIO, and active LOW when the CPU is writing to the CIO.

*When $\overline{AS}$ and $\overline{DS}$ are detected LOW at the same time (normally an illegal condition), the CIO is reset.

Z8536 Only

A_0-A_1: Address Lines (Input)

These two lines are used to select the register involved in the CPU transaction: Port A's Data register, Port B's Data register, Port C's Data register, or a control register.

$\overline{CE}$: Chip Enable (Input, Active LOW)

A LOW level on this input enables the CIO to be read from or written to.

D_0-D_7: Data Bus (Bidirectional, 3-state)

These eight data lines are used for transfers between the CPU and the CIO.

$\overline{RD}$: Read (Input, Active LOW)**

This signal indicates that a CPU is reading from the CIO. During an interrupt acknowledge cycle, this signal gates the interrupt vector onto the data bus if the CIO is the highest priority device requesting an interrupt.

$\overline{WR}$: Write (Input, Active LOW)**

This signal indicates a CPU write to the CIO.

**When $\overline{RD}$ and $\overline{WR}$ are detected LOW at the same time (normally an illegal condition), the CIO is reset.

MAXIMUM RATINGS (Above which useful life may be impaired)

Storage Temperature	65 to +150°C
Voltage at any Pin Relative V_{SS}	−0.5 to +7.0V
Power Dissipation	1.75W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

(over which the DC, switching and functional specification apply)

	4MHz	6MHz
Commercial Operating Range T_A = 0 to +70°C V_{CC} = 5V ± 5%	Z8036DC Z8036PC Z8536DC Z8536PC	Z8036ADC Z8036APC Z8536ADC Z8536APC
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ± 5%	Z8036DI Z8536DI	
Military Operating Range T_A = −55 to +125°C V_{CC} = 5V ± 10%	Z8036DMB Z8536DMB	

Notes: Add suffix B to indicate burn-in requirement.

DC ELECTRICAL CHARACTERISTICS over operating range unless otherwise specified (Note 1)

Parameter	Description	Test Conditions		Min	Typ	Max	Units
V_{IL}	Input LOW Voltage			−0.5		+0.8	Volts
V_{IH}	Input HIGH Voltage	Standard		2.0		V_{CC}	Volts
		Military		2.2			
V_{OL}	Output LOW Voltage	I_{OL} = 2.0mA				0.4	Volts
		I_{OL} = 3.2mA				0.5	
V_{OH}	Output HIGH Voltage	I_{OH} = −250μA		2.4			Volts
I_{OZL}	Output Leakage Current	V_{OUT} = 0.4V				10	μA
I_{OZH}	Output Leakage Current	V_{OUT} = V_{CC}				10	μA
I_I	Input Leakage Current					±10	μA
I_{CC}	Power Supply Current	V_{CC} = MAX	T_A = 0°C			200	mA
			T_A = −55°C			250	
C_{IN}	Input Capacitance	Unmeasured pins returned to ground f = 1MHz				10	pF
C_{OUT}	Output Capacitance					15	pF
$C_{I/O}$	Bidirectional Capacitance					20	pF

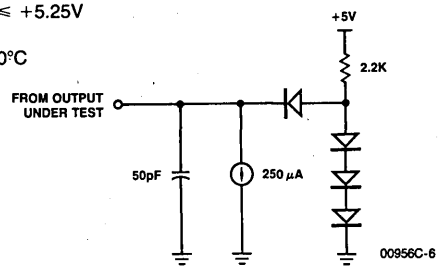
Note: See table for operating range. Typical conditions apply at T_A = 25°C, V_{CC} = 5.0V.

Standard Test Conditions

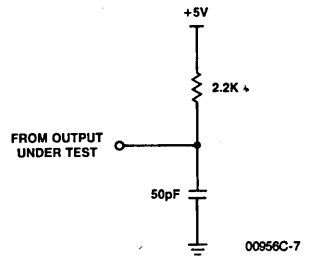
The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

+4.75V ≤ V_{CC} ≤ +5.25V
GND = 0V
0°C ≤ T_A ≤ +70°C

Standard Test Load



Open-Drain Test Load



Z8036/Z8536 HANDSHAKE TIMING (Figures 12, 13, 14)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TsDI(ACK)	Data Input to $\overline{ACKIN}$ ↓ Setup Time	0		0		ns
2	ThDI(ACK)	Data Input to $\overline{ACKIN}$ ↓ Hold Time – Strobed HS	500				ns
3	TdACKf(RFD)	$\overline{ACKIN}$ ↓ to RFD ↓ Delay	0		0		ns
4	TwACKl	$\overline{ACKIN}$ Low Width – Strobed HS	250				ns
5	TwACKh	$\overline{ACKIN}$ High Width – Strobed HS	250				ns
6	TdRFDr(ACK)	RFD ↑ to $\overline{ACKIN}$ ↓ Delay	0		0		ns
7	TsDO(DAV)	Data Out to $\overline{DAV}$ ↓ Setup Time (Note 1)	25		20		ns
8	TdDAVf(ACK)	$\overline{DAV}$ ↓ to $\overline{ACKIN}$ ↓ Delay	0		0		ns
9	ThDO(ACK)	Data Out to $\overline{ACKIN}$ ↓ Hold Time	1		1		$\overline{AS}$ cycle
10	TdACK(DAV)	$\overline{ACKIN}$ ↓ to $\overline{DAV}$ ↑ Delay	1		1		$\overline{AS}$ cycle
11	ThDI(RFD)	Data Input to RFD ↓ Hold Time – Interlocked HS	0		0		ns
12	TdRFDf(ACK)	RFD ↓ to $\overline{ACKIN}$ ↑ Delay – Interlocked HS	0		0		ns
13	TdACKr(RFD)	$\overline{ACKIN}$ ↑ ($\overline{DAV}$ ↑) to RFD ↑ Delay – Interlocked and 3-Wire HS	0		0		ns
14	TdDAVr(ACK)	$\overline{DAV}$ ↓ to $\overline{ACKIN}$ ↑ (RFD ↑) – Interlocked and 3-Wire HS	0		0		ns
15	TdACK(DAV)	$\overline{ACKIN}$ ↑ (RFD ↑) to $\overline{DAV}$ ↓ Delay – Interlocked and 3-Wire HS	0		0		ns
16	TdDAVlf(DAC)	$\overline{DAV}$ ↓ to DAC ↑ Delay – Input 3-Wire HS	0		0		ns
17	ThDI(DAC)	Data Input to DAC ↑ Hold Time – 3-Wire HS	0		0		ns
18	TdDACOr(DAV)	DAC ↑ $\overline{DAV}$ ↑ Delay – Input 3-Wire HS	0		0		ns
19	TdDAVlr(DAC)	$\overline{DAV}$ ↑ to DAC ↓ Delay – Input 3-Wire HS	0		0		ns
20	TdDAVOf(DAC)	$\overline{DAV}$ ↓ to DAC ↑ Delay – Output 3-Wire HS	0		0		ns
21	ThDO(DAC)	Data Output to DAC ↑ Hold Time – 3-Wire HS	1		1		$\overline{AS}$ cycle
22	TdDAClr(DAV)	DAC ↑ to $\overline{DAV}$ ↑ Delay – Output 3-Wire HS	1		1		$\overline{AS}$ cycle
23	TdDAVOr(DAC)	$\overline{DAV}$ ↑ to DAC ↓ Delay – Output 3-Wire HS	0		0		ns

Note: 1. This time can be extended through the use of the Deskew Timers.

*Timings are all preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 12. Strobed Handshake

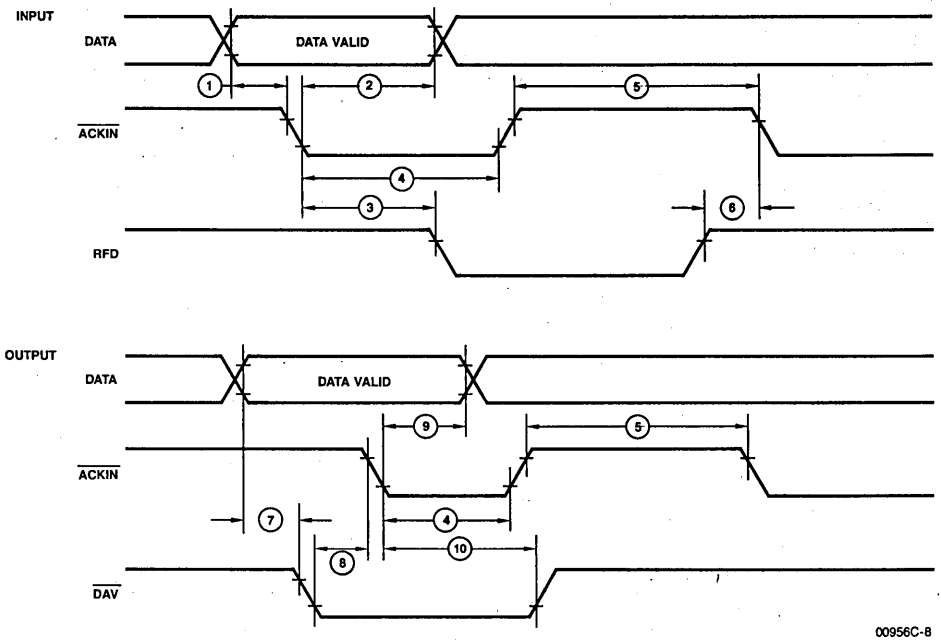


Figure 13. Interlocked Handshake

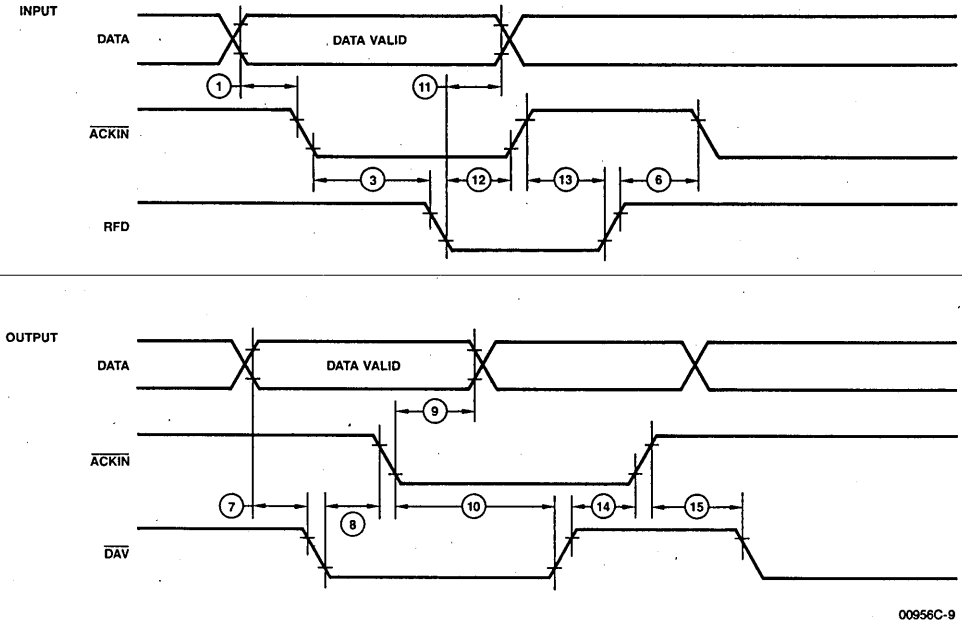
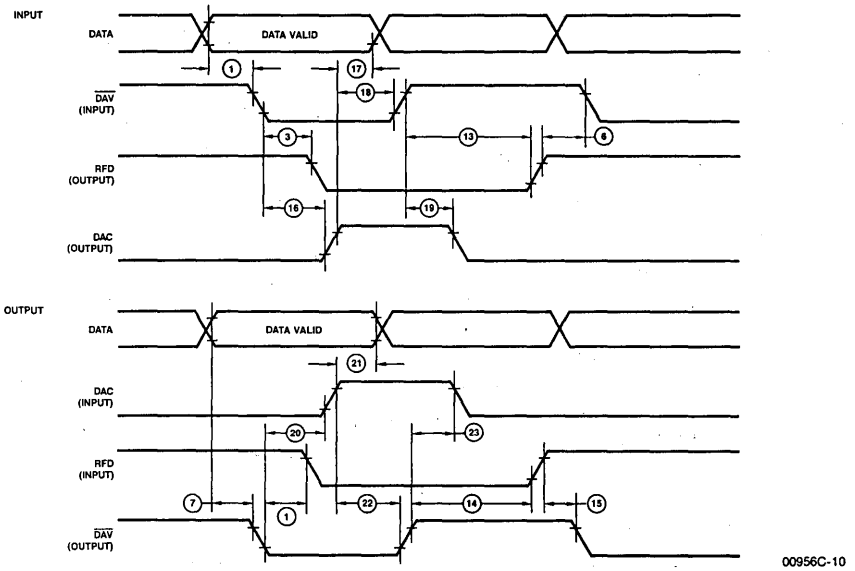


Figure 14. Three-Wire Handshake



Z8036 COUNTER/TIMER TIMING (Figure 15)

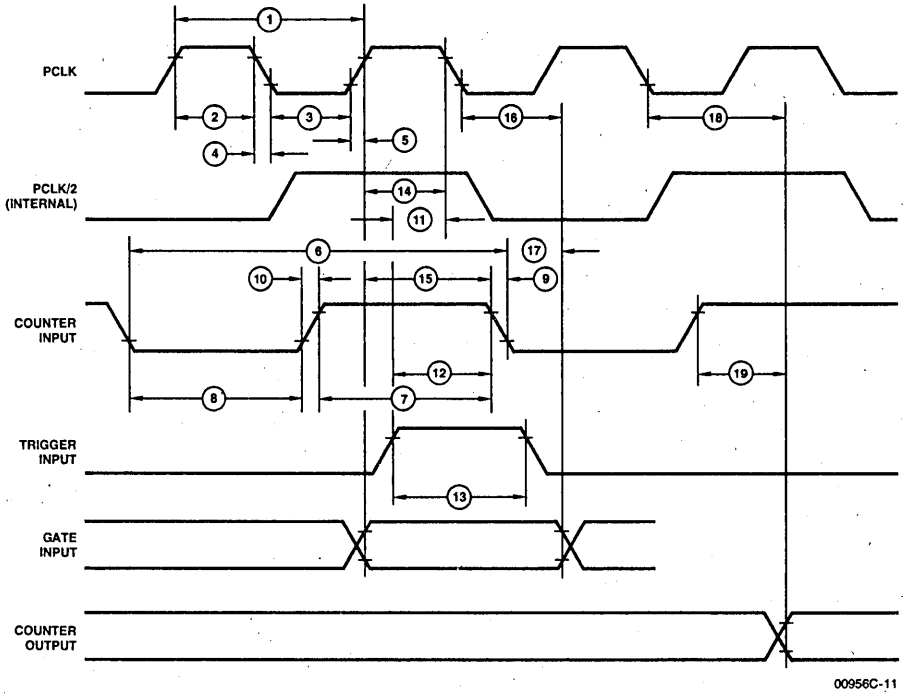
Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TcPC	PCLK Cycle Time (Note 1)	250	4000	165	4000	ns
2	TwPCh	PCLK High Width	105	2000	70	2000	ns
3	TwPCl	PCLK Low Width	105	2000	70	2000	ns
4	TfPC	PCLK Fall Time		20		10	ns
5	TrPC	PCLK Rise Time		20		15	ns
6	TcCI	Counter Layout Cycle Time	500		330		ns
7	TCIh	Counter Input High Width	230		150		ns
8	TwCIl	Counter Input Low Width	230		150		ns
9	TfCI	Counter Input Fall Time		20		15	ns
10	TrCI	Counter Input Rise Time		20		15	ns
11	TsTI(PC)	Trigger Input to PCLK Setup Time (Timer Mode) (Note 2)	150				ns
12	TsTI(CI)	Trigger Input to Counter Input Setup Time (Counter Mode) (Note 2)	150				ns
13	TwTI	Trigger Input Pulse Width (High or Low)	200				ns
14	TsGI(PC)	Gate Input to PCLK ↓ Setup Time (Timer Mode) (Note 2)	100				ns
15	TsGI(CI)	Gate Input to Counter Input ↓ Setup Time (Counter Mode) (Note 2)	100				ns
16	ThGI(PC)	Gate Input to PCLK ↓ Hold Time (Timer Mode) (Note 2)	100				ns
17	ThGI(CI)	Gate Input to Counter Input ↓ Hold Time (Counter Mode) (Note 2)	100				ns
18	TdPC(CO)	PCLK to Counter Output Delay (Timer Mode)		475			ns
19	TdCI(CO)	Counter Input to Counter Output Delay (Counter Mode)		475			ns

Notes: 1. PCLK is only used with the counter/timers (in Timer mode), the deskew timers, and the REQUEST/$\overline{\text{WAIT}}$ logic. If these functions are not used the PCLK input can be held low.

2. These parameters must be met to guarantee trigger or gate are valid for the next counter/timer cycle.

*Timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 15. Counter/Timer Timing

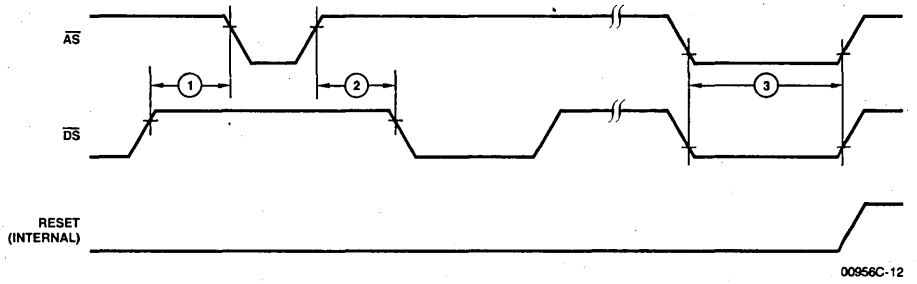


Z8036 RESET TIMING (Figure 16)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdDSQ(AS)	Delay from $\overline{DS}$ ↑ to $\overline{AS}$ ↓ for No Reset	40		15		ns
2	TdASQ(DS)	Delay from $\overline{AS}$ ↑ to $\overline{DS}$ ↓ for No Reset	50		30		ns
3	TwRES	Minimum Width of $\overline{AS}$ and $\overline{DS}$ both Low for Reset (Note 1)	250		170		ns

Note: 1. Internal circuitry allows for the reset provided by the Z8 ($\overline{DS}$ held Low while $\overline{AS}$ pulses) to be sufficient.
*Timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 16. Reset Timing

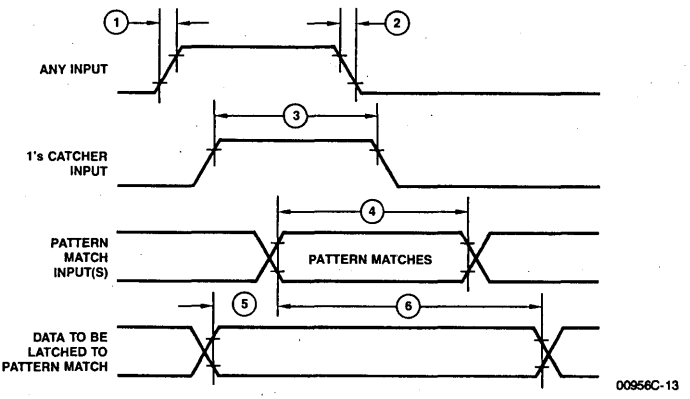


Z8036/Z8536 MISCELLANEOUS PORT TIMING (Figure 17)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	Trl	Any Input Rise Rate Time		100		100	ns
2	Tfl	Any Input Fall Time		100		100	ns
3	Tw1's	1's Catcher High Width (Note 1)	250		170		ns
4	TwPM	Pattern Match Input Valid (Bit Port)	750		500		ns
5	TsPMD	Data Latched on Pattern Match Setup Time (Bit Port)	0		0		ns
6	ThPMD	Data Latched on Pattern Match Hold Time (Bit Port)	1000		650		ns

Note: 1. If the input is programmed inverting, a Low-going pulse of the same width will be detected.
*Timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 17. Miscellaneous Port Timing



Z8036 CPU INTERFACE TIMING (Figure 18)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TwAs	$\overline{AS}$ Low Width	70	2000	50	2000	ns
2	TsA(AS)	Address to $\overline{AS}$ ↑ Setup Time (Note 1)	30		10		ns
3	ThA(AS)	Address to $\overline{AS}$ ↑ Hold Time (Note 1)	50		30		ns
4	TsA(DS)	Address to $\overline{DS}$ ↓ Setup Time (Note 1)	130		100		ns
5	TsCSO(AS)	$\overline{CS_0}$ to $\overline{AS}$ ↑ Setup Time (Note 1)	0		0		ns
6	ThCSO(AS)	$\overline{CS_0}$ to $\overline{AS}$ ↑ Hold Time (Note 1)	60		40		ns
7	TdAS(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↓ Delay (Note 1)	60		40		ns
8	TsCSI(DS)	CS_1 to $\overline{DS}$ ↓ Setup Time	100		80		ns
9	TsRWR(DS)	R/$\overline{W}$ (Read) to $\overline{DS}$ ↓ Setup Time	100		80		ns
10	TsRWW(DS)	R/$\overline{W}$ (Write) to $\overline{DS}$ ↓ Setup Time	0		0		ns
11	TwDS	$\overline{DS}$ Low Width	390		250		ns
12	TsDW(DSf)	Write Data to $\overline{DS}$ ↓ Setup Time	30		20		ns
13	TdDS(DRV)	$\overline{DS}$ (Read) ↓ to Address Data Bus Driven	0		0		ns
14	TdDSf(DR)	$\overline{DS}$ ↓ to Read Data Valid Delay		250		180	ns
15	ThDW(DS)	Write Data to $\overline{DS}$ ↑ Hold Time	30		20		ns
16	TdDSr(DR)	$\overline{DS}$ ↑ to Read Data Not Valid Delay	0		0	ns	
17	TdDS(DRz)	$\overline{DS}$ ↑ Read Data Float Delay (Note 2)		70		45	ns
18	ThRW(DS)	R/$\overline{W}$ to $\overline{DS}$ ↑ Hold Time	55		40		ns
19	ThCSI(DS)	CS_1 to $\overline{DS}$ ↑ Hold Time	55		40		ns
20	TdDS(AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	50		25		ns
21	Trc	Valid Acces Recovery Time (Note 3)	1000		650		ns
Z8036 INTERRUPT TIMING (Figure 19)							
22	TdPM(INT)	Pattern Match to INT Delay (Bit Port)		$\overline{AS}$ cycle +800ns		1	AS cycle + ns
23	TdACK(INT)	$\overline{ACKIN}$ to $\overline{INT}$ Delay (Port with Handshake) (Note 4)		4 $\overline{AS}$ cycles +600ns		4	AS cycle + ns
24	TdCI(INT)	Counter Input to $\overline{INT}$ Delay (Counter Mode)		$\overline{AS}$ cycle +700ns		1	AS cycle + ns
25	TdPC(INT)	PCLK to $\overline{INT}$ Delay (Timer Mode)		$\overline{AS}$ cycle +700ns		1	AS cycle + ns
26	TdAS(INT)	$\overline{AS}$ to $\overline{INT}$ Delay		300			ns
Z8036 INTERRUPT ACKNOWLEDGE TIMING (Figure 20)							
27	TsIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Setup Time	0		0		ns
28	ThIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Hold Time	250		250		ns
29	TsAS(DSA)	$\overline{AS}$ ↑ to $\overline{DS}$ (Acknowledge) ↓ Setup Time (Note 5)	350		250		ns
30	TdDSA(DR)	$\overline{DS}$ (Acknowledge) ↓ to Read Data Valid Delay		250		180	ns
31	TwDSA	$\overline{DS}$ (Acknowledge) Low Width	390		250		ns
32	TdAS(IEO)	$\overline{AS}$ ↓ to IEO ↓ Delay ($\overline{INTACK}$ Cycle) (Note 5)		350		250	ns
33	TdIEI(IEO)	IEI to IEO Delay (Note 5)		150		100	ns
34	TsIEI(DSA)	IEI to $\overline{DS}$ (Acknowledge) ↓ Setup Time (Note 5)	100		70		ns
35	ThIEI(DSA)	IEI to $\overline{DS}$ (Acknowledge) ↑ Hold Time	100		70		ns
36	TdDSA(INT)	$\overline{DS}$ (Acknowledge) ↓ to $\overline{INT}$ ↑ Delay		600		600	ns

Notes:
1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float Delay is measured to the time when the output has changed 0.5V from steady state with minimum ac load and maximum dc load.
3. This is the delay from $\overline{DS}$ ↑ of one CIO access to $\overline{DS}$ ↓ of another CIO access.
4. The delay is from $\overline{DAV}$ ↓ for 3-Wire Input Handshake. The delay is from DAC ↑ for 3-Wire Output Handshake. One additional $\overline{AS}$ cycle is required for ports in the Single Buffered mode.
5. The parameters for the devices in any particular daisy chain must meet the following constraint: the delay from $\overline{AS}$ ↑ to $\overline{DS}$ ↓ must be greater than the sum of TdAS(IEO) for the highest priority peripheral. TsIEI(DSA) for the lowest priority peripheral, and TdIEI(IEO) for each peripheral separating them in the chain.

*Timings are all preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 18. Z8036 CPU Interface Timing

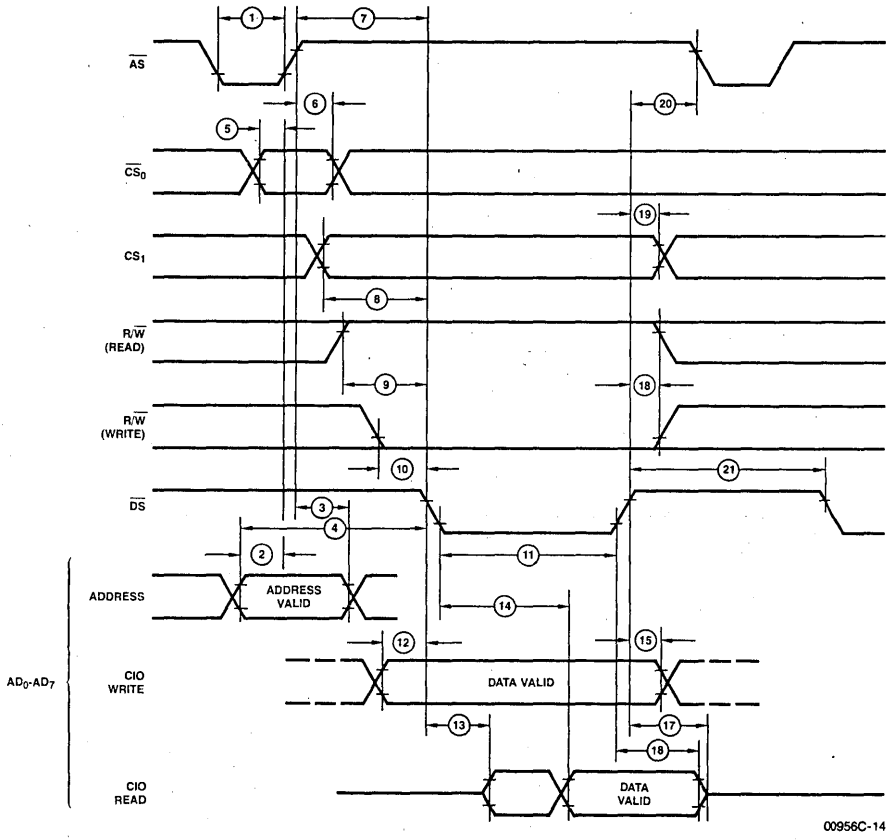


Figure 19. Z8036 Interrupt Timing

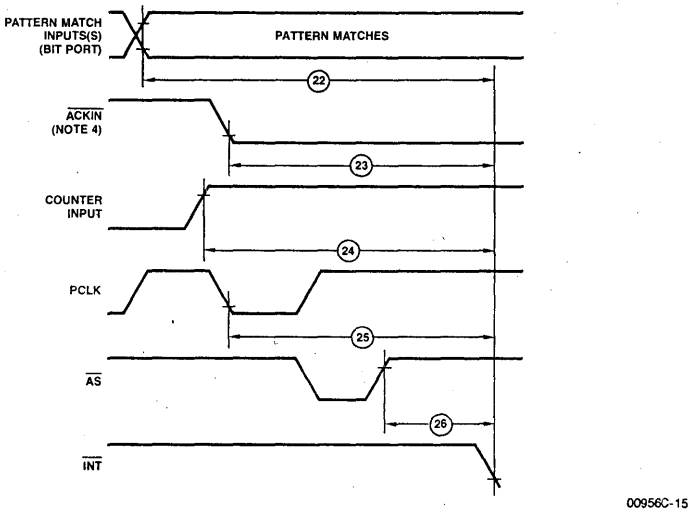
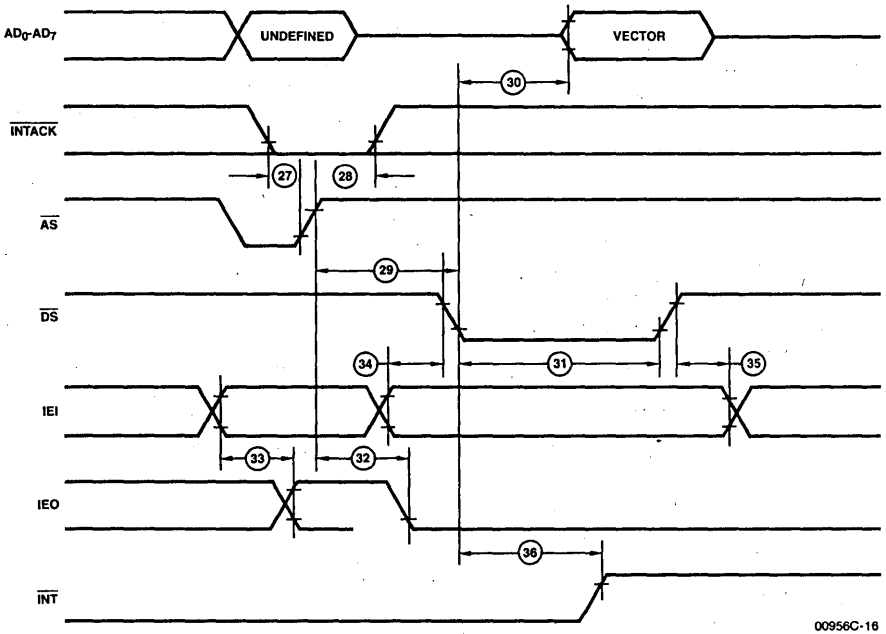


Figure 20. Z8036 Interrupt Acknowledge Timing



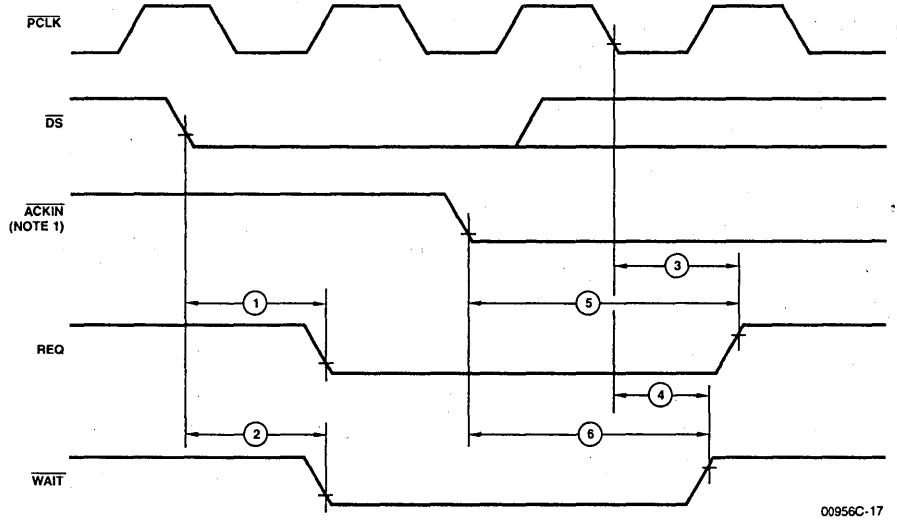
Z8036 REQUEST/$\overline{\text{WAIT}}$ TIMING (Figure 21)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdDS(REQ)	$\overline{\text{DS}}$ ↓ to REQ ↓ Delay		500			ns
2	TdDS(WAIT)	$\overline{\text{DS}}$ ↓ to $\overline{\text{WAIT}}$ ↓ Delay		500			ns
3	TdPC(REQ)	PCLK ↓ to REQ ↑ Delay		300			ns
4	TdPC(WAIT)	PCLK ↓ to $\overline{\text{WAIT}}$ ↑ Delay		300			ns
5	TdACK(REQ)	$\overline{\text{ACKIN}}$ ↓ to REQ ↑ Delay (Note 1)		3 $\overline{\text{AS}}$ + 2 PCLK + 1000ns			$\overline{\text{AS}}$ cycles + PCLK cycles + ns
6	TdACK(WAIT)	$\overline{\text{ACKIN}}$ ↓ to $\overline{\text{WAIT}}$ ↑ Delay		10 PCLK +600ns			PCLK cycles + ns

Note: 1. The delay is from $\overline{\text{DAV}}$ ↓ for 3-Wire Input Handshake. The delay is from DAC ↑ for 3-Wire Output Handshake.

*Timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 21. REQUEST/WAIT Timing



Z8536 CPU INTERFACE TIMING (Figure 22)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TcPC	PCLK Cycle Time	250	4000	165	4000	ns
2	TwPCh	PCLK Width (High)	105	2000	70	2000	ns
3	TwPCl	PCLK Width (Low)	105	2000	70	2000	ns
4	TrPC	PCLK Rise Time		20		10	ns
5	TfPC	PCLK Fall Time		20		15	ns
6	TslA(PC)	$\overline{INTACK}$ to PCLK ↑ Setup Time	100		100		ns
7	ThIA(PC)	$\overline{INTACK}$ to PCLK ↑ Hold Time	0		0		ns
8	TslA(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↓ Setup Time (Note 1)	200		200		ns
9	ThIA(RD)	$\overline{INTACK}$ to $\overline{RD}$ ↑ Hold Time	0		0		ns
10	TslA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↓ Setup Time	200		200		ns
11	ThIA(WR)	$\overline{INTACK}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
12	TsA(RD)	Address to $\overline{RD}$ ↓ Setup Time	80		80		ns
13	ThA(RD)	Address to $\overline{RD}$ ↑ Hold Time	0		0		ns
14	TsA(WR)	Address to $\overline{WR}$ ↓ Setup Time	80		80		ns
15	ThA(WR)	Address to $\overline{WR}$ ↑ Hold Time	0		0		ns
16	TsCEl(RD)	$\overline{CE}$ Low to $\overline{RD}$ ↓ Setup Time (Note 1)	0		0		ns
17	TsCEh(RD)	$\overline{CE}$ High to $\overline{RD}$ ↓ Setup Time (Note 1)	100		70		ns
18	ThCE(RD)	$\overline{CE}$ to $\overline{RD}$ ↑ Hold Time (Note 1)	0		0		ns
19	TsCEl(WR)	$\overline{CE}$ Low to $\overline{WR}$ ↓ Setup Time	0		0		ns
20	TsCEh(WR)	$\overline{CE}$ High to $\overline{WR}$ ↓ Setup Time	100		70		ns
21	ThCE(WR)	$\overline{CE}$ to $\overline{WR}$ ↑ Hold Time	0		0		ns
22	TwRDl	$\overline{RD}$ Low Width (Note 1)	390		250		ns
23	TdRD(DRA)	$\overline{RD}$ ↓ to Read Data Active Delay	0		0		ns
24	TdRDf(DR)	$\overline{RD}$ ↓ to Read Data Valid Delay		250		180	ns
25	TdRDr(DR)	$\overline{RD}$ ↑ to Read Data Not Valid Delay	0		0		ns
26	TdRD(DRz)	$\overline{RD}$ ↑ to Read Data Float Delay (Note 2)		70		45	ns
27	TwWRl	$\overline{WR}$ Low Width	390		250		ns
28	TsDW(WR)	Write Data to $\overline{WR}$ ↓ Setup Time	0		0		ns
29	ThDW(WR)	Write Data to $\overline{WR}$ ↑ Hold Time	0		0		ns
30	Trc	Valid Access Recovery Time (Note 3)	1000*		650*		ns
Z8536 INTERRUPT TIMING (Figure 23)							
31	TdPM(INT)	Pattern Match to $\overline{INT}$ Delay (Bit Port)		2		2	TcPC + ns
32	TdACK(INT)	$\overline{ACKIN}$ to $\overline{INT}$ Delay (Port with Handshake) (Note 4)		10		10	TcPC + ns
33	TdCl(INT)	Counter Input to $\overline{INT}$ Delay (Counter Mode)		2		2	TcPC + ns
34	TdPC(INT)	PCLK to $\overline{INT}$ Delay (Timer Mode)		3		3	TcPC + ns

Z8536 INTERRUPT ACKNOWLEDGE TIMING (Figure 24)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
35	TsIA(RDA)	$\overline{INTACK}$ to $\overline{RD}$ ↓ (Acknowledge) Setup Time (Note 5)	350		250		ns
36	TwRDA	$\overline{RD}$ (Acknowledge) Width	350		250		ns
37	TdRDA(DR)	$\overline{RD}$ ↓ (Acknowledge) to Read Data Valid Delay		255		180	ns
38	TdIA(IEO)	$\overline{INTACK}$ ↓ to IEO ↓ Delay (Note 5)		350		250	ns
39	TdIEI(IEO)	IEI to IEO Delay (Note 5)		150		100	ns
40	TsIEI (RDA)	IEI to $\overline{RD}$ ↓ (Acknowledge) Setup Time (Note 5)	100		70		ns
41	ThIEI(RDA)	IEI to $\overline{RD}$ ↑ (Acknowledge) Hold Time	100		70		ns
42	TdRDA(INT)	$\overline{RD}$ ↓ (Acknowledge) to $\overline{INT}$ ↑ Delay		600		600	ns

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float Delay is measured to the time when the output has changed 0.5V with minimum ac load and maximum dc load.
3. Trc is 1μs or 3 TcPC, whichever is longer.
4. The delay is from $\overline{DAV}$ ↓ for 3-Wire Input Handshake. The delay is from DAC ↑ for 3-Wire Output Handshake.
5. The parameters for the devices in any particular daisy chain must meet the following constraint: the delay from $\overline{INTACK}$ ↓ to $\overline{RD}$ ↓ must be greater than the sum of TdIA(IEO) for the highest priority peripheral. TsIEI(RDA) for the lowest priority peripheral, and TdIEI(IEO) for each peripheral separating them in the chain.

*Timings are preliminary and subject to change.

Figure 22. CPU Interface Timing.

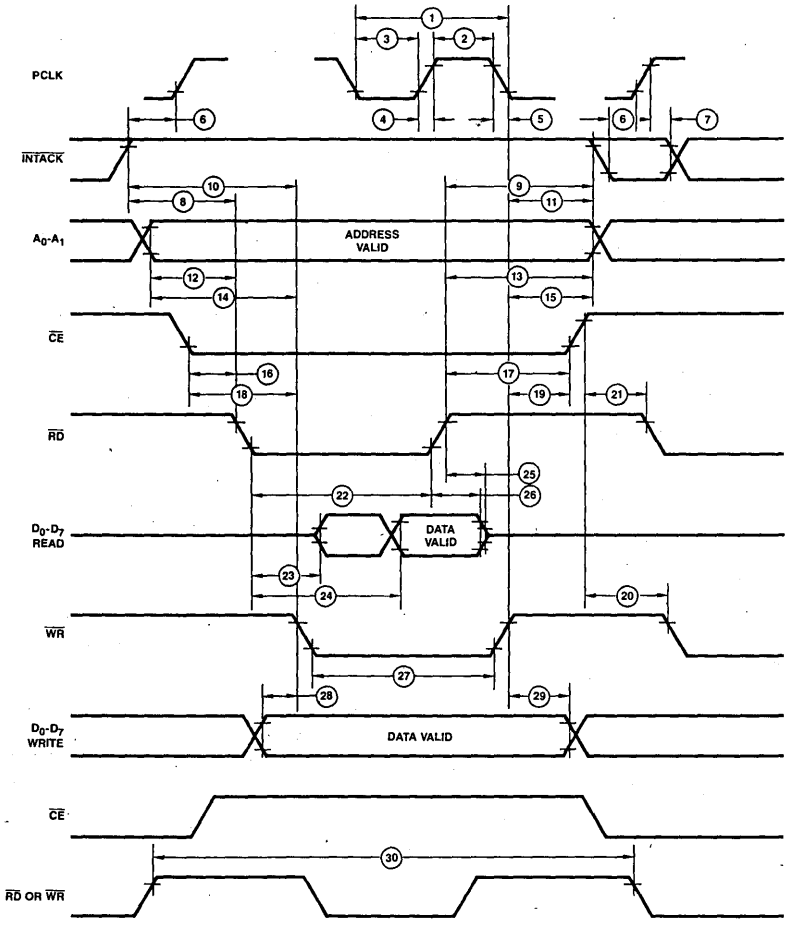


Figure 23. Interrupt Timing.

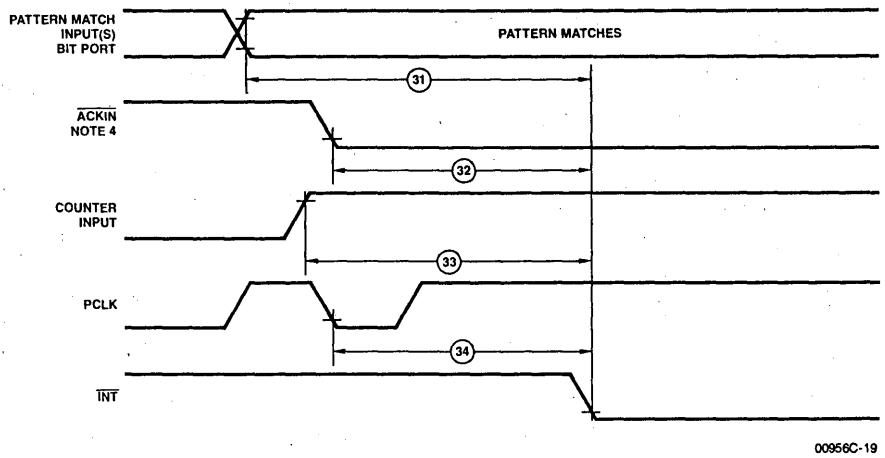
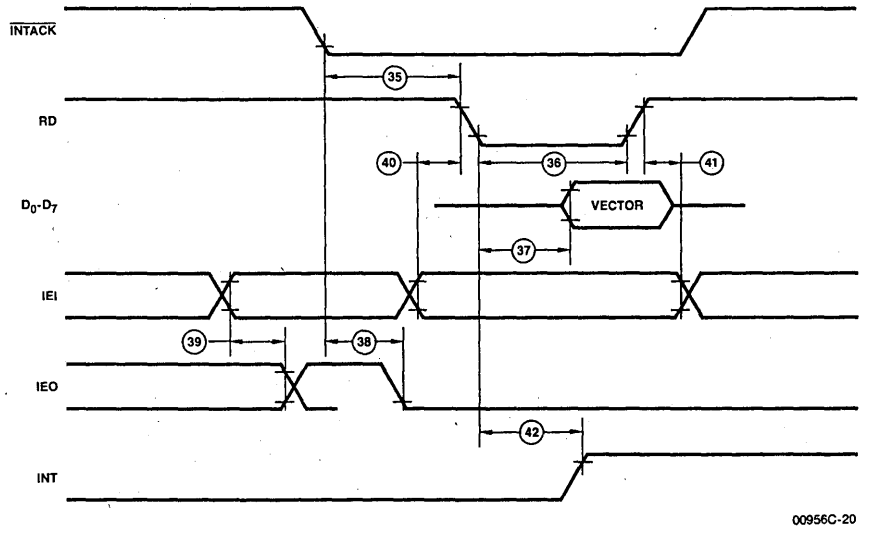


Figure 24. Interrupt Acknowledge Timing.



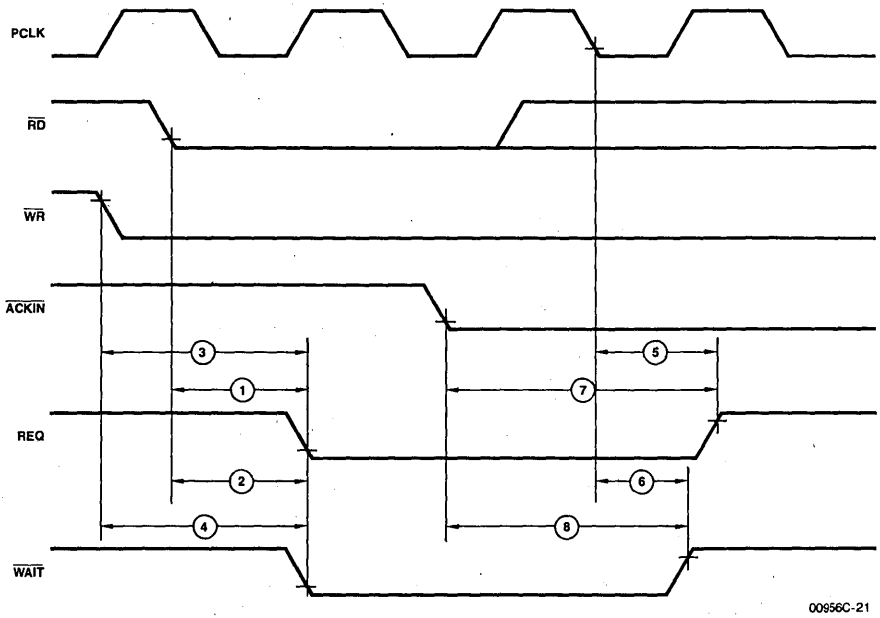
Z8536 REQUEST/$\overline{\text{WAIT}}$ TIMING (Figure 25)

Number	Parameter	Description	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TdRD(REQ)	$\overline{\text{RD}}$ ↓ to REQ ↓ Delay					ns
2	TdRD(WAIT)	$\overline{\text{RD}}$ ↓ to $\overline{\text{WAIT}}$ ↓ Delay					ns
3	TdWR(REQ)	$\overline{\text{WR}}$ ↓ to REQ ↓ Delay					ns
4	TdWR(WAIT)	$\overline{\text{WR}}$ ↓ to $\overline{\text{WAIT}}$ ↓ Delay					ns
5	TdPC(REQ)	PCLK ↓ to REQ ↑ Delay					ns
6	TdPC(WAIT)	PCLK ↓ to $\overline{\text{WAIT}}$ ↑ Delay					ns
7	TdACK(REQ)	$\overline{\text{ACKIN}}$ ↓ to REQ ↑ Delay (Note 1)					TcPC +ns
8	TdACK(WAIT)	$\overline{\text{ACKIN}}$ ↓ to $\overline{\text{WAIT}}$ ↑ Delay (Note 1)					TcPC +ns

Note: 1. The delay is from $\overline{\text{DAV}}$ ↓ for 3-Wire Input Handshake. The delay is from DAC ↑ for 3-Wire Output Handshake.

*Timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0".

Figure 25. Request/$\overline{\text{WAIT}}$ Timing.



AmZ8038 (FIO)

128 Byte FIFO I/O Port

DISTINCTIVE CHARACTERISTICS

- **Asynchronous FIFO Interface**
  128-byte FIFO provides bidirectional CPU to CPU or peripheral interface.
- **Expandable in Length and Width**
  FIOs can be connected in parallel for wider words, can be cascaded for deeper stacks.
- **2-Wire and 3-Wire handshake logic**
  Control logic on chip for interlocked two-wire handshake as well as three-wire scheme used in IEEE-488.
- **Pattern matching logic on chip**
  FIO can detect a data pattern and interrupt CPU.
- **Byte count available to software**
  An on-chip register which contains the actual number of bytes in the FIFO can be read by the software to determine stack status.

GENERAL DESCRIPTION

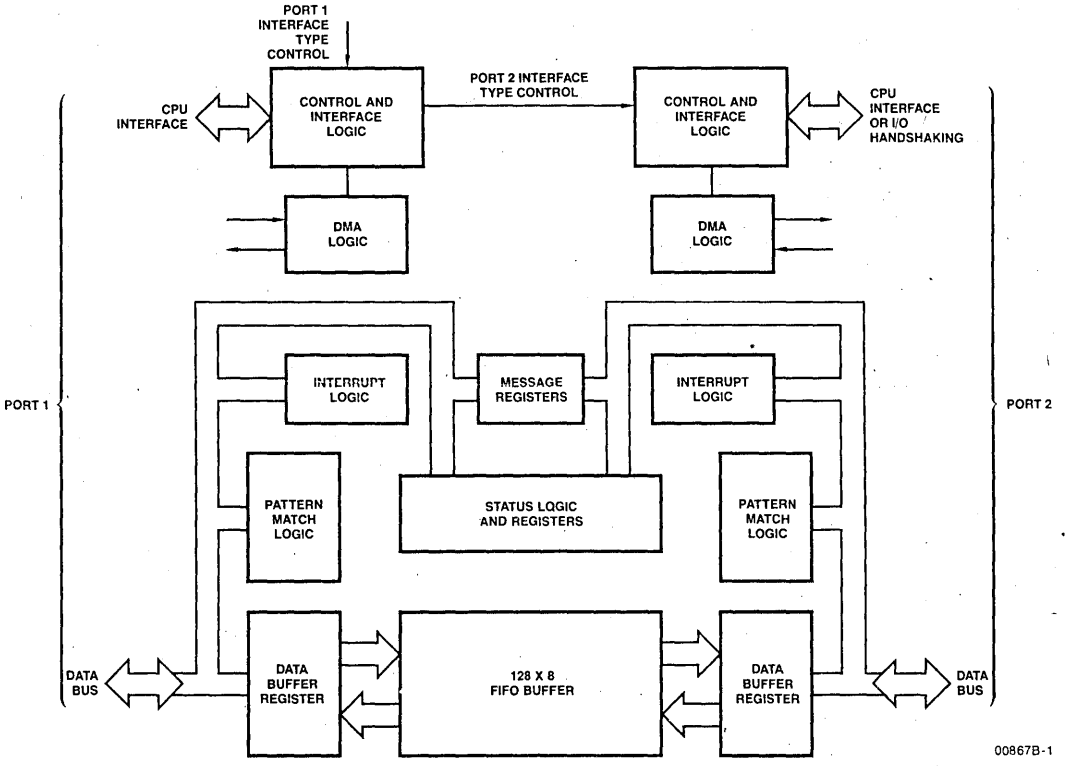
The AmZ8038* FIO provides an asynchronous 128-byte FIFO buffer between two CPUs or between a CPU and a peripheral device. This buffer interface expands to a 16-bit or wider data path and expands in depth to add as many AmZ8060 FIFOs (and an additional FIO) as are needed.

The FIO manages data transfers by assuming Z-BUS, non-Z-BUS microprocessor (a generalized microprocessor interface), Interlocked 2-Wire Handshake, and 3-Wire Handshake operating modes. These modes interface dissimilar CPUs or CPUs and peripherals running under differing speeds or protocols, allowing asynchronous data transactions and improving I/O overhead by as much as two orders of magnitude.

The FIO supports the Z-BUS interrupt protocols, generating seven sources of interrupts. Each interrupt source can be enabled or disabled, and can also place an interrupt vector on the port address/data lines.

The data transfer logic of the FIO has been specially designed to work with DMA (Direct Memory Access) devices for high-speed transfers. The FIO also supports the variably sized block length, improving system throughput when multiple variable length messages are transferred amongst several sources.

FIO BLOCK DIAGRAM



*Z8000 is a trademark of Zilog, Inc.

ARCHITECTURE

The FIO is a universal interface between two independent systems operating asynchronously. Conceptually it consists of two programmable interfaces connected by a 128-byte FIFO buffer and a pair of message registers, as shown in Figure 1.

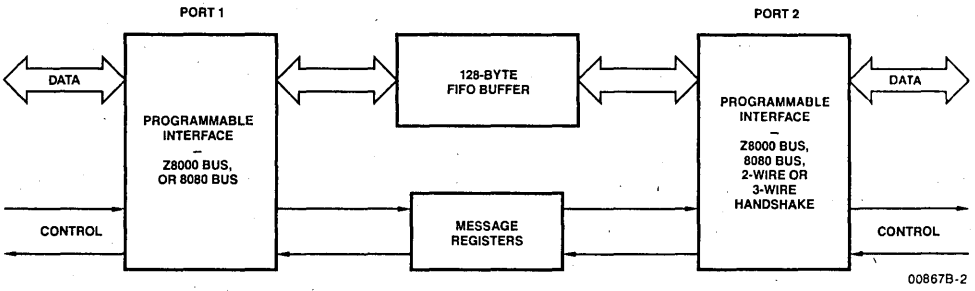
Each port contains 8 lines used for data and for programming the FIO, and 10 lines used for various control functions. The function and timing relationships of the 10 control lines change completely according to the type of interface the port is programmed to be. Either port can be programmed to interface directly to a Z8000 CPU Bus (Z-bus) or to an 8080 type bus (8080, 8085, Z80). The 10 control lines perform the functions needed for read and write strobes, DMA control, and interrupt handling. The port 2 side can also be programmed for interface to peripheral; in this case the 10 lines are used for handshaking, direction control, and cascading.

The two sides are connected by the FIFO buffer and the message registers. The buffer is used to move data between the two ports. Since it is a FIFO, reads and writes can occur simultaneously and independently. Each port is tied to its own system interface, with the FIFO providing reliable data transfer between them. The message registers are used to transmit information between the two ports without going through the FIFO. They (one going each direction) are intended for control words being sent from one CPU to another.

The CPU interface includes a vectored interrupt capability which covers all the ways in which a transfer might be terminated. The CPU, therefore, can set up the FIO to communicate with a peripheral or another CPU, and then not deal with the FIO until an interrupt occurs. Interrupts can be programmed to occur on any of these conditions:

Buffer Empty
Buffer Full
Overflow/Underflow
Byte Count Match
Data Pattern Match
Data Direction Change
Message Present

Figure 1. FIO Concept



ORDERING INFORMATION

Example Z8038 A D C B

Basic Device

Speed
Blank = 4MHz
A = 6MHz

Burn-in

Temperature Range
C = 0 to +70°C
I = −40 to +85°C
M = −55 to +125°C

Package Type
D = Hermetic DIP
P = Molded Plastic

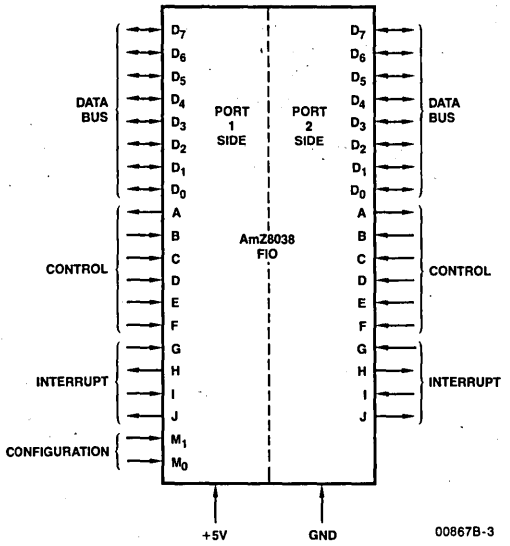
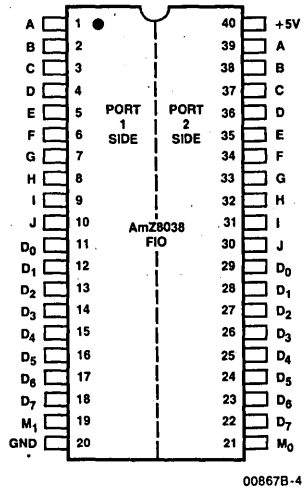


Figure 2. Pin Functions

D-40, P-40



Note: Pin 1 is marked for orientation.

Figure 3. Pin Assignments

FUNCTIONAL DESCRIPTION

OPERATING MODES

Ports 1 and 2 operate in any of twelve combinations of operating modes, listed in Table 2. Port 1 functions in either the Z-BUS or non-Z-BUS microprocessor modes, while Port 2 functions in Z-BUS, non-Z-BUS, Interlocked 2-Wire Handshake, and 3-Wire Handshake modes. Table 1 describes the signals and their corresponding pins in each of these modes.

The pin diagrams of the FIO are identical, except for two pins on the Port 1 side, which select that port's operating mode. Port 2's operating mode is programmed by two bits in Port 1's Control register 0. Table 2 describes the combinations of operating modes; Table 3 describes the control signals mapped to pins A-J in the five possible operating modes.

RESET

The FIO can be reset under either hardware or software control by one of the following methods:

- By forcing both $\overline{AS}$ and $\overline{DS}$ LOW simultaneously in Z-BUS mode (normally illegal).
- By forcing $\overline{RD}$ and $\overline{WR}$ LOW simultaneously in non-Z-BUS mode.
- By writing a 1 to the Reset bit in Control register 0 for software reset.

In the Reset state, all control bits are cleared to 0. Only after clearing the Reset bit (by writing a 0 to it) can the other command bits be programmed. This action is true for both sides of the FIO when programmed as a CPU interface.

TABLE 1. PIN ASSIGNMENTS

Control Signal Pins	Z-BUS Low Byte	Z-BUS High Byte	Non-Z-BUS	Port 2 Only	
				Interlocked HS Port	3-Wire HS Port
A	$\overline{REQ}/\overline{WT}$	$\overline{REQ}/\overline{WT}$	$\overline{REQ}/\overline{WT}$	RFD/$\overline{DAV}$	RFD/$\overline{DAV}$
B	$\overline{DMASTB}$	$\overline{DMASTB}$	$\overline{DACK}$	$\overline{ACKIN}$	$\overline{DAV}$/DAC
C	$\overline{DS}$	$\overline{DS}$	$\overline{RD}$	FULL	DAC/RFD
D	R/$\overline{W}$	R/$\overline{W}$	$\overline{WR}$	EMPTY	EMPTY
E	$\overline{CS}$	$\overline{CS}$	$\overline{CE}$	$\overline{CLEAR}$	$\overline{CLEAR}$
F	$\overline{AS}$	$\overline{AS}$	C/$\overline{D}$	DATA DIR	DATA DIR
G	$\overline{INTACK}$	A_0	$\overline{INTACK}$	IN_0	IN_0
H	IEO	A_1	IEO	OUT_1	OUT_1
I	IEI	A_2	IEI	OE	OE
J	$\overline{INT}$	A_3	$\overline{INT}$	OUT_3	OUT_3

For proper system control, when Port 1 is reset, Port 2 is also reset. In addition, all Port 2's outputs are floating and all inputs are ignored. To initiate the data transfer, Port 2 must be enabled by Port 1. The Port 2 CPU can determine when it is enabled by reading Control register 0, which reads "floating" data bus if not enabled and "01_H" if enabled.

TABLE 2. OPERATING MODES

Mode	M_1	M_0	B_1*	B_0*	Port 1	Port 2
0	0	0	0	0	Z-BUS Low Byte	Z-BUS Low Byte
1	0	0	0	1	Z-BUS Low Byte	Non-Z-BUS
2	0	0	1	0	Z-BUS Low Byte	3-Wire Handshake
3	0	0	1	1	Z-BUS Low Byte	2-Wire Handshake
4	0	1	0	0	Z-BUS High Byte	Z-BUS High Byte
5	0	1	0	1	Z-BUS High Byte	Non-Z-BUS
6	0	1	1	0	Z-BUS High Byte	3-Wire Handshake
7	0	1	1	1	Z-BUS High Byte	2-Wire Handshake
8	1	0	0	0	Non-Z-BUS	Z-BUS Low Byte
9	1	0	0	1	Non-Z-BUS	Non-Z-BUS
10	1	0	1	0	Non-Z-BUS	3-Wire Handshake
11	1	0	1	1	Non-Z-BUS	2-Wire Handshake

*Bits 3 and 2 of control register 0. Read/Write from Port 1, Read-only from Port 2.

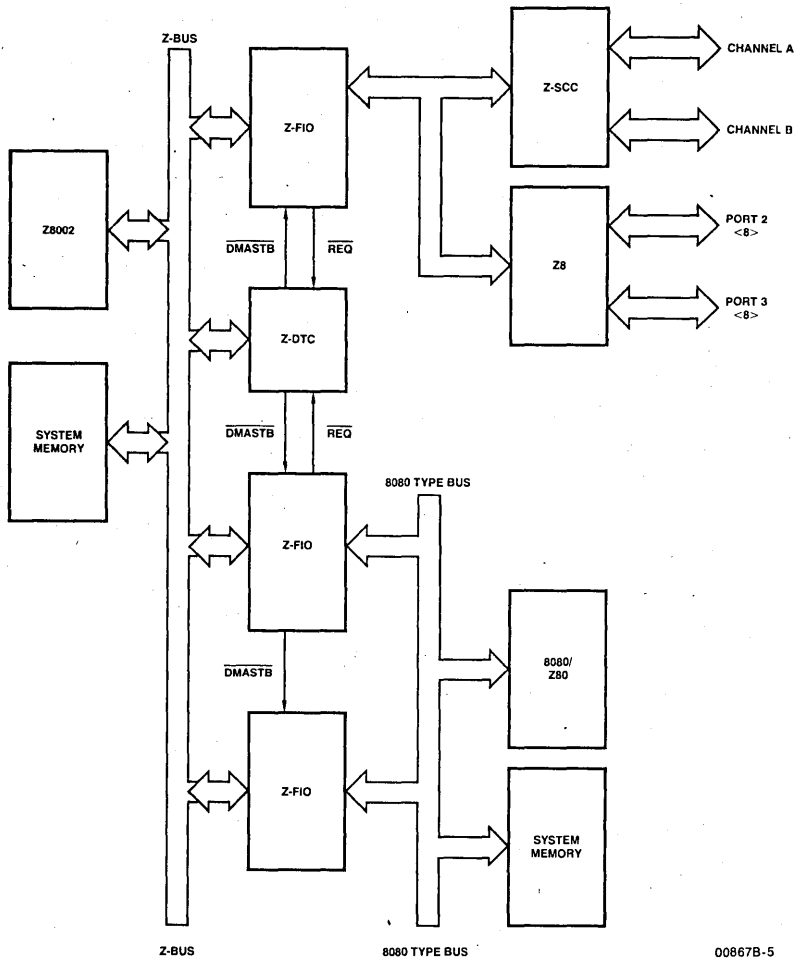


Figure 4. CPU to CPU Configuration

TABLE 3. SIGNAL/PIN DESCRIPTIONS

Pin Signals	Pin Names	Pin Numbers		Signal Description
PINS COMMON TO BOTH SIDES				
M_0	M_0	21		M_1 and M_0 program Port 1 side CPU interface
M_1	M_1	19		
+5 Vdc	+5 Vdc	40		DC power source
GND	GND	20		DC power ground
Pin Signals	**Pin Names**	**Pin Numbers Port 1**	**Port 2**	**Signal Description**
Z-BUS LOW BYTE MODE				
AD_0-AD_7 (Address/Data)	D_0-D_7	11-18	29-22	Multiplexed bidirectional address/data lines, Z-BUS compatible.
$\overline{REQ}/\overline{WAIT}$ (Request/Wait)	A	1	39	Output, active Low, REQUEST (ready) line for DMA transfer; $\overline{WAIT}$ line (open-drain) output for synchronized CPU and FIO data transfers.
$\overline{DMASTB}$ (Direct Memory Access Strobe)	B	2	38	Input, active Low. Strobes DMA data to and from the FIFO buffer.
$\overline{DS}$ (Data Strobe)	C	3	37	Input, active Low. Provides timing for data transfer to or from FIO.
$R/\overline{W}$ (Read/Write)	D	4	36	Input; active High signals CPU read from FIO; active Low signals CPU write to FIO.
$\overline{CS}$ (Chip Select)	E	5	35	Input, active Low. Enables FIO. Latched on the rising edge of $\overline{AS}$.
$\overline{AS}$ (Address Strobe)	F	6	34	Input, active Low. Addresses; $\overline{CS}$ and $\overline{INTACK}$ sampled while $\overline{AS}$ Low.
$\overline{INTACK}$ (Interrupt Acknowledge)	G	7	33	Input, active Low. Acknowledges an interrupt. Latched on the rising edge of $\overline{AS}$.
IEO (Interrupt Enable Out)	H	8	32	Output, active High. Sends interrupt enable to lower priority device IEI pin.
IEI (Interrupt Enable In)	I	9	31	Input, active High. Receives interrupt enable from higher priority device IEO signal.
$\overline{INT}$ (Interrupt)	J	10	30	Output, open drain, active Low. Signals FIO interrupt request to CPU.
Z-BUS HIGH BYTE MODE				
AD_0-AD_7 (Address/Data)	D_0-D_7	11-18	29-22	Multiplexed bidirectional address/data lines, Z-BUS compatible.
$\overline{REQ}/\overline{WAIT}$ (Request/Wait)	A	1	39	Output, active Low, REQUEST (ready) line for DMA transfer; $\overline{WAIT}$ line (open-drain) output for synchronized CPU and FIO data transfers.
$\overline{DMASTB}$ (Direct Memory Access Strobe)	B	2	38	Input, active Low. Strobes DMA data to and from the FIFO buffer.
$\overline{DS}$ (Data Strobe)	C	3	37	Input, active Low. Provides timing for transfer of data to or from FIO.
$R/\overline{W}$ (Read/Write)	D	4	36	Input, active High. Signals CPU read from FIO; active Low Signals CPU write to FIO.
$\overline{CS}$ (Chip Select)	E	5	35	Input, active Low. Enables FIO. Latched on the rising edge of $\overline{AS}$.
$\overline{AS}$ (Address Strobe)	F	6	34	Input, active Low. Addresses, $\overline{CS}$ and $\overline{INTACK}$ are sampled while $\overline{AS}$ is Low.
A_0 (Address Bit 0)	G	7	33	Input, active High. With A_1, A_2, and A_3, addresses FIO internal registers.
A_1 (Address Bit 1)	H	8	32	Input, active High. With A_0, A_2, and A_3, addresses FIO internal registers.
A_2 (Address Bit 2)	I	9	31	Input, active High. With A_0, A_1, and A_3, addresses FIO internal registers.
A_3 (Address Bit 3)	J	10	30	Input, active High. With A_0, A_1, and A_2, addresses FIO internal registers.

TABLE 3. SIGNAL/PIN DESCRIPTIONS (Cont.)

Pin Signals	Pin Names	Pin Numbers Port 1	Port 2	Signal Description
NON-Z-BUS MODE				
D_0-D_7 (Data)	D_0-D_7	11-18	29-22	Bidirectional data bus.
$\overline{REQ}/\overline{WT}$ (Request/Wait)	A	1	39	Output, active Low. REQUEST (ready) line for DMA transfer; $\overline{WAIT}$ line (open-drain) output for synchronized CPU and FIO data transfer.
$\overline{DACK}$ (DMA Acknowledge)	B	2	38	Input, active Low. DMA acknowledge.
$\overline{RD}$ (Read)	C	3	37	Input, active Low. Signals CPU read from FIO.
$\overline{WR}$ (Write)	D	4	36	Input, active Low. Signals CPU write to FIO.
$\overline{CE}$ (Chip Select)	E	5	35	Input, active Low. Used to select FIO.
$C/\overline{D}$ (Control/Data)	F	6	34	Input, active High, identifies control byte on D_0-D_7; active Low identifies data byte on D_0-D_7.
$\overline{INTACK}$ (Interrupt Acknowledge)	G	7	33	Input, active Low. Acknowledges an interrupt.
IEO (Interrupt Enable Out)	H	8	32	Output, active High. Sends interrupt enable to lower priority device IEI pin.
IEI (Interrupt Enable In)	I	9	31	Input, active High. Receives interrupt enable from higher priority device IEO signal.
$\overline{INT}$ (Interrupt)	J	10	30	Output, open drain, active Low. Signals FIO interrupt to CPU.
Pin Signals	**Pin Names**	**Pin Numbers**	**Mode**	**Signal Description**
PORT 2 — I/O PORT MODE				
D_0-D_7 (Data)	D_0-D_7	29-22	2-Wire HS* 3-Wire HS	Bidirectional data bus.
RFD/$\overline{DAV}$ (Ready for Data/Data Available)	A	39	2-Wire HS 3-Wire HS	Output, RFD active High. Signals peripherals that FIO is ready to receive data. $\overline{DAV}$ active Low signals that FIO is ready to send data to peripherals.
$\overline{ACKIN}$ (Acknowledge Input)	B	38	2-Wire HS	Input, active Low. Signals FIO that output data is received by peripherals or that input data is valid.
$\overline{DAV}$/DAC (Data Available/Data Accepted)	B	38	3-Wire HS	Input; $\overline{DAV}$ (active Low) signals that data is valid on bus. DAC (active High) signals that output data is accepted by peripherals.
FULL	C	37	2-Wire HS	Output, open drain, active High. Signals that FIO buffer is full.
DAC/RFD (Data Accepted/Ready for Data)	C	37	3-Wire HS	Direction controlled by internal programming. Both active High. DAC (an output) signals that FIO has received data from peripheral; RFD (an input) signals that the listeners are ready for data.
EMPTY	D	36	2-Wire HS 3-Wire HS	Output, open drain, active High. Signals that FIFO buffer is empty.
$\overline{CLEAR}$	E	35	2-Wire HS 3-Wire HS	Programmable input or output, active Low. Clears all data from FIFO buffer.
DATA DIR (Data Direction)	F	34	2-Wire HS 3-Wire HS	Programmable input or output. Active High signals data input to Port 2; Low signals data output from Port 2.
IN_0	G	33	2-Wire HS 3-Wire HS	Input line to D_0 of Control Register 3.
OUT_1	H	32	2-Wire HS 3-Wire HS	Output line from D_1 of Control Register 3.
$\overline{OE}$ (Output Enable)	I	31	2-Wire HS 3-Wire HS	Input, active Low. When Low, enable bus drivers. When High, floats bus drivers at high impedance.
OUT_3	J	30	2-Wire HS 3-Wire HS	Output line from D_3 of Control register 3.

*Handshake

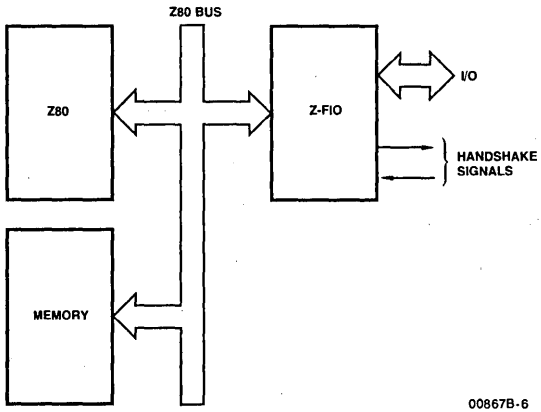


Figure 5. CPU to I/O Configuration

CPU INTERFACES

The FIO is designed to work with both Z-BUS and non-Z-BUS-type CPUs, on both Port 1 and Port 2. The Z-BUS configuration interfaces CPUs with time-multiplexed address and data information on the same pins. The AmZ8001, AmZ8002, and Z8 are examples of this type of CPU. The $\overline{AS}$ (Address Strobe) pin is used to latch the address and chip select information sent out by the CPU. The R/$\overline{W}$ (Read/Write) pin and the $\overline{DS}$ (Data Strobe) pin are used for timing reads and writes from the CPU to the FIO (Figures 6 and 7).

The non-Z-BUS configuration is used for CPUs where the address and data buses are separate. Examples of this type of CPU are the Z80 and 8080. The $\overline{RD}$ (Read) and $\overline{WR}$ (Write) pins are used to time reads and writes from the CPU to the FIO (Figures 9 and 10). The C/$\overline{D}$ (Control/Data) pin is used to directly access the FIFO buffer (C/$\overline{D}$=0) and to access the other registers (C/$\overline{D}$=1). Read and write to all registers except the FIFO buffer[1] are the two-step operations, described as follows (Figures 8). First, write the address of the register to be accessed with C/$\overline{D}$ = 1. The address goes into a pointer register, and the FIO switches to

[1]The FIFO buffer can also be accessed by this two-step operation.

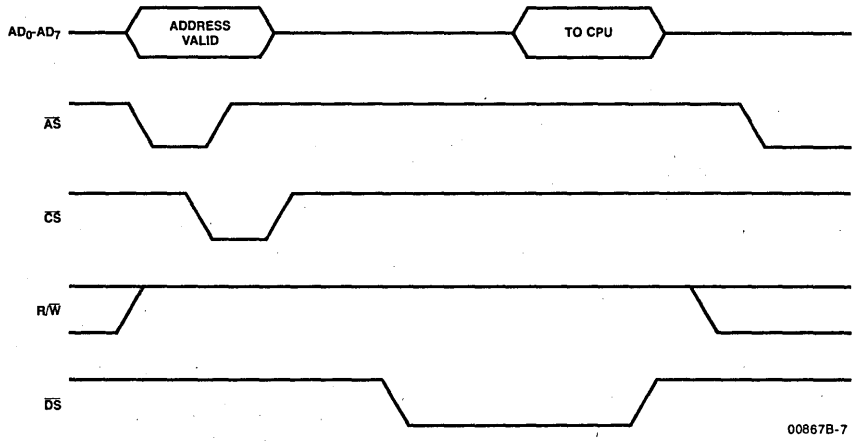


Figure 6. Z-BUS Read Cycle Timing

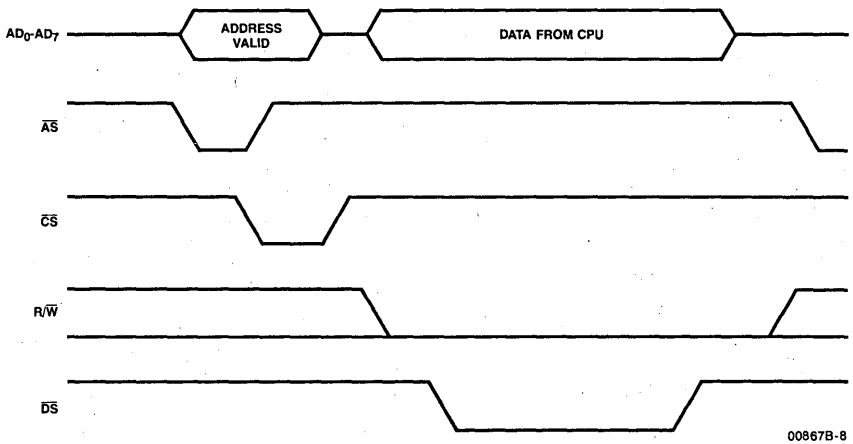


Figure 7. Z-BUS Write Cycle Timing

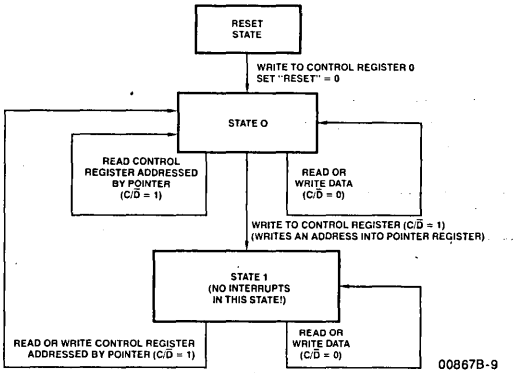


Figure 8. In Non-Z-BUS Mode, Control Registers are Accessed by First Writing the Address of the Register, then Reading or Writing the Contents of the Addressed Register.

state 1. The next read or write with C/$\overline{D}$ = 1 will be to the register pointed to. Continuous status monitoring can be performed by continuous Control Read operations (C/$\overline{D}$=1).

$\overline{WAIT}$ OPERATION

When data is output from the CPU, the $\overline{REQ}/\overline{WT}/(\overline{WAIT})$ pin is active (LOW) only when the FIFO buffer is full, the chip is selected, and FIFO buffer is addressed. $\overline{WAIT}$ goes inactive when the FIFO buffer is not full.

When data is input to the CPU, the $\overline{REQ}/\overline{WT}$ pin becomes active (LOW) only when the FIFO buffer is empty, the chip is selected, and the FIFO buffer is addressed. $\overline{WAIT}$ goes inactive when the FIFO buffer is not empty.

INTERRUPT OPERATION

The FIO supports Zilog's prioritized daisy chain interrupt protocol for both Z-BUS and non-Z-BUS operating modes.

Each side of the FIO has seven sources of interrupt. The priorities of these devices are fixed in the following order (highest to lowest): Mailbox message, Change in Data Direction, Pattern Match, Status Match, Overflow/Underflow Error, Buffer Full, and Buffer Empty. Each interrupt source has three bits that control how it generates the interrupt. These bits are Interrupt Pending (IP), Interrupt Enable (IE) and Interrupt Under Service (IUS).

In addition, each side of the FIO has an interrupt vector and four bits controlling the FIO interrupt logic. These bits are Vector Includes Status (VIS), Master Interrupt Enable (MIE), Disable Lower Chain (DLC) and No Vector (NV).

A typical Interrupt Acknowledge cycle for Z-BUS operation is shown in Figure 11 and for non-Z-BUS operation in Figure 12. The only difference is that in Z-BUS mode, $\overline{INTACK}$ is latched by $\overline{AS}$ and in non-Z-BUS mode $\overline{INTACK}$ is not latched.

When MIE = 1, reading the vector always includes status, independent of the state of the VIS bit. In this way, when VIS = 0, all information can be obtained with one additional read, thus conserving vector space. When MIE = 0, reading the vector register returns the unmodified base vector so that it can be verified.

In non-Z-BUS mode, IPs do not get set while in State 1. Therefore, in order to minimize interrupt latency, the FIO should be left in State 0.

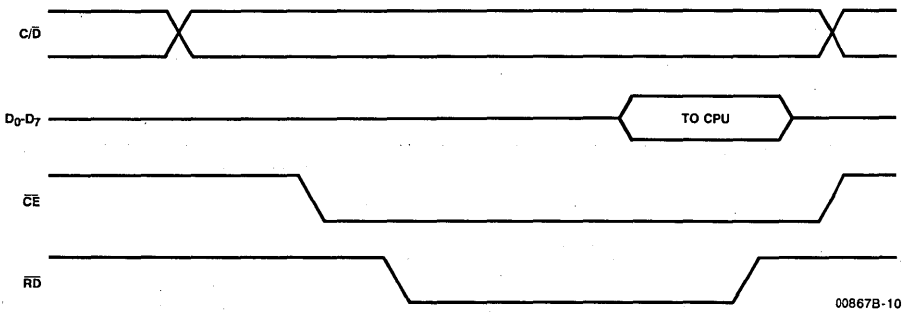


Figure 9. Non-Z-BUS Read Cycle Timing

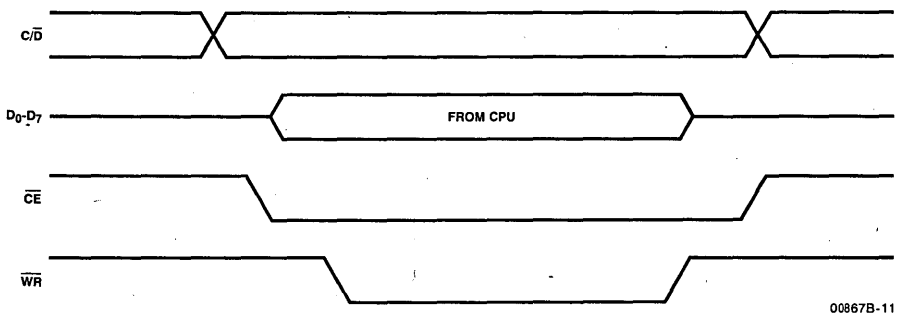


Figure 10. Non-Z-BUS Write Cycle Timing

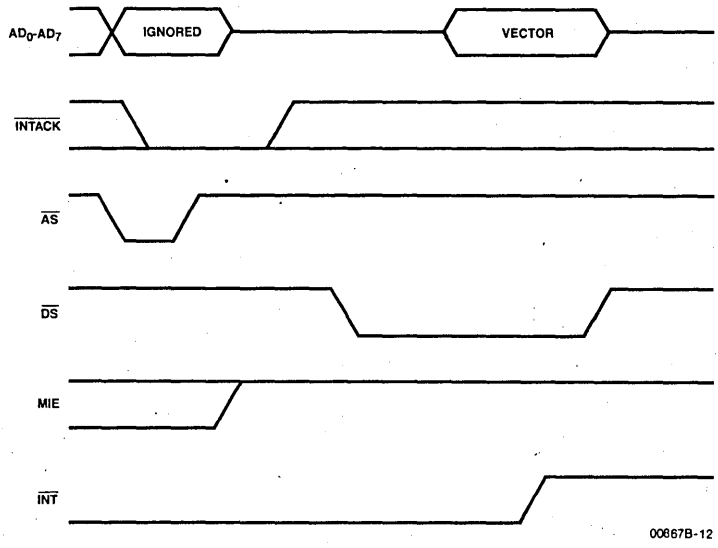


Figure 11. Z-BUS Interrupt Acknowledge Cycle

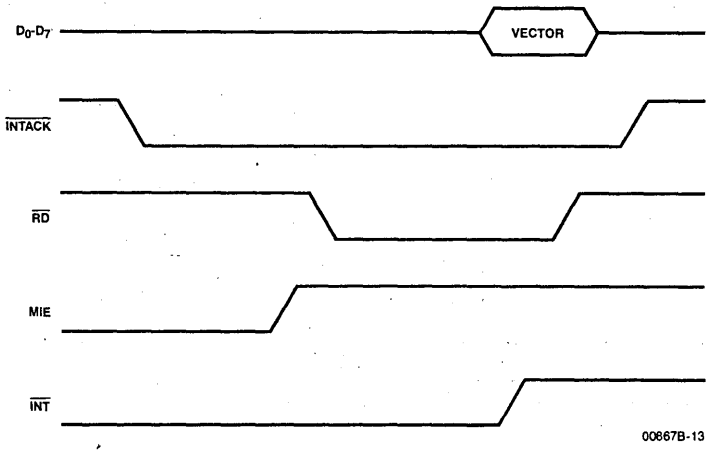


Figure 12. Non-Z-BUS Interrupt Acknowledge Cycle

CPU TO CPU OPERATIONS

DMA Operation

The FIO is particularly well suited to work with a DMA in both Z-BUS and non-Z-BUS modes. A data transfer between the FIO and system memory can take place during every machine cycle on both sides of the FIO simultaneously.

In Z-BUS mode, the $\overline{\text{DMASTB}}$ pin (DMA Strobe) is used to read or write into the FIFO buffer. The R/$\overline{\text{W}}$ (Read/Write) and $\overline{\text{DS}}$ (Data Strobe) signals are ignored by the FIO; however, the $\overline{\text{CS}}$ (Chip Select) signal is not ignored and therefore must be kept invalid. Figures 13 and 14 show typical timing.

In Non-Z-BUS mode the $\overline{\text{DACK}}$ pin (DMA Acknowledge) is used to tell the FIO that its DMA request is granted. After $\overline{\text{DACK}}$ goes Low, every read or write to the FIO goes into the FIFO buffer. Figures 15 and 16 show typical timing.

The FIO provides a special mode to enhance its DMA transfer capability. When data is written into the FIFO buffer, the $\overline{\text{REQ}}$/$\overline{\text{WT}}$ (REQUEST) pin is active (LOW) until the FIFO buffer is full. It then goes inactive and stays inactive until the number of bytes in the FIFO buffer is equal to the value programmed into the Byte Count Comparison register. Then the REQUEST signal goes active and the sequence starts over again (Figure 17).

When data is read from the FIO, the $\overline{\text{REQ}}$/$\overline{\text{WT}}$ pin (REQUEST) is inactive until the number of bytes in the FIFO buffer is equal to the value programmed in the Byte Count Comparison register. The REQUEST signal then goes active and stays active until the FIFO buffer is empty. When empty, REQUEST goes inactive and the sequence starts over again (Figure 18).

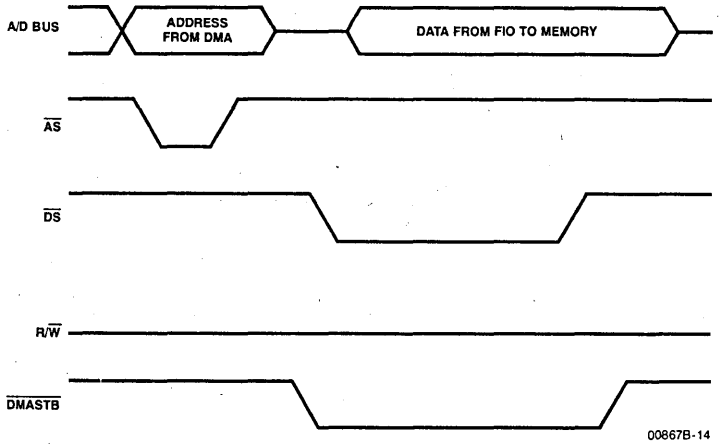


Figure 13. Z-BUS FIO to Memory Data Transaction

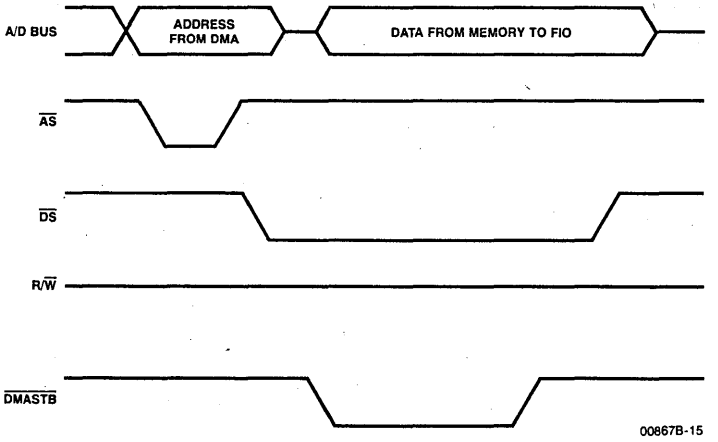


Figure 14. Z-BUS Memory to FIO Data Transaction

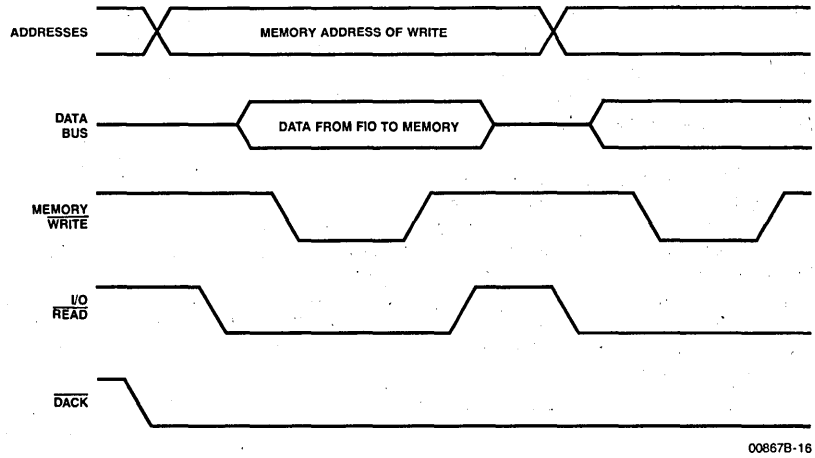


Figure 15. Non-Z-BUS FIO to Memory Transaction

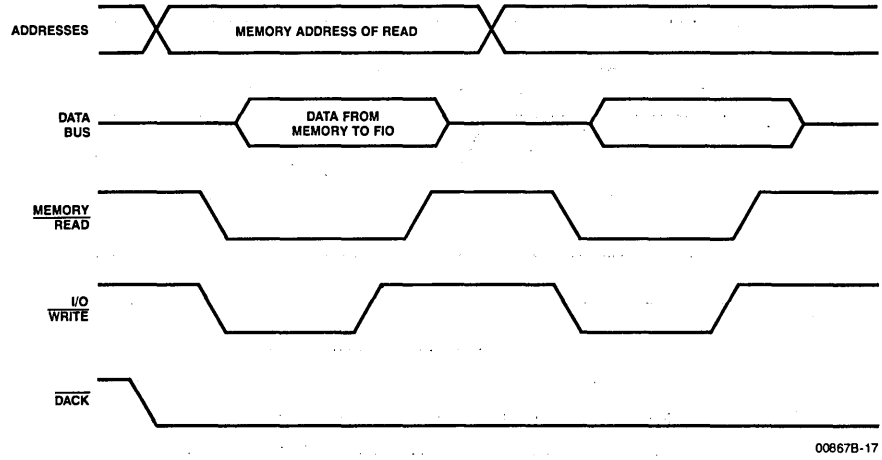
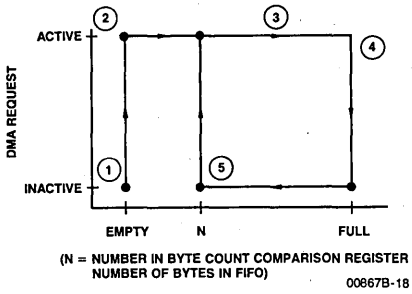
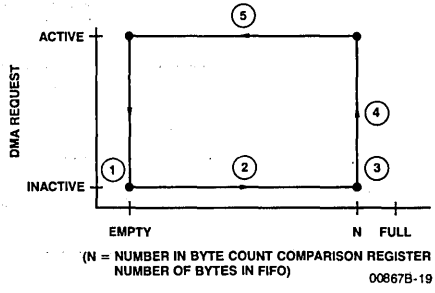


Figure 16. Non-Z-BUS Memory to FIO Data Transaction



Notes: 1. FIFO empty.
2. REQUEST enabled, FIO request DMA transfer.
3. DMA transfers data into the FIO.
4. FIFO full, REQUEST inactive.
5. The FIFO empties from the opposite port until the number of bytes in the FIFO buffer is the same as the number programmed in the Byte Count Comparison register.

Figure 17. Byte Count Control: Write to FIO



Notes: 1. FIFO empty.
2. CPU/DMA fills FIFO buffer from the opposite port.
3. Number of bytes in FIFO buffer is the same as the number of bytes programmed in the Byte Count Comparison register.
4. REQUEST goes active.
5. DMA transfers data out of FIFO until it is empty.

Figure 18. Byte Count Control: Read from FIO

Message Registers

Two CPUs can communicate through a dedicated "mailbox" register without involving the 128 x 8 bit FIFO buffer (Figure 19). This mailbox approach is useful for transferring control parameters between the interfacing devices on either side of the FIO without using the FIFO buffer. For example, when Port 1's CPU writes to the Message Out register, Port 2's message IP is set. If interrupts are enabled, Port 2's CPU is interrupted. Port 2's message status is readable from the Port 1 side via control register 2. When Port 2's CPU reads the data from its Message In register, the Port 2 IP is cleared. Thus, Port 1's CPU can tell that the message has been read and can now send another message or follow whatever protocol that is set up between the two CPU's. The same transfer can also be made from Port 2's CPU to Port 1's CPU.

CLEAR (Empty) FIFO Operation

The $\overline{\text{CLEAR}}$ FIFO bit (active LOW) clears the FIFO buffer of data. Writing a 0 to this bit empties the FIFO buffer, inactivates the $\overline{\text{REQUEST}}$ line, and disables the handshake (if programmed). The $\overline{\text{CLEAR}}$ bit does not affect any control or data register. To remove the $\overline{\text{CLEAR}}$ state, write a 1 to the $\overline{\text{CLEAR}}$ bit.

In CPU/CPU mode, under program control, only one of the ports can empty the FIFO by writing to its Control Register 3, bit 6. The Port 1 CPU must program bit 7 in Control Register 3 to determine which port controls the CLEAR FIFO operation (0 = Port 1 control; 1 = Port 2 control).

Direction of Data Transfer Operation

The Data Direction bit controls the direction of data transfer in the FIFO buffer. The Data Direction bit is defined as 0 = output from CPU and 1 = input to CPU. This bit reads correctly when read by either port's CPU. For example, if Port 1's CPU reads a 0 (CPU output) in its Data Direction bit, then Port 2's CPU reads a 1 (input to CPU) in its Data Direction bit.

In CPU/CPU mode, under program control, only one of the ports can control the direction of data transfer. The Port 1 CPU must program bit 5 in Control Register 3 to determine which port controls the data direction (0 = Port 1 control; 1 = Port 2 control). Figure 20 shows FIO data transfer options.

CPU TO I/O OPERATION

When Port 2 is programmed in the Interlocked 2-Wire Handshake mode or the 3-Wire Handshake mode, and Port A is programmed in Z-BUS or non-Z-BUS Microprocessor mode, the FIO interfaces a CPU and a peripheral device. In the Interlocked 2-Wire Handshake mode, RFD/$\overline{\text{DAV}}$ and $\overline{\text{ACKIN}}$ strobe data to and from Port 2. In the 3-Wire Handshake mode, RFD/$\overline{\text{DAV}}$, $\overline{\text{DAV}}$/DAC, and DAC/RFD signals control data flow.

Interlocked 2-Wire Handshake

In the Interlocked Handshake, the action of the FIO must be acknowledged by the other half of the handshake before the next action can take place. In output mode, Port 2 does not indicate that new data is available until the external device indicates it is ready for the data. Similarly, in input mode, Port 2 does not indicate that it is ready for new data until the data source indicates that the previous byte of the data is no longer available, thereby acknowledging Port 2's acceptance of the last byte. This allows the FIO to directly interface to a Z8's port, a CIO's port, a UPC's port, another FIO port, or another FIFO Z8060, with no external logic (Figures 21 and 22).

3-Wire Handshake

The 3-Wire Handshake is designed for applications in which one output port is communicating with many input ports simultaneously. It is essentially the same as the Interlocked Handshake, except that two signals are used to indicate that an input port is ready for new data or that it has accepted the present data. In the 3-Wire Handshake, the rising edge of the RFD status line indicates that the port is ready for data, and the rising edge of the DAC status line indicates tha the data has been accepted. With 3-Wire Handshake, the lines of many input ports can be bussed together with open-drain drivers and the output port knows when all of the ports are ready and have accepted the data. This handshake is the same handshake used in the IEEE-488 Instruments. Since the port's direction can be changed under software control, bidirectional IEEE-488-type transfers can be performed. Figures 23 and 24 show the timings associated with 3-Wire Handshake communications.

CLEAR FIFO Operation

In CPU-to-I/O operation, the CLEAR FIFO operation can be performed by the CPU side (Port 1) under software control as previously explained. The CLEAR FIFO operation can also be performed under hardware control by defining the $\overline{\text{CLEAR}}$ pin of Port 2 as an input (Control Register 3, bit 7 = 1).

For cascading purposes, the $\overline{\text{CLEAR}}$ pin can also be defined as an output (Control Register 3, bit 7 = 0), which reflects the current state of the $\overline{\text{CLEAR}}$ FIFO bit. It can then empty other FIOs or initialize other devices in the system.

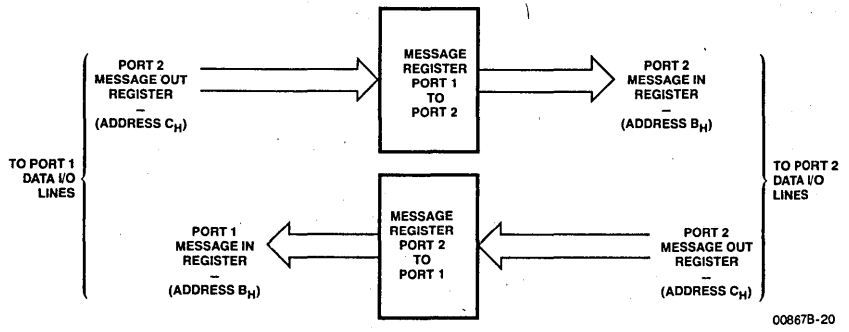


Figure 19. Message Register Operation

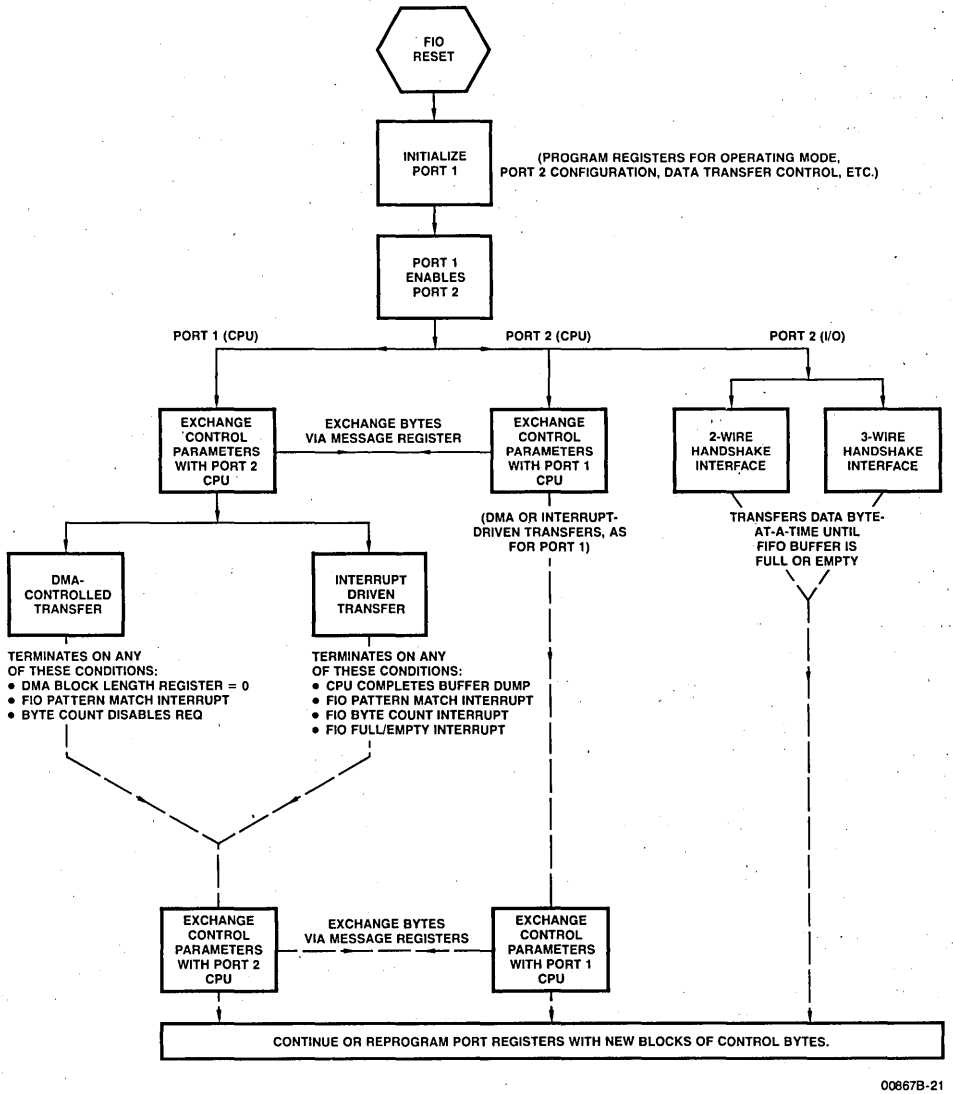


Figure 20. FIO Data Transfer Options

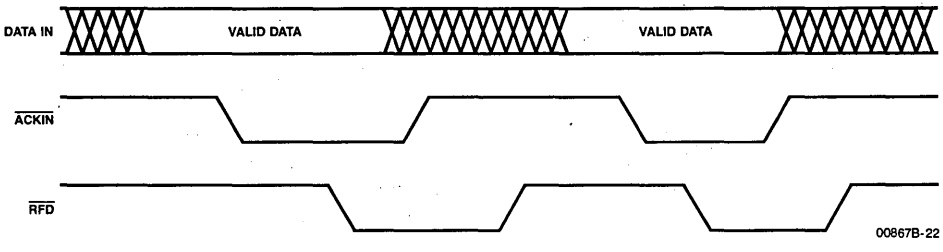


Figure 21. Input Timing for Interlocked Handshake Timing (Port 2 Side Only)

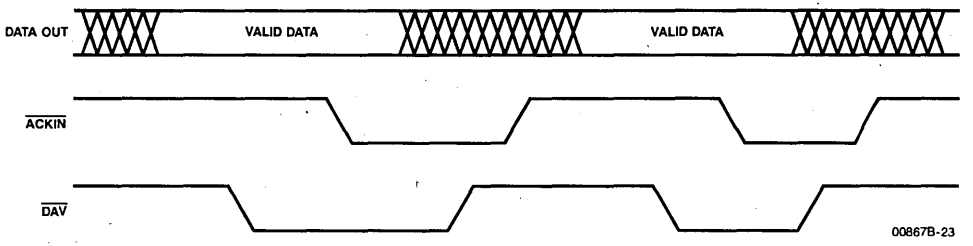


Figure 22. Output Timing for Interlocked Handshake Timing (Port 2 Side Only)

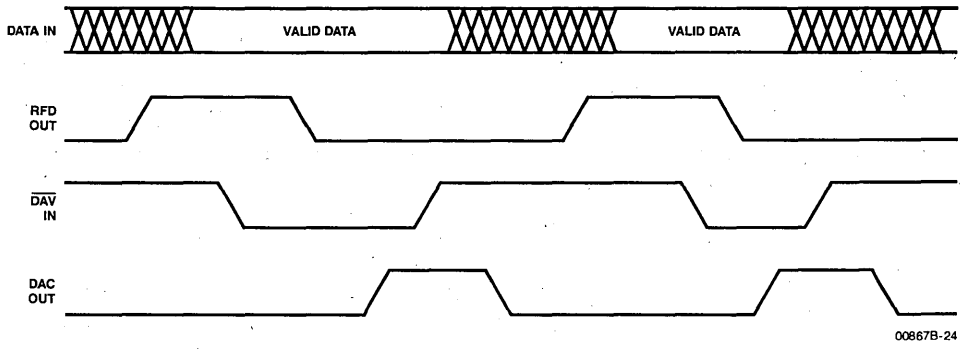


Figure 23. Input (Acceptor) Timing IEEE-488 HS (Port 2 Side Only)

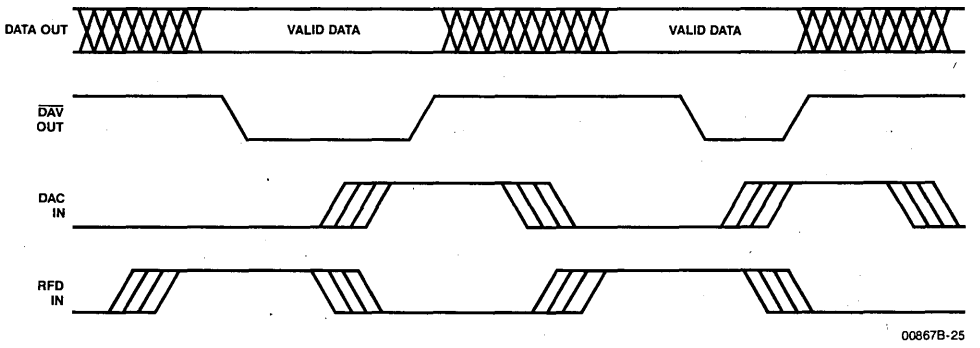


Figure 24. Output (Source) Timing IEEE-488 HS Port (Port 2 Side Only)

Data Direction Control

In CPU-to-I/O mode, the direction of data transfer can be controlled by the CPU side (Port 1) under software control as previously explained. The data direction can also be determined by hardware control by defining the Data Direction pin of Port 2 as an input (Control Register 3, bit 5 = 1).

For cascading purposes, the Data Direction pin can also be defined as an output (Control Register 3, bit 5 = 0) pin which reflects the current state of the Data Direction bit. It can then be used to control the direction of data transfer for other FIOs or for external logic.

On the Port 2 side, when data direction is 0, Port 2 is in Output Handshake mode. When data direction is 1, Port 2 is in Input Handshake mode.

PROGRAMMING

The Programming of the FIO is greatly simplified by the efficient grouping of the various operation modes in the control registers. Since all of the control registers are read/write, the need for maintaining their image in system memory is eliminated. Also, the read/write feature of the registers aids in system debugging.

Each side of the FIO has 16 registers. All 16 registers are used by the Port 1 side; Control register 2 is not used on the Port 2 side. All registers are addressable O_H through F_H.

In the Z-BUS Low Byte mode, the FIO allows two methods for register addressing under control of the Right Justify Address (RJA) bit in Control register 0. When RJA = 0, address bus bits 1-4 are used for register addressing and bits 0, 5, 6 and 7 are ignored (Table 4). When RJA = 1, bits 0-3 are used for the register addresses and bits 4-7 are ignored.

Control Registers

These four registers specify FIO operation. The Port 2 side control registers operate only if the Port 2 device is a CPU. The Port 2 CPU can control interface operations, including data direction, only when enabled by the setting of bit 0 in the Port 1 side of Control Register 2. A 1 in bit 1 of the same register enables the handshake logic.

Interrupt Status Registers

These four registers control and monitor the priority interrupt functions for the FIO.

Interrupt Vector Register

This register stores the interrupt service routine address. This vector is placed on D_0-D_7 when IUS is set by the Interrupt Acknowledge signal from the CPU. When bit 4 (Vector Includes Status) is set in Control Register 0, the reason for the interrupt is encoded within the vector address in bits 1, 2 and 3. If bit 5 is set in Control register 0, no vector is output by the FIO during an Interrupt Acknowledge cycle. However, IUS is set as usual.

Byte Count Compare Register

This register contains a value compared with the byte count in the Byte Count register. If the Byte Count Compare interrupt is enabled, an interrupt will occur upon compare.

Message Out Register

Either CPU can place a message in its Message Out register. If the opposite side Message register interrupt is enabled, the receiving side CPU will receive an interrupt request, advising that a message is present in its Message In register. Bit 5 in Control Register 1 on the initiating side is set when a message is written. It is cleared when the Byte Count register read is completed.

TABLE 4. FIO REGISTER ADDRESS SUMMARY

Non-Z-Bus		D_7-D_4	D_3	D_2	D_1	D_0	
Z-BUS High Byte			A_3	A_2	A_1	A_0	
Z-BUS Low Byte	RJA=0	AD_7-AD_5	AD_4	AD_3	AD_2	AD_1	AD_0
	RJA=1	AD_7-AD_4	AD_3	AD_2	AD_1	AD_0	
Description							
Control Register 0		X	0	0	0	0	X
Control Register 1		X	0	0	0	1	X
Interrupt Status Register 0		X	0	0	1	0	X
Interrupt Status Register 1		X	0	0	1	1	X
Interrupt Status Register 2		X	0	1	0	0	X
Interrupt Status Register 3		X	0	1	0	1	X
Interrupt Vector Register		X	0	1	1	0	X
Byte Count Register		X	0	1	1	1	X
Byte Count Comparison Register		X	1	0	0	0	X
Control Register 2*		X	1	0	0	1	X
Control Register 3		X	1	0	1	0	X
Message Out Register		X	1	0	1	1	X
Message In Register		X	1	1	0	0	X
Pattern Match Register		X	1	1	0	1	X
Pattern Mask Register		X	1	1	1	0	X
Data Buffer Register		X	1	1	1	1	X

X = Don't Care
*Register is only on Port 1 side

Message In Register

This register receives a message placed in the Message Out register by the opposite side CPU.

Pattern Match Register

This register contains a bit pattern matched against the byte in the Data Buffer register.When these patterns match, a Pattern Match interrupt will be generated, if previously enabled.

Pattern Mask Register

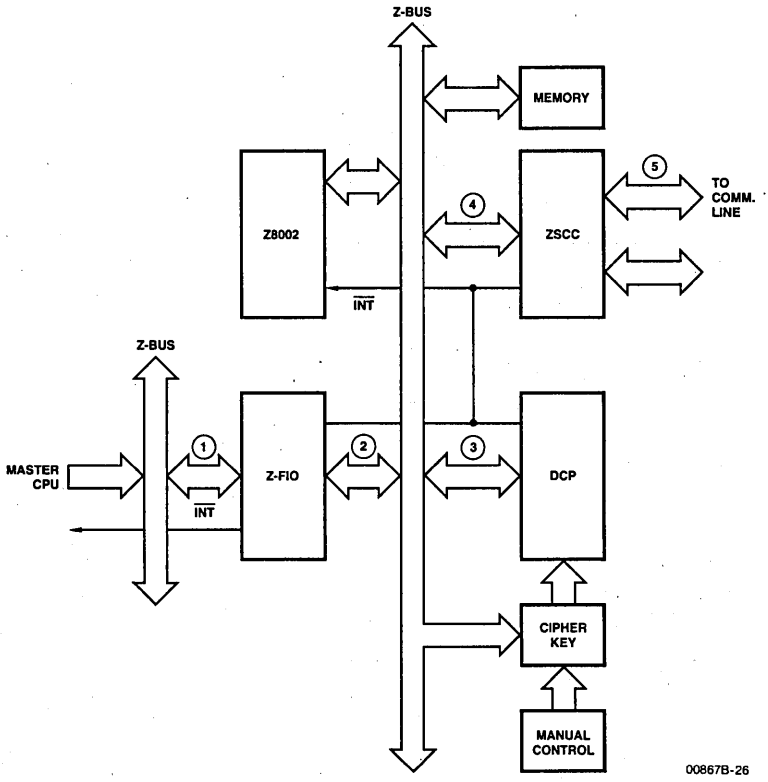
The Pattern Mask register may be programmed with a bit pattern mask that limits comparable bits in the Pattern Match register to non-masked bits (1 = mask).

Data Buffer Register

This register contains the data to be read from or written to the FIFO buffer.

Byte Count Register

This is a read-only register, containing the byte count for the FIFO buffer. The byte count is derived by subtracting the number of bytes read from the buffer from the number of bytes written into the buffer. The count is copied into a holding register for an accurate reading by setting bit 6 (Freeze Status register) in Control Register 1. This bit is cleared when the Byte Count register is completed.



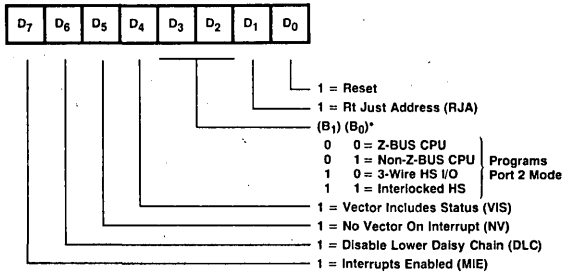
DATA FLOW: 1. Data from master CPU → Z-FIO Port 2.
2. Z-FIO Port 1 → DCP.
3. DCP → RAM.
4. RAM → Z-SCC.
5. Z-SCC → data comm. line loop.

Figure 25. Typical Application: Node Controller

REGISTERS

Control Register 0

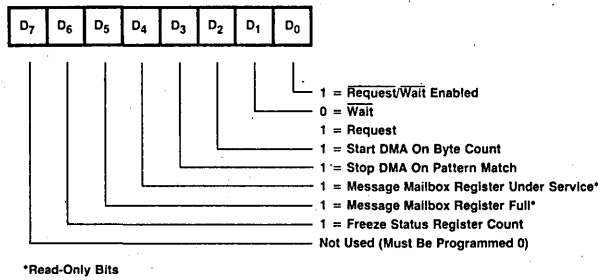
Address: 0000
(Read/Write)



*Read Only From Port 2 Side

Control Register 1

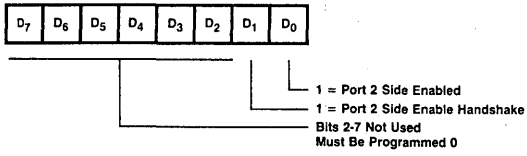
Address: 0001
(Read/Write)



*Read-Only Bits

Control Register 2*

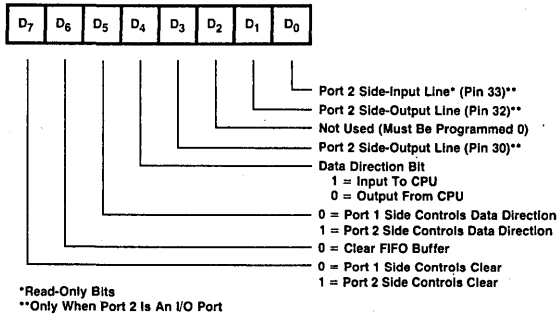
Address: 1001
(Read/Write)



*This Register Reads All 0's From Port 2 Side

Control Register 3

Address: 1010
(Read/Write)



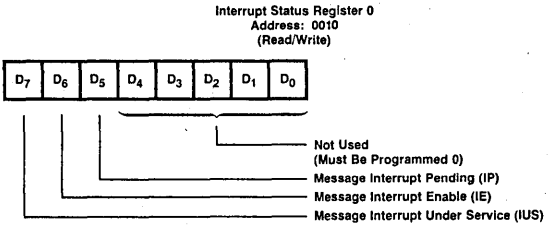
*Read-Only Bits
**Only When Port 2 Is An I/O Port

Figure 26. Control Registers

REGISTERS (Cont.)

Interrupt Status Register 0

Address: 0010
(Read/Write)

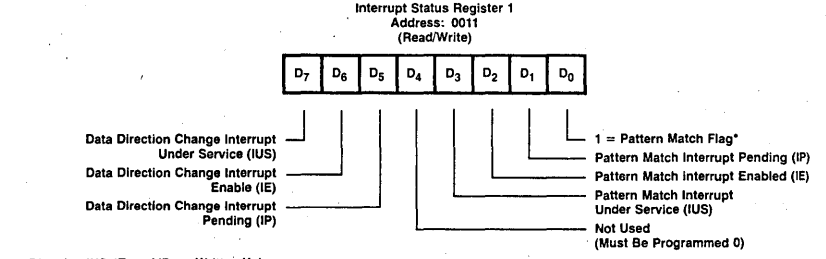


Message IUS, IE, and IP are Written Using The Following Commands ($D_4 - D_0 = 0$)

D_7	D_6	D_5	
0	0	0	Null Code (No Change)
0	0	1	Clear IP & IUS
0	1	0	Set IUS
0	1	1	Clear IUS
1	0	0	Set IP
1	0	1	Clear IP
1	1	0	Set IE
1	1	1	Clear IE

Interrupt Status Register 1

Address: 0011
(Read/Write)



Direction IUS, IE, and IP are Written Using The Following Commands: (D_0 and $D_4 = 0$)

	D_7	D_6	D_5
Null Code (No Change)	0	0	0
Clear IP & IUS	0	0	1
Set IUS	0	1	0
Clear IUS	0	1	1
Set IP	1	0	0
Clear IP	1	0	1
Set IE	1	1	0
Clear IE	1	1	1

Pattern Match IUS, IE, and IP are Written Using The Following Commands: (D_0 and $D_4 = 0$)

D_3	D_2	D_1	
0	0	0	Null Code (No Change)
0	0	1	Clear IP & IUS
0	1	0	Set IUS
0	1	1	Clear IUS
1	0	0	Set IP
1	0	1	Clear IP
1	1	0	Set IE
1	1	1	Clear IE

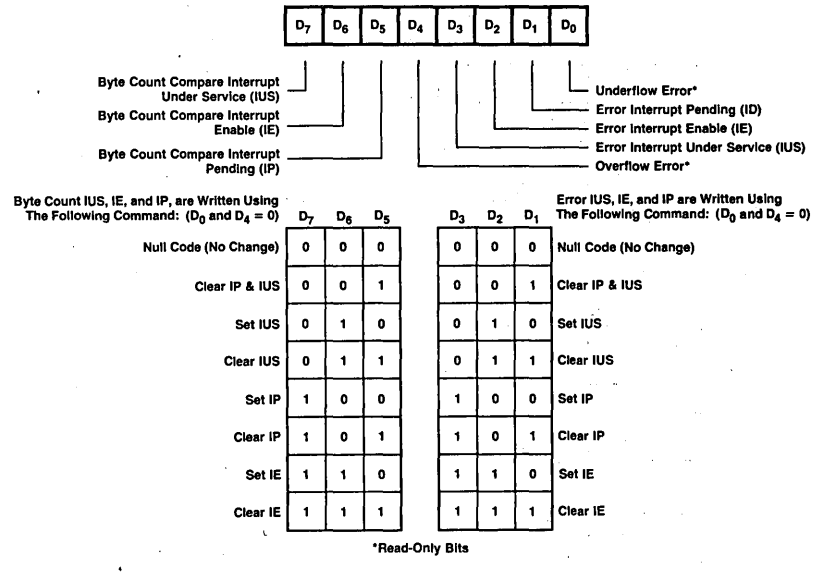
*Read only bits

Figure 27. Interrupt Status Registers

REGISTERS (Cont.)

Interupt Status Register 2

Address: 0100
(Read/Write)



Interrupt Status Register 3

Address: 0101
(Read/Write)

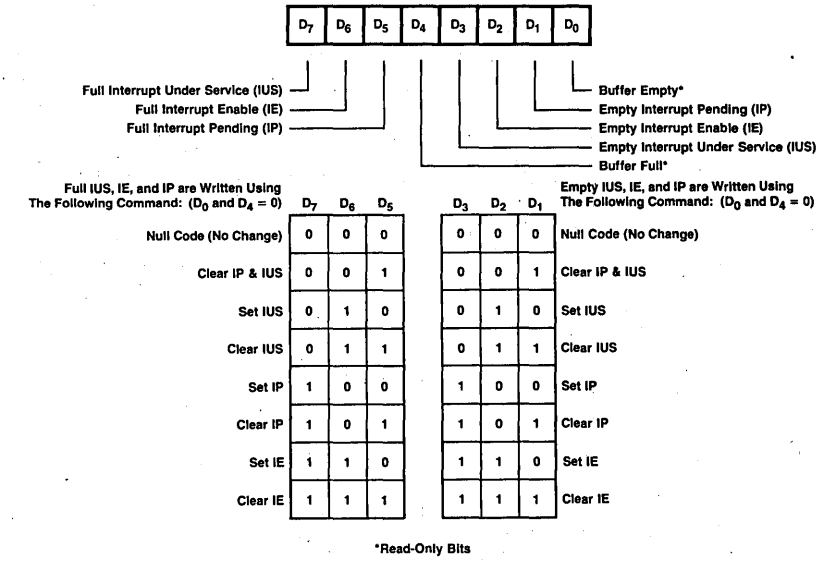
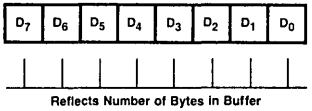


Figure 27. Interrupt Status Registers (Cont.)

REGISTERS (Cont.)

Byte Count Register

Address: 0111



(Copied from actual Byte Counter by setting bit 6 of CR1.)

Figure 28. Byte Count Register

Interrupt Vector Register

Address: 0110
(Read/Write)

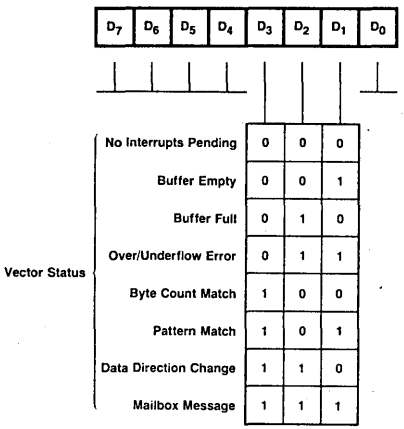


Figure 29. Interrupt Vector Register

Pattern Match Register

Address: 1101
(Read/Write)

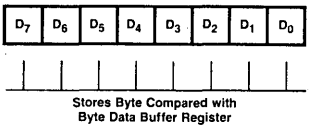


Figure 30. Pattern Match Register

Pattern Mask Register

Address: 1110
(Read/Write)

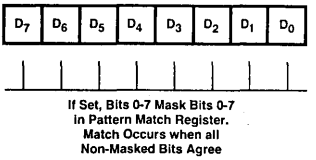


Figure 31. Pattern Mask Register

Data Buffer Register

Address: 1111
(Read/Write)

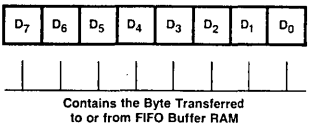


Figure 32. Data Buffer Register

Byte Count Comparison Register

Address: 1000
(Read/Write)

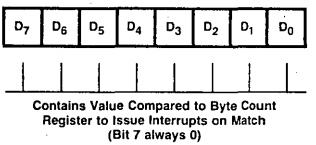


Figure 33. Byte Count Comparison Register

Message Out Register

Address: 1011
(Read/Write)

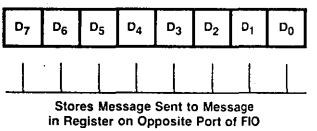


Figure 34. Message Out Register

Message In Register

Address: 1100
(Read Only)

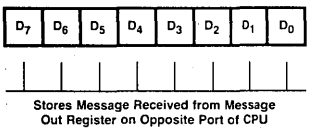


Figure 35. Message In Register

MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65 to +150°C
Voltage at any Pin Relative to V_{SS}	−0.5V to +7.0V
Power Dissipation	1.75W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

(over which the DC, switching and functional specification apply)

	4MHz	6MHz
	Z8038	Z8038A
Commercial Operating Range T_A = 0 to +70°C V_{CC} = 5V ± 5%	Z8038DC Z8038PC	Z8038ADC Z8038APC
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ± 5%	Z8038DI	
Military Operating Range T_A = −55 to +125°C V_{CC} = 5V ± 10%	Z8038DMB	

Notes: T_A denotes ambient temperature.
Add suffix B to indicate burn-in requirement.

ELECTRICAL CHARACTERISTICS over operating range unless otherwise specified (Note 1)

Parameter	Description	Test Conditions		Min	Typ	Max	Units
V_{IL}	Input LOW Voltage			−0.5		+.8	Volts
V_{IH}	Input HIGH Voltage	Standard Temp		2.0		V_{CC}	Volts
		Military Temp		2.4			
V_{OL}	Output LOW Voltage	I_{OL} = 3.2mA				0.5	Volts
		I_{OL} = 2.0mA				0.4	
V_{OH}	Output HIGH Voltage	I_{OH} = −250μA		2.4			Volts
I_{OZL}	Output Leakage Current	V_{OUT} = 0.4V				10	μA
I_{OZH}	Output Leakage Current	V_{OUT} = V_{CC}				10	μA
I_I	Input Leakage Current					±10	μA
C_{IN}	Input Capacitance	Unmeasured pins returned to ground. f = 1MHz over specified temperature range.				10	pF
$C_{I/O}$	I/O Capacitance					20	pF
C_{OUT}	Output Capacitance					15	pF
I_{CC}	Power Supply Current	V_{CC} = MAX	T_A = 0°C			200	mA
			T_A = −55°C			250	

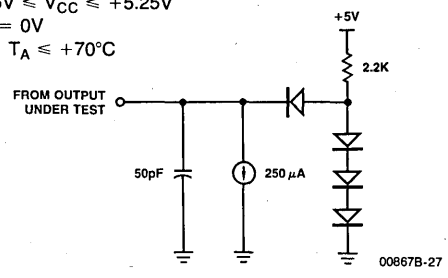
Note 1. See table for operating range. Typical conditions apply at T_A = 25°C, V_{CC} = 5.0V.

Standard Test Conditions

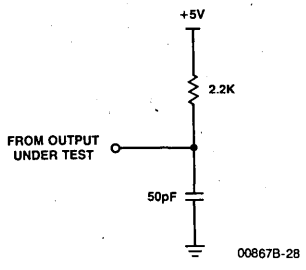
The characteristics below apply for the following standard test conditions, unless otherwise noted. All voltages are referenced to GND. Positive current flows into the referenced pin. Standard conditions are as follows:

+4.75V ≤ V_{CC} ≤ +5.25V
GND = 0V
0°C ≤ T_A ≤ +70°C

Standard Test Load



Open-Drain Test Load

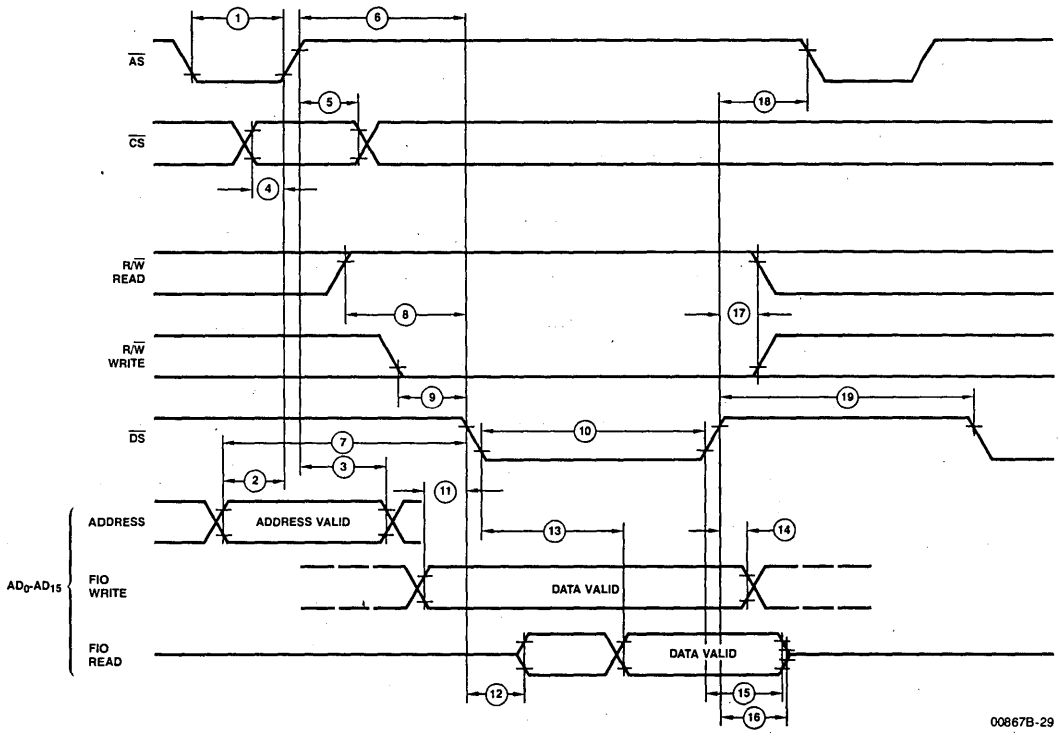


Z-BUS CPU INTERFACE TIMING

Number	Symbol	Parameter	4MHz Min	4MHz Max	6MHz Min	6MHz Max	Units	Notes*
1	TwAS	$\overline{AS}$ Low Width	70		50		ns	
2	TsA(AS)	Address to $\overline{AS}$ ↑ Setup Time	30		10		ns	1
3	ThA(AS)	Address to $\overline{AS}$ ↑ Hold Time	50		30		ns	1
4	TsCSO(AS)	$\overline{CS}$ to $\overline{AS}$ ↑ Setup Time	0		0		ns	1
5	ThCSO(AS)	$\overline{CS}$ to $\overline{AS}$ ↑ Hold Time	60		40		ns	1
6	TdAS(DS)	$\overline{AS}$ ↑ to $\overline{DS}$ ↑ Delay	60		40		ns	1
7	TsA(DS)	Address to $\overline{DS}$ ↓	120		100		ns	
8	TsRWR(DS)	R/$\overline{W}$ (Read) to $\overline{DS}$ ↓ Setup Time	100		80		ns	
9	TsRWW(DS)	R/$\overline{W}$ (Write) to $\overline{DS}$ ↓ Setup Time	0		0		ns	
10	TwDS	$\overline{DS}$ Low Width	390		250		ns	
11	TsDW(DSf)	Write Data to $\overline{DS}$ ↓ Setup Time	30		20		ns	
12	TdDS(DRV)	$\overline{DS}$ (Read) ↓ to Address Data Bus Driven	0		0		ns	
13	TdDSf(DR)	$\overline{DS}$ ↓ to Read Data Valid Delay		250		180	ns	
14	ThDW(DS)	Write Data to $\overline{DS}$ ↑ Hold Time	30		20		ns	
15	TdDSr(DR)	$\overline{DS}$ ↑ to Read Data Not Valid Delay	0		0		ns	
16	TdDS(DRz)	$\overline{DS}$ ↑ to Read Data Float Delay		70		45	ns	2
17	ThRW(DS)	R/$\overline{W}$ to $\overline{DS}$ ↑ Hold Time	55		40		ns	
18	TdDS(AS)	$\overline{DS}$ ↑ to $\overline{AS}$ ↓ Delay	50		25		ns	
19	Trc	Valid Access Recovery Time	1000		650		ns	3

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float delay is measured up to the time when the output has changed 0.5V from steady state with minimum AC load and maximum DC load.
3. This is the delay from $\overline{DS}$ of one CIO access to $\overline{DS}$ of another FIO access (either read or write).
*All timings are preliminary and subject to change. All timing references assume 2.0V for a logic "1" and 0.8V for a logic "0."

Figure 36. Z-Bus CPU Interface Timing

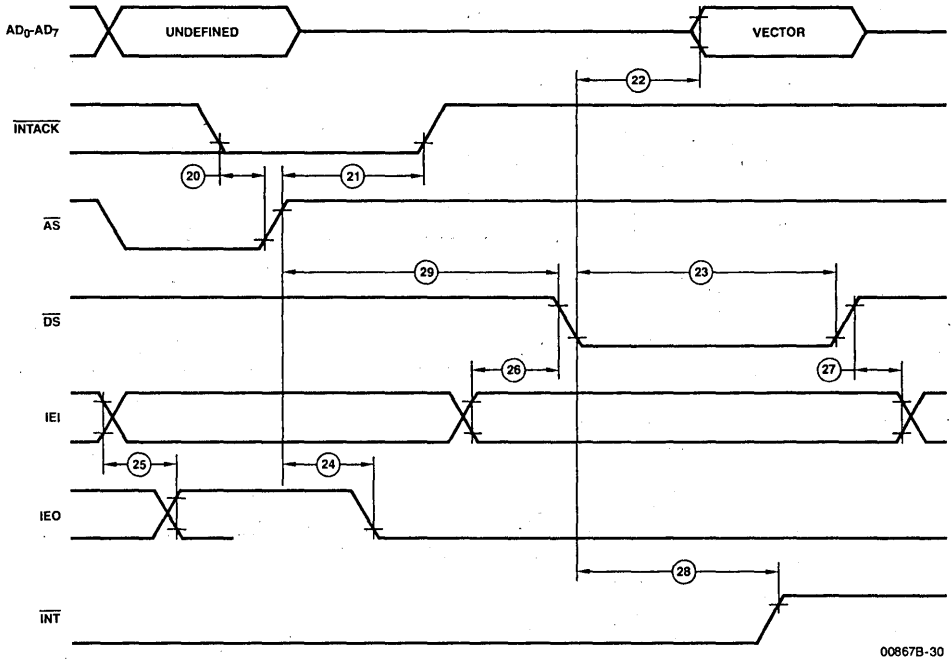


Z-BUS CPU INTERRUPT ACKNOWLEDGE TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
20	TsIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Setup Time	0		0		ns	
21	ThIA(AS)	$\overline{INTACK}$ to $\overline{AS}$ ↑ Hold Time	250		250		ns	
22	TdDSA(DR)	$\overline{DS}$ (Acknowledge) ↓ to Read Data Valid Delay		250		180	ns	
23	TwDSA	$\overline{DS}$ (Acknowledge) Low Width	390		250		ns	
24	TdAS(IEO)	$\overline{AS}$ ↑ to IEO ↓ Delay ($\overline{INTACK}$ Cycle)		350		250	ns	4
25	TdIEI(IEO)	IEI to IEO Delay		150		100	ns	4
26	TsIEI(DSA)	IEI to $\overline{DS}$ (Acknowledge) ↓ Setup Time	100		70		ns	
27	ThIEI(DSA)	IEI to $\overline{DS}$ (Acknowledge) ↑ Hold Time	50		30		ns	4
28	TdDS(INT)	$\overline{DS}$ ($\overline{INTACK}$ Cycle) to $\overline{INT}$ Delay		900		800	ns	
29	TdDCST	Interrupt Daisy Chain Settle Time					ns	4

Notes: 4. The parameters for the devices in any particular daisy chain must meet the following constraint: The delay from $\overline{AS}$ to $\overline{DS}$ must be greater than the sum of TdAS(IEO) for the highest priority peripheral, TsIEI(DSA) for the lowest priority peripheral, and TdIEI(IEO) for each peripheral separating them in the chain.

Figure 37. Z-Bus CPU Interrupt Acknowledge Timing

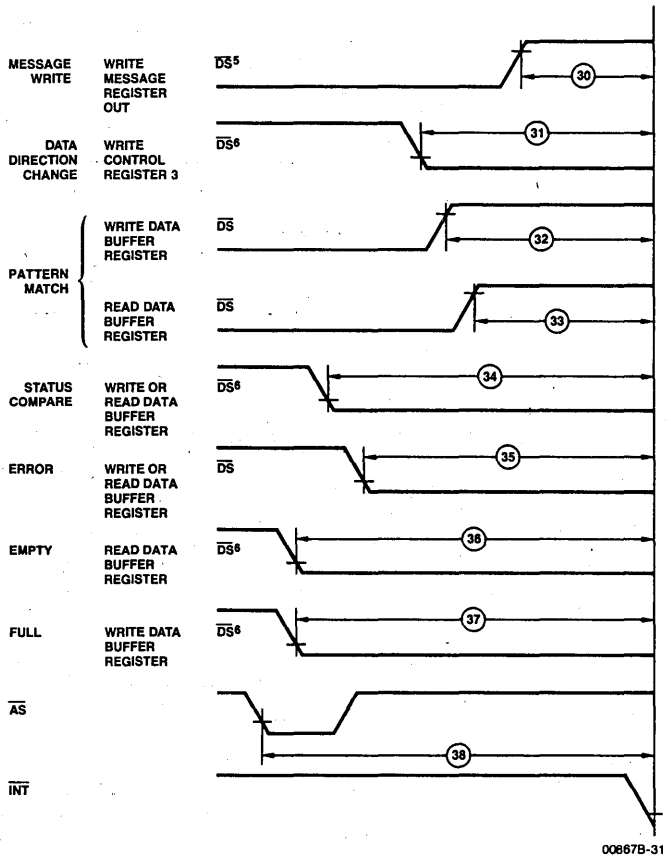


Z-BUS INTERRUPT TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
30	TdMW(INT)	Message Write to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycles +ns	5
31	TdDC(INT)	Data Direction Change to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycles +ns	6
32	TdPMW(INT)	Pattern Match to $\overline{INT}$ Delay (Write Case)		1		1	$\overline{AS}$ Cycles +ns	
33	TdPMR(INT)	Pattern Match (Read Case) to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycle +ns	
34	TdSC(INT)	Status Compare to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycles +ns	6
35	TdER(INT)	Error to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycles +ns	
36	TdEM(INT)	Empty to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycles +ns	6
37	TdFL(INT)	Full to $\overline{INT}$ Delay		1		1	$\overline{AS}$ Cycles +ns	6
38	TdAS(INT)	$\overline{AS}$ to $\overline{INT}$ Delay					$\overline{AS}$ Cycles +ns	

Notes: 5. Write is from the other side of FIO.
6. Write can be from either side, depending on programming of FIO.

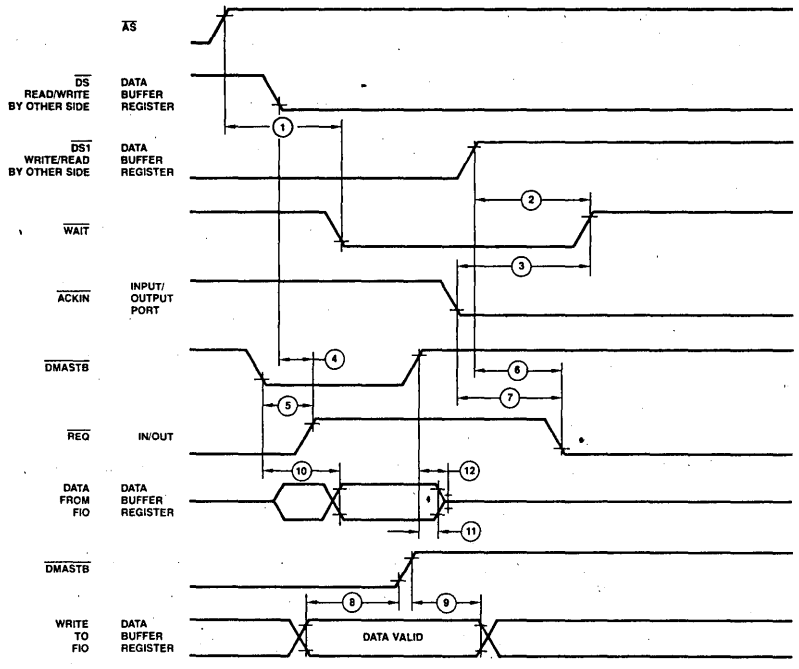
Figure 38. Z-Bus Interrupt Timing



Z-BUS REQUEST/$\overline{WAIT}$ TIMING AND WAVEFORMS (Figure 39)

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
1	TdDS(WAIT)	$\overline{AS}$ ↑ to $\overline{WAIT}$ ↓ Delay		190		160	ns	
2	TdDS1(WAIT)	$\overline{DS1}$ ↑ to $\overline{WAIT}$ ↑ Delay		1000		1000	ns	
3	TdACK(WAIT)	$\overline{ACKIN}$ ↓ to $\overline{WAIT}$ ↑ Delay		1000		1000	ns	1
4	TdDS(REQ)	$\overline{DS}$ ↓ to $\overline{REQ}$ ↑ Delay		350		300	ns	
5	TdDMA(REQ)	$\overline{DMASTB}$ ↓ to $\overline{REQ}$ ↑ Delay		350		300	ns	
6	TdDS1(REQ)	$\overline{DS1}$ ↑ to $\overline{REQ}$ ↓ Delay		1000		1000	ns	
7	TdACK(REQ)	$\overline{ACKIN}$ ↓ to $\overline{REQ}$ ↓ Delay		1000		1000	ns	
8	TdSU(DMA)	Data Setup Time to $\overline{DMASTB}$	200		150		ns	
9	TdH(DMA)	Data Hold Time to $\overline{DMASTB}$	30		20		ns	
10	TdDMA(DR)	$\overline{DMASTB}$ ↓ Data Valid		150		100	ns	
11	TdDMA(DRH)	$\overline{DMASTB}$ ↑ to Data Not Valid	0		0		ns	
12	TdDMA(DR2)	$\overline{DMASTB}$ ↑ to Data Bus Float		70		45	ns	

Notes: 1. The Delay is from $\overline{DAV}$ ↓ for 3-Wire Input Handshake. The delay is from DAC ↑ for 3-Wire Output Handshake.

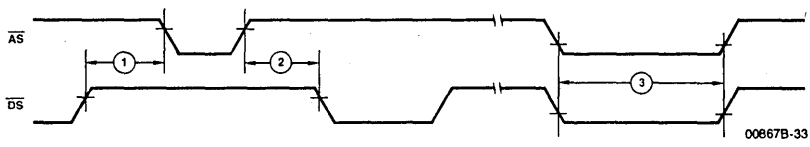


Z-BUS RESET TIMING AND WAVEFORM (Figure 40)

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
1	TdDSQ(AS)	Delay from $\overline{DS}$ ↑ to $\overline{AS}$ ↓ for No Reset	0		20		ns	
2	TdASQ(DS)	Delay for $\overline{AS}$ ↑ to $\overline{DS}$ ↓ for No Reset	50		30		ns	
3	Tw(AS + DS)	Minimum Width of $\overline{AS}$ and $\overline{DS}$ both Low for Reset	500		350		ns	1

PRELIMINARY

Notes: 1. Internal circuitry allows for the reset provided by the Z8 ($\overline{DS}$ held Low while $\overline{AS}$ pulses) to be sufficient.



NON-Z-BUS CPU INTERFACE TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
1	TsA(RD)	Address Setup to $\overline{RD}\downarrow$	80		80		ns	1
2	TsA(WR)	Address Setup to $\overline{WR}\downarrow$	80		80		ns	
3	ThA(RD)	Address Hold Time to $\overline{RD}\uparrow$	0		0		ns	1
4	ThA(WR)	Address Hold Time to $\overline{WR}\uparrow$	0		0		ns	
5	TsCEl(RD)	$\overline{CE}$ Low Setup Time to $\overline{RD}$	0		0		ns	1
6	TsCEl(WR)	$\overline{CE}$ Low Setup Time to $\overline{WR}$	0		0		ns	
7	ThCEl(RD)	$\overline{CE}$ Low Hold Time to $\overline{RD}$	0		0		ns	1
8	ThCEl(WR)	$\overline{CE}$ Low Hold Time to $\overline{WR}$	0		0		ns	
9	TsCEh(RD)	$\overline{CE}$ High Setup Time to $\overline{RD}$	100		70		ns	1
10	TsCEh(WR)	$\overline{CE}$ High Setup Time to $\overline{WR}$	100		70		ns	
11	TwRD1	$\overline{RD}$ Low Width	390		250		ns	
12	TdRD(DRA)	$\overline{RD}\downarrow$ to Read Data Active Delay	0		0		ns	
13	TdRDf(DR)	$\overline{RD}\downarrow$ to Valid Data Delay		250		180	ns	
14	TdRDr(DR)	$\overline{RD}\uparrow$ to Read Data Not Valid Delay	0		0		ns	
15	TdRD(DRz)	$\overline{RD}\uparrow$ to Data Bus Float		70		45	ns	2
16	TwWR1	$\overline{WR}$ Low Width	390		250		ns	
17	TsDW(WR)	Data Setup Time to $\overline{WR}$	0		0		ns	
18	ThDW(WR)	Data Hold Time to $\overline{WR}$	30		20		ns	
19	Trc	Valid Access Recovery Time	1000		650		ns	3

Notes: 1. Parameter does not apply to Interrupt Acknowledge transactions.
2. Float delay is measured to the time the output has changed 0.5V from steady state with minimum ac load and maximum dc load.
3. This is the delay from $\overline{RD}\uparrow$ or $\overline{WR}\uparrow$ of one FIO access to $\overline{RD}\downarrow$ or $\overline{WR}\downarrow$ of another FIO access.

Figure 41a. Non-Z-Bus CPU Interface Timing

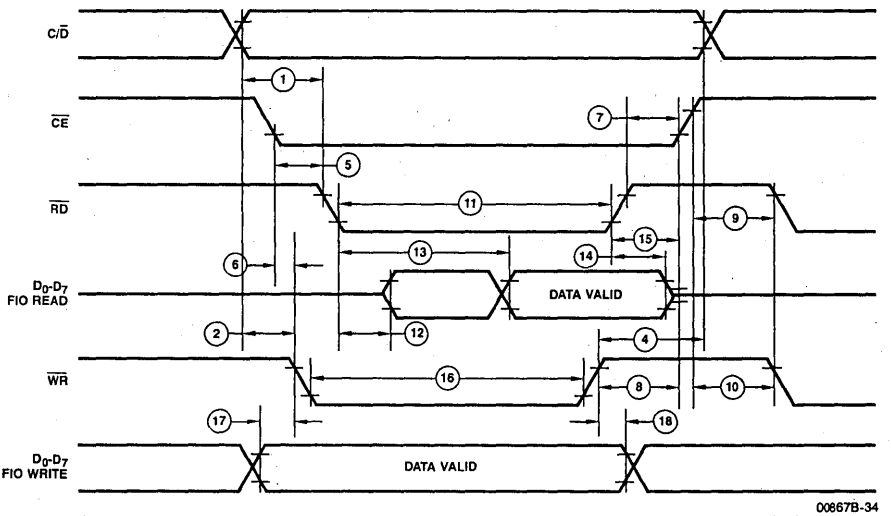
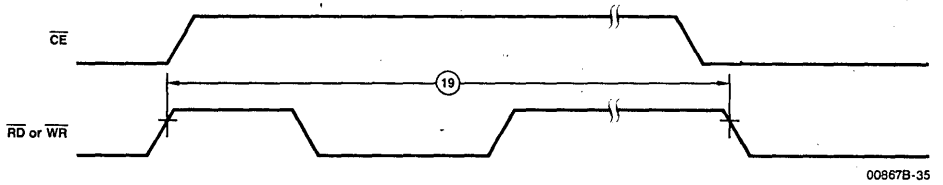


Figure 41b. Non-Z-Bus Interface Timing

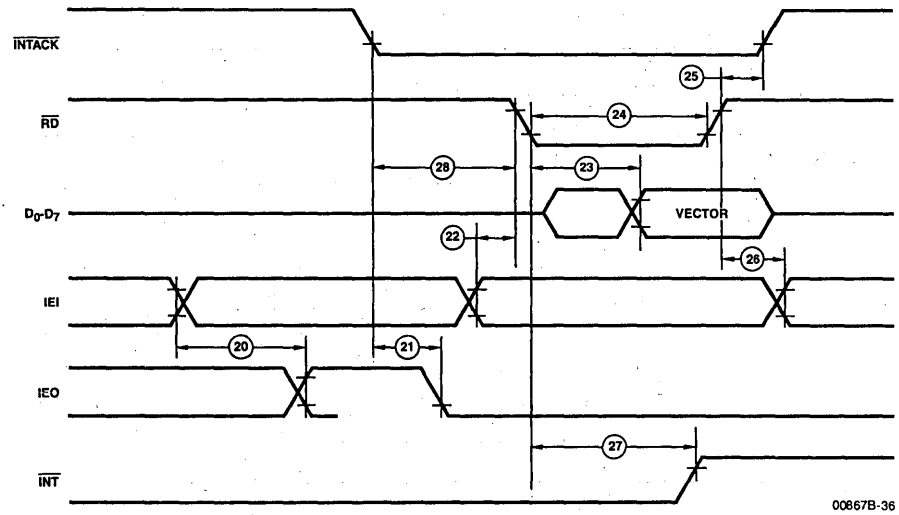


NON-Z-BUS INTERRUPT ACKNOWLEDGE TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
20	TdIEI(IEO)	IEI to IEO Delay		150		100	ns	4
21	TdI(IEO)	$\overline{\text{INTACK}}$ ↓ to IEO ↓ Delay		350		250	ns	4
22	TsIEI(RDA)	IEI Setup Time to $\overline{\text{RD}}$ (Acknowledge)	100		70		ns	4
23	TdRD(DR)	$\overline{\text{RD}}$ ↓ to Vector Valid Delay		250		180	ns	
24	TwRD1(IA)	Read Low Width (Interrupt Acknowledge)	390		250		ns	
25	ThIA(RD)	$\overline{\text{INTACK}}$ ↑ to $\overline{\text{RD}}$ ↑ Hold Time	30		20		ns	
26	ThIEI(RD)	IEI Hold Time to $\overline{\text{RD}}$ ↑	20		10		ns	
27	TdRD(INT)	$\overline{\text{RD}}$ ↑ to $\overline{\text{INT}}$ ↑ Delay		900		800	ns	
28	TdDCST	Interrupt Daisy Chain Settle Time		350		250	ns	4

Notes: 4. The parameter for the devices in any particular daisy chain must meet the following constraint: The delay from $\overline{\text{INTACK}}$ ↓ to $\overline{\text{RD}}$ ↓ must be greater than the sum of TdINA(IEO) for the highest priority peripheral, TsIEI(RD) for the lowest priority peripheral and TdIEI(IEO) for each peripheral separating them in the chain.

Figure 42. Non-Z-Bus Interrupt Acknowledge Timing

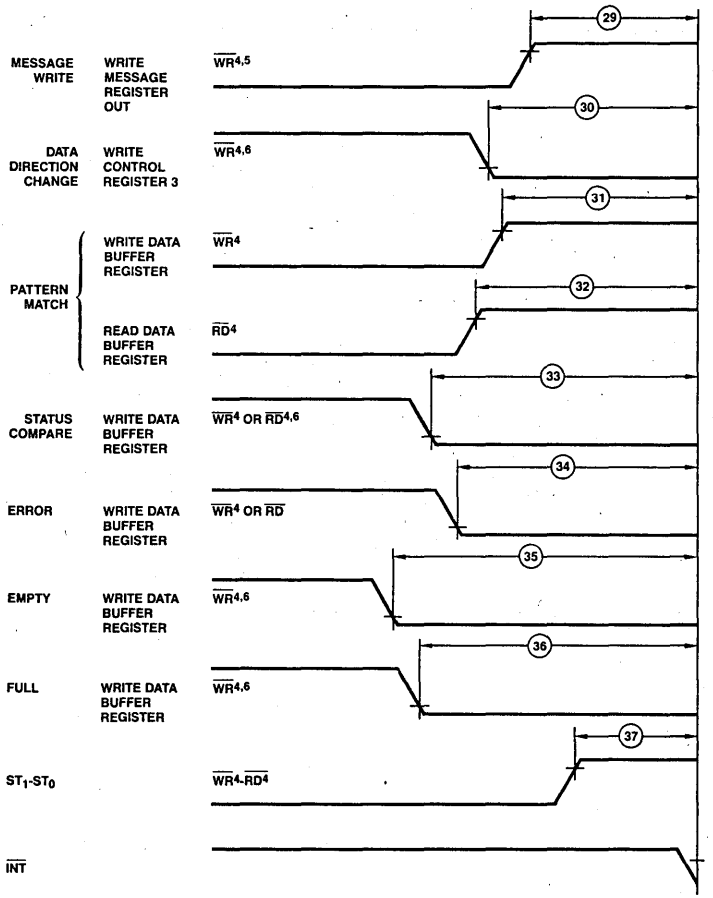


NON-Z-BUS INTERRUPT TIMING

Number	Symbol	Parameter	4MHz Min	4MHz Max	6MHz Min	6MHz Max	Units	Notes
29	TdMW(INT)	Message Write to $\overline{INT}$ Delay					ns	5,6
30	TdDC(INT)	Data Direction Change to $\overline{INT}$ Delay					ns	5,7
31	TdPMW(INT)	Pattern Match (Write Case) to $\overline{INT}$ Delay					ns	5
32	TdPMR(INT)	Pattern Match (Read Case) to $\overline{INT}$ Delay					ns	5
33	TdSC(INT)	Status Compare to $\overline{INT}$ Delay					ns	5,7
34	TdER(INT)	Error to $\overline{INT}$ Delay					ns	5,7
35	TdEM(INT)	Empty to $\overline{INT}$ Delay					ns	5,7
36	TdFL(INT)	Full to $\overline{INT}$ Delay					ns	5,7
37	TdS0(INT)	State 0 to $\overline{INT}$ Delay					ns	

Notes: 5. Delay number is valid for State 0 only.
6. Write is from other side of FIO.
7. Write can be from either side, depending on programming of FIO.

Figure 43. Non-Z-Bus Interrupt Timing



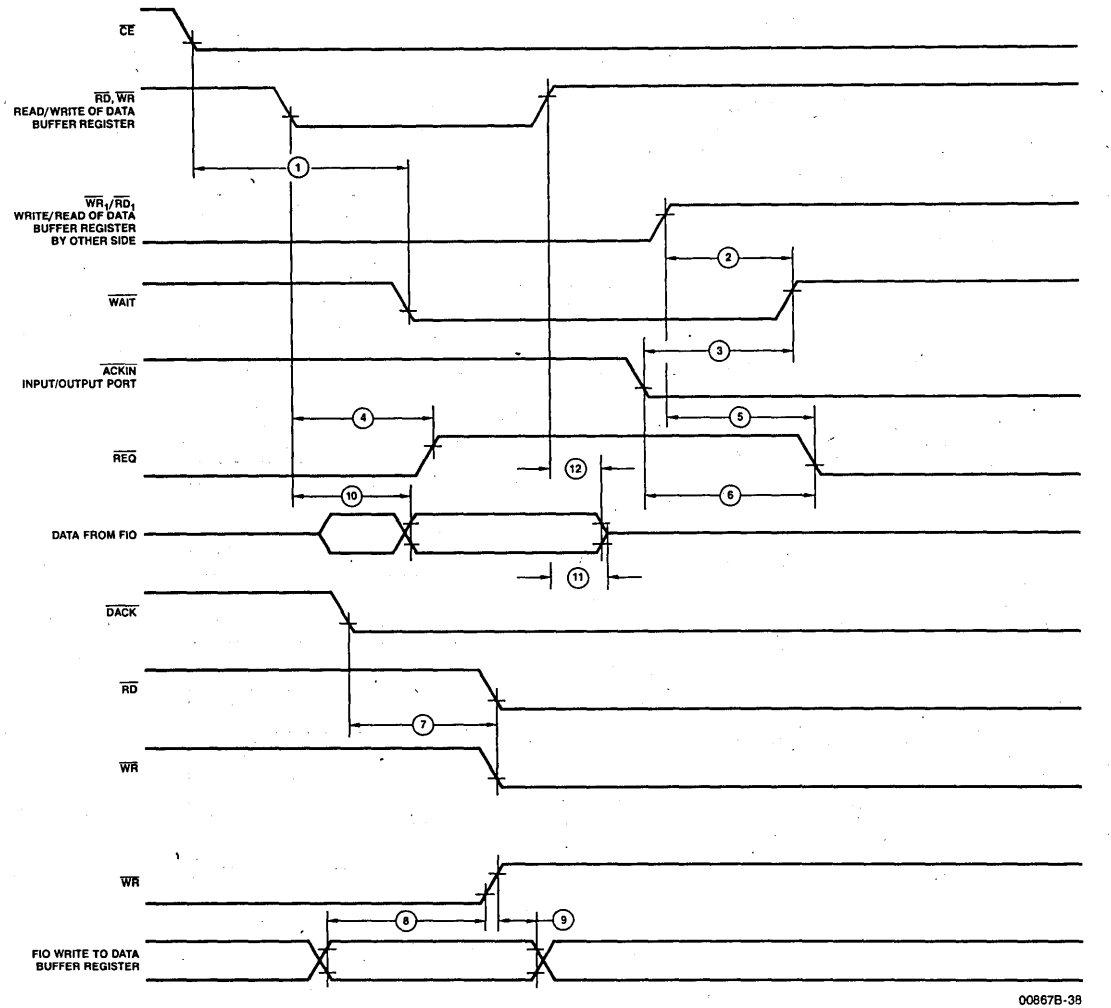
00867B-37

NON-Z-BUS REQUEST/WAIT TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
1	TdRD(WT)	$\overline{CE}$ ↓ to $\overline{WAIT}$ Active		200		170	ns	
2	TdRD1(WT)	$\overline{RD1}$ ↓ to $\overline{WAIT}$ Inactive		1000		1000	ns	
3	TdACK(WT)	$\overline{ACKIN}$ ↓ to $\overline{WAIT}$ Inactive		1000		1000	ns	1
4	TdRD(REQ)	$\overline{RD}$ ↓ to $\overline{REQ}$ Inactive		350		300	ns	
5	TdRD1(REQ)	$\overline{RD1}$ ↑ to $\overline{REQ}$ Active		1000		1000	ns	
6	TdACK(REQ)	$\overline{ACKIN}$ ↓ to $\overline{REQ}$ Active		1000		1000	ns	
7	TdDAC(RD)	$\overline{DACK}$ ↓ to $\overline{RD}$ ↓ or $\overline{WR}$ ↓	100		80		ns	
8	TSU(WR)	Data Setup Time to $\overline{WR}$	200				ns	
9	Th(WR)	Data Hold Time to $\overline{WR}$	30		20		ns	
10	TdDMA	$\overline{RD}$ ↓ to Valid Data		150		100	ns	2
11	TdDMA(DRH)	$\overline{RD}$ ↑ to Data Not Valid	0		0		ns	2
12	TdDMA(DRZ)	$\overline{RD}$ ↑ to Data Bus Float		70		45	ns	2

Notes: 1. The delay is from $\overline{DAV}$ ↓ for 3-Wire Input Handshake. The delay is from DAC ↑ for 3-Wire Input Handshake.
2. Only when $\overline{DACK}$ is active.

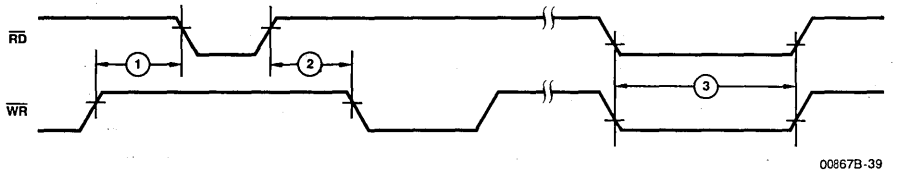
Figure 44. Non-Z-Bus Request/Wait Timing



NON-Z-BUS RESET TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
1	TdWR(RD)	Delay from $\overline{WR}\uparrow$ to $\overline{RD}\downarrow$	100		70		ns	
2	TdRD(WR)	Delay from $\overline{RD}\uparrow$ to $\overline{WR}\downarrow$	100		70		ns	
3	TwRD + WR	Width of $\overline{RD}$ and $\overline{WR}$, both Low for Reset	500		350		ns	

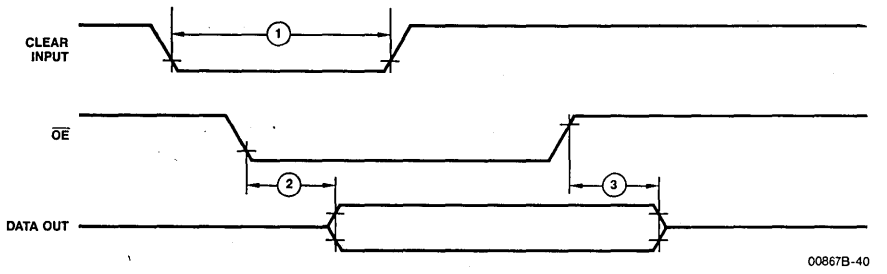
Figure 45. Non-Z-Bus Reset Timing



PORT 2 SIDE OPERATION

Number	Symbol	Parameter	4MHz		6MHz		Units	Notes
			Min	Max	Min	Max		
1	TwCLR	Width of Clear to Reset FIFO	700		700		ns	
2	TdOE(DO)	$\overline{OE}\downarrow$ to Data Bus Driven	0	150	0	150	ns	
3	TdOE(DRZ)	$\overline{OE}\uparrow$ to Data Bus Float		100		100	ns	

Figure 46. Port 2 Side Operation



FIO 2-WIRE HANDSHAKE TIMING

Number	Symbol	Parameter	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TsDI(ACK)	Data Input to $\overline{ACKIN}\downarrow$ to Setup Time	50		50		ns
2	TdACKf(RFD)	$\overline{ACKIN}\downarrow$ to RFD $\downarrow$ Delay	0	500	0	500	ns
3	TdRFDr(ACK)	RFD $\uparrow$ to $\overline{ACKIN}\downarrow$ Delay	0		0		ns
4	TsDO(DAV)	Data Out to $\overline{DAV}\downarrow$ Setup Time	25		25		ns
5	TdDAVf(ACK)	$\overline{DAV}\downarrow$ to $\overline{ACKIN}\downarrow$ Delay	0		0		ns
6	ThDO(ACK)	Data Out to $\overline{ACKIN}$ Hold Time	50		50		ns
7	TdACK(DAV)	$\overline{ACKIN}\downarrow$ to $\overline{DAV}\uparrow$ Delay	0	500	0	500	ns
8	ThDI(RFD)	Data Input to RFD $\downarrow$ Hold Time	0		0		ns
9	TdRFDf(ACK)	RFD $\downarrow$ to $\overline{ACKIN}\uparrow$ Delay	0		0		ns
10	TdACKr(RFD)	$\overline{ACKIN}\uparrow$ ($\overline{DAV}\uparrow$) to RFD $\uparrow$ Delay – Interlocked and 3-Wire Handshake	0	400	0	400	ns
11	TdDAVr(ACK)	$\overline{DAV}\uparrow$ to $\overline{ACKIN}\uparrow$ (RFD $\uparrow$)	0		0		ns
12	TdACKr(DAV)	$\overline{ACKIN}\uparrow$ to $\overline{DAV}\downarrow$	0	800	0	800	ns
13	TdACK(Empty)	$\overline{ACKIN}$ to Empty		600		600	ns
14	TdACK (Full)	$\overline{ACKIN}$ to Full		600		600	ns
15	$\overline{ACKIN}$ Clock Rate		1.0				μs

Figure 47a. 2-Wire Handshake (Port 2 Side Only) Output

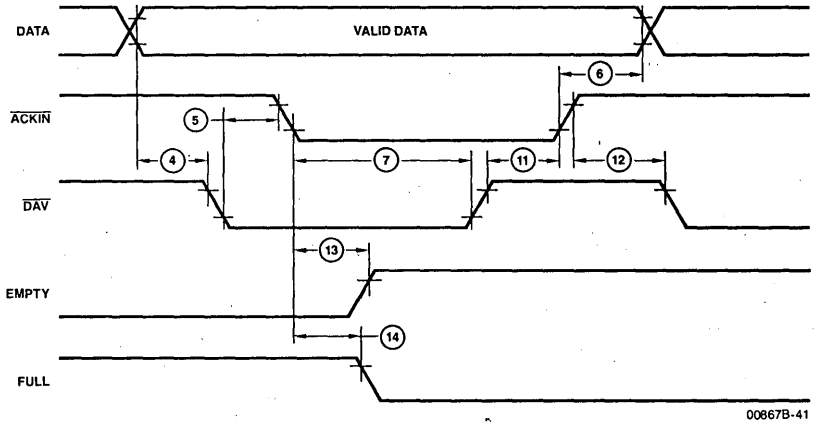
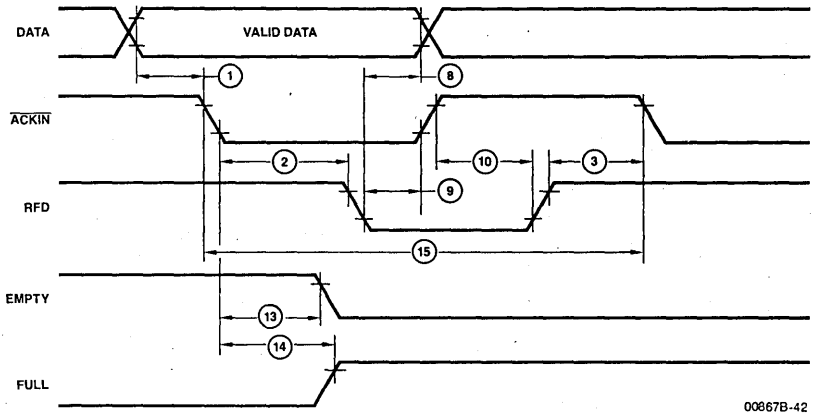


Figure 47b. 2-Wire Handshake (Port 2 Side Only) Input



3-WIRE HANDSHAKE

Number	Symbol	Parameter	4MHz		6MHz		Units
			Min	Max	Min	Max	
1	TsDI(DAV)	Data Input to $\overline{DAV}$ ↓ Setup Time	50		50		ns
2	TdDAVIf(RFD)	$\overline{DAV}$ ↓ to RFD ↓ Delay	0	500	0	500	ns
3	TdDAVf(DAC)	$\overline{DAV}$ ↓ to DAC ↑ Delay	0	500	0	500	ns
4	ThDI(DAC)	Data In to DAC ↑ Hold Time	0		0		ns
5	TdDAClr(DAV)	DAC ↑ to $\overline{DAV}$ ↑ Delay	0		0		ns
6	TdDAVIr(DAC)	$\overline{DAV}$ ↑ to DAC ↓ Delay	0	500	0	500	ns
7	TdDAVIr(RFD)	$\overline{DAV}$ ↑ to RFD ↑ Delay	0	500	0	500	ns
8	TdRFDI(DAV)	RFD ↑ to $\overline{DAV}$ ↓ Delay	0		0		ns
9	TsDO(DAC)	Data Out to $\overline{DAV}$ ↓					ns
10	TdDAVOf(RFD)	$\overline{DAV}$ ↓ to RFD ↓ Delay	0		0		ns
11	TdDAVOf(DAC)	$\overline{DAV}$ ↓ to DAC ↑ Delay	0		0		ns
12	ThDO(DAC)	Data Out to DAC ↑ Hold Time					ns
13	TdDACOr(DAV)	DAC ↑ to $\overline{DAV}$ ↑ Delay		400		400	ns
14	TdDAVOr(DAC)	$\overline{DAV}$ ↑ to DAC ↓ Delay	0		0		ns
15	TdDAVOR(RFD)	$\overline{DAV}$ ↑ to RFD ↑ Delay	0		0		ns
16	TdRFDO(DAV)	RFD ↑ to $\overline{DAV}$ ↓ Delay	0	800	0	800	ns

Figure 48a. 3-Wire Handshake Input

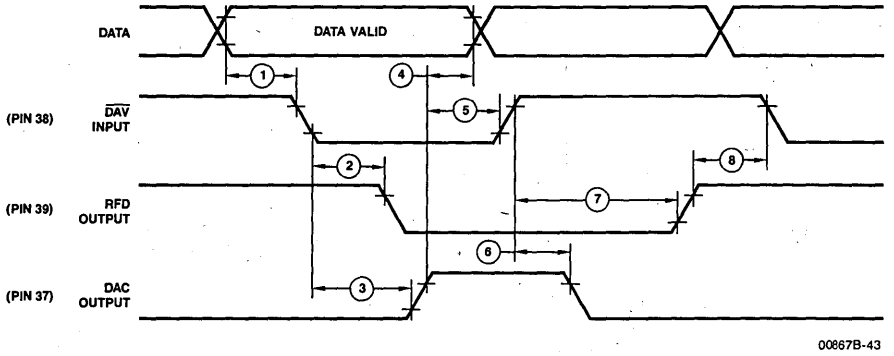
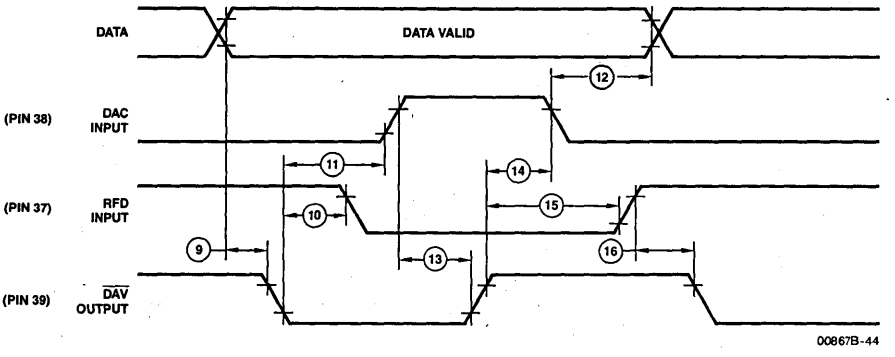


Figure 48b. 3-Wire Handshake Output



AmZ8038* FIO without Interrupt

The AmZ8038 FIO can be used as a buffer between two CPUs in multiprocessor applications by virtue of it's bidirectional feature. In this interface two FIOs are used for a high performance application. Motorola has no FIFO compatible to the AmZ8038.

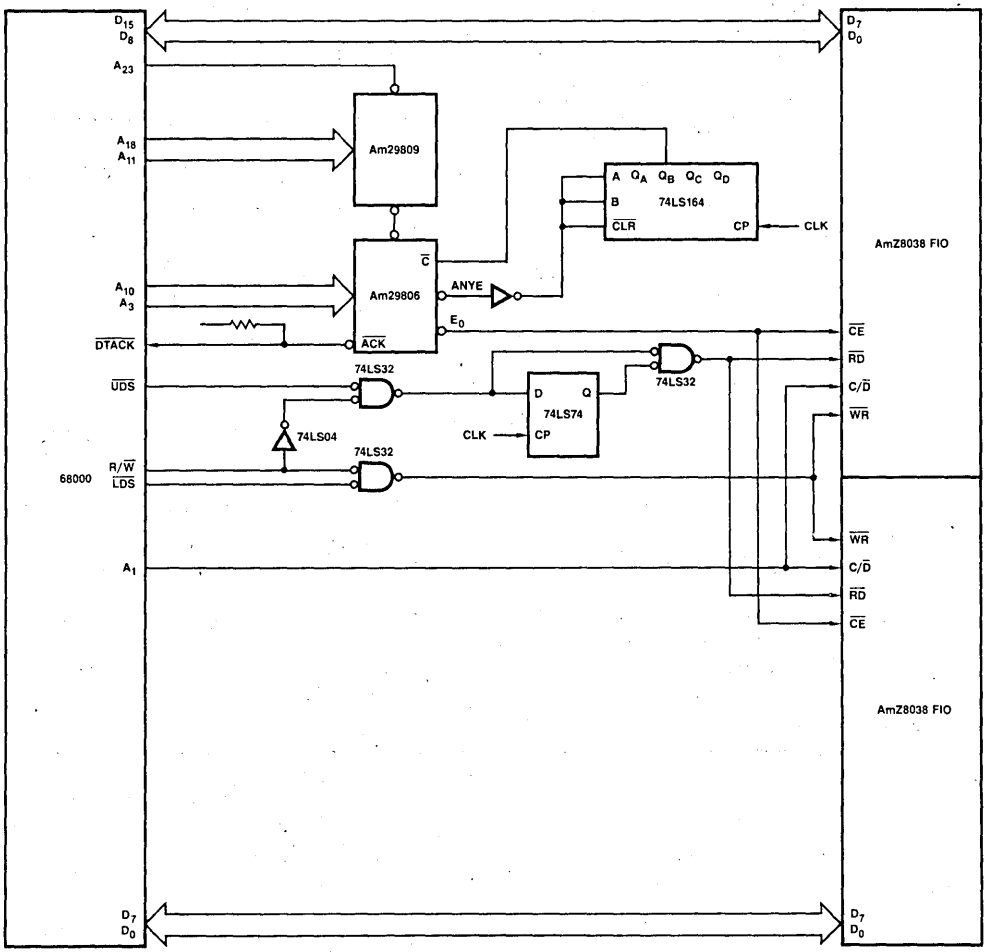
1. The data buses are directly linked together between the 68000 and the AmZ8038. The Am29806 address decoder decodes the 8-bit address. It also generates the ANYE signal that is input to the 74LS164 shift register which generates the appropriate number of wait states by controlling DTACK. This ensures that WR and RD have the required 400ns width. A_1 generates the C/D input to the AmZ8038 while the remaining address lines are connected to the Am29809 address comparator. The 8038 is strapped for RD and WR as it leads to more efficient programming.
2. OR gates 74LS32s convert UDS, LDS, and R/W into RD and WR signals to the AmZ8038.
3. The 74LS74 flip/flop delays the falling edge of RD guaranteeing that the CS to RD timing of the AmZ8038 is met.

INTERFACE DEVICES

Address Decoder	29806	1
Address Comparator	29809	1
Inverters	74LS04	1
OR	74LS32	1
Shift Register	74LS164	1
Flip/Flop	74LS74	1

To implement the interrupt feature to this interface, a circuit similar to the AmZ8530 with interrupts can be configured.

*Z8038 is a trademark of Zilog, Inc.



03347C-3

Why Design-In the AmZ8038 FIO?

The AmZ8038 is a 128 byte FIO which has a number of built-in features which permit design flexibility. Some key features are listed below.

1. **Bidirectionality:** This allows data transfer in both directions. This is a useful feature in multiprocessor applications where 2 or more CPUs talk to each other. The *only* bidirectional FIFO available.
2. **Programmable interface:** Allows easy interface with CPUs with entirely different bus structures without external logic.
3. **Unlimited expansion:** Allows easy upgrading of existing designs.
4. **Other features include:** Seven interrupt sources and 3-wire handshake modes (IEE-488 compatible).

Motorola has no FIFO with the features mentioned above. All 68000 designs where multiple CPUs are involved use a number of first generation FIFOs as buffers. The AmZ8038 would resolve design complications in such systems.

FIFO COMPARISON CHART

Features	AmZ8038	AmZ8060	Am2812	Am2813
Supply	+5V,0	+5V,0	+5V,−12V	+5V,−12V
Package	40-Pin DIP	28-Pin DIP	28-Pin DIP	28-Pin DIP
Technology	N-Channel MOS	N-Channel MOS	N-Channel MOS	N-Channel MOS
Size	128 x 8	128 x 8	32 x 8	32 x 9
Data Rate	1MHz	1MHz	1MHz	1MHz
Ripple Through Time	250ns	250ns	10μs	10μs
Serial/Parallel Opn	Parallel	Parallel	Serial/Parallel	Serial/Parallel
Bidirectionality	Yes	Yes	No	No
Pattern Detect	Yes	Yes	No	No
Interrupt Structure	7 Vectored/Non Vectored	No	No	No
Handshake Signals	Interlocked	Interlocked	No	No
Interface Capability	Programmable for Z-Bus and Non Z-Bus Type CPUs	Z-Bus Only	No	No

FIFO COMPARISON CHART

Features	Am2841	Fairchild 9043	Fairchild 9423	Western 1502
Supply	+5V,−12V	+5V,0	+5V,0	+5V,−12V
Package	16-Pin DIP	24-Pin DIP	24-Pin DIP	28-Pin DIP
Technology	N-Channel MOS	Bipolar	Bipolar	N-Channel MOS
Size	64 x 4	16 x 4	64 x 4	40 x 9
Data Rate	1-2MHz	10MHz	10MHz	0.5MHz
Ripple Through Time	10μs	450ns	2.5μs	?
Serial/Parallel Opn	Parallel	Serial/Parallel	Serial/Parallel	Parallel
Bidirectionality	No	No	No	No
Pattern Detect	No	No	No	No
Interrupt Structure	No	No	No	No
Handshake Signals	No	No	No	No
Interface Capability	No	No	No	No

Am8052/8152/8153

Alphanumeric CRT Controller Chip-Set

ADVANCED INFORMATION

DISTINCTIVE CHARACTERISTICS

- 100MHz video dot rate supports high resolution CRT monitors with 132/60 or 96/66 screen formats
- Background or window soft-scroll capability without external MSI or software overhead
- User-friendly CPU interface. Compatible with 8086, Z8000 and 68000 CPUs.
- On-chip line buffers support flicker free soft-scrolling
- Supports proportional character widths
- Automatic concatenation of up to three trailing blank pixels supports text justification
- Flexible attribute handling
- Color and bit-mapped graphics extensions

GENERAL DESCRIPTION

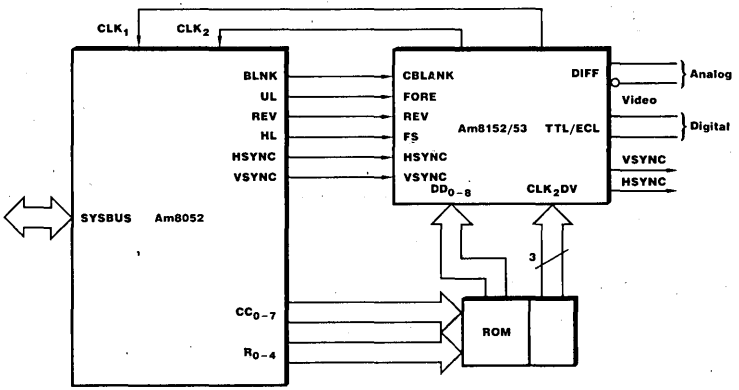
State-of-the-art CRT terminals incorporate advanced user programmable features such as flexible attribute handling, proportional spacing of characters, split screens or multiple window display, soft-scrolling of windows, and variable character width and height in full page 132 x 60 screen formats. The video subsystem of a CRT terminal with these sophisticated video features, can now be implemented with as few as three device packages with the Am8052/8152 chip-set, significantly reducing IC cost and board space without sacrificing performance. The Am8052/8152 chip-set consists of an NMOS LSI CRT Controller (CRTC), the Am8052, and the bipolar IMOX™ Video System Controller (VSC), the Am8152.

The Am8052 has on-chip DMA which operates via linked list data structures to simplify text editing. In addition, it is the only known CRT controller with three line buffers on chip to support flicker free soft-scrolling of background or windows. The Am8052 has on-chip logic that can support a number of attributes such as highlight, reverse video, underline, etc., which enhance display presentation (see Table 1). In addition, four user definable attributes are available providing user flexibility.

The Am8052 CRT controller performs all the data processing prior to video serialization. This latter task is performed by the Am8152, for dot rates up to 40MHz, or by the Am8153 for dot rates up to 100MHz. Apart from the video output lines (TTL on the Am8152 and ECL on the Am8153) these two devices are functionally identical. The high speed of the video controller, a result of AMD's patented IMOX™ technology, supports high resolution screens of 500–1000 scan lines per frame or rows with 100 or more characters. In addition, the Am8152 has on-chip logic that supports proportional spacing of characters and allows for text justification.

The CRT chip-set is designed for easy interface with all popular CPUs such as the 8086, 68000 and Z8000,* thus permitting design flexibility with minimal logic.

TYPICAL APPLICATION WITH PROPORTIONAL SPACING



MMP-008

IMOX is a trademark of Advanced Micro Devices, Inc. *Z8000 is a trademark of Zilog, Inc. 03684A-MMP

This document contains information on a product under development at Advanced Micro Devices, Inc. The information is intended to help you to evaluate this product. AMD reserves the right to change or discontinue work on this proposed product without notice.

Am8052 CRTC

Alphanumeric CRT Controller

DISTINCTIVE CHARACTERISTICS

- On-chip DMA capability, operating via linked-list data structures
- Three on-chip line buffers, each 132 characters by 20 bits support split-screen soft-scrolling
- General purpose microprocessor interface. Compatible with 8086, Z8000 and 68000 CPUs.
- Soft-scrolling capability, with minimal CPU overhead
- Multiple vertical and horizontal screen divisions, with optional soft-scrolling within a window
- Character attributes (12 bits) can be invoked on a character by character basis
- Flexible vertical and horizontal sync control
- Flexible blanking for control of front and back porch positions
- Non-interlace, repeat field interlace, video interlace options
- High resolution five-bit character generator row addressing
- 16M byte system memory addressing capability
- Programmable blink options for cursors and characters

GENERAL DESCRIPTION

The Am8052 CRT Controller is a general purpose interface device for raster scan CRT displays. The CRTC provides efficient manipulation of complex character formats and screen structures to allow sophisticated text display without undue CPU overhead.

The CRTC is a register oriented product that is fully user programmable. The timing definition and operating modes are initialized by the host CPU. Display formats are real-time programmable on a row by row basis. Character attributes are specified on a character or field basis, and are interpreted and acted upon during active display of a character row.

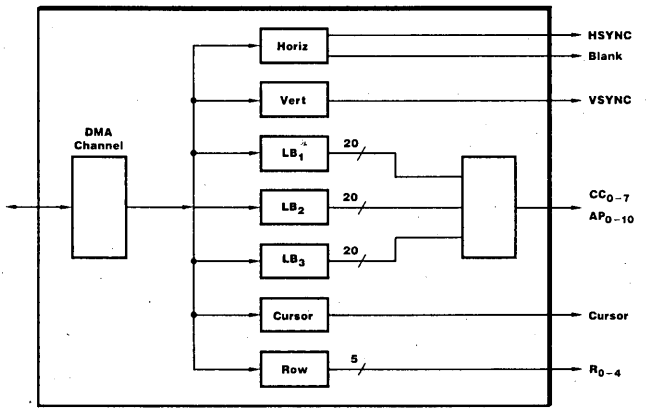
Internal DMA capability assures efficient transfer of display information to the three on-chip line buffers. These three line buffers prevent screen flashing in split-screen soft-scrolling operations. The DMA loads the line buffers via linked list data blocks which facilitates editing and text composition.

The Am8052, in conjunction with the Am8152 bipolar video system controller, allows for the flexible assignment of visual attributes. The twelve attribute bits stored in the Am8052 include superscript, subscript, blink, highlight, reverse, underline, strike through and cursor. Both character and cursor can be made to blink at three different rates and the blink duty cycle is programmable. Further flexibility is achieved by the Am8152 which allows the video stream to be manipulated by selection of background and foreground as well as background/foreground reversal.

The Am8052 and Am8152 combination also supports proportional spacing, text justification and double width characters.

The Am8052 CRTC is assembled in a 68-pin LCC/pin grid package, while the bipolar Am8152 VSC is assembled in a 48-pin DIP. These interface circuits are available as a chip-set for high performance CRT applications.

Figure 1. Am8052 Block Diagram



MMP-009

INTERFACE SIGNAL DESCRIPTION

All inputs and outputs of the CRTC are TTL compatible, unless otherwise indicated. Figure 2 shows the device pin-out.

V_{SS1}, V_{SS2} Ground

V_{CC1}, V_{CC2} +5V Power Supply

CLK_1 **Timing Clock, Input**
The Clock1 signal controls the DMA and peripheral portion of the CRTC, and times all host/memory accesses involving the CRTC. In proportional spacing applications, where CLK_2 is variable, CLK_1 must be used to time the horizontal and vertical sync rates. CLK_1 is non-TTL compatible, and should be driven by the VSC.

CLK_2 **Display Clock, Input**
The Clock2 signal is used to time character accesses from the CRTC line buffers. In applications which do not use proportional spacing CLK_2 is fixed in frequency and can be used to time horizontal and vertical sync rates, allowing CLK_1, the system clock, to be unrelated and asynchronous to the display timing. CLK_2 is non-TTL compatible, and should be driven by the VSC.

AD_0-AD_{15} **Address/Data Bus, Input/Output, Three-State**
The Address/Data Bus is a time-multiplexed, bidirectional, high-true, three-state bus. The presence of addresses is defined by the $\overline{AS}$ signal and the presence of data is defined by the $\overline{DS}$ signal. When the CRTC is in control of the system via its internal DMA capability, it controls the AD Bus; when the CRTC is idle, the CPU or other external devices control the AD Bus and may use it to access the internal registers of the CRTC. The high-order 8-bit memory address is output on the AD_0-AD_7 lines. Interrupt Vector information is also output on the AD_0-AD_7 lines.

$\overline{AS}$ **Address Strobe, Input/Output, Three-State**
Address strobe is a bidirectional, active-LOW, three-state signal. When the CRTC is in slave mode, and the bus master is accessing the CRTC's internal registers, $\overline{AS}$ can be used to optionally latch $\overline{CS}$ and C/$\overline{D}$ information during the first part of the transaction. During a DMA operation when the CRTC is in control of the system, $\overline{AS}$ is an output generated by the CRTC to indicate a valid address on the bus. In slave mode, the $\overline{AS}$ signal may be asynchronous to CLK_1.

$\overline{DS}$ **Data Strobe, Input/Output, Three-State**
Data Strobe is a bidirectional, active-LOW, three-state signal. When the CRTC is in slave mode, and the external system is transferring information to or from it, $\overline{DS}$ is a timing input used by the CRTC to move data to or from the AD bus. In the slave mode, the $\overline{DS}$ signal may be asynchronous to CLK_1. During a DMA operation when the CRTC is in control of the system, $\overline{DS}$ is an output generated by the CRTC and used by the system to move data onto the AD bus.

$\overline{CS}$ **Chip Select, Input**
The $\overline{CS}$ input is an active LOW signal used by the host processor to select the CRTC for a slave transfer.

$\overline{WAIT}$ **Wait, Input**
The $\overline{WAIT}$ input is an active LOW signal used to stretch the $\overline{DS}$ strobe whenever the CRTC has access to the host's bus for data transfer. The status of the $\overline{WAIT}$ signal is sampled on the falling edge of the CLK_1 T_2 time.

R/$\overline{W}$ **Read/$\overline{Write}$, Input/Output, Three-State**
Read/Write is a bidirectional, three-state signal indicating the data direction for the bus transaction under way, and remains stable for the length of the bus cycle. When $\overline{CS}$ input is active, Read (HIGH) indicates that the system is requesting data from the CRTC and Write (LOW) indicates that the system is presenting data to the CRTC. On the other hand, during a DMA operation when the CRTC is in control of the system, R/$\overline{W}$ is an output generated by the CRTC, with Read indicating that data is being requested by the CRTC from the addressed memory location, and Write to indicate that the CRTC is outputing a high-order address to an external latch. During a dummy DMA cycle, R/$\overline{W}$ is driven high.

$\overline{BRQ}$ **Bus Request, Input/Output**
When the CRTC requires use of the bus for DMA activity, the $\overline{BRQ}$ line is driven LOW. It remains LOW until it has ceased using the bus.

$\overline{BAI}$ **Bus Acknowledge In, Input**
Bus acknowledge In is an active LOW input. When the CRTC requires host bus access and has successfully pulled its $\overline{BRQ}$ pin LOW, a $\overline{BAI}$ LOW input signifies that the CRTC has obtained bus mastership after having internally synchronized its $\overline{BAI}$ active LOW input for two clock periods of CLK_1. The synchronization is required to alleviate metastable problems. When the CRTC does not require host bus access, the $\overline{BAI}$ input ripples to the $\overline{BAO}$.

CURSOR **Cursor, Output**
This pin is the cursor output indicator.

ESYNC **External Sync, Input**
This pin is the external synchronization input line. If the ES bit in the mode register is set, the vertical frame scan will commence after the falling edge of ESYNC.

HSYNC **Horizontal Sync, Output**
HSYNC is an active HIGH output used to cause horizontal retrace of the CRT's electron beam. The output is held active LOW while the CRTC is reset to prevent unknown synchronization to the CRT which may cause damage to high bandwidth tubes. Note that this pin can also be initialized as Horizontal Drive.

VSYNC **Vertical Sync, Output**
VSYNC is an active HIGH output used to cause vertical retrace of the CRT's electron beam. A VSYNC is an active HIGH output used to cause vertical retrace of the CRT's electron beam. VSYNC can be optionally synchronized by the

ESYNC input. VSYNC is held HIGH while the CRTC is reset to prevent damage to the CRT.

BLANK — **Blank Video, Output**
BLANK is an active HIGH output. It serves to blank out inactive display areas of the CRT. The output is held active while the CRTC is reset.

R_0-R_4 — **Row Control, Outputs**
R_0-R_4 outputs are active HIGH. These outputs represent the binary count of the active scan line being displayed. These outputs address the least significant address portion of an external character generator. The outputs are all held high for those scan lines that do not carry active video during normal character or superscript/subscript display.

CC_0-CC_7 — **Character Code Outputs**
CC_0-CC_7 outputs are active HIGH. The 8-bit character port, CC_0-CC_7, outputs eight bits of data stored in the character code section of the line buffer currently being displayed.

$\overline{INT}$ — **Interrupt Request, Output, Open Drain**
This line is used to indicate an interrupt request to the host processor. It is driven LOW by the CRTC, until an interrupt acknowledge is received on the $\overline{INTACK}$ pin.

$\overline{INTACK}$ — **Interrupt Acknowledge, Input**
When $\overline{INTACK}$ is driven LOW, the CRTC examines its IEI line to determine whether it has been granted an acknowledge by the CPU.

IEI — **Interrupt Enable-In, Input**
A HIGH on IEI during an Interrupt Acknowledge cycle is regarded as an interrupt acknowledge to the CRTC. A LOW on IEI during Interrupt Acknowledge signifies that a higher priority interrupt on the daisy chain is being acknowledged.

IEO — **Interrupt Enable-Out, Output**
IEO follows IEI during Interrupt Acknowledge if the CRTC has not made an interrupt request. IEO LOW disables lower priority devices from making interrupt requests.

$\overline{DTEN}$, $\overline{DREN}$ — **Data Transmit Enable, Data Receive Enable, Outputs, Open Drain**
Data Transmit Enable and Data Receive Enable are used to control bus transceivers external to the CRTC, should they be required. When $\overline{DTEN}$ is low, the transceivers should transmit from the CRTC onto the bus. When $\overline{DREN}$ is low, the transceivers should receive data from the bus. $\overline{DTEN}$ and $\overline{DREN}$ are never low simultaneously.

C/$\overline{D}$ — **Command/$\overline{Data}$, Input**
C/$\overline{D}$ is used by the CRTC when in slave mode, to determine if an I/O transaction with the host CPU is transferring a command or data. When the CRTC is not involved in an I/O transaction with the host, C/$\overline{D}$ is disregarded.

AP_0-AP_{10} — **Attribute Port Outputs**
These 11 lines are used to display character attribute information synchronous with each character and CLK_2. During horizontal SYNC, the row attribute information contained in the Row Redefinition Block are output on AP_0-AP_{10}.

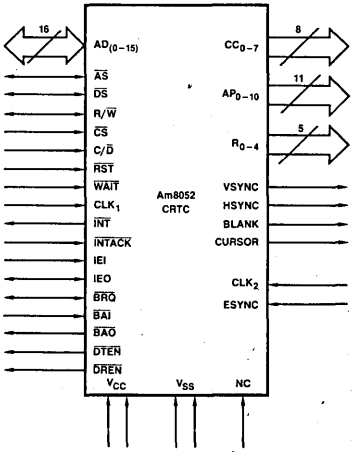
$\overline{BAO}$ — **Bus Acknowledge Out, Output**
$\overline{BAO}$ output is forced active HIGH when the CRTC requests bus mastership, otherwise the $\overline{BAI}$ input ripples out of the CRTC via the $\overline{BAO}$ output.

$\overline{RST}$ — **Reset, Input**
A Low on this input for at least 5 clock cycles is interpreted as a reset signal. The effect of reset is to drive all CRTC bus signals into the high-impedance state, to clear all bits in mode registers 1 and 2, and to force the CRTC into slave mode.

$\overline{RSTT}$ — **Test Reset, Input**
For test use only. This pin is a No Connect.

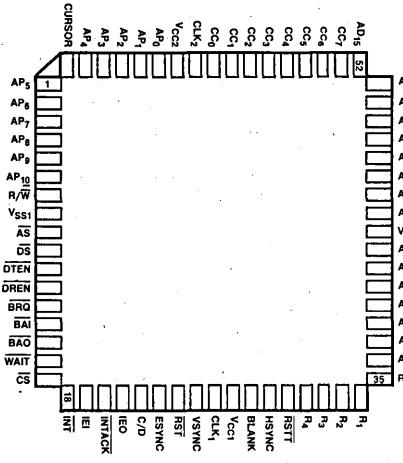
Figure 2. Am8052 Pinout

LOGIC SYMBOL



MMP-010

BOTTOM VIEW



Note: Pin 1 is marked for orientation.

MMP-011

TABLE 1.

Attribute	Effect
Reverse	– Causes the applicable character to be displayed in reverse video.
Highlight	– Highlights the applicable character.
Blink	– Blinks an applicable character at one of four programmed blink rates.
Underline	– Underlines the applicable character at a programmable scan line.
Subscript	– Causes the character to be displayed as a subscript.
Superscript	– Causes the character to be displayed as a superscript.
Shifted Underline	– A second underline.
Cursor	– Causes the attribute cursor to be displayed at the applicable character.
Latched	– Indicates that the attribute should be latched for all successive characters until changed.
Ignore	– Causes the CRTC to skip over the applicable characters. Useful for embedded control characters and protected fields that do not get displayed.
User Definable	– Four attribute bits reserved for user definition.

FUNCTIONAL OVERVIEW

The block diagram of the Am8052 CRTC is shown in Figure 1. Communication with the external host system takes place over the 16-bit Address/Data bus, $\overline{AD_0}-\overline{AD_{15}}$. Transfers over the AD bus are controlled by the $\overline{CS}$, C/$\overline{D}$, $\overline{AS}$, $\overline{DS}$, and R/$\overline{W}$ lines. When the CRTC is in slave mode, these four bus control lines are inputs. When the CRTC is in DMA mode. $\overline{AS}$, R/$\overline{W}$ and $\overline{DS}$ are outputs and control the external bus.

Following reset, the host system initializes the CRTC's timing and control registers, as well as one address pointer to the start of the display data location in the host memory. Following initialization and upon command from the host, the CRTC takes over bus control from the host and transfers display row control data, character code, and character attribute data. The CRTC requests the host bus by sampling the $\overline{BRQ}$ line for activity; if the $\overline{BRQ}$ line is HIGH, the CRTC drives it LOW, and also drives $\overline{BAO}$ HIGH, to obtain priority over lower priority bus requestors. The on-chip DMA Controller circuit controls the data transfer and performs character data loading into the on-board line buffers.

The CRTC is real-time programmable on a character row by row basis through a row control data block fetched either from the host memory or from a dedicated display memory. The row control block contains address links to the next row's row control block, a character and attribute data address for the current row and other pertinent control functions for the row. Data from the row control block is transfered into the appropriate set of registers for active control of display and data fetch operations during the subsequent display of character row data. A Top Of Page register contains the address of the Main Definition Block for the screen. The Main Definition Block in turn, points to the first Row Control Block. The character row data, comprised of character code and attribute, (if the latter is specified) is fetched starting at the address and for the character length obtained from the Row Control Block. The character code and its attribute consist of a 20-bit wide word which is stored, FIFO style, into one of the three on-board 132-character by 20-bit line buffers. Character attributes are on a character by character basis and are interpreted and acted upon by the CRTC during the active display period of the contents of a line buffer. Output lines CC_0-CC_7 form the transfer path for character code data to an external matrix type character generator, while the character attribute, after selective masking, is interpreted and acted upon.

Output lines R_0-R_4 exhibit the scan line number for the specific character being displayed, while the character Row Control logic allows alteration of the scan line number output at the R_0-R_4 lines to enable the display of normal (single character), superscript or subscript. The R_0-R_4 lines also output a 1F (hex) Code for blanking portions of a character cell matrix and for character blinking.

The HSYNC, VSYNC and BLANK output lines provide the CRT synchronization signals. The Horizontal Control and Vertical Control logic blocks contain counters and host programmable registers for deriving the timing signals from either the CLK_1 or the CLK_2 input as well as an ESYNC input line for frame synchronization to an external source, such as the power line frequency. Clock input CLK_2 is a clock which runs at the character rate of the display, and is a submultiple of the dot clock, whose frequency is determined by the Am8152 oscillator. CLK_2 controls the CRT synchronization lines HSYNC and VSYNC, as well as BLANK, and the rate of character output from the CRTC. CLK_1, which may be asynchronous to CLK_2, controls all DMA and related bus activity, associated with the CRTC. In proportional spacing applications CLK_1 may be also used to time the synchronization signals.

CHARACTER ATTRIBUTES

Character attributes affect various CRTC output signals and other operations on a character by character basis. Each attribute word occupies a 16-bit word in memory. Each character, however, need not invoke a new attribute.

Character attributes are stored in parallel with the corresponding character code in each line buffer.

The character attribute information which makes up the character attribute word is shown below:

AW_{15}	Latched/unlatched	AW_7	User definable
AW_{14}	Cursor/definable	AW_6	Highlight
AW_{13}	Ignore	AW_5	Reverse
AW_{12}	Reserved	AW_4	Superscript
AW_{11}	Reserved	AW_3	Subscript
AW_{10}	User definable	AW_2	Shifted underline/ strike through
AW_9	User definable	AW_1	Underline
AW_8	User definable	AW_0	Blink

DESCRIPTION OF CHARACTER ATTRIBUTES

LATCHED/$\overline{\text{UNLATCHED}}$

When this bit is set to 1 ("latched") the attribute information applies to all characters following the character that invoked

the attribute word. Only the presence of a further latched attribute word cancels the effect of a previous latched attribute word. If the Latched/Unlatched bit is set to 0, ("unlatched") then the attribute information only applies to the character that invoked the attribute word. All successive characters are modified by the latched attribute information that was valid prior to the unlatched attribute word. The Latched/Unlatched bit is not output to the attribute port. The initial state of the latched attribute value is undefined. At the start of any horizontal line, the latched attribute information is the same as at the end of the previous line, unless changed by a further latched attribute.

CURSOR

If this bit is set, then a cursor is displayed at the affected character position(s), dependent upon the mode of the cursor display logic. See the section on cursor display for further details.

IGNORE

When the ignore is set, it inhibits the loading of the associated character into the CRTC line buffer. Such character(s) may be used as control character or software tags, and are not displayed. Whenever the ignore encoding is detected, both the attribute word and its associated character code are not written into the line buffer, unless the DH bit in Mode Register 1 is set. Note that the ignore bit is not brought out to the attribute port.

USER DEFINABLE

The AW_7-AW_{10} attribute bits provide 4 bits of user definable attribute information. These bits are directly output on pins AP_7-AP_{10} of the attribute port. (In addition to these four user definable attribute bits, the cursor bit can also be user definable under certain conditions.)

HIGHLIGHT

When this bit is set, the character is displayed highlighted. The AP_6 pin of the attribute port goes active for each scan line of the relevant character(s).

REVERSE

When this bit is set to 1, the character is displayed reversed. The AP_5 pin of the attribute port goes active for each scan line of the relevant character(s).

SUPERSCRIPT

When this bit is set to 1, the affected character is displayed as a superscript. Its position on the character row (R_0-R_4) is determined by the superscript control field in the row redefinition block, for that particular row.

SUBSCRIPT

When this bit is set, the affected character is displayed as a subscript. Its position on the character row (R_0-R_4) is determined by the subscript control fields in the row redefinition block.

NOTE: If both superscript and subscript are set, the character appears as normal.

UNDERLINE/SHIFTED UNDERLINE

Attribute bits AW_1 and AW_2 provide underline and shifted underline display. The underline/shifted underline display information is output on the AP_1 and AP_2 attribute port pins, during applicable scan lines of the character. (The applicable scan lines have been programmed within the row redefinition blocks.)

BLINK

When this attribute is invoked, the attribute port pin AP_0 is gated with the character blink rate generator, during the time that the relevant character is output on CC_0-CC_7.

The character blink rate and character blink duty cycle are derived from the blink field of the Main Definition block.

ATTRIBUTE FETCHES

Attributes can be fetched in three different ways to suit most design philosophies (see Figure 4 below). In Option 1, one attribute is fetched per character. This option, although straight forward, imposes heavy bus overhead since the DMA has to access the attribute list from memory for every character displayed on the screen. Bus overhead can be reduced considerably by fetching attributes on a demand basis. Option 2 and 3 accomplish this in two different ways. In Option 2 one character bit is set to 1 when an attribute is required. When this bit is set to 0, the attribute will not be fetched. This option allows 7 bits of character code or a 128 character set for display with no overhead for attribute incorporation.

Option 3 makes use of an 8-bit flag which precedes the character invoking the attribute. This option allows for a 255 character set with an 8-bit overhead (the flag) per attribute.

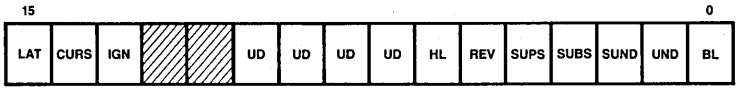
CURSOR GENERATION

The CRTC is capable of outputting a variety of different cursors, including block, underline and reverse, at variable blink rates and blink duty cycles.

Cursor information for the CRTC comes from two different sources, and each source can be independently steered to one of three different destinations. The two cursor sources are:

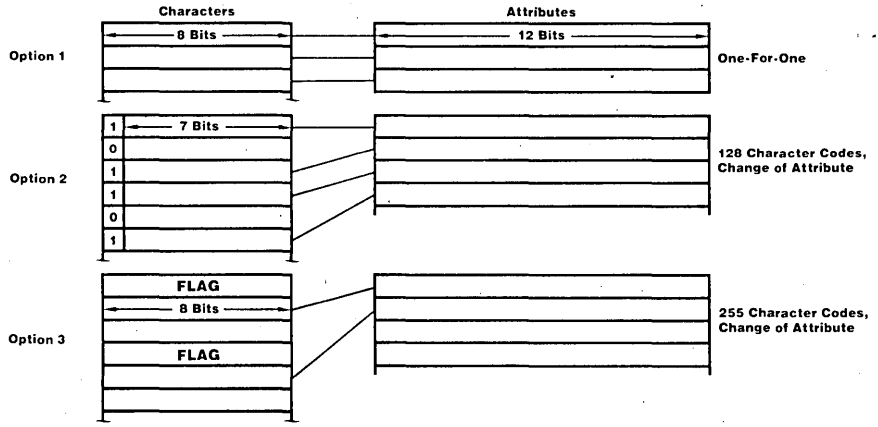
1. The XY cursor field which is held in the Main Definition Block for the screen.
2. Attribute Word bit 14 of the character attribute word. A cursor designated by an attribute will follow its row and character position whenever text is scrolled. The cursor controlled by positioning X and Y coordinates within the cursor X and Y register will be displayed on a fixed X, Y character position on the screen.

Figure 3. Am8052 Attribute Word



MMP-012

Figure 4. Attribute Fetch



MMP-013

The steering of the cursor sources is under software control of the cursor mask field within mode register 2. The field is divided into two three-bit segments, one for the XY cursor and one for the attribute cursor. Three destinations are selectable for each cursor source:

(a) The cursor pin
(b) The underline pin
(c) The reverse video pin.

If (a) is selected, then either the whole character cell or partial character cell is selectable. If whole is selected, the cursor pin will be active for every scan line of the character cell. If part is selected, then the cursor pin will only be active for those scan lines within the limits of CURSOR START and CURSOR END, as specified in the row-redefinition information.

If (b) is selected, then either an underline will be active, if CURSOR START and END have the same values, or a block, if CURSOR START and END are not coincident.

If (c) is selected, then either all or part of the character will be reversed, dependent upon the CURSOR START and CURSOR END setting as explained in (b).

In addition to these choices, either cursor can be made to blink, at the cursor blink rate, and duty cycle, as programmed into the main definition block blink field.

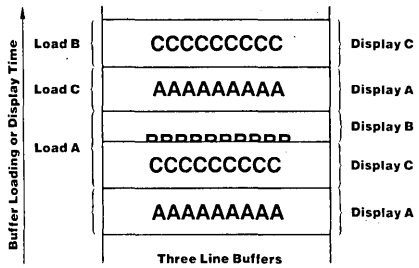
LINE BUFFERS

The on-chip DMA controller accesses the display memory and loads data from linked-list data blocks in memory into one of three line buffers. Each line buffer is 132 characters in length and 20 bits wide. The 20-bit wide locations accommodate 8-bit character codes and 12-bit attribute words associated with the character. The line buffers operate in a rotating fill-display mode whereby one buffer is being loaded while another is being displayed.

The presence of three line buffers on-chip is of significant advantage in split screen soft-scrolling operations where a character row may only be displayed for a single scan line. With two line buffers, this would not leave enough time for the reloading of the alternate line buffer. A partially filled buffer results in screen flashing. This can only be prevented by incorporating three line buffers. Figure 5 highlights this advantage.

Note that only the shaded line buffers are scrolling on the horizontally split screen.

Figure 5.



MMP-014

In the rotating fill-display mode, Line Buffer C is displayed when Line Buffer B is being loaded. Likewise the next Line Buffer C is loaded while Line Buffer A is being displayed. Because of the split-screen, Line Buffer B is displayed for one scan line only, while Line Buffer A is being loaded. By virtue of the third line buffer the loading of Buffer A can spill over into the next buffer display, thus eliminating screen flashing.

SOFT-SCROLLING

A soft-scroll is defined as the gradual displacement of a character row on a scan line by scan line basis. Soft-scrolling is achieved by a gradual offsetting of the scan line counter, on a frame by frame basis. At the start of the scroll, the offset counter is set to zero or equal to the number of scan lines per character row depending on whether the scroll is up or down. As the counter is incremented or decremented, the text travels up or down until the offset is equal to the number of scan lines or zero. The start of the screen pointer pointing to the character row is adjusted and the offset counter reset simultaneously to scroll the next successive character row. Soft-scrolling of the entire screen is thus a simple task. However, implementing a split screen soft-scroll is cost prohibitive and complex in MSI.

A number of applications require screen overlays, such as menu or status areas which must remain static while the major portion of the screen is scrolling or vice versa. The Am8052 can support multiple windows, with each being independently scrollable. This feature is described in detail in the following section.

LINKED LIST DATA STRUCTURES

The DMA channel on the Am8052 operates via linked list structures that allow for the overlaying and independent soft-scrolling of windows. The linked list data structures are particularly suited to the manipulation of data strings where insertions and deletions are common. A typical CRTC Linked List Structure is shown in Figure 6.

The linked list consists of Row Control Blocks (RCBs) for each character row on the screen. The RCB does not contain any displayable data, but contains the address which points to the character information. Each RCB is linked to the next block via an address link word (RCB ADR). The structure of the RCB linkage is shown in Figure 7. The Top of Page register on-chip points to the Main Definition Block which in turn points to a linked list of RCBs.

The Am8052 allows for the separation of attribute and character lists. By extending the RCB, split screen segments can be constructed as in the case of RCB_2 in Figure 7. In parallel with the screen or background data structure, there exists a window structure which contains Window Control Blocks (WCBs) for each row of each window. Windows can exist in any position on the screen and are overlayed on top of the screen or background information. For example, the structure shown in Figure 8 could be used to implement a menu overlay at the top of the screen together with a status overlay.

MAIN DEFINITION BLOCK

The Main Definition Block is a set of control data and addresses, located in the system memory, which allow the user to specify screen oriented features. The TOP OF PAGE register points to the first word of the Main Definition Block. Cursor position, fill code and scroll rate are set by the appropriate fields within the block. The Main Definition Block also points to the first Row Control Block.

Figure 7. Background Data Structure

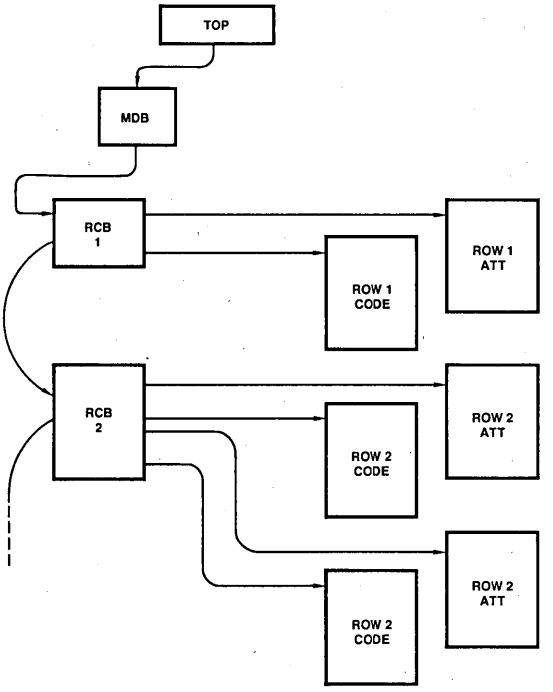


Figure 6. Am8052 Linked List Structure

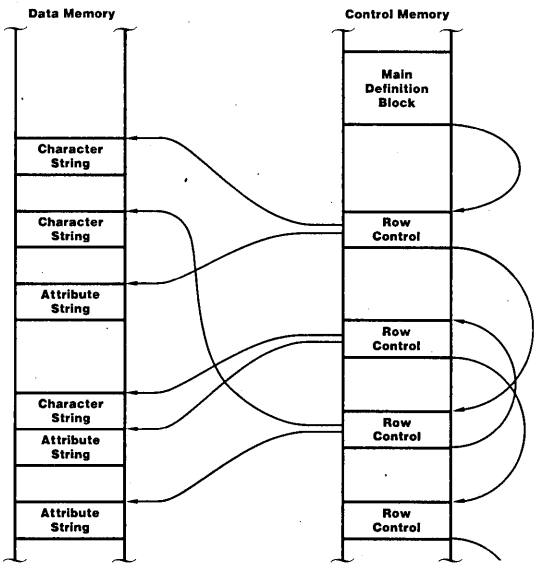
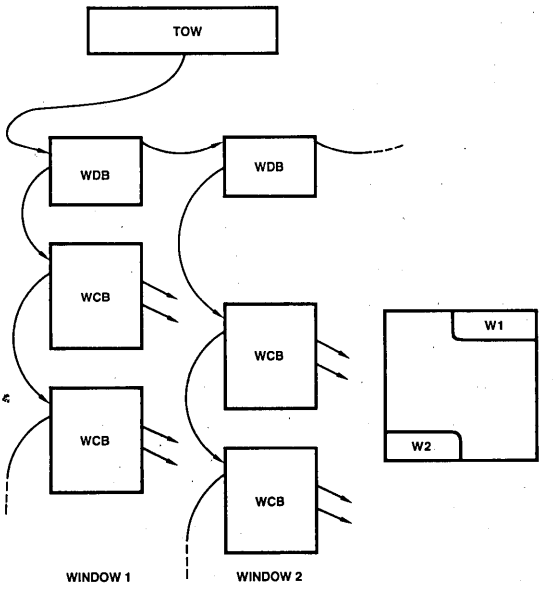


Figure 8. Window Data Structure



ROW CONTROL BLOCKS

The RCB ADR POINTER in the Main Definition Block points to the first word of the first Row Control Block of the list. Each Row Control Block in the main chain is linked to the next via the RCB ADR (Row Control Block Address) word address pointer. Changing the RCB Link Pointer within a Main Control Block allows quick insertion or deletion of a character row (line).

Attributes associated with character data exist in their own separate lists. A character row may be composed of one or more segments of data. Each segment is a block of words with consecutive addresses. A Row Control Block has a character code and character attribute segment start address pointer: SEG CODE ADR for character codes (2 character codes/word) and SEG ATR ADR for character attributes. A third word, SEG LENGTH and SEG DISPLAY defines the number of characters (byte count) contained in the segment as well as the number of displayed characters in the segment. Character attributes are in word format and there can be as many character attributes as character codes.

WINDOW DEFINITION BLOCK

The Window Definition Block defines the size and location of the window. It is the header block to a list of Window Row Control Blocks and can also point to another Window Definition Block if more than one window is displayed on the screen. The TOP OF WINDOW register points to the first word of the first Window Definition Block. Within the first Window Definition Block, the WCB ADR points to the current window's first Window Row Control Block, while the NEXT WDB ADR points to the next window's Window Definition Block. Window size is specified by two words in the Window Definition Block. STRT WINDOW ROW # and END WINDOW ROW # are byte values which position the window vertically on the screen. The window display becomes active in the character row number specified by STRT WINDOW ROW #, and will become inactive in the character row following END WINDOW ROW #.

WINDOW ROW CONTROL BLOCKS

The Window Row Control Blocks have the same format as the Row Control Blocks.

The WCB Link Pointer is the address link to the next row's Window Row Control Block. A window can also be described with segments, and the Window Row Control block contains several words for each segment: WSEG CODE ADR, WSEG ATR ADR, and WSEG LENGTH.

To hard-scroll a window, it is only necessary to change the WCB ADR in the Window Definition Block to a different Window Row Control Block.

WINDOW DISPLAY MECHANISM

A window is any bounded area on the screen which is linked in by a Window Definition Block. The window has the following size characteristics:

Width: Defined by the number of character code positions occupied within a character row. Maximum width is the length of the line buffer (132 characters), and minimum width is one character.

Height: Defined by the number of displayable character rows contained within the window. The maximum height is the total number of displayed character rows on the face of the screen. The height limit is specified by the number of Window Row Control Blocks in the window linked list. The minimum height of a window is one row.

Window Positioning

The window is originally positioned to occupy any portion of the displayable character rows. It can be as large as the full screen or as small as one row high and one character wide. The window is always unscrolled when first displayed. (The counter holding the value of the first scan line of the uppermost character row of the window is reset.)

The window must be positioned horizontally such that its left- and right-hand sides begin and end at a main character row segment boundary. Any unfilled character positions within the window segment, and following the end of the window segment to the end of the line buffer (character position 131) are filled with the fill character code, obtained from the Main Definition Block.

Multiple Windows

Multiple windows can be displayed simultaneously, but only one window can be soft-scrolled at any particular time. Windows cannot be horizontally aligned to each other, and hence must be specified on non-overlapping character row boundaries (see section on virtual windows). Each window is defined by a Window Definition Block, and the scrolling window is designated by a control bit within the Window Definition Block.

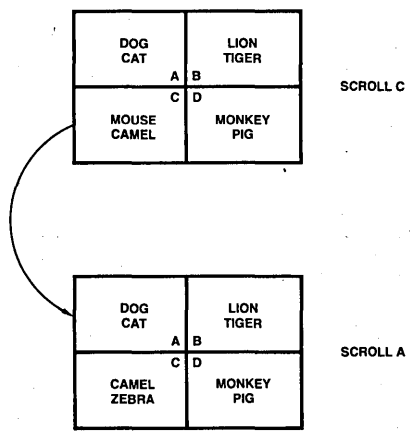
WINDOW POSITIONING

The window position is defined in the Window Definition Block. The coordinate units are background character rows (Y START), (Y END), and background character columns (X START), (X END). When the background is scrolling the window (or windows) should remain stationary on the display.

EXAMPLE OF WINDOW OVERLAYS

The example below (Figure 9) explains how windows are constructed using the Linked List feature that the Am8052 provides.

Figure 9. Example of Vertical Split Screen Smooth Scroll



MMP-018

Step 1

The first step toward constructing windows on a CRT screen is to split the screen horizontally and vertically using row control blocks with multiple data pointers. The data pointers in each RCB point to the first characters within each subscreen area defined by the horizontal/vertical splits. In this example, the RCB that controls the first character row (DOG/LION) contains two data pointers. The first points to subscreen DOG and the second to subscreen LION. The SEG LENGTH information in the RCB indicates to the DMA to switch from data field DOG to data field LION. The Linked List Structure for this example is shown in Figure 10. Note that in most applications, this split screen will have been set up prior to the invocation of the window.

Step 2

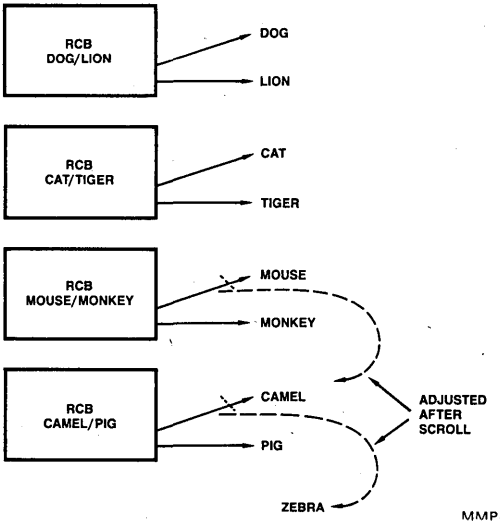
A window can now be overlayed on to the background by the creation of a window linked-list as shown in Figure 11. The scrollable window has a linked list structure pointed to by the Top of Window (TOW) pointer which functions similarly to TOP. The other information required for window definition is the STRT WINDOW CHAR # and END WINDOW CHAR # which define the start/end coordinates of the window. To effect a window scroll just one change to the TOW value is required which significantly relieves CPU overhead.

Virtual Windows

Although the rules of multiple windows do not permit overlapping windows, the background and window structures can be used to implement virtual horizontally aligned windows. This can be best described by using the illustration in Figure 9. The screen is divided into 4 subscreens A, B, C and D, each of which can be independently defined as windows using the Linked List Structures similar to Figure 11.

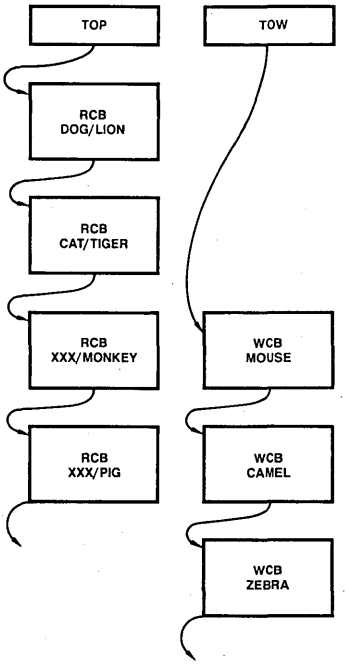
If subscreen C is defined as a window, subscreen A, B and D can be configured to be the background. Window C can be scrolled independently of the background by TOW pointer manipulation. Similarly subscreen D can be defined as a window with A, B and C configured as background. Thus, two aligned subscreens can be independently defined as windows by intelligent use of linked list structures, giving the user the illusion of aligned windows.

Figure 10. Split Screen Control Blocks



MMP-020

Figure 11. Window Overlay Structure



MMP-019

HORIZONTAL SCREEN FORMAT

The horizontal format defines the general timing of a single raster scan line. The scan line consists of two basic periods: visible raster line scan from left-to-right across the CRT screen and the right-to-left beam retrace period (or horizontal sync). The beam is always blanked during the retrace period. The front and back porch periods on either side of the horizontal sync are also blanked because no active video is desired during that time.

Horizontal scan frequencies range from a minimum of 15kHz for small screen, low bandwidth CRTs up to about 60kHz for 100MHz bandwidth large screen CRTs. The horizontal format versatility must accomodate this wide range scan frequencies. The horizontal circuit generates three basic timing signals: horizontal sync, horizontal drive, and blanking. Either horizontal sync or horizontal drive is output at the HSYNC pin as selected by a bit in the mode register. The horizontal blanking signal is "ored" with the vertical blanking signal prior to output at the BLANK pin.

HORIZONTAL TIMING CONTROL

Horizontal timing is controlled by the $\overline{RST}$ signal and the DE (Display Enable) bit in the mode register.

The HSYNC output is disabled (inactive) and the BLANK output active whenever the CRTC is reset by $\overline{RST}$ input (active low) or whenever the DE bit is reset (display disabled). $\overline{RST}$ active low is a hardware reset to the CRTC (this action also resets DE bit), and the DE bit is a software reset of the CRTC.

Am8052 VERTICAL SCREEN FORMAT

The vertical format defines the number of horizontal scan lines to be displayed in each frame. The front and rear porches, as well as the vertical retrace time, are also defined.

The CRTC operates in either an interlace or non-interlace mode. The I bits, in the Mode register, determines if the CRTC will operate in the interlace or non-interlace mode. See below for each of the interlace options.

The Vertical Line Counter is clocked by either the horizontal sync rate in the non-interlaced or twice the horizontal sync rate in the interlaced mode. In non-interlaced mode all vertical frames (period between two vertical sync pulses) are *even*. In interlaced mode, the first vertical frame following a Display Enable (setting of DE bit in the Mode register) is always *even* and alternates between odd and even from there on.

EXTERNAL SYNC OPERATION

The ESYNC input allows synchronization of the CRT display vertical frame rate to the power line frequency to eliminate interference effects. The ES bit in Mode register specifies whether the ESYNC input is used to control the Vertical Sync rate.

The ESYNC input is recognized by the CRTC during every frame. It causes the VSYNC signal to become active at the occurrence of HSYNC. In non-interlaced mode, VSYNC becomes active at the rising edge of HSYNC active; in interlaced mode, VSYNC becomes active at the next HSYNC, active when in the even frame, or the next half point between HSYNCs (2x HSYNC) in the odd frame.

INTERLACE

There are two types of interlace, Repeat Field Interlace (RFI) and Interlaced Video (IV). The effect of both schemes is to offset the vertical position of the scan lines of the odd numbered fields so that they will be physically interleaved with the scan lines of the even fields. For RFI, the same video information is displayed on both odd and even fields. The slight offset of the odd field tending to eliminate the horizontal stripes that sometimes occur between scan lines of non-interlaced displays.

Interlaced Video is used to increase the amount of information displayed on a monitor without increasing the horizontal or vertical scan rates. IV takes advantage of the Odd field scan line offset by displaying half the video in the even field (alternating lines) and half in the odd field. The effect is to essentially double the vertical character density with respect to RFI or non-interlace.

Am8152/8153 VSC

Video System Controller

DISTINCTIVE CHARACTERISTICS

- 100MHz video dot rate
- Four-level current driven (75Ω) differential video output
- Two-bit digital video output
- On-chip crystal driven oscillator
- Proportional spacing support (2 to 17 dots)
- Nine-bit dot data parallel input, with serialization capability to seventeen bits
- Trailing blanks (0 to 3 dots)
- Double width characters
- Attribute support:
  - Character blink
  - Underline
  - Over-strike
  - Reverse
  - Highlight
- Buffered and synchronized character clock outputs
- Background color selection
- Buffered and synchronized vertical and horizontal sync outputs
- TTL compatible

FUNCTIONAL DESCRIPTION

The VSC, Video System Controller, primarily supports the Am8052 CRTC for black and white or color video display. The VSC is offered in two versions. The Am8152 provides a TTL digital video output and supports video dot rates of up to 40MHz. The Am8153, on the other hand, provides an ECL video output with dot rates of up to 100MHz.

The essential functions of the VSC, shown in Figure 12, are to serialize video data, support proportional and non-proportional character display, to correctly synchronize and mix character attributes with video, and to output the video in a four level analog or digital format. Additionally, the VSC generates the CLK_1 and CLK_2 signals for the Am8052.

Proportional spacing is supported by programming, on a character-by-character basis, the number of dot clock periods per character. The character pixel information is selectable from two to seventeen pixels per character, and in addition, up to three trailing blank pixels can be concatenated to the character pixel information.

The character clock (CLK_2) output to the Am8052 is frequency modulated according to the chosen number of dots per character cell. CLK_2 is divided down from an internal crystal driven oscillator whose divide ratio is set according to the width of the character cell plus the number of trailing blanks. In addition, a double width input allows doubling the character width by further modifying CLK_2. The internal oscillator is capable of generating a dot clock frequency up to 100MHz. $MCLK_1$ (Am8052 system clock output) is also derived from the dot clock by an externally programmable divider (by 1 to 8) together with a divide by 2 prescaler. The CLK_2 clock may be internally resynchronized to CLK_1 during Blank input active. This action is necessary to prevent vertically non-aligned character cell lines at the left-end side of the display, in proportional spacing applications.

The vertical and horizontal sync inputs from the Am8052 are buffered and delayed by a CLK_1 or CLK_2 clock period in order to phase correctly with the character video output.

Principal video information is obtained from the parallel character generator inputs which are shifted out serially and mixed with attribute information such as underline, shifted underline, and any other video sources. The video shift register may be up to 17 dots wide. The first 9 bits of the shift register are loaded on the LOW to HIGH transition of CLK_2 and the remaining 8 bits are loaded on the subsequent HIGH to LOW CLK_2 transition. Video is internally encoded into one of four levels, and is output through two ports. One port provides a 4-level current source output into a 75Ω impedance and the second port outputs encoded ECL/TTL video (ECL with Am8153 and TTL with Am8152) on two pins.

The four levels of video are: White, Grey, Black and Blank – where White is the highest analog current level, and Blank is the lowest. An important distinction is being made between a black and a blank level. The VSC outputs a separate level for blank in order to provide a true blank level (and not black).

There are two distinct blank inputs to the VSC. BLANK is the CRTC's horizontal and vertical retrace period input which causes a blank output level to the display. CBLANK is an attribute input to selectively blank a character cell by forcing the video information for the particular character cell period to switch to the selected background color level.

Video information has several sources of inputs. The background color is determined by a separate pin input and the choice is either a black or white background. However, from Table 1, it can be noticed that with judicious use of the REV (Reverse Video) input, a grey background can also be selected with the background switching either to a black or white whenever Foreground SHIFT (same as highlight) is activated. Therefore, a single background select input is deemed sufficient to allow any combination of specified background color.

Foreground and video sums can be modified depending on the combination of background, foreground shift, and reverse inputs. These video inputs can be applied to Foreground to obtain any desired effect.

Figure 12. Am8152/8153 Video Dot Generator

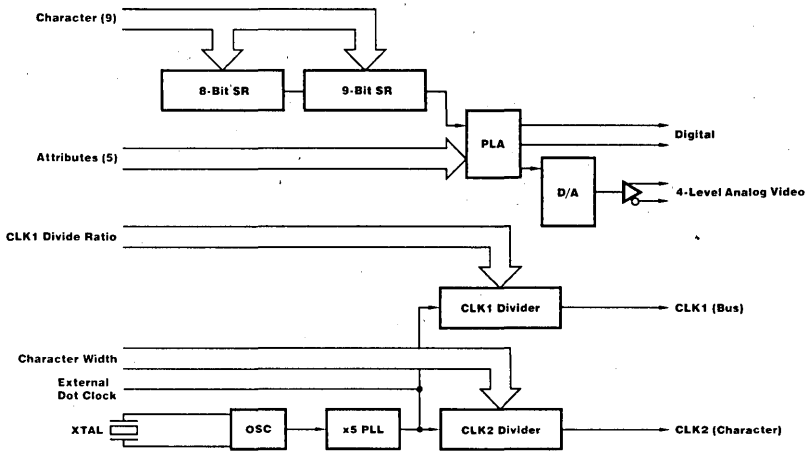
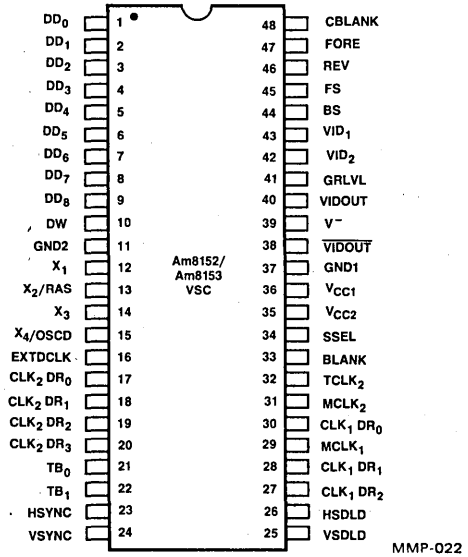


Figure 13. VSC Pinout



INTERFACE SIGNAL DESCRIPTION

(TTL compatible unless otherwise indicated)

V_{CC} 1 +5V TTL supply

V_{CC} 2 +5V ECL/Analog supply

GND 1 TTL ground

GND 2 TTL ground

V− 0V Analog Supply for Am8152, −5.2V ECL supply for Am8153

$MCLK_1$ **Clock 1, Output, Non-TTL Compatible**
$MCLK_1$ is a system clock with a 50% duty cycle. It drives the Am8052 horizontal and vertical timing circuitry, and DMA operations; $MCLK_1$ is divided down from the internal dot clock frequency according to CLK_1 DIVR (CLK_1 Divide Ratio).

CLK_1 DIVR **CLK_1 Divide Ratio, Inputs**
CLK_1 DIVR are three inputs which control the CLK_1 divider to divide by two, four, . . . , to sixteen.

X_1, X_2/RES **Inputs**
X_1 and X_2/RES are the external crystal inputs when the on-chip oscillator of the VSC is being used. The external crystal frequency is multiplied by five to produce the on-chip dot clock. If the external dot clock option is used, then X_1 should be tied LOW, and X_2/RES may be used as a reset input, to synchronize multiple VSCs. Note that the reset signal should be synchronous to the external dot clock.

X_3, X_4/OSCD **Inputs, Non-TTL Compatible**
X_3 and X_4/OSCD are used as inputs to the on-chip voltage-controlled oscillator, when the on-chip oscillator of the VSC is being used. X_3 and X_4/OSCD should be connected by an appropriate capacitor. X_4/OSCD should be connected to ground via an appropriate resistor. If the external dot clock option is used, X_3 should be tied LOW and X_4/OSCD should be tied HIGH.

CLK_2 DIVR **CLK_2 Divide Ratio, Inputs**
CLK_2 DIVR are four inputs which control an internal divider to divide the dot clock frequency by a value of two, three, . . . , or seventeen. CLK_2 DIVR specifies the width of the dot data (width of parallel-to-serial shift register).

$MCLK_2$ **Output, Non-TTL Compatible**
$MCLK_2$ is a character display clock. It is intended to control the character code and attribute data output rate from the appropriate Am8052 ports.

$TCLK_2$ **Output**
$TCLK_2$ is a TTL compatible version of $MCLK_2$.

VSYNC **Vertical Sync, Input**
VSYNC is an input which must be synchronous to either $MCLK_1$ (if SSEL is HIGH) or $MCLK_2$, (if SSEL is LOW).

VSDLD **Vertical Sync Delayed, Output**
VSLD is the delayed output of VSYNC, synchronous to $MCLK_1$, (if SSEL is HIGH) or $MCLK_2$, (if SSEL is LOW).

HSYNC **Horizontal Sync, Input**
HSYNC is an input which must be synchronous to either $MCLK_1$, (if SSEL is HIGH) or $MCLK_2$, (if SSEL is LOW).

HSDLD **Horizontal Sync Delayed, Output**
HSLD is the delayed output of HSYNC, synchronous to $MCLK_1$, (if SSEL is HIGH) or $MCLK_2$, (if SSEL is LOW).

SSEL **Sync Select, Input**
The sync select line determines if the VSYNC, HSYNC and Blank are synchronized to the $MCLK_1$ or $TCLK_2$ signals. A HIGH on SSEL selects $MCLK_1$ and a LOW selects $TCLK_2$/$MCLK_2$ as the synchronizing clocks. SSEL also determines if CLK_2 will be resynchronized to CLK_1 during blanking. (SSEL HIGH = CLK_2 resynchronization.)

BLANK **Input**
BLANK is an input normally synchronous to $MCLK_2$, although it may be synchronous to $MCLK_1$ in proportional space applications. The active pulse width of BLANK usually overlaps the active to inactive waveforms of HSYNC and VSYNC, and also the active to inactive portion of VSYNC. BLANK may become inactive sooner than the HSYNC inactive edge, particularly when the HSYNC input is specified to be a horizontal drive.

$TCLK_2$/$MCLK_2$ output may be optionally forced to synchronize to the $MCLK_1$ clock ($TCLK_2$/$MCLK_2$ output is equal to $MCLK_1$) while BLANK is active. When BLANK goes inactive, the rising edges of $MCLK_1$ and $TCLK_2$/$MCLK_2$ must be synchronized in order to prevent "dot walk" in proportional space applications.

Blank active also forces the video output level to "blank" irrespective of DD, FORE and other inputs.

CBLANK **Character Blank, Input**
CBLANK forces video output levels (VID_1 and VID_2) to switch to the background color level. It is used for blinking foreground information.

FORE **Foreground Video, Input**
FORE Video input is "or'ed" with the dot data output by the parallel-to-serial shift register. Fore is an auxiliary input allowing the user to mix video dot data with dot data to achieve a similar output color level (i.e., underlines).

REV **Reverse, Input**
REV input causes the foreground color levels to be transposed with the background color level for the total character period (including any trailing blank).

FS **Foreground Shift, Input**
FS input causes the shift in the video output levels to produce a highlight effect.

$TB_{0,1}$ **Trailing Blanks, Inputs (2)**
TB concatenates "blank" video dots to the tail end of the dot data contained in the parallel-to-serial shift register. TB can be specified to concatenate 0, 1, 2 or 3 dots. The TB value is also added to the CLK_2 DIVR value to obtain the total dot length of $TCLK_2$/$MCLK_2$.

$DD_0 - DD_8$ **Dot Data, Inputs (9)**
DD inputs accept parallel character dot matrix information for serial conversion for video output. DD data is accepted at the $TCLK_2$/$MCLK_2$ clock rate.

BS **Background Select, Input**
BS input specifies the color level of the background video. This input is overridden by BLANK active.

VIDOUT, $\overline{\text{VIDOUT}}$ **Analog, Outputs, Non-TTL Compatible**
VIDOUT, $\overline{\text{VIDOUT}}$ outputs in a differential mode the composite blank, and video dot levels into a nominal 75Ω load impedance from switched current sources.

$VID_{1,2}$ **Video Digital, Outputs (ECL Compatible with Am8153, TTL Compatible with Am8152)**
VID_1 and VID_2 are digitally encoded outputs of the video out. VID_1 is the LSB. Encoding is as follows:

	VID_2	VID_1
Blank Level	0	0
Black	0	1
Grey	1	0
White	1	1

GRLVL **Grey Level, Input, Non-TTL Compatible**
GRLVL input adjusts the current level output, via the VIDOUT and $\overline{\text{VIDOUT}}$ outputs, of the "grey" video level. The input is a voltage source derived from a resistor divider network. The voltage source can have any value between 0 and +1V.

DW **Double Width, Input**
DW, when active high, causes the dot clock supplied to the $TCLK_2$/$MCLK_2$ clock divider to be divided by two. It is used to facilitate doubling the width of a character cell matrix in the horizontal direction. The trailing blank information is also widened during a double width character.

EXTDCLK **External Dot Clock, Input (ECL Compatible with Am8153, TTL Compatible with Am8152)**
EXTDCLK is an external dot clock which serves as a reference clock for the internal VSC logic. It synchronizes the VSCs when used in a multiple configuration.

AmZ8060 FIFO

Buffer Unit and FIFO Expander

DISTINCTIVE CHARACTERISTICS

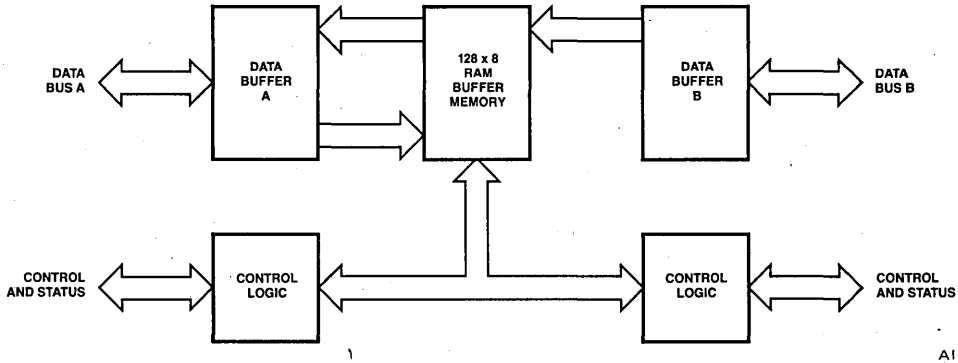
- Bidirectional, asynchronous data transfer capability
- Large 128-bit-by-8-bit buffer memory
- Two-wire, interlocked handshake protocol
- 3-state data outputs
- Wire-ORing of empty and full outputs for sensing of multiple-unit buffers
- Connects any number of FIFOs in series to form buffer of any desired length
- Connects any number of FIFOs in parallel to form buffer of any desired width

GENERAL DESCRIPTION

The AmZ8060* First-In, First-Out (FIFO) buffer unit consists of a 128-bit-by-8-bit memory, bidirectional data transfer and handshake logic.The structure of the FIFO unit is similar to that of other available buffer units. FIFO is a general-purpose unit; its handshake logic is compatible with that of other members of the AmZ8000 family.

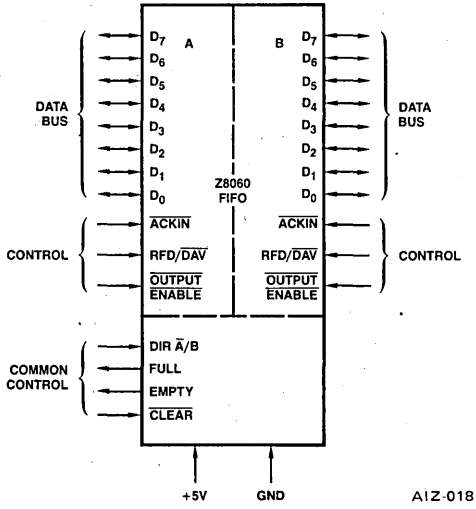
FIFOs can be cascaded end-to-end without limit to form a parallel 8-bit buffer of any desired length (in 128-byte increments). Any number of single- or multiple-unit FIFO serial buffers can be connected in parallel to form buffers of any desired width (in 8-bit increments).

FIFO BLOCK DIAGRAM



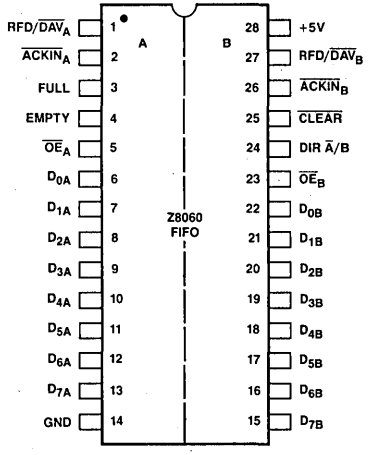
AIZ-014

LOGIC SYMBOL



AIZ-018

CONNECTION DIAGRAM – Top View
D-28, P-28



Note: Pin 1 is marked for orientation.

AIZ-019

*Z8000 is a trademark of Zilog, Inc.

FUNCTIONAL DESCRIPTION

Interlocked 2-Wire Handshake

In interlocked 2-wire handshake operation, the action of FIFO must be acknowledged by the other half of the handshake before the next action can occur. In an Output Handshake mode, the FIFO indicates that new data is available only after the external device has indicated that it is ready for the data. In an Input Handshake mode, the FIFO does not indicate that it is ready for new data until the data source indicates that the previous byte of the data is no longer available, thereby acknowledging the acceptance of the last byte. This control feature allows the FIFO, with no external logic, to directly interface with the port of any CPU in the Z8 Family – a CIO, a UPC, an FIO, or another FIFO. The timing for the input and output handshake operations is shown in Figures 1 and 2, respectively.

Resetting or Clearing the FIFO

The $\overline{\text{CLEAR}}$ input is used to initialize and clear the FIFO. A Low level on this input clears all data from the FIFO, allows the EMPTY output to go High and forces both outputs RFD/$\overline{\text{DAV}}_A$ and RFD/$\overline{\text{DAV}}_B$ High. A High level on $\overline{\text{CLEAR}}$ allows the data to transfer through the FIFO.

Bidirectional Transfer Control

The FIFO has bidirectional data transfer capability under control of the DIR $\overline{\text{A}}$/B input. When DIR $\overline{\text{A}}$/B is set Low, port A becomes input handshake and port B becomes output handshake; data transfers are then made from port A to port B. Setting DIR $\overline{\text{A}}$/B High reverses the handshake assignments and the direction of transfer. This bidirectional control is illustrated in Table 1.

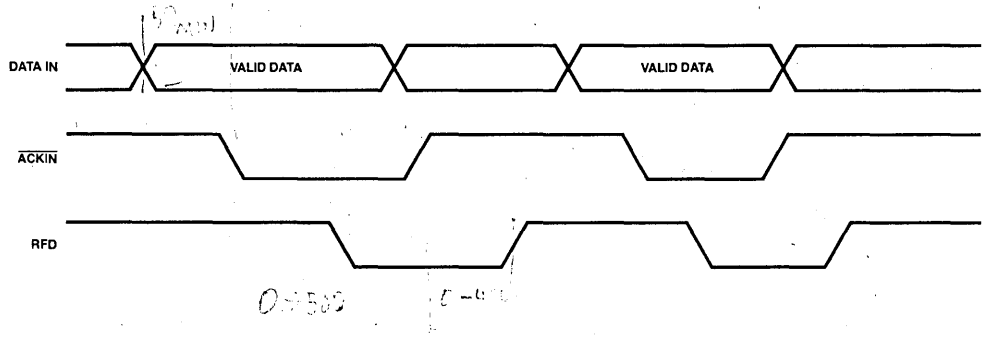
TABLE 1. BIDIRECTIONAL CONTROL FUNCTION TABLE

DIR $\overline{\text{A}}$/B	Port A Handshake	Port B Handshake	Transfer
0	Input	Output	A to B
1	Output	Input	B to A

The FIFO buffer must be empty before the direction of transfer is changed; otherwise, the results of the change will be unpredictable. If FIFO status is unknown when a transfer direction change is to be made, the recommended procedure is:

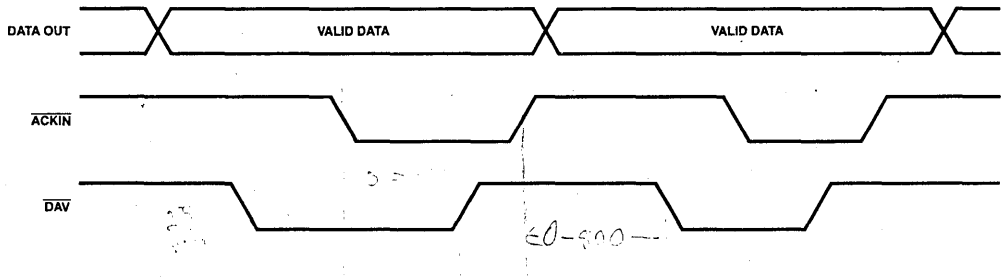
(1) Force and hold $\overline{\text{CLEAR}}$ Low
(2) Set DIR $\overline{\text{A}}$/B to the level required for the desired direction
(3) Force $\overline{\text{CLEAR}}$ High

Figure 1. Two-Wire Interlocked Handshake Timing (Input)



AIZ-015

Figure 2. Two-Wire Interlocked Handshake Timing (Output)



AIZ-016

Empty and Full Operation

The EMPTY and FULL output lines can be wire-ORed with the EMPTY and FULL lines of other FIFOs and FIOs. This capability enables the user to determine the empty/full status of a buffer consisting of multiple FIFOs, FIOs, or a combination of both. Table 2 shows the various states of EMPTY and FULL.

TABLE 2. SIGNALS EMPTY AND FULL OPERATION TABLE

Number of Bytes in FIFO	Empty	Full
0	High	Low
1-127	Low	Low
128	Low	High

Interconnection Example

A simplified block diagram showing the manner in which FIFOs can be interconnected to extend a FIO buffer is shown in Figure 3.

Output Enable Operation

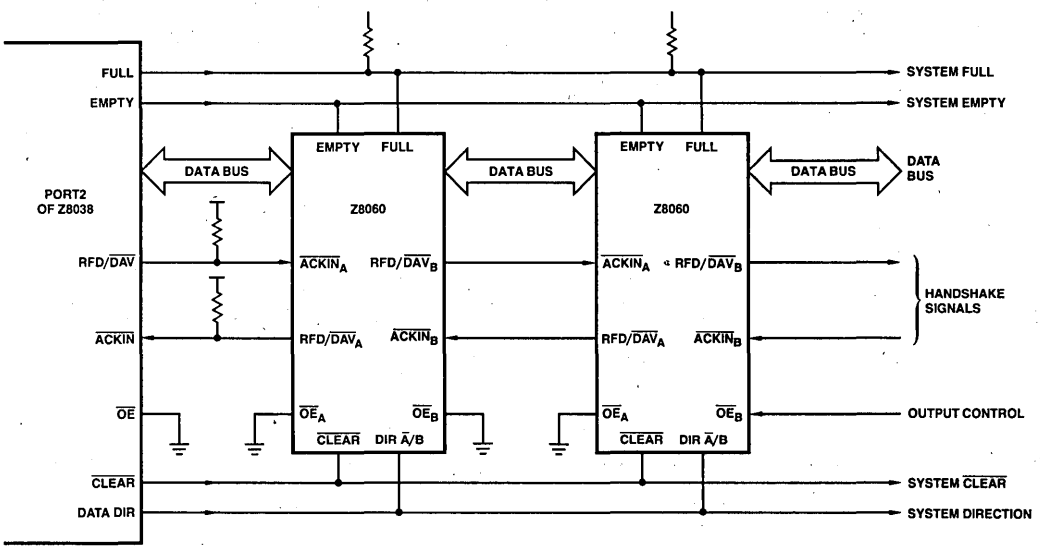
The FIFO provides a separate Output Enable ($\overline{OE}$) signal for each port of the buffer. An $\overline{OE}$ output is valid only when its port is in the Output Handshake mode. The control of this output function is shown in Table 3. Signal $\overline{OE}$ operates with lines DIR $\overline{A}$/B. A High on a valid $\overline{OE}$ line 3-states its port's data bus but does not affect the handshake operation. A Low level on a valid $\overline{OE}$ enables the data bus outputs if its port is in the Output Handshake mode. Note that the handshake operation is unaffected by the output enable pin.

TABLE 3. OUTPUT CONTROL FUNCTION TABLE

DIR $\overline{A}$/B	$\overline{OE}_A$	$\overline{OE}_B$	Function
0	X	0	Disable Port A Output Enable Port B Output
0	X	1	Disable Port A Output Disable Port B Output
1	0	X	Enable Port A Output Disable Port B Output
1	1	X	Disable Port A Output Disable Port B Output

Note: X = Don't Care.

Figure 3. Typical Interconnection (Simplified Diagram)



AIZ-017

INTERFACE SIGNAL DESCRIPTION

V_{CC}: +5V Power Supply

GND: Ground

$\overline{ACKIN}$ Acknowledge Input (input, active Low). This line signals the FIFO that output data has been received by peripherals or that input data is valid.

$\overline{CLEAR}$ Clear Buffer (input, active Low). When set to Low, this line causes all data to be cleared from the FIFO buffer.

D_0-D_7 Data Bus (inputs/outputs, bidirectional). These bidirectional lines are used by the FIFO to receive and to transmit data.

DIR $\overline{A}$/B Direction Input $\overline{A}$/B (input, two control states). A High on this line signals that input data is to be received at port B. A Low on this line signals that input data is to be received at port A.

EMPTY Buffer Status (output, active High, open-drain). A High on this line indicates that the FIFO buffer is empty.

FULL Buffer Status (output, active High, open-drain). A High on this line indicates that the FIFO buffer is full.

$\overline{OE}_A$, $\overline{OE}_B$ Output Enable A, Output Enable B (inputs, active Low). When Low, $\overline{OE}_A$ enables the bus drivers for port A; when High, $\overline{OE}_A$ causes the bus drivers to float to a high-impedance level. Input $\overline{OE}_B$ controls the bus drivers for port B in the same manner as $\overline{OE}_A$ controls those for port A.

RFD/$\overline{DAV}$ Ready-for-Data/Data Available (outputs RFD, active High, $\overline{DAV}$ active Low). RFD, when High, signals to the peripherals involved that the FIFO is ready to receive data. $\overline{DAV}$, when Low, signals to the peripherals involved that FIFO has data available to send.

ORDERING INFORMATION

Example Z8060 A D C B

Basic Device

Speed
Blank = 4MHz
A = 6MHz

Package Type
D = Hermetic DIP
P = Molded Plastic

Temperature Range
C = 0 to +70°C
I = −40 to +85°C
M = −55 to +125°C

Burn-in

MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65 to +150°C
Voltage to any Pin Relative to V_{SS}	−0.5 to +7.0V
Power Dissipation	1.8W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

(over which the DC, switching and functional specification apply)

	4MHz	6MHz
	Z8060	**Z8060A**
Commercial Operating Range T_A = 0 to +70°C V_{CC} = 5V ± 5%	Z8060DC Z8060PC	
Industrial Operating Range T_A = −40 to +85°C V_{CC} = 5V ± 5%	Z8060DI	
Military Operating Range T_A = −55 to +125°C V_{CC} = 5V ± 10%	Z8060DMB	

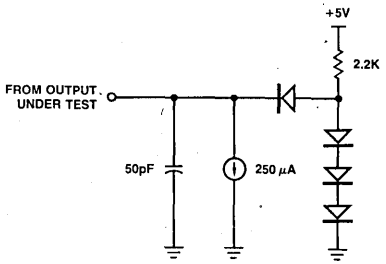
Notes: T_A denotes ambient temperature.
Add suffix B to indicate burn-in requirement.

Standard Test Conditions

The characteristics 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:

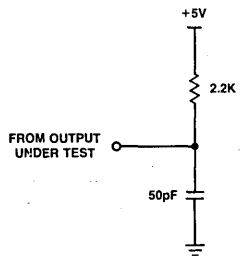
$+4.75V \leq V_{CC} \leq +5.25V$
GND = 0V
$0°C \leq T_A \leq +70°C$

Standard Test Load



AIZ-042

Open-Drain Test Load



AIZ-043

LECTRICAL CHARACTERISTICS over operating range unless otherwise specified (Note 1)

arameter	Description	Test Conditions	Min	Typ	Max	Units
V_{IL}	Input LOW Voltage		−0.3		+0.8	Volts
V_{IH}	Input HIGH Voltage	Standard Temp	2.0		V_{CC} + 0.3	Volts
		Military Temp	2.2			
V_{OL}	Output LOW Voltage	I_{OL} = 3.2mA			0.5	Volts
		I_{OL} = 2.0mA			0.4	
V_{OH}	Output HIGH Voltage	I_{OH} = −250μA	2.4			Volts
I_{OZL}	Output Leakage Current	V_{OUT} = 0.4V			10	μA
I_{OZH}	Output Leakage Current	V_{OUT} = V_{CC}			10	μA
I_I	Input Leakage Current				±10	μA
C_{IN}	Input Capacitance	Unmeasured pins returned to ground. f = 1MHz over specified temperature range.			10	pF
$C_{I/O}$	I/O Capacitance				20	pF
C_{OUT}	Output Capacitance				15	pF
I_{CC}	Power Supply Current	V_{CC} = MAX			250	mA

lote 1. See table for operating range. Typical conditions apply at T_A = 25°C, V_{CC} = 5.0V.

IFO 2-WIRE HANDSHAKE TIMING

iming for 2-wire interlocked handshake operation is shown in igure 1. The symbol, description and values for the numbered arameters (Figure 1) are given in AC Characteristics.

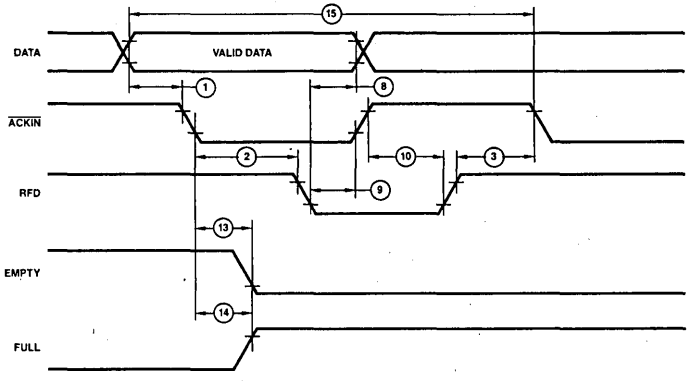
AC CHARACTERISTICS

			4MHz		6MHz		
Number	Symbol	Parameter	Min	Max	Min	Max	Units
1	TsDI(ACK)	Data Input to $\overline{ACKIN}$ ↓ to Setup Time	50				ns
2	TdACKf(RFD)	$\overline{ACKIN}$ ↓ to RFD ↓ Delay	0	500			ns
3	TdRFDr(ACK)	RFD ↑ to $\overline{ACKIN}$ ↓ Delay	0				ns
4	TsDO(DAV)	Data Out to $\overline{DAV}$ ↓ Setup Time	25				ns
5	TdDAVf(ACK)	DAV ↓ to $\overline{ACKIN}$ ↓ Delay	0				ns
6	ThDO(ACK)	Data Out to $\overline{ACKIN}$ ↑ Hold Time	50				ns
7	TdACK(DAV)	$\overline{ACKIN}$ ↓ to $\overline{DAV}$ ↑ Delay	0	500			ns
8	ThDI(RFD)	Data Input to RFD ↓ Hold Time	0				ns
9	TdRFDf(ACK)	RFD ↓ to $\overline{ACKIN}$ ↑ Delay	0				ns
10	TdACKr(RFD)	$\overline{ACKIN}$ ↑ to RFD ↑ Delay	0	400			ns
11	TdDAVr(ACK)	DAV ↑ to $\overline{ACKIN}$ ↑	0				ns
12	TdACKr(DAV)	$\overline{ACKIN}$ ↑ to $\overline{DAV}$ ↓	0	800			ns
13	TdACKINf(EMPTY)	(Input) $\overline{ACKIN}$ ↓ to EMPTY ↓ Delay		600			ns
		(Output) $\overline{ACKIN}$ ↓ to EMPTY ↑ Delay					
14	TdACKINf(FULL)	(Input) $\overline{ACKIN}$ ↓ to FULL ↑ Delay		600			ns
		(Output) $\overline{ACKIN}$ ↓ to FULL ↓ Delay					
15	ACKIN Clock Rate	(Input)		1.0			MHz
16	TdACKINf($\overline{DAV}$f)	(Bubble Time)		800			ns
17	TwCLR	Width of Clear to Reset FIFO	700				ns
18	TdOE(DO)	$\overline{OE}$ ↓ to Data Bus Driven	0	150			ns
19	TdOE(DRZ)	$\overline{OE}$ ↑ to Data Bus Float		100			ns

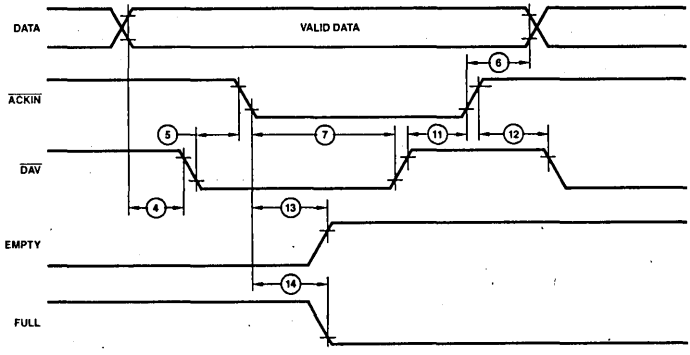
Note: All timing references assume 2.0V for a logic 1 and 0.8V for a logic 0.

WAVEFORMS

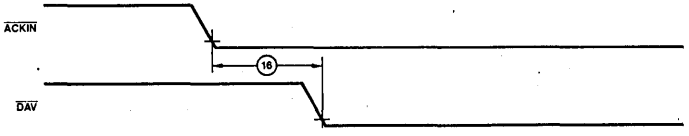
INPUT TIMING



OUTPUT TIMING



ACKNOWLEDGE INPUT DATA TO DATA AVAILABLE TIME (BUBBLE TIME)



OUTPUT ENABLE AND CLEAR

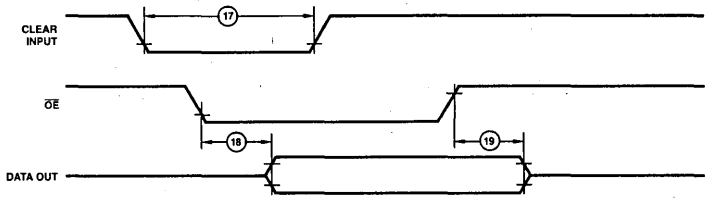


Figure 4. Timing Diagrams

AmZ8065/Am9520/Am9521

Burst Error Processor

DISTINCTIVE CHARACTERISTICS

- **Provides for detection and correction of burst errors.**
  Detects errors in serial data up to 585K bits long.
  Allows correction of error bursts of up to 12 bits.
- **High-Speed Operation.**
  Effective data rates up to 20 Mbits/second for AmZ8065/Am9520/Am9521 and 30 Mbits/second for -1 versions.
  Fast enough for high-performance hard and soft disk systems.
- **Selectable Industry-Standard Polynomials.**
  35- and 32-bit polynomials on Am9521.
  AmZ8065/Am9520 additionally has popular IBM 56- and 48-bit versions.
- **Three correction algorithms provide flexibility.**
  Full-period clock-around method for conforming to current practices.
  Chinese remainder theorem reduces correction time by orders of magnitude.
  Reciprocal polynomial makes correction possible with 48-bit code.
- **Designed for use in both microprogrammed and microprocessor disk controller systems.**
  Device complements both AmZ8000 and Am2900 microprocessor families and can also be used with other microprocessors.

GENERAL DESCRIPTION

The Burst Error Processor (BEP) provides for error detection and correction for high-performance disk systems and other systems in which high-speed serial data transfer takes place. As data density and transfer rates increase in both hard and floppy disks and other storage media, error detection and correction become increasingly important. The BEP is an LSI circuit that facilitates the most common error detection and correction schemes accommodating data streams of up to 585K bits at up to 20M bits/second effective data rate.

The BEP provides a choice of four standard polynomials, including the popular 56- and 48-bit versions, to satisfy a broad range of applications. The device divides the data stream by the selected polynomial using the rules of algebra in polynomial fields. The resulting remainder is the check word which is then appended to the data for writing on the disk as a record. When the record is read back, the BEP computes the syndrome for data validation. If an error is detected, the location and pattern of this burst in the data stream is determined for corrections.

TYPICAL APPLICATION DIAGRAM

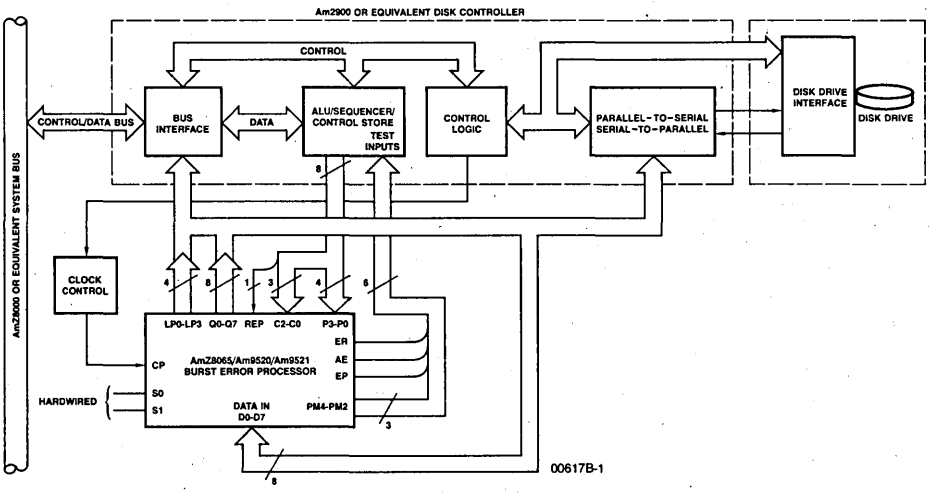


Figure 1.
AmZ8065/Am9520/Am9521
Burst Error Processor

BLOCK DIAGRAM

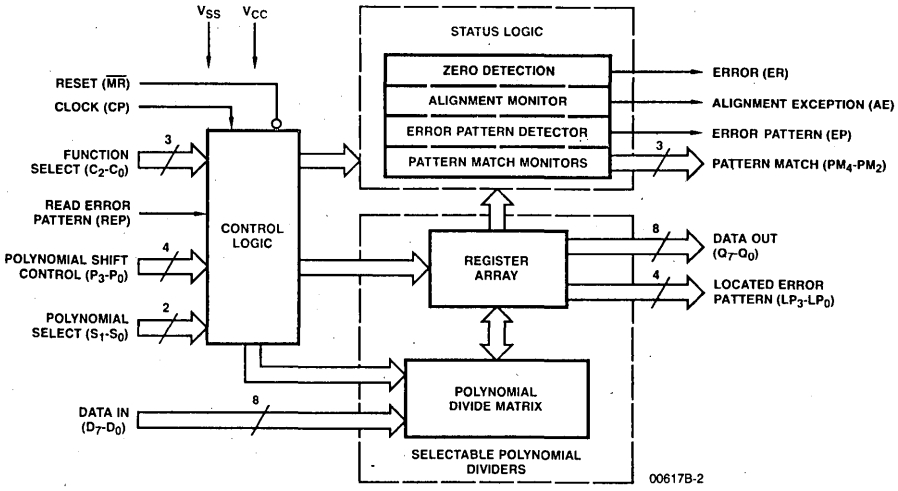


Figure 2. AmZ8065/Am9520/Am9521 Burst Error Processor

FUNCTIONAL DESCRIPTION

Figure 2 is a conceptual block diagram. It consists of four major sections – Register Array, Polynomial Divide Matrix, Status Logic and Control Logic.

Register Array

This section consists of 56 flip-flops used for check bit computation during write operation, syndrome computation during read operation and error pattern extraction during error correction operation. In general, the Polynomial Divide Matrix provides the bit patterns required for the Register Array. The combination of Register Array and Polynominal Divide Matrix mechanizes the familiar serial form of feedback shift register arrangement in an 8-bit parallel form. The Q_0-Q_7 outputs of the Am9520 are obtained from the Register Array. When correction operations are complete, the error pattern is available on 12 outputs: eight bits on the Q_0-Q_7 outputs and remaining four bits on the LP0-LP3 outputs. The Read Error Pattern (REP) input must be HIGH for the error pattern to be available. The Control Logic generates Clock signals for the Register Array.

Status Logic

This section monitors the register arrays to generate the various error detection outputs of the BEP, including ER, AE, PM_2, PM_3, PM_4 and EP.

Polynomial Divide Matrix

Polynomial Divide Matrix is the heart of the BEP. The Control Logic decodes the Polynomial Select (S_0-S_1) and Function Select (C_0-C_2) inputs to generate the necessary gating signals to the matrix. The matrix establishes connections such that a byte of data presented on the D_0-D_7 inputs will be suitably divided by the selected generator polynomial. Four different polynomials are selected by logic levels on the S_0-S_1 inputs (Table 1).

These devices can be used in three fundamentally different types of operations: write, read and correct. The various functions are selected by the C_0-C_2 control inputs.

Write

While data is being written on the disk, the BEP is in the Compute Check Bits mode looking at the data bytes without affecting the flow of data to the disk. After the last data byte, the BEP is switched into the Write Check Bits function outputting the 4, 5, 6 or 7 check bytes. This is the additional information appended to the data stream that allows the detection and correction of possible read errors.

Read

When information (data plus appended check bits) is being read, the BEP must be in either Read Normal mode or Read High Speed mode. These modes differ only in the correction algorithm that will be used if an error has occurred. In both modes parallel bytes are read into the device. After the last information byte has been entered the ER output is checked. If it is LOW, there is no error, if it is HIGH, there is an error.

Correction

After the read operation, the syndrome held in the register array contains all the information necessary to find the error location and the error pattern, i.e., to allow error correction. In the Correct Normal mode, the error location is found by counting the number of clock pulses required to make the EP output go HIGH. The error pattern is then available on the LP_0-LP_3 and Q_0-Q_7 outputs and can be used to Exclusive OR with data.

In Correct High Speed mode, the error location is also found by counting clock pulses, but they are routed in succession to the different sections of the register array. This results in slightly more complicated but substantially faster operation.

TABLE 1. POLYNOMIALS

Polynomial	Number of Check Bits	Period (Bits)	Correctable Burst Error Length (Bits)
$(X^{22} + 1) \cdot (X^{11} + X^7 + X^6 + X + 1) \cdot (X^{12} + X^{11} + X^{10} + \ldots + X + 1) \cdot (X^{11} + X^9 + X^7 + X^6 + X^5 + X + 1)$	56	585,442	11
$(X^{21} + 1) \cdot (X^{11} + X^2 + 1)$	32	42,987	11
$(X^{23} + 1) \cdot (X^{12} + X^{11} + X^8 + X^7 + X^3 + X + 1)$	35	94,185	12
$(X^{13} + 1) \cdot (X^{35} + X^{23} + X^8 + X^2 + 1)$	48	$13 \cdot (2^{35}-1)$	7

DETAILED FUNCTIONAL DESCRIPTION

Compute Check Bits

The check bits to be appended to the data are computed using this function. The S_0-S_1 inputs select the desired polynomial. The Polynomial Matrix will be configured such that the generator polynomial is in the expanded form. The expanded form of a polynomial is obtained by multiplying out its factors and combining proper terms using modulo-2 arithmetic. Assume that the 32-bit polynomial is selected: The factored form of the 32-bit polynomial in Table 1 is $(X^{21} + 1)\ (X^{11} + X^2 + 1)$. The corresponding expanded form is $X^{32} + X^{23} + X^{21} + X^{11} + X^2 + 1$.

The sequence of events to compute the check bits is as follows:

1. The CP input is in quiescent HIGH state.
2. Initialize by activating the $\overline{MR}$ input LOW and return it to HIGH.
3. Through appropriate logic levels on the S_0-S_1 inputs specify the desired polynomial. Also, select Compute Check Bits code through the C_0-C_2 inputs.
4. Establish a byte of data on the D_0-D_7 inputs.
5. Make CP input LOW and then HIGH. See timing diagram for detailed timing specifications.
6. Keep repeating from step 4 until all data bytes are entered.

Write Check Bits

In Compute Check Bits mode the polynomial matrix and the Register Array are mechanizing a feedback shift register configuration. However, when write Check Bit Code is established on the C_0-C_2 inputs, the feedback paths are disabled such that the register array will behave as a simple shift register. When the last data byte is entered in the Compute Check Bits mode, the register array holds the check bits. These check bits will be available on the Q_0-Q_7 outputs, one byte at a time. The sequence of events to obtain the check bits is as follows:

1. The CP is in quiescent HIGH state.
2. Estaolish appropriate code on the S_0-S_1 inputs. This code must be the same as that used for Compute Check Bits function.
3. Establish Write Check Bits code on the C_0-C_2 inputs.
4. After a propagation delay the Q_0-Q_7 outputs will contain the first check byte.
5. Make CP input LOW and then HIGH. The next check byte will be available on the Q_0-Q_7 outputs.
6. Keep repeating from step 5 until all check bytes that correspond to the selected polynomial are read out.

Read Normal

Two methodologies are available for error correction with these devices: (a) Full period clock around (normal method) and (b) Chinese remainder theorem (high-speed method). The Read Normal function must be used for reading data from the disk if the normal method is used for error correction. When Read Normal is selected, the Polynomial Matrix establishes the polynomial in the expanded form. In this mode, the input stream consisting of data and check bytes is divided by the selected polynomial to obtain the syndrome. If the resulting syndrome is not zero, an error is detected. The ER output indicates whether the syndrome is zero or not. HIGH on the ER output indicates non-zero syndrome.

The sequence of events for Read Normal is as follows:

1. The CP input is in quiescent HIGH state.
2. Initialize the Am9520 by activating the $\overline{MR}$ input LOW and then return it to HIGH.
3. Establish proper code on the S_0-S_1 inputs. The polynomial selected for the read operation must be the same as the one originally used for generating the check bits.
4. Establish Read Normal code on the C_0-C_2 inputs.
5. Present a byte of information read from the disk on the D_0-D_7 inputs.
6. Make the CP input LOW and then HIGH.
7. Keep repeating from step 5 until the last check byte read from the disk is processed.
8. After entering last check byte, test the ER output. HIGH on this output is indicative of an error and LOW means no error detected.

Read High Speed

This function must be used for reading data if the Chinese remainder theorem method is to be used for error correction. In general, the Chinese remainder method accomplishes error correction in fewer clock cycles than the normal method. This method of correction, however, is not available for the 48-bit polynomial due to the nature of the factors that make up this polynomial. As explained later, the reciprocal polynomial technique is used for error correction when the 48-bit polynomial is selected.

The only difference between Read Normal and Read-High Speed Modes is as follows: In the Read Normal, the input stream is divided by the expanded version of the polynomial, whereas in the Read High-Speed Mode, the input stream is simultaneously divided by all factors of the polynomial. Thus, the high-speed mode results in as many syndromes as the number of factors of the polynomial. If all syndromes are zero after entering the last check byte, the ER output will be LOW indicating error-free operation. If there was an error the ER will be HIGH.

The sequence of events in this mode are as follows:

1. The CP input is in its quiescent HIGH state.
2. Specify the polynomial on he S_0-S_1 input. This must obviously be the same polynomial that generated the check bits originally.

3. Specify Read High-Speed function on the C_0-C_2 inputs.
4. Initialize by activating the $\overline{MR}$ input LOW and then return it to HIGH.
5. Present a byte read from the disk on the D_0-D_7 inputs.
6. Make the CP input LOW and then HIGH.
7. Keep repeating from step 5 until all data and check bytes are entered.
8. Test the ER output after entering the last check byte. HIGH on this output is indicative of an error and LOW signifies no error.

Correct Normal

The syndrome obtained from Read Normal operation is manipulated to extract the error pattern as well as its location using the Correct Normal function. Of the four polynomials listed in Table 1, the 48-bit version requires a separate explanation. For all cases except the 48-bit version, the polynomial is established in the expanded form.

In the Correct Normal, the syndrome is repeatedly divided by the polynomial until the error pattern is located. This division is accomplished by repeated clocking while ignoring the D_0-D_7 inputs. HIGH on the EP output signifies that the error pattern is found. The error pattern is always characterized by a known number of consecutive zeros at specified register array locations. The exact number of zeros and their location is a function of the select polynomial. The status logic detects this unique combination to generate the EP output. The number of clock cycles needed to locate the error pattern is a measure of the error location. If the number of clock cycles has exceeded the natural period of the selected polynomial without finding the error pattern, then an uncorrectable error has occurred. The AE output must also be considered in the Correct Normal mode of operation.

The polynomial matrix is an 8-bit parallel mechanization of the familiar serial polynomial division scheme. Because of this, there are certain conditions under which the error pattern will not line up automatically. The Status Logic also monitors this condition. When such an alignment exception is detected, the AE output of the device goes HIGH.

Internally, the device switches automatically into the one-bit shift mode. Let R_1 be the number of clock cycles for the AE output to go HIGH. Let R_2 be the number of clock cycles from AE output going HIGH to EP output going HIGH. Let N be the natural period of the selected polynomial. Then, $N \cdot K - 8R_1 - R_2$ is the first bit in the error burst counting from the last check bit of the record, where K is the smallest positive integer to make this expression positive. If there is no alignment exception, then $R_2 = 0$. See Table 1 for periods of the polynomials.

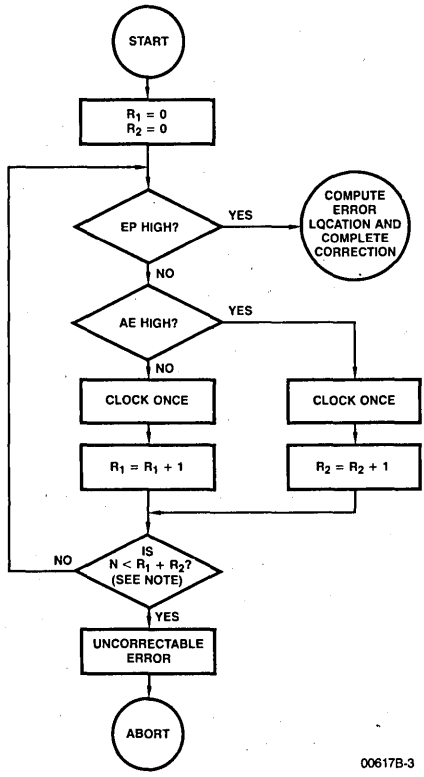
The error pattern provided is used externally to correct the error. The error pattern is available on the Q_0-Q_7 and LP_0-LP_3 outputs when the REP input is HIGH. Q_7 corresponds to the first bit in error. When an error pattern bit is HIGH, then the corresponding bit in the data stream must be complemented to accomplish correction.

The Correct Normal discussed so far can be summarized by the following sequence of events.

1. The CP input is in the quiescent HIGH state.
2. The ER output is HIGH indicating error from the previous Read Normal operation.
3. Select appropriate polynomial (S_0-S_1).
4. Select Correct Normal Code (C_0-C_2).
5. Let R_1 and R_2 be two external counters both initialized to zero.
6. Check if the EP output is HIGH. If HIGH, the error pattern is found. The error location is given by $L = N \cdot K - (8R_1 + R_2)$ except for the 35-bit polynomial where $L = N \cdot K - (8R_1 + R_2 + 5)$. ($R_2$ is always zero for this case.)
7. If the EP output is LOW, test the AE output. If the AE output is HIGH, make the CP input LOW and then HIGH. Increment R_2. If the AE output is LOW, make the CP input LOW and then HIGH. Increment R_1 instead.
8. If $R_1 + R_2$ is greater than N (N is the natural period of the selected polynomial) then an uncorrectable error ocurred. Abort the correction process.
9. If the error is correctable repeat from step 6.

The flowchart in Figure 3 explains the correction process.

Figure 3. Flowchart



Note: For the 48-bit polynomial N = number of actual data bits +48 check bits.

Now consider the Correct Normal Mode of operation with the 48-bit polynomial. The period of the 48-bit polynomial is so large that ordinary division is not practical. In this case the Polynomial Matrix establishes the reciprocal of the expanded polynomial. If G(X) is a polynomial of degree K, then its reciprocal $G^*(X) = X^K G(1/X)$. Because of this, the syndrome obtained using Read Normal Mode with the 48-bit polynomial is not used directly for extracting the error pattern and calculating its location. Instead the reciprocal of the syndrome must be used.

The procedure for forming the reciprocal must be accomplished externally as follows: Assume that Read Normal operation using 48-bit polynomial was finished and an error was detected. Read out the syndrome using Write Check Bits function. Now reverse all these syndrome bits such that the previously most significant bit becomes the least significant bit and vice versa. The result is the reciprocal syndrome.

Now load this reciprocal syndrome into the device using the Load function (see description of Load). Once the reciprocal syndrome is loaded, Correct Normal function is established on the C_0-C_2 inputs and the correction process can be started. The actual correction process is exactly the same as before except the error location in this case is given by $8R_1 + R_2 - 48$.

The sequence of events can be summarized as follows:

1. Read out the syndrome using the Write Check Bits function.
2. Form the reciprocal syndrome externally and enter it using the Load function.
3. The CP input is in its quiescent HIGH state.
4. Select the 48-bit polynomial on the S_0-S_1 inputs.
5. Select Correct Normal mode on the C_0-C_2 inputs.
6. Let R_1 and R_2 be two external counters initialized to zero.
7. Test EP output. If it is HIGH, error pattern has already been found. The error location is $8R_1 + R_2 - 48$.
8. If EP output is LOW, test the AE output. If AE is HIGH, make CP input LOW and then HIGH. Increment R_2. If the AE output is LOW, make the CP input LOW and then HIGH. Increment R_1.
9. If $R_1 + R_2 - 48$ is greater than the record length, the error is uncorrectable, so abort the correction process.
10. Keep repeating from step 7 until the error is located.

Correct High Speed

The maximum number of clock cycles needed to find the error pattern using the normal correction method is N where N is the period of the polynomial. Thus a polynomial with a large period may require a large number of clock cycles for error correction not acceptable in some applications. The BEP has facilities for high-speed correction using the Chinese remainder theorem method.

Let a polynomial consist of m factors with periods $P_1, P_2, \ldots P_m$. The period N of the composite polynomial is the product of the periods of the individual factors; i.e., $N = P_1 \cdot P_2 \cdot P_3 \ldots P_m$. If the Chinese remainder theorem is used for correction, the maximum number of clock cycles needed is $(P_1 + P_2 + \ldots P_m)$. This number is usually much smaller than N. Thus, the Chinese remainder theroem method is faster than the normal method for error correction.

To employ the Chinese remainder theorem method, the syndromes must be obtained first using the Read High-Speed function. This function gives as many syndromes as the number of factors in the polynomial. In other words, the register array is divided into a number of sections; each section implementing one factor of the polynomial. The first factor of every polynomial is of the form $(X^C + 1)$. This factor is sometimes called the error pattern polynomial. The Chinese remainder theorem method requires that the syndrome obtained by the error pattern polynomial be repeatedly divided by the error pattern polynomial until the error pattern is found. This is done in a fashion similar to the Correct Normal method described before. The register section corresponding to the error pattern polynomial is repeatedly clocked. The error pattern is always characterized by a known number of consecutive zeros at predetermined bit positions. (There can be alignment exceptions while finding the error pattern, but for the purpose of this explanation, assume that alignment exceptions do not occur.)

After locating the error pattern the error pattern register is prevented from clocking. Next, the register corresponding to the second factor is repeatedly clocked until it matches the error pattern and then this register is prevented from further clocking. This procedure is repeated for all remaining factors. As mentioned earlier, the P_0-P_3 inputs are provided to control clocking of the individual registers and the PM_2-PM_4 outputs are provided to indicate matching of each register with the error pattern.

Let M_1 be the number of clock cycles required to find the error pattern and M_2, M_3, etc. be the number of clock cycles required to match subsequent factors as described above. The error location can then be computed by a formula of the form:

$$L = N \cdot K - (A_1M_1 + A_2M_2 + A_3M_3 + A_4M_4 + \ldots).$$

Where A_1, A_2, etc. are predetermined constants for a given polynomial and K is the smallest integer that makes the right hand side of the equation positive. A_0, A_1 etc. are called Chinese remainder theorem coefficients. The number of coefficients equals the number of factors in the polynomial. Table 3 lists the coefficients for the polynomials. There is one additional adjustment for the 35-bit polynomial – the error location for this polynomial is computed by using the formula $L = N \cdot K - (A_1M_1 + A_2M_2 + 5)$. This modification is required because 35 bits are really five bytes with the last five bits being unused.

TABLE 2. POLYNOMIAL PERIODS

Polynomial	Period Factor 1	Period Factor 2	Period Factor 3	Period Factor 4	Composite Period (N)
56-Bit	22	13	89	23	585442
32-Bit	21	2047	–	–	42987
35-Bit	23	4095	–	–	94185

As in the normal method, every error detected may not necessarily be correctable. If the number of clock cycles to find the error pattern exceeds the period of the error pattern polynomial, or the number of clock cycles required to match a register exceeds the period of the polynomial corresponding to that register, the correction process must be aborted. Table 2 lists the applicable periods for polynomials.

TABLE 3. CHINESE REMAINDER THEOREM COEFFICIENTS

Polynomial	A_1	A_2	A_3	A_4
56-bit	452,387	2,521,904	578,864	2,647,216
32-bit	311,144	32,760	–	–
35-bit	32,760	720,728	–	–

The sequence of events is as follows:

1. The CP input is in the quiescent HIGH state. The ER output is HIGH indicating an error from the Read High Speed operations.
2. Select the polynomial using the S_0-S_1 inputs and specify Correct High Speed code on the C_0-C_2 inputs.
3. Set $P_1 = P_2 = P_3 =$ LOW, $P_0 =$ HIGH.
4. R_1 and R_2 are two external counters, initialized to zero.
5. Test the EP output. If the EP output is HIGH, error pattern is already found and $M_1 = 8R_1 + R_2$. Bring P_0 input LOW and go to step 10.

6. Establish HIGH on the P_0 input.
7. If the EP output is LOW, test the AE output. If the AE output is LOW, make the CP input LOW and then HIGH. Increment R_1. If the AE output is HIGH, make CP LOW and then HIGH, increment R_2.
8. If $R_1 + R_2$ is greater than the period of the first factor, abort the correction process; the error is not correctable.
9. If the error is correctable repeat from step 5.
10. Establish HIGH on the P_1 input.
11. M_2 is an external counter initialized to zero.
12. Test the PM_2 output. If the PM_2 output is HIGH, second factor located the matching error pattern. Bring P_1 LOW and go to step 16.
13. If the PM_2 output is LOW, make the CP input LOW and then HIGH. Increment M_2.
14. If M_2 is greater than the period of the second factor, abort the correction process; the error is not correctable.
15. If the error is correctable, repeat from step 12.

The following additional steps are performed only for the 56-bit polynomial. In case of the 32-bit or 35-bit polynomial proceed with computations for error location.

16. Establish HIGH on the P_2 input.
17. M_3 is an external counter initialized to zero.
18. Test the PM_3 output. If it is HIGH, the third factor located the matching error pattern. Bring P_2 input back LOW and go to step 22.
19. If the PM_3 output is LOW, make the CP input LOW and then HIGH. Increment M_3.
20. If M_3 is greater than the period of the third factor, abort the correction process; the error is not correctable.
21. If the error is correctable repeat from step 18.
22. Establish HIGH on the P_3 input.
23. M_4 is an external counter initialized to zero.
24. Test the PM_4 output. If it is HIGH, the matching error pattern is found by the fourth factor. Compute the error location.
25. If the PM_4 output is LOW make CP input LOW and then HIGH. Increment M_4. If M_4 is greater than the period of the fourth factor, abort the correction process; the error is not correctable.
26. If the error is correctable repeat from step 24.

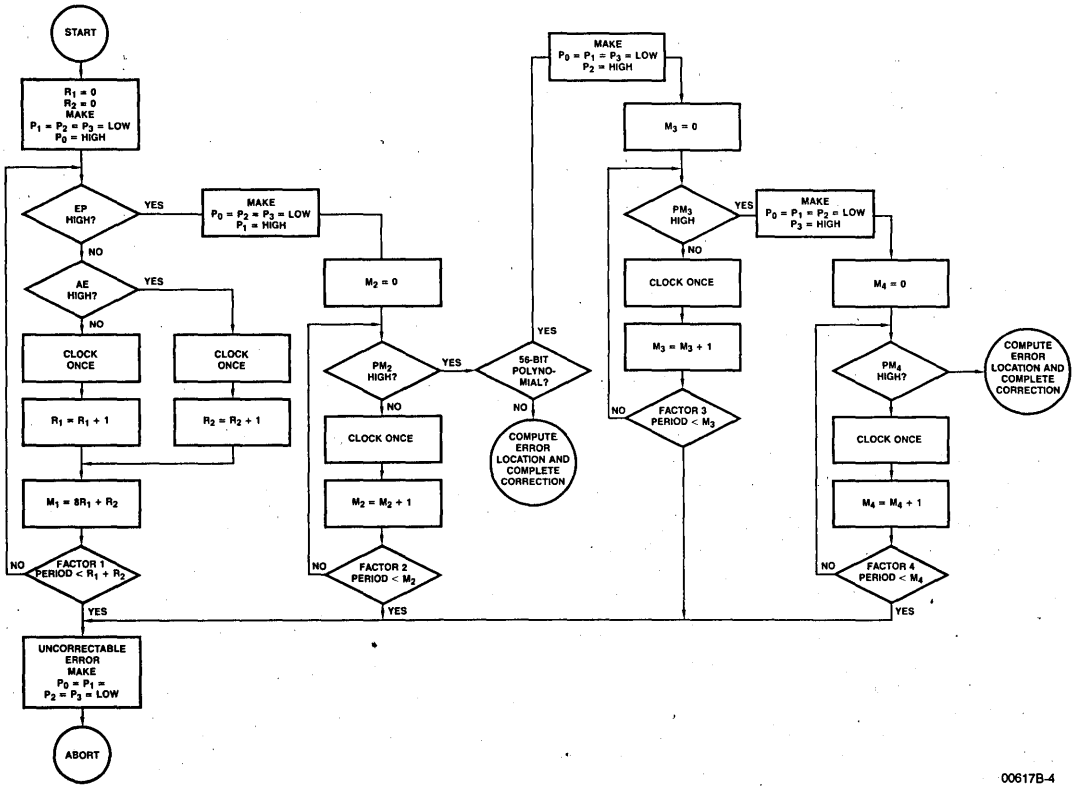
The following flowchart (Figure 4) summarizes the Correct High Speed function.

Load

This function enters the reciprocal of the syndrome into the Am9520. In the case of the 48-bit polynomial, the reciprocal of the syndrome must be formed externally and then entered into the Am9520 before error correction can start.

When the Load function is selected, the register array is configured as a simple 8-bit wide 7 deep shift register. The D_0-D_7 are the inputs to this shift register. Before starting the correction process, seven bytes must be shifted in using the Load function – the first six bytes are the reciprocal of the syndrome and the last byte is an all-zero fill byte.

Figure 4. Correct High-Speed Function



The sequence of events for accomplishing the Load function is as follows:

1. The CP input is in its quiescent HIGH state.
2. Select 48-bit polynomial on the S_0-S_1 inputs.
3. Select Load function on the C_0-C_2 inputs.
4. Set $\overline{MR}$ LOW, and then HIGH.
5. Present a byte to be loaded on the D_0-D_7 inputs.
6. Make CP input LOW and then HIGH.
7. Repeat from step 5 until all six bytes of the reciprocal are entered.
8. Make D_0-D_7 input LOW for the all-zero dummy fill byte.
9. Make CP input LOW and then HIGH.

Error Pattern Information

The discussion of Correct Normal and Correct High Speed functions described the procedure for finding the error pattern and calculating the location of the error burst. The devices provide the error pattern on 12 outputs – eight bits on the $_0$-Q_7 outputs and four bits on the LP_0-LP_3 outputs. It was also mentioned that the REP input must be HIGH to read the error pattern.

The error location calculated using the formulas given is always in number of bits. In case of 56-bit, 35-bit and 32-bit polynomials, the calculated error location value corresponds to the beginning of the error burst counting from the last check bit. The calculated error location is such that when 12 consecutive bits of the record are exclusive ORed into the error pattern, then the error burst is corrected (see Figure 5).

Figure 6 depicts error pattern information for the 48-bit polynomial. In this case, the computed error location refers in bits to the first bit in the burst. However, the burst goes towards the beginning of the data. In the case of 56-, 32- and 35-bit, the burst was towards the check bits. This difference is caused by using the reciprocal of the syndrome.

Figure 5. Error Pattern Format for 56-Bit, 35-Bit and 32-Bit Polynomials

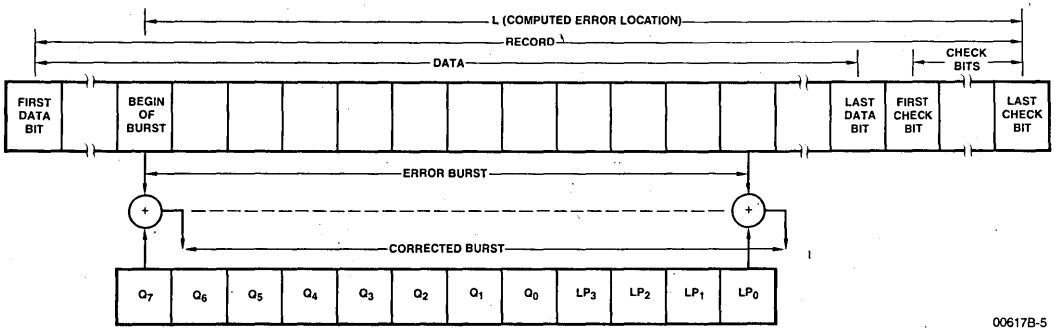
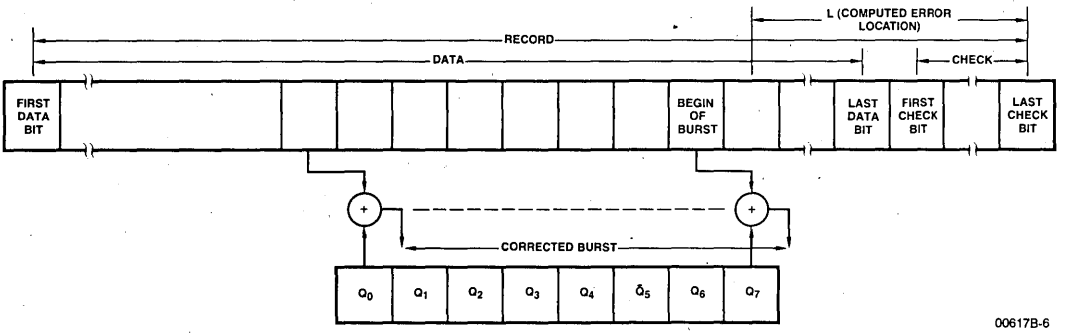


Figure 6. Error Pattern Format for 48-Bit Polynomial



INTERFACE SIGNAL DESCRIPTION

V_{CC} +5V power supply
V_{SS} Ground

S_0-S_1 Polynomial Select (Inputs)

Logic levels on these two inputs select one of the four standard polynomials provided in the Am9520. The following table specifies the polynomial select codes.

POLYNOMIAL SELECT CODES

S_1	S_0	Polynomial	Number of Check Bits
L	L	$(X^{22} + 1) \cdot (X^{11} + X^7 + X^6 + X + 1) \cdot (X^{12} + X^{11} + X^{10} + \ldots + X + 1) \cdot (X^{11} + X^9 + X^7 + X^6 + X^5 + X + 1)$	56
L	H	$(X^{21} + 1) \cdot (X^{11} + X^2 + 1)$	32
H	L	$(X^{23} + 1) \cdot (X^{12} + X^{11} + X^8 + X^7 + X^3 + X + 1)$	35
H	L	$(X^{13} + 1) \cdot (X^{35} + X^{23} + X^8 + X^2 + 1)$	48

D_0-D_7 Data In (Inputs)

These eight inputs are used for entering information. D_0 is the least significant bit and D_7 is the most significant bit position. HIGH on any input corresponds to 1 and LOW represents 0. Data entry occurs on the LOW-to-HIGH transition of the CP input. Any change on the D_0-D_7 inputs must take place only when the CP input is HIGH. See timing diagram for details on setup and hold time specifications.

C_0-C_2 Function Select (Inputs)

These three inputs specify the desired function according to the following table. Detailed description of each function is found in later sections of this document. Any change on the C_0-C_2 inputs must take place only when the CP input is HIGH. See timing diagram for setup and hold time specifications.

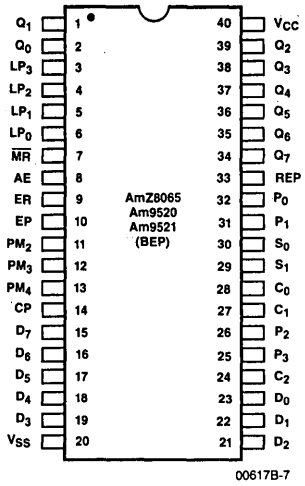
TABLE 4. FUNCTION SELECT CODES

C_2	C_1	C_0	Function
L	L	L	Compute check bits
L	L	H	Write check bits
L	H	L	Read normal
L	H	H	Read high speed
H	L	L	Load
H	L	H	Reserved
H	H	L	Correct normal (Full period clock around)
H	H	H	Correct high speed (Chinese remainder theorem method)

CP Clock (Input)

Operations are controlled by this input. Outputs become valid some propagation delay after the LOW-to-HIGH transition on the CP input. The quiescent state of the CP input is HIGH. Any changes on the data and control inputs must take place only when the CP input is HIGH. See timing diagrams for exact specifications. During operations, it may be required to stop the clock; the clock must be stopped in the HIGH state only. Also, note that requirements on the CP input during correction operations differ from those during other functions. See timing diagram for details.

CONNECTION DIAGRAM
Top View
D-40



Note: Pin 1 is marked for orientation.

Figure 7.

$\overline{MR}$ Master Reset (Input)

LOW on this input initializes the device. This input must remain LOW for a specified time to accomplish initialization before returning to the quiescent HIGH state. In general, the devices require initialization prior to performing Compute Check Bits, Read Normal, Read High Speed and Load functions.

Q_0-Q_7 Data Out (Outputs, 3-State)

The check bits are made available on these eight outputs one byte at a time. Q_0 is the least significant bit position and Q_7 is the most significant. HIGH on these outputs represents 1 and LOW 0.

The Q_0-Q_7 are active only during the following conditions:

a) The C_0-C_2 inputs specify Write Check Bits Function.
b) The REP input is HIGH.

During all other conditions Q_0-Q_7 are in a high-impedance state.

LP_0-LP_3 Located Error Pattern (Outputs, 3-state)

The LP_0-LP_3 outputs together with the Q_0-Q_7 outputs provide the 12-bit error pattern in which Q_7 is the most significant bit and LP_0 is the least significant bit position. HIGH represents 1 and LOW represents 0. The REP input must be HIGH to read the error pattern. If the REP input is LOW, the LP_0-LP_3 outputs are in the high-impedance state.

REP Read Error Pattern (Input)

A HIGH on this input activates the LP_0-LP_3 and Q_0-Q_7 outputs. This error pattern information is valid only after a HIGH is indicated on the EP output during correction operations.

ER Error (Output)

HIGH on this output indicates that the BEP has detected an error. This output must be considered valid only after the last check byte during Read Normal or Read High Speed functions has been entered. The resulting syndrome is then contained in the register array. A non-zero syndrome indicates error; zero syndrome indicates no error. The ER output always reflects the state of this register array (zero or non-zero). The ER output is LOW after initialization.

EP Error Pattern (Output)

HIGH on this output indicates that the error pattern has been found during the correction process. When the last check byte was entered during a Read function the resulting syndrome is contained in the register array. The error pattern information is buried in this syndrome. To extract the error pattern, the BEP is clocked while the appropriate (Correct Normal or Correct High Speed) code is applied to the C_0-C_2 inputs until EP goes HIGH. The number of clocks required to find the error pattern is used to calculate where in the data stream the error has occurred.

The EP output will be LOW after initialization by the $\overline{MR}$ input. The EP output is valid only during the correction operations and must be ignored at all other times. See Correct Normal and Correct High Speed functional descriptions for further details.

PM_2-PM_4 Pattern Match (Outputs)

When using the Chinese remainder theorem for error correction, information is loaded into several feedback shift registers simultaneously. The number of registers is equal to the number of factors of the polynomial. After a high speed operation, there are as many syndromes as there are factors. For correction, the register corresponding to the first factor must be shifted until the EP output indicates HIGH. Then each register corresponding to the remaining factors must be shifted until a match occurs in each register with the error pattern contained in the first register. HIGH on PM_2, PM_3 or PM_4 outputs indicates that corresponding registers match. The PM_2 corresponds to the second factor, PM_3 corresponds to the third and PM_4 corresponds to the fourth factor. If a polynomial has only two factors, then PM_3 and PM_4 outputs have no meaning. Indications on the PM_2-PM_4 outputs must be considered valid only during high-speed correct function and should be ignored at all other times.

P_0-P_3 Polynomial Shift Control (Inputs)

Correction procedure using the Chinese remainder theorem method requires that each syndrome obtained from the High-Speed Read function be shifted individually. The P_0-P_3 inputs provide this capability: P_0 corresponds to the first factor, P_1 corresponds to the second factor and so on. HIGH on an input allows the corresponding register to shift and LOW causes it to hold. These inputs have an effect only during the Correct High-Speed function. Any change on these inputs must occur only when the CP input is HIGH.

AE Alignment Exception (Output)

The devices use an 8-bit parallel mechanization of the feedback shift register configurations. Under certain conditions, the error pattern will not, therefore, automatically line up in predetermined positions of the register array during the correction operations. HIGH on the AE output indicates that such a condition is detected. The Am9520 automatically switches into the one-bit shift mode. The number of clocks for which the AE output is HIGH is used in the error location calculation. See functional description for further details.

MAXIMUM RATINGS beyond which useful life may be impaired

Storage temperature	−65 to +150°C
Voltage at any pin relative to V_{SS}	−0.5 to +7.0V
Power dissipation	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

AmZ8065/Am9520/Am9521

ELECTRICAL CHARACTERISTICS over operating range unless otherwise specified (Note 1)

Parameter	Description	Test Conditions	Min	Typ	Max	Units
V_{IL}	Input LOW Voltage		−0.5		+.8	Volts
V_{IH}	Input HIGH Voltage		2.0		V_{CC}	Volts
V_{OL}	Output LOW Voltage	I_{OL} = 3.2mA			0.45	Volts
V_{OH}	Output HIGH Voltage	I_{OH} = −400μA	2.4			Volts
I_{OL}	Output Leakage Current	V_{OUT} = 0.4V			10	μA
I_{LOH}	Output Leakage Current	V_{OUT} = V_{CC}			10	μA
C_{IN}	Input Capacitance				15	pF
CI/O	I/O Capacitance				25	pF
I_{LL}	Input Leakage Current				±10	μA
I_{CC}	Power Supply Current				275	mA

Note 1. Typical values apply at T_A = 25°C and V_{CC} = 5.0V. See table above for operating range.

AmZ8065/Am9520/Am9521
SWITCHING CHARACTERISTICS

The table below specifies the guaranteed performance of this device over the commercial operating range of 0 to +70°C with V_{CC} from 4.75V to 5.25V. All data are in nanoseconds. Switching tests are made with inputs and outputs measured at 0.8V for a LOW and 2.0V for a HIGH. Outputs are fully loaded, with $C_L \geq$ 50pF. See switching waveform figures following table for graphic illustration of timing parameters.

SWITCHING CHARACTERISTICS OVER OPERATING RANGE

				AmZ8065 Am9520 Am9521		AmZ8065-1 Am9520-1 Am9521-1	
Number	**Parameter**	**Description**	**Units**	**Min**	**Max**	**Min**	**Max**
1	TWCPL	CP Width LOW	ns	180		105	
2	TCYCP	CP Cycle Time	ns	400		250	
3	TWCPH	CP Width HIGH	ns	180		105	
4	TWMRL	$\overline{MR}$ Width LOW	ns	800		500	
5	TREC	$\overline{MR}\uparrow$ to CP$\downarrow$ Time (Recovery)	ns	250		250	
6	TSDCP	D_0-D_7 to CP$\uparrow$ Setup Time	ns	350		200	
7	THDCP	CP$\uparrow$ to D_0-D_7 Hold Time	ns	0		0	
8	TSCCP	C_0-C_2 or S_0-S_1 to CP$\uparrow$ Setup Time	ns	400		200	
9	THCCP	CP$\uparrow$ to C_0-C_2, S_0-S_1, P_0-P_3 Hold Time	ns	0		0	
10	TSCCPL	C_0-C_2 or S_0-S_1 to CP$\downarrow$ Setup Time	ns	180		95	
11	TVCQ	C_0-C_2, S_0-S_1 to Q_0-Q_7 Valid Delay	ns		200		150
12	TIVCPQ	CP$\uparrow$ to Q_0-Q_7 Invalid Delay	ns	0		0	
13	TVCPQ	CP$\uparrow$ to Q_0-Q_7 Valid Delay	ns		200		150
14	TIVCQ	C_0-C_2 to Q_0-Q_7 Three-State Delay	ns		100		100
15	TMRERL	$\overline{MR}\downarrow$ to ER$\downarrow$ Delay	ns		200		200
16	TCPER	CP$\uparrow$ to ER Valid Delay	ns		200		200
17	TWCPCL	CP Width $\overline{LOW}$ for Correct Functions	ns	450		450	
18	TWCPCH	CP Width $\overline{HIGH}$ for Correct Functions	ns	450		450	
19	TCYCPC	CP Cycle Time for Correct Functions	ns	1000		1000	
20	TCEP	C_0-C_2 to EP or AE Valid Delay	ns		250		250
21	TCPEP	CP$\downarrow$ to EP, AE, or PM_2-PM_4 Valid Delay	ns		400		400
22	TSCPS	P_0, P_1, P_2, P_3 to CP$\downarrow$ Setup Time	ns	400		400	
23	TCEP	P_0 to EP or AE Delay	ns		250		250
24	TCPC	C_0-C_2, S_0-S_1 to CP$\downarrow$ Setup Time for Correct Functions	ns	400		400	
25	TPPM	P_1, P_2, P_3 to Corresponding PM Output Delay	ns		250		250
26	TCPEPI	CP$\downarrow$ to EP, AE, PM_2, PM_3, and PM_4 Invalid Delay	ns	0		0	
27	TPEPI	$P_0\downarrow$ to EP, AE Invalid Delay	ns	0		0	
28	TWREP	REP Pulse Width $\overline{HIGH}$	ns	250		250	
29	TREPQ	REP$\uparrow$ to Q_0-Q_7 and LP_0-LP_3 Delay	ns		150		150
30	TREPQI	REP$\downarrow$ to Q_0-Q_7 and LP_0-LP_3 Three-State Delay	ns		100		100
31	TPPM	P_1, P_2, $P_3\downarrow$ to PM_2, PM_3, PM_4 Invalid	ns	0		0	
32	TCPM	C_0-C_2 to EP, AE, PM_2-PM_4 Invalid	ns	0		0	

Figure 8. Clock Waveform for All Functions Except Correct Normal or Correct High-Speed

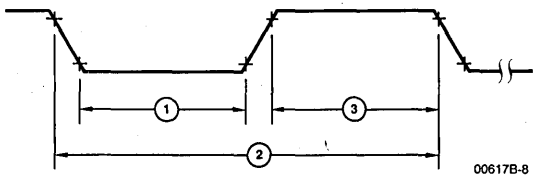


Figure 9. Timing for Compute Check Bits or Load Function

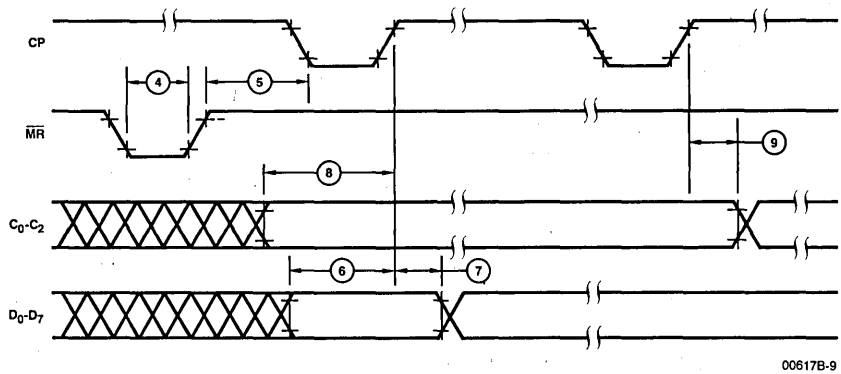
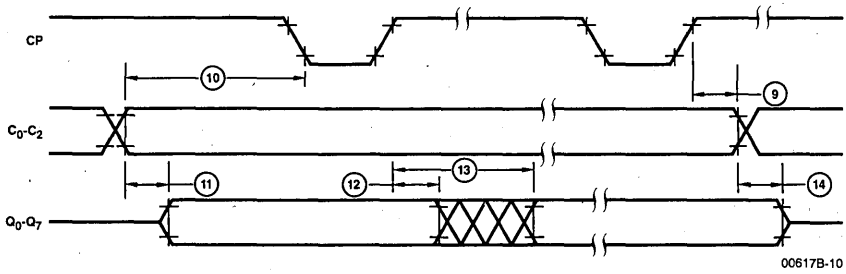
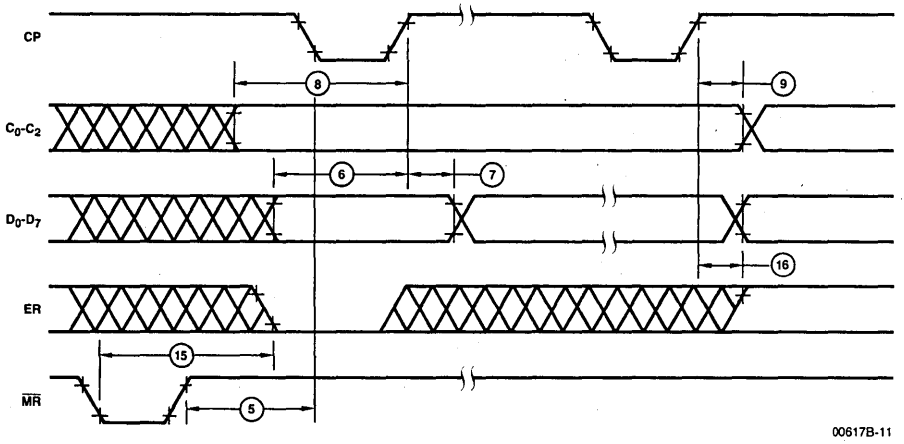


Figure 10. Timing for Write Check Bits Function



Notes: 1. REP input assumed low.
2. Q_0-Q_7 outputs will be high impedance if C_0-C_2 inputs do not specify write check bits function.

Figure 11. Timing for Read Normal or Read High-Speed Function



Note: ER output is a function of the contents in the register array flip-flops.

Figure 12. Input Output Waveforms for AC Tests

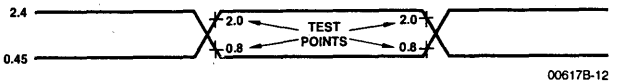


Figure 13. Clock Waveform for Correct Normal or Correct High-Speed Functions

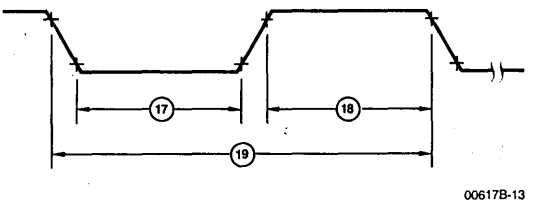
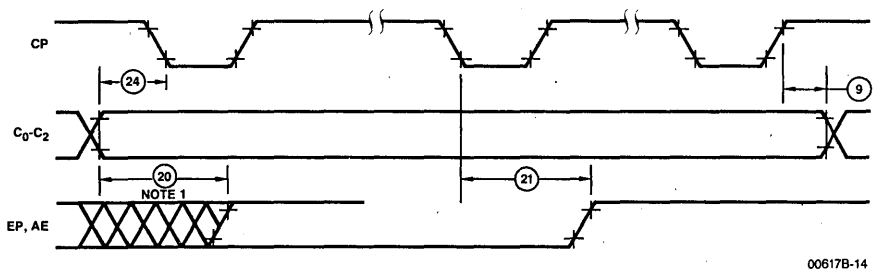
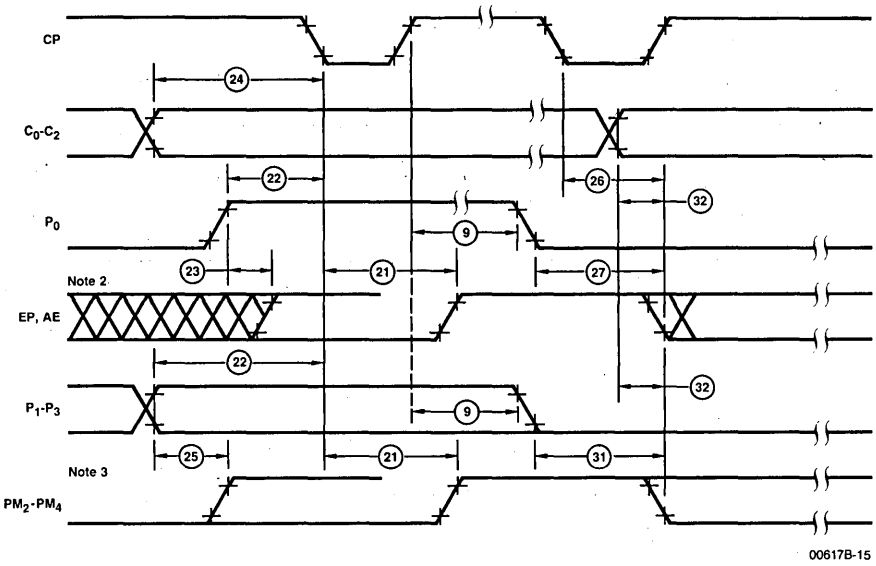


Figure 14. Timing for Correct Normal Function



Note 1: Assumes AE or EP output becomes active without any clocking.

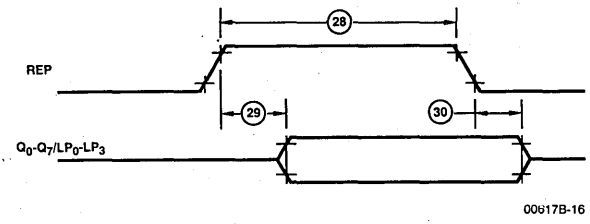
Figure 15. Timing for Correct High-Speed Function



Note 2: Assumes EP, AE becomes active without clocking.

Note 3: Assumes corresponding PM output becomes active without clocking.

Figure 16. Read Error Pattern Timing

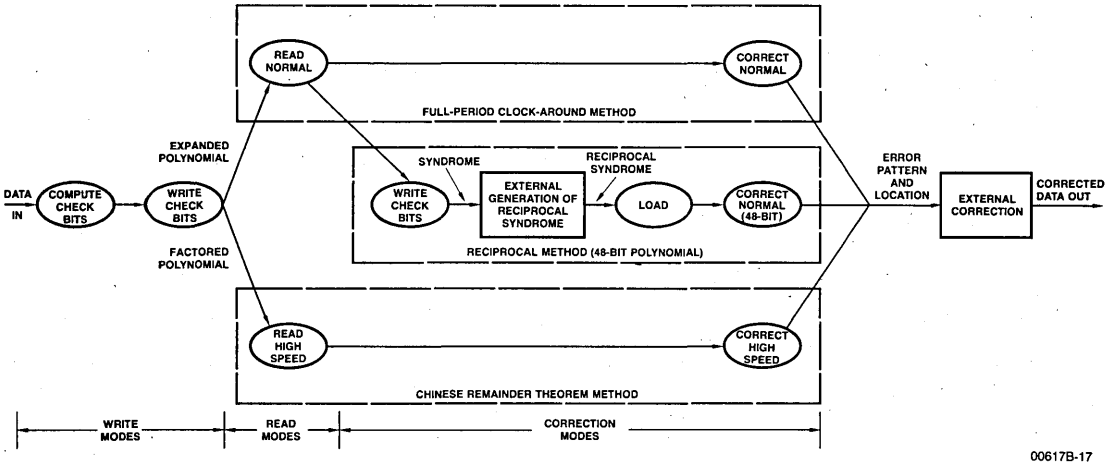


APPLICATIONS FOR THE BEP

The BEP is designed for use in both microprogrammed and microprocessor disk controller systems. The BEP operational flow diagram on page 1 shows the BEP interfacing to an Am2900 bipolar bit-slice microprogrammed disk controller. The BEP can be interfaced to microprocessor-driven disk controller systems as well.

The controller in these designs would implement the control and clocking signals for the BEP necessary to execute the write, read and correction functions for a given polynomial selection. The operational flow for the methods available is shown in Figure 8.

Figure 17. BEP Operational Flow Diagram.



ORDERING INFORMATION

Order Code		Screening Level (Note 1)	Operating Range (Note 2)	Speed	
32, 35, 48 and 56 Bit Polynomials	32 and 35 Bit Polynomials Only			Clock	Effective Data Rate
AmZ8065DC Am9520DC	Am9521DC	C-1	$0°C \leq T_A \leq 70°C$ $V_{CC} = +5V \pm 5\%$	2.5MHz	20 Mbits Per Second
AmZ8065DCB Am9520DCB	Am9521DCB	B-1			
AmZ8065DI Am9520DI	Am9521DI	C-1	$-40°C \leq T_A \leq 55°C$ $V_{CC} = +5V \pm 10\%$		
AmZ8065DIB Am9520DIB	Am9521DIB	B-1			
AmZ8065DMB Am9520DMB	Am9521DMB	B-1	$-55°C \leq T_A \leq 125°C$ $V_{CC} = +5V \pm 10\%$		
Am9520-1DC	Am9521-1DC	C-1	$0°C \leq T_A \leq 70°C$ $V_{CC} = +5V \pm 5\%$	4MHz	30 Mbits Per Second
Am9520-1DCB	Am9521-1DCB	B-1			

Notes: 1. Level C-1 conforms to MIL-STD-883, Class C. Level B-1 conforms to MIL-STD-883, Class B.
2. Range over which the DC, switching and functional specifications apply. $V_{SS} = 0V$; $V_{IL} = 0.8V$; $V_{IH} = 2.0V$.
3. All devices are packaged in 40-pin hermetic DIP.

AmZ8068/Am9518

Data Ciphering Processor

DISTINCTIVE CHARACTERSITICS

- **Encrypts and decrypts data**
  Implements National Bureau of Standards standard data encryption algorithm.
- **High-Speed Operation**
  AmZ8068 and AM9518 throughput over 1.7and 1.3M bytes per second respectively. Operates at data rates fast enough for disk controllers, high-speed DMA, telecommunication channels.
- **Supports three ciphering options**
  Electronic Code Book for disk applications.
  Chain Block Cipher for high-speed telecommunications.
  Cipher Feedback for low- to medium-speedbyte oriented communications.
- **Three separate key registers on-chip**
  Separate registers for encryption key, decryption key and master key improve system security and throughput by eliminating need to reload keys frequently.
- **Three separate data ports provide flexible interface, improved security**
  The DCP utilizes a master port, slave port and key port. Functions of the three ports can be programmed by the user to provide for simple interface to AmZ8000 and Am2900 systems and to provide total hardware separation of encrypted data, clear data and keys.

GENERAL DESCRIPTION

The AmZ8068/Am9518 Data Ciphering Processor is an N-channel silicon gate LSI product containing the circuitry necessary to encrypt and decrypt data using the National Bureau of Standards encryption algorithm. It is designed to be used in a variety of environments, including dedicated controllers, communication concentrators, terminals and peripheral task processors in general processor systems.

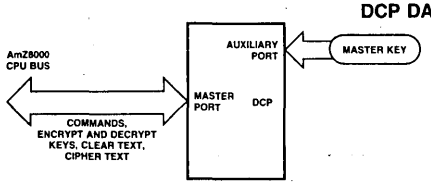
The DCP provides a high throughput rate using Cipher Feedback, Electronic Code Book or Cipher Block Chain operating modes. Separate ports for key input, clear data and enciphered data enhance security.

The system communicates with the DCP using commands entered in the master port and through auxiliary control lines. Once set up, data can flow through the DCP at high speeds because input, output and ciphering activities are all performed concurrently. External DMA control can easily be used to enhance throughput in some system configurations.

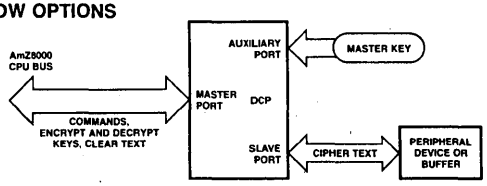
This device is designed to interface directly to the AmZ8000 CPU bus and, with a minimum of external logic, to the 2900, 8080, 8085, and 8048 families of processors.

Export of this device from the United States is subject to control by the U.S. Department of State.

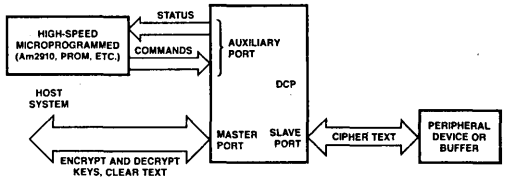
DCP DATA FLOW OPTIONS



Single-Port Configuration, Multiplexed Control



Dual-Port Configuration, Multiplexed Control



Dual-Port Configuration, Direct Control

00618B-1

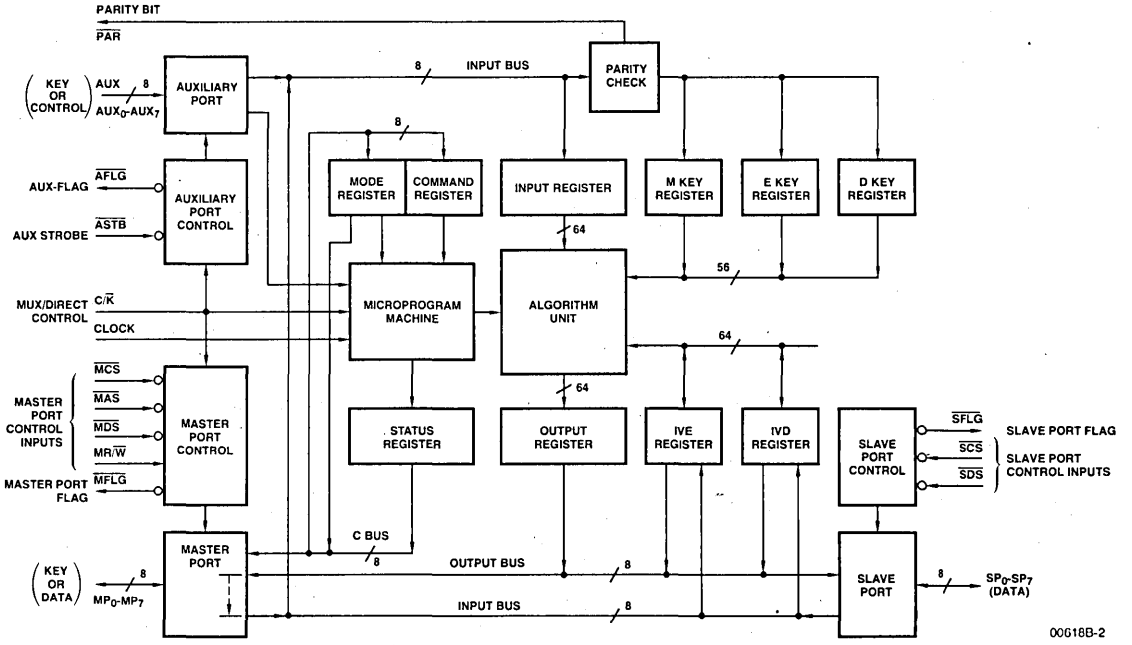
ORDERING INFORMATION

Order Code	Screening Level (Note 1)	Speed	Package Type	Operating Range (Note 2)
AmZ8068DC AmZ8068DCB	C-1 B-1	4MHz	40 Pin Hermetic DIP	T_A = 0 to +70°C V_{CC} = 4.75 to 5.25V V_{IH} = 2.2V V_{IL} = 0.8V
Am9518DC Am9518DCB	C-1 B-1	3MHz		

Notes: 1. Level C-1 conforms to MIL-STD-883, Class C.
Level B-3 conforms to MIL-STD-883, Class B.
2. Range over which the DC, switching and functional specifications apply.

00618B-MMP

AmZ8068/Am9518 DCP BLOCK DIAGRAM



FUNCTIONAL DESCRIPTION

The overall design of the DCP, as shown in the block diagram, is optimized for high data throughput. Data bytes can be transferred through both the Master and Slave ports, and key bytes can be written through both the Auxiliary and Master ports. Three 8-bit buses, Input, Output and C Bus, carry data and key bytes between the ports and the internal registers. Three 56-bit, write-only key registers are provided for the Master (M) Key, the Encryption (E) Key and the Decryption (D) Key. Parity checking is provided on incoming key bytes. Two 64-bit registers are provided for Initializing Vectors (IVE and IVD) required for chained (feedback) ciphering modes. Three 8-bit registers (Mode, Command and Status) are accessible through the Master Port for interfacing to a host microprocessor, such as the AmZ8000.

Algorithm Processing

The DCP's algorithm processing unit (see the block diagram) is designed to encrypt and decrypt data according to the National Bureau of Standards Data Encryption Standard (DES), as specified in Federal Information Processing Standards Publication 46.

The DES specifies a method for encrypting 64-bit blocks of clear data ("plain text") into corresponding 64-bit blocks of "cipher text." The DCP offers three ciphering methods, selected by the Cipher Type field of the Mode Register: Electronic Code Book (ECB), Cipher Block Chain (CBC) and Cipher Feedback (CFB). These methods are implemented in accordance with Federal Information Processing Standards Publication 46. Electronic Code Book (ECB) is a straightforward implementation of the DES: 64 bits of clear data in, 64 bits of cipher text out, with no cryptographic dependence between blocks. Cipher Block Chain (CBC) also operates on blocks of 64 bits, but includes a feedback step which chains consecutive blocks so that repetitive data in the plain text (such as ASCII blanks) does not yield repetitive cipher text; CBC also provides an error extension characteristic valuable in protecting against fraudulent data insertions and deletions.

Cipher Feedback (CFB) is an additive stream cipher method in which the DES generates a pseudorandom binary stream which is then exclusive-OR'd with the clear data to form the cipher text. The cipher text is then fed back to form a portion of the next DES input block. The DCP implements 8-bit cipher feedback, with data input, output, and feedback paths being one byte wide. This method is useful for low speed, character-at-a-time serial communications.

Multiple Key Registers

The DCP provides the necessary registers to implement a multiple-key or Master-Key system. In such an arrangement, a single Master Key, stored in the DCP M Key Register, is used only to encrypt session keys for transmission to remote DES equipment, and to decrypt session keys received from such equipment. The M Key Register may be loaded (with plain text) only through the Auxiliary Port, using the Load Clear Master Key command. (See Commands.)

In addition to the M Key Register, the DCP contains two session key registers: the E Key Register, used to encrypt clear text, and the D Key Register, used to decrypt cipher text.

All three registers are loaded by writing commands like Load Clear E Key through Master Port into the Command Register, and then writing the eight bytes of key data to the port when the Command Pending = "1" in the Status Register. (See Commands.)

Operating Modes: Multiplexed Control vs. Direct Control

The DCP can be operated in either of two basic interfacing modes, determined by the logic level on the C/$\overline{K}$ input pin. In Multiplexed Control Mode (C/$\overline{K}$ LOW), the DCP is internally configured to allow a host CPU to directly address five of the internal control/status/data registers and thereby control the device via mode and command values written to these registers. Also, in Multiplexed Control Mode, the Auxiliary Port is enabled for key-byte input.

If the logic level on C/$\overline{K}$ is brought HIGH, the DCP enters Direct Control Mode, and the Auxiliary Port pins are converted into direct hardware status or control signals that are capable of instructing the DCP to perform a functionally complete subset of its cipher processing at very high throughputs. This operating mode is particularly well suited for ciphering data for high-speed peripheral devices such as magnetic disk or tape.

Data Flow

Bits M_2, M_3 of the Mode Register control the flow of data into and out of the DCP through the Master and Slave Ports. Three basic configurations are provided: Single Port, and two Dual Port configurations.

Single Port Configuration

The simplest configuration occurs when the Mode Register configuration bits are set to Master Port only. Under this operating configuration the Encrypt/Decrypt bit (M_4) controls the processing of data. Data to be encrypted or decrypted is written to the Master Port Input Register address. To facilitate monitoring of the Input Register status, the MFLG signal goes LOW when the Input Register is not full. Data is read by the host CPU through the Master Port Output Register address. Pin SFLG goes LOW when the Output Register is not empty. Thus, MFLG is redefined as a Master Input Flag and SFLG is redefined as a Master Output Flag.

Dual Port, Master Port Clear Configuration

In the dual port configurations, both the Master and Slave Ports are used for data entry and removal. In the Master Port Clear configuration, clear text for encryption can be entered only through the Master Port, and clear text resulting from decryption can be read out only through the Master Port. Cipher text can be handled only through the Slave Port. The actual direction of data flow is controlled either by the Encrypt/Decrypt bit (M_4) in the Mode Register, or by the Start Encryption or Start Decryption commands. If encryption is specified, clear data will flow through the Master Port to the Input Register, and cipher data will be available at the Slave Port when it is ready to be read out of the Output Register. For decryption, the process is reversed, cipher data being written to the Input Register through the Slave Port, and Clear data being read from the Output Register through the Master Port.

Dual Port, Slave Port Clear Configuration

This configuration is identical to the previously described Dual Port, Master Port Clear configuration, except that the direction of ciphering is reversed. That is, all data flowing in or out of the Master Port is cipher text, and all data at the Slave Port is clear text.

Master Port Read/Write Timing

The DCP's Master Port is designed to operate directly with the multiplexed address-data bus such of the AmZ8000 processor. Several features of the Master Port logic should be stressed.

- The level on Master Port Chip Select ($\overline{MCS}$) is latched internally on the rising (trailing) edge of Master Port Address Strobe ($\overline{MAS}$), thus relieving external address decode circuitry of the responsibility for latching chip select at address time.

- The levels on MP_1, MP_2 are also latched internally on the rising edge of $\overline{MAS}$, and are subsequently decoded to enable reading and writing of the DCP's internal registers (Mode, Command, Status, Input and Output). Again, this eliminates the need for external address latching and decoding.

  Data transfers through the Master Port are controlled by the levels and transitions on Master Port Data Strobe ($\overline{MDS}$) and Master Port Read/Write (MR/$\overline{W}$), the former controlling the timing and the latter controlling the transfer direction. Note that data transfers do not disturb either the chip-select or address latches, so that once the DCP and a particular register have been selected, any number of reads or writes of that register can be accomplished without intervening address cycles. This feature could greatly speed up loading keys and data, given the necessary transfer control external to the DCP.

Loading Keys and Initializing Vector (IV) Registers

Because the key and initializing vector registers are not directly addressable through any of the DCP's ports, keys and vector data must be loaded (and, in the case of vectors, read out) via "command data sequences" (see Commands). Most of the commands recognized by the DCP are of this type: A Load or Read command is written to the Command Register through the Master Port; the command processor responds by asserting the Command Pending output; the user then either writes eight bytes of key or vector data through the Master or Auxiliary Port, as appropriate to the specific command, or reads eight bytes of vector data from the Master Port.

In Direct Control Mode, only the E Key and D Key registers can be loaded; the M Key and IV Registers are inaccessible. Loading the E and D Key registers is accomplished by asserting the proper state on the AUX_6-E/$\overline{D}$ input (HIGH for E Key, LOW for D Key) and then raising the AUX_7-K/$\overline{D}$ input, indicating that key loading is required. The command processor will attach the proper key register to the Master Port and assert the AUX_3-$\overline{CP}$ (Command Pending) signal (active LOW). The eight key bytes may then be written to the Master Port. In Multiplexed Control Mode, all key and vector registers are writeable and all but the Master (M) Key Register may be loaded with encrypted, as well as clear, data. If the operation is a Load Encrypted command, the subsequent data written to the Master or Auxiliary Port (as appropriate) is routed first to the Input Register and decrypted before being written into the specified key or vector register.

Parity Checking of Keys

Key bytes are considered to contain seven bits of key information and one parity bit. By DES designation, the low-order bit is the parity bit. The parity checking circuit is enabled whenever a byte is written to one of three key registers. The output of the parity detection circuit is connected to pin $\overline{PAR}$ and the state of this pin is reflected in Status Register bit PAR (S_3). Status Register bit PAR goes to "1" whenever a byte with even parity (an even number of "1"s) is detected. In addition to the PAR bit, the Status Register has a Latched Parity Bit (LPAR, S_4) which is set to "1" whenever the Status Register PAR bit goes to "1." Once set, the LPAR bit is not cleared until a reset occurs or a new Load Key command is issued.

When an encrypted key is entered, the parity detect logic operates only after the decrypted key is available. The encrypted data is not checked for parity. The $\overline{PAR}$ signal will reflect the state of the decrypted bytes on a byte-to-byte basis, as they are clocked through the parity check logic on their way to the Key Register. Thus, the time $\overline{PAR}$ indicates the status of a byte of decrypted key data may be as short as four clock cycles. The LPAR bit in the Status Register will indicate if any erroneous bytes of data were entered.

Initialization

The DCP can be reset in several ways:

1. By the "Software Reset" command
2. By a hardware reset, which occurs whenever both $\overline{MAS}$ and $\overline{MDS}$ go LOW simultaneously
3. By writing to the Mode Register
4. By aborting any command

All these sequences are the same internally, except that loading the Mode Register does not subsequently reset the Mode Register.

Once a reset process starts, the DCP is unable to respond to further commands for approximately five clock cycles.

If a power-up hardware reset is used, the leading edge of the reset signal should not occur until approximately 1 ms after V_{CC} has reached normal operating voltage. This delay time is needed for internal signals to stabilize.

REGISTER DESCRIPTION

The registers in the DCP which can be directly addressed through the Master Port are shown with their addresses in Figure 1. A brief description of these registers and others not directly accessible is given below.

Command Register

Data written to the 8-bit, write-only Command Register through the Master Port is interpreted as an instruction. A detailed description of each command is given under Functional Description, and the commands and their binary representations are summarized in Figure 2.

Status Register

The bit assignments in the read-only Status Register are shown in Figure 4. The PAR, AFLG, SFLG and MFLG bits indicate the status of the like-named output pins, as do the bits Busy and Command Pending when the DCP is in Direct Control Mode ($C/\overline{K}$ HIGH). In each case, the output signal will be active LOW when the corresponding status bit is a "1". The Parity bit indicates the parity of the most recently entered key byte. The LPAR bit, on the other hand, indicates whether any key byte with even parity has been encountered since the last Reset or Load Key command.

The Busy bit will be a "1" whenever the ciphering algorithm unit is actively encrypting or decrypting data, either as a reponse to a command such as Load Encrypted Key (in which case the Command Pending bit will be a "1"), or in the ciphering of regular text (indicated by the Start/Stop bit being a "1"). The Busy bit will remain a "1" even after ciphering is complete if the ciphered data cannot be transferred to the Output Register because that register still contains output from a previous ciphering cycle. Busy will be "0" at all other times, including if no ciphering is possible because no data has been written to the Input Register.

The Command Pending bit will be set to "1" by any command whose execution requires the transfer of data to or from a non-addressable internal register, such as when writing key bytes to the E Key Register or reading bytes from the IVE Register. Thus, Command Pending will be set following all commands except the

$C/\overline{K}$	MP2	MP1	$MR/\overline{W}$	$\overline{MCS}$	Register Addressed
0	X	0	0	0	Input Register
0	X	0	1	0	Output Register
0	0	1	0	0	Command Register
0	0	1	1	0	Status Register
0	1	1	X	0	Mode Register
X	X	X	X	1	No Register Accessed
1	X	X	0	0	Input Register
1	X	X	1	0	Output Register

Figure 1. Master Port Register Addresses

Hex Code	Command
90	Load Clear M Key through Auxiliary Port
91	Load Clear E Key through Auxiliary Port
92	Load Clear D Key through Auxiliary Port
11	Load Clear E Key through Master Port
12	Load Clear D Key through Master Port
B1	Load Encrypted E Key through Auxiliary Port
B2	Load Encrypted D Key through Auxiliary Port
31	Load Encrypted E Key through Master Port
32	Load Encrypted D Key through Master Port
85	Load Clear IVE through Master Port
84	Load Clear IVD through Master Port
A5	Load Encrypted IVE through Master Port
A4	Load Encrypted IVD through Master Port
8D	Read Clear IVE through Master Port
8C	Read Clear IVD through Master Port
A9	Read Encrypted IVE through Master Port
A8	Read Encrypted IVD through Master Port
39	Encrypt with Master Key
41	Start Encryption
40	Start Decryption
C0	Start
E0	Stop
00	Software Reset

Figure 2. Command Codes in Multiplexed Control Mode

Pins				Command Initiated
$C/\overline{K}$	AUX7-$K/\overline{D}$	AUX6-$E/\overline{D}$	AUX5-$S/\overline{S}$	
H	L	L	↑	Start Decryption
H	L	H	↑	Start Encryption
H	L	X	↓	Stop
H	↑	L	L	Load D Key Clear through Master Port
H	↑	H	L	Load E Key Clear through Master Port
H	↓	X	L	End Load Key Command
H	H	X	H	Not Allowed
L	Data	Data	Data	AUX Pins Become Key-Byte Inputs

Figure 3. Implicit Command Sequences in Direct Control Mode

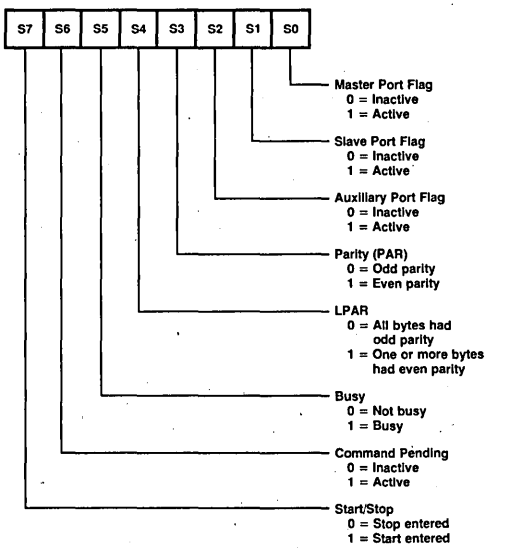


Figure 4. Status Register Bit Assignments

three Start commands, the Stop command and the Software Reset command. Command Pending will return to "0" after all eight bytes have been transferred following Load Clear, Read Clear or Read Encrypted commands; and after data has been transferred, decrypted and loaded into the desired register following Load Encrypted commands.

The Start/Stop bit is set to "1" when one of the Start commands is entered, and is reset to "0" whenever a reset occurs or when a new command other than a Start is entered.

Mode Register

Bit assignments in this 5-bit read/write register are shown in Figure 5. The Cipher Type bits (M_1, M_0) indicate to the DCP which ciphering algorithm is to be used. On reset, the Cipher Type defaults to Electronic Code Book.

Configuration bits (M_3, M_2) indicate which data ports are to be associated with the Input and Output Registers and flags. When these bits are set to the Single Port, Master-only configuration (M_3, M_2 = 10) the Slave Port is disabled and no manipulation of Slave Port Chip Select ($\overline{SCS}$) or Data Strobe ($\overline{SDS}$) can result in data movement through the Slave Port; all data transfers are accomplished through the Master Port, as described more fully in the Functional Description. Both $\overline{MFLG}$ and $\overline{SFLG}$ are used in this configuration; $\overline{MFLG}$ gives the status of the Input Register and $\overline{SFLG}$ the Output Register.

When the Configuration Bits are set to one of the Dual Port configurations (M_3, M_2 = 00 or 01), both the Master and Slave Ports are available for input and output. When M_3, M_2 = 01 (the default configuration), the Master Port handles clear data while the Slave Port handles encrypted data. Configuration M_3, M_2 = 00 reverses this assignment. Actual data direction at any particular moment is controlled by the Encrypt/Decrypt bit.

The Encrypt/Decrypt bit (M_4) instructs the DCP algorithm processor to encrypt or decrypt the data from the Input Register using the ciphering method specified by the Cipher Type bits. The Encrypt/Decrypt bit also controls data flow within the DCP. For example, when the configuration bits are "01" (Dual Port, Master Clear, Slave encrypted) and the Encrypt/Decrypt bit is "1" (encrypt), clear data will flow into the DCP through the Master

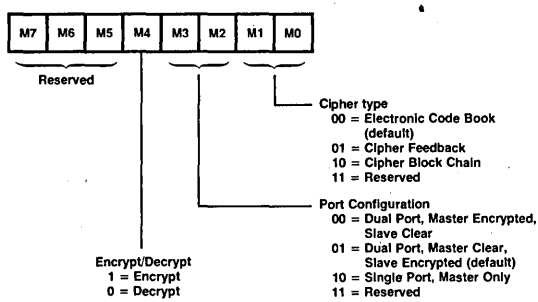


Figure 5. Mode Register Bit Assignments

Port and encrypted data will flow out through the Slave Port. When the Encrypt/Decrpyt bit is set to "0" (decrypt), data flow reverses.

Input Register

The 64-bit, write-only Input Register is organized to appear to the user as eight bytes of push down storage. A status circuit monitors the number of bytes that have been stored. The register is considered empty when the data stored in it has been or is being processed; it is considered full when one byte of data has been entered in Cipher Feedback or when eight bytes of data have been entered in Electronic Code Book or Cipher Block Chain. If the user attempts to write data into the Input Register when it is full, the Input Register will disregard the attempt; no data in the register will be destroyed.

Output Register

The 64-bit, read-only Output Register is organized to appear to the user as eight bytes of pop-up storage. A status circuit detects the number of bytes stored in the Output Register. The register is considered empty when all the data stored in it has been read out by the host CPU, and is considered full if it still contains one or more bytes of output data. If a user attempts to read data from the Output Register when it is empty, the buffers driving the output bus will remain in a three-state condition.

The following multibyte registers cannot be directly addressed, but are loaded or read in response to commands written to the Command Register. (See Commands.)

M, E, D Key Registers

There are three 64-bit, write-only key registers in the DCP: the Master (M) Key Register; the Encrypt (E) Key Register; and the Decrypt (D) Key Register. The Master Key can be loaded only with clear data through the Auxiliary Port. The Encrypt and Decrypt Keys can be loaded in any of four ways: (1) as clear data through the Auxiliary Port; (2) as clear data through the Master Port; (3) as encrypted data through the Auxiliary Port; or (4) as encrypted data through the Master Port. In the last two cases, the encrypted data is first routed to the Input Register, decrypted using the M Key, and finally written to the target key register from the Output Register.

Initializing Vector Registers

Two 64-bit registers are provided to store feedback values for Cipher Feedback and Chained Block ciphering methods. One Initializing Vector (IVE) register is used during encryption, the other (IVD) during decryption. Both registers can be loaded with either clear or encrypted data through the Master Port (in the latter case, the data is decrypted before being loaded into the IV register), and both may be read out either clear or encrypted through the Master Port. (See Commands.)

Encrypt/ Decrypt M4	Port Configuration M3	M2	Input Register Flag	Output Register Flag
0	0	0	MFLG	SFLG
0	0	1	SFLG	MFLG
0	1	0	MFLG	SFLG
1	0	0	SFLG	MFLG
1	0	1	MFLG	SFLG
1	1	0	MFLG	SFLG

Figure 6. Association of Master Port Flag (MFLG) and Slave Port Flag (SFLG) with Input and Output Registers

COMMANDS

All operations of the DCP result from command inputs, which are entered in Multiplexed Control Mode by writing a command byte to the Command Register. Command inputs are entered in Direct Control Mode by raising and lowering the logic levels on the AUX_7-K/$\overline{D}$, AUX_6-E/$\overline{D}$ and AUX_5-S/$\overline{S}$ pins. Figure 2 shows all commands that may be given in Multiplexed Control Mode. Figure 3 shows that subset executable in Direct Control Mode.

Load Clear M Key Through Auxiliary Port (90 Hex)

Load Clear E Key Through Auxiliary Port (91)

Load Clear D Key Through Auxiliary Port (92)

These commands override the data flow specifications set in the Mode Register and cause the Master (M), Encrypt (E), or Decrypt (D) Key Register to be loaded with eight bytes written to the Auxiliary Port. After the Load command is written to the Command Register, the Auxiliary Port Flag ($\overline{AFLG}$) will go active (LOW) and the corresponding bit in the Status Register (S_2) will go to "1," indicating that the device is able to accept key bytes at the Auxiliary Port pins. Additionally, the Command Pending bit (S_6) will go to "1" during the entire loading process.

Each byte is written by placing an active LOW signal on the Auxiliary Port Strobe ($\overline{ASTB}$) once data has been setup on the Auxiliary Port pins. The actual write process occurs on the rising (trailing) edge of $\overline{ASTB}$. (See Switching Characteristics for exact setup, strobe width, and hold times.)

The Auxiliary Port Flag ($\overline{AFLG}$) will go inactive immediately after the eighth strobe goes active (LOW). However, the Command Pending bit (S_6) will remain "1" for several more clock cycles, until the key loading process is completed. All key bytes are checked for correct (odd) parity as they are entered (see Parity Checking).

Load Clear E Key Through Master Port (11 Hex)

Load Clear D Key Through Master Port (12)

These commands are available in both multiplexed control and direct control modes. They override the data flow specifications set in the Mode Register and attach the Master Port inputs to the Encrypt (E) or Decrypt (D) Key Register, as appropriate, until eight key bytes have been written. In Multiplexed Control Mode, the command is initiated by writing the Load command to the Command Register. In Direct Control Mode, the command is initiated by raising the AUX_7-K/$\overline{D}$ control input while the AUX_5-S/$\overline{S}$ input is LOW. In this later case, the level on AUX_6-E/$\overline{D}$ determines which key register is written (HIGH = E Register).

Once the command has been recognized, the Command Pending bit (S_6 in the Status Register) will go to "1" and in Direct Control Mode AUX_3-$\overline{CP}$ will go active (LOW), indicating that key entry may proceed. The host system then writes exactly eight bytes to the Master Port (at the Input Register address in Multiplexed Control Mode). When the key register has been loaded, Command Pending will return to "0," and in Direct Control Mode the AUX_3-$\overline{CP}$ output will go inactive, indicating that the DCP can accept the next command.

Load Encrypted E Key Through Auxiliary Port (B1 Hex)

Load Encrypted D Key Through Auxiliary Port (B2)

Execution of these commands (in Multiplexed Control Mode only) is similar to the Load Clear E (D) Key Through Auxiliary Port, except that key bytes are first decrypted using the Electronic Code Book algorithm and the Master (M) key, and then loaded into the appropriate key register, after having passed through the parity check logic (see Parity Checking).

The Command Pending bit (S_6) will be "1" during the entire decrypt-and-load operation. In addition, the Busy bit (S_5) will be "1" during the actual decryption process.

Load Encrypted E Key Through Master Port (31 Hex)

Load Encrypted D Key Through Master Port (32)

These commands (in Multiplexed Control Mode only) are similar in effect to Load Clear E (D) Key Through Master Port, except that key bytes are initially decrypted using the Electronic Code Book algorithm and the Master (M) Key, and then loaded byte-by-byte into the target key register, after having passed through the parity check logic (see Parity Checking).

The Command Pending bit (S_6) will be "1" during the entire decrypt-and-load operation. In addition, the Busy bit (S_5) will be "1" during the actual decryption process.

Load Clear IVE Register Through Master Port (85 hex)

Load Clear IVD Register Through Master Port (84)

These commands (in Multiplexed Control Mode only) are virtually identical to Load Clear E (or D) Key Through Master Port except that the data written to the Input Register address is routed to the Encryption Initializing Vector (IVE) or Decryption Initializing Vector (IVD) Register instead of a key register, and no parity checking occurs. Command Pending (S_6) is a "1" during the entire loading process.

Load Encrypted IVE Register Through Master Port (A5 Hex)

Load Encrypted IVD Register Through Master Port (A4)

These commands are analogous to the Load Encrypted E (or D) Key Through Master Port commands. The data flow specifications set in the Mode Register are overridden and the eight vector bytes are decrypted using the Decryption (D) Key and the Electronic Code Book algorithm. The resulting clear vector bytes are loaded into the target Initializing Vector register, and no parity

checking occurs. The Busy bit (S_5) does not go to "1" during the decryption process, but Command Pending (S_6) will be "1" during the entire decryption-and-load operation.

Read Clear IVE Register Through Master Port (8D Hex)

Read Clear IVD Register Through Master Port (8C)

The effect of these commands (in Multiplexed Control Mode only) is to override the data flow specifications set in the Mode Register and to connect the appropriate Initializing Vector Register to the Master Port at the Output Register address. In this state, each IV register appears as eight bytes of FIFO storage. The first byte of data will be available 6 clocks after the loading the command register. The command pending bit will be set to "1" and will remain a "1" until sometime after the eighth byte is read out. The host system has the responsibility to read out exactly eight bytes.

Read Encrypted IVE Register Through Master Port (A9 Hex)

Read Encrypted IVD Register Through Master Port (A8)

The effect of these commands (in Multiplexed Control Mode only) is to override the specifications set in the Mode Register and to encrypt the contents of the specified Initializing Vector Register using the Electronic Code Book algorithm and the Encrypt (E) Key. The resulting cipher text is placed in the Output Register, from which it can be read out as eight bytes through the Master Port. During the actual encryption process the Busy bit (S_5) will be "1." When Busy goes to "0," the encrypted vector bytes are ready to be read out. Command Pending (S_6) will be "1" during the entire encryption-and-output process, and will go to "0" when the eighth byte is read out. The host system is responsible for reading out exactly eight bytes.

Encrypt with Master (M) Key (39 Hex)

This command in Multiplexed Control Mode only, overrides the data flow specifications set in the Mode Register and causes the DCP to accept eight bytes from the Master Port, written to the Input Register. When eight bytes have been received, the DCP encryptes the input using the Master (M) Key. The encrypted data is loaded into the Output Register, where it may be read out through the Master Port. The Command Pending (S_6) and Busy (S_5) bits are used to sense the three phases of this operation. Command Pending goes to "1" as soon as the Input Register can accept data. When exactly eight bytes have been entered, the Busy bit will go to "1" until the encryption process is complete. When Busy goes to "0," the encrypted data is available to be read out. Command Pending will return to "0" when the eighth byte has been read.

Start Encryption (41 Hex)

Start Decryption (40)

Start (C0)

The three "Start" commands begin normal data ciphering by setting the Start/Stop bit (S_7) in the Status Register to "1." The Start Encryption and Start Decryption commands explicitly specify the ciphering direction by forcing the Encrypt/Decrypt bit (M_4) in the Mode Register to "1" of "0," respectively, whereas Start uses the current state of the Encrypt/Decrypt bit, as specified in a previous Mode Register load.

When a Start command has been entered, the Port Status Flag ($\overline{MFLG}$ or $\overline{SFLG}$) associated with the Input Register will become active (LOW), indicating that data may be written to the Input Register to begin ciphering.

In Direct Control Mode, the Start command is issued by raising the level on the AUX_5-S/$\overline{S}$ input (see Figure 3). The ciphering direction is specified by the level on AUX_6-E/$\overline{D}$. If AUX_6-E/$\overline{D}$ is high when AUX_5-S/$\overline{S}$ goes HIGH, the command is Start Encryption; if AUX_6-E/$\overline{D}$ is low, it is Start Decryption.

Stop (E0 Hex)

The Stop command clears the Start/Stop bit (S_7) in the Status Register to "0." This causes the input flag ($\overline{MFLG}$ or $\overline{SFLG}$) to become inactive and inhibits the loading of any further input into the algorithm unit. If ciphering is in progress (Busy bit (S_5) is "1" or AUX_2-$\overline{BSY}$ is active), it will finish and any data in the Output Register will remain accessible.

In Direct Control Mode, the Stop command is implied when the signal level on the AUX_5-S/$\overline{S}$ input goes from HIGH to LOW (see Figure 3).

Software Reset (00)

This command has the same effect as a hardware reset ($\overline{MAS}$ and $\overline{MDS}$ low): it forces the DCP back to its default configuration, and all processing flags go into inactive mode. The default configuration includes setting the Mode Register to Electronic Code Book cipher type, and Dual Port Configuration with Master Port clear, Slave Port encrypted.

INTERFACE SIGNAL DESCRIPTION

V_{CC}: +5V power supply
V_{SS}: ground (2 pins)

CLK (Clock, Input, TTL levels)

An external timing source is input via the CLK pin. The Master and Slave Port Data Strobe signals, $\overline{MDS}$, $\overline{SDS}$ and also AUX_5-S/$\overline{S}$ in Direct Control Mode (C/$\overline{K}$ HIGH) must change synchronously with this clock input. In addition, the Auxiliary, Master and Slave Port Flag outputs ($\overline{AFLG}$, $\overline{MFLG}$ and $\overline{SFLG}$) will change synchronously with the clock. When using the DCP with the AmZ8000 in multiplexed control mode, the clock input must agree in frequency and phase with the processor clock; however, the DCP does not require the high voltage levels of the processor clock.

C/$\overline{K}$ (Control/Key Mode Control, Input)

This input is the primary control over the operating characteristics of the DCP. A LOW input on C/$\overline{K}$ places the DCP into Multiplexed Control Mode, enabling programmed access to internal registers through the Master Port and enabling input of keys through the Auxiliary Port. A HIGH input on C/$\overline{K}$ specifies operation in Direct Control Mode, wherein several of the Auxiliary Port pins become direct control/status signals which can be driven/sensed by high-speed controller logic (such as the Am29116 or Am2901/Am2903-based processors), and access to internal registers through the Master Port is limited to the Input or Output Register.

MP_0-MP_7 (Master Port Bus, Input/Output)

These eight bidirectional lines are used to specify internal register addresses in Multiplexed Control Mode (see C/$\overline{K}$) and to input and output data. The Master Port provides software access to the Status, Command and Mode Registers, as well as the Input and Output Registers. The three-state Master Port outputs will be enabled only when the Master Port is selected by Master Port Chip Select ($\overline{MCS}$) LOW, with Master Port Read/Write (MR/$\overline{W}$) HIGH, and strobed by Master Port Data Strobe ($\overline{MDS}$) LOW. MP_0 is the low-order bit. Data and key information entered is through into this port with most significant byte in first.

$\overline{MCS}$ (Master Port Chip Select, Input)

This active LOW input signal is used to select the Master Port. In Multiplexed Control Mode (C/$\overline{K}$ low), the level on $\overline{MCS}$ is latched internally on the rising edge of Master Port Address Strobe ($\overline{MAS}$). This latched level is retained as long as $\overline{MAS}$ is HIGH; when $\overline{MAS}$ is LOW, the latch becomes invisible and the internal signal will follow the $\overline{MCS}$ input. In Direct Control Mode (C/$\overline{K}$ HIGH), no latching of Master Port Chip Select occurs; the level on $\overline{MCS}$ is passed directly to the internal select circuitry irrespective of state of Address Strobe ($\overline{MAS}$).

$\overline{MAS}$ (Master Port Address Strobe, Input)

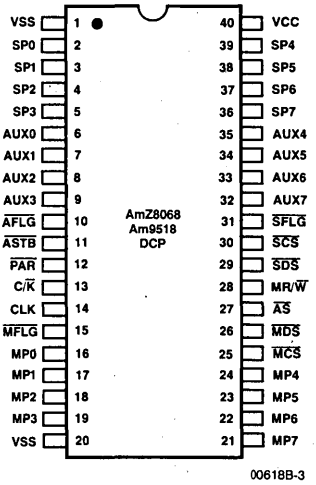
In Multiplexed Control Mode (C/$\overline{K}$ low), an active LOW signal on this pin indicates the presence of valid address and chip select information at the Master Port. This information will be latched internally on the rising edge of Address Strobe. When C/$\overline{K}$ is HIGH (Direct Control Mode), $\overline{MAS}$ may be HIGH or LOW without affecting DCP operation, except that, irregardless of C/$\overline{K}$ state, if both Master Port Address Strobe ($\overline{MAS}$) and Data Strobe ($\overline{MDS}$) are LOW simultaneously, the DCP will be reset to ECB mode and all flags inactive.

$\overline{MDS}$ (Master Port Data Strobe, Input)

This active LOW input is used in coincidence with a valid Master Port Chip Select ($\overline{MCS}$), to indicate that valid data is present on MP_0-MP_7 for an input operation or that data is to be placed on MP_0-MP_7 during output. Master Port Data Strobe and Address Strobe ($\overline{MAS}$) are normally mutually exclusive; if both go LOW simultaneously, the DCP is reset to ECB mode and all flags inactive.

CONNECTION DIAGRAM
Top View

D-40



Note: Pin 1 is marked for orientation.

MR/$\overline{W}$ (Master Port Read/Write, Input)

This input signal indicates to the DCP whether the current Master Port operation is a read (HIGH) or a write (LOW), thereby indicating that data is to be transferred from or to an internal register, respectively. MR/$\overline{W}$ is not latched internally and must be held stable while Master Port Data Strobe ($\overline{MDS}$) is LOW.

$\overline{MFLG}$ (Master Port Flag, Output)

This active LOW flag is used to indicate the need for a data transfer into or out of the Master Port during normal ciphering operation. Depending upon control bits written to the Mode Register (see Register Description), the Master Port will be associated with either the Input Register or the Output Register.

If data is to be transferred through the Master Port to the Input Register, the $\overline{MFLG}$ reflects the contents of the Input Register; after any Start command is entered, $\overline{MFLG}$ will go active (LOW) whenever the Input Register is not full. $\overline{MFLG}$ is forced HIGH by any command other than a Start. Conversely, if the Master Port is associated with the Output Register, $\overline{MFLG}$ reflects the contents of the Output Register (except in Single Port configuration; see Functional Description). $\overline{MFLG}$ will go active (LOW) whenever the Output Register is not empty. In Single Port Configuration, the Master Port Flag reflects the contents of the Input Register, while the Slave Port Flag ($\overline{SFLG}$, see below) is associated with the Output Register.

SP_0-SP_7 (Slave Port Bus, Bidirectional)

The Slave Port provides a second data input/output interface to the DCP, allowing overlapped input, output and ciphering operations. The tri-state Slave Port outputs will be driven only when Slave Port Chip Select ($\overline{SCS}$) and Slave Port Data Strobe ($\overline{SDS}$)

are both LOW and $\overline{SFLG}$ = 0, and the internal Port Control Configuration allows output to the Slave Port. SP_0 is the LOW order bit. Data entered or retrieved through this port is most significant byte in/out first.

$\overline{SCS}$ (Slave Port Chip Select, Input)

This active LOW signal is logically combined with Slave Port Data Strobe ($\overline{SDS}$) to facilitate Slave Port data transfers in a bus environment. $\overline{SCS}$ is not latched internally, and may be tied permanently LOW without impairing Slave Port operation.

$\overline{SDS}$ (Slave Port Data Strobe, Input)

This active LOW input, in coincidence with Slave Port Chip Select ($\overline{SCS}$) LOW, indicates to the DCP that valid data is on the SP_0-SP_7 lines for an input operation, or that data is to be driven onto the SP_0-SP_7 lines for output. The direction of data flow is determined by control bits in the Mode Register (see Register Description).

$\overline{SFLG}$ (Slave Port Flag, Output)

This active LOW output indicates the state of either the Input Register or the Output Register, depending on control bits in the Mode Register. In Single Port Configuration, $\overline{SFLG}$ will go active whenever the Output Register is not empty during normal processing. In Dual Port Configuration, $\overline{SFLG}$ will reflect the content of whichever register is associated with the Slave Port. If the Input Register is assigned to the Slave Port, $\overline{SFLG}$ will go active whenever the Input Register is not full, once any of the Start commands has been entered; $\overline{SFLG}$ will be forced inactive if any other command is entered. Conversely, if the Slave Port is assigned to the Output Register, $\overline{SFLG}$ will go active whenever the Output Register is not empty.

AUX_0-AUX_7 (Auxiliary Port Bus, Bidirectional)

When the DCP is operated in Multiplexed Control Mode (C/$\overline{K}$ LOW), these eight lines form a key-byte input port which may be used to enter the Master and Session Keys. In fact, this port is the only path available for entering the Master Key. (Session Keys may alternatively be entered via the Master Port.) AUX_0 is the low-order bit, and is considered to be the parity bit in key bytes. Most significant byte entered first.

When the DCP is operated in Direct Control Mode, (C/$\overline{K}$ HIGH, the Auxiliary Port's key-entry function is disabled and five of the eight lines become direct control/status lines for interfacing to high-speed microprogrammed controllers. In this case, AUX_0, AUX_1 and AUX_4 have no function, and the other pins are defined as follows:

AUX_5-S/$\overline{S}$ (Start/$\overline{Stop}$, Input)

When this pin goes LOW (Stop) the DCP will follow the sequence that would normally occur were a Stop command to be entered. Conversely, when this pin goes HIGH, a sequence equivalent to a Start Encryption or Start Decryption command will be followed. At the time AUX_5-S/$\overline{S}$ goes HIGH, the level on AUX_6-E/$\overline{D}$ (see below) selects either the Start Encryption or Start Decryption interpretation.

AUX_7-K/$\overline{D}$ (Key/$\overline{Data}$, Input)

When this signal goes HIGH, the DCP initiates a key-data input sequence as if a Load Clear E or D Key through Master Port command had been entered. The level on AUX_6-E/$\overline{D}$ will determine whether the subsequently entered clear-key bytes are written into the E Key Register (E/$\overline{D}$ HIGH) or the D Key Register (E/$\overline{D}$ LOW).

AUX_7-K/$\overline{D}$ and AUX_5-S/$\overline{S}$ are mutually exclusive control lines; when one goes active (HIGH), the other must be and remain inactive (LOW) until the first returns to an inactive state. In addition, both lines must be inactive (LOW) whenever a transition occurs on C/$\overline{K}$ (entering or exiting Direct Control Mode).

AUX_6-E/$\overline{D}$ (Encrypt/$\overline{Decrypt}$, Input)

When AUX_5-S/$\overline{S}$ goes HIGH, initiating a normal data ciphering operation, this input specifies whether the ciphering algorithm is to encrypt (E/$\overline{D}$ HIGH) or decrypt (LOW).

When AUX_7-K/$\overline{D}$ goes HIGH, initiating entry of key bytes, the level on AUX_6-E/$\overline{D}$ specifies whether the bytes are to be written into the E Key Register (E/$\overline{D}$ HIGH) or the D Key Register (E/$\overline{D}$ LOW).

The AUX_6-E/$\overline{D}$ input is not latched internally, and must be held constant whenever one or more of AUX_5-S/$\overline{S}$, AUX_7-K/$\overline{D}$, AUX_2-$\overline{BSY}$, or AUX_3-$\overline{CP}$ are active. Failure to maintain the proper level on AUX_6-E/$\overline{D}$ during loading or ciphering operations will result in scrambled data in the internal registers.

AUX_2-$\overline{BSY}$ (Busy, Output)

This active LOW status output gives a hardware indication that the ciphering algorithm is in operation. AUX_2-$\overline{BSY}$ is driven by the BSY bit in the Status Register (see Register Description), such that when the BSY bit is "1" (active), AUX_2-$\overline{BSY}$ is LOW.

AUX_3-$\overline{CP}$ (Command Pending Output)

This active LOW status output gives a hardware indication that the DCP is ready to accept input of key bytes following a LOW-to-HIGH transition on AUX_7-K/$\overline{D}$. AUX_3-$\overline{CP}$ is driven by the CP bit in the Status Register, such that when the CP bit is "1" (active), AUX_3-$\overline{CP}$ is LOW.

$\overline{ASTB}$ (Auxiliary Port Strobe, Input)

The rising (trailing) edge of $\overline{ASTB}$ strobes the key data on pins AUX_0-AUX_7 into the appropriate internal key register in Multiplexed Control Mode (C/$\overline{K}$ LOW). This input is ignored unless $\overline{AFLG}$ and C/$\overline{K}$ are both LOW. One byte of key data is entered on each $\overline{ASTB}$, Most Significant byte first.

$\overline{AFLG}$ (Auxiliary Port Flag, Output)

This active LOW output signal indicates that the DCP is expecting key data to be entered on pins AUX_0-AUX_7. This can occur only when C/$\overline{K}$ is LOW and a Load Key Through AUX Port command has been entered. $\overline{AFLG}$ will remain active (LOW) during input of all eight bytes, and will go inactive with the leading edge of the eighth strobe ($\overline{ASTB}$).

$\overline{PAR}$ (Parity, Output)

The DCP checks all key bytes for correct (odd) parity as they are entered through either the Master Port (Multiplexed or Direct Control Mode) or the Auxiliary Port (Multiplexed Control Mode only). If any key byte contains even parity, the PAR bit in the Status Register is set to "1" and $\overline{PAR}$ goes LOW. (See Parity Checking of Keys.) Least significant bit of key data is the parity.

MAXIMUM RATINGS (Above which useful life may be impaired)

Storage Temperature	−65 to +150°C
Ambient Temperature Under Bias	0 to +70°C
Voltage on Any Pin with Respect to Ground	−0.5 to +7.0V
Power Dissipation	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

AmZ8068, Am9518 ELECTRICAL CHARACTERISTICS (over operating range unless otherwise specified)

T_A = 0 to 70°C, V_{CC} = +5.0V ±5%, V_{SS} = 0V

Parameters	Description	Test Conditions	Min	Typ	Max	Units
V_{IL}	Input Low Voltage		−0.5		.8	Volts
V_{IH}	Input High Voltage		2.2		V_{CC}	Volts
V_{OL}	Output Low Voltage	I_{OL} = 3.2mA			.40	Volts
V_{OH}	Output High Voltage	I_{OH} = −400μA	2.4			Volts
I_I	Input Leakage Current	$V_{SS} \leq V_{IN} \leq V_{CC}$			±10	μA
I_{OZ}	Output Leakage Current	$V_{SS} + .40 \leq V_{IN} \leq V_{CC}$			±10	μA
I_{CC}	Supply Current (AVER.)			150	250	mA

AmZ8068/Am9518 SWITCHING CHARACTERISTICS (Note 1)

The table below specifies the guaranteed performance of this device over the commercial operating range of 0 to +70°C with V_{CC} from 4.75 to 5.25V. All data are in nanoseconds. Switching tests are made with inputs and outputs measured at 0.8V for a LOW and 2.0V for a HIGH. Outputs are fully loaded, with $C_L \geq$ 50pF. See switching waveform figures following table for graphic illustration of timing parameters.

SWITCHING CHARACTERISTICS over operating range

Parameter	Number	Description	AmZ8068			Am9518			Units
			Min	Typ	Max	Min	Typ	Max	
Clock									
TWH	1	Clock Width (HIGH)	115			150			ns
TWL	2	Clock Width (LOW)	115			150			ns
TC	3	Clock HIGH to Next Clock HIGH (Clock Cycle)	250		1000	320		1000	ns
Reset									
TG1LG1H	5	$\overline{MDS} \cdot \overline{MAS}$ LOW to $\overline{MDS} \cdot \overline{MAS}$ HIGH (Reset Pulse Width)	TC			TC			ns
TCHG1H	6	Clock HIGH to $\overline{MDS} \cdot \overline{MAS}$ HIGH	0		50	0		50	ns
Direct Control Mode									
TNLMH	9	S/$\overline{S}$ LOW to C/$\overline{K}$ HIGH (Setup)	3TC			3TC			ns
TKLMH	10	K/$\overline{D}$ LOW to C/$\overline{K}$ HIGH (Setup)	3TC			3TC			ns
TMHNH	11	C/$\overline{K}$ HIGH to S/$\overline{S}$ HIGH	6TC			6TC			ns
TMHKH	12	C/$\overline{K}$ HIGH TO K/$\overline{D}$ HIGH	6TC			6TC			ns
TEVKH	14	E/$\overline{D}$ VALID to K/$\overline{D}$ HIGH (Setup)	3TC			3TC			ns
TKHRL	15	K/$\overline{D}$ HIGH to $\overline{CP}$ LOW			300			300	ns
TKLEX	17	K/$\overline{D}$ LOW to E/$\overline{D}$ INVALID (Hold)	TC			TC			ns
TCLNV	19	Clock LOW to S/$\overline{S}$ VALID	20		80	20		80	ns
TEVNH	20	E/$\overline{D}$ VALID to S/$\overline{S}$ HIGH (Setup)	3TC			3TC			ns
TNHF1L	21	S/$\overline{S}$ HIGH to $\overline{MFLG}$ ($\overline{SFLG}$) LOW (Port Input Flag)			230			300	ns
TCHF1L	22	Clock HIGH to $\overline{MFLG}$ ($\overline{SFLG}$) LOW (Port Input Flag) (Note 2)			230			300	ns
TCHBL	24	Clock HIGH to $\overline{BSY}$ LOW			300			400	ns
TCLBH	25	Clock LOW to $\overline{BSY}$ HIGH			230			300	ns
TCHF1L	27	Clock HIGH to $\overline{MFLG}$ ($\overline{SFLG}$) LOW (Port Output Flag)			230			300	ns
TNLF1H	28	S/$\overline{S}$ LOW to $\overline{MFLG}$ ($\overline{SFLG}$) HIGH (Port Input Flag) (Note 3)			230			300	ns
Multiplexed Control Mode – Master Port									
TWA	32	$\overline{MAS}$ Width (LOW)	80			115			ns
TS1LAH	34	$\overline{MCS}$ LOW to $\overline{MAS}$ HIGH (Setup)	0			0			ns
TAHS1H	35	$\overline{MAS}$ HIGH to $\overline{MCS}$ HIGH (Hold)	60			60			ns
TD1VAH	36	Address-In VALID to $\overline{MAS}$ HIGH (Address Setup Time)	55			90			ns
TAHD1X	37	$\overline{MAS}$ HIGH to Address-In INVALID (Address Hold Time)	60			60			ns

AC SWITCHING CHARACTERISTICS

Parameter	Number	Description		AmZ8068 Min	AmZ8068 Typ	AmZ8068 Max	Am9518 Min	Am9518 Typ	Am9518 Max	Units
Master (Slave) Port Read/Write										
TS1LG1L	40	$\overline{MCS}$ ($\overline{SCS}$) LOW to $\overline{MDS}$ ($\overline{SDS}$) LOW (Select Setup) (Note 4)		100			100			ns
TG1HS1H	41	$\overline{MDS}$ ($\overline{SDS}$) HIGH to $\overline{MCS}$ ($\overline{SCS}$) HIGH (Select Hold Time) (Note 4)		25			25			ns
TWVG1L	42	MR/$\overline{W}$ VALID to $\overline{MDS}$ LOW (Setup)		100			100			ns
TG1HWX	43	$\overline{MDS}$ HIGH to MR/$\overline{W}$ INVALID (Hold)		25			25			ns
TG1LG1H	44	$\overline{MDS}$ ($\overline{SDS}$) LOW to $\overline{MDS}$ ($\overline{SDS}$) HIGH	Width – Write, Data Read	125		1000	160		1000	ns
			Width – Status Register Read	200		1000	300		1000	
TCLG1H	45	Clock LOW to $\overline{MDS}$ ($\overline{SDS}$) HIGH (Note 11)		0		TWL – 65	0		TWL – 100	
TGIHG1L	46	$\overline{MDS}$ ($\overline{SDS}$) HIGH to $\overline{MDS}$ ($\overline{SDS}$) LOW (Data Strobe Recovery Time)		125			160			ns
TD1VG1H	47	Write-Data VALID $\overline{MDS}$ ($\overline{SDS}$) HIGH	Setup Time – Key Load (Note 8)	125			160			ns
			Setup Time – Data Write	125			160			
			Setup Time – Command/ Mode Register Write	125			160			
TG1HD1X	48	$\overline{MDS}$ ($\overline{SDS}$) HIGH to Write-Data INVALID (Hold Time – All Writes)		25			25			ns
TG1LQ1V	49	$\overline{MDS}$ ($\overline{SDS}$) LOW to Read-Data VALID	Read Access Time – Status Register			200			300	ns
			Read Access Time – Data			120			150	
TG1HQ1V	50	$\overline{MDS}$ ($\overline{SDS}$) HIGH to Read-Data INVALID (Read Hold Time)		5			5			ns
TG1LF1H	51	$\overline{MDS}$ ($\overline{SDS}$) LOW to $\overline{MFLG}$ ($\overline{SFLG}$) HIGH (Last Strobe) (Note 5)				125			160	ns
TG1LRH	52	$\overline{MDS}$ HIGH to CP HIGH	Last Strobe, Key Load			TC+500			TC+500	ns
TG1HNL	53	$\overline{MDS}$ ($\overline{SDS}$) HIGH to S/$\overline{S}$ LOW (Hold Time) (Note 9)		4TC			4TC			ns
TG1HPV	54	$\overline{MDS}$ HIGH to $\overline{PAR}$ VALID (Key Write)				200			250	ns
Auxiliary Port Key Entry										
TG3LG3H	61	$\overline{ASTB}$ LOW to $\overline{ASTB}$ HIGH (Width)		160			160			ns
TCLG3H	62	Clock LOW to $\overline{ASTB}$ HIGH		0		50	0		50	ns
TG3HG3L	63	$\overline{ASTB}$ HIGH to Next $\overline{ASTB}$ LOW (Recovery Time)		250			320			ns
TD3VG3H	64	Write-Data VALID to $\overline{ASTB}$ HIGH (Data Setup Time)		200			300			ns
TG3HD3X	65	$\overline{ASTB}$ HIGH to Write-Data INVALID (Data Hold Time)		80			80			ns
TG3HPV	66	$\overline{ASTB}$ HIGH to $\overline{PAR}$ VALID				200			300	ns
TG3LF3H	67	$\overline{ASTB}$ LOW to $\overline{AFLG}$ HIGH (Last Strobe)				230			300	ns

Notes:
1. All input transition times assumed ≤20ns.
2. Parameter TCHF1L applies to all input blocks except the first (when S/$\overline{S}$ first goes HIGH).
3. When S/$\overline{S}$ goes inactive (LOW) in direct control mode, the flag associated with the input port will turn off.
4. Direct control mode only.
5. In Cipher Feedback, the port flag ($\overline{MFLG}$ or $\overline{SFLG}$) will go inactive following the leading edge of the first data strobe ($\overline{MDS}$ or $\overline{SDS}$); in all other modes and operations, the flags go inactive on the eighth data strobe.
6. Do not remove K/$\overline{D}$ until $\overline{CP}$ is inactive (HIGH).
7. Do not change E/$\overline{D}$ until $\overline{MFLG}$ ($\overline{SFLG}$) is inactive (HIGH).
8. 300ns Min if parity check is needed.
9. In Cipher Feedback mode $\overline{BSY}$ must be inactive before S/$\overline{S}$ goes LOW.
10. $\overline{AFLG}$ must go active (LOW) before $\overline{ASTB}$ goes active (LOW).
11. This limit is valid when the clock frequency is 4MHz. At slower clock rates, the range is wider.

Parameter Naming Convention for DCP

Name: T A [N] B C [N] D

A C Signal names (see below)

B D Signal States:

- H High
- L Low
- V Valid
- X Not Valid
- Z High Impedance

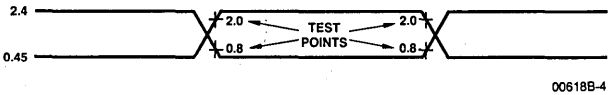
(N) Optional Port number (modifies signal name):

- 1 = Master Port
- 2 = Slave Port
- 3 = AUX (Key) Port

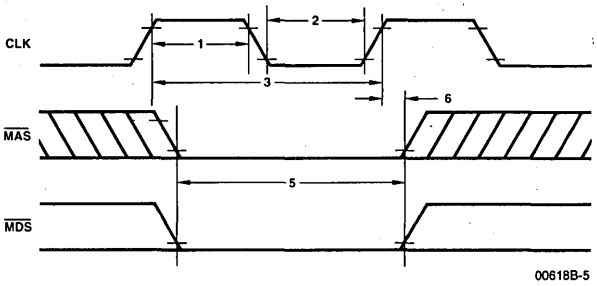
Signal Name Characters

A	Address Strobe
B	$\overline{BSY}$
C	Clock
D*	Data In (or address at Master Port) D1, D2, D3
E	E/$\overline{D}$
F*	Flag ($\overline{MFLG}$, $\overline{SFLG}$, $\overline{AFLG}$)
G*	Data Strobe ($\overline{MDS}$, $\overline{SDS}$, $\overline{ASTB}$)
K	K/$\overline{D}$
M	C/$\overline{K}$ (Mode)
N	S/$\overline{S}$ (Start)
P	$\overline{PAR}$
Q*	Data Out (Master or Slave Port)
R	$\overline{CP}$
S*	Chip Select (Master or Slave Port)
W	MR/$\overline{W}$

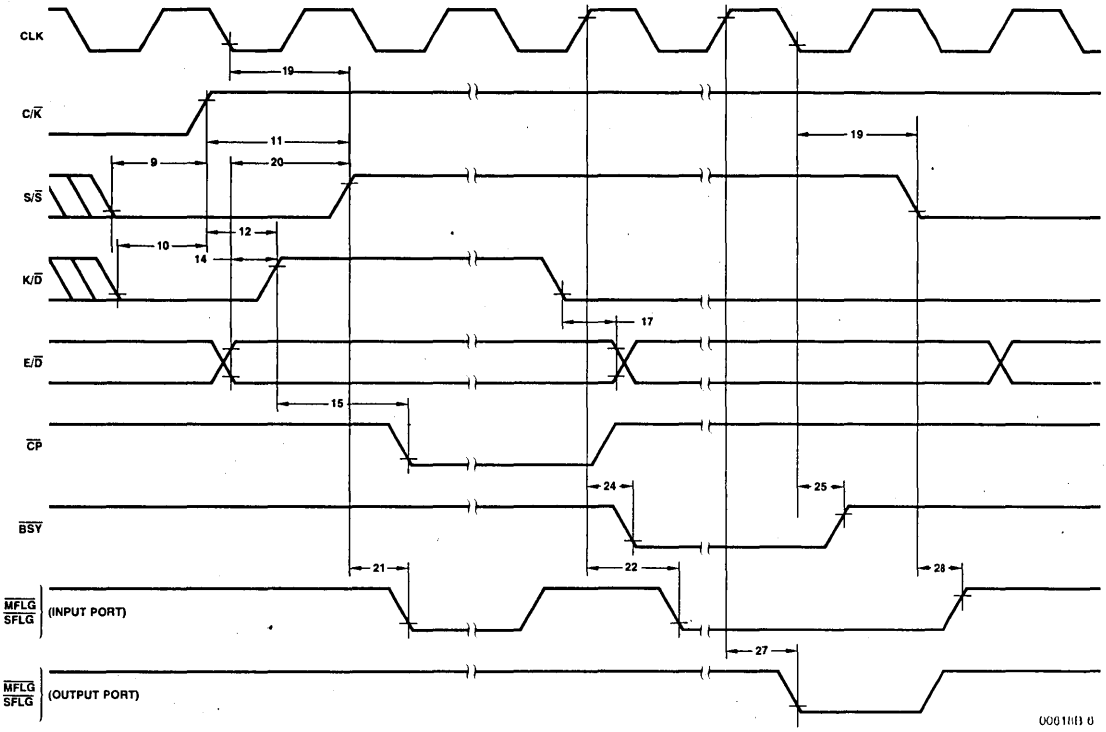
*Modified by Port number. Example: D1 = Data In, Master Port; F2 = SFLG; G3 = ASTB; Q2 = Data Out, Slave Port; S1 = MCS.



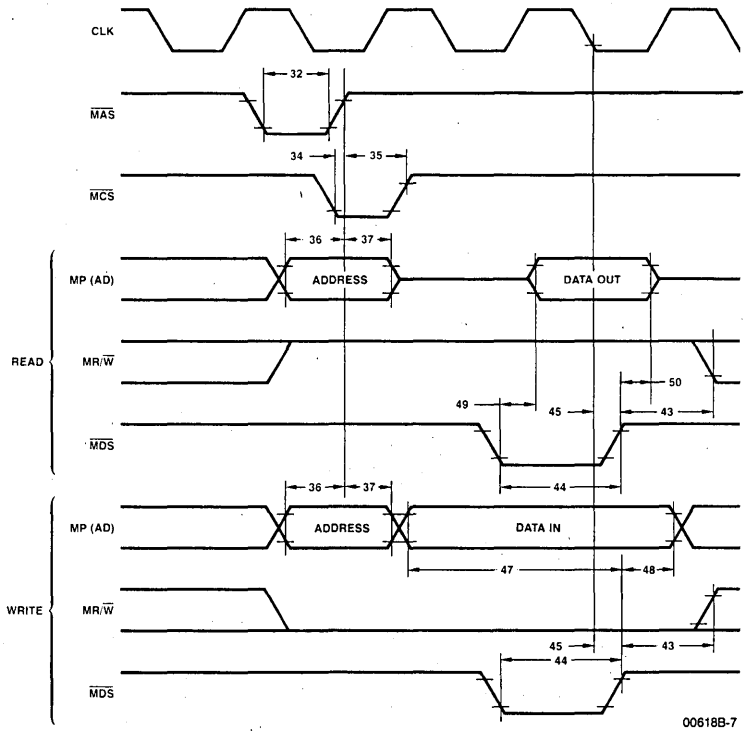
INPUT WAVEFORMS FOR A.C. TESTS



CLOCK AND RESET

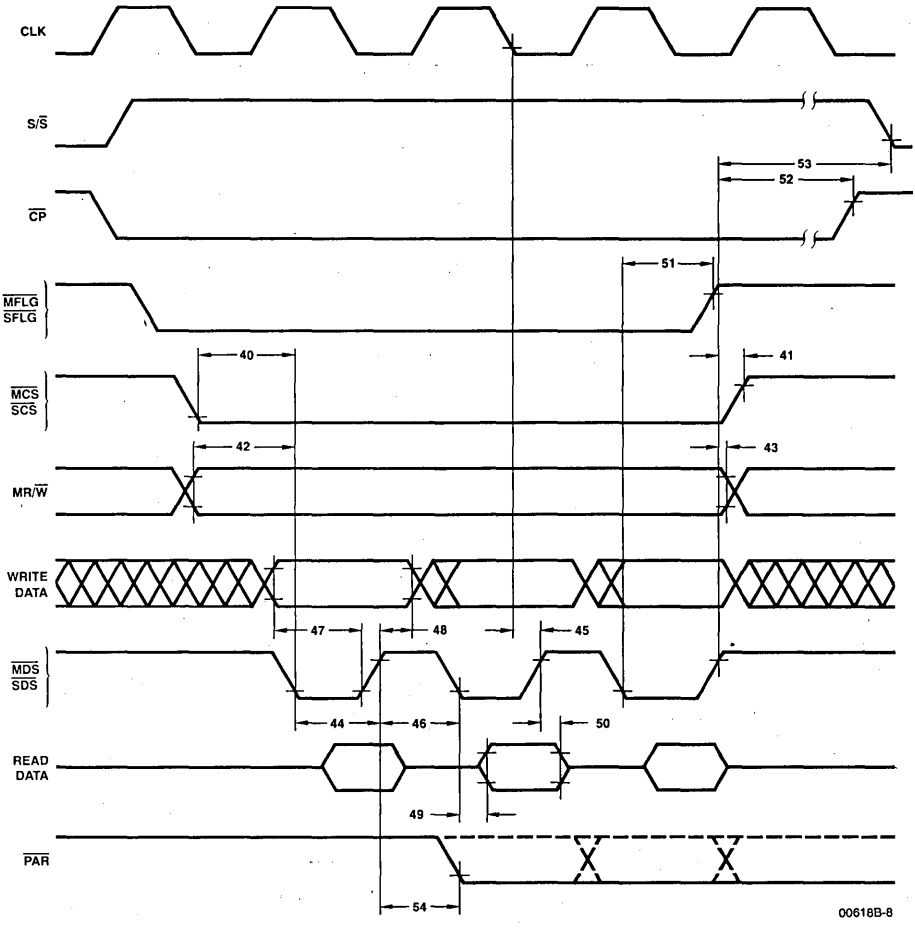


CONTROL AND STATUS SIGNALS (DIRECT CONTROL MODE)

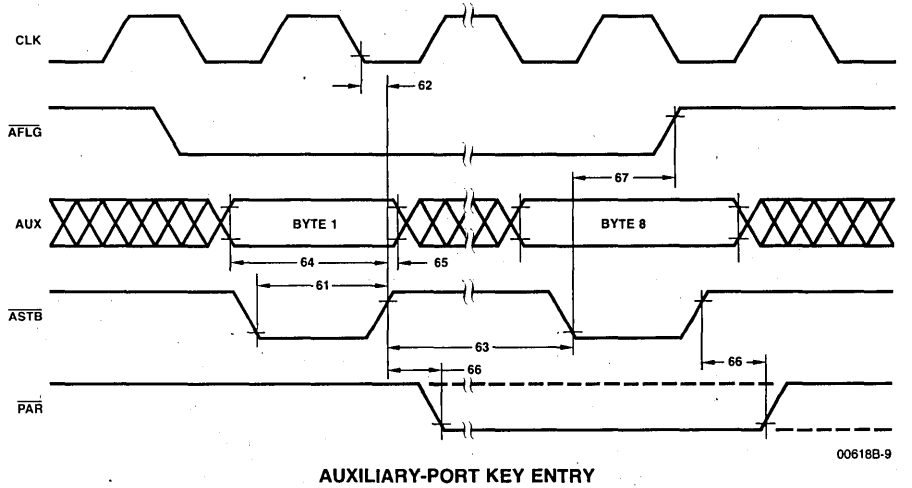


00618B-7

MASTER PORT, MULTIPLEXED CONTROL MODE READ/WRITE



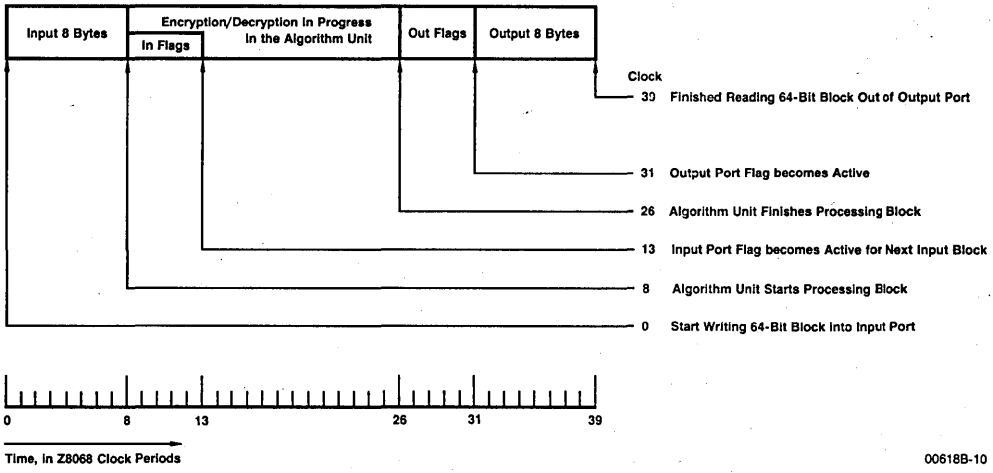
MASTER (SLAVE) PORT READ/WRITE



AUXILIARY-PORT KEY ENTRY

TIMING FOR PIPELINED, DUAL-PORT OPERATION

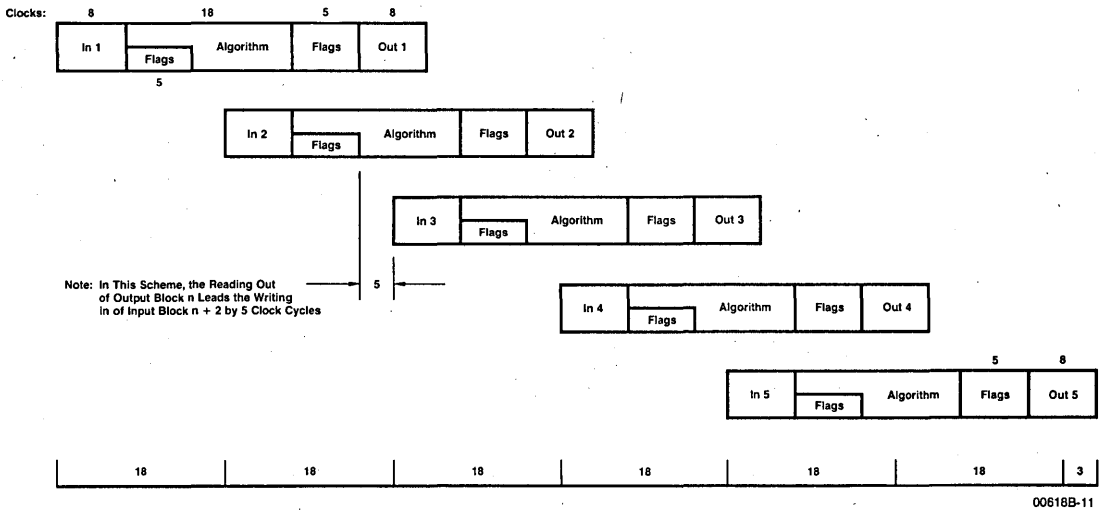
Detailed Timing of 1 Block



00618B-10

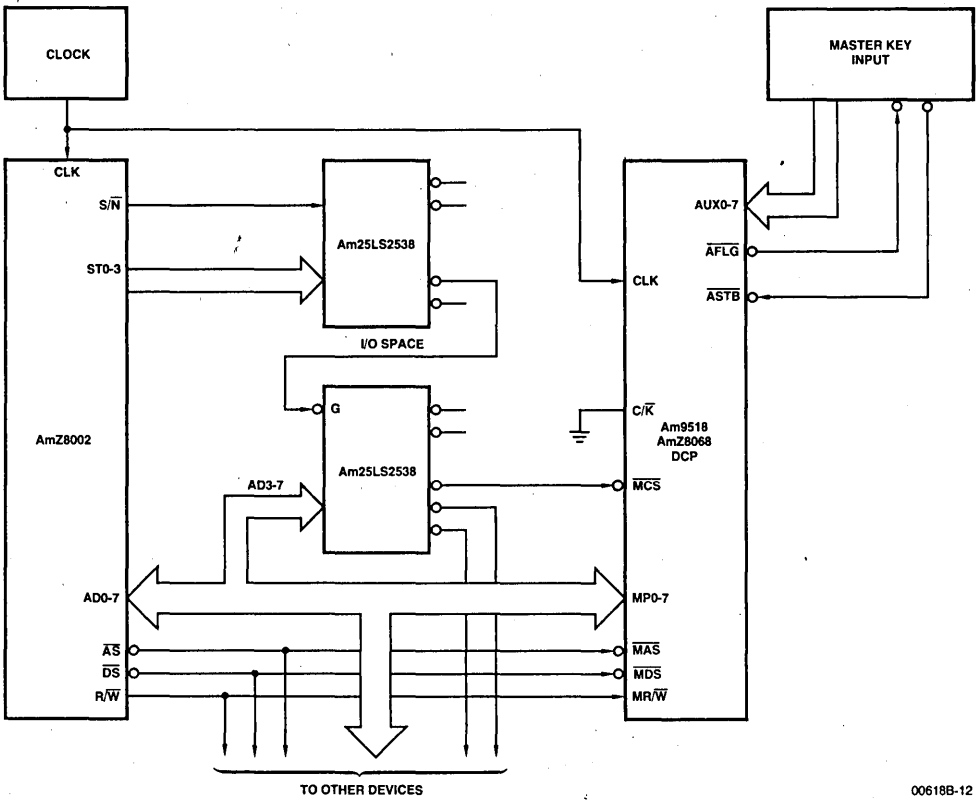
Note: AmZ8068 clock period = 250 nanoseconds

Pipelining Scheme A: Minimum Timing Operation

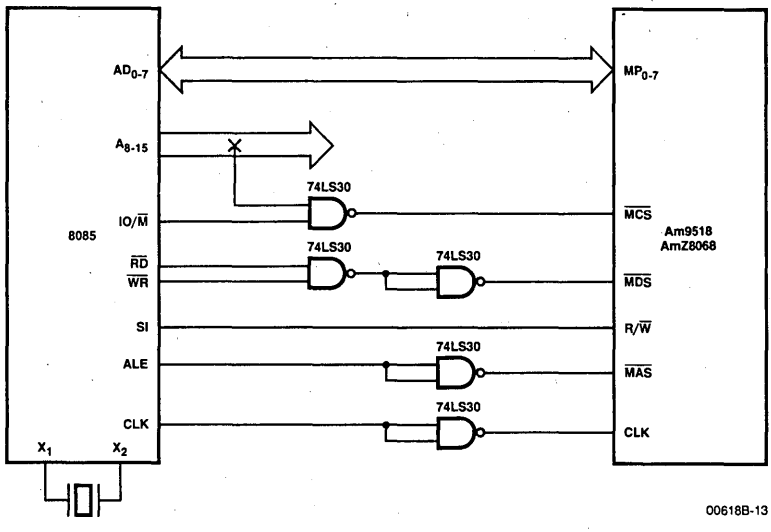


00618B-11

For n blocks, total time = (n + 1) x 18 + 3



MINIMUM AmZ8068/Am9518 INTERFACE



MINIMUM 8085 TO AmZ8068/Am9518 INTERFACE

8231A

Arithmetic Processor

DISTINCTIVE CHARACTERISTICS

- 2, 3 and 4MHz operation
- Fixed point 16 and 32 bit operations
- Floating point 32 bit operations
- Binary data formats
- Add, Subtract, Multiply and Divide
- Trigonometric and inverse trigonometric functions
- Square roots, logarithms, exponentiation
- Float to fixed and fixed to float conversions
- Stack-oriented operand storage
- DMA or programmed I/O data transfers
- End signal simplifies concurrent processing
- Synchronous/Asynchronous operations
- General purpose 8-bit data bus interface
- Standard 24 pin package
- +12 volt and +5 volt power supplies
- Advanced N-channel silicon gate MOS technology

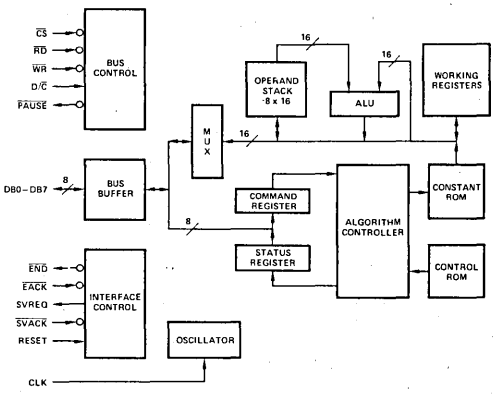
GENERAL DESCRIPTION

The 8231A Arithmetic Processing Unit (APU) is a monolithic MOS/LSI device that provides high performance fixed and floating point arithmetic and a variety of floating point trigonometric and mathematical operations. It may be used to enhance the computational capability of a wide variety of processor-oriented systems.

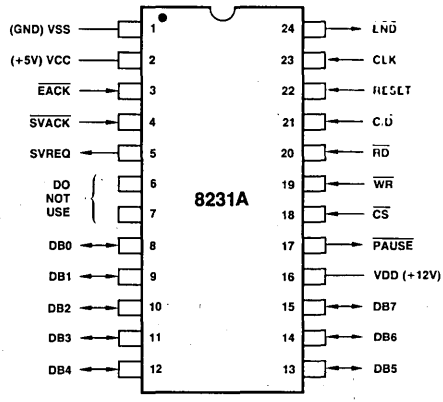
All transfers, including operand, result, status and command information, take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack and a command is issued to perform operations on the data in the stack. Results are then available to be retrieved from the stack, or additional commands may be entered.

Transfers to and from the APU may be handled by the associated processor using conventional programmed I/O, or may be handled by a direct memory access controller for improved performance. Upon completion of each command, the APU issues an end of execution signal that may be used as an interrupt by the CPU to help coordinate program execution.

BLOCK DIAGRAM



CONNECTION DIAGRAM
Top View
D-40



Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Package Type	Ambient Temperature	Maximum Clock Frequency		
		2MHz	3MHz	4MHz
Hermetic DIP	$0°C \leq T_A \leq +70°C$	D8231A-8	D8231A-3	D8231A
	$-40°C \leq T_A \leq +85°C$	ID8231A-8	ID8231A-3	
	$-55°C \leq T_A \leq +125°C$	MD8231A-8B	MD8231A-3B	

04101A-MMP

INTERFACE SIGNAL DESCRIPTION

VCC: +5V Power Supply
VDD: +12V Power Supply
VSS: Ground

CLK (Clock, Input)

An external timing source connected to the CLK input provides the necessary clocking. The CLK input can be asynchronous to the $\overline{RD}$ and $\overline{WR}$ control signals.

RESET (Reset, Input)

A HIGH on this input causes initialization. Reset terminates any operation in progress, and clears the status register to zero. The internal stack pointer is initialized and the contents of the stack may be affected but the command register is not affected by the reset operation. After a reset the $\overline{END}$ output will be HIGH, and the SVREQ output will be LOW. For proper initialization, the RESET input must be HIGH for at least five CLK periods following stable power supply voltages and stable clock.

C/$\overline{D}$ (Command/Data Select, Input)

The C/$\overline{D}$ input together with the $\overline{RD}$ and $\overline{WR}$ inputs determines the type of transfer to be performed on the data bus as follows:

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	Function
L	H	L	Push data byte into the stack
L	L	H	Pop data byte from the stack
H	H	L	Enter command byte from the data bus
H	L	H	Read Status
X	L	L	Undefined

L = LOW
H = HIGH
X = DON'T CARE

$\overline{END}$ (End of Execution, Output)

A LOW on this output indicates that execution of the current command is complete. This output will be cleared HIGH by activating the $\overline{EACK}$ input LOW or performing any read or write operation or device initialization using the RESET. If $\overline{EACK}$ is tied LOW, the $\overline{END}$ output will be a pulse (see $\overline{EACK}$ description). This is an open drain output and requires a pull up to +5V.

Reading the status register while a command execution is in progress is allowed. However any read or write operation clears the flip-flop that generates the $\overline{END}$ output. Thus such continuous reading could conflict with internal logic setting the $\overline{END}$ flip-flop at the completion of command execution.

$\overline{EACK}$ (End Acknowledge, Input)

This input when LOW makes the $\overline{END}$ output go HIGH. As mentioned earlier LOW on the $\overline{END}$ output signals completion of a command execution. The $\overline{END}$ output signal is derived from an internal flip-flop which is clocked at the completion of a command. This flip-flop is clocked to the reset state when $\overline{EACK}$ is LOW. Consequently, if the $\overline{EACK}$ is tied LOW, the $\overline{END}$ output will be a pulse that is approximately one CLK period wide.

SVREQ (Service Request, Output)

A HIGH on this output indicates completion of a command. In this sense this output is same as the $\overline{END}$ output. However, whether the SVREQ output will go HIGH at the completion of a command or not is determined by a service request bit in the command register. This bit must be 1 for SVREQ to go HIGH. The SVREQ can be cleared (i.e., go LOW) by activating the $\overline{SVACK}$ input LOW or initializing the device using the RESET. Also, the SVREQ will be automatically cleared after completion of any command that has the service request bit as 0.

$\overline{SVACK}$ (Service Acknowledge, Input)

A LOW on this input activates the reset input of the flip-flop generating the SVREQ output. If the $\overline{SVACK}$ input is permanently tied LOW, it will conflict with the internal setting of the flip-flop to generate the SVREQ output. Thus the SVREQ indication cannot be relied upon if the $\overline{SVACK}$ is tied LOW.

DB0-DB7 (Bidirectional Data Bus, Input/Output)

These eight bidirectional lines are used to transfer command, status and operand information between the device and the host processor. DB0 is the least significant and DB7 is the most significant bit position. HIGH on the data bus line corresponds to 1 and LOW corresponds to 0.

When pushing operands on the stack using the data bus, the least significant byte must be pushed first and most significant byte last. When popping the stack to read the result of an operation, the most significant byte will be available on the data bus first and the least significant byte will be the last. Moreover, for pushing operands and popping results, the number of transactions must be equal to the proper number of bytes appropriate for the chosen format. Otherwise, the internal byte pointer will not be aligned properly. The Am9511A single precision format requires 2 bytes, double precision and floating-point formats require 4 bytes.

$\overline{CS}$ (Chip Select, Input)

This input must be LOW to accomplish any read or write operation to the Am9511A.

To perform a write operation data is presented on DB0 through DB7 lines, C/$\overline{D}$ is driven to an appropriate level and the $\overline{CS}$ input is made LOW. However, actual writing into the Am9511A cannot start until $\overline{WR}$ is made LOW. After initiating the write operation by a $\overline{WR}$ HIGH to LOW transition, the $\overline{PAUSE}$ output will go LOW momentarily (TPPWW).

The $\overline{WR}$ input can go HIGH after $\overline{PAUSE}$ goes HIGH. The data lines, C/$\overline{D}$ input and the $\overline{CS}$ input can change when appropriate hold time requirements are satisfied. See write timing diagram for details.

To perform a read operation an appropriate logic level is established on the C/$\overline{D}$ input and $\overline{CS}$ is made LOW. The Read operation does not start until the $\overline{RD}$ input goes LOW. $\overline{PAUSE}$ will go LOW for a period of TPPWR. When $\overline{PAUSE}$ goes back HIGH again, it indicates that read operation is complete and the required information is available on the DB0 through DB7 lines. This information will remain on the data lines as long as $\overline{RD}$ input is LOW. The $\overline{RD}$ input can return HIGH anytime after $\overline{PAUSE}$ goes HIGH. The $\overline{CS}$ input and C/$\overline{D}$ inputs can change anytime after $\overline{RD}$ returns HIGH. See read timing diagram for details.

$\overline{RD}$ (Read, Input)

A LOW on this input is used to read information from an internal location and gate that information on to the data bus. The $\overline{CS}$ input must be LOW to accomplish the read operation. The C/$\overline{D}$ input determines what internal location is of interest. See C/$\overline{D}$, $\overline{CS}$ input descriptions and read timing diagram for details. If the END output was LOW, performing any read operation will make the $\overline{END}$ output go HIGH after the HIGH to LOW transition of the $\overline{RD}$ input (assuming $\overline{CS}$ is LOW).

$\overline{WR}$ (Write, Input)

A LOW on this input is used to transfer information from the data bus into an internal location. The $\overline{CS}$ must be LOW to accomplish the write operation. The C/$\overline{D}$ determines which internal location is to be written. See C/$\overline{D}$, $\overline{CS}$ input descriptions and write timing diagram for details.

If the $\overline{END}$ output was LOW, performing any write operation will make the $\overline{END}$ output go HIGH after the LOW to HIGH transition of the $\overline{WR}$ input (assuming $\overline{CS}$ is LOW).

$\overline{PAUSE}$ (Pause, Output)

This output is a handshake signal used while performing read or write transactions with the 8231A. A LOW at this output indicates that the 8231A has not yet completed its information transfer with the host over the data bus. During a read operation, after $\overline{CS}$ went LOW, the $\overline{PAUSE}$ will become LOW shortly (TRP) after $\overline{RD}$ goes LOW. $\overline{PAUSE}$ will return high only after the data bus contains valid output data. The $\overline{CS}$ and $\overline{RD}$ should remain LOW when $\overline{PAUSE}$ is LOW. The RD may go high anytime after $\overline{PAUSE}$ goes HIGH. During a write operation, after $\overline{CS}$ went LOW, the $\overline{PAUSE}$ will be LOW for a very short duration (TPPWN) after $\overline{WR}$ goes LOW. Since the minimum of TPPWW is 0, the $\overline{PAUSE}$ may not go LOW at all for fast devices. $\overline{WR}$ may go HIGH anytime after $\overline{PAUSE}$ goes HIGH.

FUNCTIONAL DESCRIPTION

Major functional units of the 8231A are shown in the block diagram. The 8231A employs a microprogram controlled stack oriented architecture with 16-bit wide data paths.

The Arithmetic Logic Unit (ALU) receives one of its operands from the Operand Stack. This stack is an 8-word by 16-bit 2-port memory with last in-first out (LIFO) attributes. The second operand to the ALU is supplied by the internal 16-bit bus. In addition to supplying the second operand, this bidirectional bus also carries the results from the output of the ALU when required. Writing into the Operand Stack takes place from this internal 16-bit bus when required. Also connected to this bus are the Constant ROM and Working Registers. The ROM provides the required constants to perform the mathematical operations (Chebyshev Algorithms) while the Working Registers provide storage for the intermediate values during command execution.

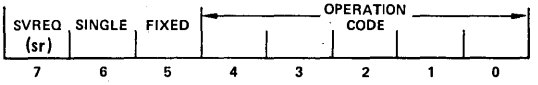
Communication between the external world and the Am9511A takes place on eight bidirectional input/output lines DB0 through DB7 (Data Bus). These signals are gated to the internal eight-bit bus through appropriate interface and buffer circuitry. Multiplexing facilities exist for bidirectional communication between the internal eight and sixteen-bit buses. The Status Register and Command Register are also accessible via the eight-bit bus.

The 8231A operations are controlled by the microprogram contained in the Control ROM. The Program Counter supplies the microprogram addresses and can be partially loaded from the Command Register. Associated with the Program Counter is the Subroutine Stack where return addresses are held during subroutine calls in the microprogram. The Microinstruction Register holds the current microinstruction being executed. This register facilitates pipelined microprogram execution. The Instruction Decode logic generates various internal control signals needed for the 8231A operation.

The Interface Control logic receives several external inputs and provides handshake related outputs to facilitate interfacing the 8231A to microprocessors.

COMMAND FORMAT

Each command entered into the 8231A consists of a single 8-bit byte having the format illustrated below:



Bits 0-4 select the operation to be performed as shown in the table. Bits 5-6 select the data format for the operation. If bit 5 is a 1, a fixed point data format is specified. If bit 5 is a 0, floating point format is specified. Bit 6 selects the precision of the data to be operated on by fixed point commands (if bit 5 = 0, bit 6 must be 0). If bit 6 is a 1, single-precision (16-bit) operands are indicated; if bit 6 is a 0, double-precision (32-bit) operands are indicated. Results are undefined for all illegal combinations of bits in the command byte. Bit 7 indicates whether a service request is to be issued after the command is executed. If bit 7 is a 1, the service request output (SVREQ) will go high at the conclusion of the command and will remain high until reset by a low level on the service acknowledge pin ($\overline{SVACK}$) or until completion of execution of a succeeding command where bit 7 is 0. Each command issued to the 8231A requests post execution service based upon the state of bit 7 in the command byte. When bit 7 is a 0, SVREQ remains low.

MAXIMUM RATINGS beyond which the useful life may be impaired

Storage Temperature	−65 to +150°C
V_{DD} with Respect to V_{SS}	−0.5 to +15.0V
V_{CC} with Respect to V_{SS}	−0.5 to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5 to +7.0V
Power Dissipation (Package Limitation)	2.0W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	Ambient Temperature	V_{SS}	V_{CC}	V_{DD}
D8231A-8	$0°C \leq T_A \leq 70°C$	0V	+5.0V ±5%	+12V ±5%
D8231A-3	$0°C \leq T_A \leq 70°C$	0V	+5.0V ±5%	+12V ±5%
D8231A	$0°C \leq T_A \leq 70°C$	0V	+5.0V ±5%	+12V ±5%
ID8231A-8	$-40°C \leq T_A \leq 85°C$	0V	+5.0V ±10%	+12V ±10%
ID8231A-3	$-40°C \leq T_A \leq 85°C$	0V	+5.0V ±10%	+12V ±10%
MD8231A-8	$-55°C \leq T_A \leq 125°C$	0V	+5.0V ±10%	+12V ±10%
MD8231A-3	$-55°C \leq T_A \leq 125°C$	0V	+5.0V ±10%	+12V ±10%

ELECTRICAL CHARACTERISTICS Over Operating Range (Note 1)

Parameters	Description	Test Conditions	Min.	Typ.	Max.	Units
VOH	Output HIGH Voltage	IOH = −200μA	3.7			Volts
VOL	Output LOW Voltage	IOL = 3.2mA			0.4	Volts
VIH	Input HIGH Voltage		2.0		VCC	Volts
VIL	Input LOW Voltage		−0.5		0.8	Volts
IIX	Input Load Current	VSS ≤ VI ≤ VCC			±10	μA
IOZ	Data Bus Leakage	VO = 0.4V			10	μA
		VO = VCC			10	
ICC	VCC Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
IDD	VDD Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
CO	Output Capacitance	fc = 1.0MHz, Inputs = 0V		8	10	pF
CI	Input Capacitance			5	8	pF
CIO	I/O Capacitance			10	12	pF

SWITCHING CHARACTERISTICS

Parameters	Description		8231A-8 Min	8231A-8 Max	8231A-3 Min	8231A-3 Max	8231A Min	8231A Max	Units
TAPW	$\overline{EACK}$ LOW Pulse Width		100		75		50		ns
TCDR	C/$\overline{D}$ to $\overline{RD}$ LOW Set-up Time		0		0		0		ns
TCDW	C/$\overline{D}$ to $\overline{WR}$ LOW Set-up Time		0		0		0		ns
TCPH	Clock Pulse HIGH Width		200		140		100		ns
TCPL	Clock Pulse LOW Width		240		160		120		ns
TCSR	$\overline{CS}$ LOW to $\overline{RD}$ LOW Set-up Time		0		0		0		ns
TCSW	$\overline{CS}$ LOW to $\overline{WR}$ LOW Set-up Time		0		0		0		ns
TCY	Clock Period		480	5000	320	3300	250	2500	ns
TDW	Data Bus Stable to $\overline{WR}$ HIGH Set-up Time		150		100 (Note 9)		100		ns
TEAE	$\overline{EACK}$ LOW to $\overline{END}$ HIGH Delay			200		175		150	ns
TEPW	$\overline{END}$ LOW Pulse Width (Note 4)		400		300		200		ns
TOP	Data Bus Output Valid to $\overline{PAUSE}$ HIGH Delay		0		0		0		ns
TPPWR	$\overline{PAUSE}$ LOW Pulse Width Read (Note 5)	Data	3.5TCY+50	5.5TCY+300	3.5TCY+50	5.5TCY+200	3.5TCY+50	5.5TCY+200	ns
		Status	1.5TCY+50	3.5TCY+300	1.5TCY+50	3.5TCY+200	1.5TCY+50	3.5TCY+200	
TPPWW	$\overline{PAUSE}$ LOW Pulse Width Write (Note 8)			50		50		50	ns
TPR	$\overline{PAUSE}$ HIGH to $\overline{RD}$ HIGH Hold Time		0		0		0		ns
TPW	$\overline{PAUSE}$ HIGH to $\overline{WR}$ HIGH Hold Time		0		0		0		ns
TRCD	$\overline{RD}$ HIGH to C/$\overline{D}$ Hold Time		0		0		0		ns
TRCS	$\overline{RD}$ HIGH to $\overline{CS}$ HIGH Hold Time		0		0		0		ns
TRO	$\overline{RD}$ LOW to Data Bus ON Delay		50		50		25		ns
TRP	$\overline{RD}$ LOW to $\overline{PAUSE}$ LOW Delay (Note 6)			150		100 (Note 9)		100	ns
TRZ	$\overline{RD}$ HIGH to Data Bus OFF Delay		50	200	50	150	25	100	ns
TSAPW	$\overline{SVACK}$ LOW Pulse Width		100		75		50		ns
TSAR	$\overline{SVACK}$ LOW to SVREQ LOW Delay			300		200		150	ns
TWCD	$\overline{WR}$ HIGH to C/$\overline{D}$ Hold Time		60		30		30		ns
TWCS	$\overline{WR}$ HIGH to $\overline{CS}$ HIGH Hold Time		60		30		30		ns
TWD	$\overline{WR}$ HIGH to Data Bus Hold Time		20		20		20		ns
TWI	Write Inactive Time	Command	3TCY		3TCY		3TCY		ns
		Data	4TCY		4TCY		4TCY		
TWP	$\overline{WR}$ LOW to $\overline{PAUSE}$ LOW Delay (Note 6)			150		100 (Note 9)		100	ns

Notes:
1. Typical values are for $T_A = 25°C$, nominal supply voltages and nominal processing parameters.
2. Switching parameters are listed in alphabetical order.
3. Test conditions assume transition times of 20ns or less, output loading of one TTL gate plus 100pF and timing reference levels of 0.8V and 2.0V.
4. $\overline{END}$ low pulse width is specified for $\overline{EACK}$ tied to VSS. Otherwise TEAE applies.
5. Minimum values shown assume no previously entered command is being executed for the data access. If a previously entered command is being executed, $\overline{PAUSE}$ LOW Pulse Width is the time to complete execution plus the time shown. Status may be read at any time without exceeding the time shown.
6. $\overline{PAUSE}$ is pulled low for both command and data operations.
7. TEX is the execution time of the current command (see the Command Execution Times table).
8. $\overline{PAUSE}$ low pulse width is less than 50ns when writing into the data port or the control port as long as the duty requirement (TWI) is observed and no previous command is being executed. TWI may be safely violated up to 500ns as long as the extended TPPWW that results is observed. If a previously entered command is being executed, $\overline{PAUSE}$ LOW Pulse Width is the time to complete execution plus the time shown.
9. 150ns for the MD8231A-3B.

8232
Floating Point Processor

DISTINCTIVE CHARACTERISTICS

- Single (32-bit) and double (64-bit) precision capability
- Add, subtract, multiply and divide functions
- Compatible with proposed IEEE format
- Easy interfacing to microprocessors
- 8-bit data bus
- Standard 24-pin package
- 12V and 5V power supplies
- Stack oriented operand storage
- Direct memory access or programmed I/O Data Transfers
- End of execution signal
- Error interrupt
- All inputs and outputs TTL level compatible
- Advanced N-channel silicon gate MOS technology

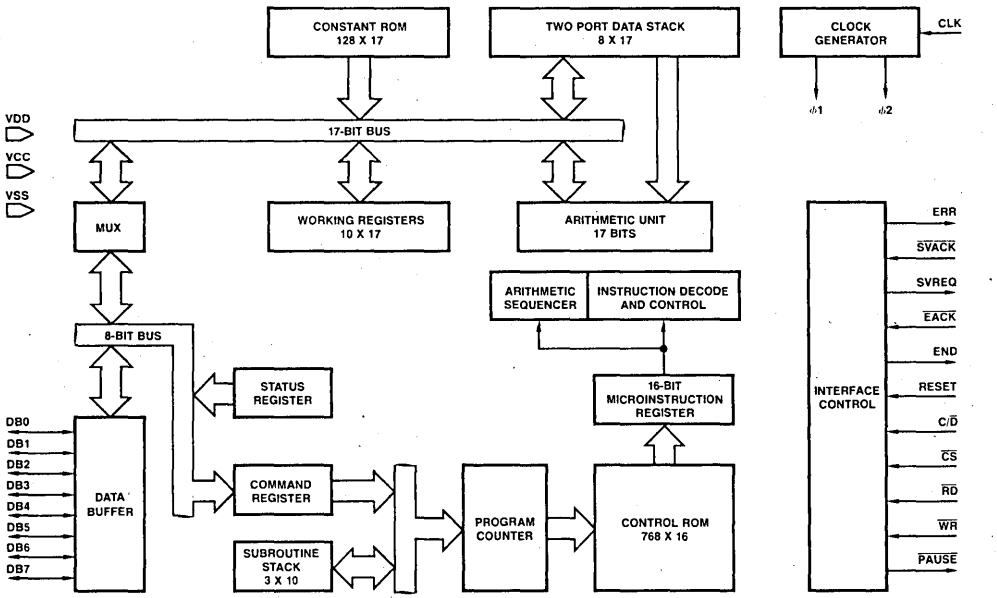
GENERAL DESCRIPTION

The 8232 is a high performance floating-point processor unit (FPU). It provides single precision (32-bit) and double precision (64-bit) add, subtract, multiply and divide operations. It can be easily interfaced to enhance the computational capabilities of the host microprocessor.

The operand, result, status and command information transfers take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack by the host processor and a command is issued to perform an operation on the data stack. The results of this operation are available to the host processor by popping the stack.

Information transfers between the 8232 and the host processor can be handled by using programmed I/O or direct memory access techniques. After completing an operation, the 8232 activates an "end of execution" signal that can be used to interrupt the host processor.

BLOCK DIAGRAM



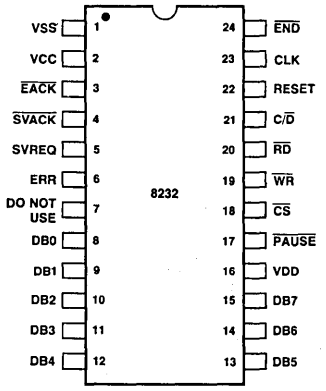
04100A-1

ORDERING INFORMATION

Package	Ambient Temperature	V_{SS}	V_{CC}	V_{DD}	Maximum Clock Frequency	
					2MHz	3MHz
Hermetic DIP	$0°C \leq T_A \leq 70°C$	0V	+5.0V ± 5%	+12V ± 5%	D8232-8	D8232-3
	$-40°C \leq T_A \leq +85°C$	0V	+5.0V ± 10%	+12V ± 10%	ID8232-8	ID8232-3
	$-55°C \leq T_A \leq +125°C$	0V	+5.0V ± 10%	+12V ± 10%	MD8232-8B	MD8232-3B

04100A-MMP

CONNECTION DIAGRAM – Top View
D-24



Note: Pin 1 is marked for orientation. 04100A-2

INTERFACE SIGNAL DESCRIPTION

VCC: +5V Power Supply

VDD: +12V Power Supply

VSS: Ground

CLK (Clock, Input)

An external timing source connected to the CLK input provides the necessary clocking.

RESET (Reset, Input)

A HIGH on this input causes initialization. Reset terminates any operation in progress, and clears the status register to zero. The internal stack pointer is initialized and the contents of the stack may be affected. After a reset the END output, the ERR output and the SVREQ output will be LOW. For proper initialization, RESET must be HIGH for at least five CLK periods following stable power supply voltages and stable clock.

C/$\overline{D}$ (Command/Data Select, Input)

The C/$\overline{D}$ input together with the $\overline{RD}$ and $\overline{WR}$ inputs determines the type of transfer to be performed on the data bus as follows:

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	Function
L	H	L	Push data byte into the stack
L	L	H	Pop data byte from the stack
H	H	L	Enter command
H	L	H	Read Status
X	L	L	Undefined

L = LOW
H = HIGH
X = DON'T CARE

END (End of Execution, Output)

A HIGH on this output indicates that execution of the current command is complete. This output will be cleared LOW by activating the $\overline{EACK}$ input LOW or performing any read or write operation or device initialization using the RESET. If $\overline{EACK}$ is tied LOW, the END output will be a pulse (see $\overline{EACK}$ description).

Reading the status register while a command execution is in progress is allowed. However any read or write operation clears the flip-flop that generates the END output. Thus such continuous reading could conflict with internal logic setting of the END flip-flop at the end of command execution.

$\overline{EACK}$ (End Acknowledge, Input)

This input when LOW makes the END output go LOW. As mentioned earlier HIGH on the END output signals completion of a command execution. The END signal is derived from an internal flip-flop which is clocked at the completion of a command. This flip-flop is clocked to the reset state when $\overline{EACK}$ is LOW. Consequently, if $\overline{EACK}$ is tied LOW, the END output will be a pulse that is approximately one CLK period wide.

SVREQ (Service Request, Output)

A HIGH on this output indicates completion of a command. In this sense this output is the same as the END output. However the Service Bit in the Command Register determines whether the SVREQ output will go HIGH at the completion of a command. This bit must be 1 for SVREQ to go HIGH. The SVREQ can be cleared (i.e., go LOW) by activating the $\overline{SVACK}$ input LOW or initializing the device using the RESET. Also, the SVREQ will be automatically cleared after completion of any command that has the service request bit as 0.

$\overline{SVACK}$ (Service Acknowledge, Input)

A LOW on this input clears SVREQ. If the $\overline{SVACK}$ input is permanently tied LOW, it will conflict with the internal setting of the SVREQ output. Thus the SVREQ indication cannot be relied upon if the $\overline{SVACK}$ is tied LOW.

DB0-DB7 (Data Bus, Input/Output)

These eight bidirectional lines are used to transfer command, status and operand information between the device and the host processor. DB0 is the least significant and DB7 is the most significant bit position. HIGH on a data bus line corresponds to 1 and LOW corresponds to 0.

When pushing operands on the stack using the data bus, the least significant byte must be pushed first and most significant byte last. When popping the stack to read the result of an operation, the most significant byte will be available on the data bus first and the least significant byte will be the last. Moreover, for pushing operands and popping results, the number of transactions must be equal to the proper number of bytes appropriate for the chosen format. Otherwise, the internal byte pointer will not be aligned properly. The 8232 single precision format requires 4 bytes and double precision format requires 8 bytes.

ERR (Error, Output)

This output goes HIGH to indicate that the current command execution resulted in an error condition. The error conditions are: attempt to divide by zero, exponent overflow and exponent underflow. The ERR output is cleared LOW on read status register operation or upon RESET.

The ERR output is derived from the error bits in the status register. These error bits will be updated internally at an appropriate time during a command execution. Thus ERR output going HIGH may not correspond with the completion of a command. Reading of the status register can be performed while a command execution is in progress. However it should be noted that reading the status register clears the ERR output. Thus reading the status register while a command execution in progress may result in an internal conflict with the ERR output.

$\overline{CS}$ (Chip Select, Input)

This input must be LOW to accomplish any read or write operation to the 8232.

To perform a write operation, appropriate data is presented on DB0 through DB7 lines, appropriate logic level on the C/$\overline{D}$ input and the $\overline{CS}$ input is made LOW. Whenever $\overline{WR}$ and $\overline{RD}$ inputs are both HIGH and $\overline{CS}$ is LOW, $\overline{PAUSE}$ goes LOW. However actual writing into the 8232 cannot start until $\overline{WR}$ is made LOW. After initiating the write operation by the HIGH to LOW transition on the $\overline{WR}$ input, the $\overline{PAUSE}$ output will go HIGH indicating the write operation has been acknowledged. The $\overline{WR}$ input can go HIGH after $\overline{PAUSE}$ goes HIGH. The data lines, C/$\overline{D}$ input and the $\overline{CS}$ input can change when appropriate hold time requirements are satisfied. See write timing diagram for details.

To perform a read operation an appropriate logic level is established on the C/$\overline{D}$ input and $\overline{CS}$ is made LOW. The $\overline{PAUSE}$ output goes LOW because $\overline{WR}$ and $\overline{RD}$ inputs are HIGH. The read operation does not start until the $\overline{RD}$ input goes LOW. $\overline{PAUSE}$ will go HIGH indicating that read operation is complete and the required information is available on the DB0 through DB7 lines. This information will remain on the data lines as long as $\overline{RD}$ is LOW. The $\overline{RD}$ input can return HIGH anytime after $\overline{PAUSE}$ goes HIGH. The $\overline{CS}$ input and C/$\overline{D}$ input can change anytime after $\overline{RD}$ returns HIGH. See read timing diagram for details. If the $\overline{CS}$ is tied LOW permanently, $\overline{PAUSE}$ will remain LOW until the next 8232 read or write access.

$\overline{RD}$ (Read, Input)

A LOW on this input is used to read information from an internal location and gate that information onto the data bus. The $\overline{CS}$ input must be LOW to accomplish the read operation. The C/$\overline{D}$ input determines what internal location is of interest. See C/$\overline{D}$, $\overline{CS}$ input descriptions and read timing diagram for details. If the END output was HIGH, performing any read operation will make the END output go LOW after the HIGH to LOW transition of the $\overline{RD}$ input (assuming $\overline{CS}$ is LOW). If the ERR output was HIGH performing a status register read operation will make the ERR output LOW. This will happen after the HIGH to LOW transition of the RD input (assuming $\overline{CS}$ is LOW).

$\overline{WR}$ (Write, Input)

A LOW on this input is used to transfer information from the data bus into an internal location. The $\overline{CS}$ must be LOW to accomplish the write operation. The C/$\overline{D}$ determines which internal location is to be written. See C/$\overline{D}$, $\overline{CS}$ input descriptions and write timing diagram for details.

If the END output was HIGH, performing any write operation will make the END output go LOW after the LOW to HIGH transition of the $\overline{WR}$ input (assuming $\overline{CS}$ is LOW).

$\overline{PAUSE}$ (Pause, Output)

This output is a handshake signal used while performing read or write transactions with the 8232. If the $\overline{WR}$ and $\overline{RD}$ inputs are both HIGH, the $\overline{PAUSE}$ output goes LOW with the $\overline{CS}$ input in anticipation of a transaction. If $\overline{WR}$ goes LOW to initiate a write transaction with proper signals established on the DB0-DB7, C/$\overline{D}$ inputs, the $\overline{PAUSE}$ will return HIGH indicating that the write operation has been accomplished. The $\overline{WR}$ can be made HIGH after this event. On the other hand, if a read operation is desired, the $\overline{RD}$ input is made LOW after activating $\overline{CS}$ LOW and establishing proper C/$\overline{D}$ input. (The $\overline{PAUSE}$ will go LOW in response to $\overline{CS}$ going LOW.) The $\overline{PAUSE}$ will return HIGH indicating completion of read. The $\overline{RD}$ can return HIGH after this event. It should be noted that a read or write operation can be initiated without any regard to whether a command execution is in progress or not. Proper device operation is assured by obeying the $\overline{PAUSE}$ output indication as described.

FUNCTIONAL DESCRIPTION

Major functional units of the 8232 are shown in the block diagram. The 8232 employs a microprogram controlled stack oriented architecture with 17-bit wide data paths.

The Arithmetic Unit receives one of its operands from the Operand Stack. This stack is an eight word by 17-bit two port memory with last in – first out (LIFO) attributes. The second operand to the Arithmetic Unit is supplied by the internal 17-bit bus. In addition to supplying the second operand, this bidirectional bus also carries the results from the output of the Arithmetic Unit when required. Writing into the Operand Stack takes place from this internal 17-bit bus when required. Also connected to this bus are the Constant ROM and Working Registers. The ROM provides the required constants to perform the mathematical operations while the Working Registers provide storage for the intermediate values during command execution.

Communication between the external world and the 8232 takes place on eight bidirectional input/output lines, DB0 through DB7 (Data Bus). These signals are gated to the internal 8-bit bus through appropriate interface and buffer circuitry. Multiplexing facilities exist for bidirectional communication between the internal eight and 17-bit buses. The Status Register and Command Register are also located on the 8-bit bus.

The 8232 operations are controlled by the microprogram contained in the Control ROM. The Program Counter supplies the microprogram addresses and can be partially loaded from the Command Register. Associated with the Program Counter is the Subroutine Stack where return addresses are held during subroutine calls in the microprogram. The Microinstruction Register holds the current microinstruction being executed. The register facilitates pipelined microprogram execution. The Instruction Decode logic generates various internal control signals needed for the 8232 operation.

The Interface Control logic receives several external inputs and provides handshake related outputs to facilitate interfacing the 8232 to microprocessors.

MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65 to +150°C
V_{DD} with Respect to V_{SS}	−0.5 to +15.0V
V_{CC} with Respect to V_{SS}	−0.5 to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5 to +7.0V
Power Dissipation (Package Limitation)	2.0W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

ELECTRICAL CHARACTERISTICS Over Operating Range (Note 1)

Parameters	Description	Test Conditions	Min.	Typ.	Max.	Units
VOH	Output HIGH Voltage	IOH = −200μA	3.7			Volts
VOL	Output LOW Voltage	IOL = 3.2mA			0.4	Volts
VIH	Input HIGH Voltage		2.0		VCC	Volts
VIL	Input LOW Voltage		−0.5		0.8	Volts
IIX	Input Load Current	VSS ≤ VI ≤ VCC			±10	μA
IOZ	Data Bus Leakage	VO = 0.4V			10	μA
		VO = VCC			10	
ICC	VCC Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
IDD	VDD Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
CO	Output Capacitance	fc = 1.0MHz, Inputs = 0V		8	10	pF
CI	Input Capacitance			5	8	pF
CIO	I/O Capacitance			10	12	pF

SWITCHING CHARACTERISTICS

Parameters	Description		8232-8 Min	8232-8 Max	8232-3 Min	8232-3 Max	Units
TAPW	$\overline{EACK}$ LOW Pulse Width		100		75		ns
TCDR	C/$\overline{D}$ to $\overline{RD}$ LOW Set-up Time		0		0		ns
TCDW	C/$\overline{D}$ to $\overline{WR}$ LOW Set-up Time		0		0		ns
TCPH	Clock Pulse HIGH Width		200	500	140	500	ns
TCPL	Clock Pulse LOW Width		240		160		ns
TCSP	$\overline{CS}$ LOW to $\overline{PAUSE}$ LOW Delay (Note 5)		150		100		ns
TCSR	$\overline{CS}$ to $\overline{RD}$ LOW Set-up Time		0		0		ns
TCSW	$\overline{CS}$ LOW to $\overline{WR}$ LOW Set-up Time		0		0		ns
TCY	Clock Period		480	5000	320	2000	ns
TDW	Data Valid to $\overline{WR}$ HIGH Delay		150		100		ns
TEAE	$\overline{EACK}$ LOW to END LOW Delay			200		175	ns
TEHPHR	END HIGH to $\overline{PAUSE}$ HIGH Data Read when Busy			5.5TCY+300		5.5TCY+200	ns
TEHPHW	END HIGH to $\overline{PAUSE}$ HIGH Write when Busy			200		175	ns
TEPW	END HIGH Pulse Width		400		300		ns
TEX	Execution Time		See Table 2				ns
TOP	Data Bus Output Valid to $\overline{PAUSE}$ HIGH Delay		0		0		ns
TPPWR	$\overline{PAUSE}$ LOW Pulse Width Read	Data	3.5TCY+50	5.5TCY+300	3.5TCY+50	5.5TCY+200	ns
		Status	1.5TCY+50	3.5TCY+300	1.5TCY+50	3.5TCY+200	
TPPWRB	END HIGH to $\overline{PAUSE}$ HIGH Read when Busy	Data	See Table 2				ns
		Status	1.5TCY+50	3.5TCY+300	1.5TCY+50	3.5TCY+200	
TPPWW	$\overline{PAUSE}$ LOW Pulse Width Write when Not Busy			TCSW+50		TCSW+50	ns
TPPWWB	$\overline{PAUSE}$ LOW Pulse Width Write when Busy		See Table 2				ns
TPR	$\overline{PAUSE}$ HIGH to Read HIGH Hold Time		0		0		ns
TPW	$\overline{PAUSE}$ HIGH to Write HIGH Hold Time		0		0		ns
TRCD	$\overline{RD}$ HIGH to C/$\overline{D}$ Hold Time		0		0		ns
TRCS	$\overline{RD}$ HIGH to $\overline{CS}$ HIGH Hold Time		0		0		ns
TRO	$\overline{RD}$ LOW to Data Bus On Delay		50		50		ns
TRZ	$\overline{RD}$ HIGH to Data Bus Off Delay		50	200	50	150	ns
TSAPW	SVACK LOW Pulse Width		100		75		ns
TSAR	SVACK LOW to SVREQ LOW Delay			300		200	ns
TWCD	$\overline{WR}$ HIGH to C/D Hold Time		60		30		ns
TWCS	$\overline{WR}$ HIGH to CS HIGH Hold Time		60		30		ns
TWD	$\overline{WR}$ HIGH to Data Bus Hold Time		20		20		ns

NOTES:
1. Typical values are for T_A = 25°C, nominal supply voltages and nominal processing parameters.
2. Switching parameters are listed in alphabetical order.
3. Test conditions assume transition times of 20ns or less, output loading of one TTL gate plus 100pF and timing reference levels of 0.8V and 2.0V.
4. END HIGH pulse width is specified for $\overline{EACK}$ tied to VSS. Otherwise TEAE applies.
5. $\overline{PAUSE}$ is pulled low for both command and data operations.
6. TEX is the execution time of the current command (see the Command Execution Times table).
7. $\overline{PAUSE}$ will go low at this point if $\overline{CS}$ is low and $\overline{RD}$ and $\overline{WR}$ are high.

EF9340/41

Videotex CRT Display Processor

DISTINCTIVE CHARACTERISTICS

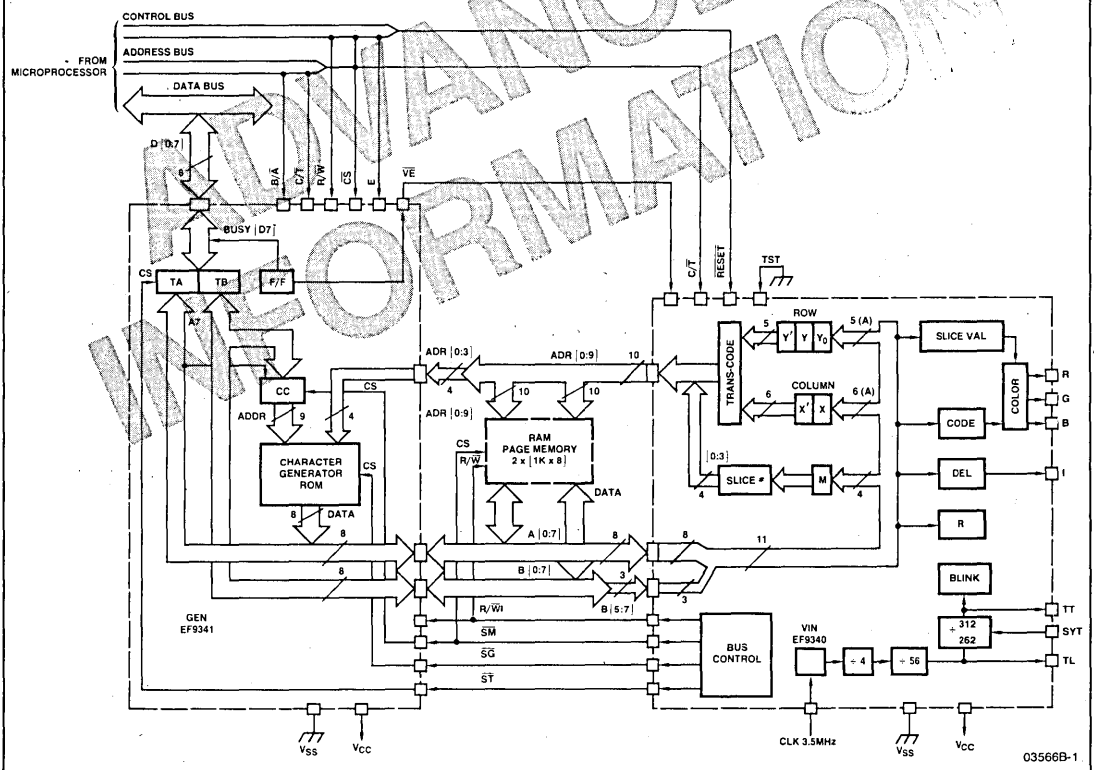
- 21 or 25 rows of 40 characters
- On-chip character generator
- 128 alphanumeric and 128 graphic characters
- Expandable character set
- On-chip R, G, B, registers
- Double height, width, reverse, blink, underline, box, conceal attributes
- Two-color graphic symbols
- Programmable roll up, roll down, zoom and cursor display.

GENERAL DESCRIPTION

The EF9340 and EF9341 together with a 1k x 16 RAM comprise a complete video display unit. The display unit is organized around a 16-bit internal bus controlled by the 9340, which contains a timing generator and display access interface logic. The 9341 consists of a character generator and two 8-bit registers which provide a buffered interface with any general purpose 8-bit bus.

A full alphanumeric font of 128 characters is provided, as are 128 separate semi-graphics characters. Also, provision is made for easy character set expansion by the simple addition of one (96 characters) or two (192 characters) standard 1k x 8 memory components.

BLOCK DIAGRAM



ADVANCED INFORMATION

This document contains information on a product under development at Advanced Micro Devices, Inc. The information is intended to help you to evaluate this product. AMD reserves the right to change or discontinue work on this proposed product without notice. 03566B-MMP

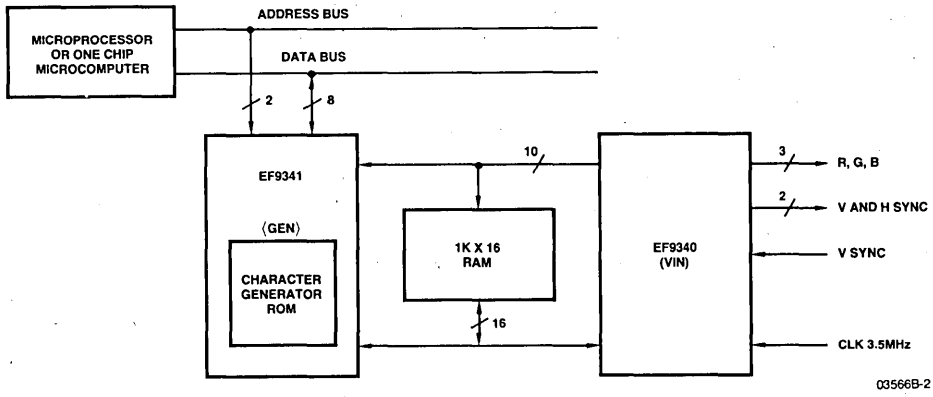
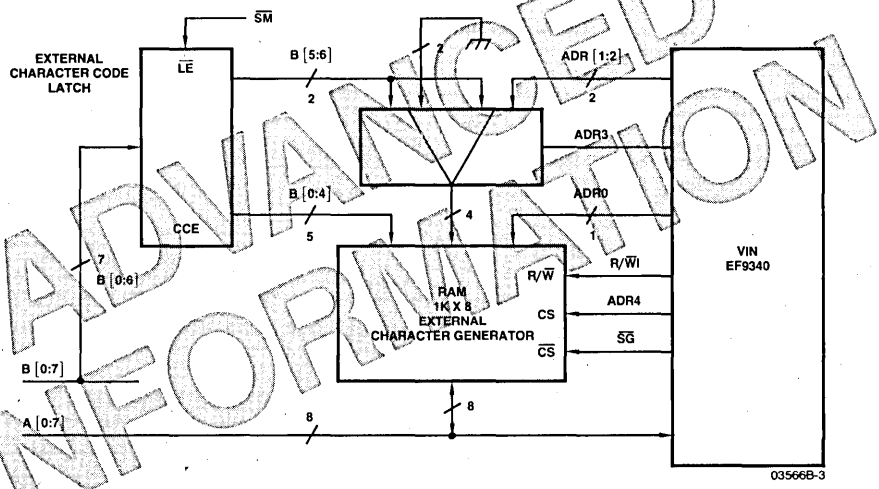


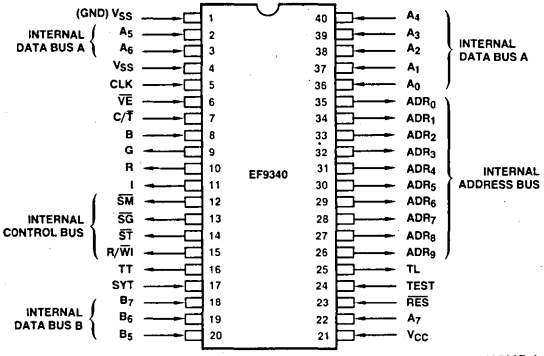
Figure 1. Adding an External Character Generator



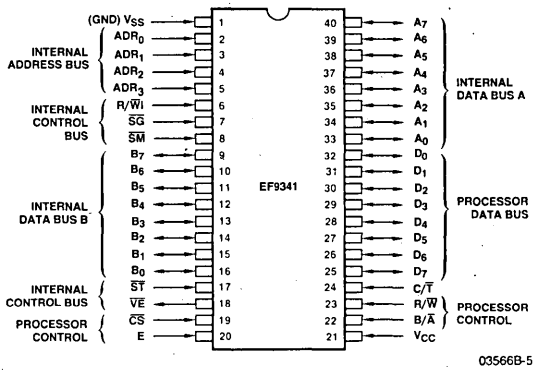
ADVANCED INFORMATION

CONNECTION DIAGRAMS

D-40, P-40



D-40, P-40



Am9511A

Arithmetic Processor

DISTINCTIVE CHARACTERISTICS

- 2, 3 and 4MHz operation
- Fixed point 16 and 32 bit operations
- Floating point 32 bit operations
- Binary data formats
- Add, Subtract, Multiply and Divide
- Trigonometric and inverse trigonometric functions
- Square roots, logarithms, exponentiation
- Float to fixed and fixed to float conversions
- Stack-oriented operand storage
- DMA or programmed I/O data transfers
- End signal simplifies concurrent processing
- Synchronous/Asynchronous operations
- General purpose 8-bit data bus interface
- Standard 24 pin package
- +12 volt and +5 volt power supplies
- Advanced N-channel silicon gate MOS technology

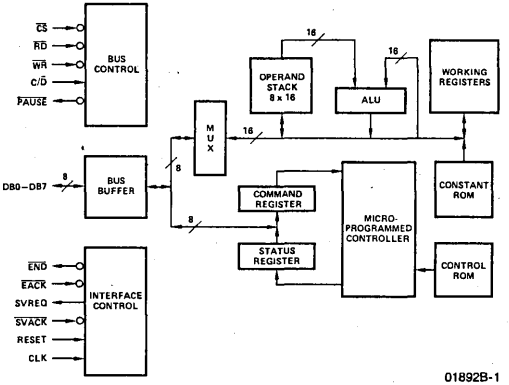
GENERAL DESCRIPTION

The Am9511A Arithmetic Processing Unit (APU) is a monolithic MOS/LSI device that provides high performance fixed and floating point arithmetic and a variety of floating point trigonometric and mathematical operations. It may be used to enhance the computational capability of a wide variety of processor-oriented systems.

All transfers, including operand, result, status and command information, take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack and a command is issued to perform operations on the data in the stack. Results are then available to be retrieved from the stack, or additional commands may be entered.

Transfers to and from the APU may be handled by the associated processor using conventional programmed I/O, or may be handled by a direct memory access controller for improved performance. Upon completion of each command, the APU issues an end of execution signal that may be used as an interrupt by the CPU to help coordinate program execution.

BLOCK DIAGRAM

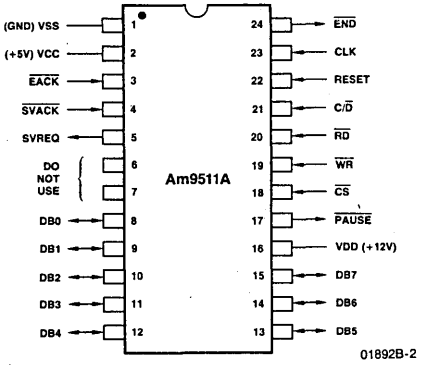


01892B-1

CONNECTION DIAGRAM

Top View

D-24-2



01892B-2

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Package Type	Ambient Temperature	Maximum Clock Frequency		
		2MHz	3MHz	4MHz
Hermetic DIP	$0°C \leq T_A \leq +70°C$	Am9511ADC	Am9511A-1DC	Am9511A-4DC
	$-40°C \leq T_A \leq +85°C$	Am9511ADI	Am9511A-1DI	
	$-55°C \leq T_A \leq +125°C$	Am9511ADMB	Am9511A-1DMB	

01892B-MMP

INTERFACE SIGNAL DESCRIPTION

VCC: +5V Power Supply
VDD: +12V Power Supply
VSS: Ground

CLK (Clock, Input)

An external timing source connected to the CLK input provides the necessary clocking. The CLK input can be asynchronous to the $\overline{RD}$ and $\overline{WR}$ control signals.

RESET (Reset, Input)

A HIGH on this input causes initialization. Reset terminates any operation in progress, and clears the status register to zero. The internal stack pointer is initialized and the contents of the stack may be affected but the command register is not affected by the reset operation. After a reset the $\overline{END}$ output will be HIGH, and the SVREQ output will be LOW. For proper initialization, the RESET input must be HIGH for at least five CLK periods following stable power supply voltages and stable clock.

C/$\overline{D}$ (Command/Data Select, Input)

The C/$\overline{D}$ input together with the $\overline{RD}$ and $\overline{WR}$ inputs determines the type of transfer to be performed on the data bus as follows:

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	Function
L	H	L	Push data byte into the stack
L	L	H	Pop data byte from the stack
H	H	L	Enter command byte from the data bus
H	L	H	Read Status
X	L	L	Undefined

L = LOW
H = HIGH
X = DON'T CARE

$\overline{END}$ (End of Execution, Output)

A LOW on this output indicates that execution of the current command is complete. This output will be cleared HIGH by activating the $\overline{EACK}$ input LOW or performing any read or write operation or device initialization using the RESET. If $\overline{EACK}$ is tied LOW, the $\overline{END}$ output will be a pulse (see $\overline{EACK}$ description). This is an open drain output and requires a pull up to +5V.

Reading the status register while a command execution is in progress is allowed. However any read or write operation clears the flip-flop that generates the $\overline{END}$ output. Thus such continuous reading could conflict with internal logic setting the $\overline{END}$ flip-flop at the completion of command execution.

$\overline{EACK}$ (End Acknowledge, Input)

This input when LOW makes the $\overline{END}$ output go HIGH. As mentioned earlier LOW on the $\overline{END}$ output signals completion of a command execution. The $\overline{END}$ output signal is derived from an internal flip-flop which is clocked at the completion of a command. This flip-flop is clocked to the reset state when $\overline{EACK}$ is LOW. Consequently, if the $\overline{EACK}$ is tied LOW, the $\overline{END}$ output will be a pulse that is approximately one CLK period wide.

SVREQ (Service Request, Output)

A HIGH on this output indicates completion of a command. In this sense this output is same as the $\overline{END}$ output. However, whether the SVREQ output will go HIGH at the completion of a command or not is determined by a service request bit in the command register. This bit must be 1 for SVREQ to go HIGH. The SVREQ can be cleared (i.e., go LOW) by activating the $\overline{SVACK}$ input LOW or initializing the device using the RESET. Also, the SVREQ will be automatically cleared after completion of any command that has the service request bit as 0.

$\overline{SVACK}$ (Service Acknowledge, Input)

A LOW on this input activates the reset input of the flip-flop generating the SVREQ output. If the $\overline{SVACK}$ input is permanently tied LOW, it will conflict with the internal setting of the flip-flop to generate the SVREQ output. Thus the SVREQ indication cannot be relied upon if the $\overline{SVACK}$ is tied LOW.

DB0-DB7 (Bidirectional Data Bus, Input/Output)

These eight bidirectional lines are used to transfer command, status and operand information between the device and the host processor. DB0 is the least significant and DB7 is the most significant bit position. HIGH on the data bus line corresponds to 1 and LOW corresponds to 0.

When pushing operands on the stack using the data bus, the least significant byte must be pushed first and most significant byte last. When popping the stack to read the result of an operation, the most significant byte will be available on the data bus first and the least significant byte will be the last. Moreover, for pushing operands and popping results, the number of transactions must be equal to the proper number of bytes appropriate for the chosen format. Otherwise, the internal byte pointer will not be aligned properly. The Am9511A single precision format requires 2 bytes, double precision and floating-point formats require 4 bytes.

$\overline{CS}$ (Chip Select, Input)

This input must be LOW to accomplish any read or write operation to the Am9511A.

To perform a write operation data is presented on DB0 through DB7 lines, C/$\overline{D}$ is driven to an appropriate level and the $\overline{CS}$ input is made LOW. However, actual writing into the Am9511A cannot start until $\overline{WR}$ is made LOW. After initiating the write operation by a $\overline{WR}$ HIGH to LOW transition, the $\overline{PAUSE}$ output will go LOW momentarily (TPPWW).

The $\overline{WR}$ input can go HIGH after $\overline{PAUSE}$ goes HIGH. The data lines, C/$\overline{D}$ input and the $\overline{CS}$ input can change when appropriate hold time requirements are satisfied. See write timing diagram for details.

To perform a read operation an appropriate logic level is established on the C/$\overline{D}$ input and $\overline{CS}$ is made LOW. The Read operation does not start until the $\overline{RD}$ input goes LOW. $\overline{PAUSE}$ will go LOW for a period of TPPWR. When $\overline{PAUSE}$ goes back HIGH again, it indicates that read operation is complete and the required information is available on the DB0 through DB7 lines. This information will remain on the data lines as long as $\overline{RD}$ input is LOW. The $\overline{RD}$ input can return HIGH anytime after $\overline{PAUSE}$ goes HIGH. The $\overline{CS}$ input and C/$\overline{D}$ inputs can change anytime after $\overline{RD}$ returns HIGH. See read timing diagram for details.

$\overline{RD}$ (Read, Input)

A LOW on this input is used to read information from an internal location and gate that information on to the data bus. The $\overline{CS}$ input must be LOW to accomplish the read operation. The C/$\overline{D}$ input determines what internal location is of interest. See C/$\overline{D}$, $\overline{CS}$ input descriptions and read timing diagram for details. If the END output was LOW, performing any read operation will make the $\overline{END}$ output go HIGH after the HIGH to LOW transition of the $\overline{RD}$ input (assuming $\overline{CS}$ is LOW).

$\overline{WR}$ (Write, Input)

A LOW on this input is used to transfer information from the data bus into an internal location. The $\overline{CS}$ must be LOW to accomplish the write operation. The C/$\overline{D}$ determines which internal location is to be written. See C/$\overline{D}$, $\overline{CS}$ input descriptions and write timing diagram for details.

If the $\overline{END}$ output was LOW, performing any write operation will make the $\overline{END}$ output go HIGH after the LOW to HIGH transition of the $\overline{WR}$ input (assuming $\overline{CS}$ is LOW).

$\overline{PAUSE}$ (Pause, Output)

This output is a handshake signal used while performing read or write transactions with the Am9511A. A LOW at this output indicates that the Am9511A has not yet completed its information transfer with the host over the data bus. During a read operation, after $\overline{CS}$ went LOW, the $\overline{PAUSE}$ will become LOW shortly (TRP) after $\overline{RD}$ goes LOW. $\overline{PAUSE}$ will return high only after the data bus contains valid output data. The $\overline{CS}$ and $\overline{RD}$ should remain LOW when $\overline{PAUSE}$ is LOW. The RD may go high anytime after $\overline{PAUSE}$ goes HIGH. During a write operation, after $\overline{CS}$ went LOW, the $\overline{PAUSE}$ will be LOW for a very short duration (TPPWN) after $\overline{WR}$ goes LOW. Since the minimum of TPPWW is 0, the $\overline{PAUSE}$ may not go LOW at all for fast devices. $\overline{WR}$ may go HIGH anytime after $\overline{PAUSE}$ goes HIGH.

FUNCTIONAL DESCRIPTION

Major functional units of the Am9511A are shown in the block diagram. The Am9511A employs a microprogram controlled stack oriented architecture with 16-bit wide data paths.

The Arithmetic Logic Unit (ALU) receives one of its operands from the Operand Stack. This stack is an 8-word by 16-bit 2-port memory with last in-first out (LIFO) attributes. The second operand to the ALU is supplied by the internal 16-bit bus. In addition to supplying the second operand, this bidirectional bus also carries the results from the output of the ALU when required. Writing into the Operand Stack takes place from this internal 16-bit bus when required. Also connected to this bus are the Constant ROM and Working Registers. The ROM provides the required constants to perform the mathematical operations (Chebyshev Algorithms) while the Working Registers provide storage for the intermediate values during command execution.

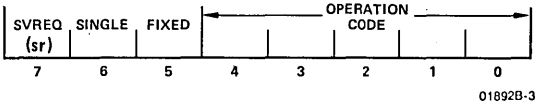
Communication between the external world and the Am9511A takes place on eight bidirectional input/output lines DB0 through DB7 (Data Bus). These signals are gated to the internal eight-bit bus through appropriate interface and buffer circuitry. Multiplexing facilities exist for bidirectional communication between the internal eight and sixteen-bit buses. The Status Register and Command Register are also accessible via the eight-bit bus.

The Am9511A operations are controlled by the microprogram contained in the Control ROM. The Program Counter supplies the microprogram addresses and can be partially loaded from the Command Register. Associated with the Program Counter is the Subroutine Stack where return addresses are held during subroutine calls in the microprogram. The Microinstruction Register holds the current microinstruction being executed. This register facilitates pipelined microprogram execution. The Instruction Decode logic generates various internal control signals needed for the Am9511A operation.

The Interface Control logic receives several external inputs and provides handshake related outputs to facilitate interfacing the Am9511A to microprocessors.

COMMAND FORMAT

Each command entered into the Am9511A consists of a single 8-bit byte having the format illustrated below:



01892B-3

Bits 0-4 select the operation to be performed as shown in the table. Bits 5-6 select the data format for the operation. If bit 5 is a 1, a fixed point data format is specified. If bit 5 is a 0, floating point format is specified. Bit 6 selects the precision of the data to be operated on by fixed point commands (if bit 5 = 0, bit 6 must be 0). If bit 6 is a 1, single-precision (16-bit) operands are indicated; if bit 6 is a 0, double-precision (32-bit) operands are indicated. Results are undefined for all illegal combinations of bits in the command byte. Bit 7 indicates whether a service request is to be issued after the command is executed. If bit 7 is a 1, the service request output (SVREQ) will go high at the conclusion of the command and will remain high until reset by a low level on the service acknowledge pin ($\overline{SVACK}$) or until completion of execution of a succeeding command where bit 7 is 0. Each command issued to the Am9511A requests post execution service based upon the state of bit 7 in the command byte. When bit 7 is a 0, SVREQ remains low.

COMMAND SUMMARY

Command Code								Command Mnemonic	Command Description
7	6	5	4	3	2	1	0		
FIXED-POINT 16-BIT									
sr	1	1	0	1	1	0	0	SADD	Add TOS to NOS. Result to NOS. Pop Stack.
sr	1	1	0	1	1	0	1	SSUB	Subtract TOS from NOS. Result to NOS. Pop Stack.
sr	1	1	0	1	1	1	0	SMUL	Multiply NOS by TOS. Lower half of result to NOS. Pop Stack.
sr	1	1	1	0	1	1	0	SMUU	Multiply NOS by TOS. Upper half of result to NOS. Pop Stack.
sr	1	1	0	1	1	1	1	SDIV	Divide NOS by TOS. Result to NOS. Pop Stack.
FIXED-POINT 32-BIT									
sr	0	1	0	1	1	0	0	DADD	Add TOS to NOS. Result to NOS. Pop Stack.
sr	0	1	0	1	1	0	1	DSUB	Subtract TOS from NOS. Result to NOS. Pop Stack.
sr	0	1	0	1	1	1	0	DMUL	Multiply NOS by TOS. Lower half of result to NOS. Pop Stack.
sr	0	1	1	0	1	1	0	DMUU	Multiply NOS by TOS. Upper half of result to NOS. Pop Stack.
sr	0	1	0	1	1	1	1	DDIV	Divide NOS by TOS. Result to NOS. Pop Stack.
FLOATING-POINT 32-BIT									
sr	0	0	1	0	0	0	0	FADD	Add TOS to NOS. Result to NOS. Pop Stack.
sr	0	0	1	0	0	0	1	FSUB	Subtract TOS from NOS. Result to NOS. Pop Stack.
sr	0	0	1	0	0	1	0	FMUL	Multiply NOS by TOS. Result to NOS. Pop Stack.
sr	0	0	1	0	0	1	1	FDIV	Divide NOS by TOS. Result to NOS. Pop Stack.
DERIVED FLOATING-POINT FUNCTIONS									
sr	0	0	0	0	0	0	1	SQRT	Square Root of TOS. Result in TOS.
sr	0	0	0	0	0	1	0	SIN	Sine of TOS. Result in TOS.
sr	0	0	0	0	0	1	1	COS	Cosine of TOS. Result in TOS.
sr	0	0	0	0	1	0	0	TAN	Tangent of TOS. Result in TOS.
sr	0	0	0	0	1	0	1	ASIN	Inverse Sine of TOS. Result in TOS.
sr	0	0	0	0	1	1	0	ACOS	Inverse Cosine of TOS. Result in TOS.
sr	0	0	0	0	1	1	1	ATAN	Inverse Tangent of TOS. Result in TOS.
sr	0	0	0	1	0	0	0	LOG	Common Logarithm (base 10) of TOS. Result in TOS.
sr	0	0	0	1	0	0	1	LN	Natural Logarithm (base e) of TOS. Result in TOS.
sr	0	0	0	1	0	1	0	EXP	Exponential (e^x) of TOS. Result in TOS.
sr	0	0	0	1	0	1	1	PWR	NOS raised to the power in TOS. Result in NOS. Pop Stack.
DATA MANIPULATION COMMANDS									
sr	0	0	0	0	0	0	0	NOP	No Operation
sr	0	0	1	1	1	1	1	FIXS	Convert TOS from floating point to 16-bit fixed point format.
sr	0	0	1	1	1	1	0	FIXD	Convert TOS from floating point to 32-bit fixed point format.
sr	0	0	1	1	1	0	1	FLTS	Convert TOS from 16-bit fixed point to floating point format.
sr	0	0	1	1	1	0	0	FLTD	Convert TOS from 32-bit fixed point to floating point format.
sr	1	1	1	0	1	0	0	CHSS	Change sign of 16-bit fixed point operand on TOS.
sr	0	1	1	0	1	0	0	CHSD	Change sign of 32-bit fixed point operand on TOS.
sr	0	0	1	0	1	0	1	CHSF	Change sign of floating point operand on TOS.
sr	1	1	1	0	1	1	1	PTOS	Push 16-bit fixed point operand on TOS to NOS (Copy)
sr	0	1	1	0	1	1	1	PTOD	Push 32-bit fixed point operand on TOS to NOS. (Copy)
sr	0	0	1	0	1	1	1	PTOF	Push floating point operand on TOS to NOS. (Copy)
sr	1	1	1	1	0	0	0	POPS	Pop 16-bit fixed point operand from TOS. NOS becomes TOS.
sr	0	1	1	1	0	0	0	POPD	Pop 32-bit fixed point operand from TOS. NOS becomes TOS.
sr	0	0	1	1	0	0	0	POPF	Pop floating point operand from TOS. NOS becomes TOS.
sr	1	1	1	1	0	0	1	XCHS	Exchange 16-bit fixed point operands TOS and NOS.
sr	0	1	1	1	0	0	1	XCHD	Exchange 32-bit fixed point operands TOS and NOS.
sr	0	0	1	1	0	0	1	XCHF	Exchange floating point operands TOS and NOS.
sr	0	0	1	1	0	1	0	PUPI	Push floating point constant "π" onto TOS. Previous TOS becomes NOS.

NOTES:

1. TOS means Top of Stack. NOS means Next on Stack.
2. AMD Application Brief "Algorithm Details for the Am9511A APU" provides detailed descriptions of each command function, including data ranges, accuracies, stack configurations, etc.
3. Many commands destroy one stack location (bottom of stack) during development of the result. The derived functions may destroy several stack locations. See Application Brief for details.
4. The trigonometric functions handle angles in radians, not degrees.
5. No remainder is available for the fixed-point divide functions.
6. Results will be undefined for any combination of command coding bits not specified in this table.

COMMAND INITIATION

After properly positioning the required operands on the stack, a command may be issued. The procedure for initiating a command execution is as follows:

1. Enter the appropriate command on the DB0-DB7 lines.
2. Establish HIGH on the C/$\overline{D}$ input.
3. Establish LOW on the $\overline{CS}$ input.
4. Establish LOW on the $\overline{WR}$ input after an appropriate set up time (see timing diagrams).
5. Sometime after the HIGH to LOW level transition of $\overline{WR}$ input, the $\overline{PAUSE}$ output will become LOW. After a delay of TPPWW, it will go HIGH to acknowledge the write operation. The WR input can return to HIGH anytime after $\overline{PAUSE}$ going HIGH. The DB0-DB7, C/$\overline{D}$ and $\overline{CS}$ inputs are allowed to change after the hold time requirements are satisfied (see timing diagram).

An attempt to issue a new command while the current command execution is in progress is allowed. Under these circumstances, the $\overline{PAUSE}$ output will not go HIGH until the current command execution is completed.

OPERAND ENTRY

The Am9511A commands operate on the operands located at the TOS and NOS and results are returned to the stack at NOS and then popped to TOS. The operands required for the Am9511A are one of three formats – single precision fixed-point (2 bytes), double precision fixed-point (4 bytes) or floating-point (4 bytes). The result of an operation has the same format as the operands except for float to fix or fix to float commands.

Operands are always entered into the stack least significant byte first and most significant byte last. The following procedure must be followed to enter operands onto the stack:

1. The lower significant operand byte is established on the DB0-DB7 lines.
2. A LOW is established on the C/$\overline{D}$ input to specify that data is to be entered into the stack.
3. The $\overline{CS}$ input is made LOW.
4. After appropriate set up time (see timing diagrams), the $\overline{WR}$ input is made LOW. The $\overline{PAUSE}$ output will become LOW.
5. Sometime after this event, the $\overline{PAUSE}$ will return HIGH to indicate that the write operation has been acknowledged.
6. Anytime after the $\overline{PAUSE}$ output goes HIGH the $\overline{WR}$ input can be made HIGH. The DB0-DB7, C/$\overline{D}$ and $\overline{CS}$ inputs can change after appropriate hold time requirements are satisfied (see timing diagrams).

The above procedure must be repeated until all bytes of the operand are pushed into the stack. It should be noted that for single precision fixed-point operands 2 bytes should be pushed and 4 bytes must be pushed for double precision fixed-point or floating-point. Not pushing all the bytes of a quantity will result in byte pointer misalignment.

The Am9511A stack can accommodate 8 single precision fixed-point quantities or 4 double precision fixed-point or floating-point quantities. Pushing more quantities than the capacity of the stack will result in loss of data which is usual with any LIFO stack.

DATA REMOVAL

Result from an operation will be available at the TOS. Results can be transferred from the stack to the data bus by reading the stack. When the stack is popped for results, the most significant byte is available first and the least significant byte last. A result is always of the same precision as the operands that produced it except for format conversion commands. Thus when the result is taken from the stack, the total number of bytes popped out should be appropriate with the precision – single precision results are 2 bytes and double precision and floating-point results are 4 bytes. The following procedure must be used for reading the result from the stack:

1. A LOW is established on the C/$\overline{D}$ input.
2. The $\overline{CS}$ input is made LOW.
3. After appropriate set up time (see timing diagrams), the $\overline{RD}$ input is made LOW. The $\overline{PAUSE}$ will become LOW.
4. Sometime after this, $\overline{PAUSE}$ will return HIGH indicating that the data is available on the DB0-DB7 lines. This data will remain on the DB0-DB7 lines as long as the $\overline{RD}$ input remains LOW.
5. Anytime after $\overline{PAUSE}$ goes HIGH, the $\overline{RD}$ input can return HIGH to complete transaction.
6. The $\overline{CS}$ and C/$\overline{D}$ inputs can change after appropriate hold time requirements are satisfied (see timing diagram).
7. Repeat this procedure until all bytes appropriate for the precision of the result are popped out.

Reading of the stack does not alter its data; it only adjusts the byte pointer. If more data is popped than the capacity of the stack, the internal byte pointer will wrap around and older data will be read again, consistent with the LIFO stack.

STATUS READ

The Am9511A status register can be read without any regard to whether a command is in progress or not. The only implication that has to be considered is the effect this might have on the END output discussed in the signal descriptions.

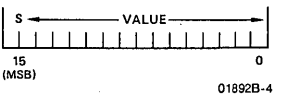
The following procedure must be followed to accomplish status register reading.

1. Establish HIGH on the C/$\overline{D}$ input.
2. Establish LOW on the $\overline{CS}$ input.
3. After appropriate set up time (see timing diagram) $\overline{RD}$ input is made LOW. The $\overline{PAUSE}$ will become LOW.
4. Sometime after the HIGH to LOW transition of $\overline{RD}$ input, the $\overline{PAUSE}$ will become HIGH indicating that status register contents are available on the DB0-DB7 lines. The status data will remain on DB0-DB7 as long as $\overline{RD}$ input is LOW.
5. The $\overline{RD}$ input can be returned HIGH anytime after $\overline{PAUSE}$ goes HIGH.
6. The C/$\overline{D}$ input and $\overline{CS}$ input can change after satisfying appropriate hold time requirements (see timing diagram).

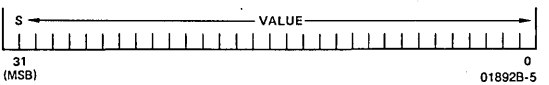
DATA FORMATS

The Am9511A Arithmetic Processing Unit handles operands in both fixed-point and floating-point formats. Fixed-point operands may be represented in either single (16-bit operands) or double precision (32-bit operands), and are always represented as binary, two's complement values.

16-BIT FIXED-POINT FORMAT



32-BIT FIXED-POINT FORMAT



The sign (positive or negative) of the operand is located in the most significant bit (MSB). Positive values are represented by a sign bit of zero (S = 0). Negative values are represented by the two's complement of the corresponding positive value with a sign bit equal to 1 (S = 1). The range of values that may be accomodated by each of these formats is −32,767 to +32,767 for single precision and −2,147,483,647 to +2,147,483,647 for double precision.

Floating point binary values are represented in a format that permits arithmetic to be performed in a fashion analogous to operations with decimal values expressed in scientific notation.

$$(5.83 \times 10^2)(8.16 \times 10^1) = (4.75728 \times 10^4)$$

In the decimal system, data may be expressed as values between 0 and 10 times 10 raised to a power that effectively shifts the implied decimal point right or left the number of places necessary to express the result in conventional form (e.g., 47,572.8). The value-portion of the data is called the mantissa. The exponent may be either negative or positive.

The concept of floating point notation has both a gain and a loss associated with it. The gain is the ability to represent the significant digits of data with values spanning a large dynamic range limited only by the capacity of the exponent field. For example, in decimal notation if the exponent field is two digits wide, and the mantissa is five digits, a range of values (positive or negative) from 1.0000×10^{-99} to $9.9999 \times 10^{+99}$ can be accommodated. The loss is that only the significant digits of the value can be represented. Thus there is no distinction in this representation between the values 123451 and 123452, for example, since each would be expressed as: 1.2345×10^5. The sixth digit has been discarded. In most applications where the dynamic range of values to be represented is large, the loss of significance, and hence accuracy of results, is a minor consideration. For greater precision a fixed point format could be chosen, although with a loss of potential dynamic range.

The Am9511 is a binary arithmetic processor and requires that floating point data be represented by a fractional mantissa value between .5 and 1 multiplied by 2 raised to an appropriate power. This is expressed as follows:

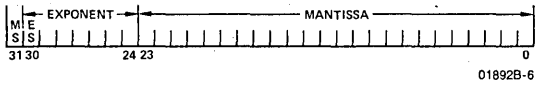
$$\text{value} = \text{mantissa} \times 2^{\text{exponent}}$$

For example, the value 100.5 expressed in this form is 0.11001001×2^7. The decimal equivalent of this value may be computed by summing the components (powers of two) of the mantissa and then multiplying by the exponent as shown below:

$$\begin{aligned}\text{value} &= (2^{-1} + 2^{-2} + 2^{-5} + 2^{-8}) \times 2^7 \\ &= (0.5 + 0.25 + 0.03125 + 0.00290625) \times 128 \\ &= 0.78515625 \times 128 \\ &= 100.5\end{aligned}$$

FLOATING POINT FORMAT

The format for floating-point values in the Am9511A is given below. The mantissa is expressed as a 24-bit (fractional) value; the exponent is expressed as an unbiased two's complement 7-bit value having a range of −64 to +63. The most significant bit is the sign of the mantissa (0 = positive, 1 = negative), for a total of 32 bits. The binary point is assumed to be to the left of the most significant mantissa bit (bit 23). All floating-point data values must be normalized. Bit 23 must be equal to 1, except for the value zero, which is represented by all zeros.



01892B-6

The range of values that can be represented in this format is $\pm(2.7 \times 10^{-20}$ to $9.2 \times 10^{18})$ and zero.

STATUS REGISTER

The Am9511A contains an eight bit status register with the following bit assignments:

BUSY	SIGN	ZERO	ERROR CODE				CARRY
7	6	5	4	3	2	1	0

01892B-5

BUSY: Indicates that Am9511A is currently executing a command (1 = Busy).

SIGN: Indicates that the value on the top of stack is negative (1 = Negative).

ZERO: Indicates that the value on the top of stack is zero (1 = Value is zero).

ERROR CODE: This field contains an indication of the validity of the result of the last operation. The error codes are:

- 0000 – No error
- 1000 – Divide by zero
- 0100 – Square root or log of negative number
- 1100 – Argument of inverse sine, cosine, or e^x too large
- XX10 – Underflow
- XX01 – Overflow

CARRY: Previous operation resulted in carry or borrow from most significant bit. (1 = Carry/Borrow, 0 = No Carry/No Borrow)

If the BUSY bit in the status register is a one, the other status bits are not defined; if zero, indicating not busy, the operation is complete and the other status bits are defined as given above.

Table 1.

Command Mnemonic	Hex Code (sr = 1)	Hex Code (sr = 0)	Execution Cycles	Summary Description
16-BIT FIXED-POINT OPERATIONS				
SADD	EC	6C	16-18	Add TOS to NOS. Result to NOS. Pop Stack.
SSUB	ED	6D	30-32	Subtract TOS from NOS. Result to NOS. Pop Stack.
SMUL	EE	6E	84-94	Multiply NOS by TOS. Lower result to NOS. Pop Stack.
SMUU	F6	76	80-98	Multiply NOS by TOS. Upper result to NOS. Pop Stack.
SDIV	EF	6F	84-94	Divide NOS by TOS. Result to NOS. Pop Stack.
32-BIT FIXED-POINT OPERATIONS				
DADD	AC	2C	20-22	Add TOS to NOS. Result to NOS. Pop Stack.
DSUB	AD	2D	38-40	Subtract TOS from NOS. Result to NOS. Pop Stack.
DMUL	AE	2E	194-210	Multiply NOS by TOS. Lower result to NOS. Pop Stack.
DMUU	B6	36	182-218	Multiply NOS by TOS. Upper result to NOS. Pop Stack.
DDIV	AF	2F	196-210	Divide NOS by TOS. Result to NOS. Pop Stack.
32-BIT FLOATING-POINT PRIMARY OPERATIONS				
FADD	90	10	54-368	Add TOS to NOS. Result to NOS. Pop Stack.
FSUB	91	11	70-370	Subtract TOS from NOS. Result to NOS. Pop Stack.
FMUL	92	12	146-168	Multiply NOS by TOS. Result to NOS. Pop Stack.
FDIV	93	13	154-184	Divide NOS by TOS. Result to NOS. Pop Stack.
32-BIT FLOATING-POINT DERIVED OPERATIONS				
SQRT	81	01	782-870	Square Root of TOS. Result to TOS.
SIN	82	02	3796-4808	Sine of TOS. Result to TOS.
COS	83	03	3840-4878	Cosine of TOS. Result to TOS.
TAN	84	04	4894-5886	Tangent of TOS. Result to TOS.
ASIN	85	05	6230-7938	Inverse Sine of TOS. Result to TOS.
ACOS	86	06	6304-8284	Inverse Cosine of TOS. Result to TOS.
ATAN	87	07	4992-6536	Inverse Tangent of TOS. Result to TOS.
LOG	88	08	4474-7132	Common Logarithm of TOS. Result to TOS.
LN	89	09	4298-6956	Natural Logarithm of TOS. Result to TOS.
EXP	8A	0A	3794-4878	e raised to power in TOS. Result to TOS.
PWR	8B	0B	8290-12032	NOS raised to power in TOS. Result to NOS. Pop Stack.
DATA AND STACK MANIPULATION OPERATIONS				
NOP	80	00	4	No Operation. Clear or set SVREQ.
FIXS	9F	1F	90-214	Convert TOS from floating point format to fixed point format.
FIXD	9E	1E	90-336	
FLTS	9D	1D	62-156	Convert TOS from fixed point format to floating point format.
FLTD	9C	1C	56-342	
CHSS	F4	74	22-24	Change sign of fixed point operand on TOS.
CHSD	B4	34	26-28	
CHSF	95	15	16-20	Change sign of floating point operand on TOS.
PTOS	F7	77	16	Push stack. Duplicate NOS in TOS.
PTOD	B7	37	20	
PTOF	97	17	20	
POPS	F8	78	10	Pop stack. Old NOS becomes new TOS. Old TOS rotates to bottom.
POPD	B8	38	12	
POPF	98	18	12	
XCHS	F9	79	18	Exchange TOS and NOS.
XCHD	B9	39	26	
XCHF	99	19	26	
PUPI	9A	1A	16	Push floating point constant π onto TOS. Previous TOS becomes NOS.

COMMAND DESCRIPTIONS

This section contains detailed descriptions of the APU commands. They are arranged in alphabetical order by command mnemonic. In the descriptions, TOS means Top Of Stack and NOS means Next On Stack.

All derived functions except Square Root use Chebyshev polynomial approximating algorithms. This approach is used to help minimize the internal microprogram, to minimize the maximum error values and to provide a relatively even distribution of errors over the data range. The basic arithmetic operations are used by the derived functions to compute the various Chebyshev terms. The basic operations may produce error codes in the status register as a result.

Execution times are listed in terms of clock cycles and may be converted into time values by multiplying by the clock period used. For example, an execution time of 44 clock cycles when running at a 3MHz rate translates to 14 microseconds (44 x 32μs = 14μs). Variations in execution cycles reflect the data dependency of the algorithms.

In some operations exponent overflow or underflow may be possible. When this occurs, the exponent returned in the result will be 128 greater or smaller than its true value.

Many of the functions use portions of the data stack as scratch storage during development of the results. Thus previous values in those stack locations will be lost. Scratch locations destroyed are listed in the command descriptions and shown with the crossed-out locations in the Stack Contents After diagram.

Table 1 is a summary of all the Am9511A commands. It shows the hex codes for each command, the mnemonic abbreviation, a brief description and the execution time in clock cycles. The commands are grouped by functional classes.

The command mnemonics in alphabetical order are shown below in Table 2.

Table 2.

Command Mnemonics in Alphabetical Order.

ACOS	ARCCOSINE
ASIN	ARCSINE
ATAN	ARCTANGENT
CHSD	CHANGE SIGN DOUBLE
CHSF	CHANGE SIGN FLOATING
CHSS	CHANGE SIGN SINGLE
COS	COSINE
DADD	DOUBLE ADD
DDIV	DOUBLE DIVIDE
DMUL	DOUBLE MULTIPLY LOWER
DMUU	DOUBLE MULTIPLY UPPER
DSUB	DOUBLE SUBTRACT
EXP	EXPONENTIATION (e^x)
FADD	FLOATING ADD
FDIV	FLOATING DIVIDE
FIXD	FIX DOUBLE
FIXS	FIX SINGLE
FLTD	FLOAT DOUBLE
FLTS	FLOAT SINGLE
FMUL	FLOATING MULTIPLY
FSUB	FLOATING SUBTRACT
LOG	COMMON LOGARITHM
LN	NATURAL LOGARITHM
NOP	NO OPERATION
POPD	POP STACK DOUBLE
POPF	POP STACK FLOATING
POPS	POP STACK SINGLE
PTOD	PUSH STACK DOUBLE
PTOF	PUSH STACK FLOATING
PTOS	PUSH STACK SINGLE
PUPI	PUSH π
PWR	POWER (X^Y)
SADD	SINGLE ADD
SDIV	SINGLE DIVIDE
SIN	SINE
SMUL	SINGLE MULTIPLY LOWER
SMUU	SINGLE MULTIPLY UPPER
SQRT	SQUARE ROOT
SSUB	SINGLE SUBTRACT
TAN	TANGENT
XCHD	EXCHANGE OPERANDS DOUBLE
XCHF	EXCHANGE OPERANDS FLOATING
XCHS	EXCHANGE OPERANDS SINGLE

ACOS

32-BIT FLOATING-POINT INVERSE COSINE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	1	1	0

Hex Coding: 86 with sr = 1
06 with sr = 0

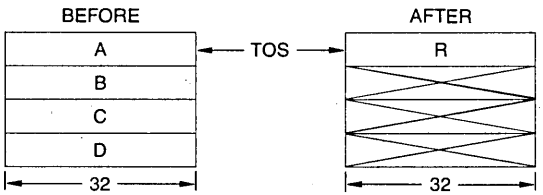
Execution Time: 6304 to 8284 clock cycles

Description:
The 32-bit floating-point operand A at the TOS is replaced by the 32-bit floating-point inverse cosine of A. The result R is a value in radians between 0 and π. Initial operands A, B, C and D are lost. ACOS will accept all input data values within the range of −1.0 to +1.0. Values outside this range will return an error code of 1100 in the status register.

Accuracy: ACOS exhibits a maximum relative error of 2.0 x 10^{-7} over the valid input data range.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



ATAN

32-BIT FLOATING-POINT INVERSE TANGENT

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	1	1	1

Hex Coding: 87 with sr = 1
07 with sr = 0

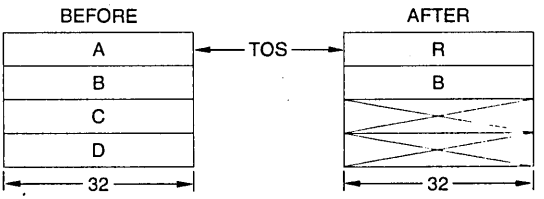
Execution Time: 4992 to 6536 clock cycles

Description:
The 32-bit floating-point operand A at the TOS is replaced by the 32-bit floating-point inverse tangent of A. The result R is a value in radians between $-\pi/2$ and $+\pi/2$. Initial operands A, C and D are lost. Operand B is unchanged.
ATAN will accept all input data values that can be represented in the floating point format.

Accuracy: ATAN exhibits a maximum relative error of 3.0 x 10^{-7} over the input data range.

Status Affected: Sign, Zero

STACK CONTENTS



ASIN

32-BIT FLOATING-POINT INVERSE SINE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	1	0	1

Hex Coding: 85 with sr = 1
05 with sr = 0

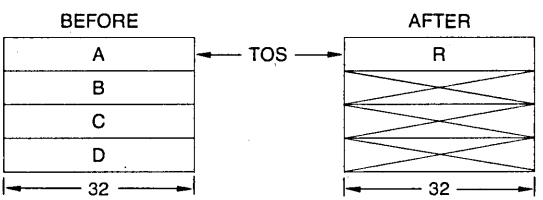
Execution Time: 6230 to 7938 clock cycles

Description:
The 32-bit floating-point operand A at the TOS is replaced by the 32-bit floating-point inverse sine of A. The result R is a value in radians between $-\pi/2$ and $+\pi/2$. Initial operands A, B, C and D are lost.
ASIN will accept all input data values within the range of −1.0 to +1.0. Values outside this range will return an error code of 1100 in the status register.

Accuracy: ASIN exhibits a maximum relative error of 4.0 x 10^{-7} over the valid input data range.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



CHSD

32-BIT FIXED-POINT SIGN CHANGE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	1	0	1	0	0

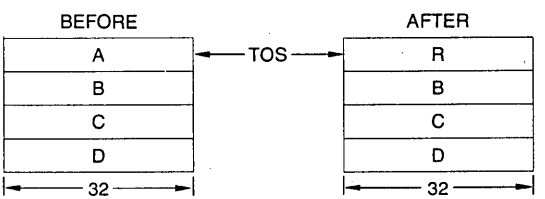
Hex Coding: B4 with sr = 1
34 with sr = 0

Execution Time: 26 to 28 clock cycles

Description:
The 32-bit fixed-point two's complement integer operand A at the TOS is subtracted from zero. The result R replaces A at the TOS. Other entries in the stack are not disturbed.
Overflow status will be set and the TOS will be returned unchanged when A is input as the most negative value possible in the format since no positive equivalent exists.

Status Affected: Sign, Zero, Error Field (overflow)

STACK CONTENTS



CHSF

32-BIT FLOATING-POINT SIGN CHANGE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	0	1	0	1

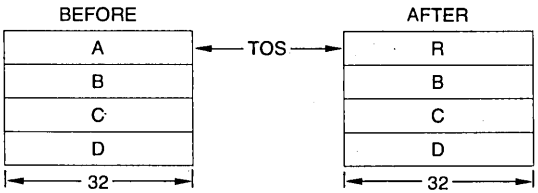
Hex Coding: 95 with sr = 1
15 with sr = 0

Execution Time: 16 to 20 clock cycles

Description:
The sign of the mantissa of the 32-bit floating-point operand A at the TOS is inverted. The result R replaces A at the TOS. Other stack entries are unchanged.
If A is input as zero (mantissa MSB = 0), no change is made.

Status Affected: Sign, Zero

STACK CONTENTS



CHSS

16-BIT FIXED-POINT SIGN CHANGE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	1	0	1	0	0

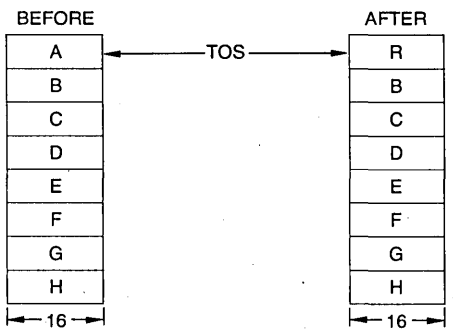
Hex Coding: F4 with sr = 1
74 with sr = 0

Execution Time: 22 to 24 clock cycles

Description:
16-bit fixed-point two's complement integer operand A at the TOS is subtracted from zero. The result R replaces A at the TOS. All other operands are unchanged.
Overflow status will be set and the TOS will be returned unchanged when A is input as the most negative value possible in the format since no positive equivalent exists.

Status Affected: Sign, Zero, Overflow

STACK CONTENTS



COS

32-BIT FLOATING-POINT COSINE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	0	1	1

Hex Coding: 83 with sr = 1
03 with sr = 0

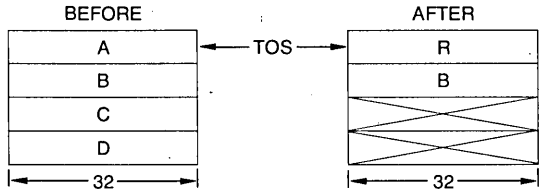
Execution Time: 3840 to 4878 clock cycles

Description:
The 32-bit floating-point operand A at the TOS is replaced by R, the 32-bit floating-point cosine of A. A is assumed to be in radians. Operands A, C and D are lost. B is unchanged.
The COS function can accept any input data value that can be represented in the data format. All input values are range reduced to fall within an interval of $-\pi/2$ to $+\pi/2$ radians.

Accuracy: COS exhibits a maximum relative error of 5.0 x 10^{-7} for all input data values in the range of -2π to $+2\pi$ radians.

Status Affected: Sign, Zero

STACK CONTENTS



DADD

32-BIT FIXED-POINT ADD

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	0	1	1	0	0

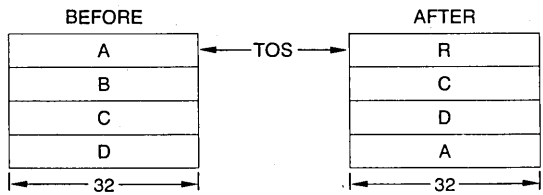
Hex Coding: AC with sr = 1
2C with sr = 0

Execution Time: 20 to 22 clock cycles

Description:
The 32-bit fixed-point two's complement integer operand A at the TOS is added to the 32-bit fixed-point two's complement integer operand B at the NOS. The result R replaces operand B and the Stack is moved up so that R occupies the TOS. Operand B is lost. Operands A, C and D are unchanged. If the addition generates a carry it is reported in the status register.
If the result is too large to be represented by the data format, the least significant 32 bits of the result are returned and overflow status is reported.

Status Affected: Sign, Zero, Carry, Error Field

STACK CONTENTS



DDIV

32-BIT FIXED-POINT DIVIDE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	0	1	1	1	1

Hex Coding: AF with sr = 1
2F with sr = 0

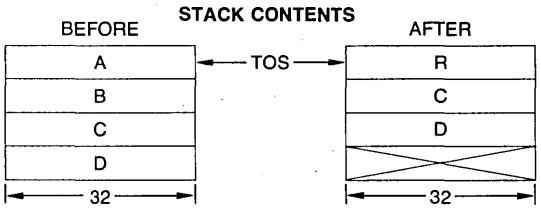
Execution Time: 196 to 210 clock cycles when A ≠ 0
18 clock cycles when A = 0.

Description:

The 32-bit fixed-point two's complement integer operand B at NOS is divided by the 32-bit fixed-point two's complement integer operand A at the TOS. The 32-bit integer quotient R replaces B and the stack is moved up so that R occupies the TOS. No remainder is generated. Operands A and B are lost. Operands C and D are unchanged.

If A is zero, R is set equal to B and the divide-by-zero error status will be reported. If either A or B is the most negative value possible in the format, R will be meaningless and the overflow error status will be reported.

Status Affected: Sign, Zero, Error Field



DMUL

32-BIT FIXED-POINT MULTIPLY, LOWER

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	0	1	1	1	0

Hex Coding: AE with sr = 1
2E with sr = 0

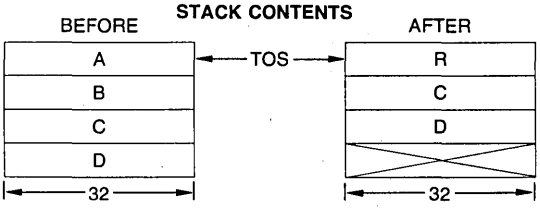
Execution Time: 194 to 210 clock cycles

Description:

The 32-bit fixed-point two's complement integer operand A at the TOS is multiplied by the 32-bit fixed-point two's complement integer operand B at the NOS. The 32-bit least significant half of the product R replaces B and the stack is moved up so that R occupies the TOS. The most significant half of the product is lost. Operands A and B are lost. Operands C and D are unchanged.

The overflow status bit is set if the discarded upper half was non-zero. If either A or B is the most negative value that can be represented in the format, that value is returned as R and the overflow status is set.

Status Affected: Sign, Zero, Overflow



DMUU

32-BIT FIXED-POINT MULTIPLY, UPPER

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	1	0	1	1	0

Hex Coding: B6 with sr = 1
36 with sr = 0

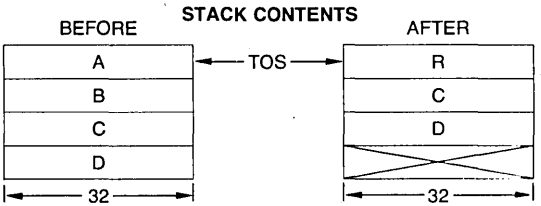
Execution Time: 182 to 218 clock cycles

Description:

The 32-bit fixed-point two's complement integer operand A at the TOS is multiplied by the 32-bit fixed-point two's complement integer operand B at the NOS. The 32-bit most significant half of the product R replaces B and the stack is moved up so that R occupies the TOS. The least significant half of the product is lost. Operands A and B are lost. Operands C and D are unchanged.

If A or B was the most negative value possible in the format, overflow status is set and R is meaningless.

Status Affected: Sign, Zero, Overflow



DSUB

32-BIT FIXED-POINT SUBTRACT

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	0	1	1	0	1

Hex Coding: AD with sr = 1
2D with sr = 0

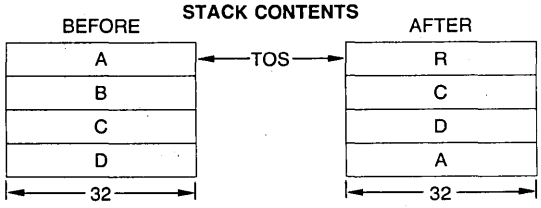
Execution Time: 38 to 40 clock cycles

Description:

The 32-bit fixed-point two's complement operand A at the TOS is subtracted from the 32-bit fixed-point two's complement operand B at the NOS. The difference R replaces operand B and the stack is moved up so that R occupies the TOS. Operand B is lost. Operands A, C and D are unchanged.

If the subtraction generates a borrow it is reported in the carry status bit. If A is the most negative value that can be represented in the format the overflow status is set. If the result cannot be represented in the data format range, the overflow bit is set and the 32 least significant bits of the result are returned as R.

Status Affected: Sign, Zero, Carry, Overflow



EXP

32-BIT FLOATING-POINT e^X

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	1	0	1	0

Hex Coding: 8A with sr = 1
0A with sr = 0

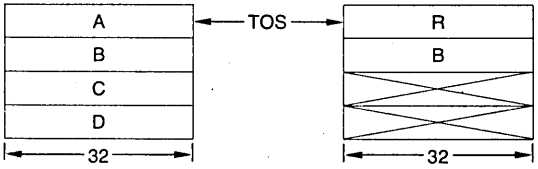
Execution Time: 3794 to 4878 clock cycles for $|A| \leq 1.0 \times 2^5$
34 clock cycles for $|A| > 1.0 \times 2^5$

Description:
The base of natural logarithms, e, is raised to an exponent value specified by the 32-bit floating-point operand A at the TOS. The result R of e^A replaces A. Operands A, C and D are lost. Operand B is unchanged.
EXP accepts all input data values within the range of $-1.0 \times 2^{+5}$ to $+1.0 \times 2^{+5}$. Input values outside this range will return a code of 1100 in the error field of the status register.

Accuracy: EXP exhibits a maximum relative error of 5.0×10^{-7} over the valid input data range.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



FADD

32-BIT FLOATING-POINT ADD

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	0	0	0	0

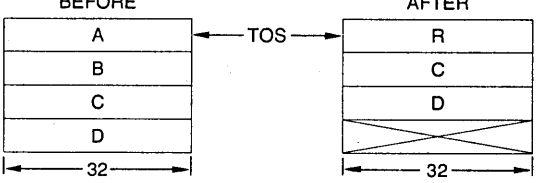
Hex Coding: 90 with sr = 1
10 with sr = 0

Execution Time: 54 to 368 clock cycles for A ≠ 0
24 clock cycles for A = 0

Description:
32-bit floating-point operand A at the TOS is added to 32-bit floating-point operand B at the NOS. The result R replaces B and the stack is moved up so that R occupies the TOS. Operands A and B are lost. Operands C and D are unchanged.
Exponent alignment before the addition and normalization of the result accounts for the variation in execution time. Exponent overflow and underflow are reported in the status register, in which case the mantissa is correct and the exponent is offset by 128.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



FDIV

32-BIT FLOATING-POINT DIVIDE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	0	0	1	1

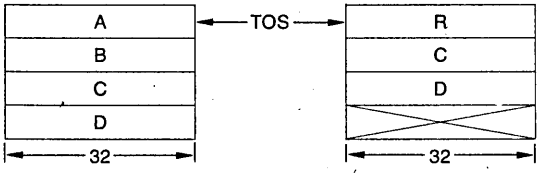
Hex Coding: 93 with sr = 1
13 with sr = 0

Execution Time: 154 to 184 clock cycles for A ≠ 0
22 clock cycles for A = 0

Description:
32-bit floating-point operand B at NOS is divided by 32-bit floating-point operand A at the TOS. The result R replaces B and the stack is moved up so that R occupies the TOS. Operands A and B are lost. Operands C and D are unchanged.
If operand A is zero, R is set equal to B and the divide-by-zero error is reported in the status register. Exponent overflow or underflow is reported in the status register, in which case the mantissa portion of the result is correct and the exponent portion is offset by 128.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



FIXD

32-BIT FLOATING-POINT TO 32-BIT FIXED-POINT CONVERSION

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	1	1	0

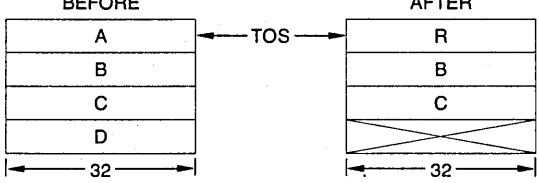
Hex Coding: 9E with sr = 1
1E with sr = 0

Execution Time: 90 to 336 clock cycles

Description:
32-bit floating-point operand A at the TOS is converted to a 32-bit fixed-point two's complement integer. The result R replaces A. Operands A and D are lost. Operands B and C are unchanged.
If the integer portion of A is larger than 31 bits when converted, the overflow status will be set and A will not be changed. Operand D, however, will still be lost.

Status Affected: Sign, Zero Overflow

STACK CONTENTS



FIXS

32-BIT FLOATING-POINT TO 16-BIT FIXED-POINT CONVERSION

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	1	1	1

Hex Coding: 9F with sr = 1
1F with sr = 0

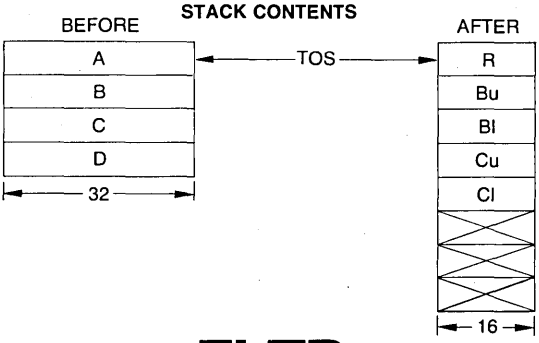
Execution Time: 90 to 214 clock cycles

Description:

32-bit floating-point operand A at the TOS is converted to a 16-bit fixed-point two's complement integer. The result R replaces the lower half of A and the stack is moved up by two bytes so that R occupies the TOS. Operands A and D are lost. Operands B and C are unchanged, but appear as upper (u) and lower (l) halves on the 16-bit wide stack if they are 32-bit operands.

If the integer portion of A is larger than 15 bits when converted, the overflow status will be set and A will not be changed. Operand D, however, will still be lost.

Status Affected: Sign, Zero, Overflow



FLTD

32-BIT FIXED-POINT TO 32-BIT FLOATING-POINT CONVERSION

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	1	0	0

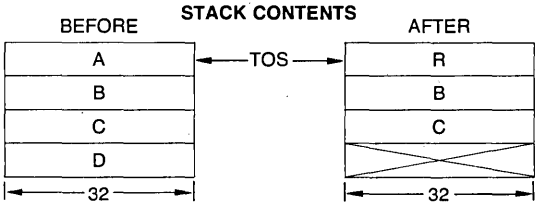
Hex Coding: 9C with sr = 1
1C with sr = 0

Execution Time: 56 to 342 clock cycles

Description:

32-bit fixed-point two's complement integer operand A at the TOS is converted to a 32-bit floating-point number. The result R replaces A at the TOS. Operands A and D are lost. Operands B and C are unchanged.

Status Affected: Sign, Zero



FLTS

16-BIT FIXED-POINT TO 32-BIT FLOATING-POINT CONVERSION

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	1	0	1

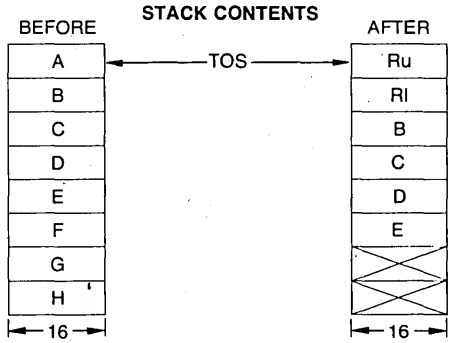
Hex Coding: 9D with sr = 1
1D with sr = 0

Execution Time: 62 to 156 clock cycles

Description:

16-bit fixed-point two's complement integer A at the TOS is converted to a 32-bit floating-point number. The lower half of the result R (Rl) replaces A, the upper half (Ru) replaces H and the stack is moved down so that Ru occupies the TOS. Operands A, F, G and H are lost. Operands B, C, D and E are unchanged.

Status Affected: Sign, Zero



FMUL

32-BIT FLOATING-POINT MULTIPLY

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	0	0	1	0

Hex Coding: 92 with sr = 1
12 with sr = 0

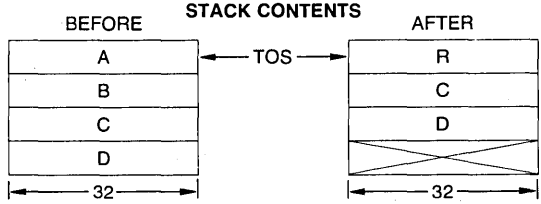
Execution Time: 146 to 168 clock cycles

Description:

32-bit floating-point operand A at the TOS is multiplied by the 32-bit floating-point operand B at the NOS. The normalized result R replaces B and the stack is moved up so that R occupies the TOS. Operands A and B are lost. Operands C and D are unchanged.

Exponent overflow or underflow is reported in the status register, in which case the mantissa portion of the result is correct and the exponent portion is offset by 128.

Status Affected: Sign, Zero, Error Field



FSUB

32-BIT FLOATING-POINT SUBTRACTION

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	0	0	0	1

Hex Coding: 91 with sr = 1
11 with sr = 0

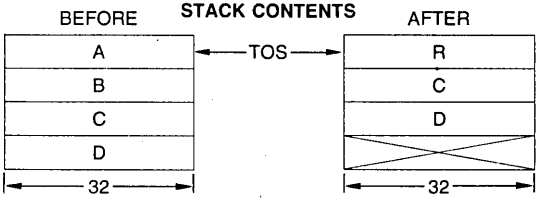
Execution Time: 70 to 370 clock cycles for A ≠ 0
26 clock cycles for A = 0

Description:
32-bit floating-point operand A at the TOS is subtracted from 32-bit floating-point operand B at the NOS. The normalized difference R replaces B and the stack is moved up so that R occupies the TOS. Operands A and B are lost. Operands C and D are unchanged.

Exponent alignment before the subtraction and normalization of the result account for the variation in execution time.

Exponent overflow or underflow is reported in the status register in which case the mantissa portion of the result is correct and the exponent portion is offset by 128.

Status Affected: Sign, Zero, Error Field (overflow)



LOG

32-BIT FLOATING-POINT COMMON LOGARITHM

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	1	0	0	0

Hex Coding: 88 with sr = 1
08 with sr = 0

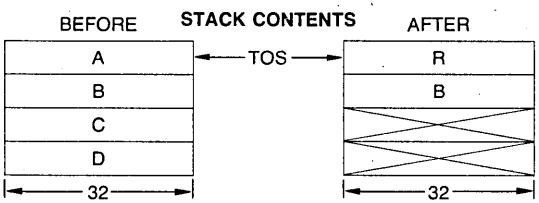
Execution Time: 4474 to 7132 clock cycles for A > 0
20 clock cycles for A ≤ 0

Description:
The 32-bit floating-point operand A at the TOS is replaced by R, the 32-bit floating-point common logarithm (base 10) of A. Operands A, C and D are lost. Operand B is unchanged.

The LOG function accepts any positive input data value that can be represented by the data format. If LOG of a non-positive value is attempted an error status of 0100 is returned.

Accuracy: LOG exhibits a maximum absolute error of 2.0×10^{-7} for the input range from 0.1 to 10, and a maximum relative error of 2.0×10^{-7} for positive values less than 0.1 or greater than 10.

Status Affected: Sign, Zero, Error Field



LN

32-BIT FLOATING-POINT NATURAL LOGARITHM

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	1	0	0	1

Hex Coding: 89 with sr = 1
09 with sr = 0

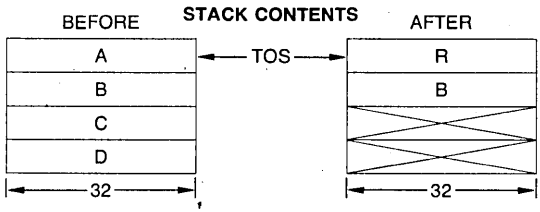
Execution Time: 4298 to 6956 clock cycles for A > 0
20 clock cycles for A ≤ 0

Description:
The 32-bit floating-point operand A at the TOS is replaced by R, the 32-bit floating-point natural logarithm (base e) of A. Operands A, C and D are lost. Operand B is unchanged.

The LN function accepts all positive input data values that can be represented by the data format. If LN of a non-positive number is attempted an error status of 0100 is returned.

Accuracy: LN exhibits a maximum absolute error of 2×10^{-7} for the input range from e^{-1} to e, and a maximum relative error of 2.0×10^{-7} for positive values less than e^{-1} or greater than e.

Status Affected: Sign, Zero, Error Field



NOP

NO OPERATION

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	0	0	0

Hex Coding: 80 with sr = 1
00 with sr = 0

Execution Time: 4 clock cycles

Description:
The NOP command performs no internal data manipulations. It may be used to set or clear the service request interface line without changing the contents of the stack.

Status Affected: The status byte is cleared to all zeroes.

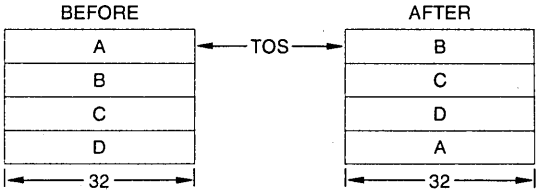
POPD

32-BIT
STACK POP

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	1	1	0	0	0

Hex Coding: B8 with sr = 1
38 with sr = 0
Execution Time: 12 clock cycles
Description:
The 32-bit stack is moved up so that the old NOS becomes the new TOS. The previous TOS rotates to the bottom of the stack. All operand values are unchanged. POPD and POPF execute the same operation.
Status Affected: Sign, Zero

STACK CONTENTS



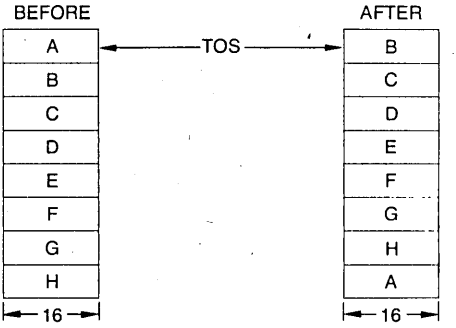
POPS

16-BIT
STACK POP

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	1	1	0	0	0

Hex Coding: F8 with sr = 1
78 with sr = 0
Execution Time: 10 clock cycles
Description:
The 16-bit stack is moved up so that the old NOS becomes the new TOS. The previous TOS rotates to the bottom of the stack. All operand values are unchanged.
Status Affected: Sign, Zero

STACK CONTENTS



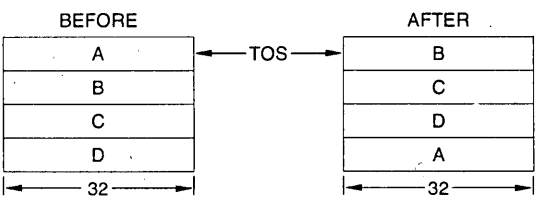
POPF

32-BIT
STACK POP

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	0	0	0

Hex Coding: 98 with sr = 1
18 with sr = 0
Execution Time: 12 clock cycles
Description:
The 32-bit stack is moved up so that the old NOS becomes the new TOS. The old TOS rotates to the bottom of the stack. All operand values are unchanged. POPF and POPD execute the same operation.
Status Affected: Sign, Zero

STACK CONTENTS



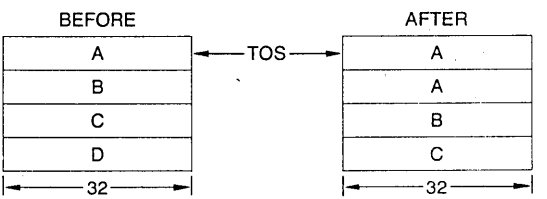
PTOD

PUSH 32-BIT
TOS ONTO STACK

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	1	0	1	1	1

Hex Coding: B7 with sr = 1
37 with sr = 0
Execution Time: 20 clock cycles
Description:
The 32-bit stack is moved down and the previous TOS is copied into the new TOS location. Operand D is lost. All other operand values are unchanged. PTOD and PTOF execute the same operation.
Status Affected: Sign, Zero

STACK CONTENTS



PTOF

PUSH 32-BIT TOS ONTO STACK

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	0	1	1	1

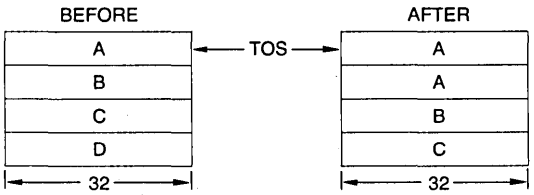
Hex Coding: 97 with sr = 1
17 with sr = 0

Execution Time: 20 clock cycles

Description:
The 32-bit stack is moved down and the previous TOS is copied into the new TOS location. Operand D is lost. All other operand values are unchanged. PTOF and PTOD execute the same operation.

Status Affected: Sign, Zero

STACK CONTENTS



PTOS

PUSH 16-BIT TOS ONTO STACK

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	1	0	1	1	1

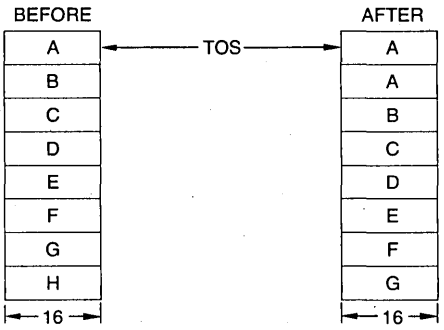
Hex Coding: F7 with sr = 1
77 with sr = 0

Execution Time: 16 clock cycles

Description:
The 16-bit stack is moved down and the previous TOS is copied into the new TOS location. Operand H is lost and all other operand values are unchanged.

Status Affected: Sign, Zero

STACK CONTENTS



PUPI

PUSH 32-BIT FLOATING-POINT π

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	0	1	0

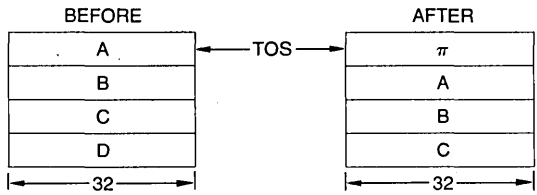
Hex Coding: 9A with sr = 1
1A with sr = 0

Execution Time: 16 clock cycles

Description:
The 32-bit stack is moved down so that the previous TOS occupies the new NOS location. 32-bit floating-point constant π is entered into the new TOS location. Operand D is lost. Operands A, B and C are unchanged.

Status Affected: Sign, Zero

STACK CONTENTS



PWR

32-BIT FLOATING-POINT X^Y

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	1	0	1	1

Hex Coding: 8B with sr = 1
0B with sr = 0

Execution Time: 8290 to 12032 clock cycles

Description:

32-bit floating-point operand B at the NOS is raised to the power specified by the 32-bit floating-point operand A at the TOS. The result R of B^A replaces B and the stack is moved up so that R occupies the TOS. Operands A, B, and D are lost. Operand C is unchanged.

The PWR function accepts all input data values that can be represented in the data format for operand A and all positive values for operand B. If operand B is non-positive an error status of 0100 will be returned. The EXP and LN functions are used to implement PWR using the relationship $B^A = EXP\,[A(LN\ B)]$. Thus if the term $[A(LN\ B)]$ is outside the range of $-1.0 \times 2^{+5}$ to $+1.0 \times 2^{+5}$ an error status of 1100 will be returned. Underflow and overflow conditions can occur.

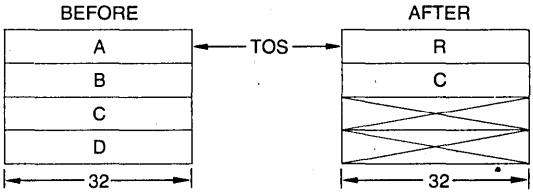
Accuracy: The error performance for PWR is a function of the LN and EXP performance as expressed by:

$|(\text{Relative Error})_{PWR}| = |(\text{Relative Error})_{EXP} + |A(\text{Absolute Error})_{LN}|$

The maximum relative error for PWR occurs when A is at its maximum value while $[A(LN\ B)]$ is near 1.0×2^5 and the EXP error is also at its maximum. For most practical applications the relative error for PWR will be less than 7.0×10^{-7}.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



SADD

16-BIT FIXED-POINT ADD

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	0	1	1	0	0

Hex Coding: EC with sr = 1
6C with sr = 0

Execution Time: 16 to 18 clock cycles

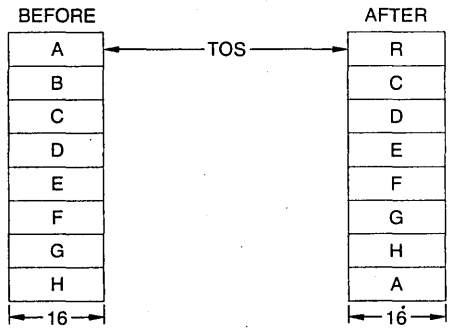
Description:

16-bit fixed-point two's complement integer operand A at the TOS is added to 16-bit fixed-point two's complement integer operand B at the NOS. The result R replaces B and the stack is moved up so that R occupies the TOS. Operand B is lost. All other operands are unchanged.

If the addition generates a carry bit it is reported in the status register. If an overflow occurs it is reported in the status register and the 16 least significant bits of the result are returned.

Status Affected: Sign, Zero, Carry, Error Field

STACK CONTENTS



SDIV

16-BIT FIXED-POINT DIVIDE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	0	1	1	1	1

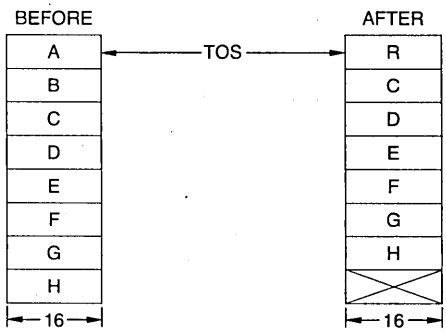
Hex Coding: EF with sr = 1
6F with sr = 0

Execution Time: 84 to 94 clock cycles for A ≠ 0
14 clock cycles for A = 0

Description:
16-bit fixed-point two's complement integer operand B at the NOS is divided by 16-bit fixed-point two's complement integer operand A at the TOS. The 16-bit integer quotient R replaces B and the stack is moved up so that R occupies the TOS. No remainder is generated. Operands A and B are lost. All other operands are unchanged.
If A is zero, R will be set equal to B and the divide-by-zero error status will be reported.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



SIN

32-BIT FLOATING-POINT SINE

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	0	1	0

Hex Coding: 82 with sr = 1
02 with sr = 0

Execution Time: 3796 to 4808 clock cycles for $|A| > 2^{-12}$ radians
30 clock cycles for $|A| \leq 2^{-12}$ radians

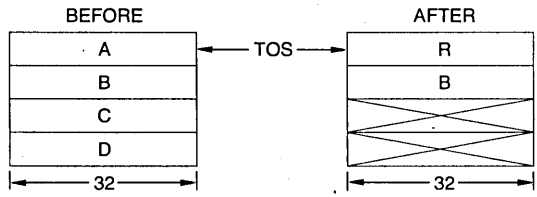
Description:
The 32-bit floating-point operand A at the TOS is replaced by R, the 32-bit floating-point sine of A. A is assumed to be in radians. Operands A, C and D are lost. Operand B is unchanged.

The SIN function will accept any input data value that can be represented by the data format. All input values are range reduced to fall within the interval $-\pi/2$ to $+\pi/2$ radians.

Accuracy: SIN exhibits a maximum relative error of 5.0×10^{-7} for input values in the range of -2π to $+2\pi$ radians.

Status Affected: Sign, Zero

STACK CONTENTS



SMUL

16-BIT FIXED-POINT MULTIPLY, LOWER

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	0	1	1	1	0

Hex Coding: EE with sr = 1
6E with sr = 0

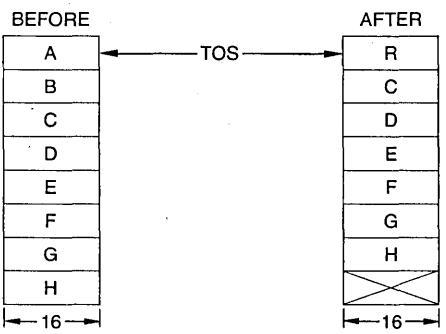
Execution Time: 84 to 94 clock cycles

Description:

16-bit fixed-point two's complement integer operand A at the TOS is multiplied by the 16-bit fixed-point two's complement integer operand B at the NOS. The 16-bit least significant half of the product R replaces B and the stack is moved up so that R occupies the TOS. The most significant half of the product is lost. Operands A and B are lost. All other operands are unchanged. The overflow status bit is set if the discarded upper half was non-zero. If either A or B is the most negative value that can be represented in the format, that value is returned as R and the overflow status is set.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



SMUU

16-BIT FIXED-POINT MULTIPLY, UPPER

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	1	0	1	1	0

Hex Coding: F6 with sr = 1
76 with sr = 0

Execution Time: 80 to 98 clock cycles

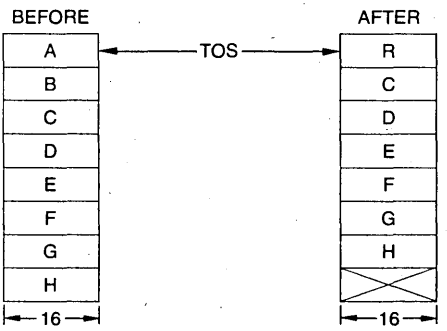
Description:

16-bit fixed-point two's complement integer operand A at the TOS is multiplied by the 16-bit fixed-point two's complement integer operand B at the NOS. The 16-bit most significant half of the product R replaces B and the stack is moved up so that R occupies the TOS. The least significant half of the product is lost. Operands A and B are lost. All other operands are unchanged.

If either A or B is the most negative value that can be represented in the format, that value is returned as R and the overflow status is set.

Status Affected: Sign, Zero, Error Field

STACK CONTENTS



SQRT

32-BIT FLOATING-POINT SQUARE ROOT

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	0	0	1

Hex Coding: 81 with sr = 1
01 with sr = 0

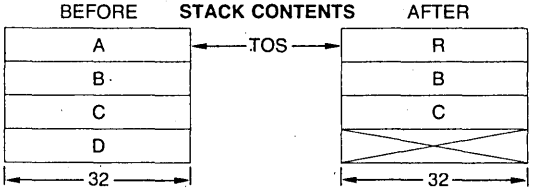
Execution Time: 782 to 870 clock cycles

Description:

32-bit floating-point operand A at the TOS is replaced by R, the 32-bit floating-point square root of A. Operands A and D are lost. Operands B and C are not changed.

SQRT will accept any non-negative input data value that can be represented by the data format. If A is negative an error code of 0100 will be returned in the status register.

Status Affected: Sign, Zero, Error Field



SSUB

16-BIT FIXED-POINT SUBTRACT

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	0	1	1	0	1

Hex Coding: ED with sr = 1
6D with sr = 0

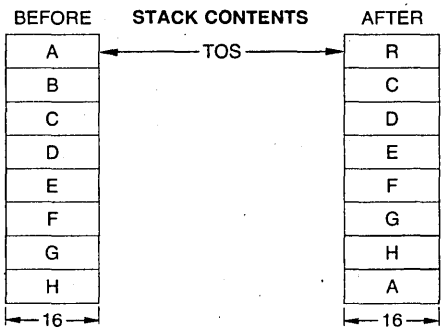
Execution Time: 30 to 32 clock cycles

Description:

16-bit fixed-point two's complement integer operand A at the TOS is subtracted from 16-bit fixed-point two's complement integer operand B at the NOS. The result R replaces B and the stack is moved up so that R occupies the TOS. Operand B is lost. All other operands are unchanged.

If the subtraction generates a borrow it is reported in the carry status bit. If A is the most negative value that can be represented in the format the overflow status is set. If the result cannot be represented in the format range, the overflow status is set and the 16 least significant bits of the result are returned as R.

Status Affected: Sign, Zero, Carry, Error Field



TAN

32-BIT FLOATING-POINT TANGENT

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	0	0	1	0	0

Hex Coding: 84 with sr = 1
04 with sr = 0

Execution Time: 4894 to 5886 clock cycles for $|A| > 2^{-12}$ radians
30 clock cycles for $|A| \leq 2^{-12}$ radians

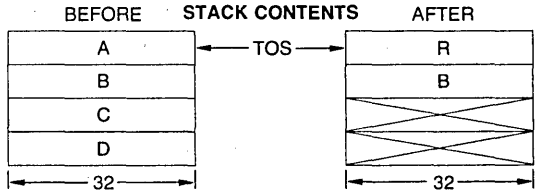
Description:

The 32-bit floating-point operand A at the TOS is replaced by the 32-bit floating-point tangent of A. Operand A is assumed to be in radians. A, C and D are lost. B is unchanged.

The TAN function will accept any input data value that can be represented in the data format. All input data values are range-reduced to fall within $-\pi/4$ to $+\pi/4$ radians. TAN is unbounded for input values near odd multiples of $\pi/2$ and in such cases the overflow bit is set in the status register. For angles smaller than 2^{-12} radians, TAN returns A as the tangent of A.

Accuracy: TAN exhibits a maximum relative error of 5.0 x 10^{-7} for input data values in the range of -2π to $+2\pi$ radians except for data values near odd multiples of $\pi/2$.

Status Affected: Sign, Zero, Error Field (overflow)



XCHD

EXCHANGE 32-BIT STACK OPERANDS

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	1	1	1	0	0	1

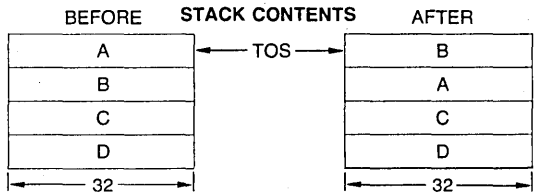
Hex Coding: B9 with sr = 1
39 with sr = 0

Execution Time: 26 clock cycles

Description:

32-bit operand A at the TOS and 32-bit operand B at the NOS are exchanged. After execution, B is at the TOS and A is at the NOS. All operands are unchanged. XCHD and XCHF execute the same operation.

Status Affected: Sign, Zero



XCHF

EXCHANGE 32-BIT STACK OPERANDS

	7	6	5	4	3	2	1	0
Binary Coding:	sr	0	0	1	1	0	0	1

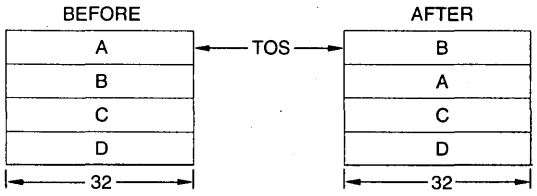
Hex Coding: 99 with sr = 1
19 with sr = 0

Execution Time: 26 clock cycles

Description:
32-bit operand A at the TOS and 32-bit operand B at the NOS are exchanged. After execution, B is at the TOS and A is at the NOS. All operands are unchanged. XCHD and XCHF execute the same operation.

Status Affected: Sign, Zero

STACK CONTENTS



XCHS

EXCHANGE 16-BIT STACK OPERANDS

	7	6	5	4	3	2	1	0
Binary Coding:	sr	1	1	1	1	0	0	1

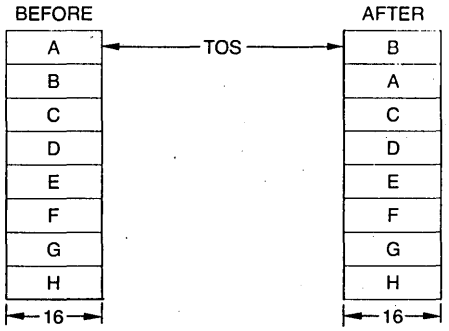
Hex Coding: F9 with sr = 1
79 with sr = 0

Execution Time: 18 clock cycles

Description:
16-bit operand A at the TOS and 16-bit operand B at the NOS are exchanged. After execution, B is at the TOS and A is at the NOS. All operand values are unchanged.

Status Affected: Sign, Zero

STACK CONTENTS



MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65 to +150°C
VDD with Respect to VSS	−0.5V to +15.0V
VCC with Respect to VSS	−0.5V to +7.0V
All Signal Voltages with Respect to VSS	−0.5V to +7.0V
Power Dissipation (Package Limitation)	2.0W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	Ambient Temperature	VSS	VCC	VDD
Am9511ADC	0°C ≤ T_A ≤ 70°C	0V	+5.0V ±5%	+12V ±5%
Am9511A-1DC	0°C ≤ T_A ≤ 70°C	0V	+5.0V ±5%	+12V ±5%
Am9511A-4DC	0°C ≤ T_A ≤ 70°C	0V	+5.0V ±5%	+12V ±5%
Am9511ADI	−40°C ≤ T_A ≤ 85°C	0V	+5.0V ±10%	+12V ±10%
Am9511A-1DI	−40°C ≤ T_A ≤ 85°C	0V	+5.0V ±10%	+12V ±10%
Am9511ADM	−55°C ≤ T_A ≤ 125°C	0V	+5.0V ±10%	+12V ±10%
Am9511A-1DM	−55°C ≤ T_A ≤ 125°C	0V	+5.0V ±10%	+12V ±10%

ELECTRICAL CHARACTERISTICS Over Operating Range (Note 1)

Parameters	Description	Test Conditions	Min.	Typ.	Max.	Units
VOH	Output HIGH Voltage	IOH = −200μA	3.7			Volts
VOL	Output LOW Voltage	IOL = 3.2mA			0.4	Volts
VIH	Input HIGH Voltage		2.0		VCC	Volts
VIL	Input LOW Voltage		−0.5		0.8	Volts
IIX	Input Load Current	VSS ≤ VI ≤ VCC			±10	μA
IOZ	Data Bus Leakage	VO = 0.4V			10	μA
		VO = VCC			10	
ICC	VCC Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
IDD	VDD Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
CO	Output Capacitance	fc = 1.0MHz, Inputs = 0V		8	10	pF
CI	Input Capacitance			5	8	pF
CIO	I/O Capacitance			10	12	pF

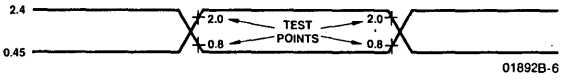
SWITCHING CHARACTERISTICS

Parameters	Description		Am9511A Min	Am9511A Max	Am9511A-1 Min	Am9511A-1 Max	Am9511A-4 Min	Am9511A-4 Max	Units
TAPW	$\overline{EACK}$ LOW Pulse Width		100		75		50		ns
TCDR	C/$\overline{D}$ to $\overline{RD}$ LOW Set-up Time		0		0		0		ns
TCDW	C/$\overline{D}$ to $\overline{WR}$ LOW Set-up Time		0		0		0		ns
TCPH	Clock Pulse HIGH Width		200		140		100		ns
TCPL	Clock Pulse LOW Width		240		160		120		ns
TCSR	$\overline{CS}$ LOW to $\overline{RD}$ LOW Set-up Time		0		0		0		ns
TCSW	$\overline{CS}$ LOW to $\overline{WR}$ LOW Set-up Time		0		0		0		ns
TCY	Clock Period		480	5000	320	3300	250	2500	ns
TDW	Data Bus Stable to $\overline{WR}$ HIGH Set-up Time		150		100 (Note 9)		100		ns
TEAE	$\overline{EACK}$ LOW to $\overline{END}$ HIGH Delay			200		175		150	ns
TEPW	$\overline{END}$ LOW Pulse Width (Note 4)		400		300		200		ns
TOP	Data Bus Output Valid to $\overline{PAUSE}$ HIGH Delay		0		0		0		ns
TPPWR	$\overline{PAUSE}$ LOW Pulse Width Read (Note 5)	Data	3.5TCY+50	5.5TCY+300	3.5TCY+50	5.5TCY+200	3.5TCY+50	5.5TCY+200	ns
		Status	1.5TCY+50	3.5TCY+300	1.5TCY+50	3.5TCY+200	1.5TCY+50	3.5TCY+200	
TPPWW	$\overline{PAUSE}$ LOW Pulse Width Write (Note 8)			50		50		50	ns
TPR	$\overline{PAUSE}$ HIGH to $\overline{RD}$ HIGH Hold Time		0		0		0		ns
TPW	$\overline{PAUSE}$ HIGH to $\overline{WR}$ HIGH Hold Time		0		0		0		ns
TRCD	$\overline{RD}$ HIGH to C/$\overline{D}$ Hold Time		0		0		0		ns
TRCS	$\overline{RD}$ HIGH to $\overline{CS}$ HIGH Hold Time		0		0		0		ns
TRO	$\overline{RD}$ LOW to Data Bus ON Delay		50		50		25		ns
TRP	$\overline{RD}$ LOW to $\overline{PAUSE}$ LOW Delay (Note 6)			150		100 (Note 9)		100	ns
TRZ	$\overline{RD}$ HIGH to Data Bus OFF Delay		50	200	50	150	25	100	ns
TSAPW	$\overline{SVACK}$ LOW Pulse Width		100		75		50		ns
TSAR	$\overline{SVACK}$ LOW to SVREQ LOW Delay			300		200		150	ns
TWCD	$\overline{WR}$ HIGH to C/$\overline{D}$ Hold Time		60		30		30		ns
TWCS	$\overline{WR}$ HIGH to $\overline{CS}$ HIGH Hold Time		60		30		30		ns
TWD	$\overline{WR}$ HIGH to Data Bus Hold Time		20		20		20		ns
TWI	Write Inactive Time	Command	3TCY		3TCY		3TCY		ns
		Data	4TCY		4TCY		4TCY		
TWP	$\overline{WR}$ LOW to $\overline{PAUSE}$ LOW Delay (Note 6)			150		100 (Note 9)		100	ns

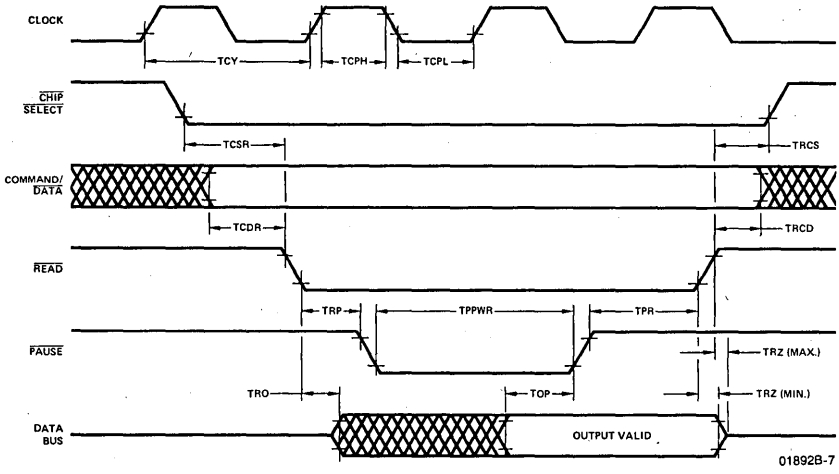
Notes:
1. Typical values are for T_A = 25°C, nominal supply voltages and nominal processing parameters.
2. Switching parameters are listed in alphabetical order.
3. Test conditions assume transition times of 20ns or less, output loading of one TTL gate plus 100pF and timing reference levels of 0.8V and 2.0V.
4. $\overline{END}$ low pulse width is specified for $\overline{EACK}$ tied to VSS. Otherwise TEAE applies.
5. Minimum values shown assume no previously entered command is being executed for the data access. If a previously entered command is being executed, $\overline{PAUSE}$ LOW Pulse Width is the time to complete execution plus the time shown. Status may be read at any time without exceeding the time shown.
6. $\overline{PAUSE}$ is pulled low for both command and data operations.
7. TEX is the execution time of the current command (see the Command Execution Times table).
8. $\overline{PAUSE}$ low pulse width is less than 50ns when writing into the data port or the control port as long as the duty requirement (TWI) is observed and no previous command is being executed. TWI may be safely violated up to 500ns as long as the extended TPPWW that results is observed. If a previously entered command is being executed, $\overline{PAUSE}$ LOW Pulse Width is the time to complete execution plus the time shown.
9. 150ns for the Am9511A-1DM.

SWITCHING WAVEFORMS

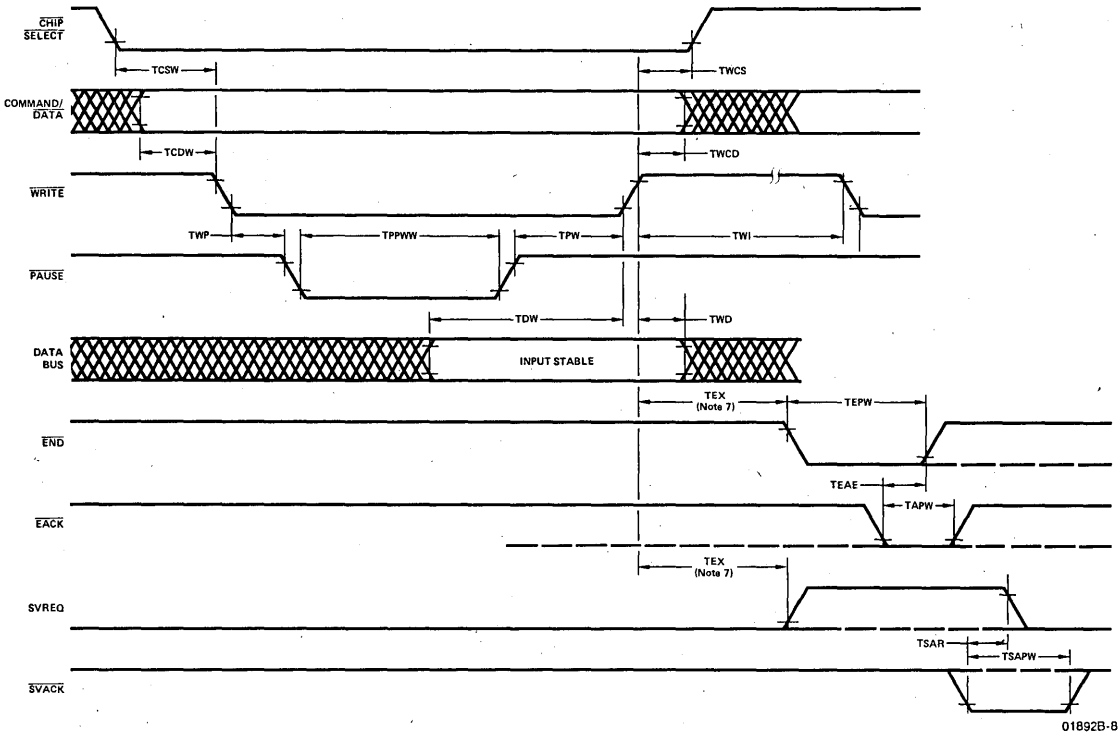
INPUT WAVEFORMS FOR AC TESTS



READ OPERATIONS



WRITE OPERATIONS



APPLICATION INFORMATION

The diagram in Figure 2 shows the interface connections for the Am9511A APU with operand transfers handled by an Am9517 DMA controller, and CPU coordination handled by an Am9519 Interrupt Controller. The APU interrupts the CPU to indicate that a command has been completed. When the performance enhancements provided by the DMA and Interrupt operations are not required, the APU interface can be simplified as shown in Figure 1. The Am9511A APU is designed with a general purpose 8-bit data bus and interface control so that it can be conveniently used with any general 8-bit processor.

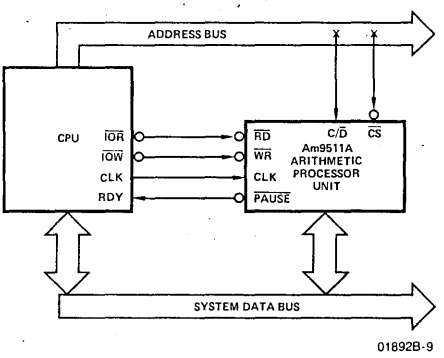


Figure 1. Am9511A Minimum Configuration Example.

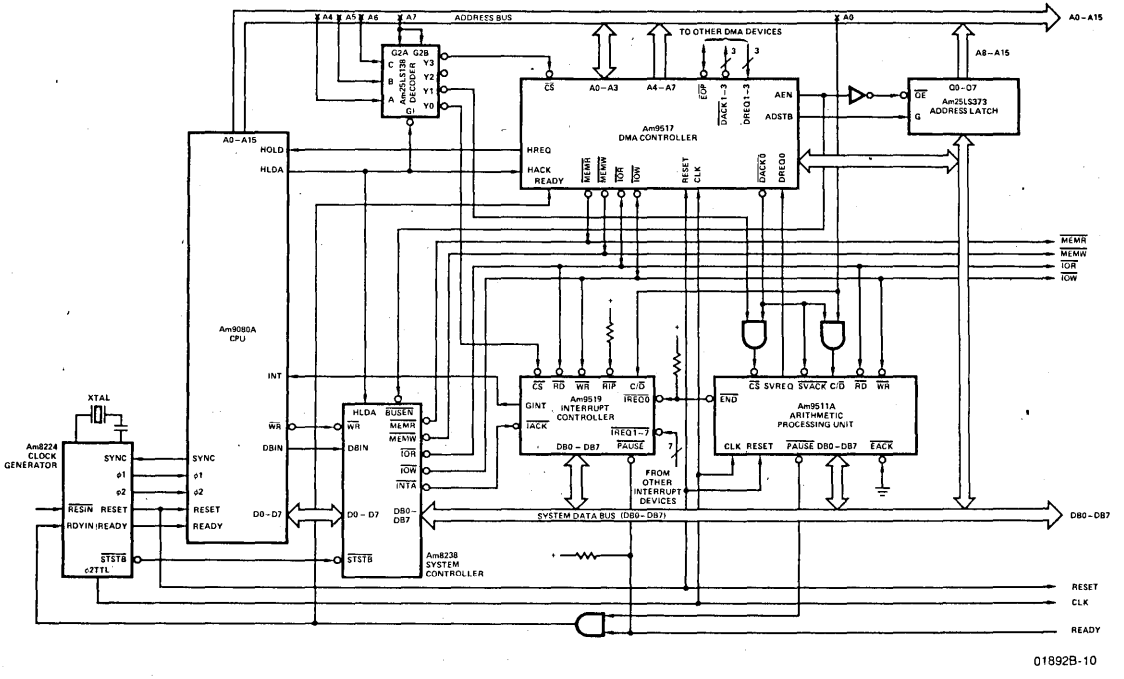


Figure 2. Am9511A High Performance Configuration Example.

Am9512

Floating Point Processor

DISTINCTIVE CHARACTERISTICS

- Single (32-bit) and double (64-bit) precision capability
- Add, subtract, multiply and divide functions
- Compatible with proposed IEEE format
- Easy interfacing to microprocessors
- 8-bit data bus
- Standard 24-pin package
- 12V and 5V power supplies
- Stack oriented operand storage
- Direct memory access or programmed I/O Data Transfers
- End of execution signal
- Error interrupt
- All inputs and outputs TTL level compatible
- Advanced N-channel silicon gate MOS technology

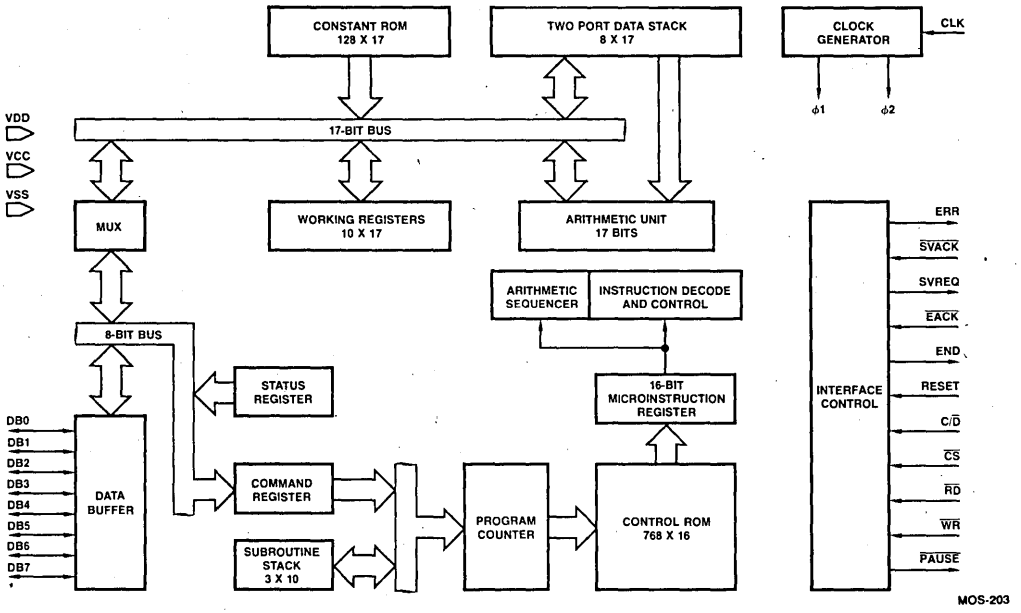
GENERAL DESCRIPTION

The Am9512 is a high performance floating-point processor unit (FPU). It provides single precision (32-bit) and double precision (64-bit) add, subtract, multiply and divide operations. It can be easily interfaced to enhance the computational capabilities of the host microprocessor.

The operand, result, status and command information transfers take place over an 8-bit bidirectional data bus. Operands are pushed onto an internal stack by the host processor and a command is issued to perform an operation on the data stack. The results of this operation are available to the host processor by popping the stack.

Information transfers between the Am9512 and the host processor can be handled by using programmed I/O or direct memory access techniques. After completing an operation, the Am9512 activates an "end of execution" signal that can be used to interrupt the host processor.

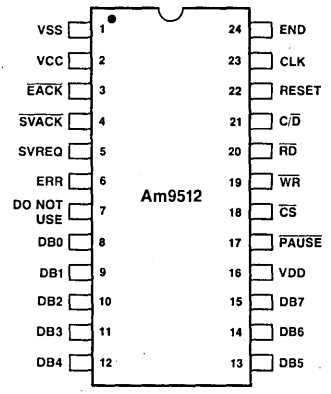
BLOCK DIAGRAM



ORDERING INFORMATION

Package	Ambient Temperature	V_{SS}	V_{CC}	V_{DD}	Maximum Clock Frequency	
					2MHz	3MHz
Hermetic DIP	$0°C \leq T_A \leq 70°C$	0V	+5.0V ±5%	+12V ±5%	Am9512DC	Am9512-1DC
	$-40°C \leq T_A \leq +85°C$	0V	+5.0V ±10%	+12V ±10%	Am9512DI	Am9512-1DI
	$-55°C \leq T_A \leq +125°C$	0V	+5.0V ±10%	+12V ±10%	Am9512DMB	Am9512-1DMB

CONNECTION DIAGRAM
Top View



Note: Pin 1 is marked for orientation. MOS-204

INTERFACE SIGNAL DESCRIPTION

VCC: +5V Power Supply

VDD: +12V Power Supply

VSS: Ground

CLK (Clock, Input)

An external timing source connected to the CLK input provides the necessary clocking.

RESET (Reset, Input)

A HIGH on this input causes initialization. Reset terminates any operation in progress, and clears the status register to zero. The internal stack pointer is initialized and the contents of the stack may be affected. After a reset the END output, the ERR output and the SVREQ output will be LOW. For proper initialization, RESET must be HIGH for at least five CLK periods following stable power supply voltages and stable clock.

C/$\overline{D}$ (Command/Data Select, Input)

The C/$\overline{D}$ input together with the $\overline{RD}$ and $\overline{WR}$ inputs determines the type of transfer to be performed on the data bus as follows:

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	Function
L	H	L	Push data byte into the stack
L	L	H	Pop data byte from the stack
H	H	L	Enter command
H	L	H	Read Status
X	L	L	Undefined

L = LOW
H = HIGH
X = DON'T CARE

END (End of Execution, Output)

A HIGH on this output indicates that execution of the current command is complete. This output will be cleared LOW by activating the $\overline{EACK}$ input LOW or performing any read or write operation or device initialization using the RESET. If $\overline{EACK}$ is tied LOW, the END output will be a pulse (see $\overline{EACK}$ description).

Reading the status register while a command execution is in progress is allowed. However any read or write operation clears the flip-flop that generates the END output. Thus such continuous reading could conflict with internal logic setting of the END flip-flop at the end of command execution.

$\overline{EACK}$ (End Acknowledge, Input)

This input when LOW makes the END output go LOW. As mentioned earlier HIGH on the END output signals completion of a command execution. The END signal is derived from an internal flip-flop which is clocked at the completion of a command. This flip-flop is clocked to the reset state when $\overline{EACK}$ is LOW. Consequently, if $\overline{EACK}$ is tied LOW, the END output will be a pulse that is approximately one CLK period wide.

SVREQ (Service Request, Output)

A HIGH on this output indicates completion of a command. In this sense this output is the same as the END output. However, the Service Bit in the Command Register determines whether the SVREQ output will go HIGH at the completion of a command. This bit must be 1 for SVREQ to go HIGH. The SVREQ can be cleared (i.e., go LOW) by activating the $\overline{SVACK}$ input LOW or initializing the device using the RESET. Also, the SVREQ will be automatically cleared after completion of any command that has the service request bit as 0.

$\overline{SVACK}$ (Service Acknowledge, Input)

A LOW on this input clears SVREQ. If the $\overline{SVACK}$ input is permanently tied LOW, it will conflict with the internal setting of the SVREQ output. Thus the SVREQ indication cannot be relied upon if the $\overline{SVACK}$ is tied LOW.

DB0-DB7 (Data Bus, Input/Output)

These eight bidirectional lines are used to transfer command, status and operand information between the device and the host processor. DB0 is the least significant and DB7 is the most significant bit position. HIGH on a data bus line corresponds to 1 and LOW corresponds to 0.

When pushing operands on the stack using the data bus, the least significant byte must be pushed first and most significant byte last. When popping the stack to read the result of an operation, the most significant byte will be available on the data bus first and the least significant byte will be the last. Moreover, for pushing operands and popping results, the number of transactions must be equal to the proper number of bytes appropriate for the chosen format. Otherwise, the internal byte pointer will not be aligned properly. The Am9512 single precision format requires 4 bytes and double precision format requires 8 bytes.

ERR (Error, Output)

This output goes HIGH to indicate that the current command execution resulted in an error condition. The error conditions are: attempt to divide by zero, exponent overflow and exponent underflow. The ERR output is cleared LOW on read status register operation or upon RESET.

The ERR output is derived from the error bits in the status register. These error bits will be updated internally at an appropriate time during a command execution. Thus ERR output going HIGH may not correspond with the completion of a command. Reading of the status register can be performed while a command execution is in progress. However it should be noted that reading the status register clears the ERR output. Thus reading the status register while a command execution in progress may result in an internal conflict with the ERR output.

$\overline{CS}$ (Chip Select, Input)

This input must be LOW to accomplish any read or write operation to the Am9512

To perform a write operation, appropriate data is presented on DB0 through DB7 lines, appropriate logic level on the C/$\overline{D}$ input and the $\overline{CS}$ input is made LOW. Whenever $\overline{WR}$ and $\overline{RD}$ inputs are both HIGH and $\overline{CS}$ is LOW, $\overline{PAUSE}$ goes LOW. However actual writing into the Am9512 cannot start until $\overline{WR}$ is made LOW. After initiating the write operation by the HIGH to LOW transition on the $\overline{WR}$ input, the $\overline{PAUSE}$ output will go HIGH indicating the write operation has been acknowledged. The $\overline{WR}$ input can go HIGH after $\overline{PAUSE}$ goes HIGH. The data lines, C/$\overline{D}$ input and the $\overline{CS}$ input can change when appropriate hold time requirements are satisfied. See write timing diagram for details.

To perform a read operation an appropriate logic level is established on the C/$\overline{D}$ input and $\overline{CS}$ is made LOW. The $\overline{PAUSE}$ output goes LOW because $\overline{WR}$ and $\overline{RD}$ inputs are HIGH. The read operation does not start until the $\overline{RD}$ input goes LOW. $\overline{PAUSE}$ will go HIGH indicating that read operation is complete and the required information is available on the DB0 through DB7 lines. This information will remain on the data lines as long as $\overline{RD}$ is LOW. The $\overline{RD}$ input can return HIGH anytime after $\overline{PAUSE}$ goes HIGH. The $\overline{CS}$ input and C/$\overline{D}$ input can change anytime after $\overline{RD}$ returns HIGH. See read timing diagram for details. If the $\overline{CS}$ is tied LOW permanently, $\overline{PAUSE}$ will remain LOW until the next Am9512 read or write access.

$\overline{RD}$ (Read, Input)

A LOW on this input is used to read information from an internal location and gate that information onto the data bus. The $\overline{CS}$ input must be LOW to accomplish the read operation. The C/$\overline{D}$ input determines what internal location is of interest. See C/$\overline{D}$, $\overline{CS}$ input descriptions and read timing diagram for details. If the END output was HIGH, performing any read operation will make the END output go LOW after the HIGH to LOW transition of the $\overline{RD}$ input (assuming $\overline{CS}$ is LOW). If the ERR output was HIGH performing a status register read operation will make the ERR output LOW. This will happen after the HIGH to LOW transition of the RD input (assuming $\overline{CS}$ is LOW).

$\overline{WR}$ (Write, Input)

A LOW on this input is used to transfer information from the data bus into an internal location. The $\overline{CS}$ must be LOW to accomplish the write operation. The C/$\overline{D}$ determines which internal location is to be written. See C/$\overline{D}$, $\overline{CS}$ input descriptions and write timing diagram for details.

If the END output was HIGH, performing any write operation will make the END output go LOW after the LOW to HIGH transition of the $\overline{WR}$ input (assuming $\overline{CS}$ is LOW).

$\overline{PAUSE}$ (Pause, Output)

This output is a handshake signal used while performing read or write transactions with the Am9512. If the $\overline{WR}$ and $\overline{RD}$ inputs are both HIGH, the $\overline{PAUSE}$ output goes LOW with the $\overline{CS}$ input in anticipation of a transaction. If $\overline{WR}$ goes LOW to initiate a write transaction with proper signals established on the DB0-DB7, C/$\overline{D}$ inputs, the $\overline{PAUSE}$ will return HIGH indicating that the write operation has been accomplished. The $\overline{WR}$ can be made HIGH after this event. On the other hand, if a read operation is desired, the $\overline{RD}$ input is made LOW after activating $\overline{CS}$ LOW and establishing proper C/$\overline{D}$ input. (The $\overline{PAUSE}$ will go LOW in response to $\overline{CS}$ going LOW.) The $\overline{PAUSE}$ will return HIGH indicating completion of read. The $\overline{RD}$ can return HIGH after this event. It should be noted that a read or write operation can be initiated without any regard to whether a command execution is in progress or not. Proper device operation is assured by obeying the $\overline{PAUSE}$ output indication as described.

FUNCTIONAL DESCRIPTION

Major functional units of the Am9512 are shown in the block diagram. The Am9512 employs a microprogram controlled stack oriented architecture with 17-bit wide data paths.

The Arithmetic Unit receives one of its operands from the Operand Stack. This stack is an eight word by 17-bit two port memory with last in – first out (LIFO) attributes. The second operand to the Arithmetic Unit is supplied by the internal 17-bit bus. In addition to supplying the second operand, this bidirectional bus also carries the results from the output of the Arithmetic Unit when required. Writing into the Operand Stack takes place from this internal 17-bit bus when required. Also connected to this bus are the Constant ROM and Working Registers. The ROM provides the required constants to perform the mathematical operations while the Working Registers provide storage for the intermediate values during command execution.

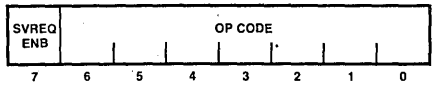
Communication between the external world and the Am9512 takes place on eight bidirectional input/output lines, DB0 through DB7 (Data Bus). These signals are gated to the internal 8-bit bus through appropriate interface and buffer circuitry. Multiplexing facilities exist for bidirectional communication between the internal eight and 17-bit buses. The Status Register and Command Register are also located on the 8-bit bus.

The Am9512 operations are controlled by the microprogram contained in the Control ROM. The Program Counter supplies the microprogram addresses and can be partially loaded from the Command Register. Associated with the Program Counter is the Subroutine Stack where return addresses are held during subroutine calls in the microprogram. The Microinstruction Register holds the current microinstruction being executed. The register facilitates pipelined microprogram execution. The Instruction Decode logic generates various internal control signals needed for the Am9512 operation.

The Interface Control logic receives several external inputs and provides handshake related outputs to facilitate interfacing the Am9512 to microprocessors.

COMMAND FORMAT

The Operation of the Am9512 is controlled from the host processor by issuing instructions called commands. The command format is shown below:



The command consists of 8 bits; the least significant 7 bits specify the operation to be performed as detailed in the accompanying table. The most significant bit is the Service Request Enable bit. This bit must be a 1 if SVREQ is to go high at end of executing a command.

The Am9512 commands fall into three categories: Single precision arithmetic, double precision arithmetic and data manipulation. There are four arithmetic operations that can be performed with single precision (32-bit), or double precision (64-bit) floating-point numbers: add, subtract, multiply and divide. These operations require two operands. The Am9512 assumes that these operands are located in the internal stack as Top of Stack

(TOS) and Next on Stack (NOS). The result will always be returned to the previous NOS which becomes the new TOS. Results from an operation are of the same precision and format as the operands. The results will be rounded to preserve the accuracy. The actual data formats and rounding procedures are described in a later section. In addition to the arithmetic operations, the Am9512 implements eight data manipulating operations. These include changing the sign of a double or single precision operand located in TOS, exchanging single precision operands located at TOS and NOS, as well as copying and popping single or double precision operands. See also the sections on status register and operand formats.

The Execution times of the Am9512 commands are all data dependent. Table 2 shows one example of each command execution time:

Table 1. Command Decoding Table.

Command Bits								Mnemonic	Description
7	6	5	4	3	2	1	0		
X	0	0	0	0	0	0	1	SADD	Add TOS to NOS Single Precision and result to NOS. Pop stack.
X	0	0	0	0	0	1	0	SSUB	Subtract TOS from NOS Single Precision and result to NOS. Pop stack.
X	0	0	0	0	0	1	1	SMUL	Multiply NOS by TOS Single Precision and result to NOS. Pop stack.
X	0	0	0	0	1	0	0	SDIV	Divide NOS by TOS Single Precision and result to NOS. Pop stack.
X	0	0	0	0	1	0	1	CHSS	Change sign of TOS Single Precision operand.
X	0	0	0	0	1	1	0	PTOS	Push Single Precision operand on TOS to NOS.
X	0	0	0	0	1	1	1	POPS	Pop Single Precision operand from TOS. NOS becomes TOS.
X	0	0	0	1	0	0	0	XCHS	Exchange TOS with NOS Single Precision.
X	0	1	0	1	1	0	1	CHSD	Change sign of TOS Double Precision operand.
X	0	1	0	1	1	1	0	PTOD	Push Double Precision operand on TOS to NOS.
X	0	1	0	1	1	1	1	POPD	Pop Double Precision operand from TOS. NOS becomes TOS.
X	0	0	0	0	0	0	0	CLR	CLR status.
X	0	1	0	1	0	0	1	DADD	Add TOS to NOS Double Precision and result to NOS. Pop stack.
X	0	1	0	1	0	1	0	DSUB	Subtract TOS from NOS Double Precision and result to NOS. Pop stack.
X	0	1	0	1	0	1	1	DMUL	Multiply NOS by TOS Double Precision and result to NOS. Pop stack.
X	0	1	0	1	1	0	0	DDIV	Divide NOS by TOS Double Precision and result to NOS. Pop Stack.

Notes: X = Don't Care Operation for bit combinations not listed above is undefined.

Table 2. Am9512 Execution Time in Cycles.

	Single Precision			Double Precision		
	Min	Typ	Max	Min	Typ	Max
Add	58	220	512	578	1200	3100
Subtract	56	220	512	578	1200	3100
Multiply	192	220	254	1720	1770	1860
Divide	228	240	264	4560	4920	5120

Note: Typical for add and subtract, assumes the operands are within six decimal orders of magnitude. Max is derived from the maximum execution time of 1000 executions with random 32-bit or 64-bit patterns.

Table 3. Some Execution Examples.

Command	TOS	NOS	Result	Clock periods
SADD	3F800000	3F800000	40000000	58
SSUB	3F800000	3F800000	00000000	56
SMUL	40400000	3FC00000	40900000	198
SDIV	40000000	3F800000	3F000000	228
CHSS	3F800000	–	BF800000	10
PTOS	3F800000	–	–	16
POPS	3F800000	–	–	14
XCHS	3F800000	4000000	–	26
CHSD	3FF0000000000000	–	BFF0000000000000	24
PTOD	3FF0000000000000	–	–	40
POPD	3FF0000000000000	–	–	26
CLR	3FF0000000000000	–	–	4
DADD	3FF00000A0000000	8000000000000000	3FF00000A0000000	578
DSUB	3FF00000A0000000	8000000000000000	3FF00000A0000000	578
DMUL	BFF8000000000000	3FF8000000000000	C002000000000000	1748
DDIV	BFF8000000000000	3FF8000000000000	BFF0000000000000	4560

Note: TOS, NOS and Result are in hexadecimal; Clock period is in decimal.

COMMAND INITIATION

After properly positioning the required operands in the stack, a command may be issued. The procedure for initiating a command execution is as follows:

1. Establish appropriate command on the DB0-DB7 lines.
2. Establish HIGH on the C/$\overline{D}$ input.
3. Establish LOW on the $\overline{CS}$ input. Whenever $\overline{WR}$ and $\overline{RD}$ inputs are HIGH the $\overline{PAUSE}$ output follows the $\overline{CS}$ input. Hence $\overline{PAUSE}$ will become LOW.
4. Establish LOW on the $\overline{WR}$ input after an appropriate set up time (see timing diagrams).
5. Sometime after the HIGH to LOW level transition of $\overline{WR}$ input, the $\overline{PAUSE}$ output will become HIGH to acknowledge the write operation. The $\overline{WR}$ input can return to HIGH anytime after $\overline{PAUSE}$ goes HIGH. The DB0-DB7, C/$\overline{D}$ and $\overline{CS}$ inputs are allowed to change after the hold time requirements are satisfied (see timing diagram).

An attempt to issue a new command while the current command execution is in progress is allowed. Under these circumstances, the $\overline{PAUSE}$ output will not go HIGH until the current command execution is completed.

OPERAND ENTRY

The Am9512 commands operate on the operands located at the TOS and NOS and results are returned to the stack at NOS and then popped to TOS. The operands required for the Am9512 are one of two formats – single precision floating-point (4 bytes) or double precision floating-point (8 bytes). The result of an operation has the same format as the operands. In other words, operations using single precision quantities always result in a single precision result while operations involving double precision quantities will result in double precision result.

Operands are always entered into the stack least significant byte first and most significant byte last. The following procedure must be followed to enter operands into the stack:

1. The lower significant operand byte is established on the DB0-DB7 lines.
2. A LOW is established on the C/$\overline{D}$ input to specify that data is to be entered into the stack.
3. The $\overline{CS}$ input is made LOW. Whenever the $\overline{WR}$ and $\overline{RD}$ inputs are HIGH, the $\overline{PAUSE}$ output will follow the $\overline{CS}$ input. Thus $\overline{PAUSE}$ output will become LOW.
4. After appropriate set up time (see timing diagrams), the $\overline{WR}$ input is made LOW.
5. Sometime after this event, $\overline{PAUSE}$ will return HIGH to indicate that the write operation has been acknowledged.
6. Anytime after the $\overline{PAUSE}$ output goes HIGH the $\overline{WR}$ input can be made HIGH. The DB0-DB7, C/$\overline{D}$ and $\overline{CS}$ inputs can change after appropriate hold time requirements are satisfied (see timing diagrams).

The above procedure must be repeated until all bytes of the operand are pushed into the stack. It should be noted that for single precision operands 4 bytes should be pushed and 8 bytes must be pushed for double precision. Not pushing all the bytes of a quantity will result in byte pointer misalignment.

The Am9512 stack can accommodate 4 single precision quantities or 2 double precision quantities. Pushing more quantities than the capacity of the stack will result in loss of data which is usual with any LIFO stack.

REMOVING THE RESULTS

Result from an operation will be available at the TOS. Results can be transferred from the stack to the data bus by reading the stack. When the stack is popped for results, the most significant byte is available first and the least significant byte last. A result is always of the same precision as the operands that produced it. Thus when the result is taken from the stack, the total number of bytes popped out should be appropriate with the precision – single precision results are 4 bytes and double precision results are 8 bytes. The following prodedure must be used for reading the result from the stack:

1. A LOW is established on the C/$\overline{D}$ input.
2. The $\overline{CS}$ input is made LOW. When $\overline{WR}$ and $\overline{RD}$ inputs are both HIGH, the $\overline{PAUSE}$ output follows the $\overline{CS}$ input, thus $\overline{PAUSE}$ will be LOW.
3. After appropriate set up time (see timing diagrams), the $\overline{RD}$ input is made LOW.
4. Sometime after this, $\overline{PAUSE}$ will return HIGH indicating that the data is available on the DB0-DB7 lines. This data will remain on the DB0-DB7 lines as long as the $\overline{RD}$ input remains LOW.
5. Anytime after $\overline{PAUSE}$ goes HIGH, the $\overline{RD}$ input can return HIGH to complete transaction.
6. The $\overline{CS}$ and C/$\overline{D}$ inputs can change after appropriate hold time requirements are satisfied (see timing diagram).
7. Repeat this procedure until all bytes appropriate for the precision of the result are popped out.

Reading of the stack does not alter its data; it only adjusts the byte pointer. If more data is popped than the capacity of the stack, the internal byte pointer will wrap around and older data will be read again, consistent with the LIFO stack.

READING STATUS REGISTER

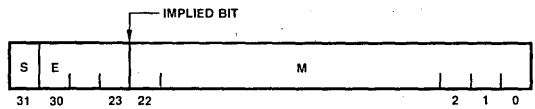
The Am9512 status register can be read without any regard to whether a command is in progress or not. The only implication that has to be considered is the effect this might have on the END and ERR outputs discussed in the signal descriptions.

The following procedure must be followed to accomplish status register reading.

1. Establish HIGH on the C/$\overline{D}$ input.
2. Establish LOW on the $\overline{CS}$ input. Whenever $\overline{WR}$ and $\overline{RD}$ inputs are HIGH, $\overline{PAUSE}$ will follow the $\overline{CS}$ input. Thus, $\overline{PAUSE}$ will go LOW.
3. After appropriate set up time (see timing diagram) $\overline{RD}$ is made LOW.
4. Sometime after the HIGH to LOW transition of $\overline{RD}$, $\overline{PAUSE}$ will become HIGH indicating that status register contents are available on the DB0-DB7 lines. These lines will contain this information as long as $\overline{RD}$ is LOW.
5. The $\overline{RD}$ input can be returned HIGH anytime after $\overline{PAUSE}$ goes HIGH.
6. The C/$\overline{D}$ input and $\overline{CS}$ input can change after satisfying appropriate hold time requirements (see timing diagram).

DATA FORMATS

The Am9512 handles floating-point quantities in two different formats – single precision and double precision. The single precision quantities are 32-bits long as shown below.



Bit 31:

S = Sign of the mantissa. 1 represents negative and 0 represents positive.

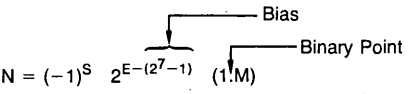
Bits 23-30

E = These 8-bits represent a biased exponent. The bias is $2^7 - 1 = 127$

Bits 0-22

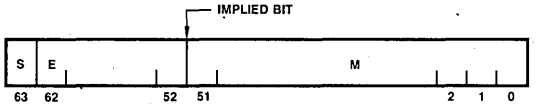
M = 23-bit mantissa. Together with the sign bit, the mantissa represents a signed fraction in sign-magitude notation. There is an implied 1 beyond the most significant bit (bit 22) of the mantissa. In other words, the mantissa is assumed to be a 24-bit normalized quantity and the most significant bit which will always be 1 due to normalization is implied. The Am9512 restores this implied bit internally before performing arithmetic; normalizes the result and strips the implied bit before returning the results to the external data bus. The binary point is between the implied bit and bit 22 of the mantissa.

The quantity N represented by the above notation is



Provided E ≠ 0 or all 1's.

A double precision quantity consists of the mantissa sign bit(s), an 11 bit biased exponent (E), and a 52-bit mantissa (M). The bias for double precision quantities is $2^{10} - 1$. The double precision format is illustrated below.



Bit 63:

S = Sign of the mantissa. 1 represents negative and 0 represents positive.

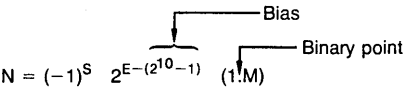
Bits 52-62

E = These 11 bits represent a biased exponent. The bias is $2^{10} - 1 = 1023$.

Bit 0-51

M = 52-bit mantissa. Together with the sign bit, the mantissa represents a signed fraction in sign-magnitude notation. There is an implied 1 beyond the most significant bit (bit 51) of the mantissa. In other words, the mantissa is assumed to a 53-bit normalized quantity and the most significant bit, which will always be a 1 due to normalization, is implied. The Am9512 restores this implied bit internally before performing arithmetic; normalizes the result and strips the implied bit before returning the result to the external data bus. The binary point is between the implied bit and bit 51 of the mantissa.

The quantity N represented by the above notation is



Provided E ≠ 0 or all 1's.

STATUS REGISTER

The Am9512 contains an 8-bit status register with the following format.

BUSY	SIGN S	ZERO Z	RESERVED	DIVIDE EXCEPTION D	EXPONENT UNDERFLOW U	EXPONENT OVERFLOW V	RESERVED
7	6	5	4	3	2	1	0

Bit 0 and bit 4 are reserved. Occurrence of exponent oerflow (V), exponent underflow (U) and divide exception (D) are indicated by bits 1, 2 and 3 respectively. An attempt to divide by zero is the only divide exception. Bits 5 and 6 represent a zero result and the sign of a result respectively. Bit 7 (Busy) of the status register indicates if the Am9512 is currently busy executing a command. All the bits are initialized to zero upon reset. Also, executing a CLR (Clear Status) command will result in all zero status register bits. A zero in Bit 7 indicates that the Am9512 is not busy and a new command may be initiated. As soon as a new command is issued, Bit 7 becomes 1 to indicate the device is busy and remains 1 until the command execution is complete, at which time it will become 0. As soon as a new command is issued, status register bits 0, 1, 2, 3, 4, 5 and 6 are cleared to zero. The status bits will be set as required during the command execution. Hence, as long as bit 7 is 1, the remainder of the status register bit indications should not be relied upon unless the ERR occurs. The following is a detailed status bit description.

Bit 0 Reserved

Bit 1 Exponent overflow (V): When 1, this bit indicates that exponent overflow has occurred. Cleared to zero otherwise.

Bit 2 Exponent Underflow (U): When 1, this bit indicates that exponent underflow has occurred. Cleared to zero otherwise.

Bit 3 Divide Exception (D): When 1, this bit indicates that an attempt to divide by zero is made. Cleared to zero otherwise.

Bit 4 Reserved

Bit 5 Zero (Z): When 1, this bit indicates that the result returned to TOS after a command is all zeros. Cleared to zero otherwise.

Bit 6 Sign (S): When 1, this bit indicates that the result returned to TOS is negative. Cleared to zero otherwise.

Bit 7 Busy: When 1, this bit indicates the Am9512 is in the process of executing a command. It will become zero after the command execution is complete.

All other status register bits are valid when the Busy bit is zero.

ALGORITHMS OF FLOATING-POINT ARITHMETIC

1. Floating Point to Decimal Conversion

   As an introduction to floating-point arithmetic, a brief description of the Decimal equivalent of the Am9512 floating-point format should help the reader to understand and verify the validity of the arithmetic operations. The Am9512 single precision format is used for the following discussions. With a minor modification of the field lengths, the discussion would also apply to the double precision format.

   There are three parts in a floating point number:

   a. The sign – the sign applies to the sign of the number. Zero means the number is positive or zero. One means the number is negative.

b. The exponent – the exponent represents the magnitude of the number. The Am9512 single precision format has an excess 127_{10} notation which means the code representation is 127_{10} higher than the actual value. The following are a few examples of actual versus coded exponent.

Actual	Coded
$+127_{10}$	$+254_{10}$
0	127_{10}
-126_{10}	$+1_{10}$

c. The mantissa – the mantissa is a 23-bit value with the binary point to the left of the most significant bit. There is a hidden 1 to the left of the binary point so the mantissa is always less than 2 and greater than or equal to 1.

To find the Decimal equivalent of the floating point number, the mantissa is multiplied by 2 to the power of the actual exponent. The number is negated if the sign bit = 1. The following are two examples of conversion:

Example 1

Floating Point No. = 0 10000011 11000000000000000000000B
Sign — Exponent — Mantissa

Coded Exponent = 10000011B
Actual Exponent = 10000011B – 01111111B = 00000100B = 4_{10}
Mantissa = 1.11000000000000000000000B
= 1 + 1/2 + 1/4 = 1.75_{10}
Decimal No. = 2^4 x 1.75 = 16 x 1.75 = 28_{10}

Example 2

Floating Point No. = 1 01111010 01100000000000000000000B
Sign — Exponent — Mantissa

Code Exponent = 01111010B
Actual Exponent = 01111010B – 01111111B = 11111011B = -5_{10}
Mantissa = 1.01100000000000000000000B
= 1 + 1/4 + 1/8 = 1.375_{10}
Decimal No. = -2^{-5} x 1.375 = $-.04296875_{10}$

2. Unpacking of the Floating-Point Numbers
The Am9512 unpacks the floating point number into three parts before any of the arithmetic operation. The number is divided into three parts as described in Section 1. The sign and exponent are copied from the original number as 1 and 8-bit numbers respectively. The mantissa is stored as a 24-bit number. The least significant 23 bits are copied from the original number and the MSB is set to 1. The binary point is assumed to the right of the MSB.

The abbreviations listed below are used in the following sections of algorithm description:

SIGN – Sign of Result
EXP – Exponent of Result
MAN – Mantissa of Result
SIGN (TOS) – Sign of Top of Stack
EXP (TOS) – Exponent of Top of Stack
MAN (TOS) – Mantissa of Top of Stack
SIGN (NOS) – Sign of Next on Stack
EXP (NOS) – Exponent of Next on Stack
MAN (NOS) – Mantissa of Next on Stack

3. Floating-Point Add/Subtract
The floating-point add and subtract essentially use the same algorithm. The only difference is that floating-point subtract changes the sign of the floating-point number at top of stack and then performs the floating-point add.

The following is a step by step description of a floating-point add algorithm (Figure 1):

a. Unpack TOS and NOS.
b. The exponent of TOS is compared to the exponent of NOS.
c. If the exponents are equal, go to step f.
d. Right shift the mantissa of the number with the smaller exponent.
e. Increment the smaller exponent and go to step b.
f. Set sign of result to sign of larger number.
g. Set exponent of result to exponent of larger number.
h. If sign of the two numbers are not equal, go to m.
i. Add Mantissas.
j. Right shift resultant mantissa by 1 and increment exponent of result by 1.
k. If MSB of exponent changes from 1 to 0 as a result of the increment, set overflow status.
l. Round if necessary and exit.
m. Subtract smaller mantissa from larger mantissa.
n. Left shift mantissa and decrement exponent of result.
o. If MSB of exponent changes from 0 to 1 as a result of the decrement, set underflow status and exit.
p. If the MSB of the resultant mantissa = 0, go to n.
q. Round if necessary and exit.

4. Floating-Point Multiply
Floating-point multiply basically involves the addition of the exponents and multiplication of the mantissas. The following is a step by step description of a floating multiplication algorithm (Figure 2):

a. Check if TOS or NOS = 0.
b. If either TOS or NOS = 0, Set result to 0 and exit.
c. Unpack TOS and NOS.
d. Convert EXP (TOS) and EXP (NOS) to unbiased form.
EXP (TOS) = EXP (TOS) -127_{10}
EXP (NOS) = EXP (NOS) -127_{10}
e. Add exponents.
EXP = EXP (TOS) + EXP (NOS)
f. If MSB of EXP (TOS) = MSB of EXP (NOS) = 0 and MSB of EXP = 1, then set overflow status and exit.
g. If MSB of EXP (TOS) = MSB of EXP (NOS) = 1 and MSB of EXP = 0, then set underflow status and exit.
h. Convert Exponent back to biased form.
EXP = EXP + 127_{10}
i. If sign of TOS = sign of NOS, set sign of result to 0, else set sign of result to 1.
j. Multiply mantissa.
k. If MSB of resultant = 1, right shift mantissa by 1 and increment exponent of resultant.
l. If MSB of exponent changes from 1 to 0 as a result of the increment, set overflow status.
m. Round if necessary and exit.

5. Floating-Point Divide
The floating-point divide basically involves the subtraction of exponents and the division of mantissas. The following is a step by step description of a division algorithm (Figure 3).

a. If TOS = 0, set divide exception error and exit.
b. If NOS = 0, set result to 0 and exit.
c. Unpack TOS and NOS.
d. Convert EXP (TOS) and EXP (NOS) to unbiased form.
EXP (TOS) = EXP (TOS) – 127_{10}
EXP (NOS) = EXP (NOS) – 127_{10}
e. Subtract exponent of TOS from exponent of NOS.
EXP = EXP (NOS) – EXP (TOS)
f. If MSB of EXP (NOS) = 0, MSB of EXP (TOS) = 1 and MSB of EXP = 1, then set overflow status and exit.
g. If MSB of EXP (NOS) = 1, MSB of EXP (TOS) = 0, and MSB of EXP = 0, then set underflow status and exit.

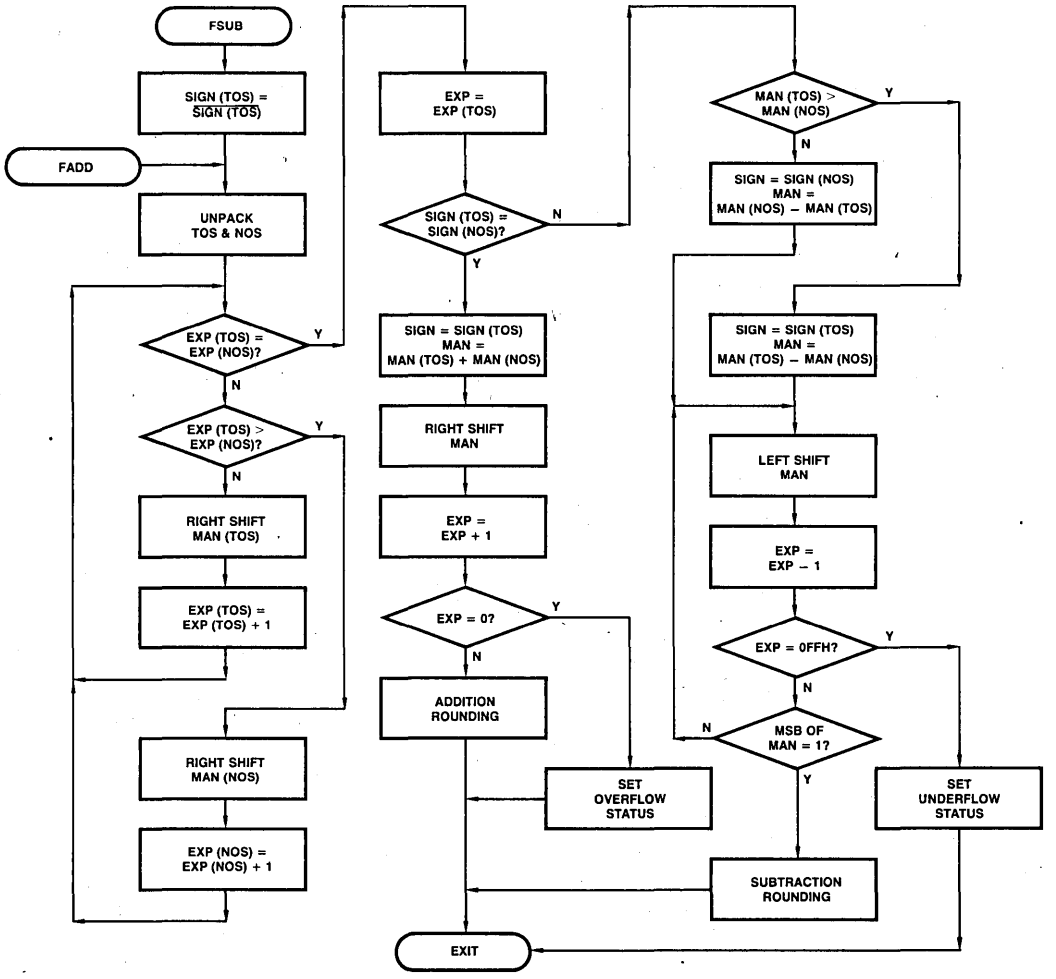


Figure 1. Conceptual Floating-Point Addition/Subtraction.

MOS-205

h. Add bias to exponent of result.
   $EXP = EXP + 127_{10}$
i. If sign of TOS = sign of NOS, set sign of result to 0, else set sign of result to 1.
j. Divide mantissa of NOS by mantissa of TOS.
k. If MSB = 0, left shift mantissa and decrement exponent of resultant, else go to n.
l. If MSB of exponent changes from 0 to 1 as a result of the decrement, set underflow status.
m. Go to k.
n. Round if necessary and exit.

The algorithms described above provide the user a means of verifying the validity of the result. They do not necessarily reflect the exact internal sequence of the Am9512.

6. Rounding

The Am9512 adopts a rounding algorithm that is consistent with the Intel® standard for floating-point arithmetic. The following description is an excerpt from the paper published in proceedings of Compsac 77, November 1977, pp. 107-112 by Dr. John F. Palmer of Intel Corporation.

The method used for doing the rounding during floating-point arithmetic is known as "Round to Even", i.e., if the resultant number is exactly halfway between two floating point numbers, the number is rounded to the nearest floating-point number whose LSB of the mantissa is 0. In order to simplify the explanation, the algorithms will be illustrated with 4-bit arithmetic. The existence of an accumulator will be assumed as shown:

OF	B1	B2	B3	B4	G	R	ST

The bit labels denote:

OF – The overflow bit
B1-B4 – The 4 mantissa bits
G – The Guard bit
R – The Rounding bit
ST – The "Sticky" bit

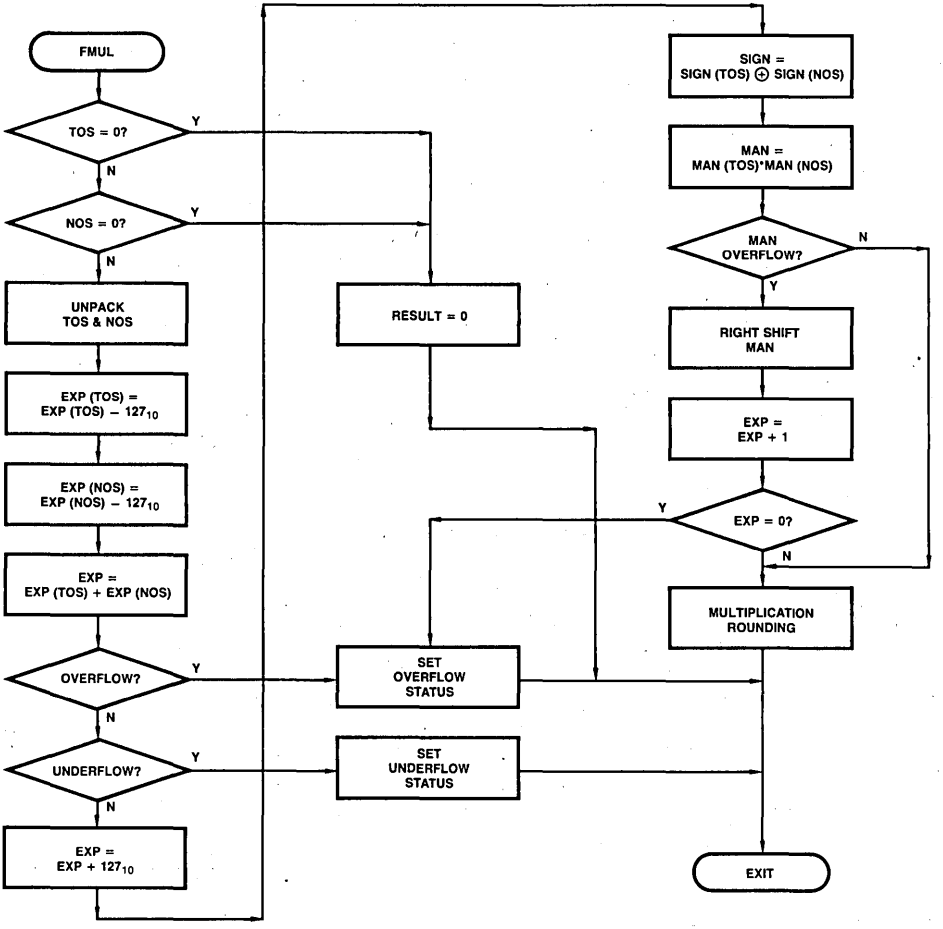


Figure 2. Conceptual Floating-Point Multiplication.

MOS-206

The Sticky bit is set to one if any ones are shifted right of the rounding bit in the process of denormalization. If the Sticky bit becomes set, it remains set throughout the operation. All shifting in the Accumulator involves the OF, G, R and ST bits. The ST bit is not affected by left shifts but, zeros are introduced into OF by right shifts.

Rounding during addition of magnitudes – add 1 to the G position, then if G=R=ST=0, set B4 to 0 ("Rounding to Even").

Rounding during subtraction of magnitudes – if more than one left shift was performed, no rounding is needed, otherwise round the same way as addition of magnitudes.

Rounding during multiplication – let the normalized double length product be:

B1	B2	B3	B4	B5	B6	B7	B8

Then G=B5, R=B6, ST=B7 V B8. The rounding is then performed as in addition of magnitudes.

Rounding during division – let the first six bits of the normalized quotient be

B1	B2	B3	B4	B5	B6

Then G=B5, R=B6, ST=0 if and only if remainder = 0. The rounding is then performed as in addition of magnitudes.

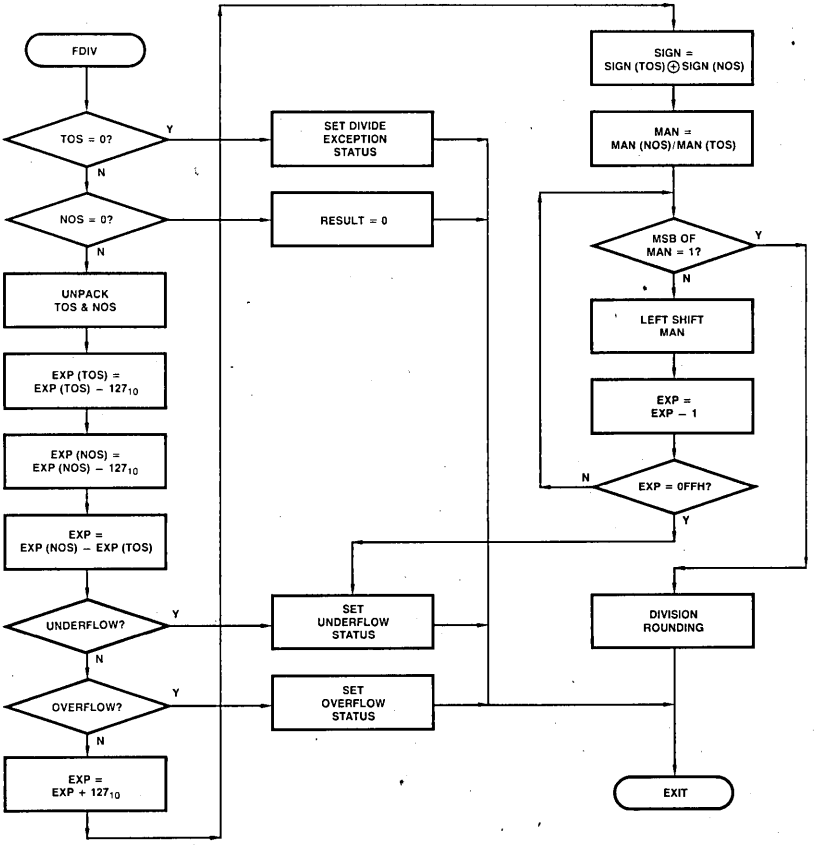


Figure 3. Conceptual Floating-Point Division.

MOS-207

CHSD

CHANGE SIGN DOUBLE PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	1	0	1	1	0	1

Hex Coding: AD IF SRE = 1
2D IF SRE = 0

Execution Time: See Table 2

Description:

The sign of the double precision TOS operand A is complemented. The double precision result R is returned to TOS. If the double precision operand A is zero, then the sign is not affected. The status bit S and Z indicate the sign of the result and if the result is zero. The status bits U, V and D are always cleared to zero.

Status Affected: S, Z. (U, V, D always zero.)

STACK CONTENTS

BEFORE		AFTER
A	TOS	R
B	NOS	B

CHSS

CHANGE SIGN SINGLE PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	1	0	1

Hex Coding: 85 IF SRE = 1
05 IF SRE = 0

Execution Time: See Table 2

Description:

The sign of the single precision operand A at TOS is complemented. The single precision result R is returned to TOS. If the exponent field of A is zero, all bits of R will be zeros. The status bits S and Z indicate the sign of the result and if the result is zero. The status bits U, V and D are cleared to zero.

Status Affected: S, Z. (U, V, D always zero.)

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B	← NOS →	B
C		C
D		D

CLR

CLEAR STATUS

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	0	0	0

Hex Coding: 80 IF SRE = 1
00 IF SRE = 0

Execution Time: 4 clock cycles

Description:
The status bits S, Z, D, U, V are cleared to zero. The stack is not affected. This essentially is a no operation command as far as operands are concerned.

Status Affected: S, Z, D, U, V always zero.

DSUB

DOUBLE PRECISION FLOATING-POINT SUBTRACT

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	1	0	1	0	1	0

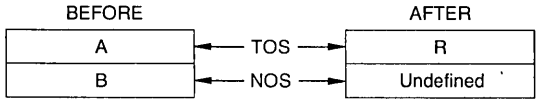
Hex Coding: AA IF SRE = 1
2A IF SRE = 0

Execution Time: See Table 2

Description:
The double precision operand A at TOS is subtracted from the double precision operand B at NOS. The result is rounded to obtain the final double precision result R which is returned to TOS. The status bits S, Z, U and V are affected to report sign of the result, if the result is zero, exponent underflow and exponent overflow respectively. The status bit D will be cleared to zero.

Status Affected: S, Z, U, V. (D always zero.)

STACK CONTENTS



DADD

DOUBLE PRECISION FLOATING-POINT ADD

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	1	0	1	0	0	1

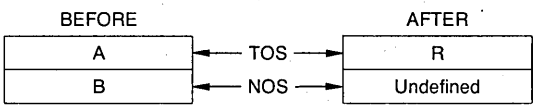
Hex Coding: A9 IF SRE = 1
29 IF SRE = 0

Execution Time: See Table 2

Description:
The double precision operand A from TOS is added to the double precision operand B from NOS. The result is rounded to obtain the final double precision result R which is returned to TOS. The status bits S, Z, U and V are affected to report sign of the result, if the result is zero, exponent underflow and exponent overflow respectively. The status bit D will be cleared to zero.

Status Affected: S, Z, U, V. (D always zero.)

STACK CONTENTS



DMUL

DOUBLE PRECISION FLOATING-POINT MULTIPLY

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	1	0	1	0	1	1

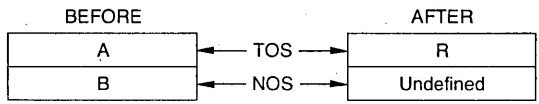
Hex Coding: AB IF SRE = 1
2B IF SRE = 0

Execution Time: See Table 2

Description:
The double precision operand A from TOS is multiplied by the double precision operand B from NOS. The result is rounded to obtain the final double precision result R which is returned to TOS. The status bits S, Z, U and V are affected to report sign of the result, if the result is zero, exponent underflow and exponent overflow respectively. The status bit D will be cleared to zero.

Status Affected: S, Z, U, V. (D always zero.)

STACK CONTENTS



DDIV

DOUBLE PRECISION FLOATING-POINT DIVIDE

	7	6	5	4	3	2	1	0
Binary Code:	SRE	0	1	0	1	1	0	0

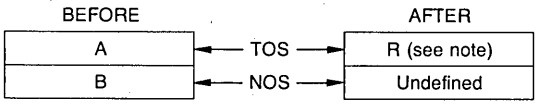
Hex Coding: AC IF SRE = 1
2C IF SRE = 0

Execution Time: See Table 2

Description:
The double precision operand B from NOS is divided by the double precision operand A from TOS. The result (quotient) is rounded to obtain the final double precision result R which is returned to TOS. The status bits, S, Z, D, U and V are affected to report sign of the result, if the result is zero, attempt to divide by zero, exponent underflow and exponent overflow respectively.

Status Affected: S, Z, D, U, V

STACK CONTENT



Note: If A is zero, then R = B (Divide exception).

SADD

SINGLE PRECISION FLOATING-POINT ADD

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	0	0	1

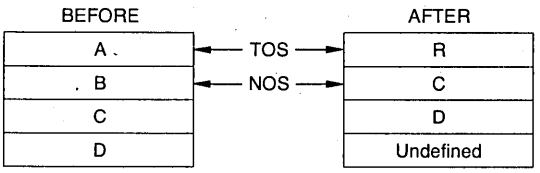
Hex Coding: 81 IF SRE = 1
01 IF SRE = 0

Execution Time: See Table 2

Description:
The single precision operand A from TOS is added to the single precision operand B from NOS. The result is rounded to obtain the final single precision result R which is returned to TOS. The status bits S, Z, U and V are affected to report the sign of the result, if the result is zero, exponent underflow and exponent overflow respectively. The status bit D will be cleared to zero.

Status Affected: S, Z, U, V. (D always zero.)

STACK CONTENT



SSUB

SINGLE PRECISION FLOATING-POINT SUBTRACT

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	0	1	0

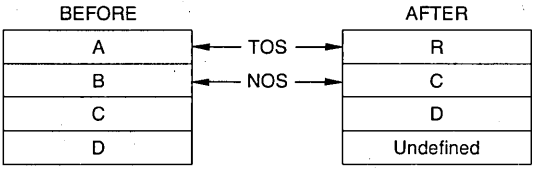
Hex Coding: 82 IF SRE = 1
02 IF SRE = 0

Execution Time: See Table 2

Description:
The single precision operand A at TOS is subtracted from the single precision operand B at NOS. The result is rounded to obtain the final single precision result R which is returned to TOS. The status bits S, Z, U and V are affected to report the sign of the result, if the result is zero, exponent underflow and exponent overflow respectively. The status bit D will be cleared to zero.

Status Affected: S, Z, U, V. (D always zero.)

STACK CONTENTS



SMUL

SINGLE PRECISION FLOATING-POINT MULTIPLY

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	0	1	1

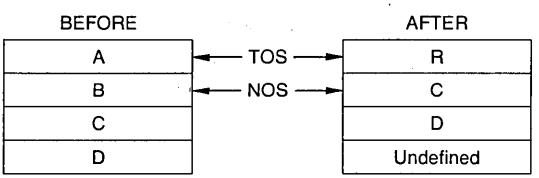
Hex Coding: 83 IF SRE = 1
03 IF SRE = 0

Execution Time: See Table 2

Description:
The single precision operand A from TOS is multiplied by the single precision operand B from NOS. The result is rounded to obtain the final single precision result R which is returned to TOS. The status bits S, Z, U and V are affected to report the sign of the result, if the result is zero, exponent underflow and exponent overflow respectively. The status bit D will be cleared to zero.

Status Affected: S, Z, U, V. (D always zero.)

STACK CONTENTS



SDIV

SINGLE PRECISION FLOATING-POINT DIVIDE

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	1	0	0

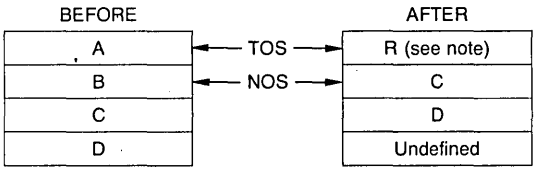
Hex Coding: 84 IF SRE = 1
04 IF SRE = 0

Execution Time: See Table 2

Description:
The single precision operand B from NOS is divided by the single precision operand A from TOS. The result (quotient) is rounded to obtain the final result R which is returned to TOS. The status bits S, Z, D, U and V are affected to report the sign of the result, if the result is zero, attempt to divide by zero, exponent underflow and exponent overflow respectively.

Status Affected: S, Z, D, U, V

STACK CONTENTS



Note: If exponent field of A is zero then R = B (Divide exception).

POPS

POP STACK SINGLE PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	1	1	1

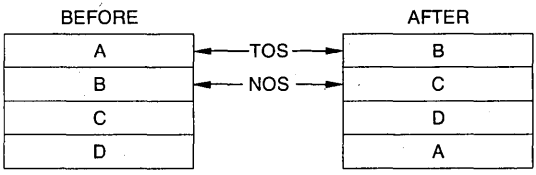
Hex Coding: 87 IF SRE = 1
07 IF SRE = 0

Execution Time: See Table 2

Description:
The single precision operand A is popped from the stack. The internal stack control mechanism is such that A will be written at the bottom of the stack. The status bits S and Z are affected to report the sign of the new operand at TOS and if it is zero, respectively. The status bits U, V and D will be cleared to zero. Note that only the exponent field of the new TOS is checked for zero, if it is zero status bit Z will set to 1.

Status Affected: S, Z. (U, V, D always zero.)

STACK CONTENTS



PTOD

PUSH STACK DOUBLE PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	1	0	1	1	1	0

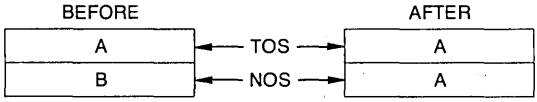
Hex Coding: AE IF SRE = 1
2E IF SRE = 0

Execution Time: See Table 2

Description:
The double precision operand A from the TOS is pushed back on to the stack. This is effectively a duplication of A into two consecutive stack locations. The status S and Z are affected to report sign of the new TOS and if the new TOS is zero respectively. The status bits U, V and D will be cleared to zero.

Status Affected: S, Z. (U, V, D always zero.)

STACK CONTENTS



PTOS

PUSH STACK SINGLE PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	0	1	1	0

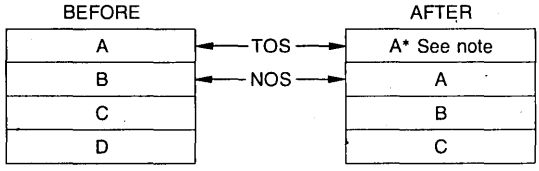
Hex Coding: 86 IF SRE = 1
06 IF SRE = 0

Execution Time: See Table 2

Description:
This instruction effectively pushes the single precision operand from TOS on to the stack. This amounts to duplicating the operand at two locations in the stack. However, if the operand at TOS prior to the PTOS command has only its exponent field as zero, the new content of the TOS will all be zeroes. The contents of NOS will be an exact copy of the old TOS. The status bits S and Z are affected to report the sign of the new TOS and if the content of TOS is zero, respectively. The status bits U, V and D will be cleared to zero.

Status Affected: S, Z. (U, V, D always zero.)

STACK CONTENTS



Note: A* = A if Exponent field of A is not zero.
A* = 0 if Exponent field of A is zero.

POPD

POP STACK DOUBLE PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	1	0	1	1	1	1

Hex Coding: AF IF SRE = 1
2F IF SRE = 0

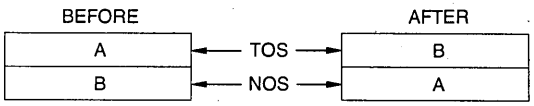
Execution Time: See Table 2

Description:

The double precision operand A is popped from the stack. The internal stack control mechanism is such that A will be written at the bottom of the stack. This operation has the same effect as exchanging TOS and NOS. The status bits S and Z are affected to report the sign of the new operand at TOS and if it is zero, respectively. The status bits U, V and D will be cleared to zero.

Status Affected: S, Z (U, V and D always zero.)

STACK CONTENTS



XCHS

EXCHANGE TOS AND NOS SINGLE-PRECISION

	7	6	5	4	3	2	1	0
Binary Coding:	SRE	0	0	0	1	0	0	0

Hex Coding: 88 IF SRE = 1
08 IF SRE = 0

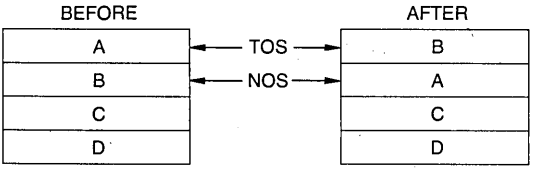
Execution Time: See Table 2

Description:

The single precision operand A at the TOS and the single precision operand B at the NOS are exchanged. After execution, B is at the TOS and A is at the NOS. All other operands are unchanged.

Status Affected: S, Z (U, V and D always zero.)

STACK CONTENTS



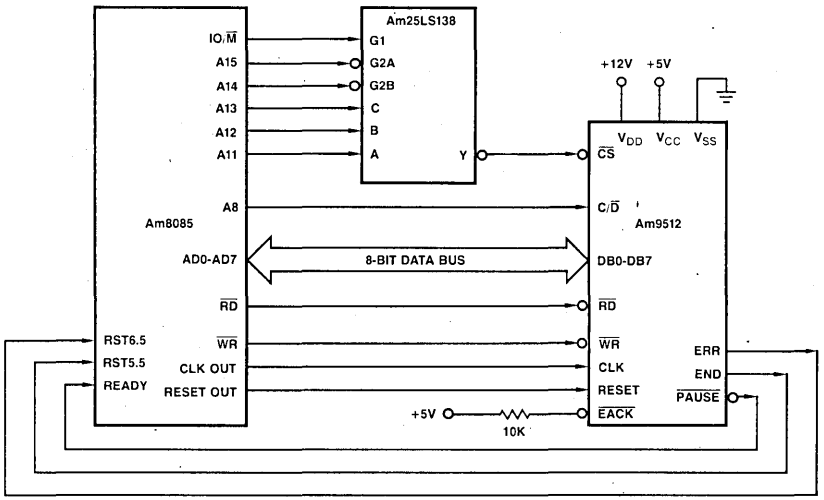


Figure 4. Am9512 to Am8085 Interface.

MOS-213

MAXIMUM RATINGS beyond which useful life may be impaired

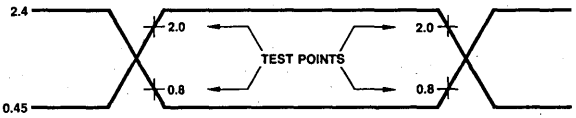
Storage Temperature	−65 to +150°C
V_{DD} with Respect to V_{SS}	−0.5 to +15.0V
V_{CC} with Respect to V_{SS}	−0.5 to +7.0V
All Signal Voltages with Respect to V_{SS}	−0.5 to +7.0V
Power Dissipation (Package Limitation)	2.0W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

ELECTRICAL CHARACTERISTICS Over Operating Range (Note 1)

Parameters	Description	Test Conditions	Min.	Typ.	Max.	Units
VOH	Output HIGH Voltage	IOH = −200μA	3.7			Volts
VOL	Output LOW Voltage	IOL = 3.2mA			0.4	Volts
VIH	Input HIGH Voltage		2.0		VCC	Volts
VIL	Input LOW Voltage		−0.5		0.8	Volts
IIX	Input Load Current	VSS ≤ VI ≤ VCC			±10	μA
IOZ	Data Bus Leakage	VO = 0.4V			10	μA
		VO = VCC			10	
ICC	VCC Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
IDD	VDD Supply Current	T_A = +25°C		50	90	mA
		T_A = 0°C			95	
		T_A = −55°C			100	
CO	Output Capacitance	fc = 1.0MHz, Inputs = 0V		8	10	pF
CI	Input Capacitance			5	8	pF
CIO	I/O Capacitance			10	12	pF

INPUT AND OUTPUT WAVEFORMS FOR AC TESTS



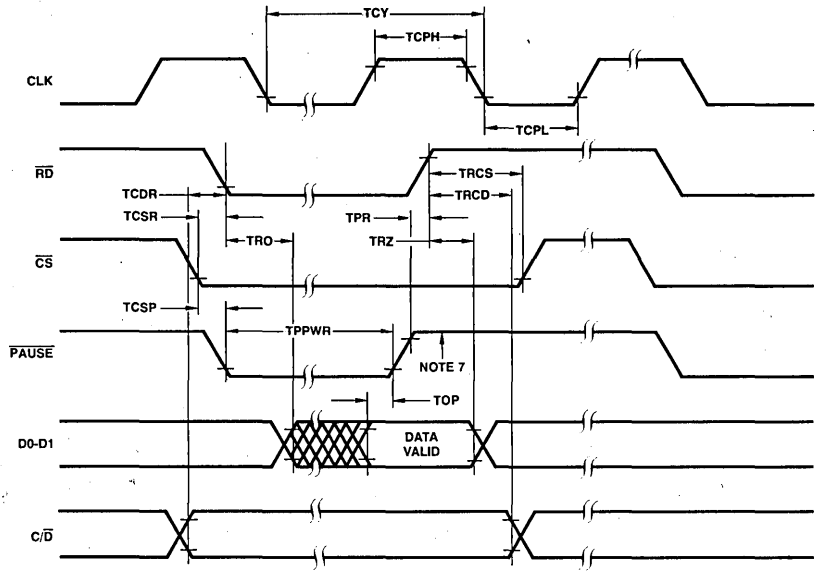
SWITCHING CHARACTERISTICS

Parameters	Description		Am9512DC Min	Am9512DC Max	Am9512-1DC Min	Am9512-1DC Max	Units
TAPW	$\overline{EACK}$ LOW Pulse Width		100		75		ns
TCDR	C/$\overline{D}$ to $\overline{RD}$ LOW Set-up Time		0		0		ns
TCDW	C/$\overline{D}$ to $\overline{WR}$ LOW Set-up Time		0		0		ns
TCPH	Clock Pulse HIGH Width		200	500	140	500	ns
TCPL	Clock Pulse LOW Width		240		160		ns
TCSP	$\overline{CS}$ LOW to $\overline{PAUSE}$ LOW Delay (Note 5)		150		100		ns
TCSR	$\overline{CS}$ to $\overline{RD}$ LOW Set-up Time		0		0		ns
TCSW	$\overline{CS}$ LOW to $\overline{WR}$ LOW Set-up Time		0		0		ns
TCY	Clock Period		480	5000	320	2000	ns
TDW	Data Valid to $\overline{WR}$ HIGH Delay		150		100		ns
TEAE	$\overline{EACK}$ LOW to END LOW Delay			200		175	ns
TEHPHR	END HIGH to $\overline{PAUSE}$ HIGH Data Read when Busy			5.5TCY+300		5.5TCY+200	ns
TEHPHW	END HIGH to $\overline{PAUSE}$ HIGH Write when Busy			200		175	ns
TEPW	END HIGH Pulse Width		400		300		ns
TEX	Execution Time		See Table 2				ns
TOP	Data Bus Output Valid to $\overline{PAUSE}$ HIGH Delay		0		0		ns
TPPWR	$\overline{PAUSE}$ LOW Pulse Width Read	Data	3.5TCY+50	5.5TCY+300	3.5TCY+50	5.5TCY+200	ns
		Status	1.5TCY+50	3.5TCY+300	1.5TCY+50	3.5TCY+200	
TPPWRB	END HIGH to $\overline{PAUSE}$ HIGH Read when Busy	Data	See Table 2				ns
		Status	1.5TCY+50	3.5TCY+300	1.5TCY+50	3.5TCY+200	
TPPWW	$\overline{PAUSE}$ LOW Pulse Width Write when Not Busy			TCSW+50		TCSW+50	ns
TPPWWB	$\overline{PAUSE}$ LOW Pulse Width Write when Busy		See Table 2				ns
TPR	$\overline{PAUSE}$ HIGH to Read HIGH Hold Time		0		0		ns
TPW	$\overline{PAUSE}$ HIGH to Write HIGH Hold Time		0		0		ns
TRCD	$\overline{RD}$ HIGH to C/$\overline{D}$ Hold Time		0		0		ns
TRCS	$\overline{RD}$ HIGH to $\overline{CS}$ HIGH Hold Time		0		0		ns
TRO	$\overline{RD}$ LOW to Data Bus On Delay		50		50		ns
TRZ	$\overline{RD}$ HIGH to Data Bus Off Delay		50	200	50	150	ns
TSAPW	SVACK LOW Pulse Width		100		75		ns
TSAR	SVACK LOW to SVREQ LOW Delay			300		200	ns
TWCD	$\overline{WR}$ HIGH to C/D Hold Time		60		30		ns
TWCS	$\overline{WR}$ HIGH to CS HIGH Hold Time		60		30		ns
TWD	$\overline{WR}$ HIGH to Data Bus Hold Time		20		20		ns

NOTES:
1. Typical values are for T_A = 25°C, nominal supply voltages and nominal processing parameters.
2. Switching parameters are listed in alphabetical order.
3. Test conditions assume transition times of 20ns or less, output loading of one TTL gate plus 100pF and timing reference levels of 0.8V and 2.0V.
4. END HIGH pulse width is specified for $\overline{EACK}$ tied to VSS. Otherwise TEAE applies.
5. $\overline{PAUSE}$ is pulled low for both command and data operations.
6. TEX is the execution time of the current command (see the Command Execution Times table).
7. $\overline{PAUSE}$ will go low at this point if $\overline{CS}$ is low and $\overline{RD}$ and $\overline{WR}$ are high.

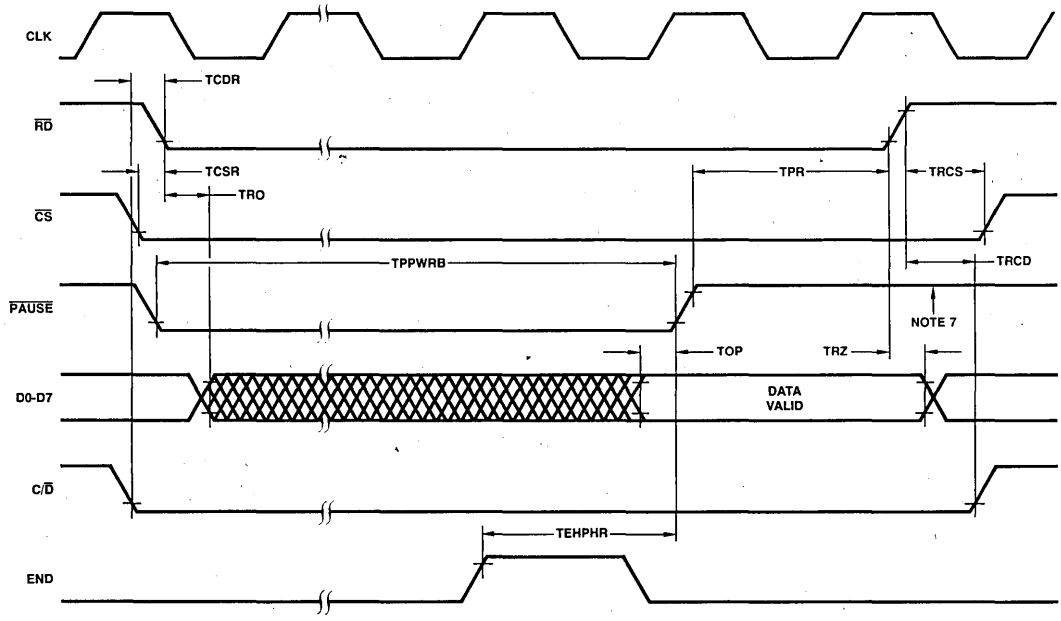
TIMING DIAGRAMS

READ OPERATION



MOS-208

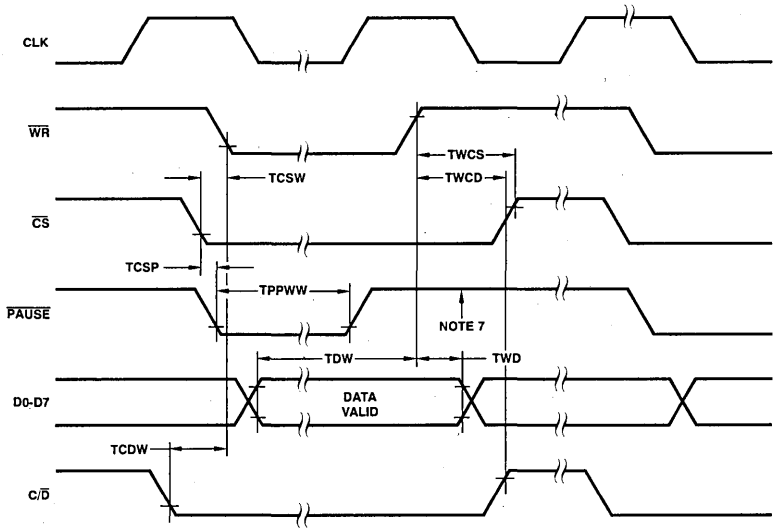
OPERAND READ WHEN Am9512 IS BUSY



MOS-209

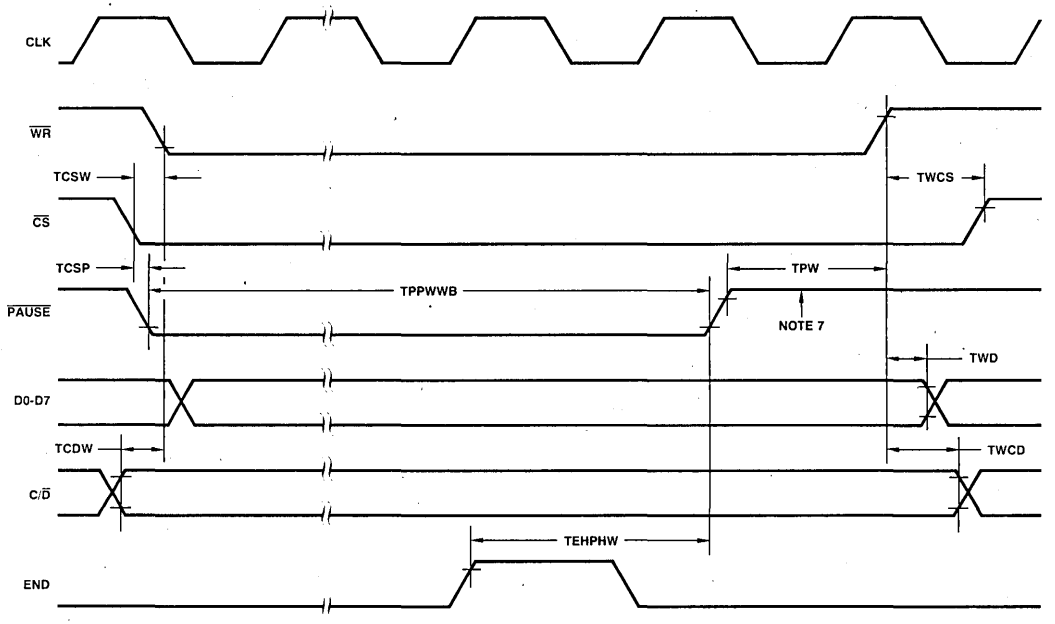
TIMING DIAGRAMS (Cont.)

OPERAND ENTRY



MOS-210

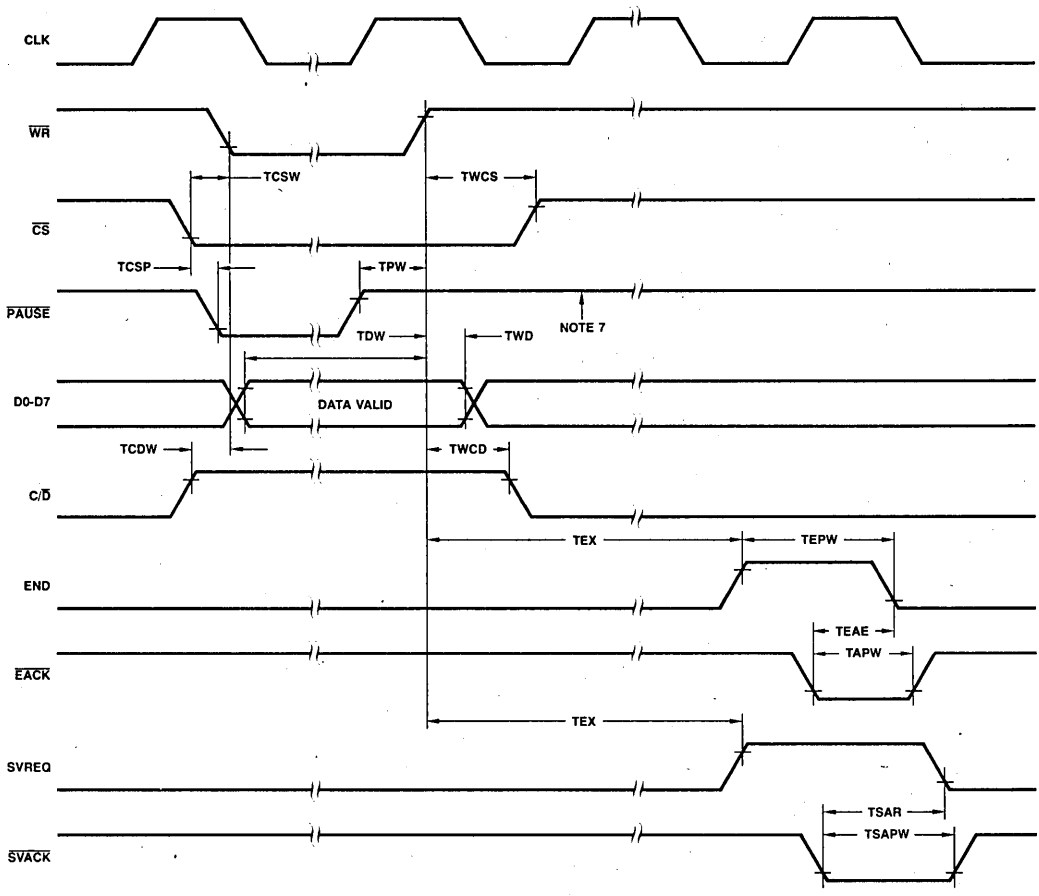
COMMAND OR DATA WRITE WHEN Am9512 IS BUSY



MOS-211

TIMING DIAGRAMS (Cont.)

COMMAND INITIATION



MOS-212

Am9513 • AmZ8073

System Timing Controller

DISTINCTIVE CHARACTERISTICS

- Five independent 16-bit counters
- High speed counting rates
- Up/down and binary/BCD counting
- Internal oscillator frequency source
- Tapped frequency scaler
- Programmable frequency output
- 8-bit or 16-bit bus interface
- Time-of-day option
- Alarm comparators on counters 1 and 2
- Complex duty cycle outputs
- One-shot or continuous outputs
- Programmable count/gate source selection
- Programmable input and output polarities
- Programmable gating functions
- Retriggering capability
- +5 volt power supply
- Standard 40-pin package

GENERAL DESCRIPTION

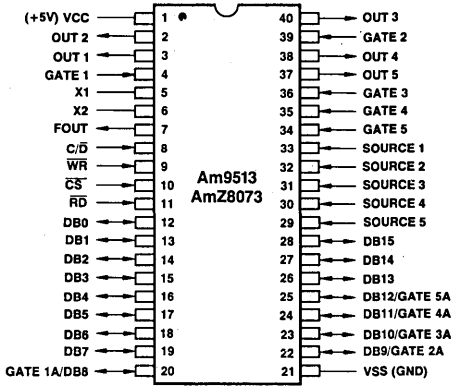
The Am9513 System Timing Controller is an LSI circuit designed to service many types of counting, sequencing and timing applications. It provides the capability for programmable frequency synthesis, high resolution programmable duty cycle waveforms, retriggerable digital one-shots, time-of-day clocking, coincidence alarms, complex pulse generation, high resolution baud rate generation, frequency shift keying, stop-watching timing, event count accumulation, waveform analysis and many more. A variety of programmable operating modes and control features allow the Am9513 to be personalized for particular applications as well as dynamically reconfigured under program control.

The STC includes five general-purpose 16-bit counters. A variety of internal frequency sources and external pins may be selected as inputs for individual counters with software selectable active-high or active-low input polarity. Both hardware and software gating of each counter is available. Three-state outputs for each counter provide pulses or levels and can be active-high or active-low. The counters can be programmed to count up or down in either binary or BCD. The host processor may read an accumulated count at any time without disturbing the counting process. Any of the counters may be internally concatenated to form any effective counter length up to 80 bits.

The AmZ8073 is functionally equivalent to the Am9513 with timing enhancements which allow it to be fully speed compatible with the AmZ8001 and AmZ8002 microprocessors.

CONNECTION DIAGRAM

D-40
P-40



Pin 1 is marked for orientation.

Figure 1. MMC-088

ORDERING INFORMATION

Package Type	Temperature Range	Counting Frequency 7MHz
Molded	$0°C \leq T_A \leq +70°C$	AM9513PC
Hermetic		AM9513DC
Hermetic	$-40°C \leq T_A \leq +85°C$	AM9513DI
Hermetic	$-55°C \leq T_A \leq +125°C$	AM9513DMB
Molded	$0°C \leq T_A \leq +70°C$	AMZ8073PC
Hermetic		AMZ8073DC
Hermetic	$-40°C \leq T_A \leq +85°C$	AMZ8073DI

See the "Am9513 System Timing Controller Handbook" – 1983 edition (03402B) for detailed technical information, application examples and software considerations.

Z8000 is a trademark of Zilog, Inc.

01731C-MMP

Am9513 MAXIMUM RATINGS beyond which useful life may be impaired

Storage Temperature	−65°C to +150°C
Ambient Temperature Under Bias	−55°C to +125°C
VCC with Respect to VSS	−0.5V to +7.0V
All Signal Voltages with Respect to VSS	−0.5V to +7.0V
Power Dissipation (Package Limitation)	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	Temperature	VCC	VSS
Am9513DC/PC	$0°C \leq T_A \leq +70°C$	+5V ±5%	0V
Am9513DI	$-40°C \leq T_A \leq +85°C$	+5V ±5%	0V
Am9513DM	$-55°C \leq T_A \leq +125°C$	+5V ±5%	0V
AmZ8073DC/PC	$0°C \leq T_A \leq +70°C$	+5V ±5%	0V
AmZ8073DI	$-40°C \leq T_A \leq +85°C$	+5V ±5%	0V

ELECTRICAL CHARACTERISTICS over operating range (Notes 1 and 2)

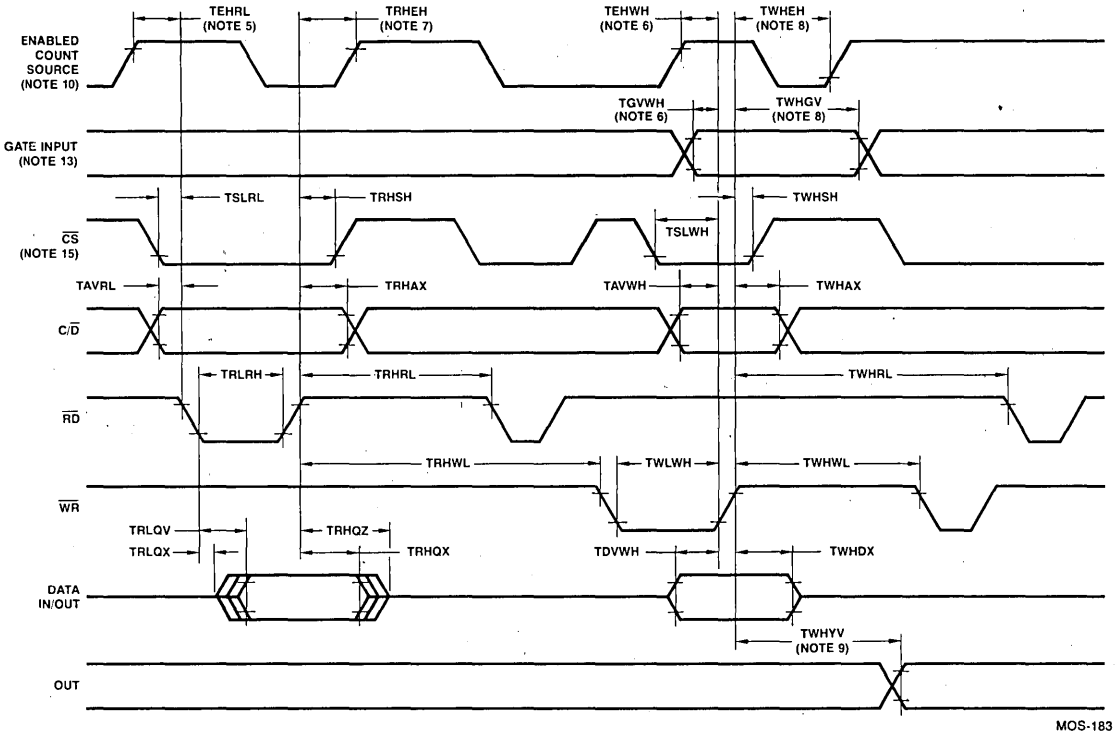
Parameter	Description		Test Conditions	Min	Typ	Max	Units
VIL	Input Low Voltage	All Inputs Except X2		VSS−0.5		0.8	Volts
		X2 Input		VSS−0.5		0.8	
VIH	Input High Voltage	All Inputs Except X2		2.2V		VCC	Volts
		X2 Input		3.8		VCC	
VITH	Input Hysteresis (SRC and GATE Inputs Only)			0.2	0.3		Volts
VOL	Output Low Voltage		IOL = 3.2mA			0.4	Volts
VOH	Output High Voltage		IOH = −200μA	2.4			Volts
IIX	Input Load Current (Except X2)		VSS ≤ VIN ≤ VCC			±10	μA
IOZ	Output Leakage Current (Except X1)		VSS + 0.4 ≤ VOUT ≤ VCC High Impedance State			±25	μA
ICC	VCC Supply Current (Steady State)		T_A = −55°C			275	mA
			T_A = 0°C			255	
			T_A = +25°C		190	235	
CIN	Input Capacitance		f = 1MHz, T_A = +25°C, All pins not under test at 0V.			10	pF
COUT	Output Capacitance					15	
CIO	IN/OUT Capacitance					20	

INPUT WAVEFORMS FOR AC TESTS

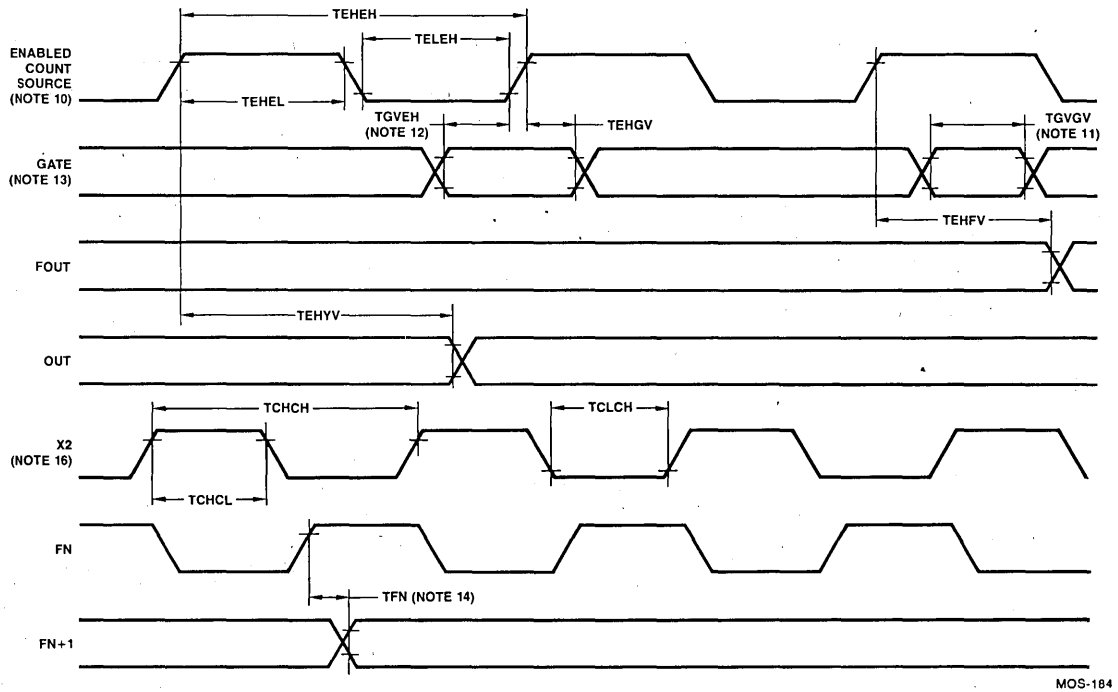
SWITCHING CHARACTERISTICS over operating range (Notes 2, 3, 4)

Parameter	Description		Figure	Am9513 Min	Am9513 Max	AmZ8073 Min	AmZ8073 Max	Units
TAVRL	C/$\overline{D}$ Valid to Read Low		23	25		25		ns
TAVWH	C/$\overline{D}$ Valid to Write High		23	170		170		ns
TCHCH	X2 High to X2 High (X2 Period)		24	145		145		ns
TCHCL	X2 High to X2 Low (X2 High Pulse Width)		24	70		70		ns
TCLCH	X2 Low to X2 High (X2 Low Pulse Width)		24	70		70		ns
TDVWH	Data In Valid to Write High		23	80		80		ns
TEHEH	Count Source High to Count Source High (Source Cycle Time) (Note 10)		24	145		145		ns
TEHEL TELEH	Count Source Pulse Duration (Note 10)		24	70		70		ns
TEHFV	Count Source High to FOUT Valid (Note 10)		24		500		500	ns
TEHGV	Count Source High to Gate Valid (Level Gating Hold Time) (Notes 10, 12, 13)		24	10		10		ns
TEHRL	Count Source High to Read Low (Set-up Time) (Notes 5, 10)		23	190		190		ns
TEHWH	Count Source High to Write High (Set-up Time) (Notes 6, 10)		23	−100		−100		ns
TEHYV	Count Source High to Out Valid (Note 10)	TC Output	24		300		300	ns
		Immediate or Delayed Toggle Output	24		300		300	
		Comparator Output	24		350		350	
TFN	FN High to FN+1 Valid (Note 14)		24		75		75	ns
TGVEH	Gate Valid to Count Source High (Level Gating Set-up Time) (Notes 10, 12, 13)		24	100		100		ns
TGVGV	Gate Valid to Gate Valid (Gate Pulse Duration) (Notes 11, 13)		24	145		145		ns
TGVWH	Gate Valid to Write High (Notes 6, 13)		23	−100		−100		ns
TRHAX	Read High to C/$\overline{D}$ Don't Care		23	0		0		ns
TRHEH	Read High to Count Source High (Notes 7, 10)		23	0		0		ns
TRHQX	Read High to Data Out Invalid		23	10		10		ns
TRHQZ	Read High to Data Out at High Impedance (Data Bus Release Time)		23		85		85	ns
TRHRL	Read High to Read Low (Read Recovery Time)		23	1000		1000		ns
TRHSH	Read High to $\overline{CS}$ High (Note 15)		23	0		0		ns
TRHWL	Read High to Write Low (Read Recovery Time)		23	1000		1000		ns
TRLQV	Read Low to Data Out Valid		23		110		110	ns
TRLQX	Read Low to Data Bus Driven (Data Bus Drive Time)		23	20		20		ns
TRLRH	Read Low to Read High (Read Pulse Duration) (Note 15)		23	160		160		ns
TSLRL	$\overline{CS}$ Low to Read Low (Note 15)		23	20		20		ns
TSLWH	$\overline{CS}$ Low to Write High (Note 15)		23	170		170		ns
TWHAX	Write High to C/$\overline{D}$ Don't Care		23	20		20		ns
TWHDX	Write High to Data In Don't Care		23	20		20		ns
TWHEH	Write High to Count Source High (Notes 8, 10, 17)		23	475		475		ns
TWHGV	Write High to Gate Valid (Notes 8, 13, 17)		23	500		500		ns
TWHRL	Write High to Read Low (Write Recovery Time)		23	1500		1000		ns
TWHSH	Write High to $\overline{CS}$ High (Note 15)		23	20		20		ns
TWHWL	Write High to Write Low (Write Recovery Time)		23	1500		1000		ns
TWHYV	Write High to Out Valid (Note 9, 17)		23		650		650	ns
TWLWH	Write Low to Write High (Write Pulse Duration) (Note 15)		23	150		150		ns

Bus Transfer Switching Waveforms



Counter Switching Waveforms



NOTES:

1. Typical values are for T_A = 25°C, nominal supply voltage and nominal processing parameters.
2. Test conditions assume transition times of 10ns or less, timing reference levels of 0.8V and 2.0V and output loading of one TTL gate plus 100pF, unless otherwise noted.
3. Abbreviations used for the switching parameter symbols are given as the letter T followed by four or five characters. The first and third characters represent the signal names on which the measurements start and end. Signal abbreviations used are:

   A (Address) = C/$\overline{D}$
   C (Clock) = X2
   D (Data In) = DB0-DB15
   E (Enabled counter source input) = SRC1-SRC5, GATE1-GATE5, F1-F5, TCN−1
   F = FOUT
   G (Counter gate input) = GATE1-GATE5, TCN−1
   Q (Data Out) = DB0-DB15
   R (Read) = $\overline{RD}$
   S (Chip Select) = $\overline{CS}$
   W (Write) = $\overline{WR}$
   Y (Output) = OUT1-OUT5

   The second and fourth letters designate the reference states of the signals named in the first and third letters respectively, using the following abbreviations.

   H = High
   L = Low
   V = Valid
   X = unknown or don't care
   Z = high impedance

4. Switching parameters are listed in alphabetical order.
5. Any input transition that occurs before this minimum setup requirement will be reflected in the contents read from the status register.
6. Any input transition that occurs before this minimum setup requirement will act on the counter before the execution of the operation initiated by the write. Failure to meet this setup time when issuing commands to the counter may result in incorrect counter operation.
7. Any input transition that occurs after this minimum hold time is guaranteed to not influence the contents read from the status register on the current read operation.
8. Any input transition that occurs after this minimum hold time is guaranteed to be seen by the counter as occurring after the action initiated by the write operation. Failure to meet this hold time when issuing commands to the counter may result in incorrect counter operation.
9. This parameter applies to cases where the write operation causes a change in the output bit.
10. The enabled count source is one of F1-F5, TCN−1, SRC1-SRC5 or GATE1-GATE5, as selected in the applicable Counter Mode register. The timing diagram assumes the counter counts on rising source edges. The timing specifications are the same for falling-edge counting.
11. This parameter applies to edge gating (CM15-CM13 = 110 or 111) and gating when both CM7 = 1 and CM15-CM13 ≠ 000. This parameter represents the minimum GATE pulse width needed to ensure that the pulse initiates counting or counter reloading.
12. This parameter applies to both edge and level gating (CM15-CM13 = 001 through 111) and gating when both CM7 = 1 and CM15-CM13 = 000. This parameter represents the minimum setup or hold times to ensure that the Gate input is seen at the intended level on the active source edge. Failure to meet the required setup and hold times may result in incorrect counter operation.
13. This parameter assumes that the GATENA input is unused (16-bit bus mode) or is tied high. In cases where the GATENA input is used, this timing specification must be met by both the GATE and GATENA inputs.
14. Signals F1-F5 cannot be directly monitored by the user. The phase difference between these signals will manifest itself by causing counters using two different F signals to count at different times on nominally simultaneous transitions in the F signals.
15. This timing specification assumes that $\overline{CS}$ is active whenever $\overline{RD}$ or $\overline{WR}$ are active. $\overline{CS}$ may be held active indefinitely.
16. This parameter assumes X2 is driven from an external gate with a square wave.
17. This parameter assumes that the write operation is to the command register.

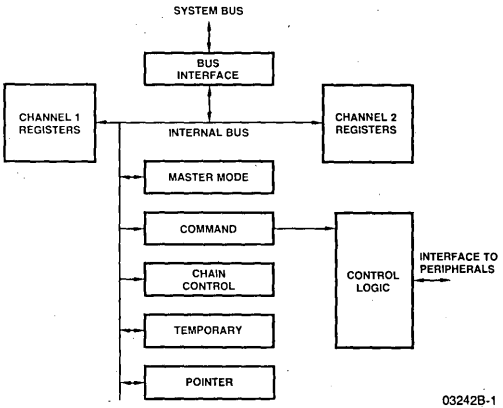
Am9516

Universal DMA Controller (UDC)

DISTINCTIVE CHARACTERISTICS

- Two independent multi-function channels
- Transfer Modes: Single, demand dedicated with bus hold, demand dedicated with bus release, demand interleave
- Data types: Byte-to-Byte, Word-to-Word, Byte/Word funnelling
- 16 Megabyte physical addressing range in each address space
- Address increment, decrement or hold
- Automatic loading/reloading of control parameters by each channel
- Optional automatic chaining of operations
- Channel interleave operations
- Interleave operations with system bus
- Base registers for efficient repetitive operations
- Reload word table for efficient channel initialization
- Masked data pattern matching for search operations
- Vectored interrupts on selected transfer conditions
- Software DMA request
- Software or hardware wait state insertion
- Memory/peripheral transfer up to 2.66 Megabyte/second at 4MHz and 4 Megabyte/second at 6MHz
- Memory/memory transfer up to 1.33 Megabyte/second at 4MHz and 2 Megabyte/second at 6MHz

Figure 1. Block Diagram



GENERAL DESCRIPTION

The Am9516 Universal DMA Controller (UDC) is a high performance peripheral interface circuit for 8086 and 68000 CPUs. In addition to providing data block transfer capability between memory and peripherals, each of the UDC's two channels can perform peripheral-to-peripheral as well as memory-to-memory transfer. A special Search Mode of Operation compares data read from a memory or peripheral source to the content of a pattern register.

For all DMA operations (search, transfer, and transfer-and-search), the UDC can operate with either byte or word data sizes. In some system configurations it may be necessary to transfer between, word-organized memory and a byte-oriented peripheral. The UDC provides a byte packing/unpacking capability through its byte-word funnelling transfer or transfer-and-search option. Some DMA applications may continuously transfer data between the same two memory areas. These applications may not require the flexibility inherent in reloading registers from memory tables. To service these repetitive DMA operations, base registers are provided on each channel which re-initialize the current source and destination Address and Operation Count registers. To change the data transfer direction under CPU control, provision is made for reassigning the source address as a destination and the destination as a source, eliminating the need for actual reloading of these address registers.

Frequently DMA devices must interface to slow peripherals or slow memory. In addition to providing a hardware $\overline{\text{WAIT}}$ input, the Am9516 UDC allows the user to select independently, for both source and destination addresses, automatic insertion of 0, 1, 2 or 4 wait states. The user may even disable the $\overline{\text{WAIT}}$ input pin function altogether and use these software programmed wait states exclusively.

High throughput and powerful transfer options are of limited usefulness if a DMA requires frequent reloading by the host CPU. The Am9516 UDC minimizes CPU interactions by allowing each channel to load its control parameters from memory into the channel's control registers. The only action required of the CPU is to load the address of the control parameter table into the channel's Chain Address register and then issue a "Start Chain" Command to start the register loading operation.

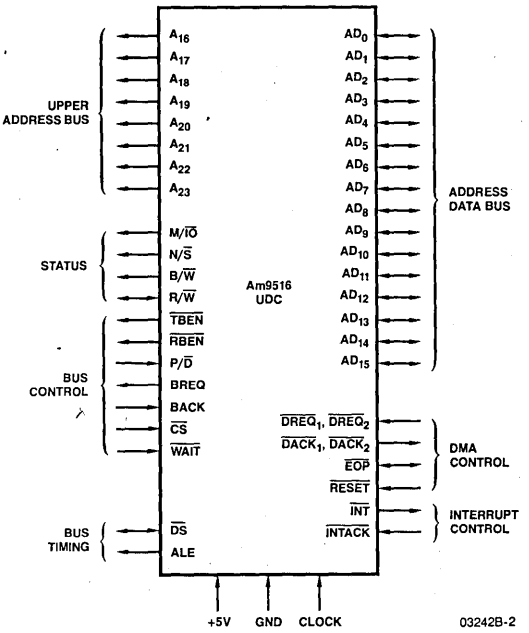
The Am9516 UDC is packaged in a 48-pin DIP and uses a single +5V power supply.

ORDERING INFORMATION

Example: Am9516 A D C B

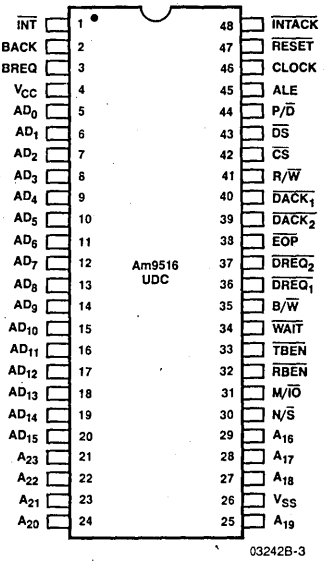
- Basic Device
- Speed: Blank = 4MHz; A = 6MHz
- Temperature Range: C = 0 to +70°C; I = −40 to +85°C; M = −55 to +125°C
- Package Type: D = Ceramic DIP; P = Plastic DIP
- Burn-in

LOGIC SYMBOL



CONNECTION DIAGRAM
Top View

D-48
P-48



Note: Pin 1 is marked for orientation.

INTERFACE SIGNAL DESCRIPTION

All inputs to the UDC, except the clock are directly TTL compatible;

VCC: **+5V Power Supply**
VSS: **Ground**

CLOCK: (Clock, Input)

The Clock signal controls the internal operations and the rates of data transfers. It is usually derived from a master system clock or the associated CPU clock. The Clock input requires a high voltage input signal. Many UDC input signals can make transitions independent of the UDC clock, these signals can be asynchronous to the UDC clock. On other signals, such as $\overline{WAIT}$ inputs, transitions must meet setup and hold requirements relative to the UDC clock. See the timing diagrams for details.

$AD_0 - AD_{15}$ (Address-Data Bus, Input/Output, Three-state)
The Address-Data Bus is a time-multiplexed, bidirectional, active High, three-state bus used for all I/O and memory transactions. HIGH on the bus corresponds to 1 and LOW corresponds to 0. AD_0 is the least significant bit position and AD_{15} is the most significant. The presence of addresses is defined by the timing edge of ALE and the asserted or requested presence of data is defined by the $\overline{DS}$ signal. The status output lines $M/\overline{IO}$ and $N/\overline{S}$ indicate the type of transaction, either memory or I/O. The $R/\overline{W}$ line indicates the direction of the transaction. When the UDC is in control of the system bus, it dominates the AD Bus; when the UDC is not in control of the system bus, the CPU or other external devices dominate the AD Bus.

The presence of address or data on the $AD_0 - AD_{15}$ bus is defined only by ALE and $\overline{DS}$. When the UDC is not in control of the bus; there is no required relation between the presence of address or data and the UDC clock. This allows the UDC to be used with a system bus which does not have a bussed clock signal.

$\overline{DS}$ (Data Strobe, Input/Output, Three-State)
Data Strobe is a bidirectional, active-low, three-state signal. A LOW on this signal indicates that the $AD_0 - AD_{15}$ bus is being used for data transfer. When the UDC is not in control of the system bus and the external system is transferring information to or from the UDC, $\overline{DS}$ is a timing input used by the UDC to move data to or from the $AD_0 - AD_{15}$ bus. Data is written into the UDC by the external system on the LOW-to-HIGH DS transition. Data is read from the UDC by the external system while $\overline{DS}$ is LOW. There are no timing requirements between $\overline{DS}$ as an input and the UDC clock; this allows use of the UDC with a system bus which does not have a bussed clock. During a DMA operation when the UDC is in control of the system, $\overline{DS}$ is an output generated by the UDC and used by the system to move data to or from the $AD_0 - AD_{15}$ bus. When the UDC has bus control, it writes to the external system by placing data on the $AD_0 - AD_{15}$ bus before the HIGH-to-LOW $\overline{DS}$ transition and holding the data stable until after the LOW-to-HIGH $\overline{DS}$ transition; while reading from the external system the LOW-to-HIGH transition of $\overline{DS}$ inputs data from the $AD_0 - AD_{15}$ bus into the UDC (see timing diagram).

R/$\overline{W}$ (Read/Write, Input/Output, Three-State)
Read/Write is a bidirectional, three-state signal. Read polarity is HIGH and write polarity is LOW. R/$\overline{W}$ indicates the data direction of the current bus transaction, and is stable starting when ALE is HIGH until the bus transaction ends (see timing diagram). When the UDC is not in control of the system bus and the external system is transferring information to or from the UDC, R/$\overline{W}$ is a status input used by the UDC to determine if data is entering or leaving on the AD_0-AD_{15} bus during $\overline{DS}$ time. In such a case, Read (HIGH) indicates that the system is requesting data from the UDC and Write (LOW) indicates that the system is presenting data to the UDC. There are no timing requirements between R/$\overline{W}$ as an input and the UDC clock; transitions on R/$\overline{W}$ as an input are only defined relative to $\overline{DS}$. When the UDC is in control of the system bus, R/$\overline{W}$ is an output generated by the UDC, 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. Flyby DMA operations are a special case where R/$\overline{W}$ is valid for the normally addressed memory or peripheral locations and must be interpreted in reverse by the "Flyby" peripheral that uses it.

$\overline{TBEN}$ (Transmit Buffer Enable, Output, Open Drain)
Transmit Buffer Enable is an active-low, open drain output. When UDC is a bus master, a LOW on this output indicates that the data is being transferred from the UDC to the data bus lines through the buffer. The purpose of this signal is to eliminate bus contention. When UDC is not in control of the system bus, these pins float to three-state OFF.

$\overline{RBEN}$ (Receive Buffer Enable, Output, Open Drain)
Receive Buffer Enable is an active-low, open drain output. When UDC is in control of system bus, a LOW on this output indicates that the data is being transferred from the data bus lines to the UDC through the buffer. The purpose of this signal is to eliminate bus contention. This pin floats to three-state OFF when the UDC is not in control of the system bus.

ALE (Address Latch Enable, Output)
This active HIGH signal is provided by the UDC to latch the address signals AD_0-AD_{15} into the address latch. This pin is never floated.

P/$\overline{D}$ (Pointer/Data, Input)
Pointer/Data is an input signal to indicate the information on the AD_0-AD_{15} bus only when the UDC is the bus slave. A HIGH on this signal indicates the information on the AD bus is an address of the internal register to be accessed. The data on AD-bus is loaded into the Pointer register of UDC. A LOW on this signal indicates that a data transfer is taking place between the bus and the internal register designated by the Pointer register. Note that if a transaction is carried out with R/$\overline{W}$ HIGH and P/$\overline{D}$ HIGH, the contents of the Pointer register will be read.

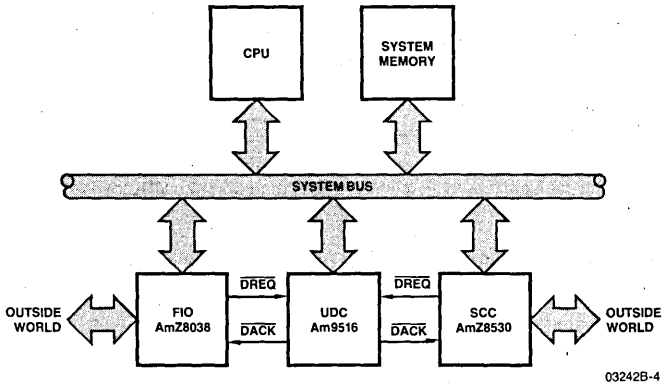
M/$\overline{IO}$ (Memory/Input-Output, Output, Three-state)
This signal specifies the type of transaction. A HIGH on this pin indicates a memory transaction. A LOW on this pin indicates an I/O transaction. It floats to three-state OFF when UDC is not in control of the system bus.

N/$\overline{S}$ (Normal/System, Output, Three-state)
This output is a three-state signal activated only when the UDC is the bus master. Normal is indicated when N/$\overline{S}$ is HIGH and System is indicated when N/$\overline{S}$ is LOW. This signal supplements the M/$\overline{IO}$ line and is used to indicate which memory or I/O space is being accessed.

B/$\overline{W}$ (Byte/Word, Output, Three-state)
This output indicates the size of data transferred on the AD_0-AD_{15} bus. HIGH indicates a byte (8-bit) and LOW indicates a word (16-bit) transfer. This output is activated when ALE is HIGH and remains valid for the duration of the whole transaction (see timing diagram). All word-sized data are word aligned and must be addressed by even addresses ($A_0 = 0$). When addressing byte read transactions, the least significant address bit determines which byte is needed; an even address specifies the most significant byte (AD_8-AD_{15}) and an odd address specifies the least significant byte (AD_0-AD_7). (Note that the higher address specifies the least significant byte!) This addressing mechanism applies to memory accesses as well as I/O accesses. When the UDC is a slave, it ignores the B/$\overline{W}$ signal and this pin floats to three-state OFF.

$\overline{CS}$ (Chip Select, Input)
This pin is an active-low input. A CPU or other external device uses $\overline{CS}$ to activate the UDC for reading and writing of its

Figure 2. UDC Configurations



internal registers. There are no timing requirements between the $\overline{CS}$ input and the UDC clock; the $\overline{CS}$ input timing requirements are only defined relative to DS. This pin is ignored when UDC is in control of system bus.

$\overline{WAIT}$ (Wait, Input)
This pin is an active-low input. Slow memories and peripheral devices may use $\overline{WAIT}$ to extend $\overline{DS}$ and $\overline{RBEN}$ or $\overline{TBEN}$ during operation. Unlike the $\overline{CS}$ input, transitions on the $\overline{WAIT}$ input must meet certain timing requirements relative to the UDC clock. See Timing Diagram 4 for details. The Wait function may be disabled using a control bit in the Master Mode register (MM2).

BREQ (Bus Request, Output)
Bus Request is an active-HIGH signal used by the UDC to obtain control of the bus from the CPU. BREQ lines from multiple devices are connected to a priority encoder.

BACK (Bus Acknowledge, Input)
BACK is an active-HIGH, asynchronous input indicating that the CPU has relinquished the bus and that no higher priority device has assummed bus control. Since BACK is internally synchronized by the UDC before being used, transitions on BACK do not have to be synchronous with the UDC clock. The BACK input is usually connected to the HLDA line from the CPU or to the output of a priority decoder.

$\overline{INT}$ (Interrupt Request, Output, Open Drain)
Interrupt Request is an active-low output used to interrupt the CPU. It is driven LOW whenever the IP and CIE bits of the Status Register are set. It is cleared by UDC after receiving a clear IP command.

$\overline{INTACK}$ (Interrupt Acknowledge, Input)
Interrupt Acknowledge is an active-low input indicating that the request for interrupt has been granted. The UDC will place a vector onto the AD bus if the No Vector or Interrupt bit (MM3) is reset.

$\overline{RESET}$ (Reset, Input)
Reset is an active-low input to disable the UDC and clear its Master Mode register.

$\overline{DREQ_1}$, $\overline{DREQ_2}$ (DMA Request, Inputs)
The DMA Request lines are two active-low inputs, one per channel. They may make transitions independent of the UDC clock and are used by external logic to initiate and control DMA operations performed by the UDC.

$\overline{DACK_1}$, $\overline{DACK_2}$ (DMA Acknowledge, Output)
The DMA Acknowledge lines are active-low outputs, one per channel, which indicate that the channel is performing a DMA operation. $\overline{DACK}$ is pulsed, held active or held inactive during DMA operations as programmed in the Channel Mode register. For Flowthru operations, the peripheral is fully addressed using the conventional I/O addressing protocols and therefore may choose to ignore $\overline{DACK}$. $\overline{DACK}$ is always output as programmed in the Channel Mode register for a DMA operation, even when the operation is initiated by a CPU software request command or as a result of chaining. $\overline{DACK}$ is not output during the chaining operations.

$\overline{EOP}$ (End of Process, Input/Output)
$\overline{EOP}$ is an active-low, open-drain, bidirectional signal. It must be pulled up with an external resistor of 1.8kohm or more. The UDC emits an output pulse on $\overline{EOP}$ when a TC or MC termination occurs, as defined later. An external source may terminate a DMA operation in progress by driving $\overline{EOP}$ low. $\overline{EOP}$ always applies to the active channel; if no channel is active, $\overline{EOP}$ is ignored.

$A_{16}-A_{23}$ (Upper Address Bus, Output, Three-state)
The $A_{16}-A_{23}$ address lines are three-state outputs activated only when the UDC is controlling the system bus. Combined with the lower 16 address bits appearing on AD_0 through AD_{15} respectively, this 24-bit linear address allows the UDC to access anywhere within 16 Megabytes of memory.

REGISTER DESCRIPTION

The Am9516 UDC block diagram illustrates the internal registers. Figure 3 lists each register along with its size and read/write access restrictions. Registers which can be read by the CPU are either fast (F) or slow (S) readable. Fast registers can be read by a normal CPU I/O operation without additional wait states. Reading slow registers requires multiple wait states. Registers can be written to by the host CPU (W) and/or can be loaded by the DMA channel itself during chaining (C). All reads or writes must be word accesses since the UDC ignores the B/$\overline{W}$ line in slave mode. It is the responsibility of the user to supply the necessary external logic if slow readable registers are to be read.

The UDC registers can be categorized into chip-level registers, which control the overall operation and configuration of the UDC, and channel-level registers which are duplicated for each channel. The five chip-level registers are the Master Mode register, the Command register, the Chain Control register, the Pointer register, and the Temporary register. The Master Mode register selects the way the UDC chip interfaces to the system. The Command register is written to by the host CPU to initiate certain operations within the UDC chip, such as resetting the unit. The Chain Control register is used by a channel while it is reloading its channel-level registers from memory. The Pointer register is written to by the host CPU when the P/$\overline{D}$ input is HIGH. The data in Pointer register is the address of the internal register to be accessed. The Temporary register is used to hold data for Flowthru Transfer/Transfer-and-Searches.

The channel-level registers can be divided into two subcategories: general purpose registers, which would be found on most DMA chips, and special purpose registers, which provide additional features and functionality. The seven general purpose registers are the Base and Current Operation Count registers, the Base and Current Address registers A and B, and the Channel Mode register. The special purpose registers are the Pattern and Mask registers, the Status register, the Interrupt Vector register, the Interrupt Save register, and the Chain Address register.

The internal registers are read or written in two steps. First, the address of the register to be accessed is written to the Pointer register, when the P/$\overline{D}$ input is HIGH. Then, the data is read from or written into the desired register which is indicated by the Pointer register, when P/$\overline{D}$ input is LOW. Note that a read with P/$\overline{D}$ HIGH causes the contents of the Pointer register to be read on AD_1 through AD_6.

MASTER MODE REGISTER

The 4-bit Master Mode register, shown in Figure 4, controls the chip-level interfaces. It can be read from and written to by the host CPU without wait states through pins AD_0-AD_3, but it is not loadable by chaining. On a reset, the Master Mode register is cleared to all zeroes. The function of each of the Master Mode bits is described in the following paragraphs.

The Chip Enable bit CE = 1 enables the UDC to request the bus. When enabled, the UDC can perform DMA Operations and

Figure 3. UDC Internal Register

Name	Size	Number	Access Type	Port Address CH-1/CH-2
Master Mode Register	4 bits	1	FW	38
Pointer Register	6 bits	1	FW	
Chain-Control Register	10 bits	1	C	
Temporary Register	16 bits	1	D	
Command Register	8 bits	1	W	2E/2C*
Current Address Register – A:				
Up-Addr/Tag field	14 bits	2	CFW	1A/18
Lower Address field	16 bits	2	CFW	0A/08
Current Address Register – B:				
Up-Addr/Tag field	14 bits	2	CFW	12/10
Lower Address field	16 bits	2	CFW	02/00
Base Address Register – A:				
Up-Addr/Tag field	14 bits	2	CFW	1E/1C
Lower Address field	16 bits	2	CFW	0E/0C
Base Address Register – B:				
Up-Addr/Tag field	14 bits	2	CFW	16/14
Lower Address field	16 bits	2	CFW	06/04
Current Operation Count	16 bits	2	CFW	32/30
Base Operation Count	16 bits	2	CFW	36/34
Pattern Register	16 bits	2	CSW	4A/48
Mask Register	16 bits	2	CSW	4E/4C
Status Register	16 bits	2	F	2E/2C
Interrupt Save Register	16 bits	2	F	2A/28
Interrupt Vector Register	8 bits	2	CSW	5A/58
Channel Mode Register – High	5 bits	2	CS	56/54
Channel Mode Register – Low	16 bits	2	CSW	52/50
Chain Address Register:				
Up-Addr/Tag field	10 bits	2	CFW	26/24
Lower Address field	16 bits	2	CFW	22/20

Access Codes:
C = Chain Loadable
D = Accessible by UDC channel
F = Fast Readable
S = Slow Readable
W = Writeable by CPU

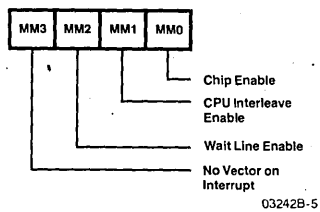
Note: The address of the register to be accessed is stored in the Pointer register.

*Port addresses of Command register can be used alternately for both channels except when issuing a "set or clear IP" command.

reload registers. It can always issue interrupts and respond to interrupt acknowledges. When the Chip Enable bit is cleared, the UDC is inhibited from requesting control of the system bus and, therefore, inhibited from performing chaining or DMA operations.

The CPU Interleave bit enables interleaving between the CPU and the UDC.

Figure 4. Master Mode Register



The Wait Line Enable bit is used to enable sampling of the $\overline{WAIT}$ line during Memory and I/O transactions. Because the UDC provides the ability to insert software programmable wait states, many users may disable sampling of the $\overline{WAIT}$ pin to eliminate the logic driving this pin. The Wait Line Enable bit provides this flexibility. See the "Wait States" section of this document for details on wait state insertion.

The "No Vector on Interrupt" bit selects whether the UDC channel or a peripheral returns a vector during interrupt acknowledge cycles. When this bit is cleared, a channel receiving an interrupt acknowledge will drive the contents of its Interrupt Save register onto the $AD_0 - AD_{15}$ data bus while $\overline{INTACK}$ is LOW. If this bit is set, interrupts are serviced in an identical manner, but the $AD_0 - AD_{15}$ data bus remains in a high impedance state throughout the acknowledge cycle.

POINTER REGISTER

The Pointer register contains the address of the internal register to be accessed. It can be read or written by CPU when the P/$\overline{D}$ line is HIGH.

CHAIN CONTROL REGISTER

When a channel starts a chaining operation, it fetches a Reload word from the memory location pointed to by the Chain Address register (Figure 11). This word is then stored in the Chain Control register. The Chain Control register cannot be written to or read from by the CPU. Once a channel starts a chain operation, the channel will not relinquish bus control until all registers specified in the Reload word are reloaded unless an $\overline{EOP}$ signal is issued to the chip. Issuing an $\overline{EOP}$ to a channel during chaining will prevent the chain operation from resuming and the contents of the Reload Word register can be discarded.

TEMPORARY REGISTER

The Temporary register is used to stage data during Flowthru transfers and to hold data being compared during a Search or a Transfer-and-Search. The Temporary register cannot be written to or read from by the CPU. In byte-word funnelling, data may be loaded into or from the Temporary register on a byte-by-byte basis, with bytes sometimes moving between the low byte of the data bus and the high byte of the Temporary register or vice-versa. See the "Transfer" section for details.

COMMAND REGISTER

The UDC Command register (Figure 20) is an 8-bit write-only register written to by the host CPU. The Command register is loaded from the data on $AD_7 - AD_0$; the data on $AD_{15} - AD_8$ is disregarded. A complete discussion of the commands is given in the "Command Descriptions" section.

CURRENT AND BASE ADDRESS REGISTER A AND B

The Current Address registers A and B (Current ARA and ARB) are used to point to the source and destination addresses for DMA operations. The contents of the Base ARA and ARB registers are loaded into the Current ARA and ARB registers at the end of a DMA operation if the user enables Base-to-Current reloading in the Completion Field of the Channel Mode register. This facilitates DMA operations without reloading of the Current registers. The ARA and ARB registers can be loaded during chaining, can be written to by the host CPU without wait states and can be read by the CPU.

Each of the Base and Current ARA and ARB registers consists of two words organized as a 6-bit Tag Field and an 8-bit Upper Address in one word and a 16-bit Lower Address in the other. See Figure 5. The Tag Field selects whether the address is to be incremented, decremented or left unchanged, and the status codes associated with the address. The Tag field also allows the user to insert 0, 1, 2 or 4 wait states into memory or I/O accesses addressed by the offset and segment fields.

The Address Reference Select Field in the Tag field selects whether the address pertains to memory space or I/O space. Note that the N/$\overline{S}$ output pin may be either HIGH (indicating Normal) or LOW (indicating System) for space. At the end of each iteration of a DMA Operation, the user may select to leave the address unchanged or to increment it or to decrement it. I/O addresses, if changed, are always incremented/decremented by 2. Memory addresses are changed by 1 if the address points to a byte operand (as programmed in the Channel Mode register's Operation field) and by 2 if the address points to a word operand. Note that if an I/O or memory address is used to point to a word operand, the address must be even to avoid unpredictable results. An address used to point to a byte operand may be even or odd. Since memory byte operand addresses will increment/decrement by 1, they will toggle between even and odd values. Since I/O byte operand addresses will increment/decrement by 2, once programmed to an even or odd value, they will remain even or odd, allowing consecutive I/O operations to access the same half of the data bus. High bus is for even address and low bus for odd.

CURRENT AND BASE OPERATION COUNT REGISTERS

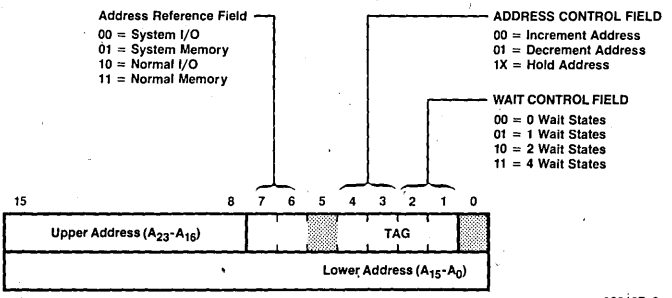
Both the Current and Base Operation Count registers may be loaded during chaining, and may be written to and read from by the host CPU.

The 16-bit Current Operation Count register is used to specify the number of words or bytes to be transferred, searched or transferred-and-searched. For word-to-word operations and byte-word funnelling, the Current Operation Count register must be programmed with the number of words to be transferred or searched.

Each time a datum is transferred or searched, the Operation Count register is decremented by 1. Once all of the data is transferred or searched, the transfer or search operation will stop, the Current Operation Count register will contain all zeroes, and the TC bit in Status Register will be '1'. If the transfer or search stops before the Current Operation Count register reaches 0, the contents of the register will indicate the number of bytes or words remaining to be transferred or searched. This allows a channel which had been stopped prematurely to be restarted where it left off without requiring reloading of the Current Operation Count register.

For the byte-to-byte operations, the Current Operation Count register should specify the number of bytes to be transferred or searched. The maximum number of bytes which can be specified is 64K bytes; by setting the Current Operation Count register to 0000.

Figure 5. Address Register A and B



PATTERN AND MASK REGISTERS

The 16-bit Pattern and Mask registers are used in Search and Transfer-and-Search operations. Both the Pattern and Mask registers may be loaded by chaining, may be written to by the host CPU, and may be read from by the host CPU, provided wait states are inserted, since these registers are slow readable. The Pattern register contains the pattern that the read data is compared to. Setting a Mask register bit to '1' specifies that the bit always matches. See the "Search" and "Transfer-and-Search" sections for further details.

STATUS REGISTER

The two 16-bit Status registers, depicted in Figure 6, are read-only registers which can be read by the CPU without wait states. Each of these registers reports on the status of its associated channel.

The Interrupt Status Field in the Status register contains the Channel Interrupt Enable (CIE), and Interrupt Pending (IP) bits. These bits are described in detail in the "Interrupt" section of this document.

The UDC status field contains the current channel status. The "channel initialized and waiting for request" status is not explicitly stated – it is reflected by Status register bits ST_{12} through ST_9 being all zero. The "Waiting for Bus" (WFB) status will cause bit ST_{10} to be set and indicates that the channel wants bus control to perform a DMA operation. The channel may or may not actually be asserting BREQ HIGH, depending on the programming of the Master Mode Chip Enable bit (MM0) when the channel decided it wanted the bus. See the "Bus Request/Grant" section for details. If a channel completes a DMA operation and neither Base-to-Current reloading nor auto-chaining were enabled, the No Auto-Reload or Chaining (NAC) bit will be set. The NAC bit will be reset when the channel receives a "Start Chain Command." If two interrupts are queued, the Second Interrupt Pending bit (SIP) will be set and the channel will be inhibited from further activity until an interrupt acknowledge occurs. See the "Interrupt" section for details. Finally, if the channel is issued an EOP during chaining, the Chaining Abort (CA), and the NAC will be set. These bits are also set when a "reset" is issued to the UDC. The CA bit holds the NAC bit in the set state. The CA bit is cleared when a new Chain Upper Address and Tag word or Lower Address word is loaded into the channel.

The Hardware Interface Field provides a Hardware Request (HRQ) bit which provides a means of monitoring the channels $\overline{DREQ}$ input pin. When the $\overline{DREQ}$ pin is LOW, the HRQ bit will be '1' and vice-versa. The Hardware Mask (HM) bit, when set, prevents the UDC from responding to a LOW on $\overline{DREQ}$. Note, however, that the Hardware Request bit always reports the true (unmasked) status of $\overline{DREQ}$ regardless of the setting of the HM bit.

The Completion Field stores data at the end of each DMA operation. This data indicates why the DMA operation ended. When the next DMA operation ends, new data is loaded into these bits overwriting, and thereby erasing the old setting. Three bits indicate whether the DMA operation ended as a result of a TC, MC or EOP termination. The TC bit will be '1' if the Operation Count reaching zero ended the DMA operation. The MC bit will be '1' if an MC termination occurred regardless of whether Stop-on-Match or Stop-on-no-Match was selected. The EOP bit is set only when an external $\overline{EOP}$ ends a DMA transfer; it is not set for $\overline{EOP}$ issued during chaining. Note that two or even all three of MC, TC and EOP may be set if multiple reasons existed for ending the DMA operation. The MCH and MCL bits report on the match state of the upper and lower comparator bytes respectively. These bits are set when the associated comparator byte has a match and are reset otherwise, regardless of whether Stop-on-Match or Stop-on-no-Match is programmed. Regardless of the DMA operation performed, these bits will reflect the comparator status at the end of the DMA operation. These two bits are provided to help determine which byte matched or did not match when using 8-bit matches with word searches and transfer-and-searches. The three reserved bits return zeroes during reads.

INTERRUPT VECTOR AND INTERRUPT SAVE REGISTERS

Each channel has an Interrupt Vector register and an Interrupt Save register. The Interrupt Vector is 8-bit wide and is written to and read from on $AD_0 - AD_7$. The Interrupt Save register may be read by the CPU without wait states. The Interrupt Vector register contains the vector or identifier to be output during an Interrupt Acknowledge cycle. When an interrupt occurs (IP = 1) either because a DMA operation terminated or because $\overline{EOP}$ was driven LOW during chaining, the contents of the Interrupt Vector register and part of the Channel Status register are stored in the 16-bit Interrupt Save register (See Figure 7).

Figure 6. Status Register

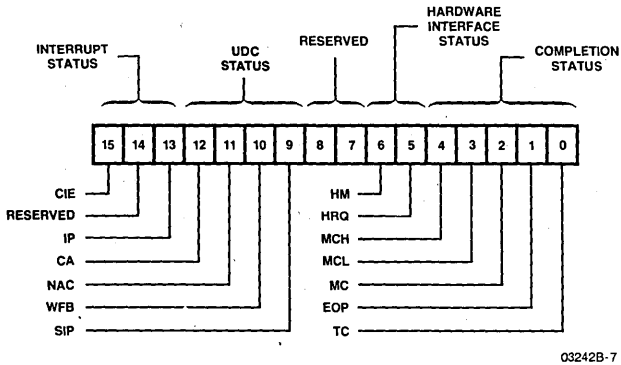
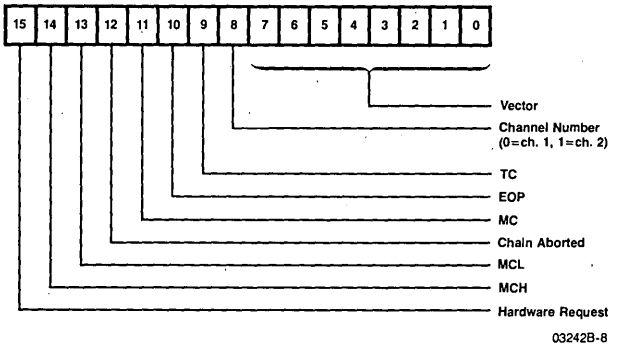


Figure 7. Interrupt Save Register



Because the vector and status are stored, a new vector can be loaded into Interrupt Vector register during chaining and a new DMA operation can be performed before an interrupt acknowledge cycle occurs. If another interrupt occurs on the channel before the first is acknowledged, further channel activity is suspended.

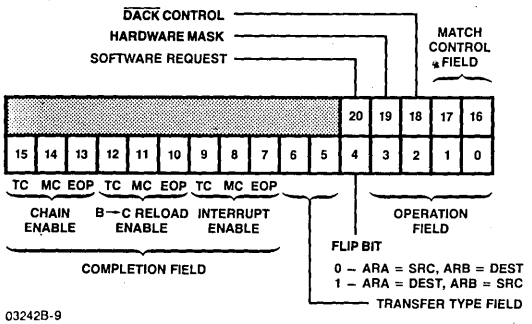
As soon as the first clear IP command is issued, the status and vector for the second interrupt is loaded into the Interrupt Save register and channel operation resumes. The UDC can retain only two interrupts for each channel; a third operation cannot be initiated until the first interrupt has been cleared. See the "Interrupt" section for further details.

CHANNEL MODE REGISTER

The channel Mode registers are two words wide. There are 21 bits defined in each Channel Mode register, the other 11 bits are unused. See Figure 8. The Channel Mode registers may be loaded during chaining and may be read by the host CPU. CPU reads of the Channel Mode register are slow reads and require insertion of multiple wait states. The Channel Mode Low word (bits 0–15) may be written to directly by the host CPU. The Channel Mode register selects what type of DMA operation the channel is to perform, how the operation is to be executed, and what action, if any, is to be taken when the channel finishes.

The Data Operation Field and the Transfer Type field select the type of operation the channel is to perform. It also selects the operand size of bytes or words (see Figure 9 for code-definition). The different types of operations are described in detail in the "DMA Operations" section. The Flip bit is used to select whether the Current ARA register points to the source and the Current ARB register points to the destination or vice-versa.

Figure 8. Channel Mode Register



The Completion Field is used to program the action taken by the channel at the end of a DMA operation. This field is discussed in the "Completion Options" section. The 2-bit Match Control field selects whether matches use an 8-bit or 16-bit pattern and whether the channel is to stop-on-match or stop-on-no-match. See Figure 9 and the "Search" section for details. The Software Request bit and Hardware Mask bit can be set and cleared by software command in addition to being loaded in parallel with other Channel Mode bits. These bits are described in detail in the "Initiating DMA Operations" section.

The $\overline{\text{DACK}}$ Control bit is used to specify when the $\overline{\text{DACK}}$ pin is driven active. When this bit is cleared, the channel's $\overline{\text{DACK}}$ pin will be active whenever the channel is performing a DMA Operation, regardless of the type of transaction. Note that the pin will not be active while the channel is chaining. If this bit is set, the $\overline{\text{DACK}}$ pin will be inactive during chaining, during both Flowthru

Figure 9. Channel Mode Coding

DATA OPERATION FIELD			
Code/Operation	Operand Size ARA	Operand Size ARB	Transaction Type
Transfer			
0001	Byte	Byte	Flowthru
100X	Byte	Word	Flowthru
0000	Word	Word	Flowthru
0011	Byte	Byte	Flyby
0010	Word	Word	Flyby
Transfer-and-Search			
0101	Byte	Byte	Flowthru
110X	Byte	Word	Flowthru
0100	Word	Word	Flowthru
0111	Byte	Byte	Flyby
0110	Word	Word	Flyby
Search			
1111	Byte	Byte	N/A
1110	Word	Word	N/A
101X	Illegal		

TRANSFER TYPE FIELD AND MATCH CONTROL FIELD		
Transfer Type	Code	Match Control
Single Transfer	00	Stop on No Match
Demand Dedicated/Bus Hold	01	Stop on No Match
Demand Dedicated/Bus Release	10	Stop on Word Match
Demand Interleave	11	Stop on Byte Match

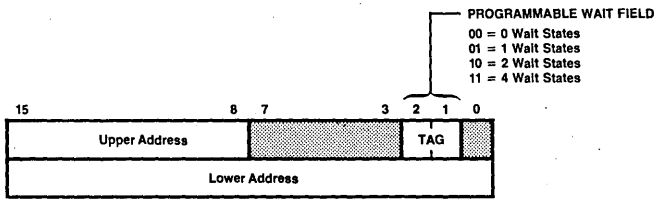
Transfers and Flowthru Transfer-and-Searches and during Searches, but $\overline{DACK}$ will be pulsed active during Flyby Transfers and Flyby Transfers-and-Searches at the time necessary to strobe data into or out of the Flyby peripheral. Flyby operations are discussed in detail in the "Flyby Transactions" section.

CHAIN ADDRESS REGISTER

Each channel has a Chain Address register which points to the chain control table in memory containing data to be loaded into the channel's registers. The Chain Address register, as shown in Figure 10, is two words long. The first word consists of an Upper Address and Tag field. The second word contains the 16-bit Lower Address portion of the memory address. The Tag field contains 2 bits used to designate the number of wait states to be inserted during accesses to the Chain Control Table.

The Chain Address register may be loaded during chaining and may be read from and written to by the host CPU without wait states. If an EOP is issued to the UDC during chaining, the Chain Address register holds the old address. This is true even if the access failure occurred while new Chain Address data was being loaded, since the old data is restored unless both words of the new data are successfully read. Note, however, that EOPs that occur when chaining and while loading a new Chain Address cause the new data to be lost.

Figure 10. Chain Address Register



03242B-10

FUNCTIONAL DESCRIPTION

Any given DMA operation, be it a transfer, a search or a transfer-and-search operation, consists of three phases. In the first phase, the channel's registers are initialized to specify and control the desired DMA operation. In the second phase, the DMA operation itself is started and performed. The final phase involves terminating the DMA operation and performing any actions selected to occur on termination. Each of these different phases is described in detail in the following sections.

RESET

The UDC can be reset either by hardware or software. The software reset command is described in the "Commands" section. Hardware resets are applied by pulling $\overline{RESET}$ LOW. The UDC may be in control of the bus when a reset is applied. BACK is removed internally causing the outputs to go tri-state. If BACK remains HIGH after reset the UDC will not drive the bus unless BREQ is active. As soon as BACK goes inactive, the UDC places the $\overline{AD_0}-\overline{AD_{15}}$, $A_{16}-\overline{AD_{23}}$, R/$\overline{W}$, $\overline{DS}$, N/$\overline{S}$, M/$\overline{IO}$ B/$\overline{W}$, $\overline{TBEN}$ and $\overline{RBEN}$ signals in the high impedance state.

Both software and hardware resets clear the Master Mode register, clear the CIE, IP and SIP bits, and set the CA and NAC bits in each Channel's Status register. The contents of all other UDC registers will be unchanged for a software reset. Since a hardware reset may have been applied partway through a DMA operation being performed by a UDC channel, the channel's registers should be assumed to contain indeterminate data following a hardware reset.

The Master Mode register contains all zeroes after a reset. The UDC is disabled and the CPU interleave and hardware wait are inhibited.

Because the CA and NAC bits in the Status register are set by a reset, the channel will be prevented from starting a DMA operation until its Chain Address register's Segment, Tag and Offset fields are programmed and the channel is issued a "Start Chain" Command.

CHANNEL INITIALIZATION

The philosophy behind the Am9516 UDC design is that the UDC should be able to operate with a minimum of interaction with the host CPU. This goal is achieved by having the UDC load its own control parameters from memory into each channel. The CPU has to program only the Master Mode register and each Channel's Chain Address register. All other registers are loaded by the channels themselves from a table located in the System memory space and pointed to by the Chain Address register. This reloading operation is called chaining and the table is called the Chain Control Table.

The Upper and Lower Address fields of the Chain Address Register form a 24-bit address which points to a location in system memory space. Chaining is performed by repetitively reading words from memory. Note that the Chain Address register should always be loaded with an even Address; loading an odd Address will cause unpredictable results. The 2-bit Tag field facilitates interfacing to slow memory by allowing the user to select 0, 1, 2 or 4 programmable wait states. The UDC will automatically insert the programmed number of wait states in each memory access during chaining.

The Chain Address register points to the first word in the Chain Control Table. This word is called the Reload Word. See Figure 11. The purpose of the Reload Word is to specify which registers in the channel are to be reloaded. Reload Word bits 10-15 are undefined and may be 0 or 1. Each of bits 0 through 9 in the Reload Word correspond to either one or two registers in the channel (see Figure 12). When a Reload Word bit is '1', it means that the register or registers corresponding to that bit are to be reloaded. If a Reload Word bit is '0', the register or registers corresponding to that bit are not to be reloaded. The data to be loaded into the selected register(s) follow(s) the Reload Word in memory (i.e., the data are stored at successively larger memory addresses). The Chain Control Table is a variable length table. Only the data to be loaded are in the table and the data are packed together.

Figure 11. Chaining and Chain Control Tables

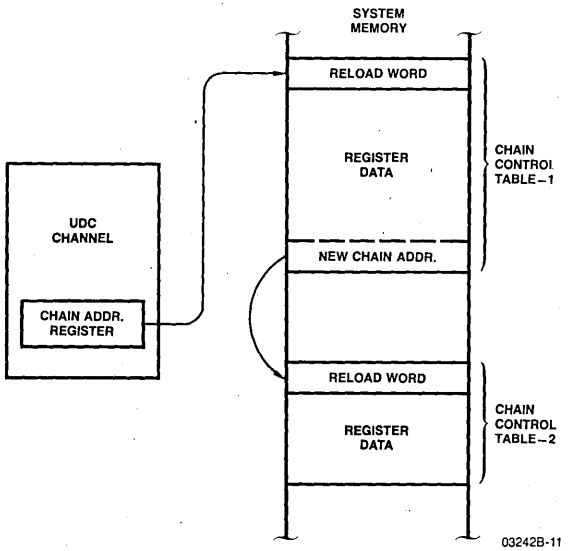
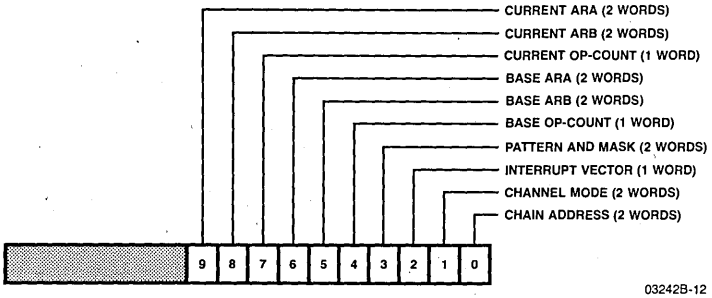


Figure 12. Reload Word/Chain Control Register



When the channel is to reload itself, it first uses the Chain Address register contents to load the Reload Word into the UDC's Chain Control Register. Next, the Chain Address register contents are incremented by two to point to the next word in memory. The channel then scans the Reload Word register from bit 9 down to bit 0 to see which registers are to be reloaded. If no registers are specified (bits 9–0 are all 0), no registers will be reloaded. If at least one of bits 9–0 are set to '1', the register(s) corresponding to the set bit are reloaded, the bit is cleared and the Chain Address register is incremented by 2. The channel continues this operation of scanning the bits from the most significant to least significant bit position, clearing each set bit after reloading its associated registers and incrementing the Chain Address register by 2. If all of bits 9 to 0 are set, all the registers will be reloaded in the order: Current ARA, Current ARB, Current Operation Count, . . . Channel Mode and Chain Address. Figure 13 shows two examples of Chain Control Tables. Example 1 shows the ordering of data when all registers are to be reloaded. In example 2 only some registers are reloaded. Once the channel is reloaded, it is ready to perform a DMA operation. Note when loading address registers, the Upper Address and Tag word are loaded first, then the Lower Address word. Also, the Pattern Register is loaded before the Mask Register.

INITIATING DMA OPERATIONS

DMA Operations can be initiated in one of three ways – by software request, by hardware request and by loading a set software request bit into the Channel Mode register during Chaining.

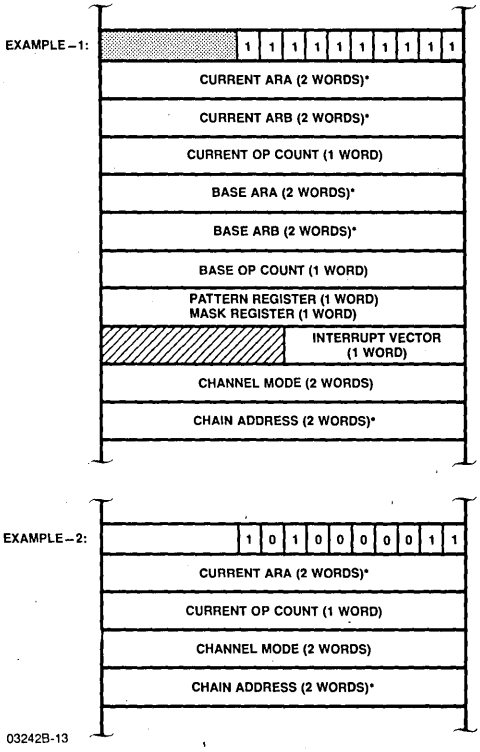
Starting After Chaining

If the software request bit of the Channel Mode register is loaded with a '1' during chaining, the channel will perform the programmed DMA operation at the end of chaining. If the channel is programmed for Single Operation or Demand, it will perform the operation immediately. The channel will give up the bus after chaining and before the operation if the CPU interleave bit in the Master Mode register is set. See the "Channel Response" section for details. Note that once a channel starts a chaining operation by fetching a Reload Word, it retains bus control at least until chaining of the last register's data is performed.

Software Requests

The CPU can issue Software Request commands to start DMA Operations on a channel. This will cause the channel to request the bus and perform transfers. See the description of the software request command for details.

Figure 13. Examples of Chain Control Table



*Load the Upper Address and Tag Word first, then the Lower Address Word.

Hardware Requests

DMA operations will often be started by applying a LOW on the channel's $\overline{DREQ}$ input. The "Channel Response" section describes when the LOW $\overline{DREQ}$ signals are sampled and when the $\overline{DREQ}$ requests can be applied to start the next DMA operation after chaining (see Timing Diagrams 1 and 2).

BUS REQUEST/GRANT

Before the UDC can perform a DMA Operation, it must gain control of the system bus. The BREQ and BACK interface pins provide connections between the UDC and the host CPU and other devices, if present, to arbitrate which device has control of the system bus. When the UDC wants to gain bus control, it drives BREQ HIGH.

Some period of time after the UDC drives BREQ HIGH, the CPU will relinquish bus control and drive its HLDA signal HIGH. When the UDC's BACK input goes HIGH, it may begin performing operations on the system bus. When the UDC finishes its operation, it stops driving BREQ HIGH.

When more than one device is used, a priority encoder and a priority decoder are used to decide the bus grant priority.

DMA OPERATIONS

There are three types of DMA operations: Transfer, Search and Transfer-and-Search. Transfers move data from a source location to a destination location. Two types of transfers are provided: Flowthru and Flyby. Searches read data from a source and compare the read data to the contents of the Pattern register. A Mask register allows the user to declare "don't care" bits.

The user can program that the search is to stop either when the read data matches the masked pattern or when the read data fails to match the masked pattern. This capability is called Stop-on-Match and Stop-on-no-Match. Transfer-and-Search combines the two functions to facilitate the transferring of variable length data blocks. Like transfer, Transfer-and-Search can be performed in either Flowthru or Flyby mode.

Transfers

Transfers use four of the Channel registers to control the transfer operation: the Current ARA and ARB registers; the Current Operation Count register; and the Channel Mode register. Channel Mode register bit CM_4 is called the Flip bit and is used to select whether ARA is to point to the source and ARB is to point to the destination or vice-versa. The Current Operation Count register specifies the number of words or bytes to be transferred.

Bits $CM_3 - CM_0$ in the Channel Mode register program whether a Flowthru or Flyby transfer is to be performed. Flowthru transfers are performed in either two or three steps. First, the channel outputs the address of the source and reads the source data into the UDC's Temporary register. In two-step Flowthru Transfer, the channel will then address the destination and write the Temporary register data to the destination location. The three-step Flowthru operation (i.e., the byte-word funnelling) is described later in this section. The source and destination for Flowthru Transfers can both be memory locations or both peripheral devices or one may be a memory location and the other a peripheral device. The $\overline{DACK}$ output for the transferring channel may be programmed to be inactive throughout the transfer or active during the transfer. This is controlled by bit CM_{18} in the Channel Mode register.

Flyby transfers provide improved transfer throughput over Flowthru but are restricted to transfers between memory and peripherals or between two peripherals. Flyby operations are described in detail in the "Flyby Transactions" section.

Transfers can use both byte- and word-sized data. Flowthru byte-to-byte transfers are performed by reading a byte from the source and writing a byte to the destination. The Current Operation Count register must be loaded with the number of bytes to be transferred. Both the Current ARA and Current ARB registers, if programmed to increment/decrement, will change by ±1 if the register points to a memory space ($TG_6 = 2$) and by ±2 if the register points to an I/O space ($TG_6 = 0$).

Flowthru word-to-word transfers require that the Current Operation Count specify the number of words to be transferred. Both the Current ARA and Current ARB registers, if programmed to increment/decrement, will change by ±2 regardless of whether the register points memory or I/O space.

Byte-word funnelling provides packing and unpacking of byte data to facilitate high speed transfers between byte and word peripherals and/or memory. This funnelling option can only be used in Flowthru mode. Funnelled Flowthru transfers are performed in three steps. For transfers from a byte source to a word destination, two consecutive byte reads are performed from the source address. The data read is assembled into the UDC's Temporary register. In the third step, the Temporary register data is written to the destination address in a word transfer. Funnelled transfers from a word source to a byte destination are performed by first loading a word from the source into the UDC's Temporary register. The word is then written out to the destination in two byte writes. For funnel operations, the byte-oriented

address must be in the Current ARA register and the word-oriented address must be in the Current ARB register. The Flip bit (CM_4) in the Channel Mode register is used to specify which address is the source and which is the destination. When the byte address is to be incremented or decremented, the increment/decrement operation occurs after each of the two reads or writes. The Current Operation Count Register must be loaded with the number of words to be transferred.

In byte-to-word funnelling operations it is necessary to specify which half of the Temporary register (upper or lower byte) is loaded with the first byte of data. Similarly, for word-to-byte funnelling operations it is necessary to define which half of the Temporary register is written out first. Figure 14 summarizes these characteristics for both byte-to-word and word-to-byte funnelling operations. The criteria used to determine the packing/unpacking order is based on whether the Current ARB register is programmed for incrementing or decrementing of the address. Note that if the address is to remain unchanged (i.e., if bit TG_4 in the Tag Field of the Current ARB register is 1), the increment/decrement bit (bit TG_3) still specifies the packing order.

Search

Searches use five of the Channel registers to control the operation: either the Current ARA or ARB; the Operation Count; the Pattern and Mask registers; and the Channel Mode register. Channel Mode register bit CM_4 is called the Flip bit and is used to select either Current ARA or ARB as the register specifying the source for the search. Only one of the Current Address registers is used for search operations since there is no destination address required. The Current Operation Count register specifies the maximum number of words or bytes to be searched.

Search operations involve repetitive reads from the peripheral or memory until the specified match condition is met. The search then stops. This is called a Match Condition or MC termination. Each time a read is performed, the Source address, if so programmed, is incremented or decremented and the Operation Count is decremented by 1. If the match condition has not been met by the time the Operation Count reaches zero, the zero value will force a TC termination, ending the search. Searches can also stop due to a LOW being applied to the $\overline{EOP}$ interface pin. During a search operation, the channel's $\overline{DACK}$ output will be either inactive or active throughout the search. This is controlled by bit CM_{18} in the Channel Mode register. The reads from the peripheral or memory performed during search follow the timing sequences described in the "Flowthru Transactions" sections.

On each read during a Search operation, the UDC's Temporary register is loaded with data and compared to the Pattern register. The user can select that the search is to stop when the Pattern and Temporary register contents match or when they don't match. This Stop-On-Match/Stop-On-No-Match feature is programmed in bit CM_{17} of the Channel Mode register. CM_2 is an enable for the output of the comparator and allows the MC signal to be generated. A Mask register allows the user to exclude or mask selected Temporary register bits from the comparison by setting the corresponding Mask register bit to '1'. The masked bits are defined to always match. Thus, in Stop-On-Match, successful matching of the unmasked bits, in conjunction with the always-matched masked bits, will cause the search to stop. For Stop-On-No-Match, the always-matched masked bits are by definition excluded from not matching and therefore excluded from stopping the search.

For word reads the user may select either 8-bit or 16-bit compares through Channel Mode register bit CM_{16}. In an 8-bit, Stop-On-Match, word-read operation, successful matching of either the upper or lower byte of unmasked Pattern and Temporary registers bits will stop the search. Both bytes do not have to match. In 16-bit Stop-On-Match with word reads, all unmasked Pattern and Temporary register bits must match to stop the search. In an 8-bit or 16-bit, Stop-On-No-Match, word-read Search operation, failure of any bit to match will terminate the Search operation.

In an 8-bit Stop-On-Match with byte-reads, the Search will stop if either the upper or lower byte of unmasked Pattern and Temporary register bits match. For an 8-bit Stop-On-No-Match with byte reads, failure of matching in any unmasked Pattern and Temporary register bit will cause the search to stop. For 8-bit searches, the upper and lower bytes of the Pattern and Mask register should usually be programmed with the same data. Failure to set the upper and lower bytes of the Pattern and Mask registers to identical values will result in different comparison criteria being used for the upper and lower bytes of the Temporary register. Users failing to program identical values for the upper and lower bytes can predict the results by recognizing that in 8-bit Stop-On-Match, the search will end if all the unmasked bits in either the upper or lower byte match, and for 8-bit Stop-On-No-Match, the failure of any unmasked bit to match will end the search. For accurate predictions, it is also necessary to know that for word reads the Temporary register high and low bytes are loaded from $AD_{15}-AD_8$ and AD_7-AD_0 respectively. In byte reads, the read byte is duplicated in both halves of the Temporary register except in funnelling.

Figure 14. Byte/Word Funnelling

Funnelling Direction	Current ARB Tag Field		Increment/Decrement and Packing/Unpacking Rules
	TG_4	TG_3	
Word-to-Byte ($CM_4 = 1$)	0	0	Increment ARB, Write High Byte First
	0	1	Decrement ARB, Write Low Byte First
	1	0	Hold ARB, Write High Byte First
	1	1	Hold ARB, Write Low Byte First
Byte-to-Word ($CM_4 = 0$)	0	0	Increment ARB, Read High Half of Word First
	0	1	Decrement ARB, Read Low Half of Word First
	1	0	Hold ARB, Read High Half of Word First
	1	1	Hold ARB, Read Low Half of Word First

Transfer-and-Search

Transfer-and-Search combines the operations of the Transfer and the Search functions. The registers used to control Transfer-and-Searches are the Current ARA and ARB registers; the Operation Count register; the Pattern and Mask registers; and the Channel Mode register.

A Transfer-and-Search operation will end when the data transferred meets the match condition specified in Channel Mode register bits $CM_{17}-CM_{16}$. The Mask and Pattern registers indicate those bits being compared with the Temporary register contents. Like Transfers and Searches, Transfers-and-Searches will also be terminated if the operation count goes to zero or if a LOW is applied to the $\overline{EOP}$ pin. Regardless of whether Transfer-and-Search stops because of a TC, MC or EOP, it will always complete the iteration by writing to the destination address before ending (writing twice for word-to-byte funnelling).

In Flowthru mode, the Transfer-and-Search timing is identical to Flowthru Transfer. While the data is in the Temporary register, it is masked by the Mask register and compared to the Pattern register. For word Transfer and Transfer-and-Search, the high and low bytes of the Temporary register are always written to and read from $AD_{15}-AD_8$ and AD_7-AD_0 respectively. For byte Transfer and Transfer-and-Search, the byte read is always loaded into both halves of the Temporary register and the entire register is driven directly out onto the $AD_{15}-AD_0$ bus. Transfer-and-Search can also be used with byte word funnelling. In funnelling, the match is an 8-bit match or 16-bit match as determined by the setting of bit CM_{16}.

Flyby Transfer-and-Search can be used to increase throughput for transfer between two peripherals or between memory and a peripheral. Memory-to-Memory Flyby is not supported. Also, in Flyby, the operand sizes of the source and destination must be the same, funnelling is not supported. A complete discussion of Flyby timing is given in the "Flyby Transactions" section. During a Flyby Transfer-and-Search, data is loaded into the Temporary register to facilitate the comparison operation and at the same time data is transferred from the source to the destination. When byte operands are used, data is loaded into both bytes of the Temporary register, from the $AD_{15}-AD_8$ bus if the Current ARA register is even and from AD_7-AD_0 line if the Current ARA register is odd. This will alternate for memory bytes so the user must drive both halves of the bus to use the search. When word operands are used, data is loaded directly from $AD_{15}-AD_8$ and AD_7-AD_0 into the Temporary register's high and low bytes respectively.

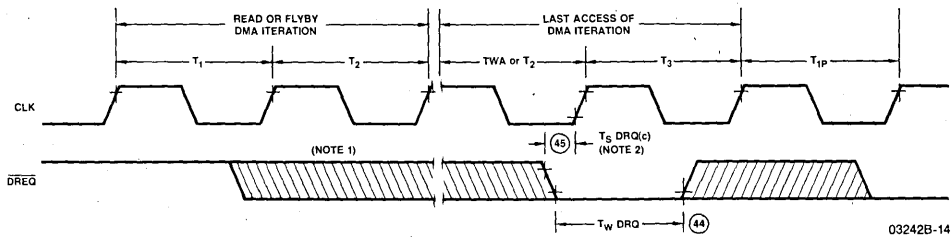
CHANNEL RESPONSE

Channel Mode register bits CM_6-CM_5 select the channel's response to the request to start a DMA operation. The response falls into either of two types: Single Operation or Demand. There are three subtypes for Demand operations: Demand Dedicated with Bus Hold, Demand Dedicated with Bus Release, and Demand Interleave. To make discussions clear, it is necessary to define the term "single iteration of a DMA operation". For Search operations, one iteration consists of a single read operation and a comparison of the read data to the unmasked Pattern register bits. The Operation Count will be decremented by 1 and the Current Address register used incremented or decremented if so programmed. For Transfer and Transfer-and-Search operations, a single iteration comprises reading a datum from the source, writing it to the destination, comparing the read datum to the unmasked Pattern register bits (Transfer-and-Search only), decrementing the Operation Count by 1 and incrementing/decrementing the Current ARA and ARB registers if so programmed. In byte-word funnelling, a single iteration consists of two reads followed by a write (Byte-to-Word funnelling) or one read followed by two writes (Word-to-Byte funnelling). In all Transfer and Transfer-and-Search cases the iteration will not stop until the data in the Temporary register is written to the destination. See Appendix B for flowchart.

Single Operation

The Single Operation response is intended for use with peripherals which transfer single bytes or words at irregular intervals. Each application of a Software request command will cause the channel to perform a single iteration of the DMA operation. Similarly, if the Software request bit is set by chaining, at the end of chaining the channel will perform a single iteration of the DMA operation. Each application of a HIGH-to-LOW transition on the $\overline{DREQ}$ input will also cause a single iteration of the DMA operation. If the Hardware mask bit is set when the transition is made, the iteration will be performed when the mask is cleared, providing the DMA operation has not terminated. See the Set/Clear Hardware mask bit command for details. Each time a Single Operation ends, the channel will give up control of the bus unless a new transition has occurred on $\overline{DREQ}$. The new transition can occur anytime after the HIGH-to-LOW ALE transition of a read or Flyby memory or I/O access of the DMA iteration. Timing Diagram 1 shows the times after which a new transition can be applied and recognized to avoid giving up the bus at the end of the current iteration.

TIMING DIAGRAM 1. Sampling DREQ During Single Transfer DMA Operations*



Notes:
1. HIGH-to-LOW $\overline{DREQ}$ transitions will only be recognized after the HIGH-to-LOW transition of the clock during T_1 of a read or flyby DMA iteration.
2. A HIGH-to-LOW $\overline{DREQ}$ transition must meet the conditions in Note 1 and must occur TsDRQ(c) before state T_3 of the last access of the DMA iteration if the channel is to retain bus control and immediately start the next iteration. $\overline{DREQ}$ may go HIGH before TsDRQ(c) if it has met the TwDRQ parameter.
3. Flyby and Search transactions have only a single access; parameter TsDRQ(c) should be referenced to the start of T_3 of the access. All other operations will always have two or three accesses per iteration.

*See Appendix D for timing parameters (page 7-268)

Demand Dedicated With Bus Hold

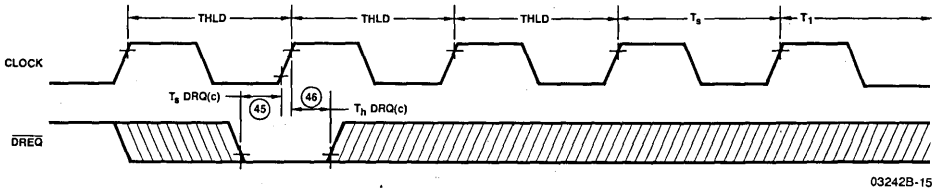
In Demand Dedicated with Bus Hold (abbreviated Bus Hold), the application of a Software request command or the setting of the software request bit during chaining or applying a LOW level on the $\overline{DREQ}$ input will cause the channel to acquire bus control.

If $\overline{DACK}$ is programmed as a level output ($CM_{18} = 0$), $\overline{DACK}$ will be active from when the channel acquires bus control to when it relinquishes control. A Software Request will cause the channel to request the bus and perform the DMA operations until TC, MC or EOP.

Once the channel gains bus control due to a LOW $\overline{DREQ}$ level, it samples $\overline{DREQ}$ as shown in Timing Diagram 2. If $\overline{DREQ}$ is LOW, an iteration of the DMA operation is performed. If $\overline{DREQ}$ is HIGH, the channel retains bus control and continues to drive all bus control signals active or inactive, but performs no DMA operation. Thus the user can start or stop execution of DMA operations by modulating $\overline{DREQ}$. Once TC, MC or EOP occurs, the channel will either release the bus or, if chaining or Base-to-Current reloading is to occur, perform the desired operation. After chaining or Base-to-Current reloading, if the channel is still in Bus Hold mode and does not have a set software request bit (set either by chaining or command), the channel will relinquish bus control unless a LOW $\overline{DREQ}$ level occurs within the time limits.

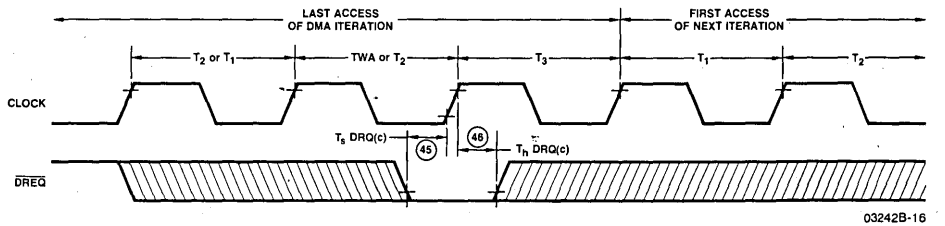
TIMING DIAGRAM 2. $\overline{DREQ}$ Sampling in Demand Mode

a) Sampling of $\overline{DREQ}$ while in Bus Hold Mode



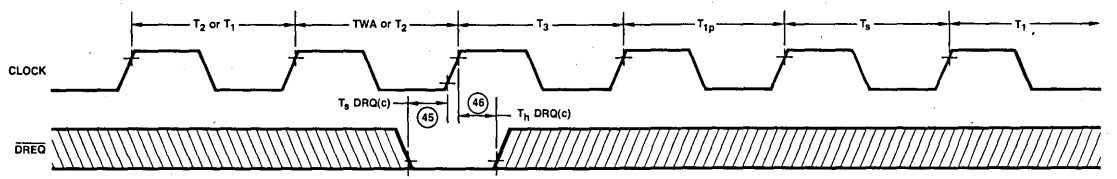
03242B-15

b) $\overline{DREQ}$ Sampling in Demand Mode During DMA Operations



03242B-16

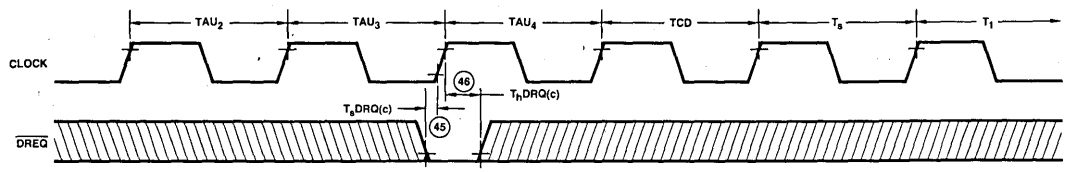
c) Sampling $\overline{DREQ}$ at the End of Chaining



(C) Sampling $\overline{DREQ}$ at the end of Chaining

03242B-17

d) Sampling $\overline{DREQ}$ at the End of Base-to-Current Reloading



03242B-18

Notes:
1. $\overline{DREQ}$ must be LOW from the start of TsDRQ(c) to the end of ThDRQ(c) to ensure that the request is recognized.
2. Failure to meet this setup time will result in the channel releasing the bus.
3. Ts is a setup state, generated before entering DMA operation cycle.
4. TAU_2, TAU_3 and TAU_4 are auto-reload states, followed by TCD (chain decision) state.

Demand Dedicated With Bus Release

In Demand Dedicated with Bus Release (abbreviated Bus Release), the application of a Software Request will cause the channel to request the bus and perform the programmed DMA operation until TC, MC or EOP. If the channel was programmed for Bus Release and the software request bit was set during chaining, the channel will start the DMA operation as soon as chaining ends, without releasing the bus, and will continue performing the operation until TC, MC or EOP.

When an active LOW $\overline{DREQ}$ is applied to a channel programmed for Bus Release, the channel will acquire the bus and perform DMA operations (a) until TC, MC or EOP or (b) until $\overline{DREQ}$ goes inactive. Timing Diagram 2 shows when $\overline{DREQ}$ is sampled to determine if the channel should perform another cycle or release the bus. Note that this sampling also occurs on the last cycle of a chaining operation. If a channel has an active $\overline{DREQ}$ at the end of chaining, it will begin performing DMA operations immediately, without releasing the bus. When a TC, MC or EOP occurs, terminating a Bus Release mode operation, the channel, if enabled for chaining and/or Base-to-Current reloading, will perform reloading and/or chaining (assuming the Status register's SIP bit is clear) without releasing the bus.

If the SIP bit of Channel Mode register is set when a DMA termination occurs, the channel will relinquish the bus control until an Interrupt Acknowledge has been received and the SIP bit is cleared. After an interrupt has been serviced, the channel will perform the Base-to-Current reloading and/or chaining, if enabled for the termination.

If an active request is not applied and the channel is in Demand Dedicated with Bus Hold, the channel will go into state THLD (see Timing Diagram 2(a)). If an active request is not applied and the channel is in Demand Dedicated with Bus Release or Demand Interleave mode, it will release the bus. Note that even if an active request is applied in Demand Interleave, the channel may still release the bus. The request for Demand Interleave should continue to be applied to ensure that the channel eventually responds to the request by acquiring the bus (i.e., the request is not latched by the channel).

Demand Interleave

Demand Interleave behaves in different ways depending on the setting of Master Mode register bit MM_2. If MM_2 is set, the UDC will always relinquish bus control and then re-request it after each DMA iteration. This permits the CPU and other devices to gain bus control. If MM_2 is clear, control can pass from one UDC channel to the other without requiring the UDC to release bus control. If both channels have active requests, control will pass to the channel which did not just have control. For instance if MM_2 is clear and both channels have active requests and are in Demand Interleave mode, control will toggle between the channels after each DMA operation iteration and the UDC will retain bus control until both channels are finished with the bus. If MM_2 is set and both channels have active requests and are in Demand Interleave mode, each channel will relinquish control to the CPU after each iteration resulting in the following control sequence: channel 1, CPU, channel 2, CPU, etc. Note that if there are other devices on the bus, they may gain control during the part of the sequence labelled CPU. See Appendix B for flowchart.

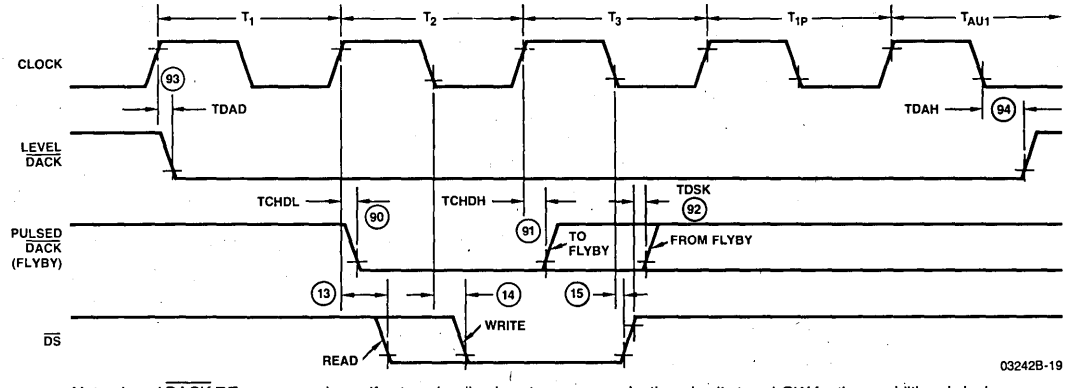
A software or hardware request will cause a channel programmed for Demand Interleave to perform interleaved DMA operations until TC, MC or EOP. If the Software request bit is set during chaining, the channel will retain the bus after chaining and will immediately start performing a DMA iteration and will interleave all DMA iterations after the first. If $\overline{DREQ}$ is LOW on the last cycle during chaining, the channel will perform a single iteration immediately after chaining and interleave thereafter until (a) TC, MC or EOP or (b) $\overline{DREQ}$ goes HIGH. If (b) occurs, the channel will relinquish the bus until $\overline{DREQ}$ goes LOW again and the channel again starts performing interleaved operations. If (a) occurs, the channel will not interleave before first performing chaining and/or Base-to-Current reloading (assuming SIP is cleared).

The waveform of $\overline{DACK}$ is programmed in Channel Mode Register (CM_{18}). The Pulsed $\overline{DACK}$ is for flyby transaction only. See Timing Diagram 3. Note: This figure shows a single Search or Flyby iteration. State TWA is optionally inserted if programmed. For more than one iteration, the level $\overline{DACK}$ output would stay active during the time the channel had bus control. When CM_{18} is set, the $\overline{DACK}$ output will be inactive for all non-flyby modes.

WAIT STATES

The number of wait states to be added to the memory or I/O transfer can be programmed by the user as 0, 1, 2 or 4 and can be separately programmed for the Current Address registers A and B and for the Chain Address register. This allows different speed memories and peripherals to be associated with each of these addresses. The Base Address registers A and B also have a Tag Field which is loaded into the Current ARA and ARB registers during Base-to-Current reloading. Because many users utilizing the software programmable wait states will not

TIMING DIAGRAM 3. DACK Timing



Note: Level $\overline{DACK}$ RE occurs as shown if auto-reloading is not programmed, otherwise it stays LOW for three additional clocks.

need the ability to generate hardware wait states through the $\overline{WAIT}$ pin, the wait function can be disabled by clearing the Wait Line Enable bit (MM_2) in the Master Mode register.

During DMA transactions, the $\overline{WAIT}$ input is sampled in the middle of the T_2 state. If $\overline{WAIT}$ is HIGH, and if no programmable wait states are selected, the UDC will proceed to state T_3. Otherwise, at least one wait state will be inserted. The $\overline{WAIT}$ line is then sampled in the middle of state TWA. If $\overline{WAIT}$ is HIGH the UDC will proceed to state T_3. Otherwise additional wait states will be inserted. (See Timing Diagram 4.)

Consider what happens in a transaction when both hardware and software wait states are inserted. Each time the $\overline{WAIT}$ line is sampled, if it is LOW, a hardware wait state will be inserted in the next cycle. The software wait state insertion will be suspended until $\overline{WAIT}$ is sampled and is HIGH. The hardware wait states may be inserted anytime during the software wait state sequence. It is important to note that hardware wait states are served consecutively rather than concurrently with software wait states. For example, assume for a Flowthru I/O Transaction that a user has programmed 4 software wait states. Driving a LOW on the $\overline{WAIT}$ input during T_2 for 2 cycles would insert 2 hardware wait states. Driving $\overline{WAIT}$ HIGH for 3 cycles would allow insertion of three of the four software wait states. Driving $\overline{WAIT}$ LOW for 2 more cycles would insert 2 more hardware wait states. Finally, driving $\overline{WAIT}$ HIGH would allow the final software wait state to be inserted. During this last software wait state, the $\overline{WAIT}$ pin would be sampled for the last time. If it is HIGH, the channel will proceed to state T_3. If the pin is LOW, the channel will insert hardware wait states until the pin goes HIGH and the channel would then enter state T_3 to complete the I/O transaction.

DMA TRANSACTIONS

There are three types of transactions performed by the Am9516 UDC: Flowthru, Flyby and Search. Figures 15 and 16 show the configurations of Flowthru and Flyby Transactions.

Flowthru Transactions

A Flowthru Transaction consists of Read and Write cycles. Each cycle consists of three states: T_1, T_2, and T_3 as shown in Timing Diagram 5. The user may select to insert software wait states through the Tag fields of the Current ARA and ARB registers. In addition, if Master Mode register bit $MM_2 = 1$, hardware wait states may be inserted by driving a LOW signal on the $\overline{WAIT}$ pin.

The M/$\overline{IO}$ and N/$\overline{S}$ lines will reflect the appropriate level for the current cycle early in T_1. The TG_6 and TG_7 bits of the current ARA and ARB registers should be programmed properly. The ALE output will be pulsed HIGH to mark the beginning of the cycle. The offset portion of the address for the peripheral being accessed will appear on AD_0-AD_{15} during T_1. The R/$\overline{W}$ and B/$\overline{W}$ lines will select a read or write operation for bytes or words. The R/$\overline{W}$, N/$\overline{S}$, M/$\overline{IO}$ and B/$\overline{W}$ lines will become stable during T_1 and will remain stable until after T_3.

I/O address space is byte-addressed but both 8- and 16-bit data sizes are supported. During I/O transactions the B/$\overline{W}$ output signal will be HIGH for byte transactions and LOW for word transactions. For I/O transactions, both even and odd addresses can be output, hence the address bit output on AD_0 may be 0 or 1.

The channel can perform both I/O read and I/O write operations, the M/$\overline{IO}$ line will be LOW. During an I/O read, the AD_0-AD_{15}

TIMING DIAGRAM 4. $\overline{WAIT}$ Timing

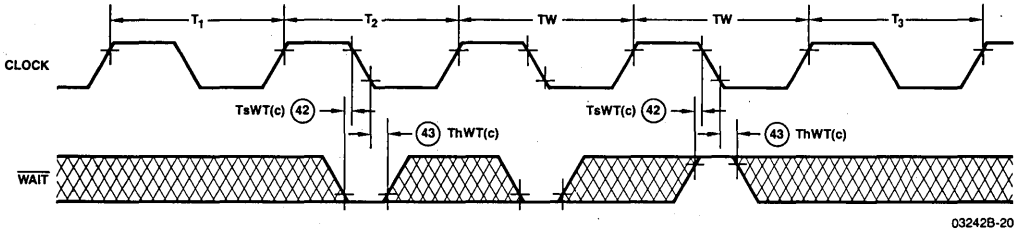


Figure 15. Configuration of Flowthru Transaction

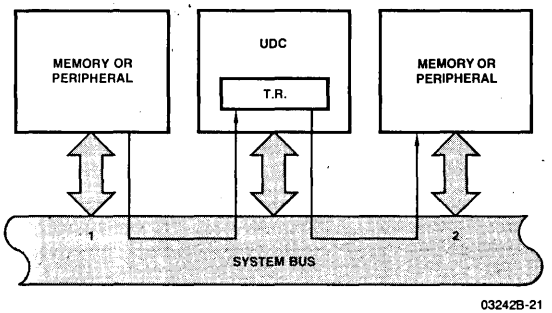
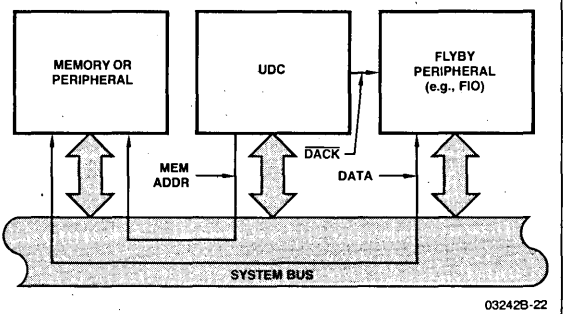


Figure 16. Configuration of Flyby Transaction



bus will be placed in the high impedance state by the UDC during T_2. The UDC will drive the $\overline{DS}$ output LOW to signal the peripheral that data can be gated onto the bus. The UDC will strobe the data into its Temporary register during T_3. $\overline{DS}$ will be driven HIGH to signal the end of the I/O transaction. During I/O write, the UDC will drive the contents of the Temporary register onto the AD_0-AD_{15} bus and shortly after will drive the $\overline{DS}$ output LOW until T_3. Peripherals may strobe the data on AD bus into their internal registers on either the falling or rising edge. If the peripheral is to be accessed in a Flyby transaction also, data should be written on the rising edge of $\overline{DS}$ only.

For byte I/O writes, the channel will drive the same data on data bus lines AD_0-AD_7 and AD_8-AD_{15}. During byte I/O reads when the address bit on AD_0 is 0, the UDC will strobe data in from data lines AD_8-AD_{15}. During byte I/O reads when the address bit on AD_0 is 1, the UDC will strobe data in from data lines AD_0-AD_7. Thus, when an 8-bit peripheral is connected to the bus, its internal registers will typically be mapped at all even or all odd addresses. To simplify accesses to 8-bit peripherals, byte oriented I/O addresses are incremented/decremented by 2.

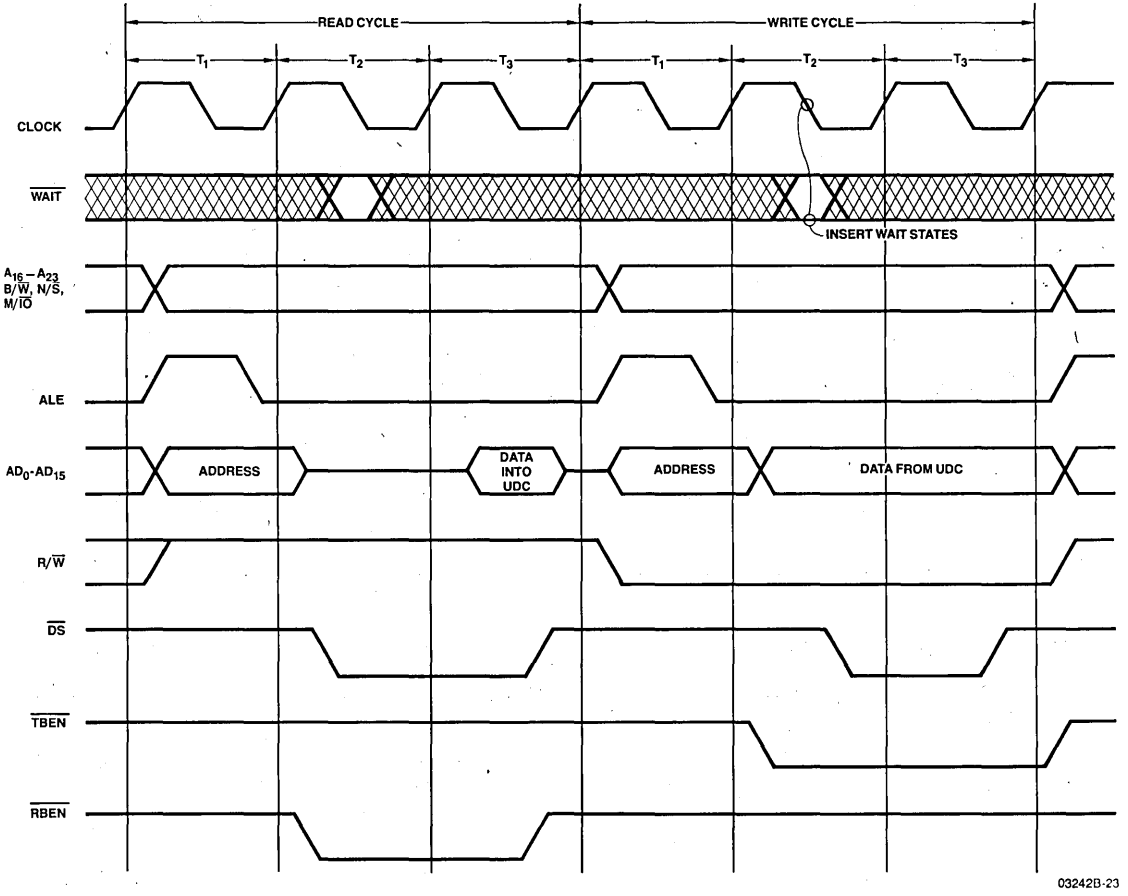
The channel can perform the I/O read and memory write operation, the memory read and I/O write operation, and the memory read and memory write operation, also. The timing for all Flowthru transactions is the same.

During chaining operations the UDC reads words from an address in System memory pointed to by the active channel's Chain Address register. Those chaining operations are performed identically to the Flowthru memory read transactions, except that the data is loaded into an internal UDC channel register rather than the Temporary register. Note that chaining never causes a write or a byte read; thus all memory writes or all byte accesses are due to DMA operations. A typical memory operation consists of three states: T_1, T_2, and T_3, as shown in Timing Diagram 5. The user may select to insert 1, 2 or 4 software wait states after state T_2 and before state T_3 by programming the Tag field of the Current Address register or the chain address register. If the Wait Line Enable bit in the Master Mode register is set, the user may also insert hardware wait states after state T_2 and before state T_3 by driving a LOW on the $\overline{WAIT}$ line. The operation of Flowthru memory transactions is performed identical to the Flowthru I/O transactions. (See Timing Diagram 5.)

Flyby Transactions

Flyby transfer and Flyby transfer-and-search operations are performed in a single cycle, providing a transfer rate significantly

TIMING DIAGRAM 5. Flowthru Transactions



faster than that available from Flowthrus. In Flyby, operations can only be performed between memory and peripheral or between peripheral and peripheral. Memory-to-memory operations can not be performed in Flyby mode; these must be done using Flowthru.

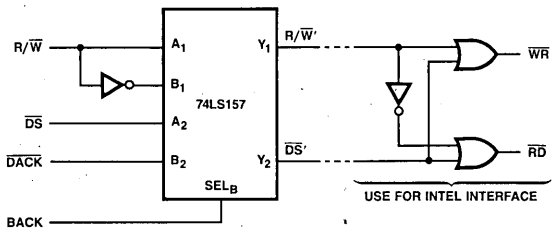
The Flyby Transaction can only be used with peripherals having a special Flyby signal input or with external logic. This Flyby input is connected to the channel's $\overline{DACK}$ output. For memory-peripheral Flyby, the address of the source memory location, must be programmed in the Current ARA register. The Current ARB register must be programmed with the destination memory location for peripheral-memory Flyby. For Flyby peripheral-to-peripheral transaction, if both peripherals have a Flyby input, only one (called "flyby peripheral") should be connected to $\overline{DACK}$; the other peripheral's Flyby input should be held high during the Flyby operation. The address of the peripheral (called "non-flyby peripheral") not connected to the channel's $\overline{DACK}$ output should be programmed in the Current ARB register when it is a destination. When the non-flyby peripheral is a source, its address should be programmed in the current ARA register. Note that a set Flip bit (CM_4 = 1) is for Flyby peripheral to Non-Flyby peripheral or Memory Write transaction (defined as "From Flyby Transaction") and a clear Flip bit (CM_4 = 0) is for the membry or non-flyby peripheral read to Flyby peripheral transaction, (defined as "To Flyby Transaction").

Transaction	CM_4	R/$\overline{W}$	Address of Memory or Non-Flyby Peripheral
To Flyby	0	HIGH	ARA
From Flyby	1	LOW	ARB

A Flyby operation is performed using three states: T_1, T_2, and T_3. During T1 the channel pulses ALE and outputs the address information. See Timing Diagram 6. The R/$\overline{W}$ line is HIGH for "To Flyby" Transaction and R/$\overline{W}$ line is LOW for "From Flyby" Transaction.

The channel's M/$\overline{IO}$ and N/$\overline{S}$ lines are coded as specified by the Current ARA or ARB Tag field. The B/$\overline{W}$ line indicates the operand size programmed in the Channel Mode register Operation field. During state T_1 the channel drives R/$\overline{W}$ line to indicate the transaction direction. during state T_2 the channel drives both $\overline{DS}$ and $\overline{DACK}$ active. The Flyby Peripheral connected to $\overline{DACK}$ inverts the R/$\overline{W}$ signal to determine whether it is being read from or writtin to (see Figure 17).

Figure 17. Flyby Peripheral Interface



$$R/\overline{W}' = \overline{BACK \cdot R/\overline{W} + \overline{BACK} \cdot \overline{R/W}}$$
$$\overline{DS}' = \overline{BACK \cdot DACK + \overline{BACK} \cdot DS}$$
$$\overline{RD} = \overline{DACK \cdot \overline{R/W} \cdot BACK + DS \cdot R/\overline{W} \cdot \overline{BACK}}$$
$$\overline{WR} = \overline{DACK \cdot R/W \cdot BACK + \overline{DS} \cdot R/W \cdot \overline{BACK}}$$

03242B-24

The pulsed $\overline{DACK}$ input serves two purposes: To select the peripheral for the Read/Write, and to provide timing information on when to drive data onto or input data from the AD_0-AD_{15} bus. Note that because the "Flyby Peripheral" never gets explicitly addressed by AD_0-AD_{15}, it must know which internal register is to be loaded from or driven onto the AD_0-AD_{15} bus. On state T_3, the $\overline{DS}$ and $\overline{DACK}$ lines are driven inactive to conclude the transfer. In Transfer-and-Search mode, data is loaded into the UDC's Temporary register on the LOW-to-HIGH DS transition in order to perform the search function.

To provide adequate data setup time, the rising edge of $\overline{DS}$ or $\overline{DACK}$ should be the edge used to perform the write to the transfer destination. To extend the active time of $\overline{DS}$ and $\overline{DACK}$, wait states can be inserted between T_2 and T_3. Software wait states can be inserted by programming the appropriate code in the Tag field of the Current ARA or ARB registers. Hardware wait states can be inserted by pulling $\overline{WAIT}$ LOW if the Wait Line Enable bit in the Master Mode register is set. The $\overline{WAIT}$ line is sampled in the middle of the T_2 or TWA state.

TERMINATION

There are three ways a Transfer-and-Search or Search operation can end and two ways a Transfer operation can end. When a channel's Current Operation Count goes to 0, the DMA operation being performed will end. This is called a TC or Terminal Count termination. A DMA operation can also be stopped by driving the $\overline{EOP}$ pin LOW with external logic. This is called an EOP termination. Search and Transfer-and-Search operations have a third method of terminating called Match Condition or MC termination. An MC termination occurs when the data being Transferred-and-Searched or Searched meets the match condition programmed in Channel Mode register bits $CM_{17}-CM_{16}$. These bits allow the user to stop when a match occurs between the unmasked Pattern register bits and the data read from the source, or when a no-match occurs. Both byte and word matches are supported. MC terminations do not apply to Transfer operations since the pattern matching logic is disabled in Transfer mode.

End-of-Process

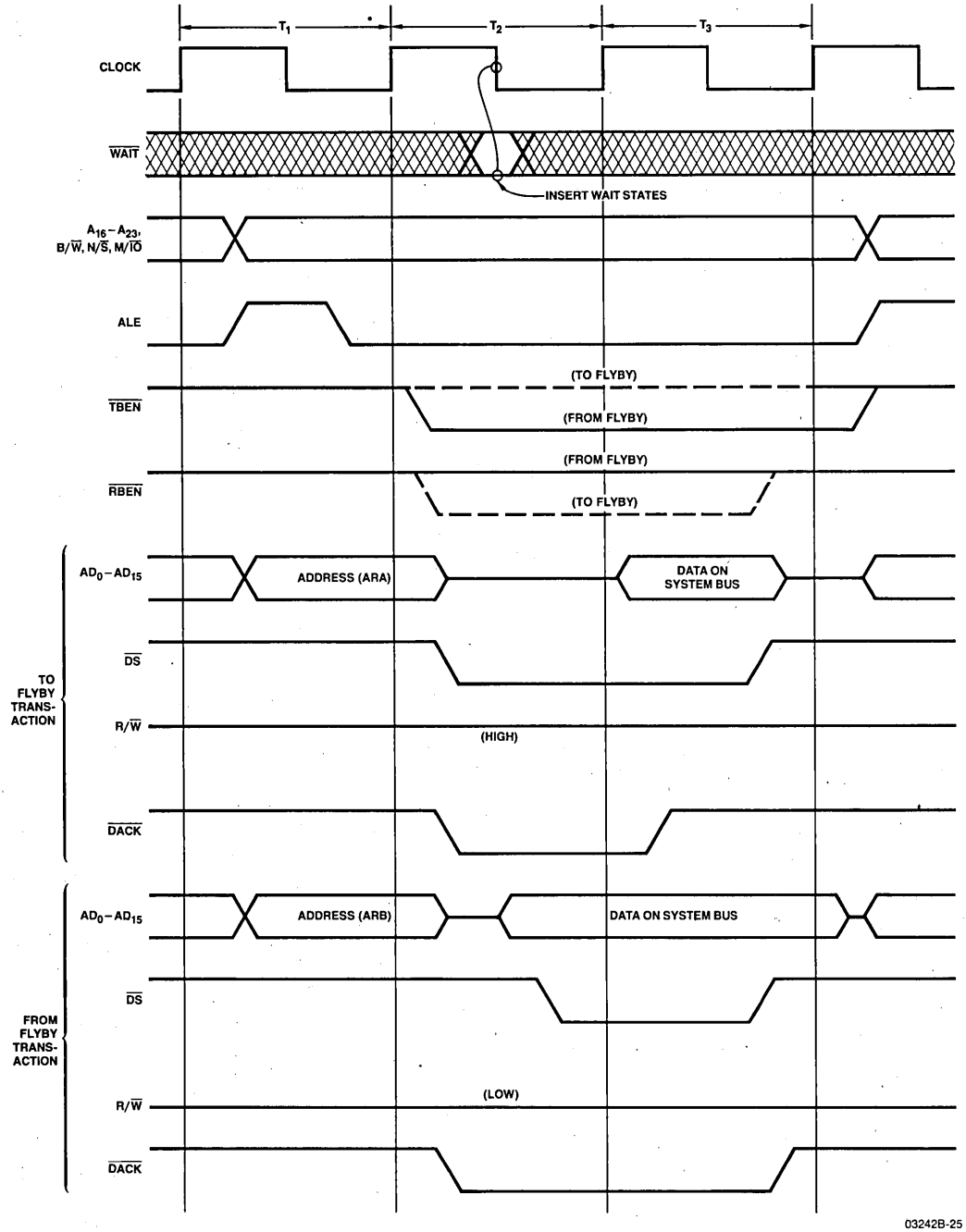
The End-of-Process (EOP) interface pin is a bi-directional signal. Whenever a TC, MC or EOP termination occurs, the UDC will drive the $\overline{EOP}$ pin LOW. During DMA operations, the $\overline{EOP}$ pin is sampled by the UDC to determine if $\overline{EOP}$ is being driven LOW by external logic. Timing Diagram 7 shows when internal EOPs are generated marking termination of all Transfers. These figures also show the point during the DMA iteration when the EOP pin is sampled. The generation of internal EOPs and sampling of external EOPs for Transfer-and-Searches follows the same timing used for Transfers. Since there is a single $\overline{EOP}$ pin for both channels, $\overline{EOP}$ should only be driven LOW by a channel while that channel is being serviced. This can be accomplished by selecting a level $\overline{DACK}$ output (CMR_{18} = 0) and gating each channel's $\overline{EOP}$ request with $\overline{DACK}$, as shown in Figure 18.

If an EOP is detected while the channel is trying to reload the Chain Address register, the new Chain Address Offset and Segment are discarded and the old address +2 is preserved to allow inspection of the erroneous address.

Programming Completion Options

When a channel ends a DMA operation, the reason for ending is stored in the Completion Status Field of the channel's Status register. See Figure 6. This information is retained until the next DMA operation ends at which time the Status register is updated to reflect the reason(s) for the latest termination. Note that it is conceivable that more than one bit in the Completion Field could be set. As an extreme example, if a channel decremented its

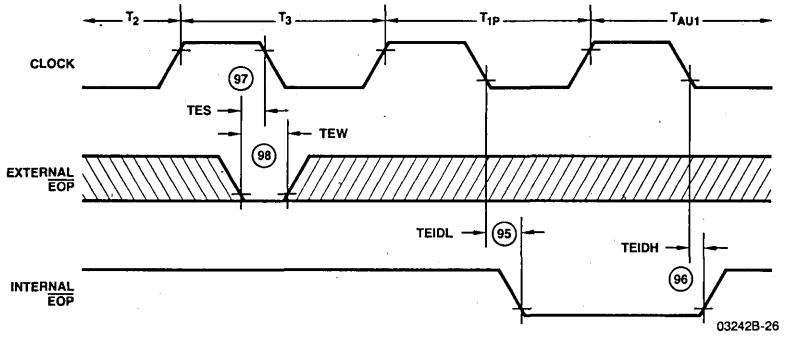
TIMING DIAGRAM 6. Flyby Transactions



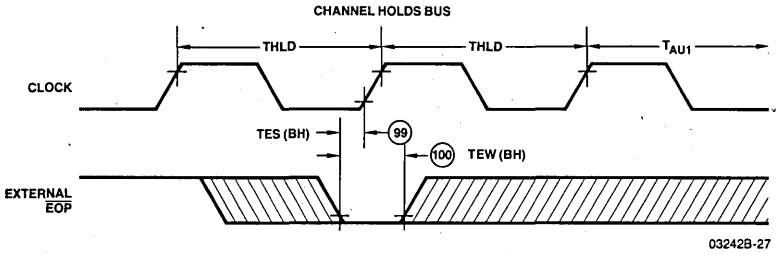
03242B-25

TIMING DIAGRAM 7. EOP Timing

a) EOP Sampling and Generation During DMA Operations

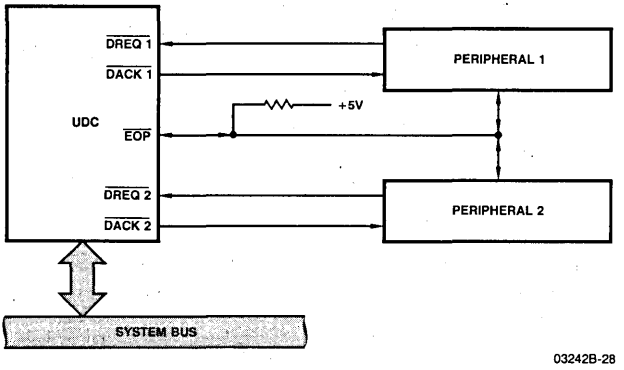


b) Sampling of EOP During Bus Hold



Notes: 1. The diagram lists state names for both I/O and memory accesses. Sampling of $\overline{EOP}$ will occur on the falling edge of state T3.
2. State T1P is a pseudo-T1 state, generated following termination of any DMA operation.
3. TAU1 is an auto-initialization state, generated following the TC, MC or $\overline{EOP}$ termination.

Figure 18. EOP Connection



Current Operation Count to zero, causing a TC termination; input data from the source generated a match causing an MC termination; and there was a LOW on $\overline{EOP}$ pin resulting in an EOP termination, all three of the channel's Status register completion bits would be set.

When a DMA operation ends, the channel can:

(a) Issue an Interrupt request (i.e., setting the IP or SIP bit of the channel's Status register);
(b) Perform Base-to-Current reloading;
(c) Chain reload the next DMA operation;
(d) Perform any combination of the above; or
(e) None of the above.

The user selects the action to be performed by the channel in the Completion option field of the Channel Mode register. For each type of termination (TC, MC or EOP) the user can choose which action or actions are to be taken. If no reloading is selected for the type of termination that occurred, the NAC bit in the Status register will be set.

More than one action can occur when a DMA operation ends. This may arise because more than one action was programmed for the applicable termination. The priorities of those actions are Interrupt request first, Base-to-Current reloading second, and then chaining. The Interrupt can not be serviced unless the UDC has relinquished the bus.

Interrupts

In order to allow the UDC to start executing a new DMA operation after issuing an interrupt, but before an interrupt acknowledge is received, a two-deep interrupt queue is implemented on each channel. The following discussion will describe the standard interrupt structure and then elaborate on the additional interrupt queuing capability of the UDC.

A complete interrupt cycle consists of an interrupt request followed by an interrupt-acknowledge transaction. The request, which consists of $\overline{INT}$ being pulled LOW, notifies the CPU that an interrupt is pending. The interrupt-acknowledge transaction, which is initiated by the CPU as a result of the request, performs two functions: it selects the peripheral whose interrupt is to be acknowledged, and it obtains a vector that identifies the selected device and operation; cause of interrupt.

A peripheral can have one or more sources of interrupt. Each interrupt source has two bits that control how it generates interrupts. These bits are a Channel Interrupt Enable bit (CIE), and an Interrupt Pending bit (IP). On the UDC, each channel is an interrupt source. The two interrupt control bits are located in bits CM_{15} and CM_{13} of each channel's Status register.

Each channel has its own vector register for identifying the source of the interrupt during an interrupt acknowledge transaction. There is one bit (MM_3) in the Master Mode register used for controlling interrupt behavior for the whole device.

Once a channel issues an interrupt, it is desirable to allow the channel to proceed with the next DMA operation before the interrupt is acknowledged. This could lead to problems if the UDC channel attempted to chain reload the Vector register contents. In such a situation, it may not be clear whether the old or new vector would be returned during the acknowledge. This dilemma is resolved in the UDC by providing each channel with an Interrupt Save register. When the channel sets IP as part of the procedure followed to issue an interrupt, the contents of the Vector register and some of the Status register bits are saved in an Interrupt Save register. See Figure 8. When an Interrupt Acknowledge cycle is performed, the contents of the Interrupt Save register are driven onto the bus. Although the use of an Interrupt Save register allows the channel to proceed with a new task, problems can still potentially arise if a second interrupt is to be issued by the channel before the first interrupt is acknowledged. To avoid conflicts between the first and second interrupt, each channel has a Second Interrupt Pending (SIP) bit in its Status register. When a second interrupt is to be issued before the first interrupt is acknowledged, the SIP bit is set and the channel relinquishes the bus until an acknowledge occurs. For compatibility with polled interrupt schemes, the Interrupt save register can be read by the host CPU without wait states. As an aid to debugging a system's interrupt logic, whenever IP is set, the Interrupt Save register is loaded from the Vector and Status registers.

Note that the SIP bit is transferred to the IP bit when IP is cleared by the host CPU. Whenever CIE is set $\overline{INT}$ will go LOW as soon as IP is set.

Base-to-Current Reloading

When a channel finishes a DMA operation, the user may select to perform a Base-to-Current Reload. (Base-to-Current reloading is also referred to as Auto-reloading in this document.) In this type of reload, the Current Address Registers A and B are loaded with the data in the Base Address Registers A and B respectively, and the Current Operation Count register is loaded with the data in the Base Operation Count. The Base-to-Current reload operation facilitates repetitive DMA operations without the multiple memory accesses required by chaining. Although the channel must have bus control to perform Base-to-Current reloading, the complete reloading operation occurs in four clock cycles (i.e., TAU_1 through TAU_4). Note that if the channel had to relinquish the bus because two unacknowledged interrupts were queued, it will have to regain bus control to perform any Base-to-Current reloading (or chaining, for that matter). In this case it acquires the system bus once an interrupt acknowledge is received, even if it immediately afterward will relinquish the bus because no hardware software request is present.

Chaining

If the channel is programmed to chain at the end of a DMA operation, it will use the Chain Address register to point to a Chain Control Table in memory. The first word in the table is a Reload word, specifying the register(s) to be loaded. Following the Reload word are the data values to be transferred into the register(s). Chaining is described in detail in the "Channel Initialization" section.

Because chaining occurs after Base-to-Current reloading, it is possible to reset the Current Address registers A and B and the Current Operation Count register to the values used for previous DMA operations and then chain reload one or two of these registers to some special value to be used perhaps for this DMA operation only. If the Base values are not reloaded during chaining, the channel can revert back to the Base values at a later cycle.

If an all zero Reload word is fetched during chaining, the chain operation will not reload any registers but in all other respects it will perform like any other chaining operation. Thus, the Chain Address will be incremented by 2 to point to the next word in memory and at the end of the all Zero-Reload word chain operation, the channel will be ready to perform a DMA operation. All zero Reload words are useful as "Stubs" to start or terminate linked lists of DMA operations traversed by chaining. On the other hand, care must be taken in their use since the channel may perform an erroneous operation if it is unintentionally started after the chaining operation.

COMMAND DESCRIPTIONS

Figure 19 shows a list of UDC commands. The commands are executed immediately after being written by the host CPU into the UDC's Command register (Figure 20). A description of each command follows.

Reset (00)
This command causes the UDC to be set to the same state generated by a Hardware Reset. The Master Mode register is set to all zeros, the CIE, IP and SIP bits are cleared, the NAC and CA bits in each channel's Status register are set and the channel activity is forbidden. The Chain Address must be programmed since its state may be indeterminate after a Reset. The lockout preventing channel activity is cleared by issuing a Start Chain command.

Start Chain Channel 1/Channel 2 (A_0/A_1)
This command causes the selected channel to clear the No Auto-Reload or Chain (NAC) bit in the channel's Status register, and to start a chain reload operation of the channel's registers, as described in the "Channel Initialization" section. These effects will take place even if the Reload word fetched is all zeros. This command will only be honored if the Chain Abort (CA) bit and Second Interrupt Pending (SIP) bit in the Channel's Status register are clear. If either the CA or SIP bit is set, this command is disregarded.

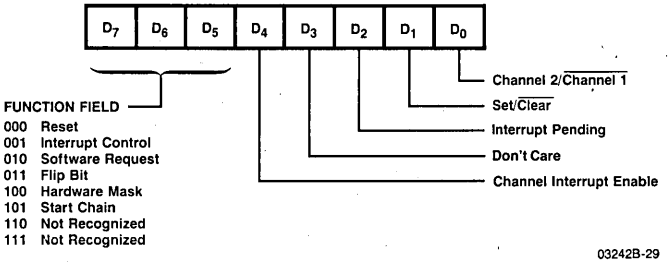
When the Waiting For Bus (WFB) bit of Status Register is set, if the "Start Chain" command is issued, the channel will honor the

Figure 19. UDC Command Summary

Command	Opcode Bits 7654	Opcode Bits 3210	Example Code HEX
Reset	000X	XXXX	00
Start Chain Channel 1	101X	XXX0	A0
Start Chain Channel 2	101X	XXX1	A1
Set Software Request Channel 1	010X	XX10	42
Set Software Request Channel 2	010X	XX11	43
Clear Software Request Channel 1	010X	XX00	40
Clear Software Request Channel 2	010X	XX01	41
Set Hardware Mask Channel 1	100X	XX10	82
Set Hardware Mask Channel 2	100X	XX11	83
Clear Hardware Mask Channel 1	100X	XX00	80
Clear Hardware Mask Channel 2	100X	XX01	81
Set CIE, or, IP Channel 1	001E	XP10	32*
Set CIE, or, IP Channel 2	001E	XP11	33*
Clear CIE, or, IP Channel 1	001E	XP00	30*
Clear CIE, or, IP Channel 2	001E	XP01	31*
Set Flip Bit Channel 1	011X	XX10	62
Set Flip Bit Channel 2	011X	XX11	63
Clear Flip Bit Channel 1	011X	XX00	60
Clear Flip Bit Channel 2	011X	XX01	61

*Notes: 1. E = Set to 1 to perform set/clear on CIE; Clear to 0 for no effect on CIE
2. P = Set to 1 to perform set/clear on IP; Clear to 0 for no effect on IP
3. X = "don't care" bit. This bit is not decoded and may be 0 or 1

Figure 20. Command Register



command after one DMA iteration. It is nearly impossible for the CPU to issue a command when WFB = 1 and the UDC is enabled.

Software Request Channel 1/Channel 2 (Set: 42/43, Clear: 40/41)

This command sets or clears the software request bit in the selected channel's Mode register. If the Second Interrupt Pending (SIP) bit and No Auto-Reload or Chain (NAC) bit in the channel's Status register are both cleared, the channel will start executing the programmed DMA operation. If either the SIP or NAC bit is set, the channel will not start executing a DMA operation until both bits are cleared. The SIP bit will clear when the channel receives an interrupt acknowledge. One way to clear the NAC bit is to issue a Start Chain command to the channel. If the fetched Reload Word is all zeros, the channel's registers will remain unchanged and the software request bit, if set earlier by command, will cause the programmed DMA operation to start immediately. If during chaining new information is loaded into the Channel Mode register this new information will, of course, overwrite the software request bit.

Set/Clear Hardware Mask 1/Mask 2 (Set: 82/83; Clear: 80/81)

This command sets or clears the Hardware Mask bit in the selected channel's Mode register. This command always takes effect. The Hardware Mask bit inhibits recognition of an active signal on the channel's $\overline{DREQ}$ input; this bit does not affect recognition of a software request. If the channel is in single transfer mode, it performs DMA operations upon receipt of a transition on $\overline{DREQ}$ rather than in response to a $\overline{DREQ}$ level. Transitions occurring while the Hardware Mask bit is set will be stored and serviced when the Hardware Mask is cleared, assuming the Channel has not chained. The UDC will request the system bus 1 1/2 to 2 clocks after the receipt of any $\overline{DREQ}$, after which a minimum of one DMA iteration is unavoidable. $\overline{DREQ}$ transitions are only stored for the current DMA operation. If the channel performs a chain operation of single transfer mode, any $\overline{DREQ}$ transition stored for later service is cleared.

Timing Diagrams 1 and 2 show the minimum times when a new $\overline{DREQ}$ can be applied if it is to be serviced by the new DMA operation. Note in Diagram 1 the notation of First iteration and Last iteration. This means for example, $\overline{DREQ}$ may be asserted during the write cycle T_1 of a Flowthru transaction, but may never be asserted during T_1 of a Flyby transaction since Flyby is done in one iteration.

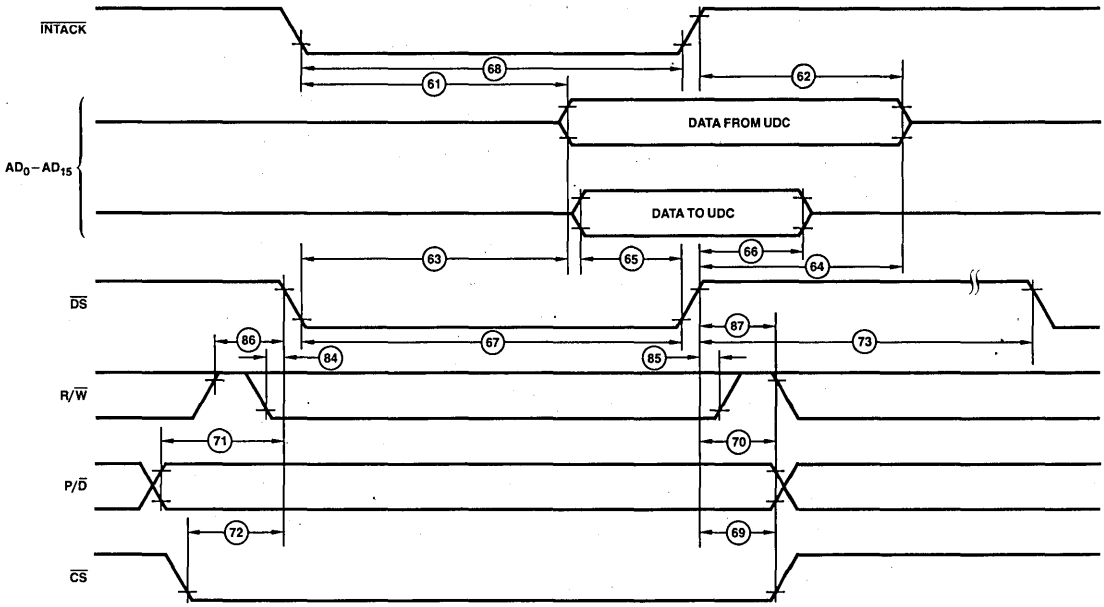
Set/Clear CIE, and IP Channel 1/Channel 2 (see Figure 19)

This command allows the user to either set or clear any combination of the CIE, and IP bits in the selected channel's Status register. These bits control the operation of the channel's interrupt structure and are described in detail in the "Interrupts" section. Setting the IP bit causes the Interrupt Save register to be loaded with the current Vector and Status. The IP bit is cleared to facilitate an efficient conclusion to the processing of an interrupt.

Set/Clear Flip Bit Channel 1/Channel 2 (Set: 62/63; Clear: 60/61)

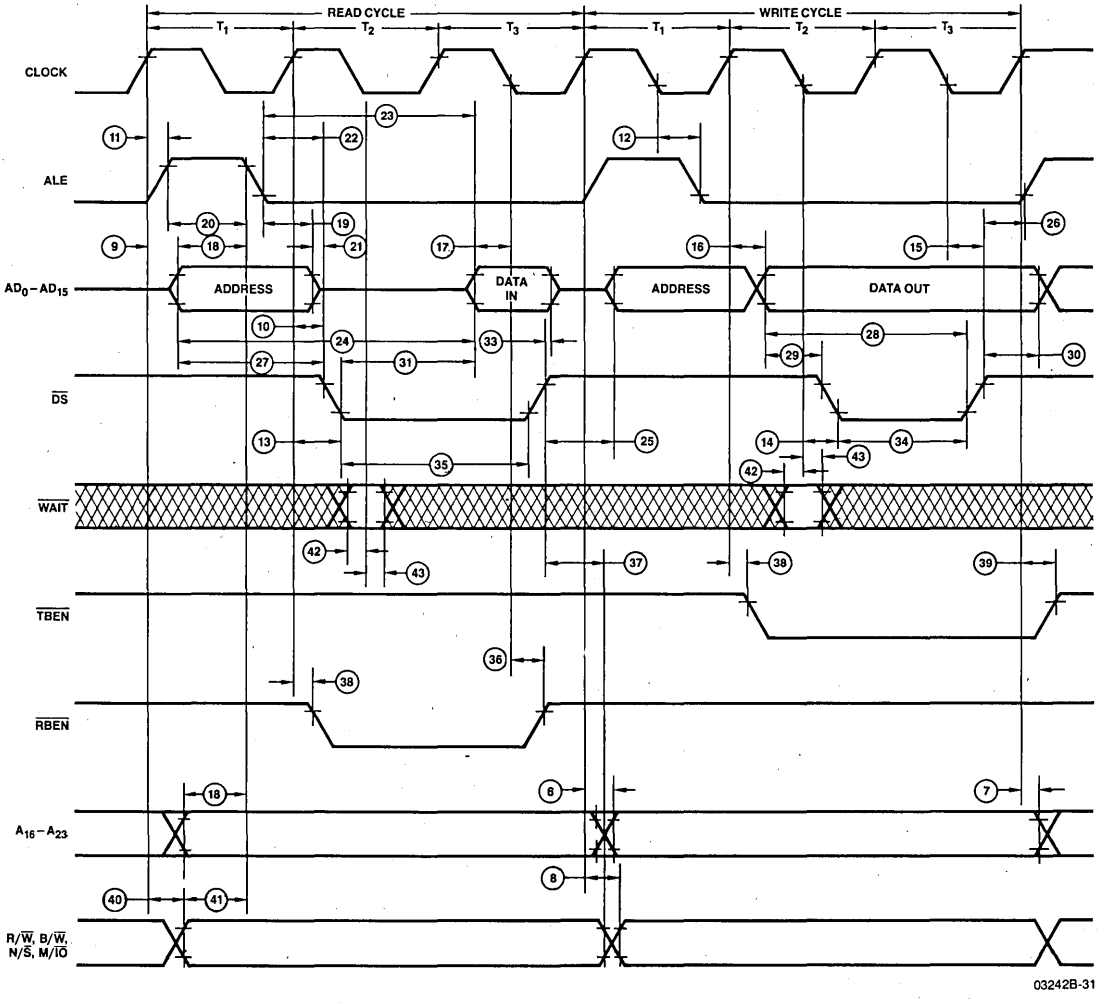
The Flip Bit in the selected channel's Mode register can be cleared and set by this command. This allows the user to reverse the source and destination and thereby reverse the data transfer direction without reprogramming the channel. This command will be most useful when repetitive DMA operations are being performed by the channel, using Base-to-Current reloading for channel reinitialization and using this command to control the direction of transfer. Chaining new information into the Channel Mode register will, of course, overwrite the Flip bit.

TIMING DIAGRAM 8. AC Timing when UDC is a Bus Slave



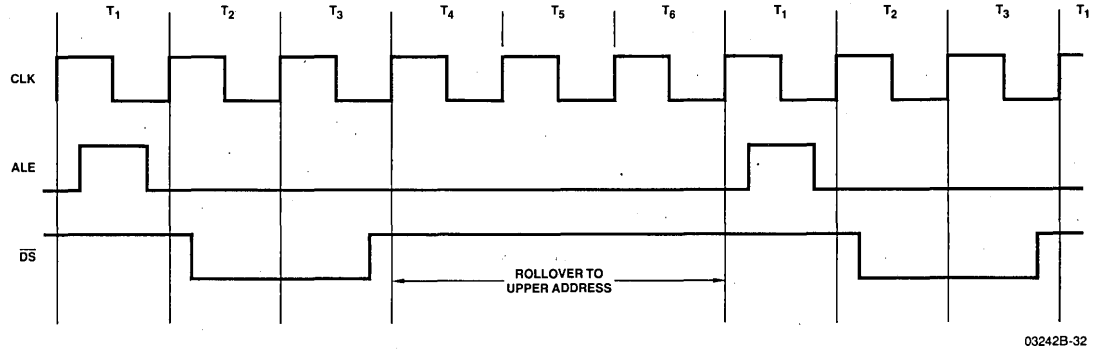
03242B-30

TIMING DIAGRAM 9. AC Timing when UDC is a Bus Master



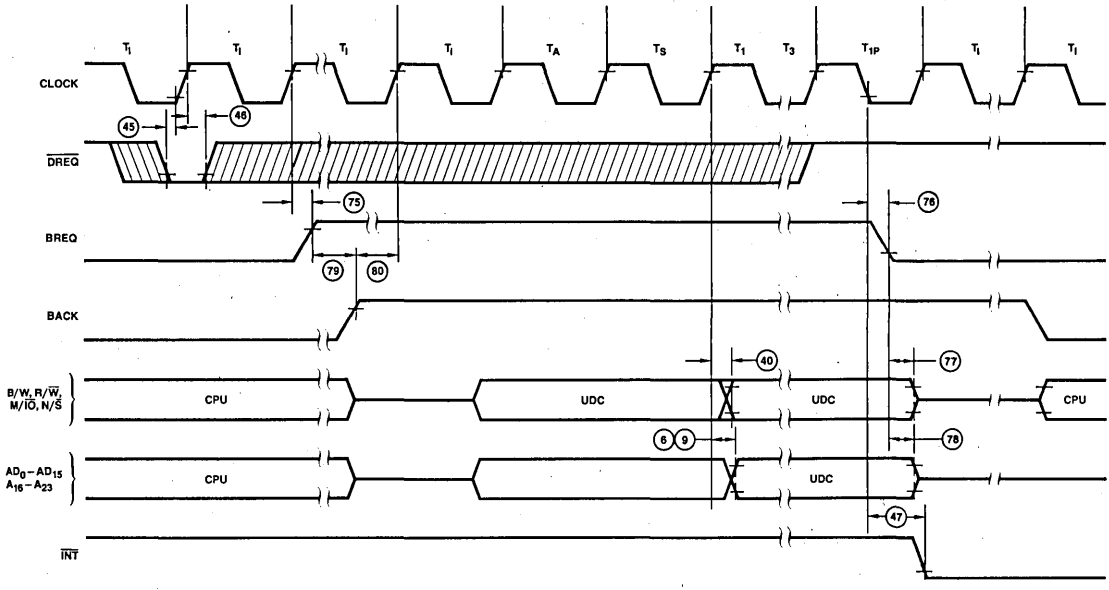
03242B-31

TIMING DIAGRAM 10. Upper Address Rollover Timing



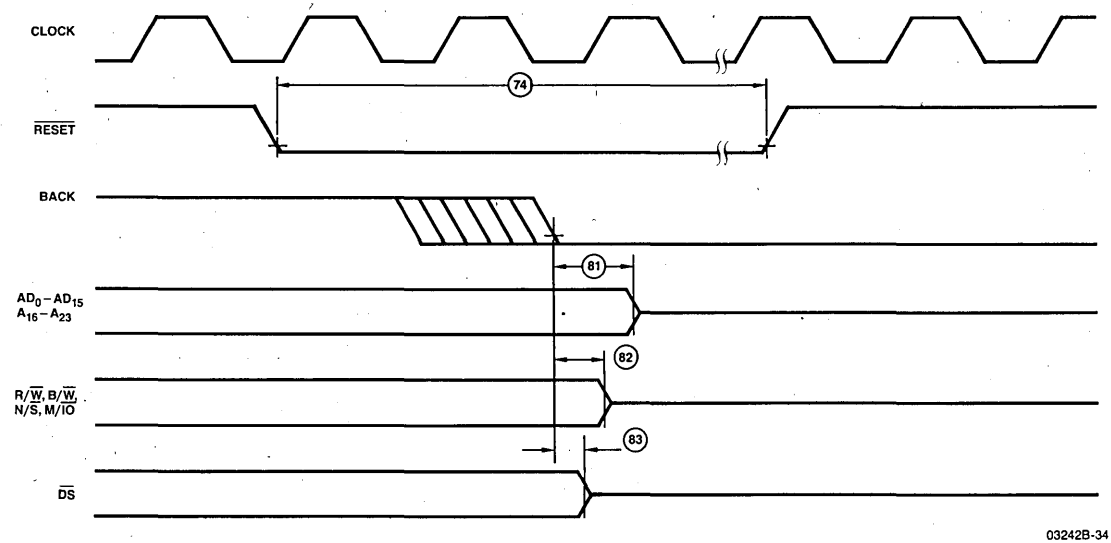
03242B-32

TIMING DIAGRAM 11. Bus Exchange Timing



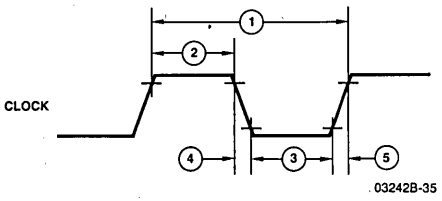
03242B-33

TIMING DIAGRAM 12. Reset Timing



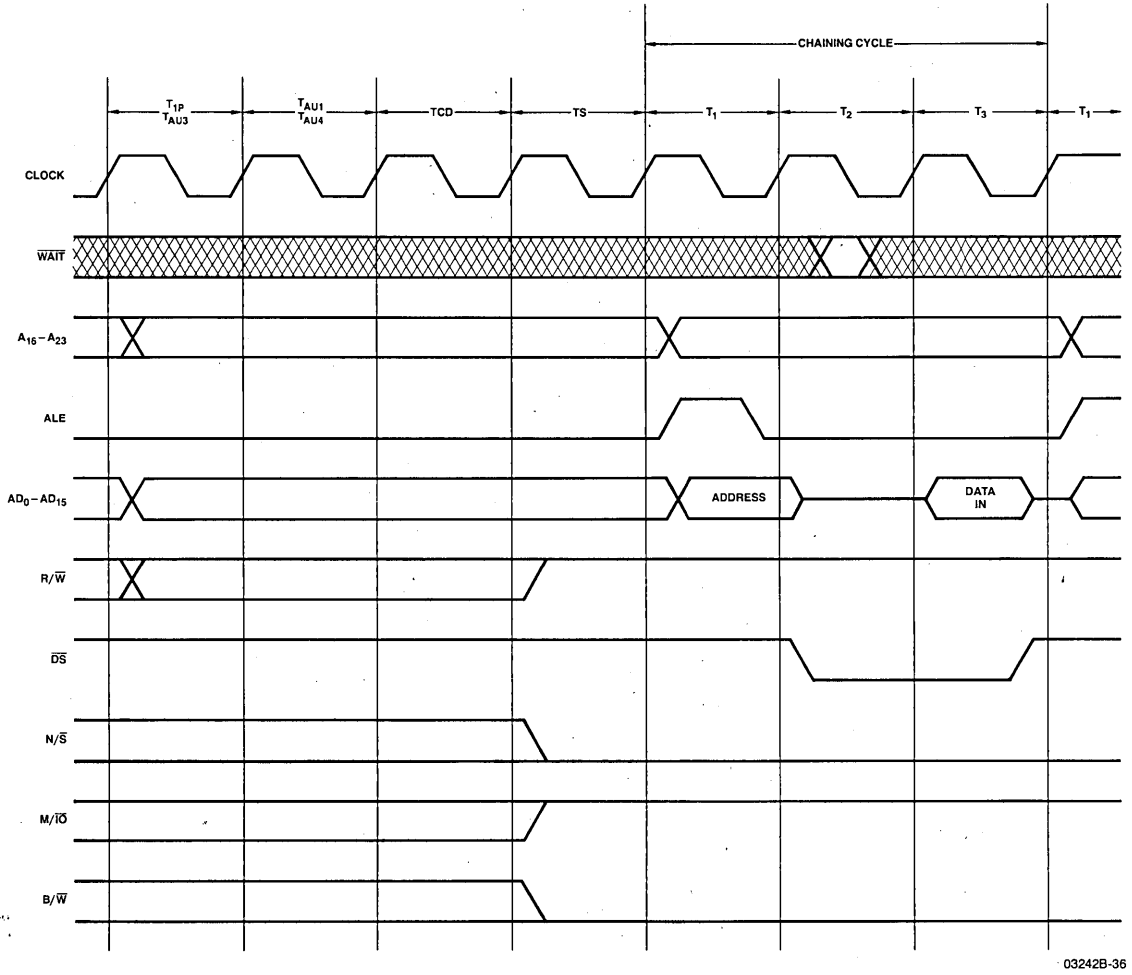
03242B-34

TIMING DIAGRAM 13. Clock Waveform



03242B-35

TIMING DIAGRAM 14. Timing During Chaining

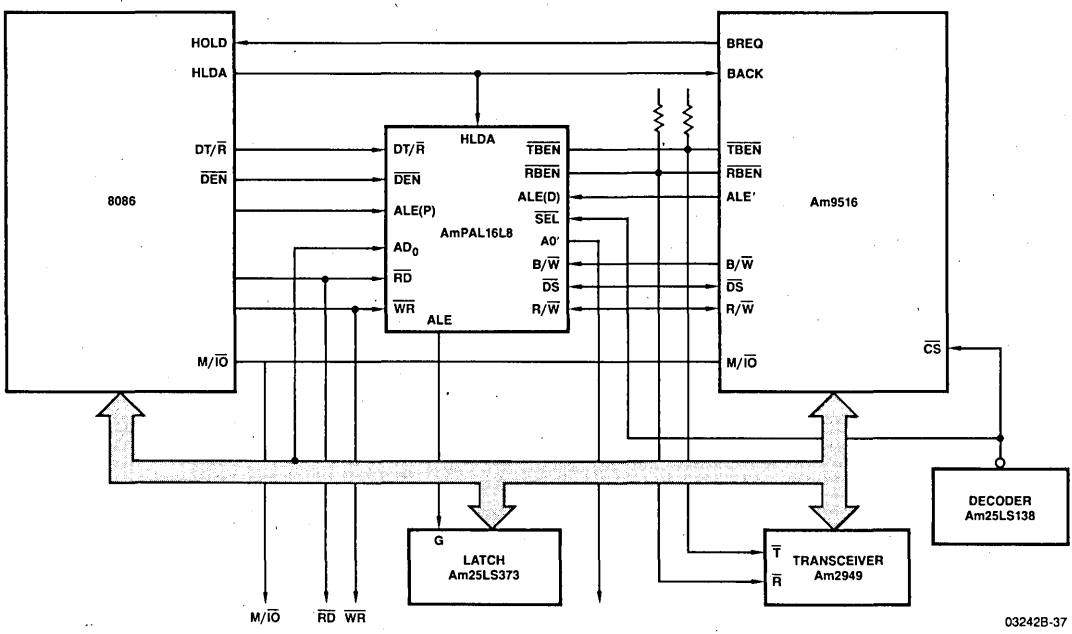


03242B-36

APPLICATION INFORMATION

Figures 21(a) and 21(b) show the configuration of an Am9516 UDC and an Am8086 microprocessor on the same board. Figure 22 shows a configuration for them when the Am9516 UDC is on a different board. The configuration of an Am9516 UDC to 68000 CPU interface is shown in Figure 23. An example of an Am8086 initialization program is shown in Figure 24. Figure 25 shows the reload table for chaining. The details of the Programmable Array Logic (PAL*) for those interfaces are described in Appendix B.

Figure 21(a). Am9516 UDC to 8086 CPU Interface (Minimum Mode)



AmPAL16L8 PALASM FILE

PAL16L8
Pat 001
Am9516 to 8086 min mode interface chip
Advanced Micro Devices

NC ALED ALEP HLDA BW AD_0 DT /DEN /SEL GND
NC /RBEN /RD ALE A_0 /RW /DS /WR /TBEN V_{CC}

If (/HLDA) DS = RD + WR

If (/HLDA) RW = DT

If (/HLDA) TBEN = /DT*/SEL*DEN

If (/HLDA) RBEN = DT*/SEL*DEN

If (HLDA) RD = /RW * DS

If (HLDA) WR = RW * DS

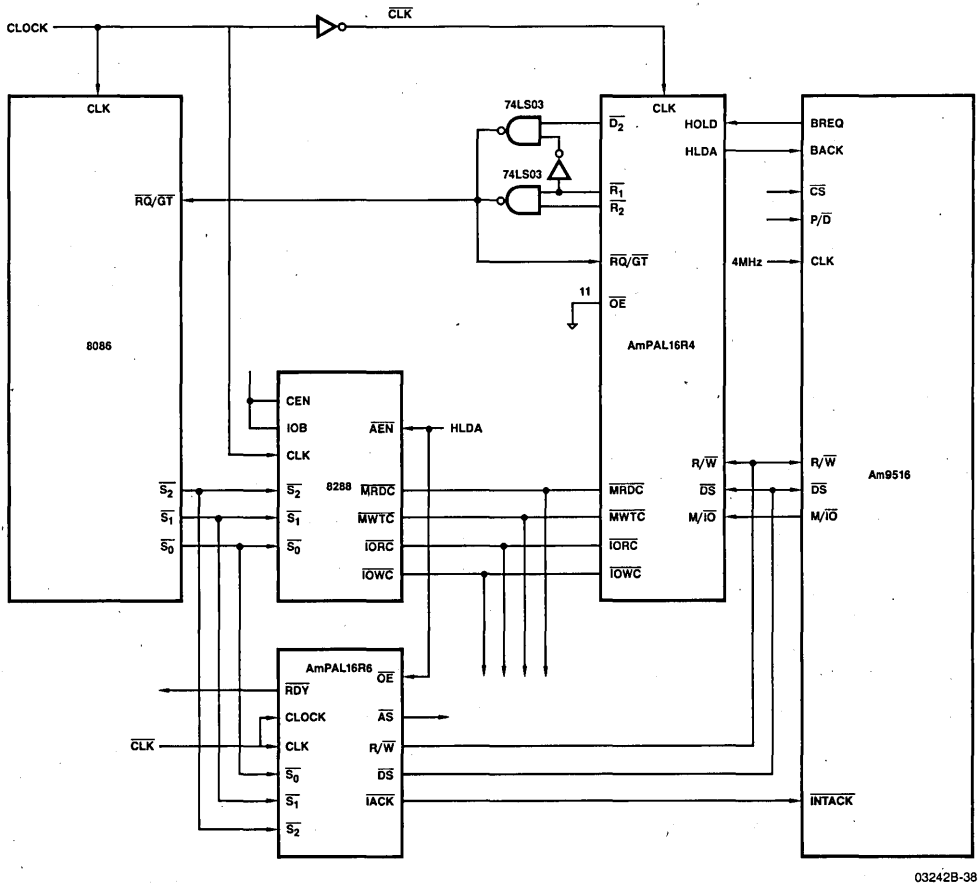
ALE = /ALEP * /ALED

A_0 = /AD_0*/BW*HLDA*ALED +
AD_0*BW*HLDA*ALED +
/AD_0*/HLDA*ALEP + A_0*/ALEP + A_0 */ALED

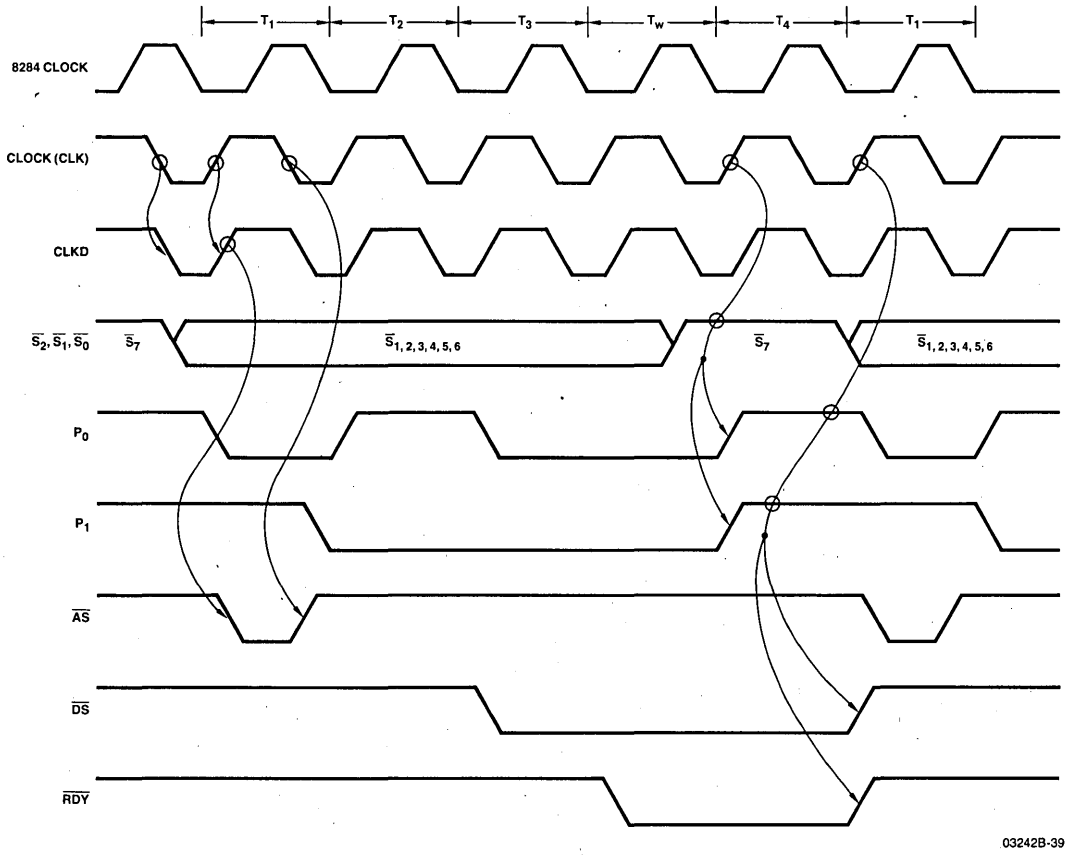
DESCRIPTION

This PAL converts the control signals to interface the 8086 in min mode to the Am9516 DMA controller. Another example shows how this is done in max mode.

Figure 21(b). Am9516 UDC to 8086 CPU Interface (Maximum Mode)



Timing Diagram of AmPAL16R6



AmPAL16R6 PALASM FILE

AmPAL16R6
PAT003
8086 to 85XX Peripheral Interface
Advanced Micro Devices

CLOCK RESET CLK /S_0 /S_1 /S_2 NC NC NC GND
/OE /AS /P_1 /RW /DS /PO /IACK /RDY CLKD V_{CC}

P_0 := /RESET*S_0*/P_0*/P_1 +
/RESET*S_1*/P_0*/P_1 +
/RESET*S_2*/P_0*/P_1 +
/RESET*S_0*P_1 +
/RESET*S_1*P_1 +
/RESET*S_2*P_1

P_1 := /RESET*P_0*/P_1 +
/RESET*P_1*S_0 +
/RESET*P_1*S_1 +
/RESET*P_1*S_2

DS :=/IACK*/P_0*P_1*S_0*/S_1*S_2 +
/IACK*/P_0*P_1*/S_0*S_1*S_2 +
IACK*S_0*S_1*S_2 +
DS*P_0*P_1

RW := S_0*/S_1

IACK := /RESET*S_0*S_1*S_2 + IACK*P_0*P_1*/DS +
IACK*/P_0*/P_1

RDY := /RESET*S_0*/S_1*S_2*P_0*P_1 +
/RESET*/S_0*S_1*S_2*P_0*P_1 +
/RESET*DS*RDY*P_0*P_1

/CLKD = CLK

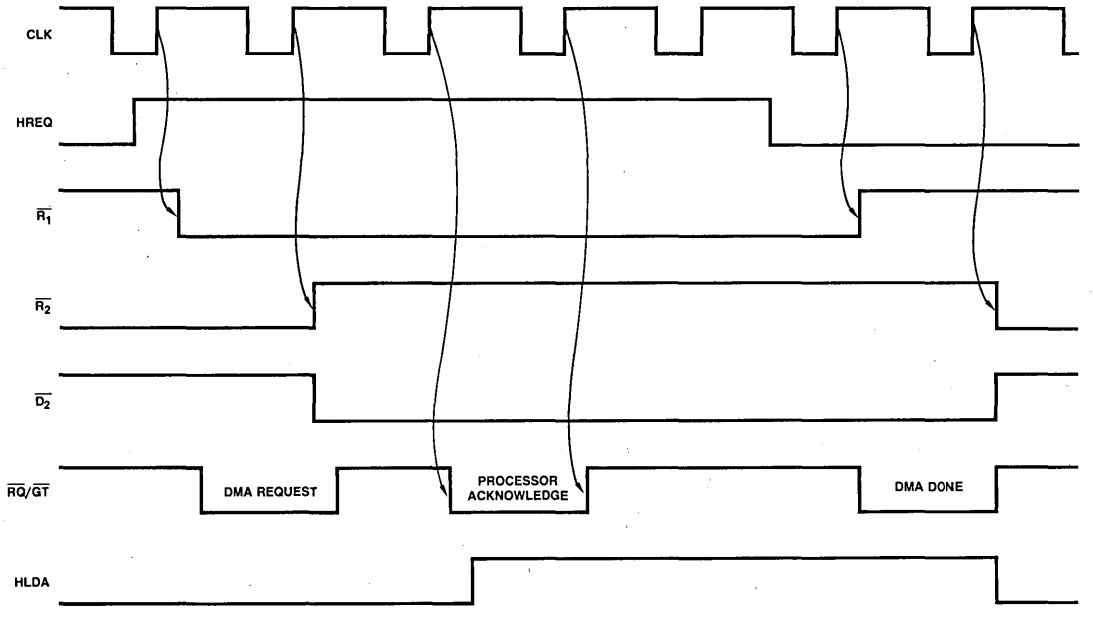
AS = /CLKD*P_0*/P_1*/IACK*CLK

DESCRIPTION

This PAL translates 8086 bus signals into compatible signals for the 9516. It is also applicable to 85XX peripherals by altering /RW and /DS to /RD and /WR. One flip flop is available to give the necessary delay to the falling edge of /WR.

Note: The CLK signal must be externally inverted for this design.
A>

AmPAL16R4 Timing



03242B-40

AmPAL16R4 PALASM FILE

B > Type Am9516 PAL
PAL16R4

8086 to Am9516 interface
Advanced Micro Devices
CLK /RQGT HOLD NC NC NC /RW /DS MIO GND
/OE /MWTC /MRDC HLDA /D_2 /R_2 /R_1 /IOWC /IORC V_{CC}

If (HLDA) IORC = /MIO*DS*/RW
If (HLDA) IOWC = /MIO*DS*RW
If (HLDA) MRDC = MIO*DS*/RW
If (HLDA) MWTC = MIO*DS*RW

R_1 := HOLD

R_2 := /R_1

D_2 := R_1

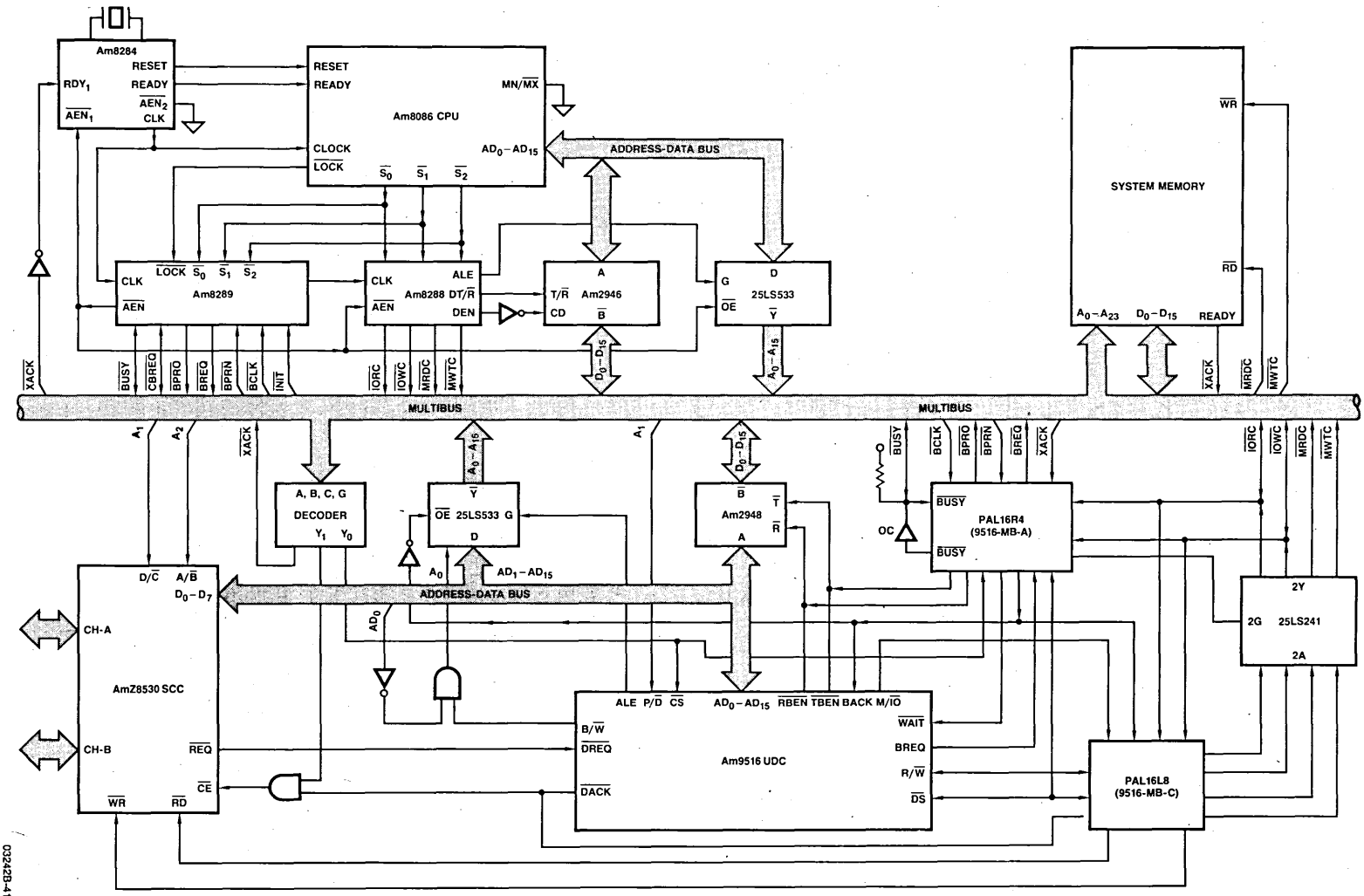
/HLDA := /R_1 + /D_2*/HLDA +/RQGT*/HLDA

DESCRIPTION

This device converts the min mode signals HOLD and HLDA to the max mode /RQGT protocol. Additionally it generates the 8288 equivalent control outputs /MRDC, /MWTC, /IORC, and /IOWC. This PAL was used to connect the Am9516 to the 8086 in max mode.

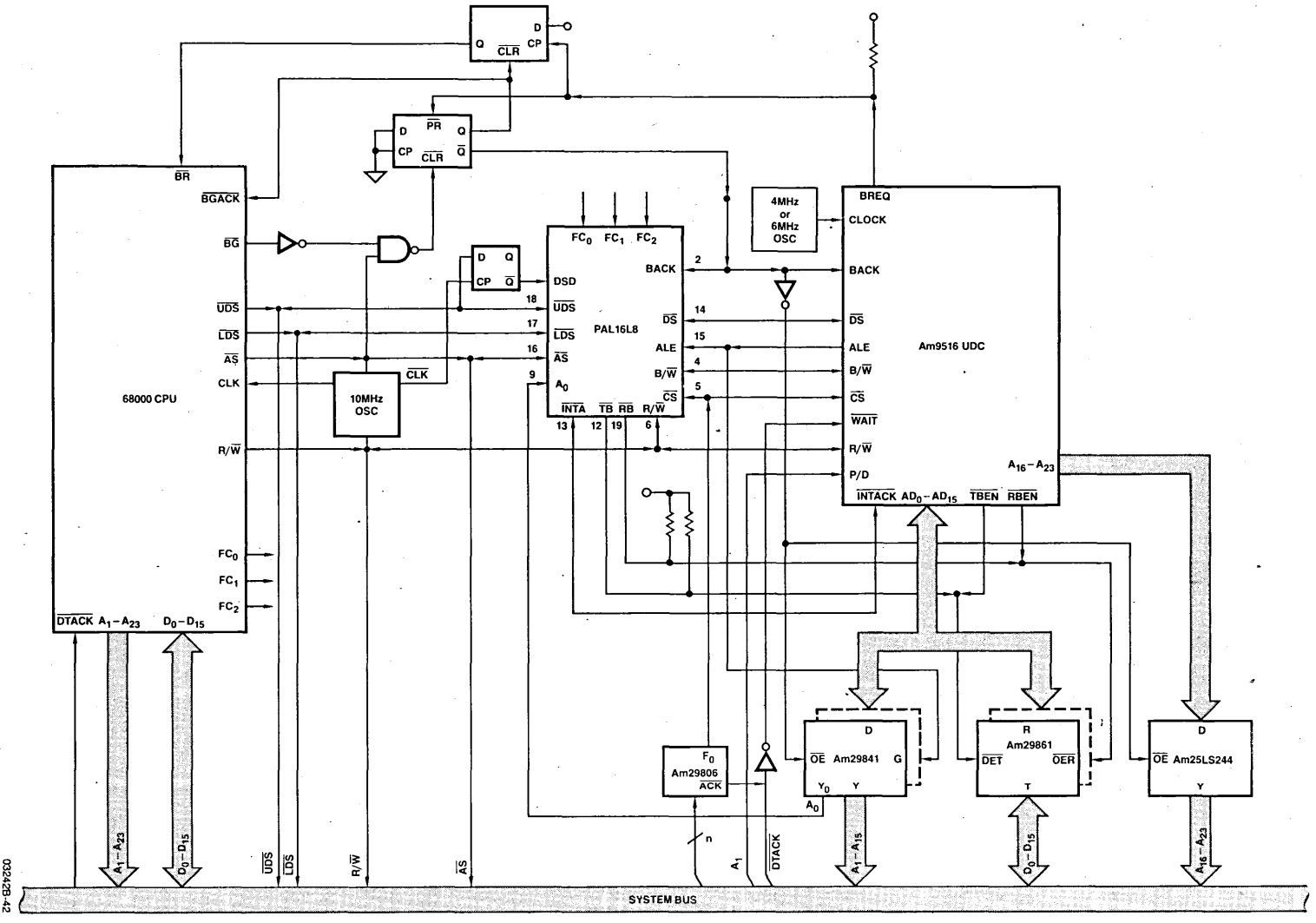
B>

Figure 22. Am9516 UDC to Multibus System Interface Configuration



03242B-41

Figure 23. Am9516 UDC to 68000 CPU Interface Configuration



AmPAL16L8 PALASM FILE

PAL16L8
Pat 001
68000 to Am9516 interface PAL
Advanced Micro Devices

NC BACK NC ALE /CS RW FC_0 FC_1 A_0 GND
FC_2 /TB /INTA /DS /BW /AS /LDS /UDS /RB V_{CC}

If (BACK) UDS = DS*/A_0*BW + DS*/BW

If (BACK) LDS = DS*A_0*BW + DS*/BW

If (BACK) AS = /ALE

If (/BACK) DS = UDS + LDS

If (/BACK) BW = UDS * LDS

If (/BACK) TB = CS*RW*LDS + CS*RW*UDS + TB*DSD*/RW + CS*/RW*UDS

If (/BACK) RB = CS*/RW + RB*DSD

INTA = FC_0*FC_1*FC_2

DESCRIPTION

This PAL provides the necessary bidirectional buffering as well as translating signals as appropriate. The first four equations apply when the Am9516 is bus master the others apply when it is a slave. Note that the /RW terms of TB latch the data to meet the Am9516 data hold time requirement.

Figure 24. Initialization Program for 8086 CPU

```
.                .
.                .
.                .
B0 38        MOV     AL,38H      ;LOADING POINTER OF MASTER
E6 12        OUT     12H         ;MODE REGISTER
B8 07 00     MOV     AX,007H     ;LOADING MMR CODE
E7 10        OUTW    10H         ;
B0 26        MOV     AL,26H      ;LOADING POINTER OF CHAIN
E6 12        OUT     12H         ;ADDRESS REGISTER'S SEGMENT
B8 00 00     MOV     AX,0000H    ;LOADING SEGMENT OF CAR-1
E7 10        OUTW    10H         ;
B0 22        MOV     AL,22H      ;LOADING POINTER OF CHAIN
E6 12        OUT     12H         ;ADDRESS REGISTER'S OFFSET
B8 20 10     MOV     AX,1020H    ;LOADING OFFSET OF CAR-1
E7 10        OUTW    10H         ;
B0 2C        MOV     AL,2CH      ;LOADING POINTER OF COMMAND
E6 12        OUT     12H         ;REGISTER
B0 A0        MOV     AL,A0H      ;LOADING "START CHAIN" COMMAND
E6 10        OUT     10H         ;ISSUING "START CHAIN" COMMAND
.                .
.                .
.                .
```

Notes: The P/$\overline{D}$ input is connected to A1 line; $\overline{CS}$ is decoded from A7 through A4 (all 0).

Figure 25. Reload Table for Chaining

ADDRESS	0	2	4	6	8	A	C	E
1000	0000	1020	0000	1020	0007	0005	0006	0005
1010	0002	AAAA	0009	00A0	0004	0042	0042	0001
1020	03FF	0000	1F00	0000	1060	0010	0000	1F00
1030	0000	1080	0012	0000	FFFF	0001	0000	8020
1040	0000	1020	1111	1111	0000	FFFF	2004	0000
1050	0010	0000	0000	1020	0018	1020	2222	1007
1060	CACA	CACA	CACA	CACA	CACA	CACA	CACA	CACA
1070	CACA	CACA	CACA	CACA	CACA	CACA	CACA	CACA

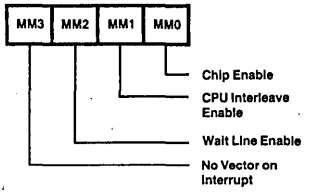
Reload Word

APPENDIX A
UDC REGISTER SUMMARY

Master Mode Register

	A7							A0
Address	X	0	1	1	1	0	0	X

Fast Readable
Writeable

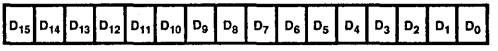


03242B-43

Miscellaneous Registers

	A7							A0		
Address	X	0	1	1	0	0	1	X	Current Operation Count	CH1
	X	0	1	1	0	0	0	X	Current Operation Count	CH2
	X	0	1	1	0	1	1	X	Base Operation Count	CH1
	X	0	1	1	0	1	0	X	Base Operation Count	CH2
	X	1	0	0	1	0	1	X	Pattern	CH1
	X	1	0	0	1	0	0	X	Pattern	CH2
	X	1	0	0	1	1	1	X	Mask	CH1
	X	1	0	0	1	1	0	X	Mask	CH2

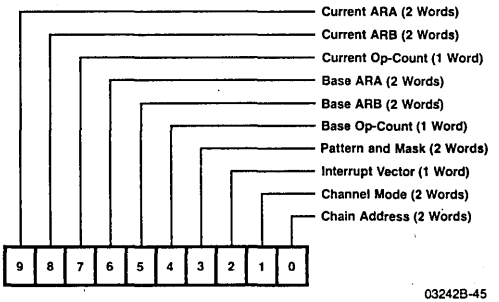
Chain Loadable
Writeable
Pattern and Mask – Slow Readable
Operation Count – Fast Readable



03242B-44

Chain Control Register

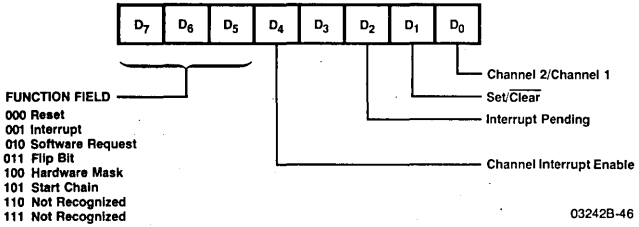
Chain Loadable Only



03242B-45

Command Register

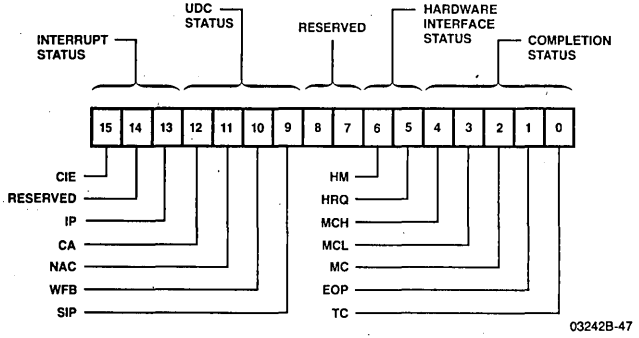
	A7						A0	
Address	X	0	1	0	1	1	1	X	CH1
Writeable Only	X	0	1	0	1	1	0	X	CH2



Status Register

	A7							A0	
Address	X	0	1	0	1	1	1	X	CH1
	X	0	1	0	1	1	0	X	CH2

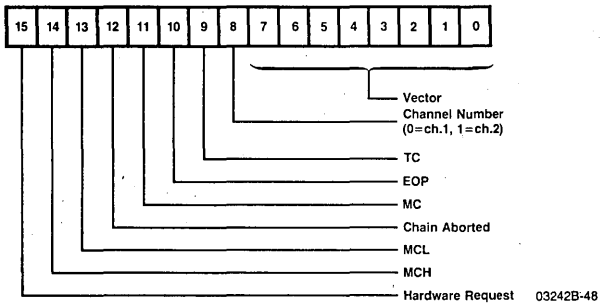
Fast Readable



Interrupt Save Register

	A7							A0	
Address	X	0	1	0	1	0	1	X	CH1
	X	0	1	0	1	0	0	X	CH2

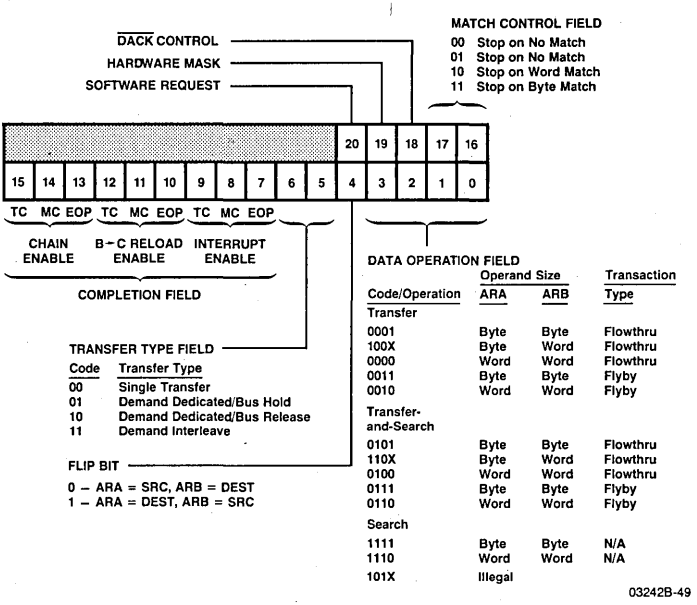
Fast Readable



Channel Mode Register

A7							A0	
X	1	0	1	0	1	1	X	High CH1
X	1	0	1	0	1	0	X	High CH2
X	1	0	1	0	0	1	X	Low CH1
X	1	0	1	0	0	0	X	Low CH2

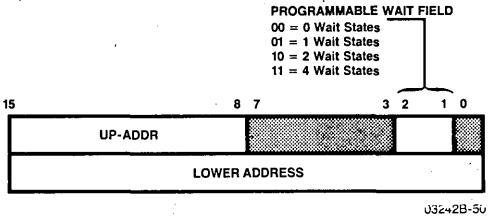
Chain Loadable
Writable (Lower 16 bits)
Slow Readable



03242B-49

Chain Address Register

	A7						A0	
Address	X	0	1	0	0	1	1	X	Up-Addr CH1
	X	0	1	0	0	1	0	X	Up-Addr CH2
	X	0	1	0	0	0	1	X	Low-Addr CH1
	X	0	1	0	0	0	0	X	Low-Addr CH2

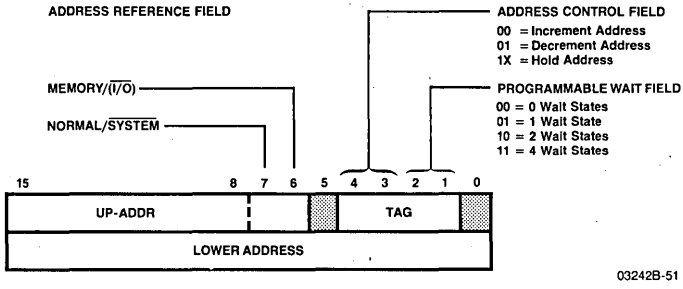


03242B-50

Address Registers

	A7							A0		
Address	X	0	0	1	1	0	1	X	Current ARA Up-Addr/Tag	CH1
	X	0	0	1	1	0	0	X	Current ARA Up-Addr/Tag	CH2
	X	0	0	0	1	0	1	X	Current ARA Low-Addr	CH1
	X	0	0	0	1	0	0	X	Current ARA Low-Addr	CH2
	X	0	0	1	0	0	1	X	Current ARB Up-Addr/Tag	CH1
	X	0	0	1	0	0	0	X	Current ARB Up-Addr/Tag	CH2
	X	0	0	0	0	0	1	X	Current ARB Low-Addr	CH1
	X	0	0	0	0	0	0	X	Current ARB Low-Addr	CH2
	X	0	0	1	1	1	1	X	Base ARA Up-Addr/Tag	CH1
	X	0	0	1	1	1	0	X	Base ARA Up-Addr/Tag	CH2
	X	0	0	0	1	1	1	X	Base ARA Low-Addr	CH1
	X	0	0	0	1	1	0	X	Base ARA Low-Addr	CH2
	X	0	0	1	0	1	1	X	Base ARB Up-Addr/Tag	CH1
	X	0	0	1	0	1	0	X	Base ARB Up-Addr/Tag	CH2
	X	0	0	0	0	1	1	X	Base ARB Low-Addr	CH1
	X	0	0	0	0	1	0	X	Base ARB Low-Addr	CH2

Chain Loadable
Fast Readable and Writeable



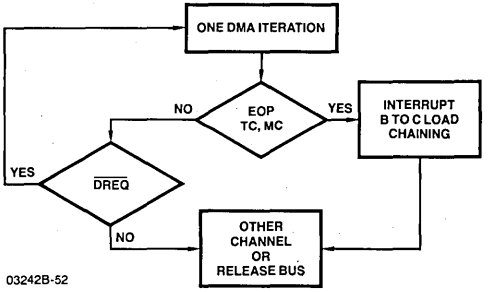
APPENDIX B

(1) Flow Charts of DMA Operations:
Figure B-1 shows the basic DMA operations with software or hardware request. The Demand Interleave operations are shown in Figure B-2.

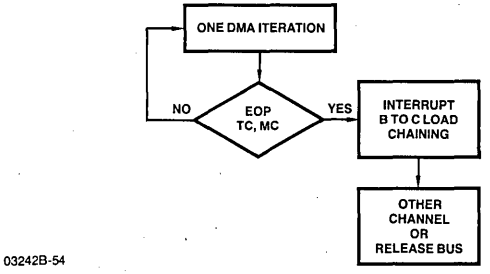
(2) Designs of PAL for UDC Interface:
Figure B-3 is the PAL for 6800 CPU to Am9516 UDC interface. Figure B-4 and B-5 are for the interface between UDC and Multibus**.

Figure B-1. Basic DMA Operations of Am9516 UDC

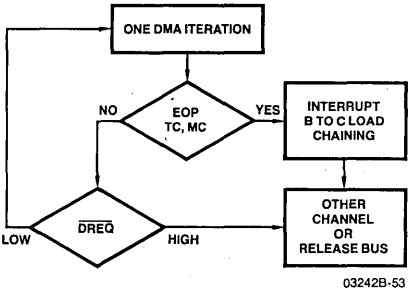
a) Single Operation



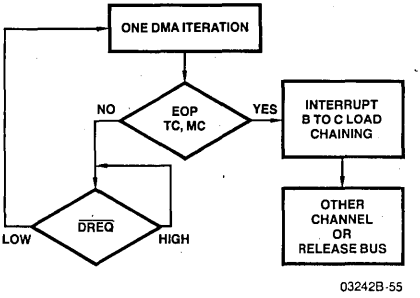
b) Demand Operation when Software Requesting



c) Demand Dedicated with Bus Release (Hardware Request)

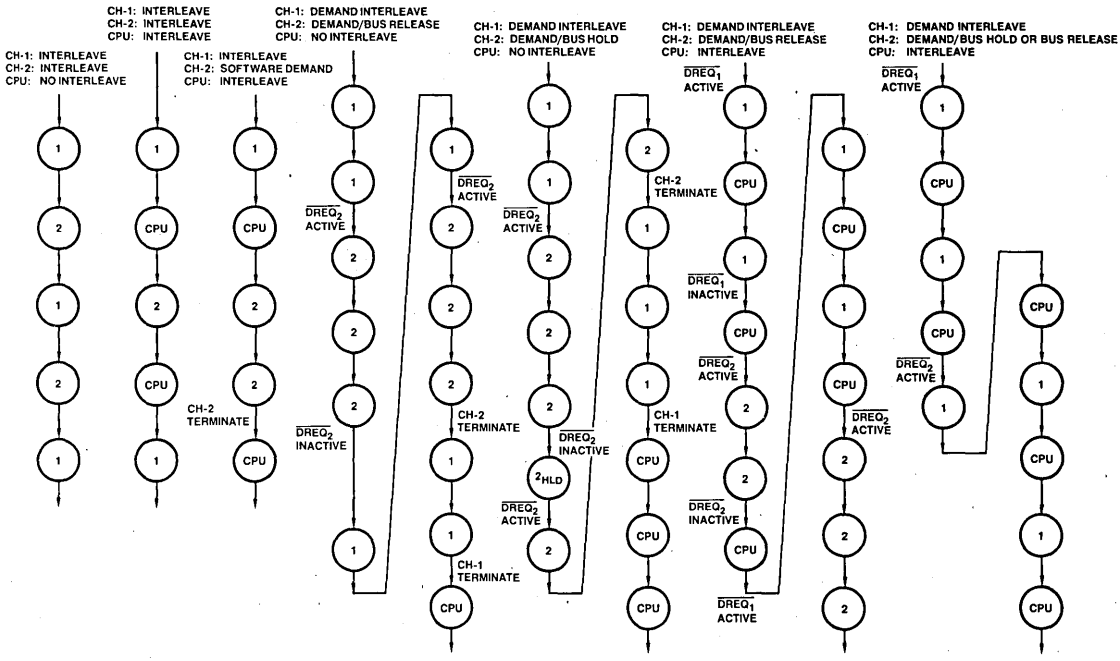


d) Demand Dedicated with Bus Hold (Hardware Request)



**Multibus is a trademark of Intel Corp.

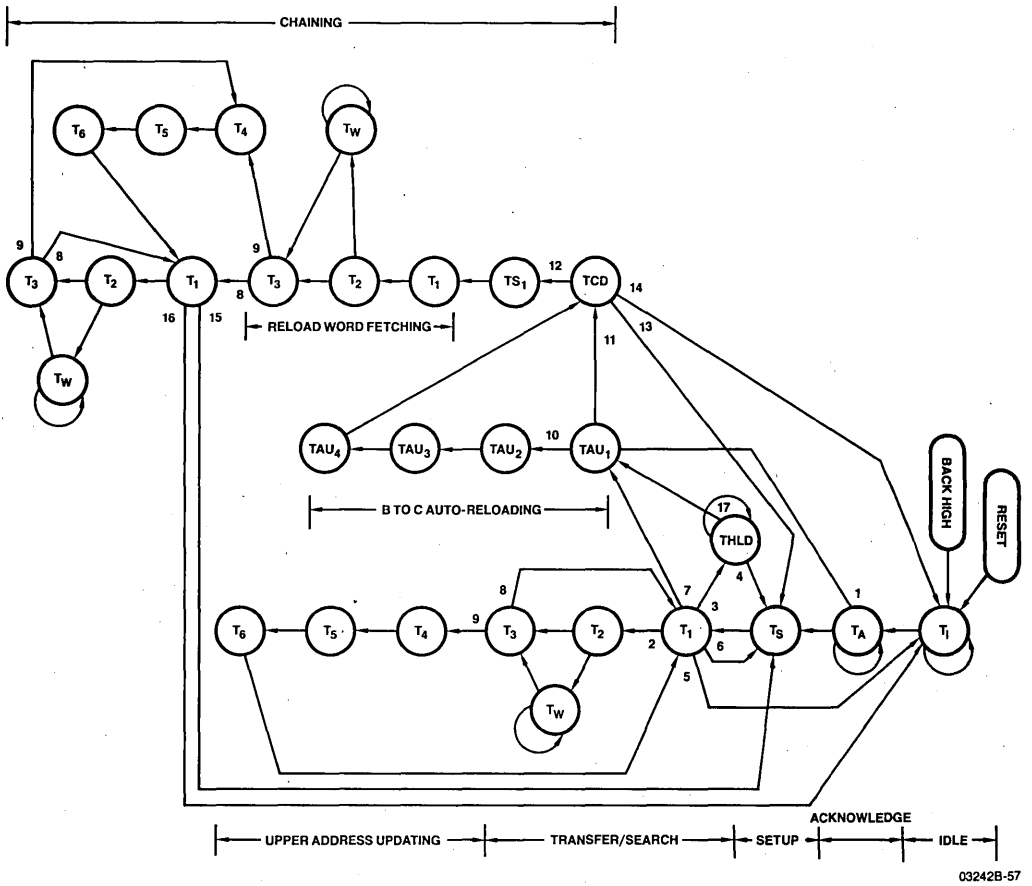
Figure B-2. Demand Interleave Operations of Am9516 UDC



03242B-56

APPENDIX C

Am9516 STATE DIAGRAM



Am9516 INTERNAL OPERATION ROUTINES

1. "Start Chain" command issued or start updating routine* after an interrupt has been served.**
2. Normal DMA operation.
3. Demand with Bus hold while DREQ is inactive.
4. DREQ is active while bus held.
5. Single transfer, CPU interleave enabled, or demand with bus release while current DREQ is inactive and no DMA request is pending.
6. Single Transfer or Demand/Bus release while current DREQ is inactive, but the other DMA request is pending.
7. TC, MC or EOP termination occurs.
8. One DMA or chain transaction is done and the upper address is not changed.
9. One DMA or chain transaction is done and the upper address is changed.
10. Base-to-current auto-reloading is enabled.
11. Base-to-current auto-reloading is disabled.
12. Chaining is enabled.
13. Chaining is disabled and another DMA request is pending.
14. Chaining is disabled and no DMA request is pending.
15. Chaining ends and another DMA request is pending.
16. Chaining ends and no DMA request is pending.
17. EOP termination of Bus Hold.

*Updating routine includes base-to-current auto-reloading and chaining.

**When a second interrupt is to be issued before the first interrupt is acknowledged, the SIP bit of a Status Register is set and the channel relinquishes the bus until the first interrupt has been served. If the channel was to perform the updating routine, once the SIP bit is cleared, DTC will re-acquire the bus and perform the appropriate operation (i.e., 1).

Appendix D.

SWITCHING CHARACTERISTICS

TIMING FOR UDC AS BUS MASTER

(See TIMING DIAGRAMS starting on page 2-240)

Number	Parameter	Description	4MHz		5MHz		Units
			Min	Max	Min	Max	
1	TcC	Clock Cycle Time	250	2000	200		ns
2	TwCh	Clock Width (HIGH)	105		69*		ns
3	TwCl	Clock Width (LOW)	105		85*		ns
4	TfC	Clock Fall Time		20		10 or 15	ns
5	TrC	Clock Rise Time		20		10 or 15	ns
6	TdC(AUv)	Clock RE to Upper Address (A_{16}-A_{23}) Valid Delay		90		85	ns
7	ThC(AUv)	Clock RE to Upper Address Valid Hold Time	20		10		ns
8	TdC(ST)	Clock RE to R/$\overline{W}$ and B/$\overline{W}$ Valid Delay		110		100	ns
9	TdC(A)	Clock RE to Lower Address (A_0-A_{15}) Valid Delay		90		85	ns
10	TdC(Az)	Clock RE to Lower Address (A_0-A_{15}) Float Delay		60		60	ns
11	TdC(ALr)	Clock RE to ALE RE Delay		70		70	ns
12	TdC(AL)	Clock FE to ALE FE Delay		70		70	ns
13	TdC(DS)	Clock RE to $\overline{DS}$ (Read) FE Delay		60		60	ns
14	TdC(DSf)	Clock FE to $\overline{DS}$ (Write) FE Delay		60		60	ns
15	TdC(DSr)	Clock FE to $\overline{DS}$ RE Delay		60		60	ns
16	TdC(DO)	Clock RE to Data Out Valid Delay		90		85	ns
17	TsDI(C)	Data In to Clock FE Setup Time	20		20		ns
18	TdA(AL)	Address Valid to ALE FE Delay	50		40		ns
19	ThAL(A)	ALE FE to Lower Address Valid Hold Time	60		50		ns
20	TwAL	ALE Width (HIGH)	80		twCh-15		ns
21	TdAz(DS)	Lower Address Float to $\overline{DS}$ FE Delay	0		0		ns
22	TdAL(DS)	ALE FE to $\overline{DS}$ (Read) FE Delay	75		60		ns
23	TdAL(DI)	ALE FE to Data In Required Valid Delay		300		250	ns
24	TdA(DI)	Address Valid to Data In Required Valid Delay		410		350	ns
25	TdDS(A)	$\overline{DS}$ RE to Address Active Delay	80		60		ns
26	TdDS(AL)	$\overline{DS}$ RE to ALE RE Delay	75		60		ns
27	TdA(DS)	Address Valid to $\overline{DS}$ (Read) FE Delay	160		150		ns
28	TdDO(DSr)	Data Out Valid to $\overline{DS}$ RE Delay	230		215		ns
29	TdDO(DSf)	Data Out Valid to $\overline{DS}$ FE Delay	55		45		ns
30	ThDS(DO)	$\overline{DS}$ RE to Data Out Valid Hold Time	85		65		ns
31	TdDS(DI)	$\overline{DS}$ (Read) FE to Data In Required Valid Delay		205		180	ns
33	ThDI(DS)	$\overline{DS}$ RE to Data In Hold Time	0		0		ns
34	TwDSmw	$\overline{DS}$ (Write) Width (LOW)	185		150		ns
35	TwDSmr	$\overline{DS}$ (Read) Width (LOW)	275		250		ns
36	TdC(RBr)	Clock FE to $\overline{RBEN}$ RE Delay*		70		65	ns
37	ThDS(ST)	$\overline{DS}$ RE to B/$\overline{W}$, N/S, R/$\overline{W}$ and M/$\overline{IO}$ Valid Hold Time	75		60		ns
38	TdC(TRf)	Clock RE to $\overline{TBEN}$ or $\overline{RBEN}$ FE Delay		60		60	ns
39	TdC(TRr)	Clock RE to $\overline{TBEN}$ RE Delay		60		60	ns
40	TdC(ST)	Clock RE to M/$\overline{IO}$ and N/$\overline{S}$ Valid Delay	90			80	ns
41	TdS(AL)	R/$\overline{W}$, M/$\overline{IO}$, B/$\overline{W}$ and N/$\overline{S}$ Valid to ALE FE Delay	60		45		ns
42	TsWT(C)	$\overline{WAIT}$ to Clock FE Setup Time	20		20		ns

*These must not occur simultaneously.

WITCHING CHARACTERISTICS
MING FOR UDC AS BUS MASTER (Cont.)

umber	Parameter	Description	4MHz Min	4MHz Max	5MHz Min	5MHz Max	Units
43	ThWT(C)	$\overline{WAIT}$ to Clock FE Hold Time	20		15		ns
44	TwDRQ	$\overline{DREQ}$ Pulse Width (Single Transfer Mode)	20		20		ns
45	TsDRQ(C)	$\overline{DREQ}$ Valid to Clock RE Setup Time	50		50		ns
46	ThDRQ(C)	Clock RE to $\overline{DREQ}$ Valid Hold Time	20		20		ns
47	TdC(INTf)	Clock FE to $\overline{INT}$ FE Delay		150		150	ns

his parameter is slower than parameter 15.

ote: RE = rising edge
FE = falling edge

WITCHING CHARACTERISTICS
IDC AS BUS SLAVE BUS EXCHANGE

lumber	Parameter	Description	4MHz Min	4MHz Max	5MHz Min	5MHz Max	Units
61	TdIN(DO)	$\overline{INTACK}$ FE to Data Output Valid Delay		135		135	ns
62	TdIN(DOz)	$\overline{INTACK}$ RE to Data Output Float Delay		75		75	ns
63	TdDS(DO)	$\overline{DS}$ FE (IOR) to Data Output Driven Delay		135*		135*	ns
64	TdDS(DOz)	$\overline{DS}$ RE (IOR) to Data Output Float Delay		75		75	ns
65	TsDI(DS)	Data Valid to $\overline{DS}$ RE (IOW) Setup Time	40		40		ns
66	ThDS(DI)	$\overline{DS}$ RE (IOW) to Data Valid Hold Time	30		30		ns
67	TwDS	$\overline{DS}$ Low Width	150*		150*		ns
68	TwIN	$\overline{INTACK}$ Low Width	150		150		ns
69	ThDS(CS)	$\overline{DS}$ RE to $\overline{CS}$ Valid Hold Time	20		20		ns
70	ThDS(PD)	$\overline{DS}$ RE to P/$\overline{D}$ Valid Hold Time	20		20		ns
71	TsPD(DS)	P/$\overline{D}$ Valid to $\overline{DS}$ FE Setup Time (IOR)	10		10		ns
		P/$\overline{D}$ Valid to $\overline{DS}$ FE Setup Time (IOW)	50		50		
72	TsCS(DS)	$\overline{CS}$ Valid to $\overline{DS}$ FE Setup Time	30		30		ns
73	TrDS	$\overline{DS}$ RE to $\overline{DS}$ FE Recovery Time (for Commands Only)	4TcC		4TcC		ns
74	TwRST	$\overline{RESET}$ Low Width	3TcC		3TcC		ns
75	TdC(BRQf)	Clock RE to BREQ RE Delay		150		150	ns
76	TdC(BRQr)	Clock FE to BREQ FE Delay		150		150	ns
77	TdBRQ(CTRz)	BREQ FE to Control Bus Float Delay		140		140	ns
78	TdBRQ(ADz)	BREQ FE to AD Bus Float Delay		140		140	ns
79	TdBRQ(BAK)	BREQ RE to BACK RE Required Delay	0		0		ns
80	TsBAK(C)	BACK Valid to Clock RE Setup Time	40		35		ns
81	TdBAK(ADz)	BACK FE to A and AD Buses Float Delay (Reset)		135		135	ns
82	TdBAK(CTRz)	BACK FE to Control Bus Float Delay (Reset)		100		100	ns
83	TdBAK(DSz)	BACK FE to $\overline{DS}$ Float Delay (Reset)		80		80	ns
84	TsRW(DS)	R/$\overline{W}$ Valid to $\overline{DS}$ FE Setup Time (IOW)	2		2		ns
85	ThDS(RW)	$\overline{DS}$ RE to R/$\overline{W}$ Valid Hold Time (IOW)	−10		−10		ns
86	TsRW(DS)	R/$\overline{W}$ Valid to $\overline{DS}$ FE Setup Time (IOR)	20		20		ns
87	ThDS(RW)	$\overline{DS}$ RE to R/$\overline{W}$ Valid Hold Time (IOR)	20		20		ns

*2000ns for slow readable registers (worst case)

Note: RE = rising edge
FE = falling edge

SWITCHING CHARACTERISTICS

UDC-PERIPHERAL INTERFACE

Number	Parameter	Description	4MHz Min	4MHz Max	5MHz Min	5MHz Max	Units
90	TCHDL	Clock RE to Pulsed $\overline{DACK}$ FE Delay (Flyby Transactions Only)		100		90	ns
91	TCHDH	Clock RE to Pulsed $\overline{DACK}$ RE Delay (To Flyby Transactions Only)		100		90	ns
92	TDSK	$\overline{DS}$ RE to Pulsed $\overline{DACK}$ RE Delay (FROM Flyby Transactions Only)	30		30		ns
93	TDAD	Clock RE to Level $\overline{DACK}$ Valid Delay		100		90	ns
94	TDAH	Clock FE to Level $\overline{DACK}$ Valid Hold Time		100		90	ns
95	TEIDL	Clock FE to Internal $\overline{EOP}$ LOW Delay		100		85	ns
96	TEIDH	Clock FE to Internal $\overline{EOP}$ RE Delay		100		85	ns
97	TES	External $\overline{EOP}$ Valid to Clock FE Setup Time During Operation	10		10		ns
98	TEW	External $\overline{EOP}$ Pulse Width Required During Operation	20		20		ns
99	TES(BH)	External $\overline{EOP}$ Valid to Clock FE Setup Time During Bus Hold	10		10		ns
100	TEW(BH)	External $\overline{EOP}$ Pulse Width Required During Bus Hold	20		20		ns

Note: RE = rising edge
FE = falling edge

Am9517A

Multimode DMA Controller

DISTINCTIVE CHARACTERISTICS

- Four independent DMA channels, each with separate registers for Mode Control, Current Address, Base Address, Current Word Count and Base Word Count.
- Transfer modes: Block, Demand, Single Word, Cascade
- Independent autoinitialization of all channels
- Memory-to-memory transfers
- Memory block initialization
- Address increment or decrement
- Master system disable
- Enable/disable control of individual DMA requests
- Directly expandable to any number of channels
- End of Process input for terminating transfers
- Software DMA requests
- Independent polarity control for DREQ and DACK signals
- Compressed timing option speeds transfers -- up to 2.5M bytes/second
- +5 volt power supply
- Advanced N-channel silicon gate MOS technology
- 40 pin Hermetic DIP package
- New 9517A-5 5MHz version for higher speed CPU compatability

GENERAL DESCRIPTION

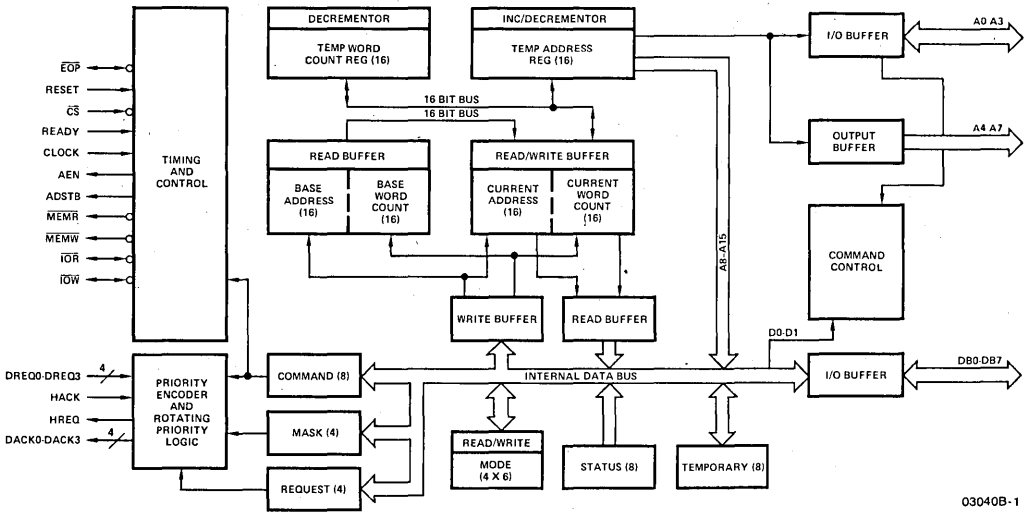
The Am9517A Multimode Direct Memory Access (DMA) Controller is a peripheral interface circuit for microprocessor systems. It is designed to improve system performance by allowing external devices to directly transfer information to or from the system memory. Memory-to-memory transfer capability is also provided. The Am9517A offers a wide variety of programmable control features to enhance data throughput and system optimization and to allow dynamic reconfiguration under program control.

The Am9517A is designed to be used in conjunction with an external 8-bit address register such as the Am74LS373. It contains four independent channels and may be expanded to any number of channels by cascading additional controller chips.

The three basic transfer modes allow programmability of the types of DMA service by the user. Each channel can be individually programmed to Autoinitialize to its original condition following an End of Process ($\overline{EOP}$).

Each channel has a full 64K address and word count capability. An external $\overline{EOP}$ signal can terminate a DMA or memory-to-memory transfer. This is useful for block search or compare operations using external comparators or for intelligent peripherals to abort erroneous services.

BLOCK DIAGRAM



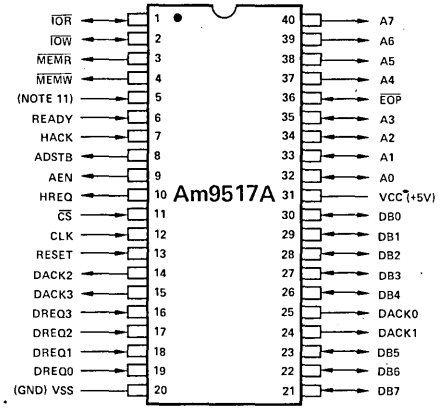
03040B-1

ORDERING INFORMATION

Package Type	Ambient Temperature	Maximum Clock Frequency		
		3MHz	4MHz	5MHz
Hermetic DIP/ Molded DIP	$0°C \leq T_A \leq +70°C$	AM9517ADC/PC AM9517A-1DC/PC	AM9517A-4DC/PC	AM9517A-5DC/PC
Hermetic DIP	$-40°C \leq T_A \leq +85°C$	AM9517ADI AM9517A-1DI	AM9517A-4DI/PI	
Hermetic DIP	$-55°C \leq T_A \leq +125°C$	AM9517ADMB		

03040B-MMP

Figure 1. Connection Diagram
D-40, P-40



Top View
Pin 1 is marked for orientation.

INTERFACE SIGNAL DESCRIPTION

VCC: +5 Volt Supply
VSS: Ground

CLK (Clock, Input)

This input controls the internal operations of the Am9517A and its rate of data transfers. The input may be driven at up to 3MHz for the standard Am9517A, up to 4MHz for the Am9517A-4, and up to 5MHz for the Am9517A-5.

$\overline{CS}$ (Chip Select, Input)

Chip Select is an active low input used to select the Am9517A as an I/O device during an I/O Read or I/O Write by the host CPU. This allows CPU communication on the data bus. During multiple transfers to or from the Am9517A by the host CPU, $\overline{CS}$ may be held low providing $\overline{IOR}$ or $\overline{IOW}$ is toggled following each transfer.

RESET (Reset, Input)

Reset is an asynchronous active high input which clears the Command, Status, Request and Temporary registers. It also clears the First/Last Flip/Flop and sets the Mask register. Following a Reset the device is in the Idle cycle.

READY (Ready, Input)

Ready is an input used to extend the memory read and write pulses from the Am9517A to accommodate slow memories or I/O peripheral devices.

HACK (Hold Acknowledge, Input)

The active high Hold Acknowledge from the CPU indicates that control of the system buses has been relinquished.

DREQ0-DREQ3 (DMA Request, Input)

The DMA Request lines are individual asynchronous channel request inputs used by peripheral circuits to obtain DMA service. In Fixed Priority, DREQ0 has the highest priority and DREQ3 has the lowest priority. Polarity of DREQ is programmable. Reset initializes these lines to active high.

DB0-DB7 (Data Bus, Input/Output)

The Data Bus lines are bidirectional three-state signals connected to the system data bus. The outputs are enabled during the I/O Read by the host CPU, permitting the CPU to examine the contents of an Address register, the Status register, the Temporary register or a Word Count register. The Data Bus is enabled to input data during a host CPU I/O write, allowing the CPU to program the Am9517A control registers. During DMA cycles the most significant eight bits of the address are output onto the data bus to be strobed into an external latch by ADSTB. In memory-to-memory operations data from the source memory location comes into the Am9517A's Temporary register on the read-from-memory half of the operation. On the write-to-memory half of the operation, the data bus outputs the Temporary register data into the destination memory location.

$\overline{IOR}$ (I/O Read, Input/Output)

I/O Read is a bidirectional active low three-state line. In the Idle cycle, it is an input control signal used by the CPU to read the control registers. In the Active cycle, it is an output control signal used by the Am9517A to access data from a peripheral during a DMA Write transfer.

$\overline{IOW}$ (I/O Write, Input/Output)

I/O Write is a bidirectional active low three-state line. In the Idle cycle it is an input control signal used by the CPU to load information into the Am9517A. In the Active cycle it is an output control signal used by the Am9517A to load data to the peripheral during a DMA Read transfer.

Write operations by the CPU to the Am9517A require a rising $\overline{IOW}$ edge following each data byte transfer. It is not sufficient to hold the $\overline{IOW}$ pin low and toggle $\overline{CS}$.

$\overline{EOP}$ (End of Process, Input/Output)

$\overline{EOP}$ is an active low bidirectional open-drain signal providing information concerning the completion of DMA service. When a channel's Word Count goes to zero, the Am9517A pulses $\overline{EOP}$ low to provide the peripheral with a completion signal. $\overline{EOP}$ may also be pulled low by the peripheral to cause premature completion. The reception of $\overline{EOP}$, either internal or external, causes the currently active channel to terminate the service, to set its TC bit in the Status register and to reset its request bit. If Autoinitialization is selected for the channel, the current registers will be updated from the base registers. Otherwise the channel's mask bit will be set and the register contents will remain unaltered.

During memory-to-memory transfers, $\overline{EOP}$ will be output when the TC for channel 1 occurs. $\overline{EOP}$ always applies to the channel with an active DACK; external $\overline{EOP}$s are disregarded when DACK0-DACK3 are all inactive if the DMA is in state SI.

In situations where two or more Am9517A DMAs are cascaded, the $\overline{EOP}$ pins should be logically OR'ed (not wire-OR'ed).

Because $\overline{EOP}$ is an open-drain signal, an external pullup resistor is required. Values of 3.3K or 4.7K are recommended; the $\overline{EOP}$ pin cannot sink the current passed by a 1K pullup.

A0-A3 (Address, Input/Output)

The four least significant address lines are bidirectional 3-state signals. During DMA Idle cycles they are inputs and allow the host CPU to load or read control registers. When the DMA is active, they are outputs and provide the lower 4-bits of the output address.

A4-A7 (Address, Output)

The four most significant address lines are three-state outputs and provide four bits of address. These lines are enabled only during DMA service.

HREQ (Hold Request, Output)

The Hold Request to the CPU is used by the DMA to request control of the system bus. Software requests or unmasked DREQs cause the Am9517A to issue HREQ.

DACK0-DACK3 (DMA Acknowledge, Output)

The DMA Acknowledge lines indicate that a channel is active. In many systems they will be used to select a peripheral. Only one DACK will be active at a time and none will be active unless the DMA is in control of the bus. The polarity of these lines is programmable. Reset initializes them to active-low.

AEN (Address Enable, Output)

Address Enable is an active high signal used to disable the system bus during DMA cycles to enable the output of the external latch which holds the upper byte of the address. Note that during DMA transfers HACK and AEN should be used to deselect all other I/O peripherals which may erroneously be accessed as programmed I/O during the DMA operation. The Am9517A automatically deselects itself by disabling the $\overline{CS}$ input during DMA transfers.

ADSTB (Address Strobe, Output)

The active high Address Strobe is used to strobe the upper address byte from DB0-DB7 into an external latch.

Figure 2. Am9517A Internal Registers.

Name	Size	Number
Base Address Registers	16 bits	4
Base Word Count Registers	16 bits	4
Current Address Registers	16 bits	4
Current Word Count Registers	16 bits	4
Temporary Address Register	16 bits	1
Temporary Word Count Register	16 bits	1
Status Register	8 bits	1
Command Register	8 bits	1
Temporary Register	8 bits	1
Mode Registers	6 bits	4
Mask Register	4 bits	1
Request Register	4 bits	1

$\overline{MEMR}$ (Memory Read, Output)

The Memory Read signal is an active low three-state output used to access data from the selected memory location during a memory-to-peripheral or a memory-to-memory transfer.

$\overline{MEMW}$ (Memory Write, Output)

The Memory Write signal is an active low three-state output used to write data to the selected memory location during a peripheral-to-memory or a memory-to-memory transfer.

FUNCTIONAL DESCRIPTION

The Am9517A block diagram includes the major logic blocks and all of the internal registers. The data interconnection paths are also shown. Not shown are the various control signals between the blocks. The Am9517A contains 344 bits of internal memory in the form of registers. Figure 2 lists these registers by name and shows the size of each. A detailed description of the registers and their functions can be found under Register Description.

The Am9517A contains three basic blocks of control logic. The Timing Control block generates internal timing and external control signals for the Am9517A. The Program Command Control block decodes the various commands given to the Am9517A by the microprocessor prior to servicing a DMA Request. It also decodes each channel's Mode Control word. The Priority Encoder block resolves priority contention among DMA channels requesting service simultaneously.

The Timing Control block derives internal timing from the clock input. In Am9080A systems this input will usually be the ϕ2 TTL clock from an Am8224. However, any appropriate system clock will suffice.

DMA Operation

The Am9517A is designed to operate in two major cycles. These are called Idle and Active cycles. Each device cycle is made up of a number of states. The Am9517A can assume seven separate states, each composed of one full clock period. State I (SI) is the inactive state. It is entered when the Am9517A has no valid DMA requests pending. While in SI, the DMA controller is inactive but may be in the Program Condition, being programmed by the processor. State 0 (S0) is the first state of a DMA service. The Am9517A has requested a hold but the processor has not yet returned an acknowledge. An acknowledge from the CPU will signal that transfers may begin. S1, S2, S3 and S4 are the working states of the DMA service. If more time is needed to complete a transfer than is available with normal timing, wait states (SW) can be inserted before S4 by the use of the Ready line on the Am9517A.

Memory-to-memory transfers require a read-from and a write-to-memory to complete each transfer. The states, which resemble the normal working states, use two digit numbers for identification. Eight states are required for each complete transfer. The first four states (S11, S12, S13, S14) are used for the read-from-memory half and the last four states (S21, S22, S23 and S24) for the write-to-memory half of the transfer. The Temporary Data register is used for intermediate storage of the memory byte.

IDLE Cycle

When no channel is requesting service, the Am9517A will enter the Idle cycle and perform "SI" states. In this cycle the Am9517A will sample the DREQ lines every clock cycle to determine if any channel is requesting a DMA service. The device will also sample $\overline{CS}$, looking for an attempt by the microprocessor to write or read the internal registers of the Am9517A. When

$\overline{CS}$ is low and HACK is low the Am9517A enters the Program Condition. The CPU can now establish, change or inspect the internal definition of the part by reading from or writing to the internal registers. Address lines A0-A3 are inputs to the device and select which registers will be read or written. The $\overline{IOR}$ and $\overline{IOW}$ lines are used to select and time reads or writes. Due to the number and size of the internal registers, an internal flip/flop is used to generate an additional bit of address. This bit is used to determine the upper or lower byte of the 16-bit Address and Word Count registers. The flip/flop is reset by Master Clear or Reset. A separate software command can also reset this flip/flop.

Special software commands can be executed by the Am9517A in the Program Condition. These commands are decoded as sets of addresses when both $\overline{CS}$ and $\overline{IOW}$ are active and do not make use of the data bus. Functions include Clear First/Last Flip/Flop and Master Clear.

ACTIVE CYCLE

When the Am9517A is in the Idle cycle and a channel requests a DMA service, the device will output a HREQ to the microprocessor and enter the Active cycle. It is in this cycle that the DMA service will take place, in one of four modes:

Single Transfer Mode: In Single Transfer mode, the Am9517A will make a one-byte transfer during each HREQ/HACK handshake. When DREQ goes active, HREQ will go active. After the CPU responds by driving HACK active, a one-byte transfer will take place. Following the transfer, HREQ will go inactive, the word count will be decremented and the address will be either incremented or decremented. When the word count goes to zero a Terminal Count (TC) will cause an Autoinitialize if the channel has been programmed to do so.

To perform a single transfer, DREQ must be held active only until the corresponding DACK goes active. If DREQ is held continuously active, HREQ will go inactive following each transfer and then will go active again and a new one-byte transfer will be made following each rising edge of HACK. In 8080A/9080A systems this will ensure one full machine cycle of execution between DMA transfers. Details of timing between the Am9517A and other bus control protocols will depend upon the characteristics of the microprocessor involved.

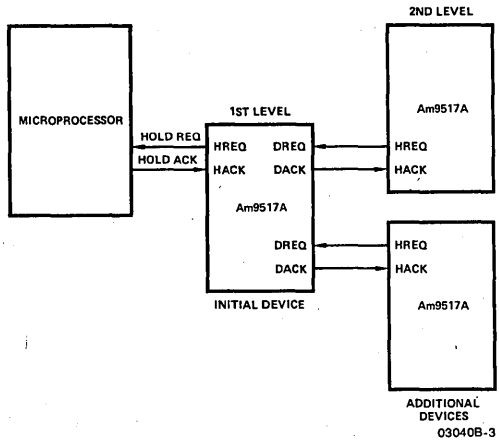
Block Transfer Mode: In Block Transfer mode, the Am9517A will continue making transfers until a TC (caused by the word count going to zero) or an external End of Process ($\overline{EOP}$) is encountered. DREQ need be held active only until DACK becomes active. An autoinitialize will occur at the end of the service if the channel has been programmed for it.

Demand Transfer Mode: In Demand Transfer mode the device will continue making transfers until a TC or external $\overline{EOP}$ is encountered or until DREQ goes inactive. Thus, the device requesting service may discontinue transfers by bringing DREQ inactive. Service may be resumed by asserting an active DREQ once again. During the time between services when the microprocessor is allowed to operate, the intermediate values of address and word count may be read from the Am9517A Current Address and Current Word Count registers. Autoinitialization will only occur following a TC or $\overline{EOP}$ at the end of service. Following Autoinitialization, an active-going DREQ edge is required to initiate a new DMA service.

Cascade Mode: This mode is used to cascade more than one Am9517A together for simple system expansion. The HREQ and HACK signals from the additional Am9517A are connected to the DREQ and DACK signals of a channel of the initial Am9517A. This allows the DMA requests of the additional device to propagate through the priority network circuitry of the preceding device. The priority chain is preserved and the new device must wait for its turn to acknowledge requests. Since the cascade channel in the initial device is used only for prioritizing the additional device, it does not output any address or control signals of its own. These would conflict with the outputs of the active channel in the added device. The Am9517A will respond to DREQ with DACK but all other outputs except HREQ will be disabled.

Figure 3 shows two additional devices cascaded into an initial device using two of the previous channels. This forms a two level DMA system. More Am9517As could be added at the second level by using the remaining channels of the first level. Additional devices can also be added by cascading into the channels of the second level devices forming a third level.

Figure 3. Cascaded Am9517As.



TRANSFER TYPES

Each of the three active transfer modes can perform three different types of transfers. These are Read, Write and Verify. Write transfers move data from an I/O device to the memory by activating $\overline{IOR}$ and $\overline{MEMW}$. Read transfers move data from memory to an I/O device by activating $\overline{MEMR}$ and $\overline{IOW}$. Verify transfers are pseudo transfers; the Am9517A operates as in Read or Write transfers generating addresses, responding to $\overline{EOP}$, etc., however, the memory and I/O control lines remain inactive.

Memory-to-Memory: The Am9517A includes a block move capability that allows blocks of data to be moved from one memory address space to another. When Bit C0 in the Command register is set to a logical 1, channels 0 and 1 will operate as memory-to-memory transfer channels. Channel 0 forms the source address and channel 1 forms the destination address. The channel 1 word count is used. A memory-to-memory transfer is initiated by setting a software DMA request for channel 0. Block Transfer Mode should be used for memory-to-memory. When channel 0 is programmed for a fixed source address, a single source word may be written into a block of memory.

When setting up the Am9517A for memory-to-memory operation, it is suggested that both channels 0 and 1 be masked out.

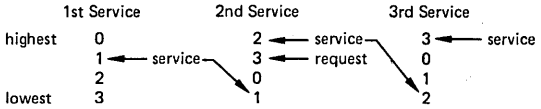
Further, the channel 0 word count should be initialized to the same value used in channel 1. No DACK outputs will be active during memory-to-memory transfers.

The Am9517A will respond to external $\overline{EOP}$ signals during memory-to-memory transfers. Data comparators in block search schemes may use this input to terminate the service when a match is found. The timing of memory-to-memory transfers may be found in Timing Diagram 4.

Autointialize: By programming a bit in the Mode register a channel may be set up for an Autoinitialize operation. During Autoinitialization, the original values of the Current Address and Current Word Count registers are automatically restored from the Base Address and Base Word Count registers of that channel following $\overline{EOP}$. The base registers are loaded simultaneously with the current registers by the microprocessor and remain unchanged throughout the DMA service. The mask bit is not set by $\overline{EOP}$ when the channel is in Autoinitialize. Following Autoinitialize the channel is ready to repeat its service without CPU intervention.

Priority: The Am9517A has two types of priority encoding available as software selectable options. The first is Fixed Priority which fixes the channels in priority order based upon the descending value of their number. The channel with the lowest priority is 3 followed by 2, 1 and the highest priority channel, 0.

The second scheme is Rotating Priority. The last channel to get service becomes the lowest priority channel with the others rotating accordingly. With Rotating Priority in a single chip DMA system, any device requesting service is guaranteed to be recognized after no more than three higher priority services have occurred. This prevents any one channel from monopolizing the system.



The priority encoder selects the highest priority channel requesting service on each active-going HACK edge. Once a channel is started, its operation will not be suspended if a request is received by a higher priority channel. The high priority channel will only gain control after the lower priority channel releases HREQ. When control is passed from one channel to another, the CPU will always gain bus control. This ensures generation of rising HACK edge to be used to initiate selection of the new highest-priority requesting channel.

Compressed Timing: In order to achieve even greater throughput where system characteristics permit, the Am9517A can compress the transfer time to two clock cycles. From Timing Diagram 3 it can be seen that state S3 is used to extend the access time of the read pulse. By removing state S3 the read pulse width is made equal to the write pulse width and a transfer consists only of state S2 to change the address and state S4 to perform the read/write. S1 states will still occur when A8-A15 need updating (see Address Generation). Timing for compressed transfers is found in Timing Diagram 6.

Extended Write: For Flyby Transactions late write is normally used, as this allows sufficient time for the $\overline{IOR}$ signal to get data from the peripheral onto the bus before $\overline{MEMW}$ is activated. In some systems, performance can be improved by starting the write cycle earlier. This is especially true for memory-to-memory transactions.

Address Generation: In order to reduce pin count, the Am9517A multiplexes the eight higher order address bits on the data lines. State S1 is used to output the higher order address bits to an external latch from which they may be placed on the address bus. The falling edge of Address Strobe (ADSTB) is used to load these bits from the data lines to the latch. Address Enable (AEN) is used to enable the bits onto the address bus through a 3-state enable. The lower order address bits are output by the Am9517A directly. Lines A0-A7 should be connected to the address bus. Timing Diagram 3 shows the time relationships between CLK, AEN, ADSTB, DB0-DB7 and A0-A7.

During Block and Demand Transfer mode services which include multiple transfers, the addresses generated will be sequential. For many transfers the data held in the external address latch will remain the same. This data need only change when a carry or borrow from A7 to A8 takes place in the normal sequence of addresses. To save time and speed transfers, the Am9517A executes S1 states only when updating of A8-A15 in the latch is necessary. This means for long services, S1 states may occur only once every 256 transfers, a savings of 255 clock cycles for each 256 transfers.

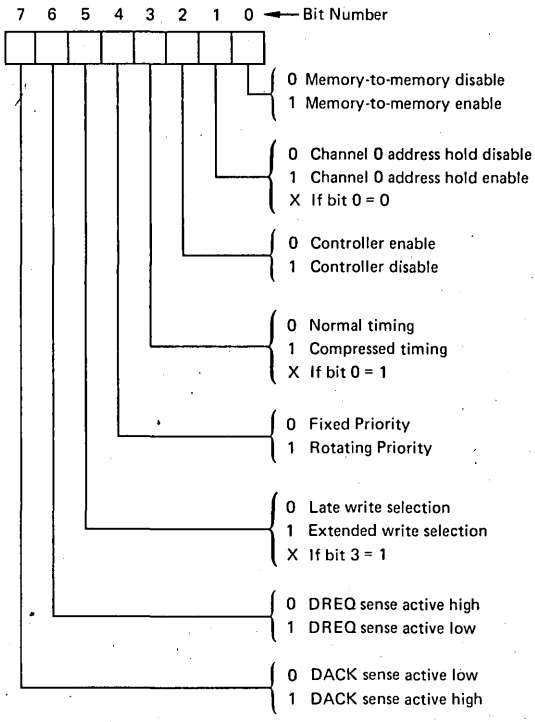
REGISTER DESCRIPTION

Current Address Register: Each channel has a 16-bit Current Address register. This register holds the value of the address used during DMA transfers. The address is automatically incremented or decremented after each transfer and the intermediate values of the address are stored in the Current Address register during the transfer. This register is written or read by the microprocessor in successive 8-bit bytes. It may also be reinitialized by an Autoinitialize back to its original value. Autoinitialization takes place only after an $\overline{EOP}$.

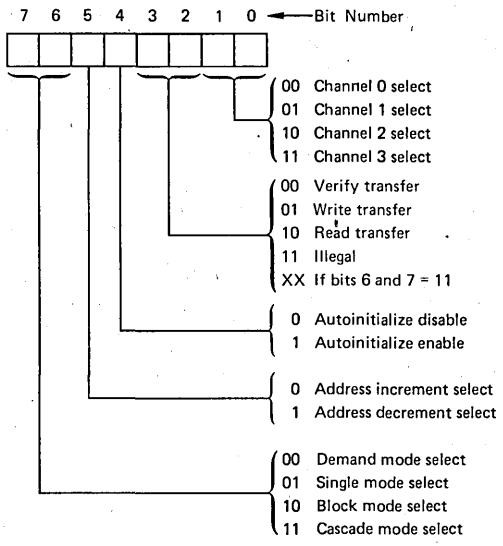
Current Word Count Register: Each channel has a 16-bit Current Word Count register. This register should be programmed with, and will return on a CPU read, a value one less than the number of words to be transferred. The word count is decremented after each transfer. The intermediate value of the word count is stored in the register during the transfer. When the value in the register goes to zero, a TC will be generated. This register is loaded or read in successive 8-bit bytes by the microprocessor in the Program Condition. Following the end of a DMA service it may also be reinitialized by an Autoinitialize back to its original value. Autoinitialize can occur only when an $\overline{EOP}$ occurs. Note that the contents of the Word Count register will be FFFF (hex) following on internally generated $\overline{EOP}$.

Base Address and Base Word Count Registers: Each channel has a pair of Base Address and Base Word Count registers. These 16-bit registers store the original values of their associated current registers. During Autoinitialize these values are used to restore the current registers to their original values. The base registers are written simultaneously with their corresponding current register in 8-bit bytes during DMA programming by the microprocessor. Accordingly, writing to these registers when intermediate values are in the Current registers will overwrite the intermediate values. The Base registers cannot be read by the microprocessor.

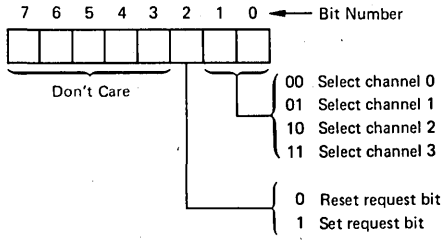
Command Register: This 8-bit register controls the operation of the Am9517A. It is programmed by the microprocessor in the Program Condition and is cleared by Reset. The following table lists the function of the command bits. See Figure 4 for address coding.



Mode Register: Each channel has a 6-bit Mode register associated with it. When the register is being written to by the microprocessor in the Program Condition, bits 0 and 1 determine which channel Mode register it to be written.

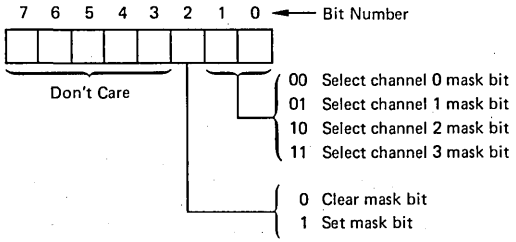


Request Register: The Am9517A can respond to requests for DMA service which are initiated by software as well as by a DREQ. Each channel has a request bit associated with it in the 4-bit Request register. These are nonmaskable and subject to prioritization by the Priority Encoder network. Each register bit is set or reset separately under software control or is cleared upon generation of a TC or external EOP. The entire register is cleared by a Reset. To set or reset a bit, the software loads the proper form of the data word. See Figure 4 for address coding.

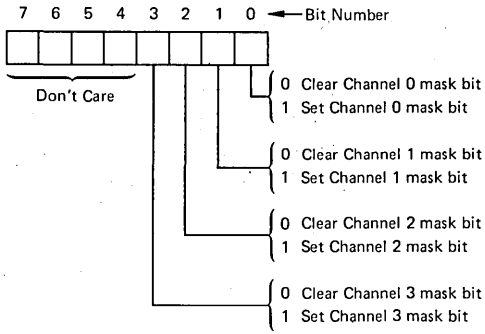


Software requests will be serviced only if the channel is in Block mode. When initiating a memory-to-memory transfer, the software request for channel 0 should be set.

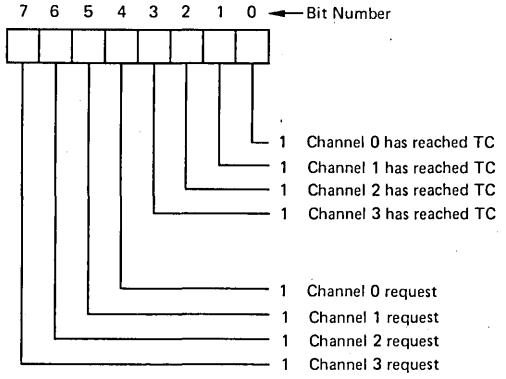
Mask Register: Each channel has associated with it a mask bit which can be set to disable the incoming DREQ. Each mask bit is set when its associated channel produces an EOP if the channel is not programmed for Autoinitialize. Each bit of the 4-bit Mask register may also be set or cleared separately under software control. The entire register is also set by a Reset. This disables all DMA requests until a clear Mask register instruction allows them to occur. The instruction to separately set or clear the mask bits is similar in form to that used with the Request register. See Figure 4 for instruction addressing.



All four bits of the Mask Register may also be written with a single command.



Status Register: The Status registers may be read out of the Am9517A by the microprocessor. It indicates which channels have reached a terminal count and which channels have pending DMA requests. Bits 0-3 are set each time a TC is reached by that channel, including after each Autoinitialization. These bits are cleared by Reset and each Status Read. Bits 4-7 are set whenever their corresponding channel is requesting service.



Temporary Register: The Temporary register is used to hold data during memory-to-memory transfers. Following the completion of the transfers, the last word moved can be read by the microprocessor in the Program Condition. The Temporary register always contains the last byte transferred in the previous memory-to-memory operation, unless cleared by a Reset.

Software Commands: There are two special software commands which can be executed in the Program Condition. They do not depend on any specific bit pattern on the data bus. The two software commands are:

Clear First/Last Flip/Flop: This command may be issued prior to writing or reading Am9517A address or word count information. This initializes the Flip/Flop to a known state so that subsequent accesses to register contents by the microprocessor will address lower and upper bytes in the correct sequence. When the Flip/Flop is cleared it addresses the lower byte and when set it addresses the upper byte.

Master Clear: This software instruction has the same effect as the hardware Reset. The Command, Status, Request, Temporary and Internal First/Last Flip/Flop registers are cleared and the Mask register is set. The Am9517A will enter the Idle cycle.

Figure 4 lists the address codes for the software commands.

Figure 4. Register and Function Addressing.

Interface Signals						Operation
A3	A2	A1	A0	$\overline{IOR}$	$\overline{IOW}$	
1	0	0	0	0	1	Read Status Register
1	0	0	0	1	0	Write Command Register
1	0	0	1	0	1	Illegal
1	0	0	1	1	0	Write Request Register
1	0	1	0	0	1	Illegal
1	0	1	0	1	0	Write Single Mask Register Bit
1	0	1	1	0	1	Illegal
1	0	1	1	1	0	Write Mode Register
1	1	0	0	0	1	Illegal
1	1	0	0	1	0	Clear Byte Pointer Flip/Flop
1	1	0	1	0	1	Read Temporary Register
1	1	0	1	1	0	Master Clear
1	1	1	0	0	1	Illegal
1	1	1	0	1	0	Illegal
1	1	1	1	0	1	Illegal
1	1	1	1	1	0	Write All Mask Register Bits

Figure 5. Word Count and Address Register Command Codes.

Channel	Register	Operation	Signals							Internal Flip/Flop	Data Bus DB0–DB7
			$\overline{CS}$	$\overline{IOR}$	$\overline{IOW}$	A3	A2	A1	A0		
0	Base & Current Address	Write	0	1	0	0	0	0	0	0	A0-A7
			0	1	0	0	0	0	0	1	A8-A15
	Current Address	Read	0	0	1	0	0	0	0	0	A0-A7
			0	0	1	0	0	0	0	1	A8-A15
	Base & Current Word Count	Write	0	1	0	0	0	0	1	0	W0-W7
			0	1	0	0	0	0	1	1	W8-W15
	Current Word Count	Read	0	0	1	0	0	0	1	0	W0-W7
			0	0	1	0	0	0	1	1	W8-W15
1	Base & Current Address	Write	0	1	0	0	0	1	0	0	A0-A7
			0	1	0	0	0	1	0	1	A8-A15
	Current Address	Read	0	0	1	0	0	1	0	0	A0-A7
			0	0	1	0	0	1	0	1	A8-A15
	Base & Current Word Count	Write	0	1	0	0	0	1	1	0	W0-W7
			0	1	0	0	0	1	1	1	W8-W15
	Current Word Count	Read	0	0	1	0	0	1	1	0	W0-W7
			0	0	1	0	0	1	1	1	W8-W15
2	Base & Current Address	Write	0	1	0	0	1	0	0	0	A0-A7
			0	1	0	0	1	0	0	1	A8-A15
	Current Address	Read	0	0	1	0	1	0	0	0	A0-A7
			0	0	1	0	1	0	0	1	A8-A15
	Base & Current Word Count	Write	0	1	0	0	1	0	1	0	W0-W7
			0	1	0	0	1	0	1	1	W8-W15
	Current Word Count	Read	0	0	1	0	1	0	1	0	W0-W7
			0	0	1	0	1	0	1	1	W8-W15
3	Base & Current Address	Write	0	1	0	0	1	1	0	0	A0-A7
			0	1	0	0	1	1	0	1	A8-A15
	Current Address	Read	0	0	1	0	1	1	0	0	A0-A7
			0	0	1	0	1	1	0	1	A8-A15
	Base & Current Word Count	Write	0	1	0	0	1	1	1	0	W0-W7
			0	1	0	0	1	1	1	1	W8-W15
	Current Word Count	Read	0	0	1	0	1	1	1	0	W0-W7
			0	0	1	0	1	1	1	1	W8-W15

MAXIMUM RATINGS above which useful life may be impaired

Storage Temperature	−65 to +150°C
VCC with Respect to VSS	−0.5 to +7.0V
All Signal Voltages with Respect to VSS	−0.5V to +7.0V
Power Dissipation (Package Limitation)	1.5W

The products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of static charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid exposure to excessive voltages.

OPERATING RANGE

Part Number	T_A	V_{CC}
Am9517ADC/PC	0 to +70°C	5.0V ±5%
Am9517A-1DC/PC	0 to +70°C	5.0V ±5%
Am9517A-4DC/PC	0 to +70°C	5.0V ±5%
Am9517A-5DC/PC	0 to +70°C	5.0V ±5%
Am9517ADI	−40 to +85°C	5.0V ±10%
Am9517A-1DI	−40 to +85°C	5.0V ±10%
Am9517A-4DI	−40 to +85°C	5.0V ±10%
Am9517ADMB	−55 to +125°C	5.0V ±10%

ELECTRICAL CHARACTERISTICS over operating range (Note 1)

Parameter	Description	Test Conditions	Min	Typ	Max	Unit
VOH	Output HIGH Voltage	IOH = −200μA	2.4			Volts
		IOH = −100μA, (HREQ Only)	3.3			
VOL	Output LOW Voltage	IOL = 3.2mA			0.45	Volts
VIH	Input HIGH Voltage		2.0		VCC+0.5	Volts
VIL	Input LOW Voltage		−0.5		0.8	Volts
IIX	Input Load Current	VSS ≤ VI ≤ VCC	−10		+10	μA
IOZ	Output Leakage Current	VCC ≤ VO ≤ VSS+.40	−10		+10	μA
ICC	VCC Supply Current	T_A = +25°C		65	130	mA
		T_A = 0°C		75	150	
		T_A = −55°C			175	
CO	Output Capacitance	fc = 1.0MHz, Inputs = 0V		4	8	pF
CI	Input Capacitance			8	15	pF
CIO	I/O Capacitance			10	18	pF

NOTES:

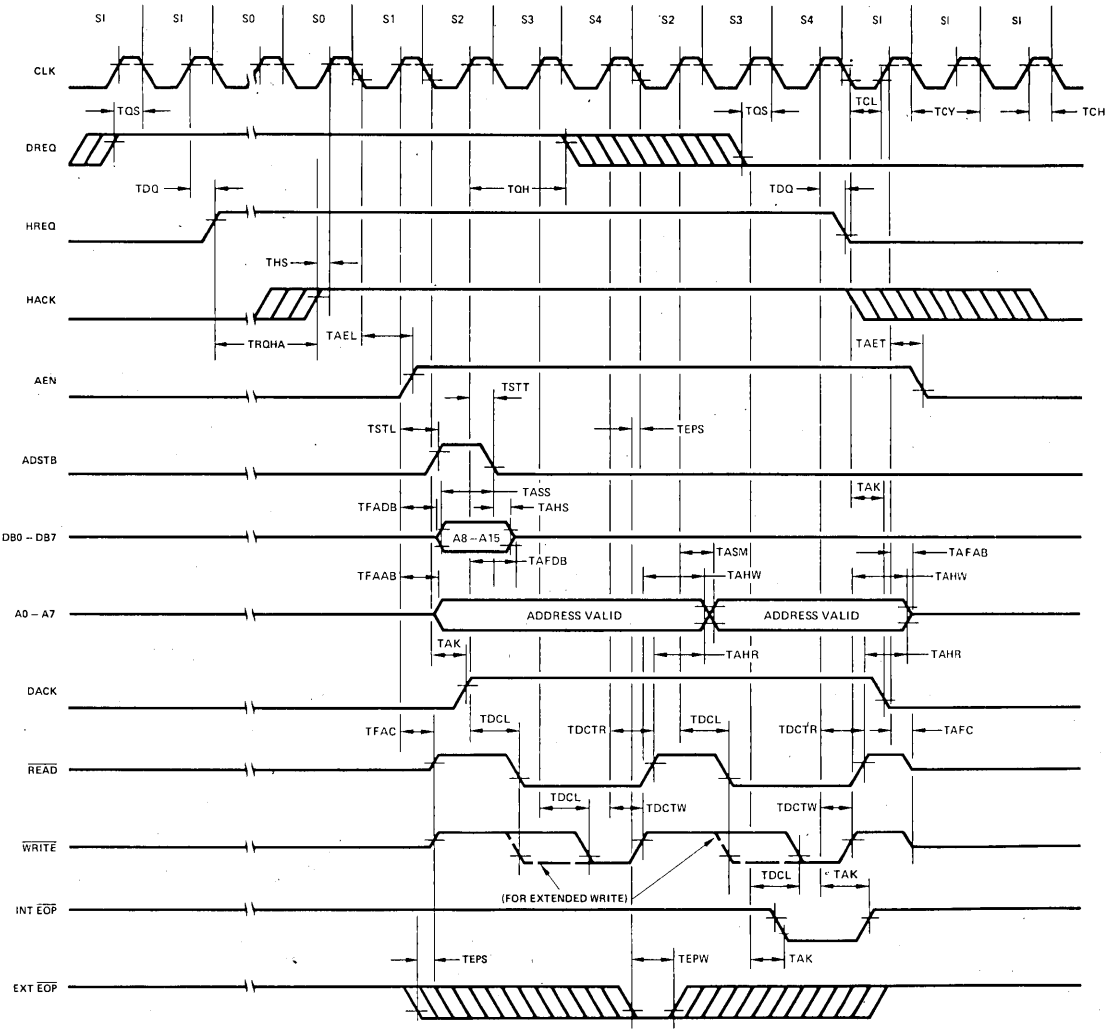
1. Typical values are for T_A = 25°C, nominal supply voltage and nominal processing parameters.
2. Input timing parameters assume transition times of 20ns or less. Waveform measurement points for both input and output signals are 2.0V for High and 0.8V for Low, unless otherwise noted.
3. Output loading is 1 Standard TTL gate plus 50pF capacitance unless noted otherwise.
4. The new $\overline{IOW}$ or $\overline{MEMW}$ pulse width for normal write will be TCY-100ns and for extended write will be 2TCY-100ns. The net $\overline{IOR}$ or $\overline{MEMR}$ pulse width for normal read will be 2TCY-50ns and for compressed read will be TCY-50ns.
5. TDQ is specified for two different output HIGH levels. TDQ1 is measured at 2.0V. TDQ2 is measured at 3.3V. The value for TDQ2 assumes an external 3.3kΩ pull-up resistor connected from HREQ to VCC.
6. DREQ should be held active until DACK is returned.
7. DREQ and DACK signals may be active high or active low. Timing diagrams assume the active high mode.
8. Output loading on the data bus is 1 Standard TTL gate plus 15pF for the minimum value and 1 Standard TTL gate plus 100pF for the maximum value.
9. Successive read and/or write operations by the external processor to program or examine the controller must be timed to allow at least 600ns for the Am9517A or Am9517A-1, at least 450ns for the Am9517A-4 and 400ns for the Am9517A-5 as recovery time between active read or write pulses.
10. Parameters are listed in alphabetical order.
11. Pin 5 is an input that should always be at a logic high level. An internal pull-up resistor will establish a logic high when the pin is left floating. Alternatively, pin 5 may be tied to VCC.
12. Signals $\overline{READ}$ and $\overline{WRITE}$ refer to $\overline{IOR}$ and $\overline{MEMW}$ respectively for peripheral-to-memory DMA operations and to $\overline{MEMR}$ and $\overline{IOW}$ respectively for memory-to-peripheral DMA operations.
13. If N wait states are added during the write-to-memory half of a memory-to-memory transfer, this parameter will increase by N (TCY).

Am9517A
SWITCHING CHARACTERISTICS
ACTIVE CYCLE (Notes 2, 3, 10, 11 and 12)

Parameters	Description	Am9517A Min	Am9517A Max	Am9517A-1 Min	Am9517A-1 Max	Am9517A-4 Min	Am9517A-4 Max	Am9517A-5 Min	Am9517A-5 Max	Units
TAEL	AEN HIGH from CLK LOW (S1) Delay Time		300		300		225		200	ns
TAET	AEN LOW from CLK HIGH (S1) Delay Time		200		200		150		130	ns
TAFAB	ADR Active to Float Delay from CLK HIGH		150		150		120		90	ns
TAFC	$\overline{READ}$ or $\overline{WRITE}$ Float from CLK HIGH		150		150		120		120	ns
TAFDB	DB Active to Float Delay from CLK HIGH		250		250		190		170	ns
TAHR	ADR from $\overline{READ}$ HIGH Hold Time	TCY-100		TCY-100		TCY-100		TCY-100		ns
TAHS	DB from ADSTB LOW Hold Time	50		50		40		30		ns
TAHW	ADR from WRITE HIGH Hold Time	TCY-50		TCY-50		TCY-50		TCY-50		ns
TAK	DACK Valid from CLK LOW Delay Time		280		280		220		170	ns
	$\overline{EOP}$ HIGH from CLK HIGH Delay Time		250		250		190		170	ns
	$\overline{EOP}$ LOW to CLK HIGH Delay Time		250		250		190		100	ns
TASM	ADR Stable from CLK HIGH		250		250		190		170	ns
TASS	DB to ADSTB LOW Setup Time	100		100		100		100		ns
TCH	Clock High Time (Transitions ≤ 10ns)	120		120		100		80		ns
TCL	Clock Low Time (Transitions ≤ 10ns)	150		150		110		68		ns
TCY	CLK Cycle Time	320		320		250		200		ns
TDCL	CLK HIGH to $\overline{READ}$ or $\overline{WRITE}$ LOW Delay (Note 4)		270		270		200		190	ns
TDCTR	$\overline{Read}$ HIGH from CLK HIGH (S4) Delay Time (Note 4)		270		270		210		190	ns
TDCTW	$\overline{WRITE}$ HIGH from CLK HIGH (S4) Delay Time (Note 4)		200		200		150		130	ns
TDQ1	HREQ Valid from CLK HIGH Delay Time (Note 5)		160		160		120		120	ns
TDQ2			250		250		190		120	ns
TEPS	$\overline{EOP}$ LOW from CLK LOW Setup Time	60		60		45		40		ns
TEPW	$\overline{EOP}$ Pulse Width	300		300		225		220		ns
TFAAB	ADR Float to Active Delay from CLK HIGH		250		250		190		170	ns
TFAC	$\overline{READ}$ or $\overline{WRITE}$ Active from CLK HIGH		200		200		150		150	ns
TFADB	DB Float to Active Delay from CLK HIGH		300		300		225		200	ns
THS	HACK Valid to CLK HIGH Setup Time	100		100		75		75		ns
TIDH	Input Data from $\overline{MEMR}$ HIGH Hold Time	0		0		0		0		ns
TIDS	Input Data to $\overline{MEMR}$ HIGH Setup Time	250		250		190		170		ns
TODH	Output Data from $\overline{MEMW}$ HIGH Hold Time	20		20		20		10		ns
TODV	Output Data Valid to $\overline{MEMW}$ HIGH (Note 13)	200		200		125		125		ns
TQS	DREQ to CLK LOW (S1, S4) Setup TIme	120		0		0		0		ns
TRH	CLK to READY LOW Hold Time	20		20		20		20		ns
TRS	READY to CLK LOW Setup Time	100		100		60		60		ns
TSTL	ADSTB HIGH from CLK HIGH Delay Time		200		200		150		130	ns
TSTT	ADSTB LOW from CLK HIGH Delay Time		140		140		110		90	ns
TQH	DREQ from DACK Valid Hold Time	0		0		0		0		ns
TRQHA	HREQ to HACK Delay Time	1		1		1		1		clk

SWITCHING WAVEFORMS

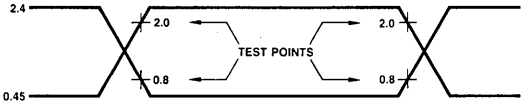
Timing Diagram 1. Active Cycle Timing Diagram



Note: EOP must precede AEN in single transfer mode.

03040B-4

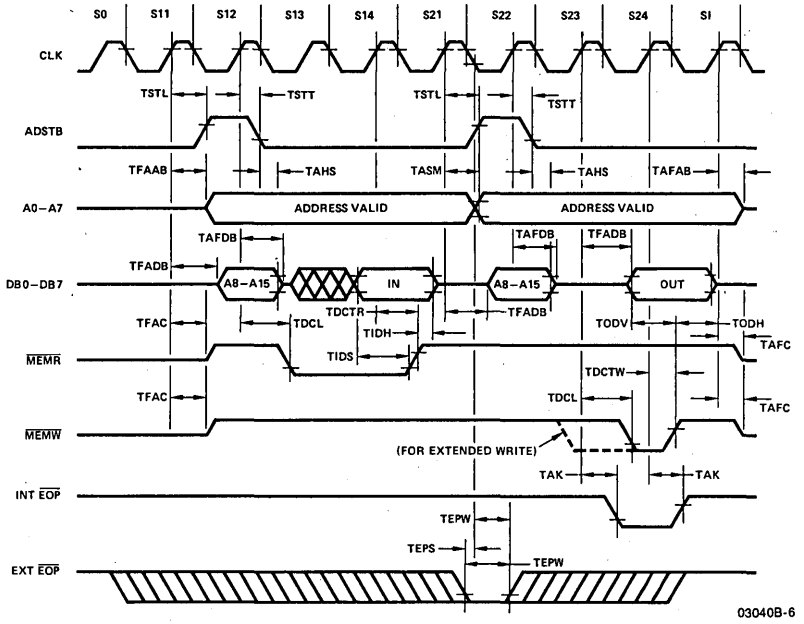
INPUT WAVEFORMS FOR AC TESTS



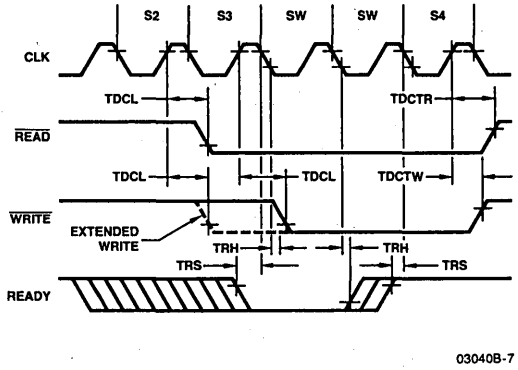
03040B-5

SWITCHING WAVEFORMS (Cont.)

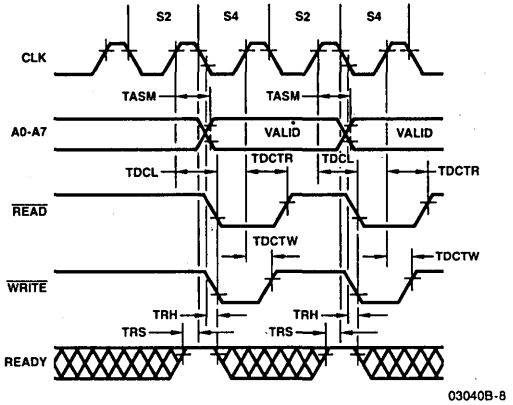
Timing Diagram 2. Memory-to-Memory



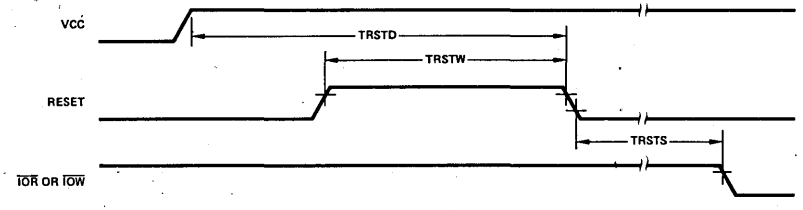
Timing Diagram 3. Ready Timing



Timing Diagram 4. Compressed Timing



Timing Diagram 5. Reset Timing



SWITCHING CHARACTERISTICS

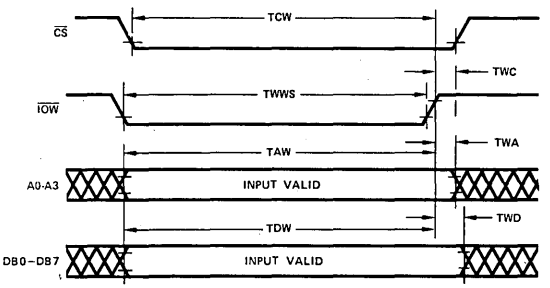
PROGRAM CONDITION (IDLE CYCLE)

(Notes 2, 3, 10, and 11)

Parameters	Description	Am9517A Min	Am9517A Max	Am9517A-1 Min	Am9517A-1 Max	Am9517A-4 Min	Am9517A-4 Max	Am9517A-5 Min	Am9517A-5 Max	Units
TAR	ADR Valid or $\overline{CS}$ LOW to $\overline{READ}$ LOW	50		50		50		50		ns
TAW	ADR Valid to $\overline{WRITE}$ HIGH Setup Time	200		200		150		130		ns
TCW	CS LOW to $\overline{WRITE}$ HIGH Setup Time	200		200		150		130		ns
TDW	Data Valid to $\overline{WRITE}$ HIGH Setup Time	200		200		150		130		ns
TRA	ADR or CS Hold from $\overline{READ}$ HIGH	0		0		0		0		ns
TRDE	Data Access from $\overline{READ}$ LOW (Note 8)		300		200		200		140	ns
TRDF	DB Float Delay from $\overline{READ}$ HIGH	20	150	20	100	20	100	0	70	ns
TRSTD	Power Supply HIGH to RESET LOW Setup Time	500		500		500		500		ns
TRSTS	RESET to First $\overline{IOWR}$	2TCY		2TCY		2TCY		2TCY		ns
TRSTW	RESET Pulse Width	300		300		300		300		ns
TRW	$\overline{READ}$ Width	300		300		250		200		ns
TWA	ADR from $\overline{WRITE}$ HIGH Hold Time	20		20		20		20		ns
TWC	CS HIGH from $\overline{WRITE}$ HIGH Hold Time	20		20		20		20		ns
TWD	Data from $\overline{WRITE}$ HIGH Hold Time	30		30		30		30		ns
TWWS	Write Width	200		200		200		160		ns

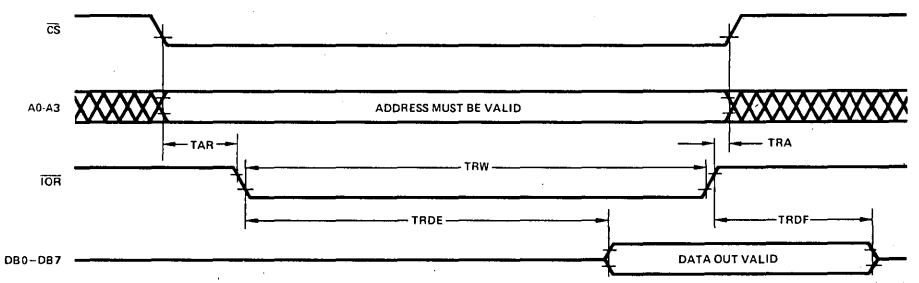
SWITCHING WAVEFORMS (Cont.)

Timing Diagram 6. Program Condition Write Timing (Note 9)



03040B-10

Timing Diagram 7. Program Condition Read Cycle (Note 9)

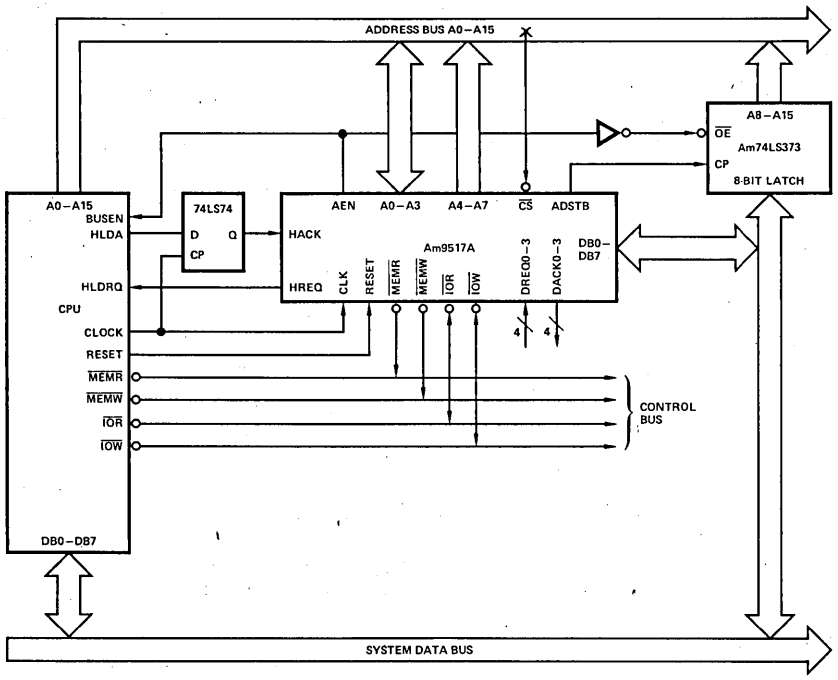


03040B-11

APPLICATION INFORMATION

Figure 6 shows a convenient method for configuring a DMA system with the Am9517A Controller and a microprocessor system. The Multimode DMA Controller issues a Hold Request to the processor whenever there is at least one valid DMA Request from a peripheral device. When the processor replies with a Hold Acknowledge signal, the Am9517A takes control of the Address Bus, the Data Bus and the Control Bus. The address for the first transfer operation comes out in two bytes – the least significant eight bits on the eight Address outputs and the most significant eight bits on the Data Bus. The contents of the Data Bus are then latched into the Am74LS373 register to complete the full 16 bits of the Address Bus. The Am74LS373 is a high speed, low power, 8-bit, 3-state register in a 20-pin package. After the initial transfer takes place, the register is updated only after a carry or borrow is generated in the least significant address byte. Four DMA channels are provided when one Am9517A is used.

Figure 6. Basic DMA Configuration.



03040B-12

Am9519A

Universal Interrupt Controller

DISTINCTIVE CHARACTERISTICS

- Eight individually maskable interrupt inputs reduce CPU overhead
- Unlimited interrupt channel expansion with no extra hardware
- Programmable 1-to 4-byte response provides vector address and message protocol for 8-bit CPUs
- Rotating and fixed priority resolution logic
- Software interrupt request capability
- Common vector and polled mode options
- Automatic hardware clear of in-service interrupts reduces software overhead
- Polarity control of interrupt inputs and outputs
- Reset minimizes software initialization by automatically generating CALL to location zero

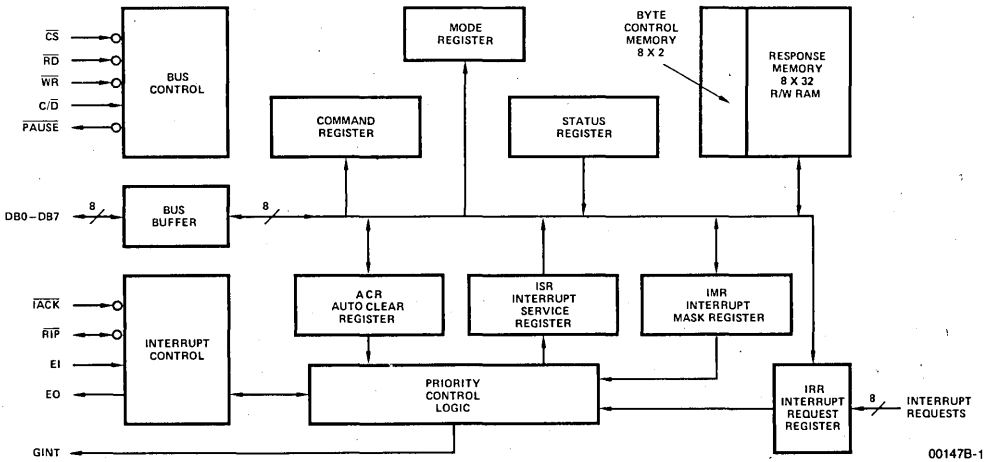
GENERAL DESCRIPTION

The Am9519A Universal Interrupt Controller is a processor support circuit that provides a powerful interrupt structure to increase the efficiency and versatility of microcomputer-based systems. A single Am9519A manages up to eight maskable interrupt request inputs, resolves priorities and supplies up to four bytes of fully programmable response for each interrupt. It uses a simple expansion structure that allows many units to be cascaded for control of large numbers of interrupts. Several programmable control features are provided to enhance system flexibility and optimization.

The Universal Interrupt Controller is designed with a general purpose interface to facilitate its use with a wide range of digital systems, including most popular 8-bit microprocessors. Since the response bytes are fully programmable, any instruction or vectoring protocol appropriate for the host processor may be used.

When the Am9519A controller receives an unmasked Interrupt Request, it issues a Group Interrupt output to the CPU. When the interrupt is acknowledged, the controller outputs the one-to-four byte response associated with the highest priority unmasked interrupt request. The ability of the CPU to set interrupt requests under software control permits hardware prioritization of software tasks and aids system diagnostic and maintenance procedures.

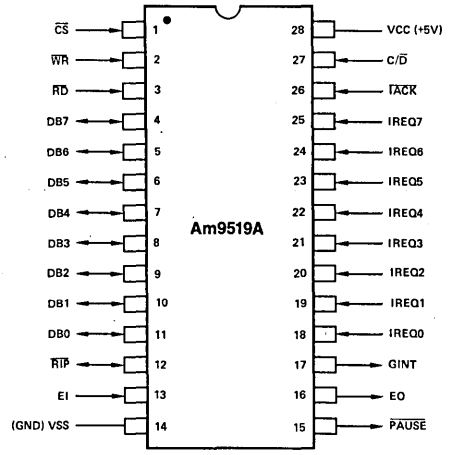
BLOCK DIAGRAM



ORDERING INFORMATION

Package Type	Ambient Temperature	Timing Options	
Molded DIP	$0°C \leq T_A \leq +70°C$	Am9519APC	Am9519A-1PC
Hermetic DIP		Am9519ADC	Am9519A-1DC
Hermetic DIP	$-40°C \leq T_A \leq +85°C$	Am9519ADI	
Hermetic DIP	$-55°C \leq T_A \leq +125°C$	Am9519ADMB	

CONNECTION DIAGRAM — Top View
D-28, P-28



Top View

Pin 1 is marked for orientation. 00147B-2

INTERFACE SIGNAL DESCRIPTION

VCC: +5 Volt Power Supply
VSS: Ground

DB0 – DB7 (Data Bus, Input/Output)

The eight bidirectional data bus signals are used to transfer information between the Am9519A and the system data bus. The direction of transfer is controlled by the $\overline{IACK}$, $\overline{WR}$ and $\overline{RD}$ input signals. Programming and control information are written into the device; status and response data are output by it.

$\overline{CS}$ (Chip Select, Input)

The active low Chip Select input enables read and write operations on the data bus. Interrupt acknowledge responses are not conditioned by $\overline{CS}$.

$\overline{RD}$ (Read, Input)

The active low Read signal is conditioned by $\overline{CS}$ and indicates that information is to be transferred from the Am9519A to the data bus.

$\overline{WR}$ (Write, Input)

The active low Write signal is conditioned by $\overline{CS}$ and indicates that data bus information is to be transferred from the data bus to a location within the Am9519A.

C/$\overline{D}$ (Control/Data, Input)

The C/$\overline{D}$ control signal selects source and destination locations for data bus read and write operations. Data read or write transfers are made to or from preselected internal registers or memory locations. Control write operations load the command register and control read operations output the status register.

IREQ0 – IREQ7 (Interrupt Request, Input)

The Interrupt Request signals are used by external devices to indicate that service by the host CPU is desired. IREQ inputs are accepted asynchronously and they may be programmed for either a high-to-low or low-to-high edge transition. Active inputs are latched internally in the Interrupt Request Register. After the IRR bit is cleared, an IREQ transition of the programmed polarity must occur to initiate another request.

$\overline{RIP}$ (Response In Process, Input/Output)

Response In Process is a bidirectional signal used when two or more Am9519A circuits are cascaded. It permits multibyte response transfers to be completed without interference from higher priority interrupts. An Am9519A that is responding to an acknowledged interrupt will treat $\overline{RIP}$ as an output and hold it low until the acknowledge response is finished. An Am9519A without an acknowledged interrupt will treat $\overline{RIP}$ as an input and will ignore $\overline{IACK}$ pulses as long as $\overline{RIP}$ is low. The $\overline{RIP}$ output is open drain and requires an external pullup resistor to VCC.

$\overline{IACK}$ (Interrupt Acknowledge, Input)

The active low Interrupt Acknowledge line indicates that the external system is asking for interrupt response information. Depending on the programmed state of the Am9519A, it will accept 1, 2, 3 or 4 $\overline{IACK}$ pulses; one response byte is transferred per pulse. The first $\overline{IACK}$ pulse causes selection of the highest priority unmasked pending interrupt request and generates a $\overline{RIP}$ output signal.

$\overline{PAUSE}$ (Pause, Output)

The active-low Pause signal is used to coordinate interrupt responses with data bus and control timing. Pause goes low when the first $\overline{IACK}$ is received and remains low until $\overline{RIP}$ goes low. The external system can use Pause to stretch the acknowledge cycle and allow the control timing to automatically adjust to the actual priority resolution delays in the interrupt system. Second, third and fourth response bytes do not cause Pause to go low. Pause is an open drain output and requires an external pullup resistor to VCC.

EO (Enable Out, Output)

The active high EO signal is used to implement daisy-chained cascading of several Am9519A circuits. EO is connected to the EI input of the next lower priority chip. On receipt of an interrupt acknowledge, each EO will go inactive until it has been determined that no valid interrupt request is pending on that chip. If an active request is present, EO remains low. EO is also held low when the master mask bit is active, thus disabling all lower priority chips.

EI (Enable In, Input)

The active high EI signal is used to implement daisy-chained cascading of several Am9519A circuits. EI is connected to EO of the next higher priority chip. It may also be used as a hardware disable input for the interrupt system. When EI is low $\overline{IACK}$ inputs will not affect ISR, however, $\overline{PAUSE}$ will go low until $\overline{RIP}$ goes low. EI is internally pulled up to VCC so that no external pullup is needed when EI is not used.

GINT (Group Interrupt, Output)

The Group Interrupt output signal indicates that at least one unmasked interrupt request is pending. It may be programmed for active high or active low polarity. When active low, the output is open drain and requires an external pull up resistor to VCC. Since a glitch on GINT occurs approximately 100nsec after the last IACK pulse this pin should not be connected to edge sensitive devices.

REGISTER DESCRIPTION

Interrupt Request Register (IRR): The 8-bit IRR is used to store pending interrupt requests. A bit in the IRR is set whenever the corresponding IREQ input goes active. Bits may also be set under program control from the CPU, thus permitting software generated interrupts. IRR bits may be cleared under program control. An IRR bit is automatically cleared when its interrupt is acknowledged. All IRR bits are cleared by a reset function.

Interrupt Service Register (ISR): The 8-bit ISR contains one bit for each IREQ input. It is used to indicate that a pending interrupt has been acknowledged and to mask all lower priority interrupts. When a bit is set by the acknowledge logic in the ISR, the corresponding IRR bit is cleared. If an acknowledged interrupt is not programmed to be automatically cleared, its ISR bit must be cleared by the CPU under program control when it is desired to permit interrupts from lower priority devices. When the interrupt is programmed for automatic clearing, the ISR bit is automatically reset during the acknowledge sequence. All ISR bits are cleared by a reset function.

Interrupt Mask Register (IMR): The 8-bit IMR is used to enable or disable the individual interrupt inputs. The IMR bits correspond to the IREQ inputs and all eight may be loaded, set or cleared in parallel under program control. In addition, individual IMR bits may be set or cleared by the CPU. Care must be taken therefore when disabling a specific channel by setting its IMR bit. If that bit is causing the GINT pin to be active a lock-up condition can occur if the CPU recognizes the interrupt and then the Am9519A removes the request. During the $\overline{IACK}$ cycle $\overline{PAUSE}$ will go low and stay low. The solution is to disable CPU interrupts prior to writing to the IMR and then re-enable them. A reset function will set all eight mask bits, disabling all requests. A mask bit that is set does not disable the IRR, and an IREQ that arrives while a corresponding mask bit is set will cause an interrupt later when the mask bit is cleared. Only unmasked interrupt inputs can generate a Group Interrupt output.

Response Memory: An 8 x 32 read/write response memory is included in the Am9519A. It is used to store up to four bytes of response information for each of the eight interrupt request inputs. All bits in the memory are programmable, allowing any desired vector, opcode, instruction or other data to be entered. The Am9519A transfers the interrupt response information for the highest priority unmasked interrupt from the memory to the data bus when the $\overline{IACK}$ input is active.

Auto Clear Register: The 8-bit Auto Clear register contains one bit for each IREQ input and specifies the operating mode for each of the ISR bits. When an auto clear bit is off, the corresponding ISR bit is set when that interrupt is acknowledged and is cleared by software command. When an auto clear bit is on, the corresponding ISR bit is cleared by the hardware by the rising edge of the last acknowledge pulse. A reset function clears all auto clear bits.

Status Register: The 8-bit Status register contains information concerning the internal state of the chip. It is especially useful when operating in the polled mode in order to identify interrupting devices. Figure 1 shows the status register bit assignments. The polarity of the GINT bit 7 is not affected by the GINT polarity control (Mode bit status register bit assignments. Bits S0-S2 are set asynchronously to a status register read operation. It is recommended to read the register twice and to compare the binary vectors for equality prior to the proceeding with device service in polled mode. The polarity of the GINT bit 7 is not affected by the GINT polarity control (Mode bit 3). The Status register is read by executing a read operation ($\overline{CS} = 0$, RS = 0) with the control location selected (C/$\overline{D}$ = 1).

Mode Register: The 8-bit Mode register controls the operating options of the Am9519A. Figure 2 shows the bit assignments for the Mode register. The five low order mode bits (0 through 4) are loaded in parallel by command. Bits 5, 6 and 7 are controlled by separate commands. (See Figure 4.) The Mode register cannot be read out directly to the data bus, but Mode bits 0, 2 and 7 are available as part of the Status register.

Command Register: The 8-bit Command register stores the last command entered. Depending upon the command opcode, it may initiate internal actions or precondition the part for subsequent data bus transfers. The Command register is loaded by executing a write operation ($\overline{WR}$ = 0) with the control location selected (C/$\overline{D}$ = 1), as shown in Figure 3.

Byte Count Register: The length in bytes of the response associated with each interrupt is independently programmed so that different interrupts may have different length responses. The byte count for each response is stored in eight 2-bit Byte Count registers. For a given interrupt the Am9519A will expect to receive a number of $\overline{IACK}$ pulses that equals the corresponding byte count, and will hold $\overline{RIP}$ low until the count is satisfied.

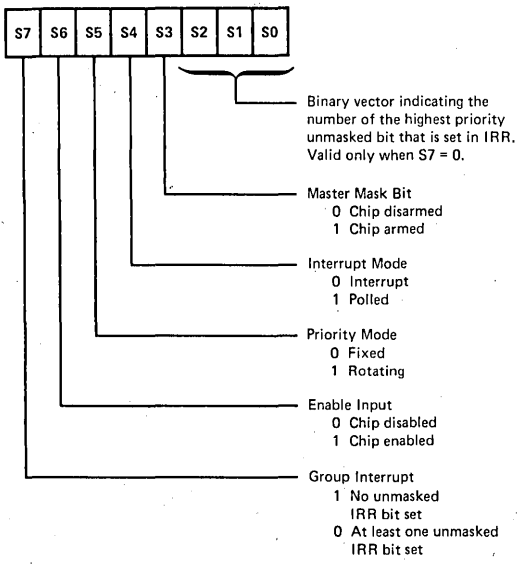


Figure 1. Status Register Bit Assignments. 00147B-3

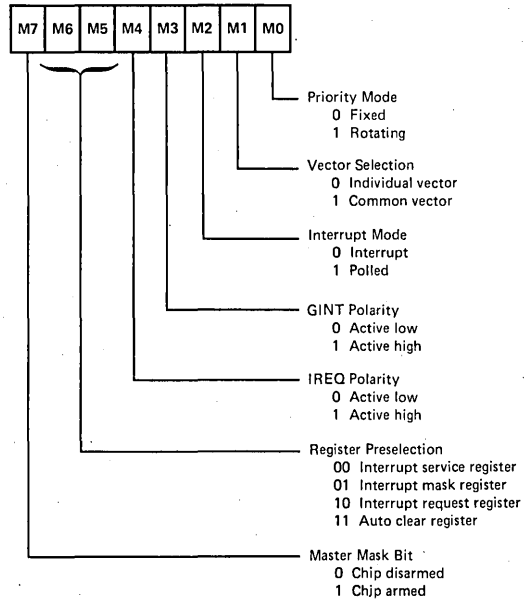


Figure 2. Mode Register Bit Assignments. 00147B-4

FUNCTIONAL DESCRIPTION

Interrupts are used to improve system throughput and response time by eliminating heavy dependence on software polling procedures. Interrupts allow external devices to asynchronously modify the instruction sequence of a program being executed. In systems with multiple interrupts, vectoring can further improve performance by allowing direct identification of the interrupting device and its associated service routine. The Am9519A Universal Interrupt Controller contains, on one chip, all of the circuitry necessary to detect, prioritize and manage eight vectored interrupts. It includes many options and operating modes that permit the design of sophisticated interrupt systems.

Reset

The reset function is accomplished by software command or automatically during power-up. The reset command may be issued by the CPU at any time. Internal power up circuitry is triggered when VCC reaches a predetermined threshold, causing a brief internal reset pulse. In both cases, the resulting internal state of the machine is that all registers are cleared except the Mask register which is set. Thus no Group Interrupt will be generated and no interrupt requests will be recognized. The response memory and Byte Count registers are not affected by reset. Their contents after power-up are unpredictable and must be established by the host CPU during initialization.

Operating Sequence

A brief description of a typical sequence of events in an operating interrupt system will illustrate the general interactions among the host CPU, the interrupt controller and the interrupting peripheral.

1. The Am9519A controller is initialized by the CPU in order to customize its configuration and operation for the application at hand. Both the controller and the CPU are then enabled to accept interrupts.
2. One (or more) of the interrupt request inputs to the controller becomes active indicating that peripheral equipment is asking for service. The controller asynchronously accepts and latches the request(s).
3. If the request is masked, no further action takes place. If the request is not masked, a Group Interrupt output is generated by the controller.
4. The GINT signal is recognized by the CPU which normally will complete the execution of the current instruction, insert an interrupt acknowledge sequence into its instruction execution stream, and disable its internal interrupt structure. The controller expects to receive one or more $\overline{IACK}$ signals from the CPU during the acknowledge sequence.
5. When the controller receives the $\overline{IACK}$ signal, it brings $\overline{PAUSE}$ low and selects the highest priority unmasked pending request. When selection is complete, the $\overline{RIP}$ output is brought low and the first byte in the response memory associated with the selected request is output on the data bus. $\overline{PAUSE}$ stays low until RIP goes low. $\overline{RIP}$ stays low until the last byte of the response has been transferred.
6. During the acknowledge sequence, the IRR bit corresponding to the selected request is automatically cleared, and the corresponding ISR bit is set by the falling edge of $\overline{IACK}$. When the ISR bit is set, the Group Interrupt output is disabled until a higher priority request arrives or the ISR bit is cleared. The ISR bit will be cleared by either hardware or software.
7. If a higher priority request arrives while the current request is being serviced, GINT will be output by the controller, but will be recognized and acknowledged only if the CPU has its interrupt input enabled. If acknowledged, the corresponding higher priority ISR bit will be set and the requests nested.

Information Transfers

Figure 3 shows the control signal configurations for all information transfer operations between the Am9519 and the data bus. The following conventions are assumed: $\overline{RD}$ and $\overline{WR}$ active are mutually exclusive; $\overline{RD}$, $\overline{WR}$ and C/$\overline{D}$ have no meaning unless $\overline{CS}$ is low; active $\overline{IACK}$ pulses occur only when $\overline{CS}$ is high.

For reading, the Status register is selected directly by the C/$\overline{D}$ control input. Other internal registers are read by preselecting the desired register with mode bits 5 and 6, and then executing a data read. The response memory can be read only with $\overline{IACK}$ pulses. For writing, the Command register is selected directly by the C/$\overline{D}$ control input. The Mask and Auto Clear registers are loaded following specific commands to that effect. To load each level of the response memory, the response preselect command is issued to select the desired level. An appropriate number of data write operations are then executed to load that level.

CONTROL INPUT					DATA BUS OPERATION
$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{IACK}$	
0	0	0	1	1	Transfer contents of preselected data register to data bus
0	0	1	0	1	Transfer contents of data bus to preselected data register
0	1	0	1	1	Transfer contents of status register to data bus
0	1	1	0	1	Transfer contents of data bus to command register
1	X	X	X	0	Transfer contents of selected response memory location to data bus
1	X	X	X	1	No information transferred

Figure 3. Summary of Data Bus Transfers.

The Pause output may be used by the host CPU to ensure that proper timing relationships are maintained with the Am9519A when $\overline{IACK}$ is active. The $\overline{IACK}$ pulse width required depends on several variables, including: operating temperature, internal logic delays, number of interrupt controllers chained together, and the priority level of the interrupt being acknowledged. When delays in these variables combine to delay selection of a request following the falling edge of the first $\overline{IACK}$, the Pause output may be used to extend the $\overline{IACK}$ pulse, if necessary. Pause will remain low until a request has been selected, as indicated by the falling edge of $\overline{RIP}$. Typically, the internal interrupt selection process is quite fast, especially for systems with a single Am9519A and Pause will consequently remain low for only a very brief interval and will not cause extension of the IACK timing.

Operating Options

The Mode register specifies the various combinations of operating options that may be selected by the CPU. It is cleared by power-up or by a reset command.

Mode bit 0 specifies the rotating/fixed priority mode (see Figure 2). In the fixed mode, priority is assigned to the request inputs based upon their physical location at the chip interface, with IREQ0 the highest and IREQ7 the lowest. In the rotating mode, relative priority is the same as for the fixed mode and the most recently serviced request is assigned the lowest priority. In the fixed mode, a lower priority request might never receive service if enough higher priority requests are active. In the rotating mode, any request will receive service within a maximum of seven other service cycles no matter what pattern the request inputs follow.

Mode bit 1 selects the individual/common vector option. Individual vectoring provides a unique location in the response memory for each interrupt request. The common vector option always supplies the response associated with IREQ0 no matter which request is being acknowledged.

Mode bit 2 specifies interrupt or polled operation. In the polled mode the Group Interrupt output is disabled. The CPU may read the Status register to determine if a request is pending. Since $\overline{IACK}$ pulses are not normally supplied in polled mode, the IRR bit is not automatically cleared, but may be cleared by command. With no $\overline{IACK}$ input the ISR and the response memory are not used. An Am9519A in the polled mode has EI connected to EO so that in multichip interrupt systems the polled chip is functionally removed from the priority hierarchy.

Mode bit 3 specifies the sense of the GINT output. When active high polarity is selected the output is a two-state configuration. For active low polarity, the output is open drain and requires an external pull-up resistor to provide the high logic level. The open drain output allows wired-or configurations with other similar output signals.

Mode bit 4 specifies the sense of the IREQ inputs. When active low polarity is selected, the IRR responds to falling edges on the request inputs. When active high is selected, the IRR responds to rising edges.

Mode bits 5 and 6 specify the register that will be read on subsequent data read operations (C/$\overline{D}$ = 0, $\overline{RD}$ = 0). This preselection remains valid until changed by a reset or a command.

Mode bit 7 is the master mask bit that disables all request inputs. It is used to disable all interrupts without modifying the IMR so that the previous IMR contents are valid when interrupts are re-enabled. When the master mask bit is low, it causes the EO line to remain disabled (low). Thus, for multiple-chip interrupt systems, one master mask bit can disable the whole interrupt structure. Alternatively, portions of the structure may be disabled. The state of the master mask bit is available as bit S3 of the Status register.

Programming

After reset, the Am9519A must be initialized by the CPU in order to perform useful work. At a minimum, the master mask bit and at least one of the IMR bits should be enabled. If vectoring is to be used, the response memory must be loaded; if not, the mode must be changed to a non-vectored configuration. Normally, the first step will be to modify the Mode register and the Auto clear register in order to establish the configuration desired for the application. Then the response memory and byte count will be loaded for those request levels that will be in use. The response memory for every channel must be written even if the channel is not used. Every byte need not be written only those specified by the byte count. Finally, the master mask bit and at least portions of the IMR will be enabled to allow interrupt processing to proceed.

Commands

The host CPU configures, changes and inspects the internal condition of the Am9519A using the set of commands shown in Figure 4. An "X" entry in the table indicates a "don't care" state. All commands are entered by directly loading the Command register as shown in Figure 3 ($C/\overline{D}$ = 1, $\overline{WR}$ = 0). Figure 5 shows the coding assignments for the Byte Count registers. A detailed description of each command is contained in the Am9519A Application Note AMPUB-071.

BY1	BY0	COUNT
0	0	1
0	1	2
1	0	3
1	1	4

Figure 5. Byte Count Coding.

COMMAND CODE								COMMAND DESCRIPTION
7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	Reset
0	0	0	1	0	X	X	X	Clear all IRR and all IMR bits
0	0	0	1	1	B2	B1	B0	Clear IRR and IMR bit specified by B2, B1, B0
0	0	1	0	0	X	X	X	Clear all IMR bits
0	0	1	0	1	B2	B1	B0	Clear IMR bit specified by B2, B1, B0
0	0	1	1	0	X	X	X	Set all IMR bits
0	0	1	1	1	B2	B1	B0	Set IMR bit specified by B2, B1, B0
0	1	0	0	0	X	X	X	Clear all IRR bits
0	1	0	0	1	B2	B1	B0	Clear IRR bit specified by B2, B1, B0
0	1	0	1	0	X	X	X	Set all IRR bits
0	1	0	1	1	B2	B1	B0	Set IRR bit specified by B2, B1, B0
0	1	1	0	X	X	X	X	Clear highest priority ISR bit
0	1	1	1	0	X	X	X	Clear all ISR bits
0	1	1	1	1	B2	B1	B0	Clear ISR bit specified by B2, B1, B0
1	0	0	M4	M3	M2	M1	M0	Load Mode register bits 0–4 with specified pattern
1	0	1	0	M6	M5	0	0	Load Mode register bits 5, 6 with specified pattern
1	0	1	0	M6	M5	0	1	Load Mode register bits 5, 6 and set mode bit 7
1	0	1	0	M6	M5	1	0	Load Mode register bits 5, 6 and clear mode bit 7
1	0	1	1	X	X	X	X	Preselect IMR for subsequent loading from data bus
1	1	0	0	X	X	X	X	Preselect Auto Clear register for subsequent loading from data bus
1	1	1	BY1	BY0	L2	L1	L0	Load BY1, BY0 into byte count register and preselect response memory level specified by L2, L1, L0 for subsequent loading from data bus

Figure 4. Am9519A Command Summary.

XIMUM RATINGS above which useful life may be impaired

torage Temperature	−65°C to +150°C
Ambient Temperature Under Bias	−55°C to +125°C
/CC with Respect to VSS	−0.5V to +7.0V
All Signal Voltages with Respect to VSS	−0.5V to +7.0V
Power Dissipation (Package Limitation)	1.5W

products described by this specification include internal circuitry designed to protect input devices from damaging accumulations of ic charge. It is suggested, nevertheless, that conventional precautions be observed during storage, handling and use in order to avoid osure to excessive voltages.

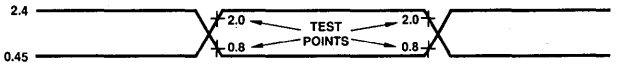
ERATING RANGE

rt Number	Ambient Temperature	V_{CC}	V_{SS}
Am9519APC, Am9519ADC	0°C ≤ T_A ≤ 70°C	5.0V ±5%	0V
Am9519A-1PC, Am9519A-1DC			
Am9519ADI	−40°C ≤ T_A ≤ 85°C	5.0V ±10%	0V
Am9519ADMB	−55°C ≤ T_A ≤ 125°C	5.0V ±10%	0V

DC CHARACTERISTICS Over Operating Range (Note 1)

Parameter	Description	Test Conditions		Min	Typ	Max	Units
VOH	Output High Voltage (Note 12)	IOH = −200μA		2.4			Volts
		IOH = −100μA (EO only)		2.4			
VOL	Output Low Voltage	IOL = 3.2mA				0.4	Volts
		IOL = 1.0mA (EO only)				0.4	
VIH	Input High Voltage			2.0		VCC	Volts
VIL	Input Low Voltage			−0.5		0.8	Volts
IIX	Input Load Current	VSS ≤ VIN ≤ VCC	EI Input	−60		10	μA
			Other Inputs	−10		10	
IOZ	Output Leakage Current	VSS ≤ VOUT ≤ VCC, Output Off		−10		10	μA
ICC	VCC Supply Current	T_A = +25°C			80	125	mA
		T_A = 0°C			100	145	
		T_A = −55°C				200	
CO	Output Capacitance	fc = 1.0MHz				15	pF
CI	Input Capacitance	T_A = 25°C				10	
CIO	I/O Capacitance	All pins at 0V				20	

AC TESTING INPUT, OUTPUT WAVEFORM
Input/Output



00147B-5

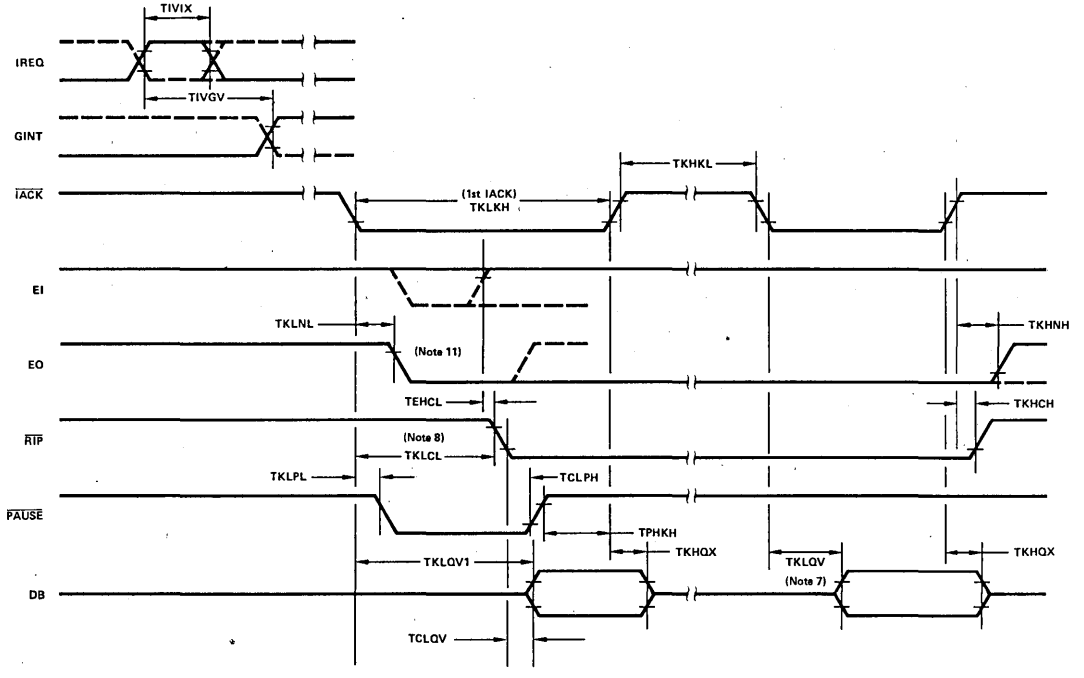
AC CHARACTERISTICS Over Operating Range (Notes 2, 3, 4, 5)

Parameters	Description		Am9519A Min	Am9519A Max	Am9519A-1 Min	Am9519A-1 Max	Units
TAVRL	C/$\overline{D}$ Valid and $\overline{CS}$ LOW to Read LOW		0		0		ns
TAVWL	C/$\overline{D}$ Valid and $\overline{CS}$ LOW to Write LOW		0		0		ns
TCLPH	$\overline{RIP}$ LOW to $\overline{PAUSE}$ HIGH (Note 6)		75	375	75	375	ns
TCLQV	$\overline{RIP}$ LOW to Data Out Valid (Note 7)			50		40	ns
TDVWH	Data In Valid to Write HIGH		250		200		ns
TEHCL	Enable in HIGH to $\overline{RIP}$ LOW (Notes 8, 9)		30	300	30	300	ns
TIVGV	Interrupt Request Valid to Group Interrupt Valid		100	800		650	ns
TIVIX	Interrupt Request Valid to Interrupt Request Don't Care (IREQ Pulse Duration)		250		250		ns
TKHCH	$\overline{IACK}$ HIGH to $\overline{RIP}$ HIGH (Note 8)			450		350	ns
TKHKL	$\overline{IACK}$ HIGH to $\overline{IACK}$ LOW ($\overline{IACK}$ Recovery)		500		300		ns
TKHNH	$\overline{IACK}$ HIGH to EO HIGH (Notes 10, 11)			975		750	ns
TKHQX	$\overline{IACK}$ HIGH to Data Out Invalid		20	200	20	100	ns
TKLCL	$\overline{IACK}$ LOW to $\overline{RIP}$ LOW (Note 8, 13)	COM'L	75	600	75	450	ns
		IND	75	600			ns
		MIL	75	650			ns
TKLKH	$\overline{IACK}$ LOW to $\overline{IACK}$ HIGH (1st IACK) (Note 13)		975		800		ns
TKLNL	$\overline{IACK}$ LOW to EO LOW (Notes 10, 11, 13)			125		100	ns
TKLPL	$\overline{IACK}$ LOW to $\overline{PAUSE}$ LOW (Note 13)		25	175	25	125	ns
TKLQV	$\overline{IACK}$ LOW to Data Out Valid (Note 7, 13)		25	300	25	200	ns
TKLQV1	1st $\overline{IACK}$ LOW to Data Out Valid (Note 13)		75	650	75	490	ns
TPHKH	$\overline{PAUSE}$ HIGH to $\overline{IACK}$ HIGH		0		0		ns
TRHAX	Read HIGH to C/$\overline{D}$ and $\overline{CS}$ Don't Care		0		0		ns
TRHQX	Read HIGH to Data Out Invalid		20	200	20	100	ns
TRLQV	Read LOW to Data Out Valid			300		200	ns
TRLQX	Read LOW to Data Out Unknown		50		50		ns
TRLRH	Read LOW to Read HIGH ($\overline{RD}$ Pulse Duration)		300		250		ns
TWHAX	Write HIGH to C/$\overline{D}$ and $\overline{CS}$ Don't Care		25		25		ns
TWHDX	Write HIGH to Data In Don't Care		25		25		ns
TWHRW	Write HIGH to Read or Write LOW (Write Recovery)		600		400		ns
TWLWH	Write LOW to Write HIGH ($\overline{WR}$ Pulse Duration)		300		250		ns

NOTES:

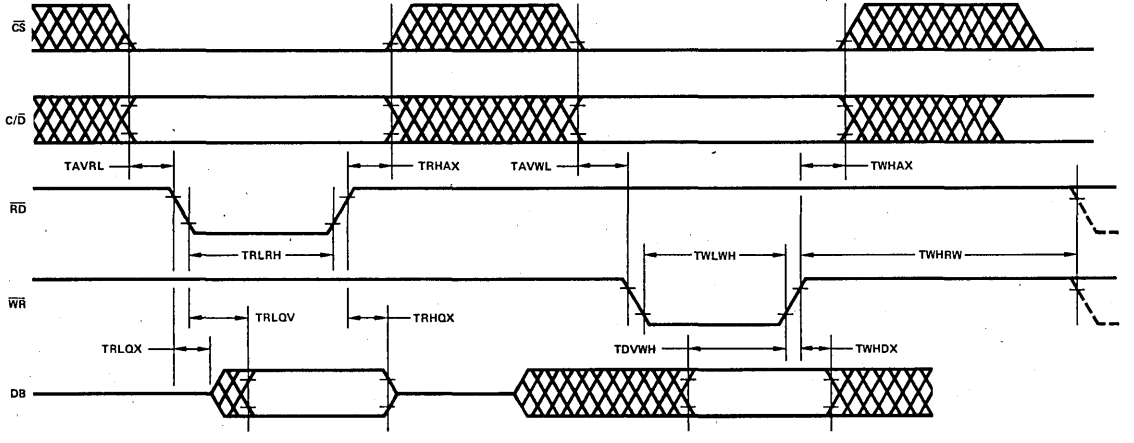
1. Typical values for T_A = 25°C, nominal supply voltage and nominal processing parameters.
2. Test conditions assume transition times of 20ns or less, timing reference levels of 0.8V and 2.0V and output loading of one TTL gate plus 100pF, unless otherwise noted.
3. Transition abbreviations used for the switching parameter symbols include: H = High, L = Low, V = Valid, X = unknown or don't care, Z = high impedance.
4. Signal abbreviations used for the switching parameter symbols include: R = Read, W = Write, Q = Data Out, D = Data In, A = Address (CS and C/D), K = Interrupt Acknowledge, N = Enable Out, E = Enable In, P = Pause, C = $\overline{RIP}$.
5. Switching parameters are listed in alphabetical order.
6. During the first $\overline{IACK}$ pulse, $\overline{PAUSE}$ will be low long enough to allow for priority resolution and will not go high until after RIP goes low (TCLPH).
7. TKLQV applies only to second, third and fourth $\overline{IACK}$ pulses while $\overline{RIP}$ is low. During the first $\overline{IACK}$ pulse, Data Out will be valid following the falling edge of $\overline{RIP}$ (TCLQV).
8. $\overline{RIP}$ is pulled low to indicate that an interrupt request has been selected. $\overline{RIP}$ cannot be pulled low until EI is high following an internal delay. TKLCL will govern the falling edge of $\overline{RIP}$ when EI is always high or is high early in the acknowledge cycle. TEHCL will govern when EI goes high later in the cycle. The rising edge of EI will be determined by the length of the preceding priority resolution chain. $\overline{RIP}$ remains low until after the rising edge of the $\overline{IACK}$ pulse that transfers the last response byte for the selected IREQ.
9. Test conditions for the EI line assume timing reference levels of 0.8V and 2.0V with transition times of 10ns or less.
10. Test conditions for the EO line assume output loading of two LS TTL gates plus 30pF and timing reference levels of 0.8V and 2.0V. Since EO normally only drives EI of another Am9519A, higher speed operation can be specified with this more realistic test condition.
11. The arrival of $\overline{IACK}$ will cause EO to go low, disabling additional circuits that may be connected to EO. If no valid interrupt is pending, EO will return high when EI is high. If a pending request is selected. EO will stay low until after the last $\overline{IACK}$ pulse for that interrupt is complete and $\overline{RIP}$ goes high.
12. VOH specifications do not apply to $\overline{RIP}$ or to $\overline{GINT}$ when active-low. These outputs are open-drain and VOH levels will be determined by external circuitry.
13. $\overline{CS}$ must be High for at least 100ns prior to $\overline{IACK}$ going Low.

SWITCHING WAVEFORMS



Interrupt Operations

00147B-6



Data Bus Transfers

00147B-7

APPLICATIONS

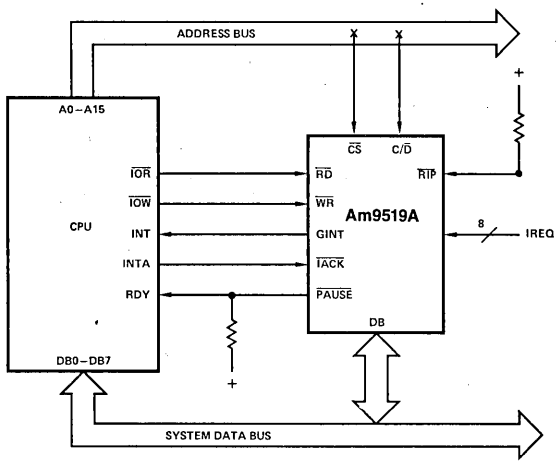


Figure 6. Base Interrupt System Configuration. 00147B-8

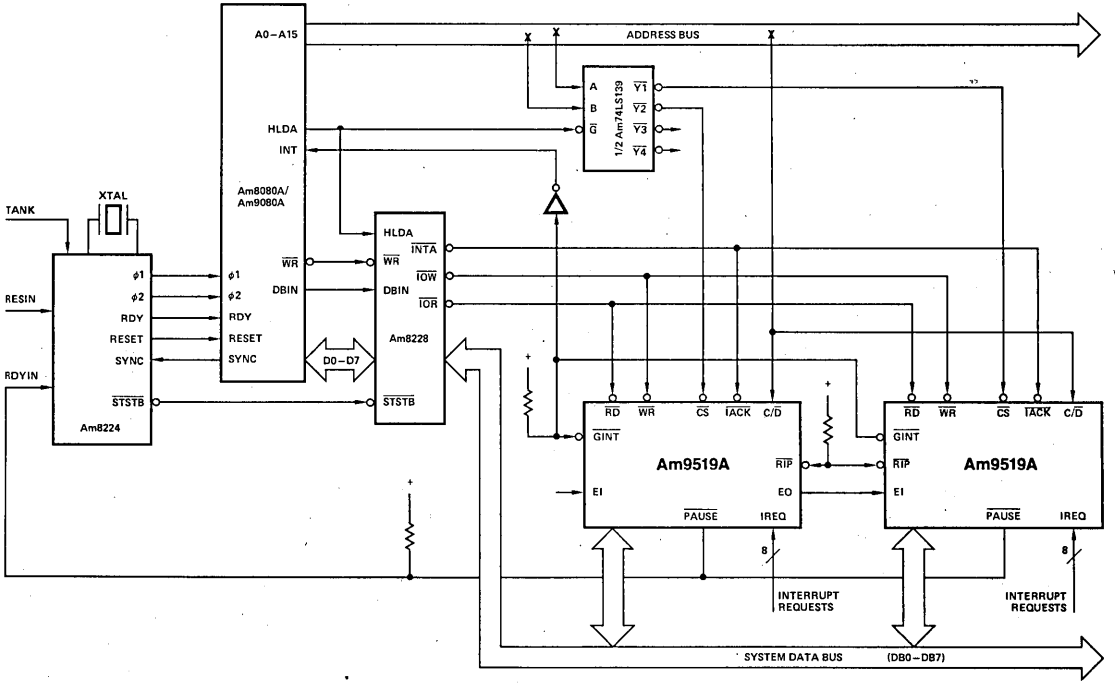


Figure 7. Expanded Interrupt System Configuration. 00147B-9

General Information Index

Z-Bus/68000 Microprogrammable Bus Translator

Anthony DiColli
Advanced Micro Devices

04193A-MMP

ABSTRACT

This paper describes an interface technique that permits all speed versions of the 68000 CPU to communicate with all Z8000 peripherals. Further, the microprogrammable nature of this interface allows intermixing of various speed peripherals on the same Z-Bus by dynamically modifying the bus translator's timing characteristics on a cycle by cycle basis. Included are a circuit description, PROM programs, PAL equations and a discussion on a typical systems architecture.

BUS TRANSLATOR CIRCUIT DESCRIPTION (Figures 1 and 2)

The bus multiplexer contains one 8 bit bus buffer (Am29827) and two bus transceivers (Am29863). These components accept the separate address and data buses of the 68000 and using three-state techniques, multiplexes them together to form the Z-Bus address/data path (AD Bus). Only six address lines are required to directly address all control and data registers in the Z8000 peripherals. The timing sequence of this multiplexing operation is derived from the state sequencer and the translator logic.

The translator logic contains hard-wired logic elements that accept 68000 bus control signals (CLK, $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$) as inputs and in conjunction with the state sequencer inputs (T_1, T_2, T_3, T_4, T_5), produces Z-Bus control signals ($\overline{AS}$, $\overline{DS}$, A_0, R/$\overline{W}$, B/$\overline{W}$), 68000 $\overline{DTACK}$, and the bus multiplexer control signals.

In addition, a Z8000 peripheral inhibit input (ZINH) is provided to prevent the bus translator from responding to 68000 bus operations when none of the Z8000 peripherals are being accessed (i.e., a 68000 instruction fetch). Also, a translator output enable ($\overline{TOE}$) is provided, so that control of the Z-Bus can be relinquished to another master, such as a DMA Controller, if required (see Figure 9). Both $\overline{TOE}$ and $\overline{ZINH}$ are generated by an external chip select decoder (see systems architecture section). This translator logic can easily be implemented utilizing a PAL (AmPAL16R4) as shown in Figure 3. The fuse map for this PAL is detailed in Figure 4 and the design equations are:

AOE = /T_2 * MAS
ZAS = /T_1 * MAS
DOE := T_3 * MUDS + T_3 * MLDS
ZDS := T_4 * MUDS + T_4 * MLDS
Z$\overline{B}$/W:= /MUDS :+: /MLDS
MDTACK: = T_5 * MUDS + T_5 * MLDS
/ZA_0 + MLDS
;when $\overline{TOE}$ is LOW

The state sequencer, shown in Figure 2, is a registered PROM (Am27S35) which provides microprogramming ability for the bus translator. The registered PROM contains a fusible-link PROM memory array, an output register, and a (user-programmable) initialize word (Figure 5). When $\overline{INIT}$ is pulled LOW, the contents of the initialize word is sent to the output, regardless of the state of the clock or the address inputs. In this application, the initialize word is programmed to OOH. Outputs O_0 through O_4 generate timing signals T_1 through T_5, which are used as gating inputs by the translator logic. These gating signals along with the 68000 control signals determine when the Z-Bus control signals will be activated and the duration of these signals. The program sequences stored in this PROM are user definable and are a function of the CPU and peripheral types and speeds implemented in the system.

The next address to be accessed is determined by the clock strobed data outputs, O_0 through O_7. In order for this to work properly, a unique and non-redundant output word must exist for every clock cycle. Since there will be instances where T_1 through T_5 will not change for many clock cycles, a "counter" function must be included via O_5, O_6 and O_7. This counter function allows up to 8 idle states (83.3ns per idle state) to be inserted between T_1 through T_5 transitions. Address lines A_8 and A_9 provide the ability to dynamically select up to 4 individual sequences. Figure 6 is a PROM program required to generate the read and write cycles shown in Figure 8. When A_8 is LOW, a read cycle is selected, with A_8 in the HIGH state, a write cycle is selected; with A_8 LOW and A_9 HIGH, a "read interrupt" cycle is selected. Other possible sequences would support intermixing of 4 and 6MHz peripherals on a cycle by cycle basis.

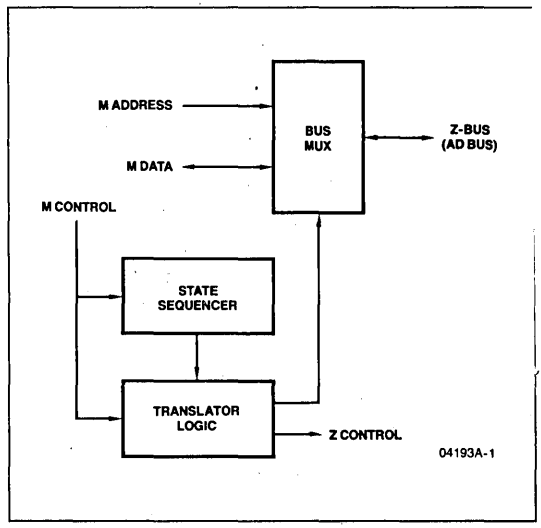


Figure 1.

This state sequencer design assumes that a minimum chip count is preferred and that PROM space is inexpensive. In the example of Figure 6, only 29 locations out of 1,000 are used. If smaller memories are preferred and chip count is not critical, then the state sequencer in Figure 7 can be substituted. In this example a hardware counter is provided so that no "counter" function is required in the PROM. Also, the PROM array need only be 64 words deep. This approach also allows more sequences to be added without drastically increasing the PROM array size. For instance, a separate read and write cycle for 4 and 6MHz peripherals, a separate read interrupt vector for 4 and 6MHz peripherals, and a two speed CPU cycle would require only 256 words of PROM space.

It should be noted that the address inputs, used to select the sequence to be enabled, are generated by the same external chip select decoder that generates $\overline{ZINH}$.

BUS TRANSLATOR TIMING ANALYSIS

Figure 8 illustrates the timing relationships between a 12MHz 68000 and all 4MHz Z8000 peripherals. The signals shown are the 68000 control, address and data inputs to the bus translator of Figure 2 and the corresponding Z-Bus control, address and data outputs. The following discussion assumes that the 68000 CPU, bus translator and Z8000 peripherals are physically located on the same circuit board. If a backplane is used, its propagation delays must also be considered, however the setup and hold times in this illustration will be sufficient for most system architectures.

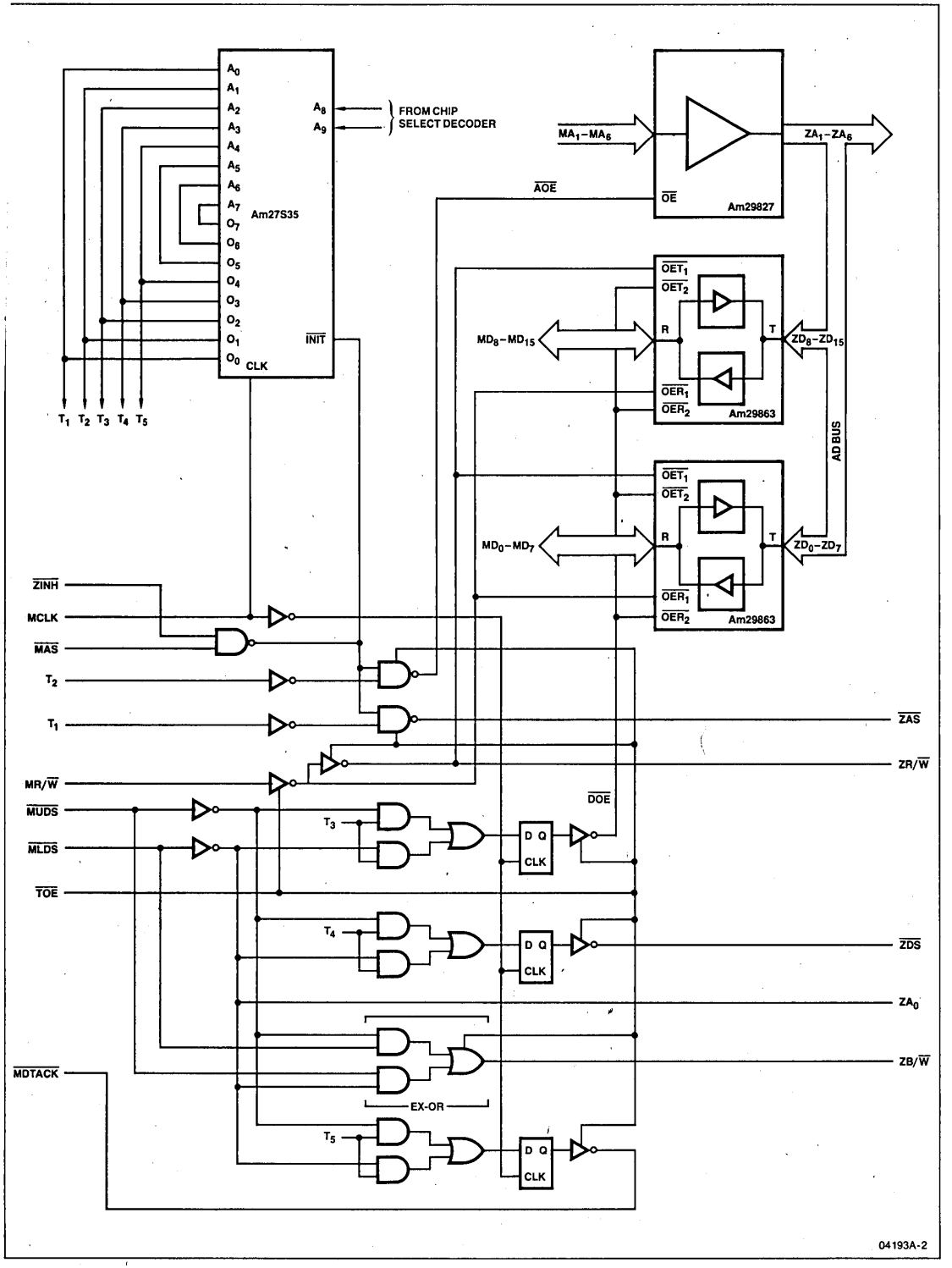

A0
A1
A2
A3
A4
A5
A6
A7
O7
O6
O5
O4
O3
O2
O1
O0
CLK
Am27S35
A8
A9
FROM CHIP SELECT DECODER
INIT
T1 T2 T3 T4 T5
MA1–MA6
ZA1–ZA6
AOE
OE
Am29827
OET1
OET2
R
T
MD8–MD15
ZD8–ZD15
OER1
OER2
Am29863
AD BUS
MD0–MD7
ZD0–ZD7
ZINH
MCLK
MAS
T2
T1
MR/W
MUDS
MLDS
TOE
T3
T4
T5
D Q
CLK
DOE
ZAS
ZR/W
ZDS
ZA0
ZB/W
MDTACK
EX-OR
04193A-2

Figure 2.

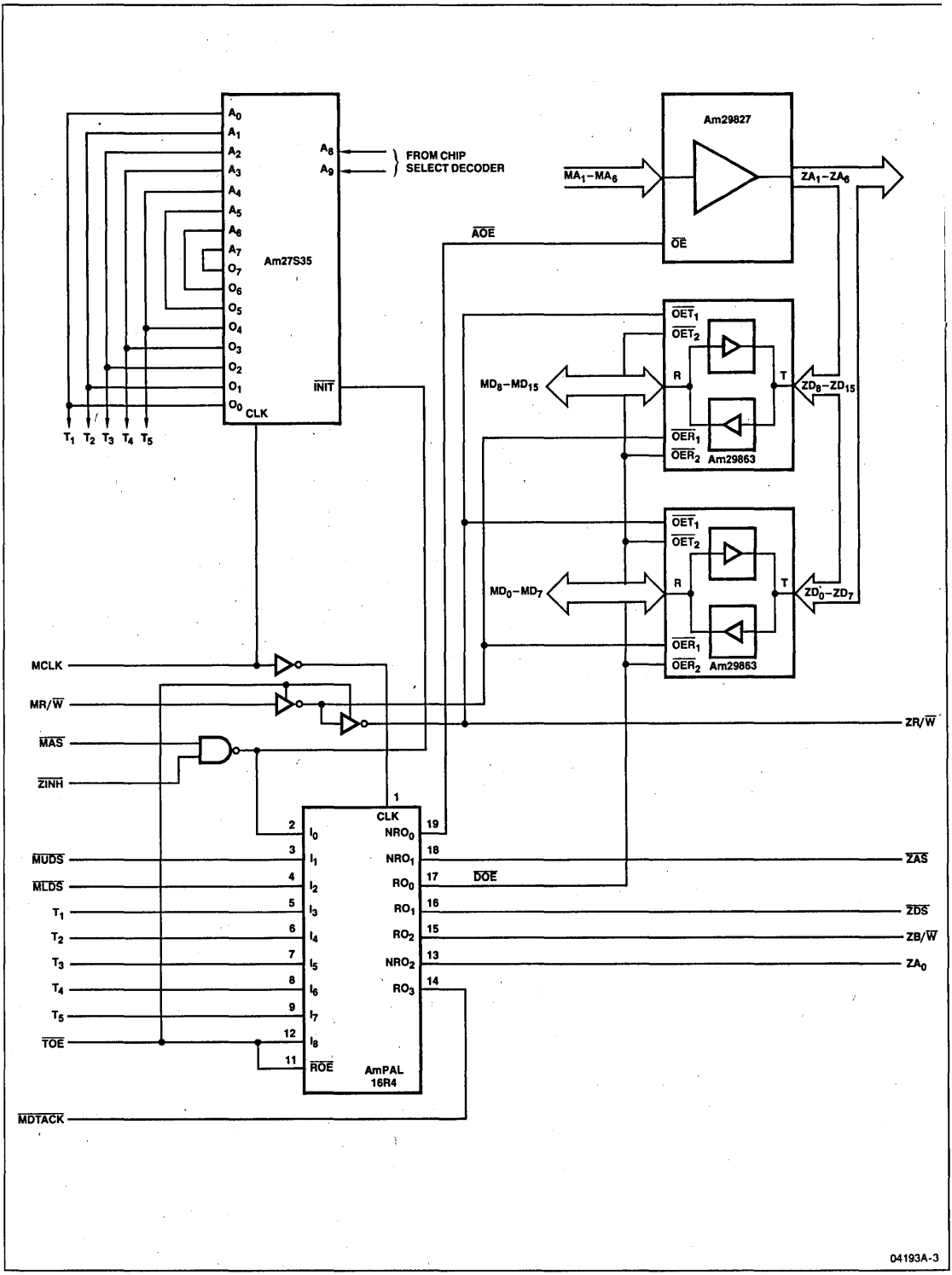

Am27S35
FROM CHIP SELECT DECODER
Am29827
Am29863
Am29863
AmPAL 16R4
MCLK
MR/W
MAS
ZINH
MUDS
MLDS
TOE
MDTACK
ZR/W
ZAS
ZDS
ZB/W
ZA0
04193A-3

Figure 3.

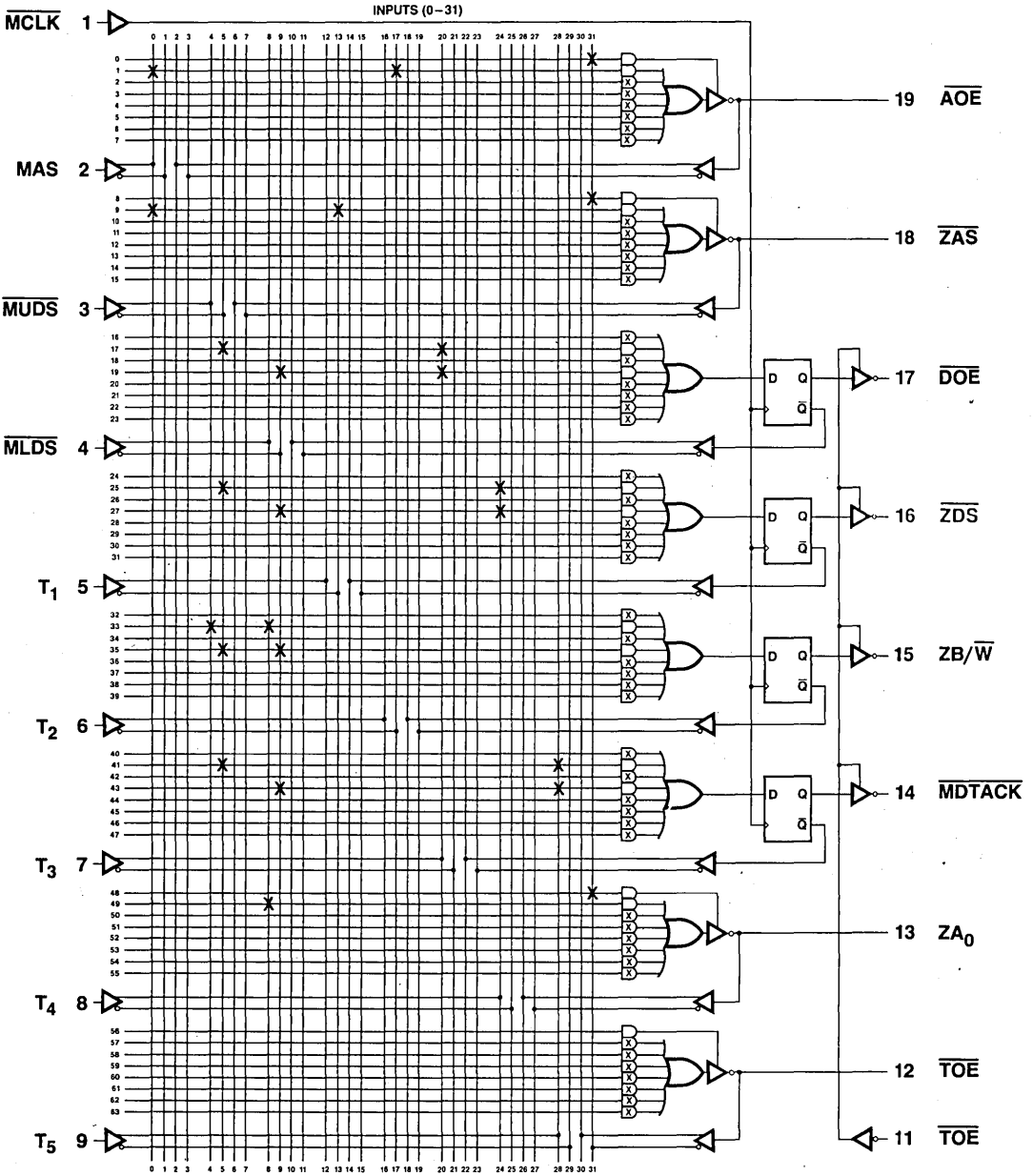


Figure 4. Logic Diagram PAL16R4

During S_0 the read/write line from the 68K will be set HIGH to indicate a read operation. MR/$\overline{W}$ puts the transceivers in the receive mode and is used by the Z-Bus as ZR/$\overline{W}$. In a read operation, the 68K address will become valid sometime during S_1, and the external chip select decoder will read this address and determine whether a Z8000 I/O operation is being started. If it is a Z8000 I/O operation, $\overline{ZINH}$, A_8,and A_9 will be appropriately set. $\overline{MAS}$ will be asserted during S_2. This action: 1) releases the state sequencer via $\overline{INIT}$ and begins the Tx state on the next positive edge of MCLK (shown as 0_1), 2) causes $\overline{AOE}$ to go active, thus driving the AD bus with the address, and 3) asserts $\overline{ZAS}$. $\overline{MUDS}$ and/or $\overline{MLDS}$ are also asserted, in S_2, and causes ZA_0 and ZB/W to be appropriately set. At 0_2 the state sequencer asserts T_1, causing ZAS to be negated. This phase was chosen to meet the required 70ns (min) ZAS pulse width (TwAS), and 30ns (min) address to $\overline{ZAS}$ setup time (TsA (AS)). At 0_3 the state sequencer asserts T_2 and T_4. T_2 causes the $\overline{AOE}$ line to negate which removes the address and three-states the AD bus. This meets the required 50ns (min) address to $\overline{ZAS}$ hold time (ThA (AS)). T_4 causes the $\overline{ZDS}$ to be asserted on the negative edge of 0_3. This meets the $\overline{ZAS}$ to $\overline{ZDS}$ delay of 60ns (min).

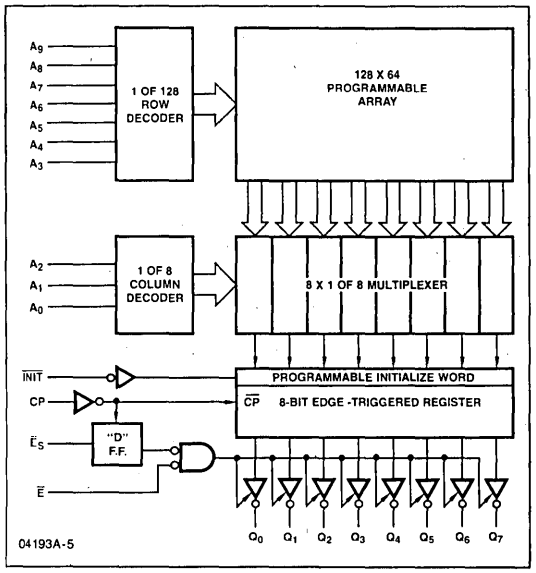


Figure 5. Am27S35 Block Diagram

On the falling edge of $\overline{ZDS}$ the peripheral will drive the AD bus and data on the AD bus will be valid 250ns later (an exception to this occurs in the Z8030 which requires 520ns from rising edge of $\overline{ZAS}$ to valid read data). At 0_5 the sequencer sets T_5, which in turn causes $\overline{MDTACK}$ to be asserted on the negative edge of 0_5. The 68K samples this line on the negative edge of 0_6 and accepts the input data on the negative edge of 0_7. At 0_6 the T_3 line is asserted, which activates $\overline{DOE}$ on the negative edge of 0_6. This enables AD bus data to the 68K data bus. T_3 may be set to occur anytime before the negative edge of 0_7. During S_7 the 68K will negate $\overline{MAS}$, $\overline{MUDS}$, and $\overline{MLDS}$. This action causes the sequencer to reset (via $\overline{INIT}$), and on the negative edge of 0_8 $\overline{ZDS}$, $\overline{MDTACK}$, and $\overline{DOE}$ will be negated to end the read cycle. Note that the peripheral provides the zero data hold time required by the 68000. It should also be noted that $\overline{DTACK}$ is used to insert wait states (2 wait states = 1 MCLK). The assertion of $\overline{DTACK}$ via T_5 is a function of the minimum required $\overline{ZDS}$ pulse width (in this case 390ns), and the minimum time required for the peripheral to provide valid data to the AD bus. Therefore, in using the Z8030, T_5 would not occur (this is accomplished by inserting 2 idle states in the sequencer) until 0_7, and the rest of the read cycle would be proportionately extended.

			ADDR INPUT											PROM OUTPUT									REG. OUTPUT								
$\overline{INIT}$	CLOCK		A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	HEX	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	HEX	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	HEX
0	X		0	0	0	0	0	0	0	0	0	0	000	0	0	0	0	0	1	0	0	20	0	0	0	0	0	0	0	0	00
1	0_1		0	0	0	0	0	1	0	0	0	0	020	1	0	0	0	0	0	0	0	01	0	0	0	0	0	1	0	0	20
1	0_2		1	0	0	0	0	0	0	0	0	0	001	1	1	0	1	0	0	0	0	0B	1	0	0	0	0	0	0	0	01
1	0_3		1	1	0	1	0	0	0	0	0	0	00B	1	1	0	1	0	1	0	0	2B	1	1	0	1	0	0	0	0	0B
1	0_4		1	1	0	1	0	1	0	0	0	0	02B	1	1	0	1	1	0	0	0	1B	1	1	0	1	0	1	0	0	2B
1	0_5		1	1	0	1	1	0	0	0	0	0	01B	1	1	1	1	1	0	0	0	1F	1	1	0	1	1	0	0	0	1B
1	0_6	READ CYCLE	1	1	1	1	1	0	0	0	0	0	01F	1	1	1	1	1	0	0	0	1F	1	1	1	1	1	0	0	0	1F
0	X		0	0	0	0	0	0	0	0	1	0	100	0	0	0	0	0	1	0	0	20	0	0	0	0	0	0	0	0	00
1	0_1		0	0	0	0	0	1	0	0	1	0	120	1	0	0	0	0	0	0	0	01	0	0	0	0	0	1	0	0	20
1	0_2		1	0	0	0	0	0	0	0	1	0	101	1	1	1	0	0	0	0	0	07	1	0	0	0	0	0	0	0	01
1	0_3		1	1	1	0	0	0	0	0	1	0	107	1	1	1	1	0	0	0	0	0F	1	1	1	0	0	0	0	0	07
1	0_4		1	1	1	1	0	0	0	0	1	0	10F	1	1	1	1	0	1	0	0	2F	1	1	1	1	0	0	0	0	0F
1	0_5		1	1	1	1	0	1	0	0	1	0	12F	1	1	1	1	0	0	1	0	4F	1	1	1	1	0	1	0	0	2F
1	0_6	WRITE CYCLE	1	1	1	1	0	0	1	0	1	0	14F	1	1	1	1	0	1	1	0	6F	1	1	1	1	0	0	1	0	4F
1	0_7		1	1	1	1	0	1	1	0	1	0	16F	1	1	1	1	1	0	0	0	1F	1	1	1	1	0	1	1	0	6F
1	0_8		1	1	1	1	1	0	0	0	1	0	11F	1	1	1	0	1	0	0	0	17	1	1	1	1	1	0	0	0	1F
1	0_9		1	1	1	0	1	0	0	0	1	0	117	1	1	1	0	1	0	0	0	17	1	1	1	0	1	0	0	0	17
0	X		0	0	0	0	0	0	0	0	0	1	200	0	0	0	0	0	1	0	0	20	0	0	0	0	0	0	0	0	00
1	0_1		0	0	0	0	0	1	0	0	0	1	220	1	0	0	0	0	0	0	0	01	0	0	0	0	0	1	0	0	20
1	0_2		1	0	0	0	0	0	0	0	0	1	201	1	1	0	0	0	0	0	0	03	1	0	0	0	0	0	0	0	01
1	0_3		1	1	0	0	0	0	0	0	0	1	203	1	1	0	0	0	1	0	0	23	1	1	0	0	0	0	0	0	03
1	0_4		1	1	0	0	0	1	0	0	0	1	223	1	1	0	0	0	1	0	0	43	1	1	0	0	0	1	0	0	23
1	0_5		1	1	0	0	0	1	0	0	0	1	243	1	1	0	0	0	1	1	0	63	1	1	0	0	0	0	1	0	43
1	0_6		1	1	0	0	0	1	1	0	0	1	263	1	1	0	0	0	0	0	1	83	1	1	0	0	0	0	1	1	63
1	0_7		1	1	0	0	0	0	0	1	0	1	283	1	1	0	0	0	1	0	1	A3	1	1	0	0	0	0	0	1	83
1	0_8		1	1	0	0	0	1	0	1	0	1	2A3	1	1	0	0	0	0	1	1	C3	1	1	0	0	0	1	0	1	A3
1	0_9		1	1	0	0	0	0	1	1	0	1	2C3	1	1	0	0	0	1	1	1	E3	1	1	0	0	0	0	1	1	C3
1	0_{10}	READ INTERRUPT VECTOR CYCLE	1	1	0	0	0	1	1	0	0	1	2E3	1	1	0	1	0	0	0	0	0B	1	1	0	0	0	1	1	1	E3
1	0_{11}		1	1	0	1	0	0	0	0	0	1	20B	1	1	0	1	0	1	0	0	2B	1	1	0	1	0	0	0	0	0B
1	0_{12}		1	1	0	1	0	1	0	0	0	1	22B	1	1	0	1	1	0	0	0	1B	1	1	0	1	0	1	0	0	2B
1	0_{13}		1	1	0	1	1	0	0	0	0	1	21B	1	1	1	1	1	0	0	0	1F	1	1	0	1	1	0	0	0	1B

04193A-6

Figure 6.

The write cycle operates in a similar manner with two exceptions. First, there is a required data setup time of 30ns with respect to the falling edge of $\overline{ZDS}$ (TsDW (DSF)). Therefore, one sequencer idle state is required between T_2 and T_4. Secondly, there is a required data hold time of 30ns with respect to the rising edge of $\overline{ZDS}$ (ThDW(DS)). Therefore T_5 is asserted at 0_8 while T_4 is negated at 0_9. If bidirectional registers are substituted for the data transceivers, four wait states can be eliminated from the write cycle.

TIMING PARAMETERS

During a read interrupt vector cycle, timing parameters are the same as a normal read operation with the exception of the $\overline{ZAS}$ to $\overline{ZDS}$ delay. In a normal read this parameter is 60ns. However, during an interrupt this parameter (TdDCST) must be equal to or greater than the interrupt daisy chain settling time. When five Z8000 peripherals are in this chain, this parameter is approximately 710ns. To accommodate this requirement, 8 idle states must be inserted between T_2 and T_4 by the state sequencer. The state sequencer detects an interrupt operation via A_9 (reference Figure 6).

SYSTEM ARCHITECTURE

Figure 9 illustrates an architecture that allows the 68K to perform other tasks while an I/O operation is in progress. The ZI/O chip select decoder accepts:

1) CPU Status – indicates an interrupt read cycle is required. This activates RIV (Read Interrupt Vector).
2) R/W – indicates that the current cycle is a read or a write. This activates I/OR/$\overline{W}$.
3) MAn – A user-definable number of address lines (6 lines minimum) that identifies which peripheral is being addressed. This activates the appropriate $\overline{CS}$ line, ZI/O REQ, $\overline{ZINH}$ and 4/6MHz.

The ZI/O bus arbiter is used to determine who has I/O bus control and is only required if a DMA controller is part of the Z peripheral chain.

It accepts

1) ZI/O REQ – indicates the CPU is requesting the I/O bus.
2) DMA BUS REQ – indicates the DMA controller is requesting the I/O bus.
3) M BUS REQ – Allows the 68K to request the I/O bus before the I/O cycle is started.

and generates

1) ZBUS ERR – flags the 68K when the CPU starts a ZI/O operation and the I/O bus is busy.
2) ZI/O BUS BUSY – A Z-bus status line that can be polled by the CPU.
3) DMA BUS ACQ – grants control of the ZI/O bus to the DMA controller.
4) $\overline{TOE}$ – grants control of the ZI/O bus to the Bus Translator.

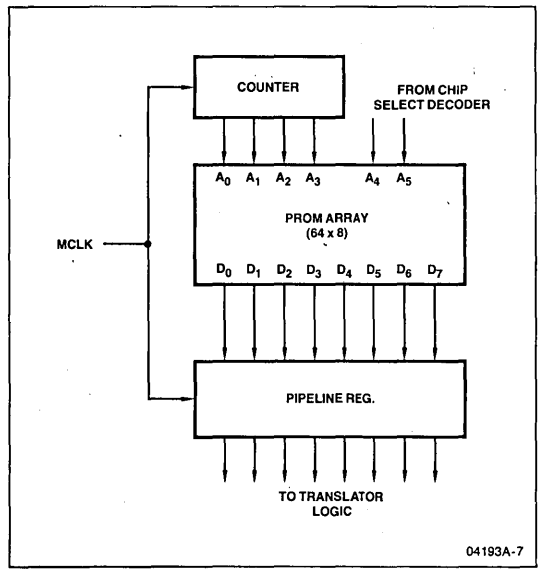


Figure 7.

This type of architecture is useful in I/O intensive applications where the data must be operated on concurrently by the CPU (i.e., PBX systems, disk controllers, etc.). Data is received and transmitted by this system via the DMA controller. Transmitted data is loaded into the Buffer Memory by the CPU and is then transmitted via a DMA operation. Received data is loaded into the Buffer Memory via a DMA operation and is then read by the CPU. In extreme cases this Buffer Memory can be dual ported so that true concurrent operation between CPU and I/O is achieved. In single ported Buffer Memory architectures, a Bus Arbiter is required to grant ZI/O bus control to CPU or DMA controller. In this example, the CPU can poll the arbiter for ZI/O bus control and lock out the DMA controller until the CPU has completed its I/O operation. Alternately, the CPU can start a ZI/O bus operation, and if the bus is busy, a bus error is generated. The operation of this arbiter is straightforward and can be constructed using a minimal number of gates and flip-flops.

The I/O chip select decoder is a combination chip selector and memory mapper. Note that the 4MHz/6MHz signal is used by the bus translator to allow intermixing of slow and fast peripheral chips.

The DMA controller is a Z8016 which utilizes "link-lists" to update its control registers. This further reduces the need for CPU intervention, thus increasing system performance.

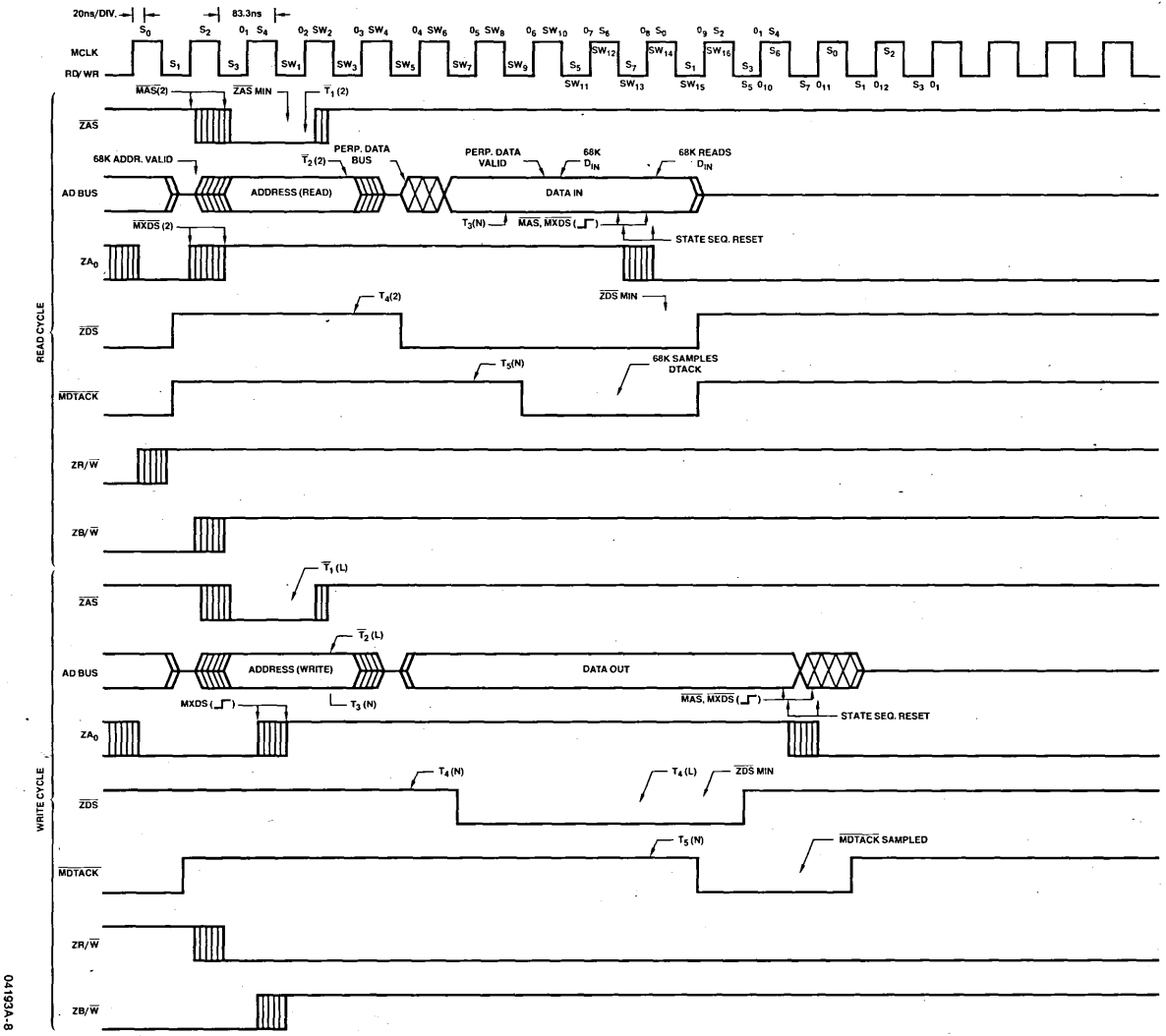


Figure 8. 12MHz 68000 to 4MHz Peripherals Timing Relationships

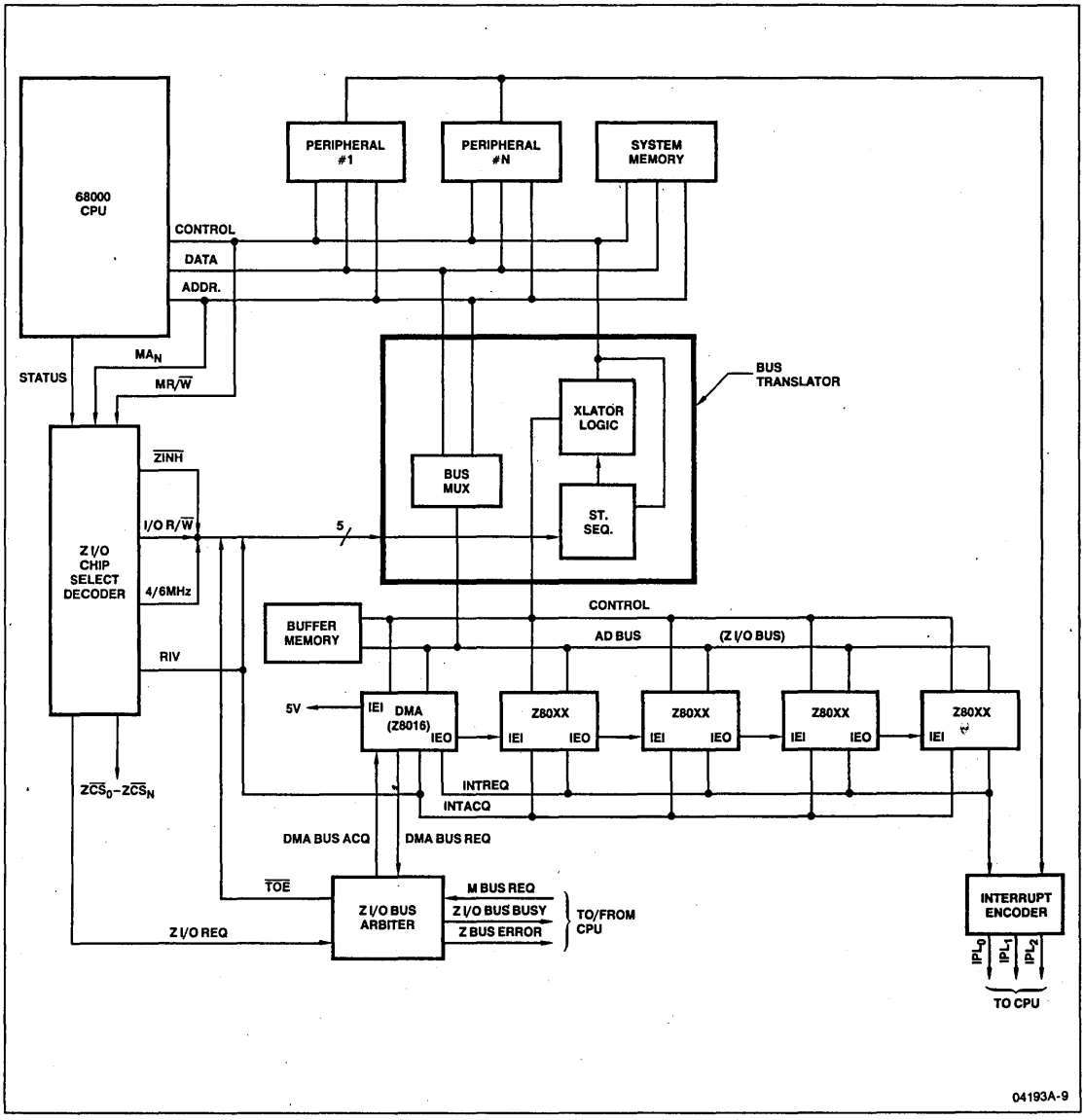


Figure 9.

CONCLUSIONS

This circuit is a highly efficient and flexible bus translator that allows Z8000 peripherals and a 68000 CPU to intercommunicate in a uniform manner; the same technique is also applicable to other CPU's such as iAPX86, LSI-11 and N16000.

Interface Standards for Peripherals

Advanced Micro Devices

INTRODUCTION

AMD manufactures a large number of microprocessor peripherals, and because of the large number of CPU types available it is desirable to have a standard bus interface specification.

Since it is impossible for our designers to know the characteristic of every processor and its subtleties, this document is intended to provide guide lines based on a study of the 8086, Z8000, and 68000. One of the goals has been to specify setup and hold times such that peripherals will work with both fast and slow CPU's. The only requirement is the insertion of a wait state in some cases.

PIN DESCRIPTIONS (Not all will be used in all devices)

AD_0-AD_N (Data Bus, Input/Output)

The bidirectional Data Bus lines are used for information exchanges with the host processor. HIGH on a Data Bus line corresponds to one and LOW corresponds to zero. When operating in the slave mode these lines act as inputs when $\overline{WR}$ and $\overline{CS}$ are active and as outputs when $\overline{RD}$ and $\overline{CS}$ are active. When $\overline{CS}$ is inactive, these pins are placed in a high-impedance state. For Bus Masters, data is driven onto these lines when $\overline{WR}$ is active and read from these lines when $\overline{RD}$ is active.

$\overline{CS}$ (Chip Select, Input)

The active-LOW Chip Select input enables Read and Write operations to the peripheral. When Chip Select is HIGH, the Read and Write inputs are ignored. The circuit should be designed so that $\overline{CS}$ may be tied LOW. This input is ignored when the peripheral is in control of the Bus (i.e. Bus Master).

C/$\overline{D}$ (Control/Data, Input)

The Control-Data signal selects destination locations for Read and Write operations to the peripheral. Control Write operations load the Command register and the Data Pointer, while Control Read operations make the Status register output available. Data Read and Data Write transfers communicate with all other internal registers. Indirect addressing at the data port is controlled internally by the Data Pointer register. It is important to make the Data Pointer readable as this simplifies interrupt service routines (see Figure 2).

CLK (Clock/Input)

This input is required on all Bus Masters and is optional on Bus Slaves. It controls the internal operations of the device and its rate of data transfers.

$\overline{RESET}$ (Reset, Input)

Reset is an asynchronous active LOW input which initializes the registers and state machine internally to a state at which it can respond to CPU commands. This signal is optional on slave peripherals but is required on Bus Masters. All peripherals should also have a software reset command, capable of resetting the peripheral even if no hardware reset occurs.

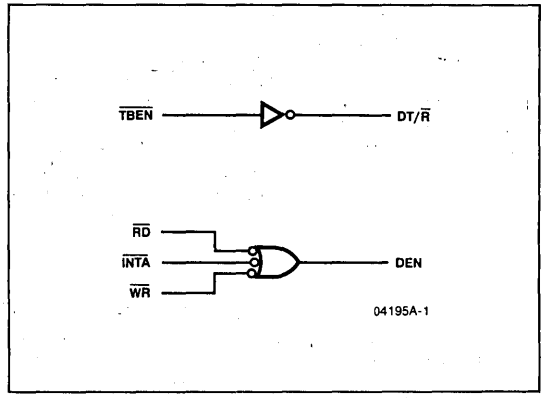


Figure 2b. Implementing DT/$\overline{R}$ and DEN

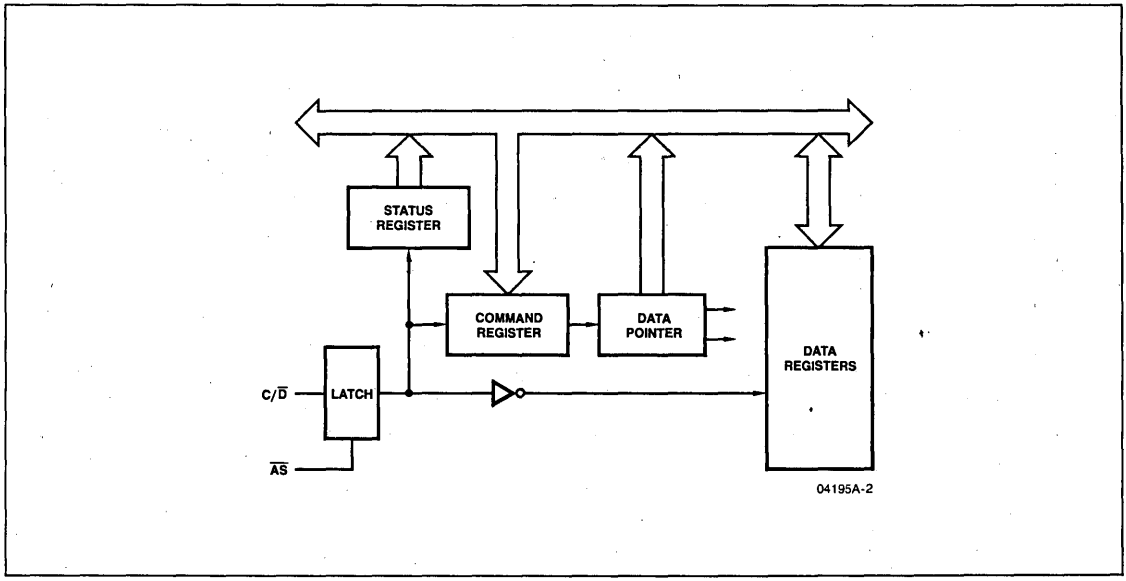


Figure 2a. C/$\overline{D}$ Operation

$\overline{AS}$: Address Strobe (Input), (Output), or (Input, Output)

LOW on this pin indicates the bus contains address information. The address information is stable by the time of the LOW-to-HIGH transition of the $\overline{AS}$ output (see timing diagrams). This is driven by the peripheral when it is a Bus Master to indicate when the address is on the bus. When in the slave mode this signal is a don't care. A Bus Master which does not use $\overline{AS}$ as an input now has the option of programming its polarity. Another possible variation is to use $\overline{AS}$ to latch $\overline{CS}$, C/$\overline{D}$, or R/$\overline{W}$, which reduces external logic in systems with multiplexed buses. When not used in multiplexed address data systems, this pin can be grounded to make R/$\overline{W}$, C/$\overline{D}$ and $\overline{CS}$ transparent. See Figure 2a.

$\overline{DS}$: Data Strobe (Input, Output)

LOW on this pin indicates that the AD bus is being used for data transfer. When operating as Bus Master the R/$\overline{W}$ output indicates the direction of data transfer – Read (or in) means data into the peripheral and Write (or out) means data from the peripheral. During a read operation, data can be gated on to the bus when $\overline{DS}$ goes LOW. A LOW-to-HIGH transition on the $\overline{DS}$ output indicates that the peripheral has accepted the data (see timing diagram). During a write operation, LOW on the $\overline{DS}$ output indicates that data is setup on the bus. Data will be removed sometime after the LOW-to-HIGH transition of the $\overline{DS}$ output (see timing diagram). When operating as Bus Slave, Read (or out) means the peripheral gates data onto the data bus when $\overline{DS}$ is LOW. Write (or in) means the peripheral will accept data from the bus on the LOW-to-HIGH transition of Data Strobe.

R/$\overline{W}$: Read/Write (Input, Output)

When the peripheral is Bus Master, this output indicates the direction of data flow on the AD bus. HIGH indicates a read operation while LOW indicates a write. This output is activated at the same time as $\overline{AS}$ goes LOW and remains stable for the duration of the whole transaction (see timing diagram). When the peripheral is a slave, HIGH indicates data out of the peripheral and a LOW indicates data into the peripheral.

$\overline{RBEN}$ Receive Buffer Enable (Output, Open Drain)

When the peripheral is Bus Master, $\overline{RBEN}$ is driven LOW when doing a read and it is inactive when doing a write. The purpose of this signal is to eliminate bus contention. When the peripheral is a slave, $\overline{RBEN}$ will be inactive. See timing diagram.

$\overline{TBEN}$ Transmit Buffer Enable (Output, Open Drain)

When the peripheral is Bus Master, $\overline{TBEN}$ is driven LOW when doing a write. It is inactive during reads. This signal is also used to eliminate bus contention and EMI. See timing diagram.

$\overline{RBEN}$ and $\overline{TBEN}$ provide control of all transceivers in the system in such a way that no bus contention occurs. In some systems, contention results as a result of propagation delays in a cable or through the buffers/transceivers control logic.

Note that $\overline{TBEN}$ is generated early so that it can function as a R/$\overline{W}$ indication as well as buffer control. Although $\overline{TBEN}$ goes LOW during the address portion of the cycle in the LANCE Chip, it is not required in general as only the 8030 and 8036 require addresses to be available during the first part of the cycle. $\overline{TBEN}$ cannot be used to generate DT/$\overline{R}$ in the case of the LANCE Chip. Another reason for not perpetuating this form of $\overline{TBEN}$ is that our address period and ALE ($\overline{AS}$) are too short for easy interfacing when buffer delays and skews are accounted for.

When the peripheral is a slave and $\overline{CS}$ is active, $\overline{TBEN}$ will be LOW when the CPU is reading from the peripheral. See timing diagram.

$IRA_1 - IRA_K$ Internal Register Address

These optional address pins may provide for direct addressing of registers in a peripheral in lieu of using C/$\overline{D}$ and setting a data pointer. This is always the more desirable approach but due to pin limitations is less often implemented.

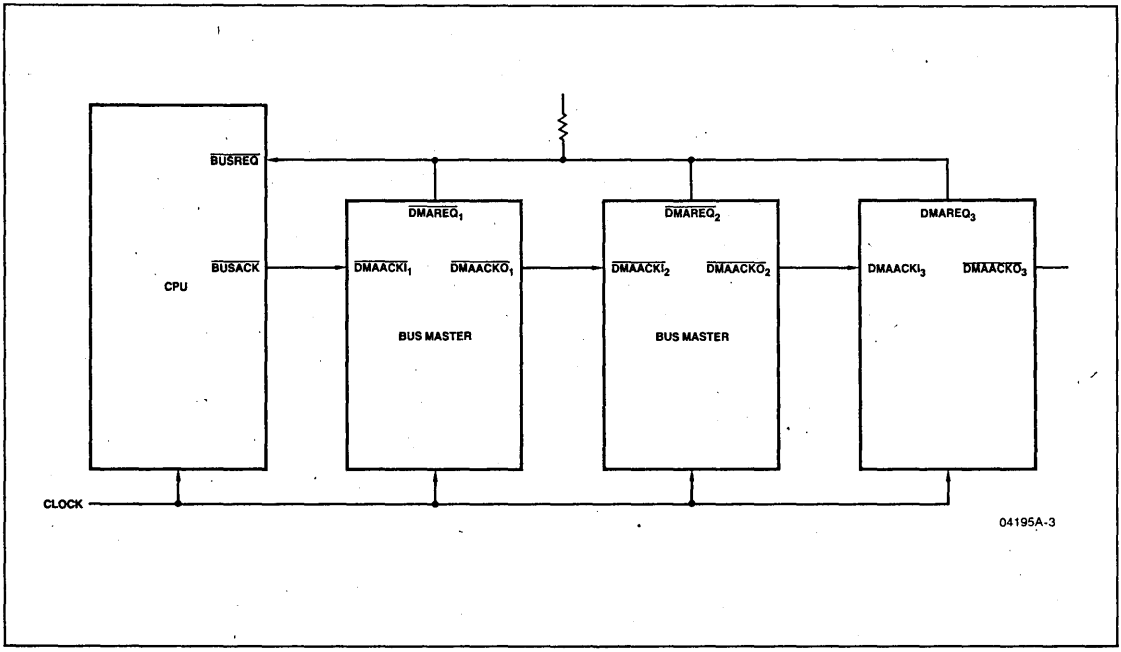


Figure 3a.

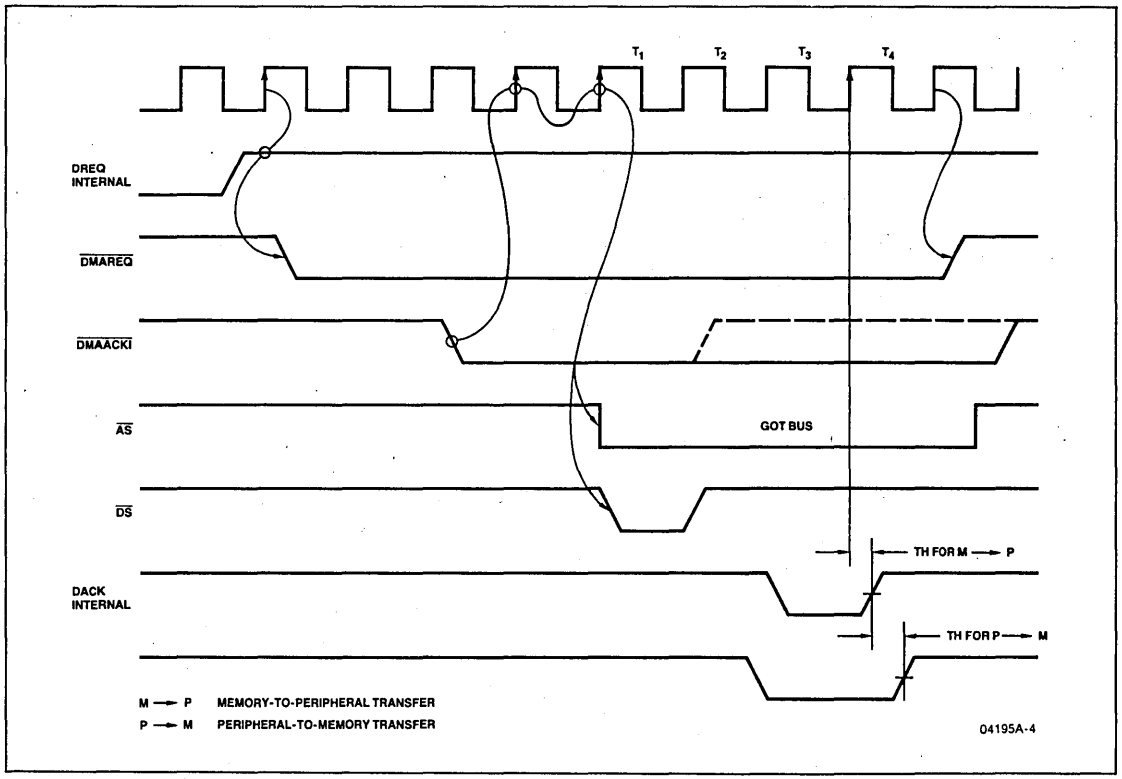


Figure 3b. Bus Exchange Protocol

B/$\overline{W}$: Byte/Word Output Optional

This output, present on Bus Masters only, indicates the type of data transferred on the AD_0-AD_{15} bus. HIGH indicates byte (8-bit) and LOW indicates word (16-bit) transfer. This output is activated at the same time as $\overline{AS}$ goes LOW and remains valid for the duration of the whole transaction (see timing diagram). The address generated by the peripheral is always a byte address, however the memory is organized as 16-bit words. All instructions and word operands are word aligned and are addressed by even addresses. Thus, for all word transactions with the memory the least significant address bit will be zero. When addressing the memory for byte transactions, the least significant address bit determines which byte of the memory word is needed; an even address specifies the most significant byte and an odd address specifies the least significant byte. When possible, the bus interface should contain Byte Swap Logic to make the even address be the least significant byte and odd address be the most significant byte to be more compatible with Intel parts.

Note: The Bus exchange protocol described here although slanted towards the Z8000 can be used with HOLD, HLDA protocol simply by adding two inverters. Thus the added advantage of being able to daisy chain parts with no external logic!

$\overline{DMAREQ}$: Bus Request (Input/Output, Open Drain)

On Bus Masters only, a LOW on this output indicates to the CPU that the peripheral is requesting to take control of the bus. The

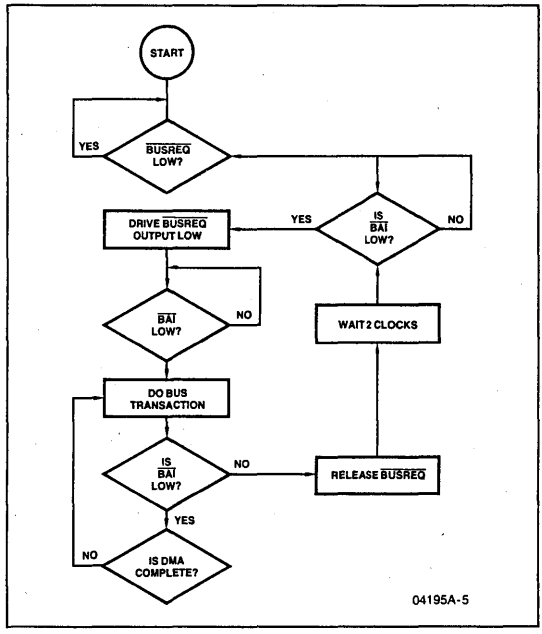


Figure 3c. Preemptive Bus Request Protocol

peripheral samples $\overline{\text{DMAREQ}}$ to see if it is LOW before driving it active. The $\overline{\text{DMAREQ}}$ output can be driven LOW anytime with respect to the CPU since the CPU synchronizes this input internally. Figure 3a shows how Bus Masters are connected. Figure 3b shows conventional bus exchange timing.

The CPU responds by activating the $\overline{\text{DMAACK}}$ output LOW to indicate that the bus has been relinquished. The peripheral should control the bus in an identical fashion to the CPU to accomplish transactions. The $\overline{\text{DMAREQ}}$ input must remain LOW as long as needed to perform all the transactions, and the CPU will keep the $\overline{\text{DMAACK}}$ output LOW. After completing the transactions, the peripheral device must disable driving the bus and stop driving the $\overline{\text{DMAREQ}}$ input LOW. The CPU will make $\overline{\text{DMAACK}}$ output HIGH sometime later and resume bus control. The DMA device must not reassert $\overline{\text{DMAREQ}}$ until $\overline{\text{DMAACK}}$ has gone inactive and 2 clock cycles have elapsed.

$\overline{\text{DMAACKI}}$: Bus Acknowledge (Input)

LOW on this input indicates that the CPU has relinquished control of the bus in response to a bus request. The peripheral synchronizes the acknowledge then enables its drivers to assume bus control. If $\overline{\text{DMAACKI}}$ is removed while the peripheral has control of the bus, it will release the bus after it completes the current transaction. $\overline{\text{DMAREQ}}$ will be driven HIGH, indicating to the higher priority device the bus was released. This allows preemptive DMA. The preempted device waits 2 clock periods

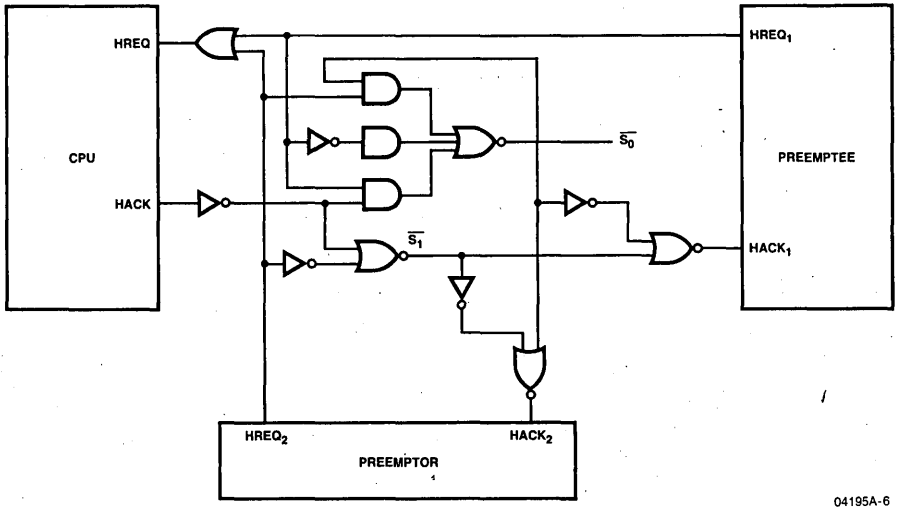


Figure 3d. Implementation of Preemptive DMA #1

and then reasserts $\overline{\text{DMAREQ}}$, and on receipt of $\overline{\text{DMAACK}}$ resumes from where it left off. Figure 3c, show preemptive operation. The preemption protocol defined allows two types of preemption to be implemented. In Implementation #1 the preemptor and preemptee can exchange the bus with the CPU unaware of the exchange. Alternately in Implementation #2 the CPU is allowed to do one bus cycle. Preemption has also been called Control Override Sequence in previous documents. See Figures 3d through 3g for timing and implementation examples.

$\overline{\text{WAIT}}$ or READY: Wait (Input, Output, Open Drain)

When peripheral is a Bus Master a LOW on this input indicates to the peripheral that memory or I/O is not ready for the data transfer and hence the current transaction should be stretched. The $\overline{\text{WAIT}}$ input is sampled by the peripheral at certain instances during the transaction (see timing diagram). If $\overline{\text{WAIT}}$ input is LOW at these instances, the peripheral will go into a wait state to prolong the transaction. The wait state will repeat until the $\overline{\text{WAIT}}$ input is HIGH at the sampling instant. This signal should be sampled as late in the cycle as possible.

When operating as a slave this line indicates to the processor that the peripheral has accepted data during a write or that data is present on the AD Bus during a read. The time this signal is asserted is determined by the slave. The mode register should never require a wait state during a write and the status register should never require a wait state during a read.

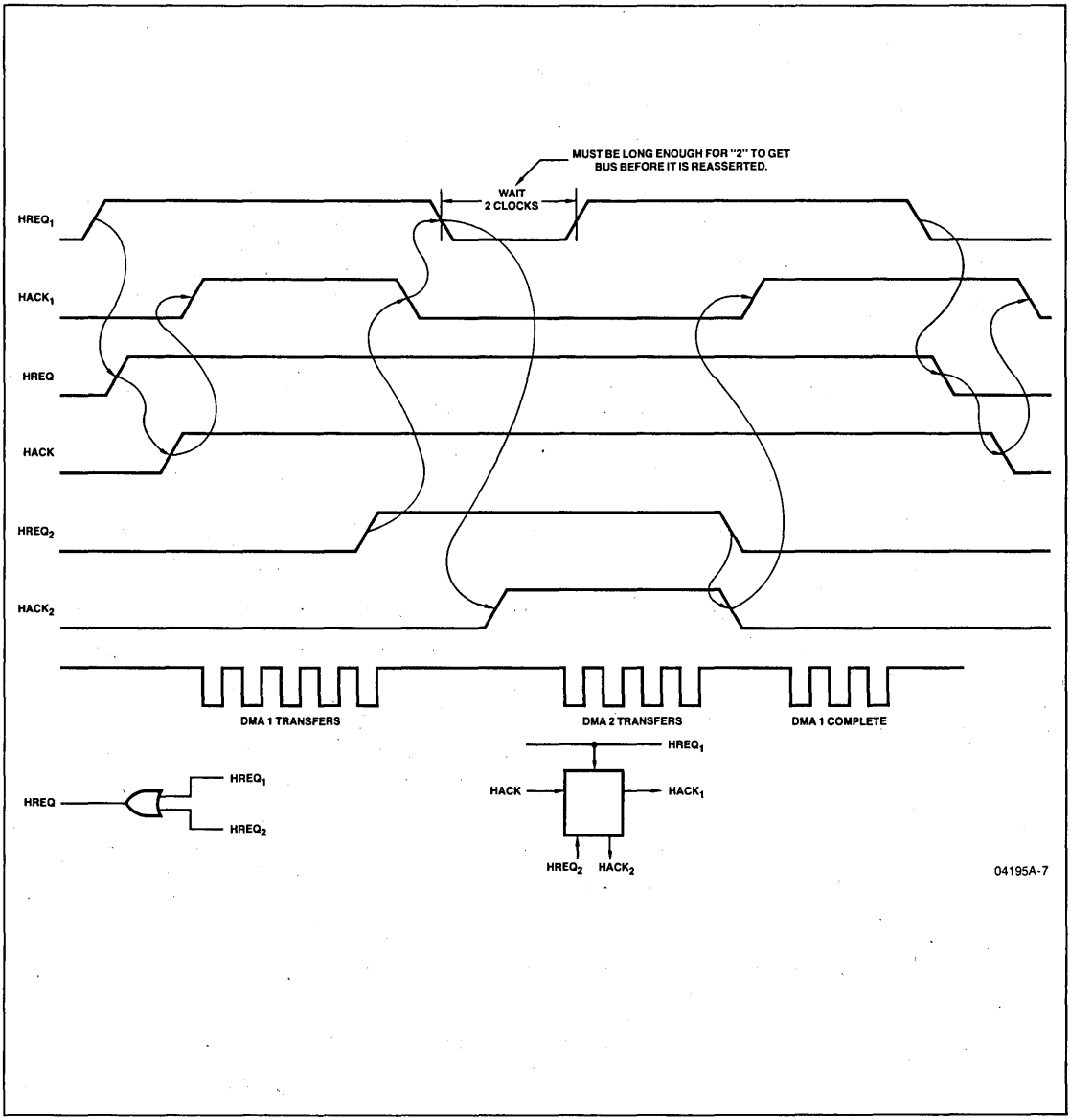


Figure 3e. Preemptive DMA Notes

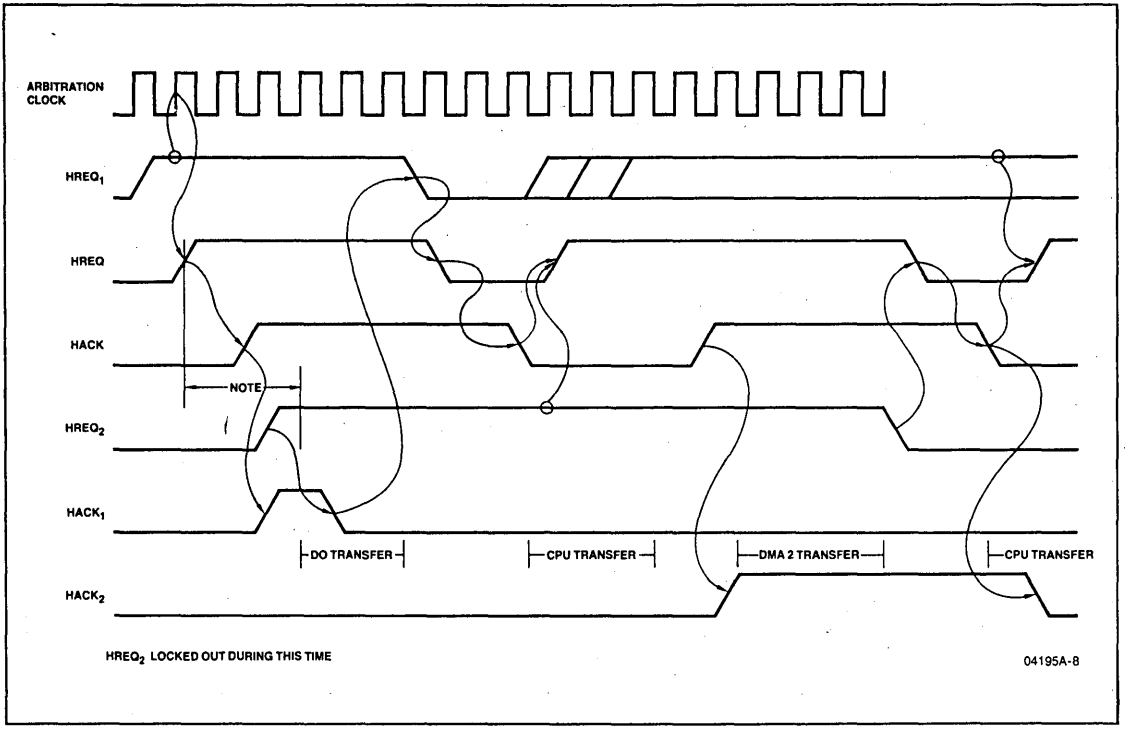
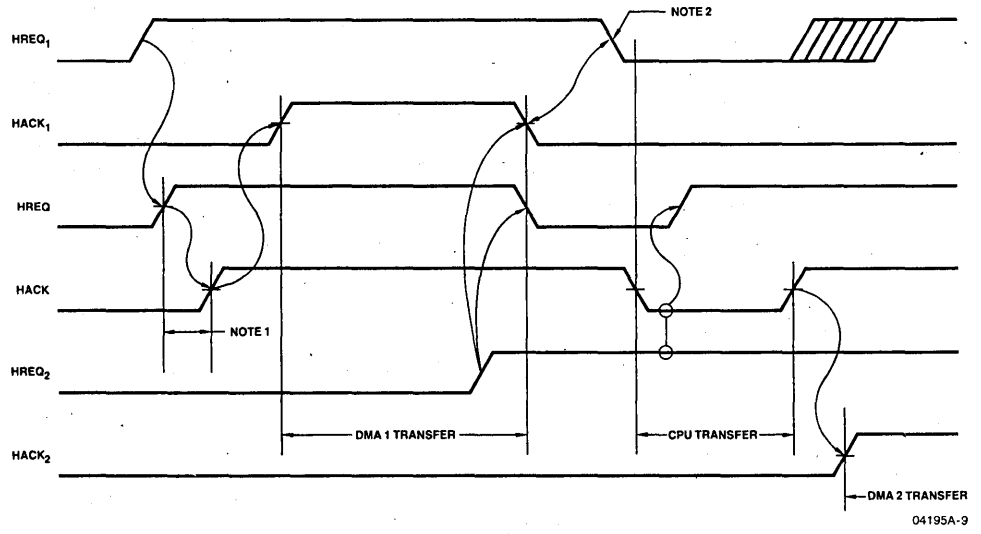


Figure 3f. Alternate Preemption Scheme



Notes: 1. Arbiter must be designed such that if HREQ is active and HACK is inactive any other request that occurs is ignored.
2. $HREQ_1$ must go LOW before HACK goes LOW or bus contention could result. One way to ensure this is by gating $HREQ_2$ with $\overline{DS}$. This ensures the minumum amount of time between $\overline{HACK}_1$ going inactive and $\overline{HREQ}_1$ going inactive is met.

Figure 3g. Implementation of Preemptive DMA #2

$\overline{XACK}$ Transfer Acknowledge

This is preferred over READY/$\overline{WAIT}$ as it makes system design easier due to timing requirements. See Figure 4a for timing examples.

$ADR_N - ADR_K$: Extended Address (Output, Three-State)

The extended address lines may be used to extend the address space of the peripheral. This applies to Bus Masters only. If pin limitations do not allow for this a second address strobe may be added thus the upper address may be latched and then the lower address latched by the second strobe. This imposes a slight time penalty but only when a 64K boundary is crossed. See timing diagram Figure 5 for this option.

$\overline{INT}$: INTERRUPT (Output, Open Drain)

This active Low output is used to interrupt the CPU. It may be connected to any of the CPU interrupt inputs, and may be wire-ORed with other sources of interrupts. An external pull-up resistor is required.

OPTIONAL PINS

$\overline{ADRACK}$: Address Acknowledge (Input)

This is an option on Bus Masters, used to extend the time the address is held on the bus. This is a valuable function in systems with a multimaster multiplex back plane bus. It is also valuable in hierarchical bus structures where extra delays may be in the address path (see timing diagram Figure 4b for effect of this pin). An additional advantage of including this function is that the wider address latch and additional address setup time is compatible with DEC's II Bus.

M/$\overline{IO}$: Memory/Input-Output (Output, Three-State)

This pin is only used by Bus Masters to indicate when it is doing a memory or I/O transaction and is under the optional heading because not all Bus Masters need this capability. For example, in a system with only memory mapped I/O it is not required. For parts such as the CRT or LAN the source on destination is always memory, hence it is not required. Indeed the only time it is required is when it makes sense to use a DMA to control data transfers one peripheral to another. A possible example would be dumping a disk file to a printer.

$\overline{LW}$: Long Word (Input, Output)

This pin is required on any 32-bit peripheral. When HIGH it indicates a 16-bit operation; when LOW it indicates a 32-bit operation.

$\overline{INTA}$: Interrupt Acknowledge (Input)

This pin is only required if it is desired to support vectored interrupts without requiring an interrupt controller. It is used to gate the vector onto the bus. Otherwise the interrupt can be cleared by software.

LEN: Latch Enable (Output)

This pin is only used by Bus Masters that multiplex both the HIGH and LOW addresses on the bus. This is done whenever a 64K boundary is crossed, hence the extra overhead required is not excessive. This can be used to save package pin count. See Figure 5.

IEI: Interrupt Enable In (Input)

This signal is used to implement a daisy chain priority scheme. When HIGH the part responds to $\overline{INTA}$.

IEO: Interrupt Enable Out (Output)

This signal is HIGH to indicate the next device has priority and no higher priority device is generating an interrupt. For further details of interrupt structure see the ZBUS specification. Note that this interrupt architecture is compatible with Intel devices.

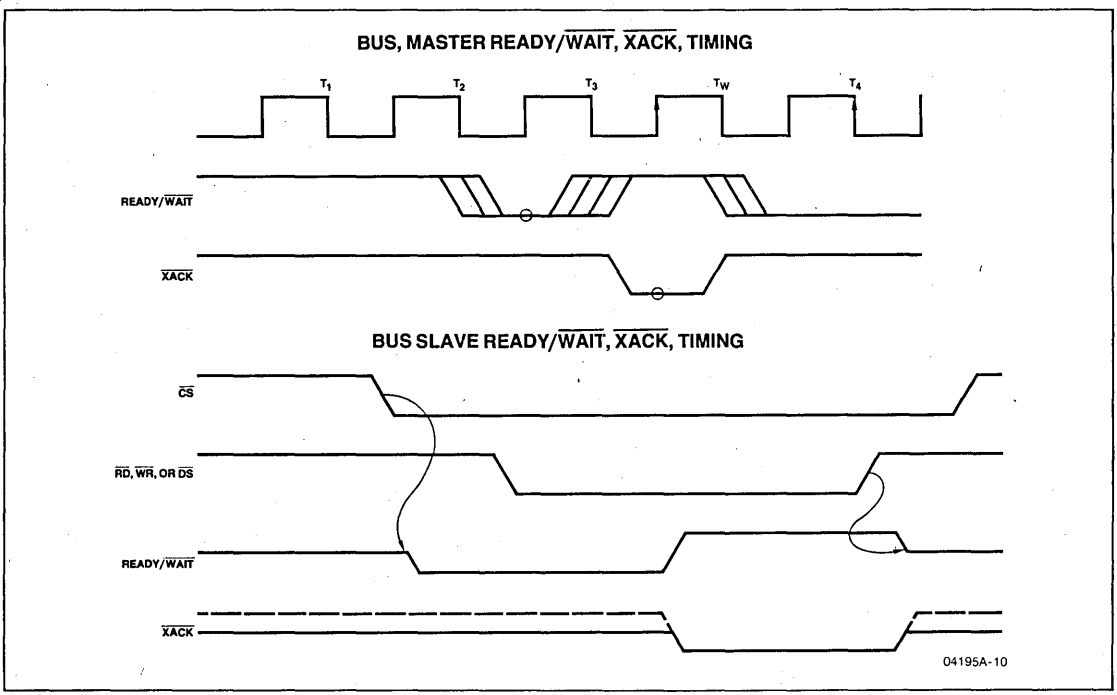


Figure 4a.

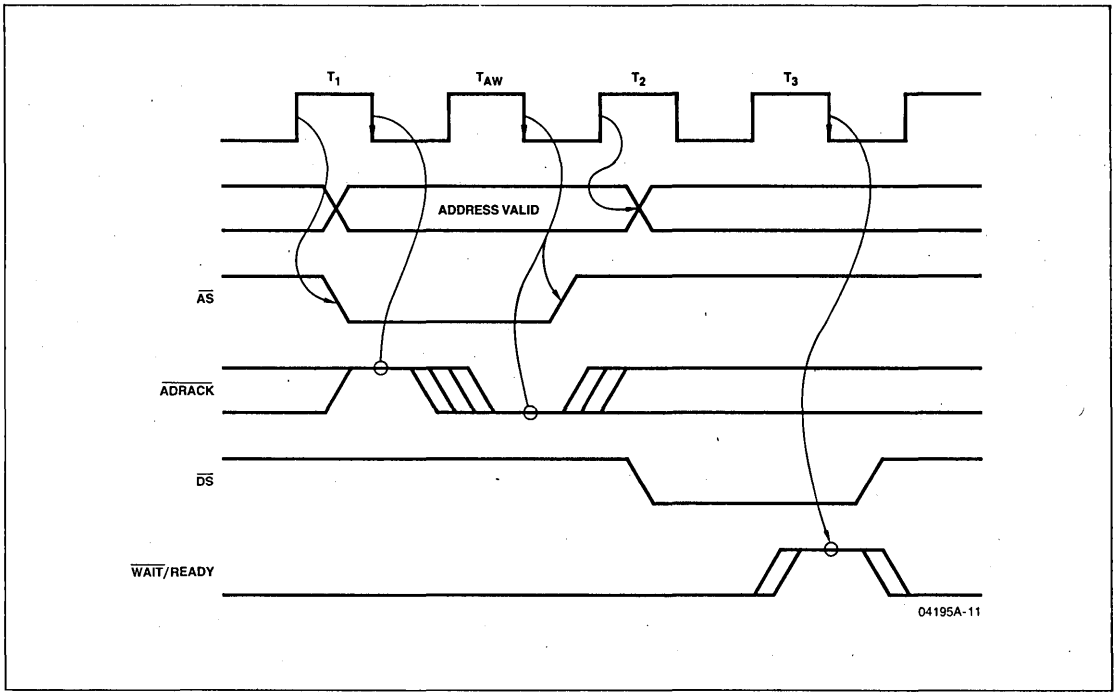


Figure 4b. Effect of $\overline{\text{ADRACK}}$

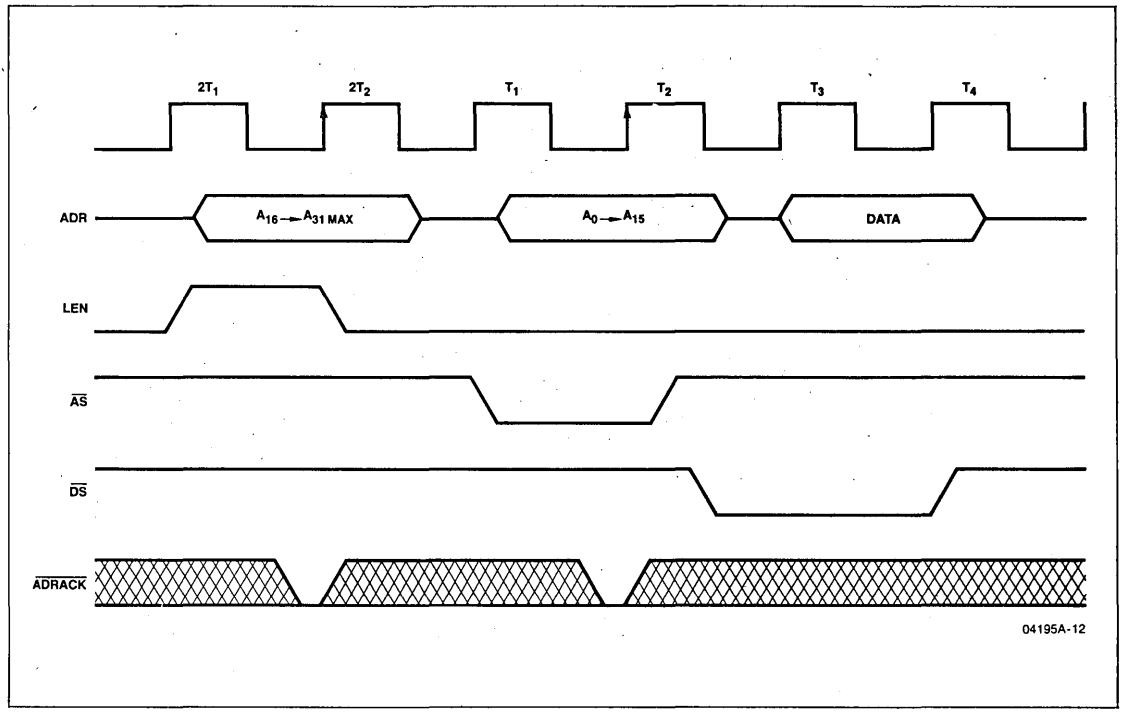


Figure 5. Lens Option

APPENDIX A

1) Slave Peripheral Interface
2) DMA Peripheral Interface
3) R/$\overline{W}$ and DS to $\overline{RD}$, $\overline{WR}$ and vice/versa
4) 286 Bus Exchange

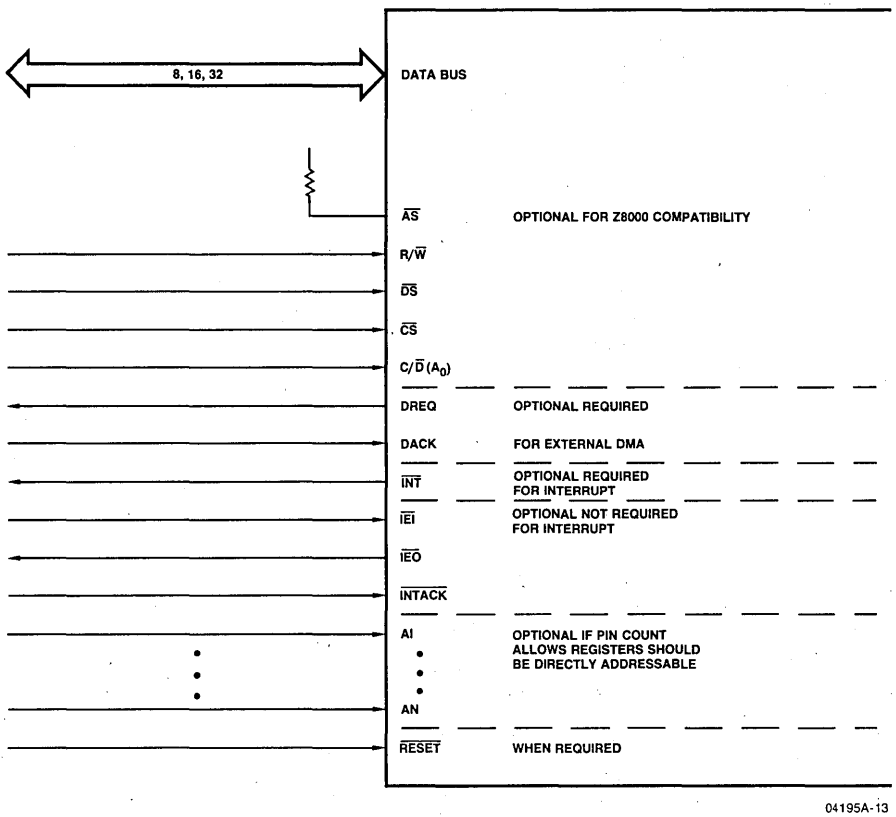


Figure A-1. Slave Peripheral Interface

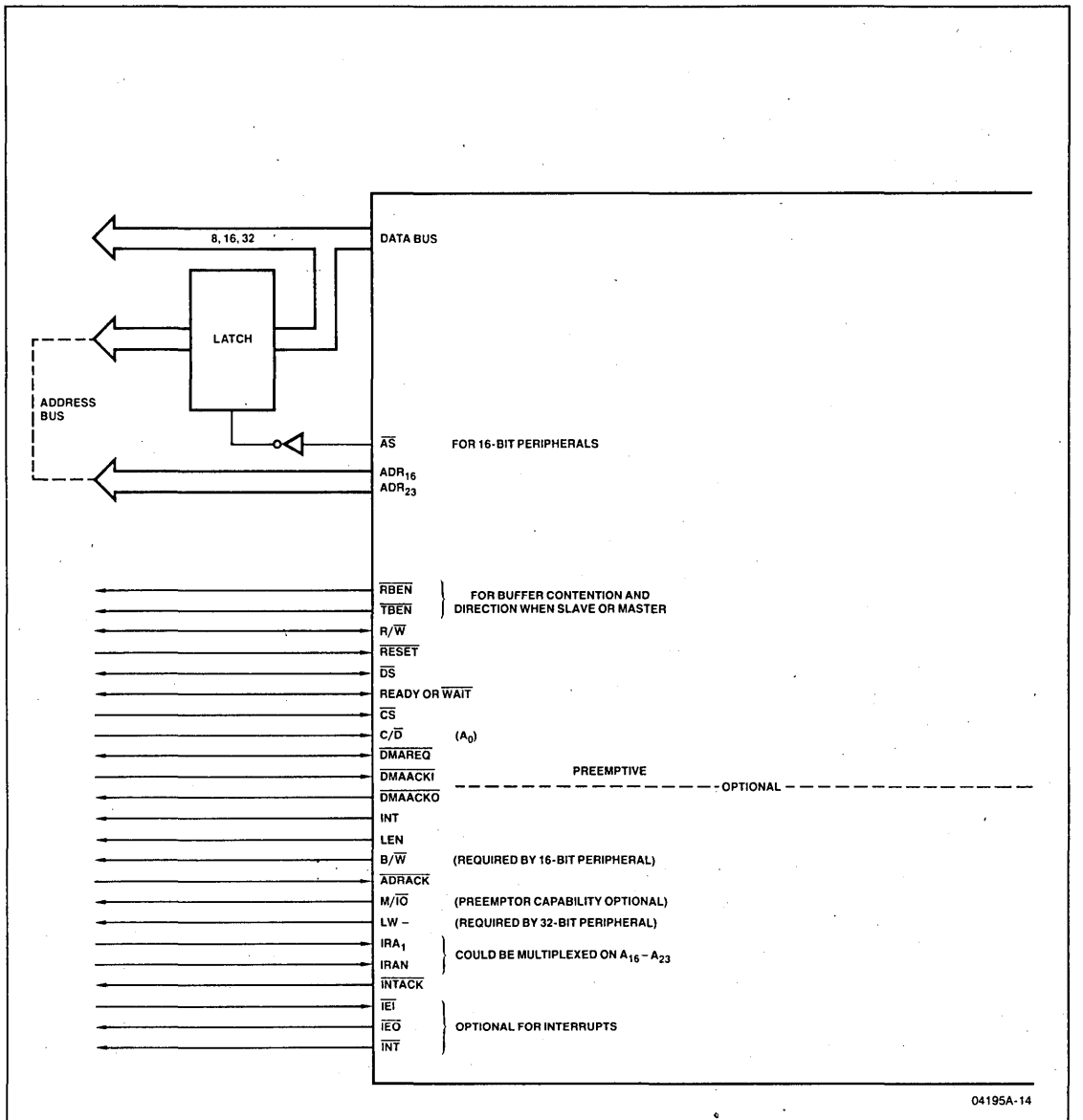
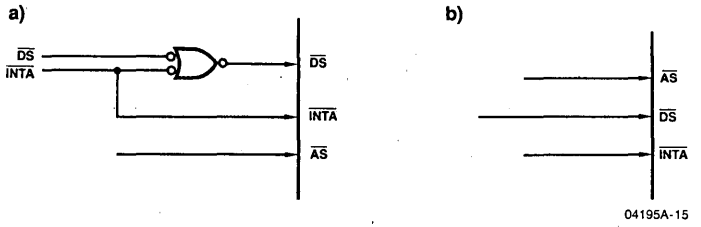


Figure A-2. Peripheral with DMA Interface



a) In non-Z8000 applications, parallel prioritization or an interrupt controller is required. In the case of parallel prioritization, the falling edge of $\overline{INTA}$ freezes the prioritization process. When using an interrupt controller, the interrupt pending bit is cleared by a software command. The OR gate shown is used for processors that do not generate $\overline{DS}$ or $\overline{RD}$ during interrupt acknowledge gates.

b) To use serial prioritization, $\overline{AS}$ is required to indicate priority chain must be frozen. The $\overline{INTA}$ status combined with $\overline{DS}$ is then used to place the vector on the bus.

Figure A-3.

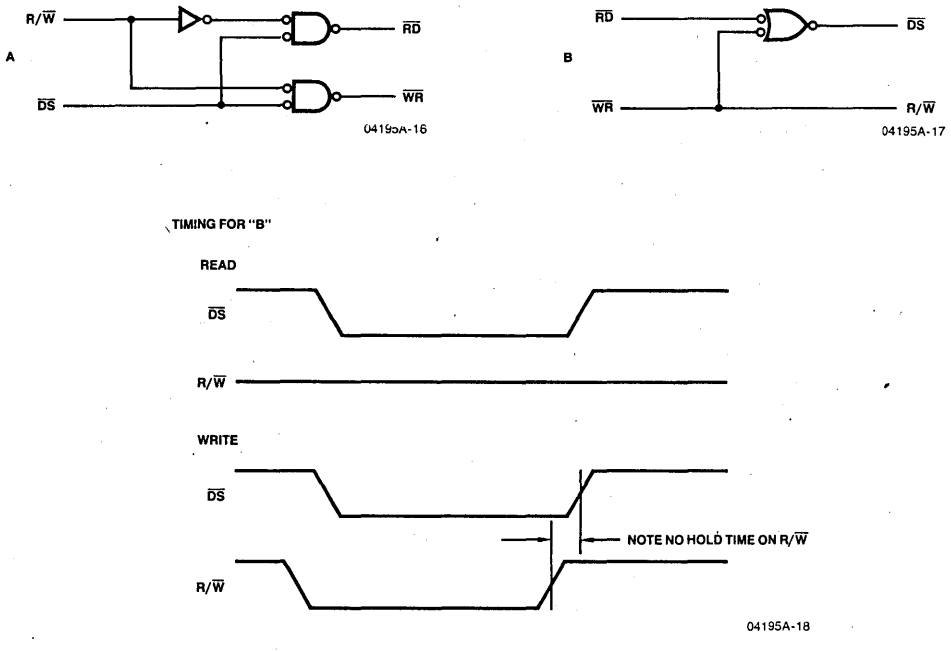


Figure A-4.

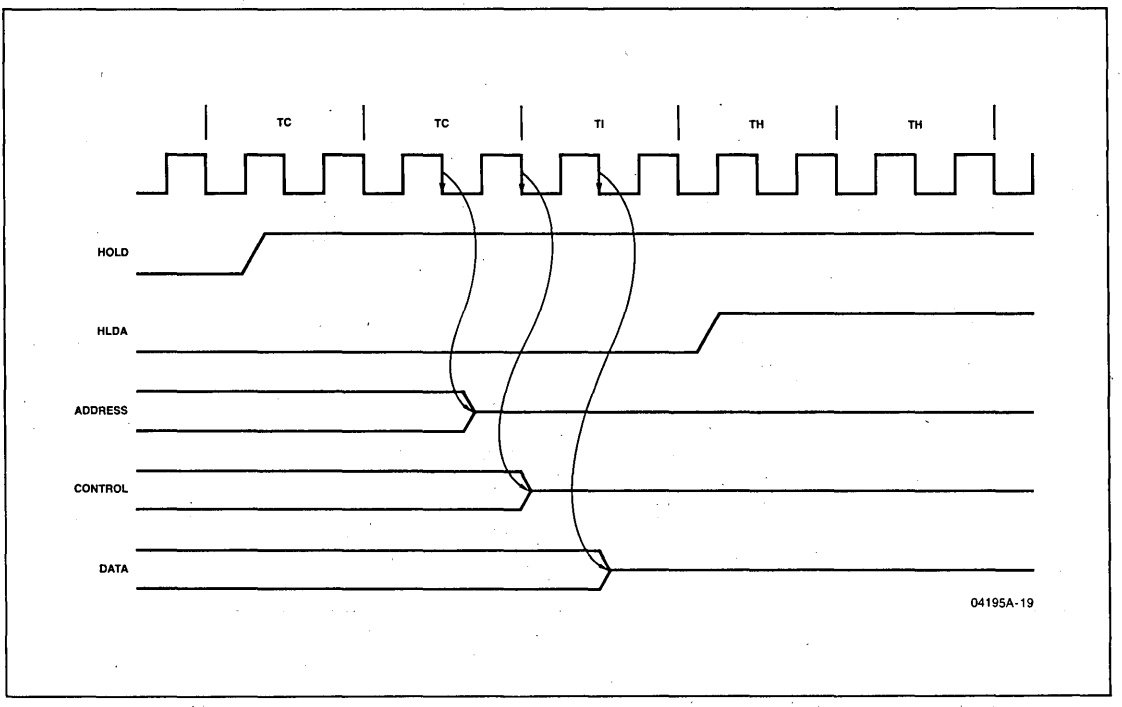


Figure A-5. iAPX286 Bus Exchange

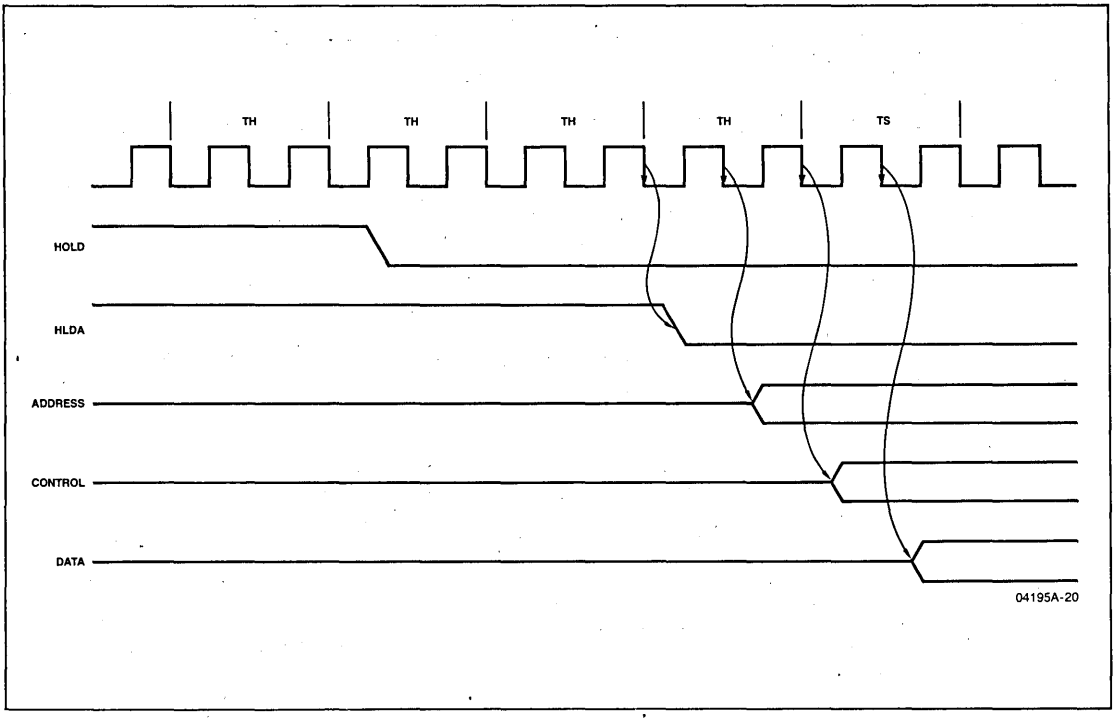
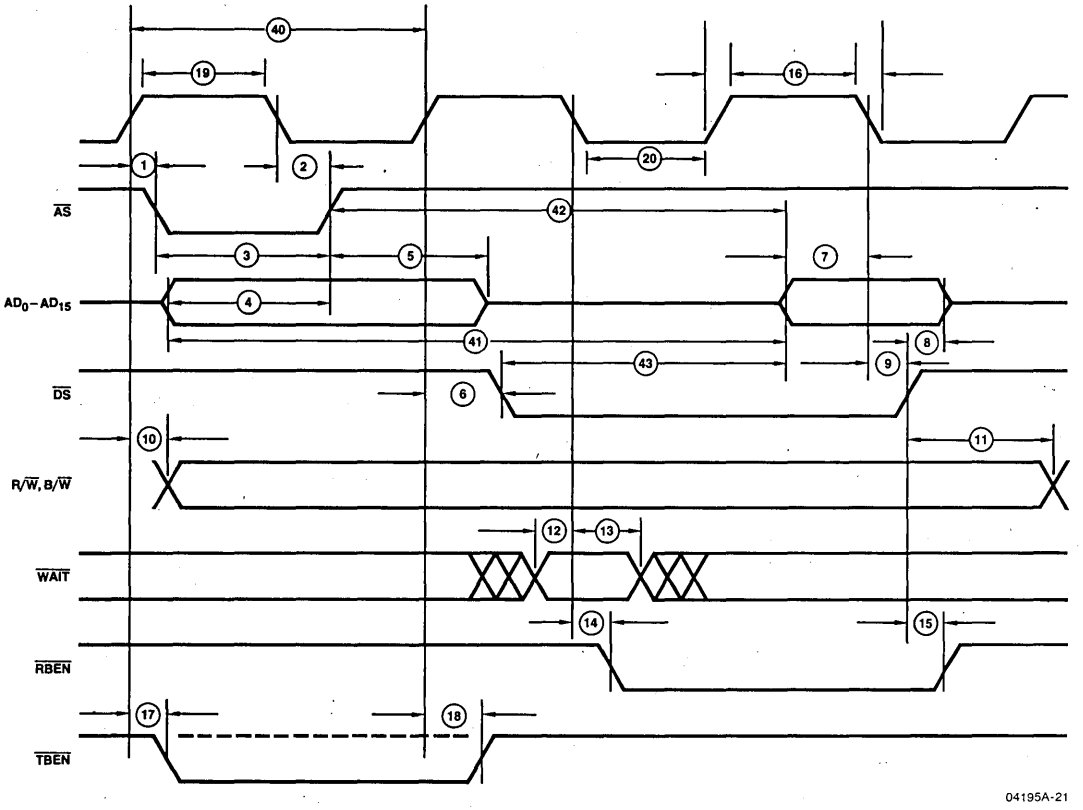


Figure A-6. iAPX286 Bus Exchange

APPENDIX B

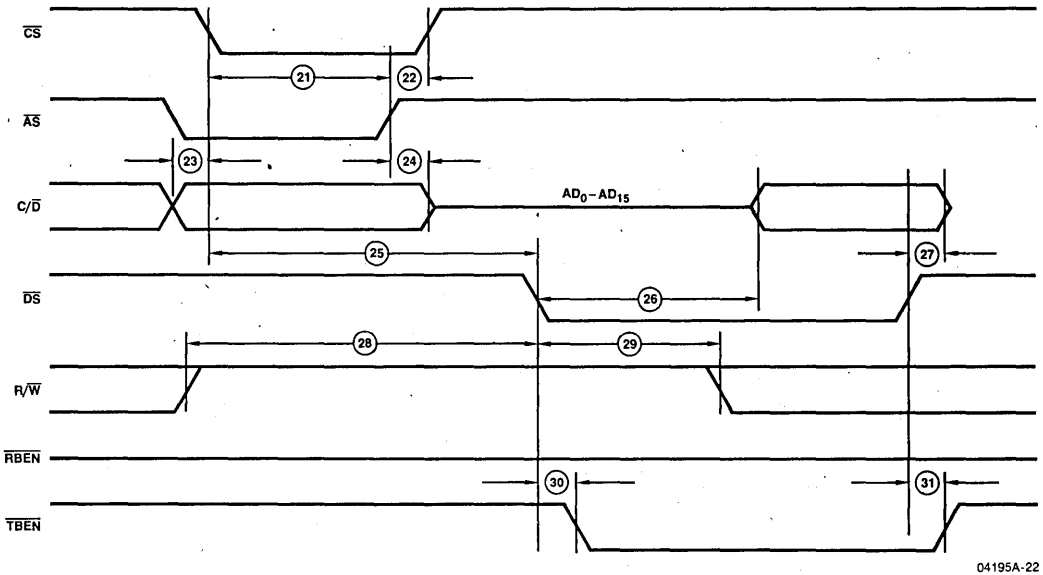
3 Cycle Bus Master and Slave Timing



			Min	Max
1	t_{PHL}	CLK↑ to $\overline{AS}$↓		30
2	t_{PLH}	CLK↓ to $\overline{AS}$↑		30
3	t_{pw}	$\overline{AS}$ PULSE WIDTH	45	
4	t_s	Address to $\overline{AS}$↑	40	
5	t_h	Address to $\overline{AS}$↑	40	
6	t_{PHL}	CLK↑ to $\overline{DS}$↓		30
7	t_s	DATA to CLK↓	10	
8	t_h	DATA to $\overline{DS}$↑	0	
9	t_{PLH}	CLK↓ to $\overline{DS}$↑		30
10	t_{pd}	CLK↑ to R/$\overline{W}$, B/$\overline{W}$ VALID		30
11	t_{pd}	CLK↑ to R/$\overline{W}$, B/$\overline{W}$ NOT VALID	30	
12	t_s	$\overline{WAIT}$ to CLK↓	10	
13	t_h	$\overline{WAIT}$ to CLK↓	20	

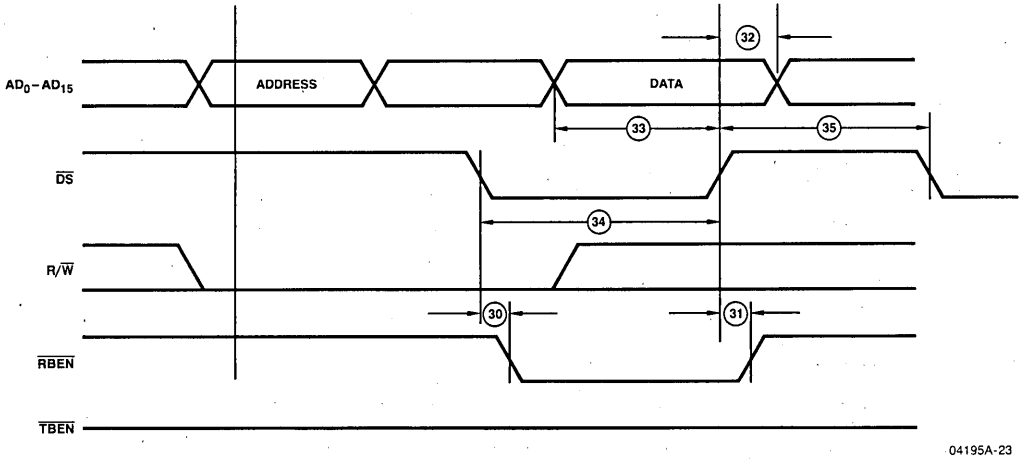
			Min	Max
14	t_{PHL}	CLK↓ to $\overline{RBEN}$↓		30
15	t_{PLH}	DS↑ to $\overline{RBEN}$↑	0	
16	t_r, t_f	Rise and Fall Time		10
17	t_{PHL}	CLK↑ to $\overline{TBEN}$↓		20
18	t_{PLH}	CLK↑ to $\overline{TBEN}$↑		30
19	t_{pw}	CLK HIGH PULSE WIDTH	45	
20	t_{pw}	CLK LOW PULSE WIDTH	45	
40	t_{cyc}	Clock Period	125	
41	t_{AVDV}	Address VALID to DATA in Required VALID		260
42	t_{ASDV}	AS↑ to DATA VALID		205
43	t_{DSDV}	DS↓ to DATA VALID		150

Figure B-1. 8052 Bus Master Read



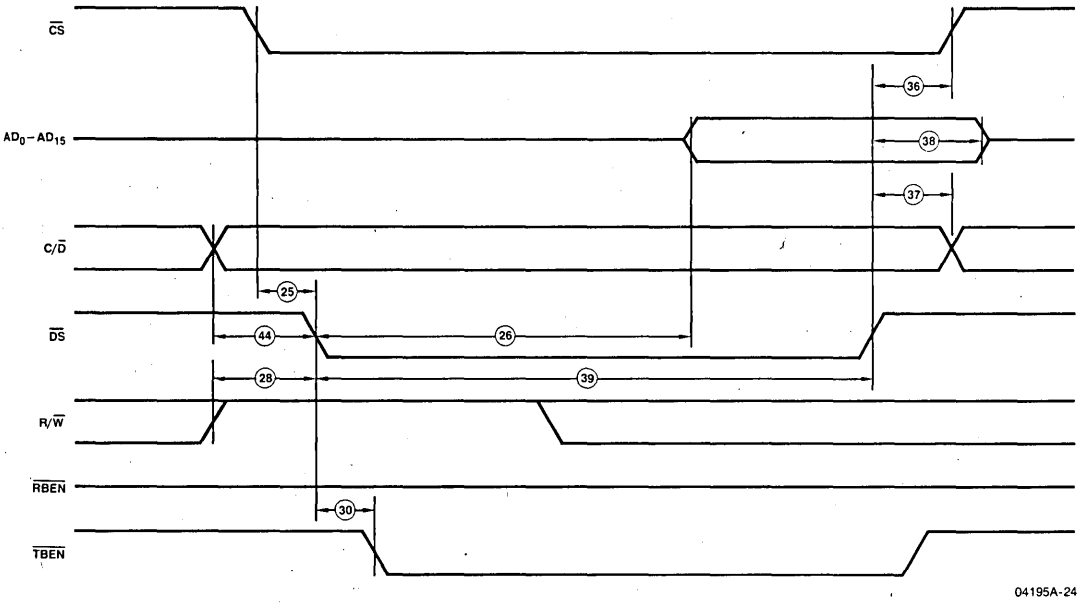
			Min	Max
21	t_s	$\overline{CS}\downarrow$ to $\overline{AS}\uparrow$	0	
22	t_h	$\overline{CS}\uparrow$ to $\overline{AS}\uparrow$	20	
23	t_s	C/$\overline{D}$ to $\overline{AS}\uparrow$	0	
24	t_h	C/$\overline{D}$ to $\overline{AS}\uparrow$	20	
25	t_{pd}	$\overline{CS}\downarrow$ to $\overline{DS}\downarrow$	30	
26	t_A	$\overline{DS}\downarrow$ to DATA VALID		150
27	t_h	$\overline{DS}\uparrow$ to DATA NOT VALID	15	
28	t_s	R/$\overline{W}$ to $\overline{DS}\downarrow$	0	
29	t_h	R/$\overline{W}$ to $\overline{DS}\downarrow$	40	
30	t_{PHL}	$\overline{DS}\downarrow$ to $\overline{TBEN}\downarrow$		30
31	t_{PLH}	$\overline{DS}\uparrow$ to $\overline{TBEN}\uparrow$		30

Figure B-2. 8052 Bus Slave Read Latched



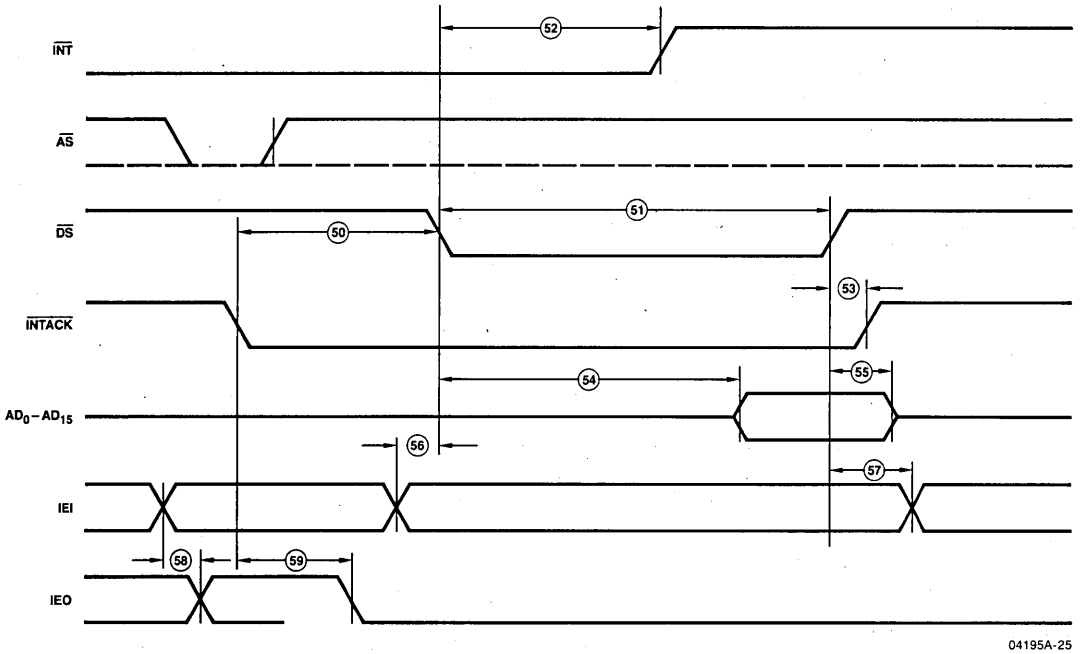
			Min	Max
32	t_h	$\overline{DS}\uparrow$ to DATA NOT VALID	0	
33	t_s	DATA VALID to $\overline{DS}\uparrow$		80
34	t_{pw}	$\overline{DS}\downarrow$ to $\overline{DS}\uparrow$ Write	100	
35	t_{RP}	Recovery Time Read or Write		2 t_{RP}

Figure B-3. 8052 Bus Slave Write



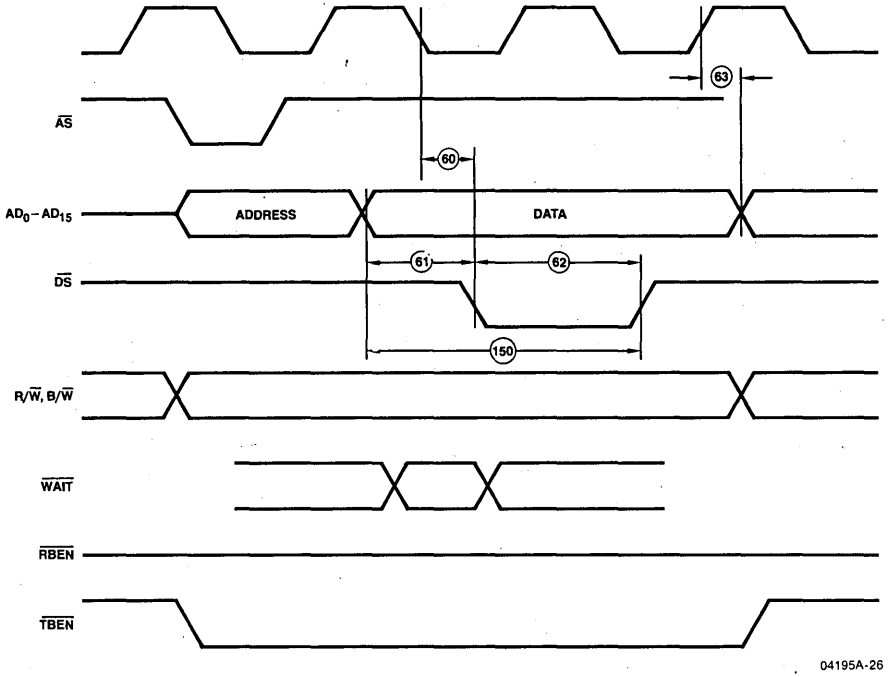
			Min	Max
36	t_h	$\overline{DS}\uparrow$ to $\overline{CS}\uparrow$	5	
37	t_h	$\overline{DS}\uparrow$ to C/$\overline{D}$	5	
38	t_z	$\overline{DS}\uparrow$ to DATA NOT VALID	10	50
39	t_{pw}	$\overline{DS}\downarrow$ to $\overline{DS}\uparrow$ Read	150	
44	t_s	C/D to $\overline{DS}\downarrow$	30	

Figure B-4. 8052 Bus Slave Read Unlatched



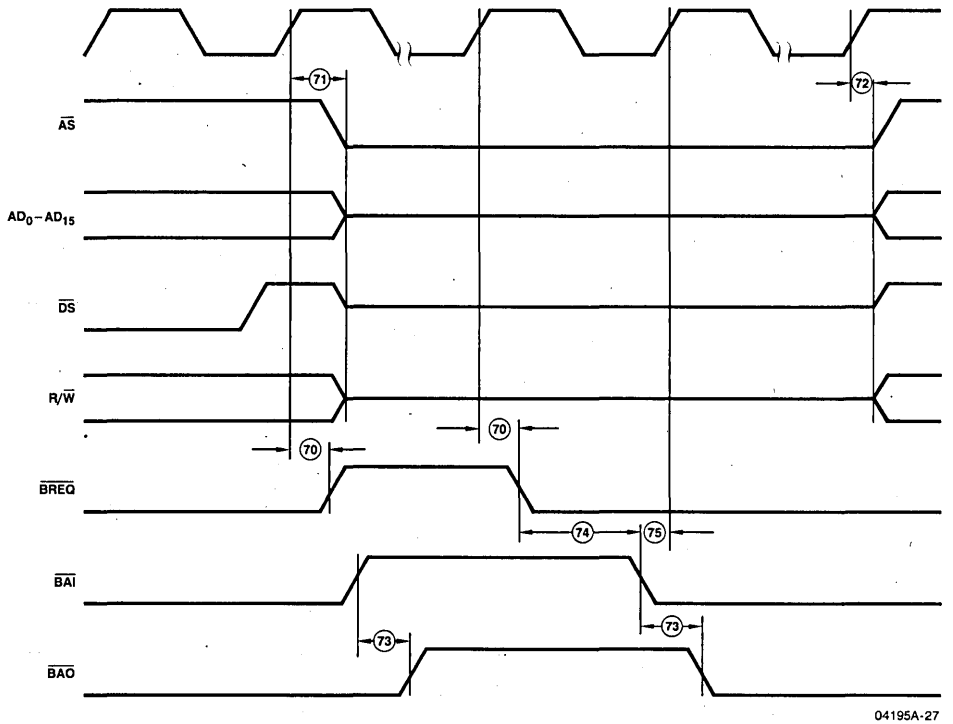
			Min	Max
50	t_d	$\overline{INTACK}$ to $\overline{DS}\downarrow$ Delay		200
51	t_{pw}	$\overline{DS}\downarrow$ to $\overline{DS}\uparrow$ Acknowledge	150	
52	t_{pd}	$\overline{DS}\downarrow$ to $\overline{DS}\uparrow$		200
53	t_h	$\overline{INTACK}$ to $\overline{DS}\uparrow$ Hold Time	0	
54	t_A	$\overline{DS}\downarrow$ to Vector VALID		150
55	t_{pd}	DS↑ to Vector NOT VALID	0	
56	t_s	IEI to $\overline{DS}\downarrow$	80	
57	t_h	IEI to $\overline{DS}\uparrow$	0	
58	t_{pd}	IEI to IEO		80
59	t_{pd}	$\overline{INTACK}$ to IEO		150

Figure B-5. Interrupt Acknowledge Timing



			Min	Max
60	t_{PHL}	CLK↓ to DS↓		30
61	t_s	DATA VALID to DS↓	40	
62	t_{pw}	DS↓ to DS↑	110	
63	t_{pd}	CLH↑ to DATA NOT VALID		30

Figure B-6. Bus Master Write

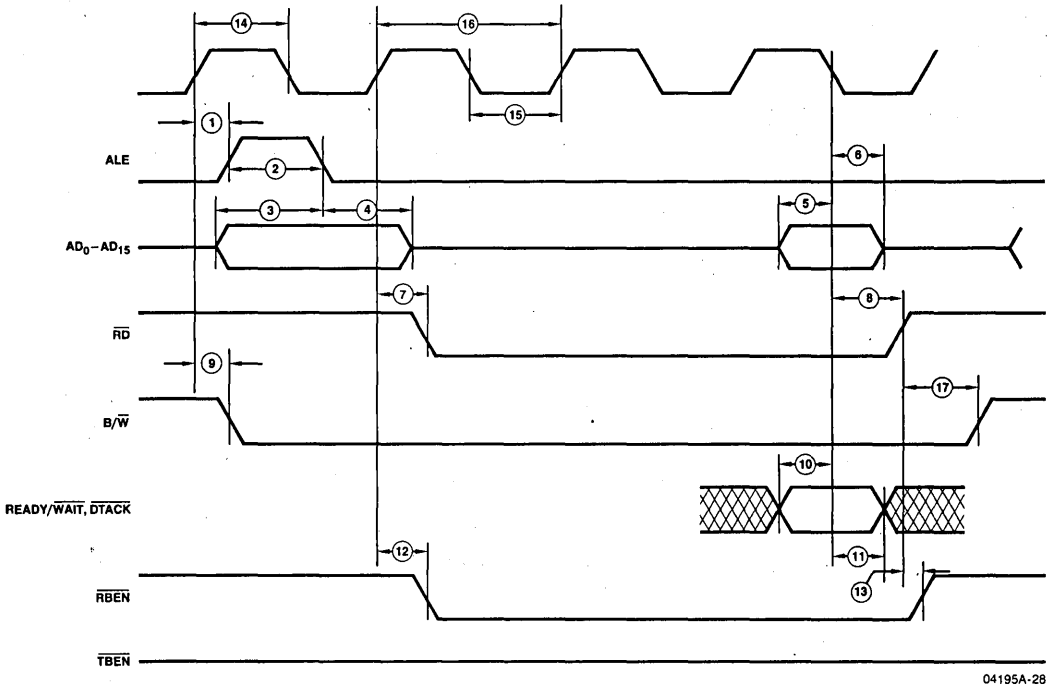


			Min	Max
70	t_{pd}	CLK↑ to $\overline{BREQ}$		100
71	t_{pz}	CLK↑ to Bus Float		100
72	t_{zd}	CLK↑ to Bus Driven		100
73	t_{pd}	BAI to BAO		40
74	t_{pd}	$\overline{BREQ}$ to $\overline{BAI}$ Delay	0	
75	t_s	BAI to CLK↑ Setup	20	

Figure B-7. 8052 Bus Exchange

APPENDIX C

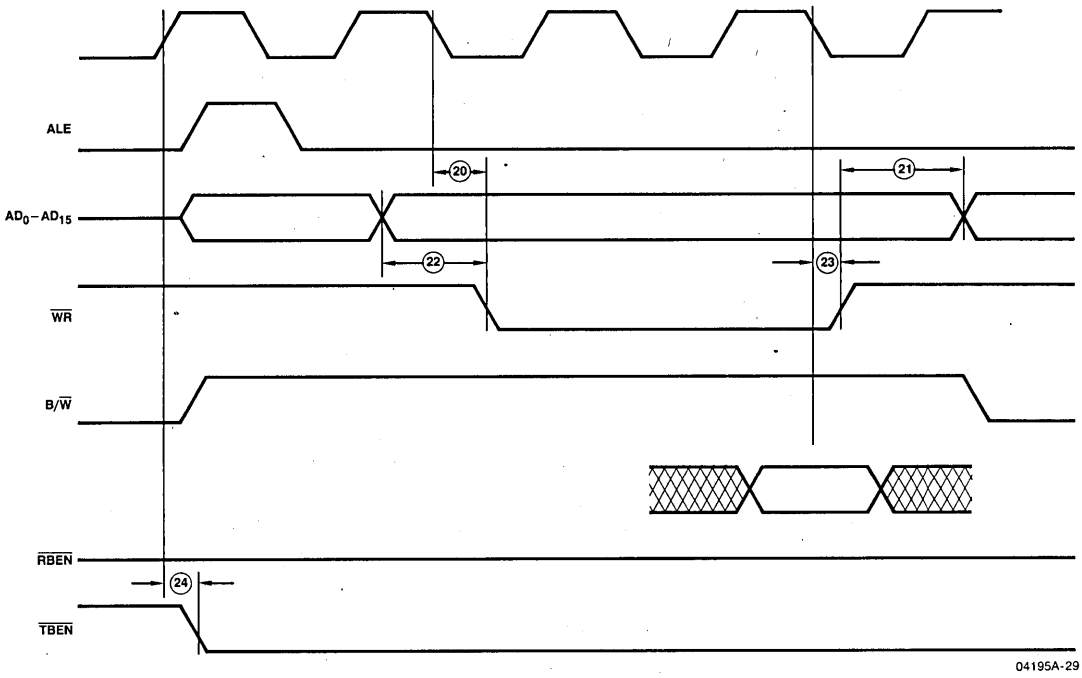
4 Cycle Bus Master and SlaveTiming



04195A-28

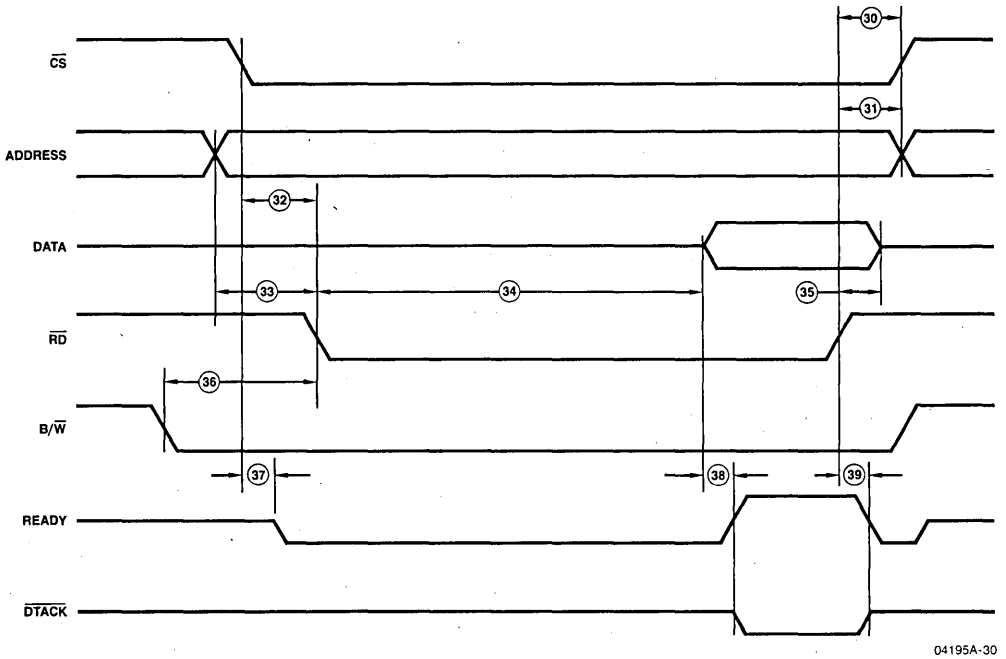
			Min	Max
1	t_{PLH}	CLK↑ to ALE↑		30
2	t_{pw}	ALE PULSE WIDTH	30	
3	t_s	Address to ALE↓	40	
4	t_h	ALE to Address VALID	40	
5	t_s	DATA to CLK↓	10	
6	t_h	CLK↓ to DATA	10	
7	t_{PHL}	CLK↑ to $\overline{RD}$↓		30
8	t_{PLH}	CLK↓ to $\overline{RD}$↑		30
9	t_{pd}	CLK↑ to B/W VALID		30
10	t_s	Ready to CLK↓	20	
11	t_h	CLK↓ to Ready	0	
12	t_{PHL}	CLK↑ to $\overline{RBEN}$↓		30
13	t_{PLH}	$\overline{RD}$↑ to $\overline{RBEN}$↑	0	
14	t_{pw}	Clock HIGH	30	
15	t_{pw}	Clock LOW	30	
16	t_{cyc}	Clock Period	100	
17	t_h	$\overline{RD}$ or $\overline{WR}$ to B/$\overline{W}$	40	

Figure C-1. Bus Master Read (4 Cycles)



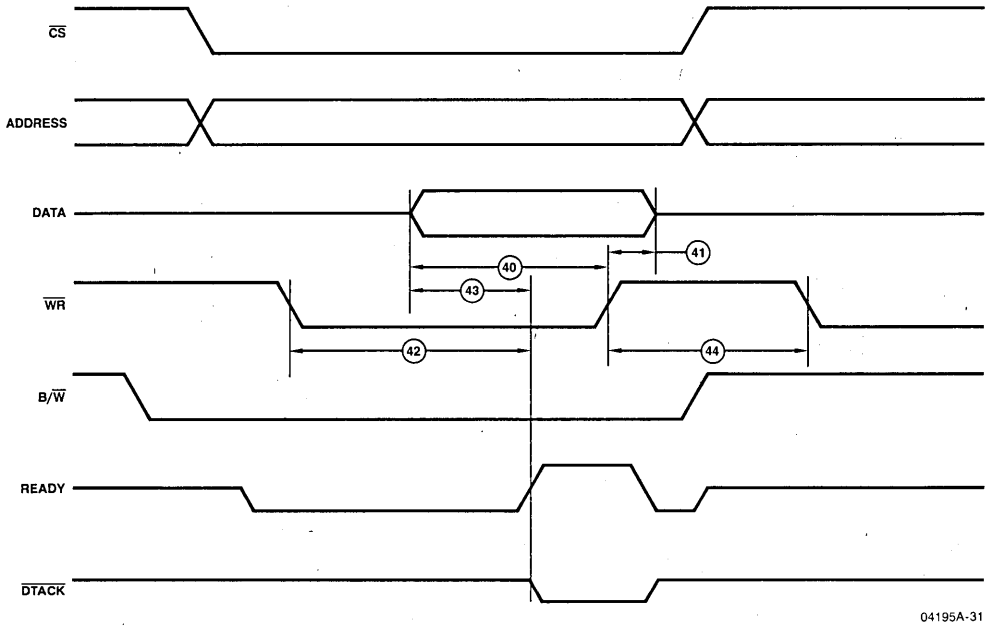
			Min	Max
20	t_{PHL}	CLK↓ to $\overline{WR}$↓		30
21	t_h	$\overline{WR}$ to DATA	40	
22	t_s	DATA to $\overline{WR}$↓	40	
23	t_{PLH}	CLK↓ to $\overline{WR}$↑		30
24	t_{PLH}	CLK↑ to $\overline{TBEN}$		30

Figure C-2. Bus Master Write (4 Cycles)



			Min	Max
30	t_h	$\overline{RD}\uparrow$ to $\overline{CS}$	10	
31	t_h	$\overline{RD}\uparrow$ to Address	10	
32	t_s	$\overline{CS}$ to $\overline{RD}\downarrow$		10
33	t_s	Address to $\overline{RD}\downarrow$ or $\overline{WR}$		20
34	t_{pd}	$\overline{RD}\downarrow$ to DATA VALID		150
35	t_{pz}	$\overline{RD}\uparrow$ to DATA FLOAT		30
36	t_s	B/$\overline{W}$ to $\overline{RD}\downarrow$	10	
37	t_{PZL}	$\overline{CS}\downarrow$ to Ready		30
38	t_{pd}	DATA VALID to Ready	0	
39	t_{pd}	$\overline{RD}\uparrow$ to Ready		30

Figure C-3. Slave Read



			Min	Max
40	t_s	DATA to $\overline{WR}\uparrow$		150
41	t_h	$\overline{WR}\uparrow$ to DATA	10	
42	t_{pd}	$\overline{WR}$ to Ready		100
43	t_s	DATA to Ready		50
44	t_r	Write Recovery Time	200	

Figure C-4. Slave Write

APPENDIX D

Multiple Preemptive Arbiter

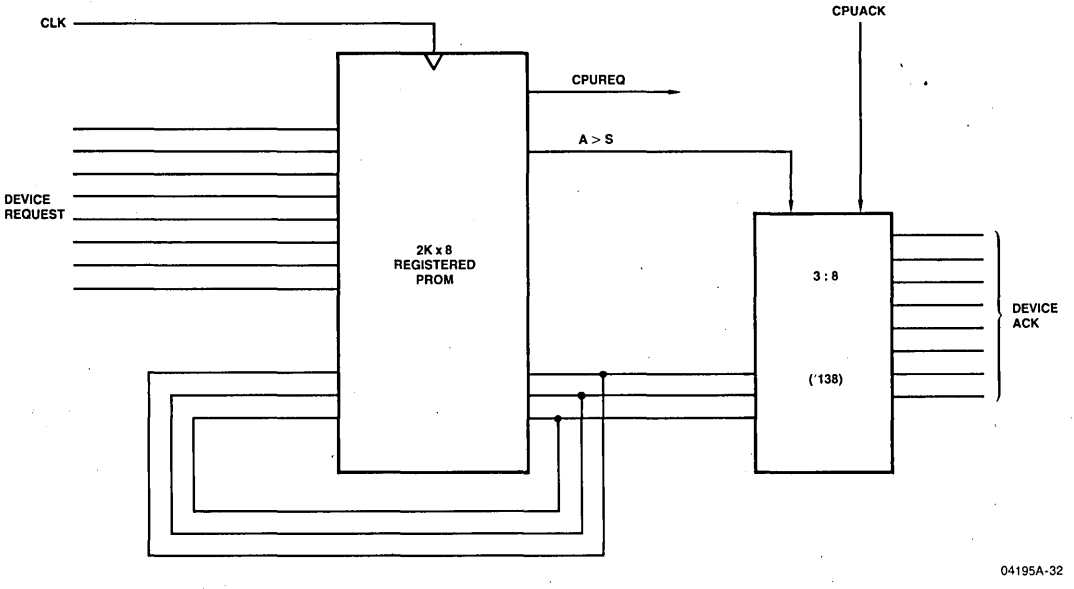
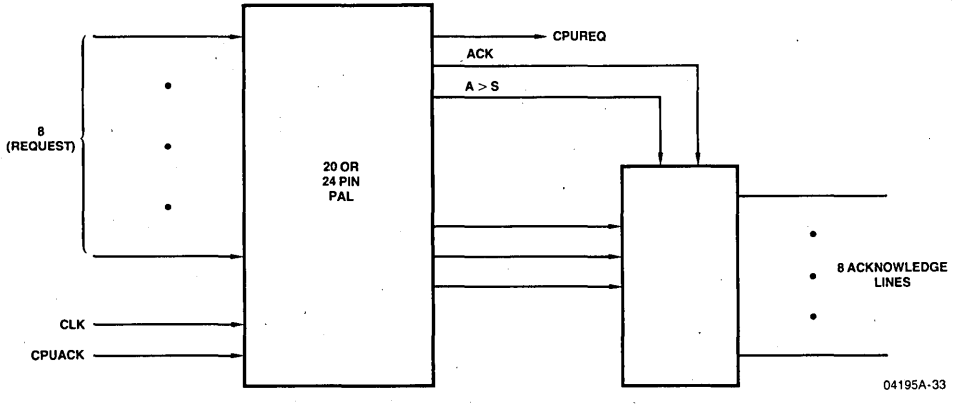


Figure D-1. Priority Based Preemptive Arbiter (PROM Version)



*PAL is a trademark of Monolithic Memories, Inc.

Figure D-2. Priority Based Preemptive Arbiter (PAL* Version)

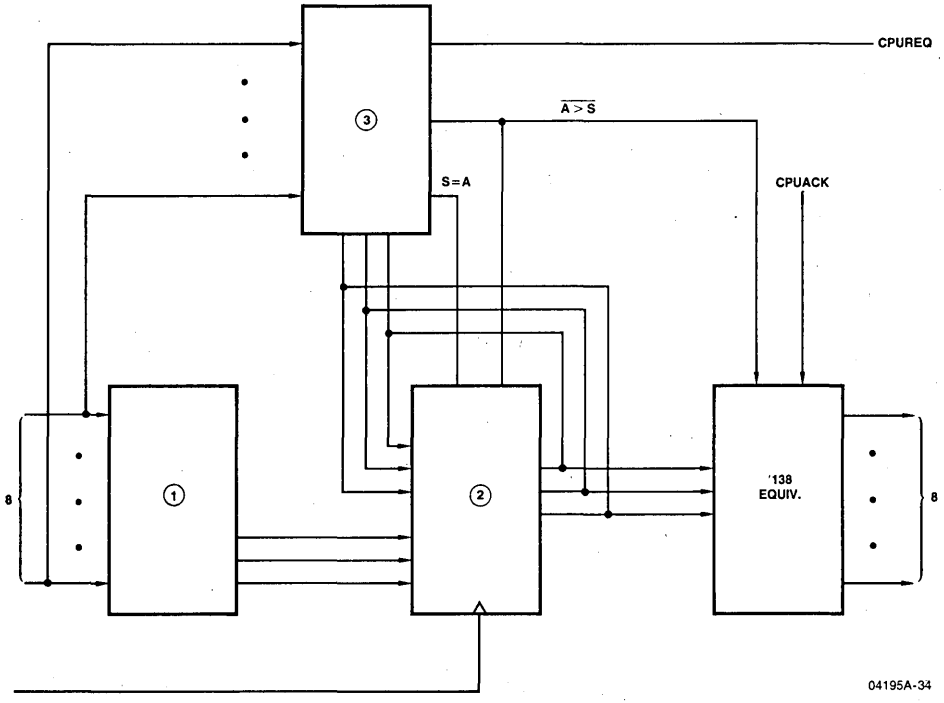


Figure D-3. Multiple Priority Based Preemptive Arbiter Logic

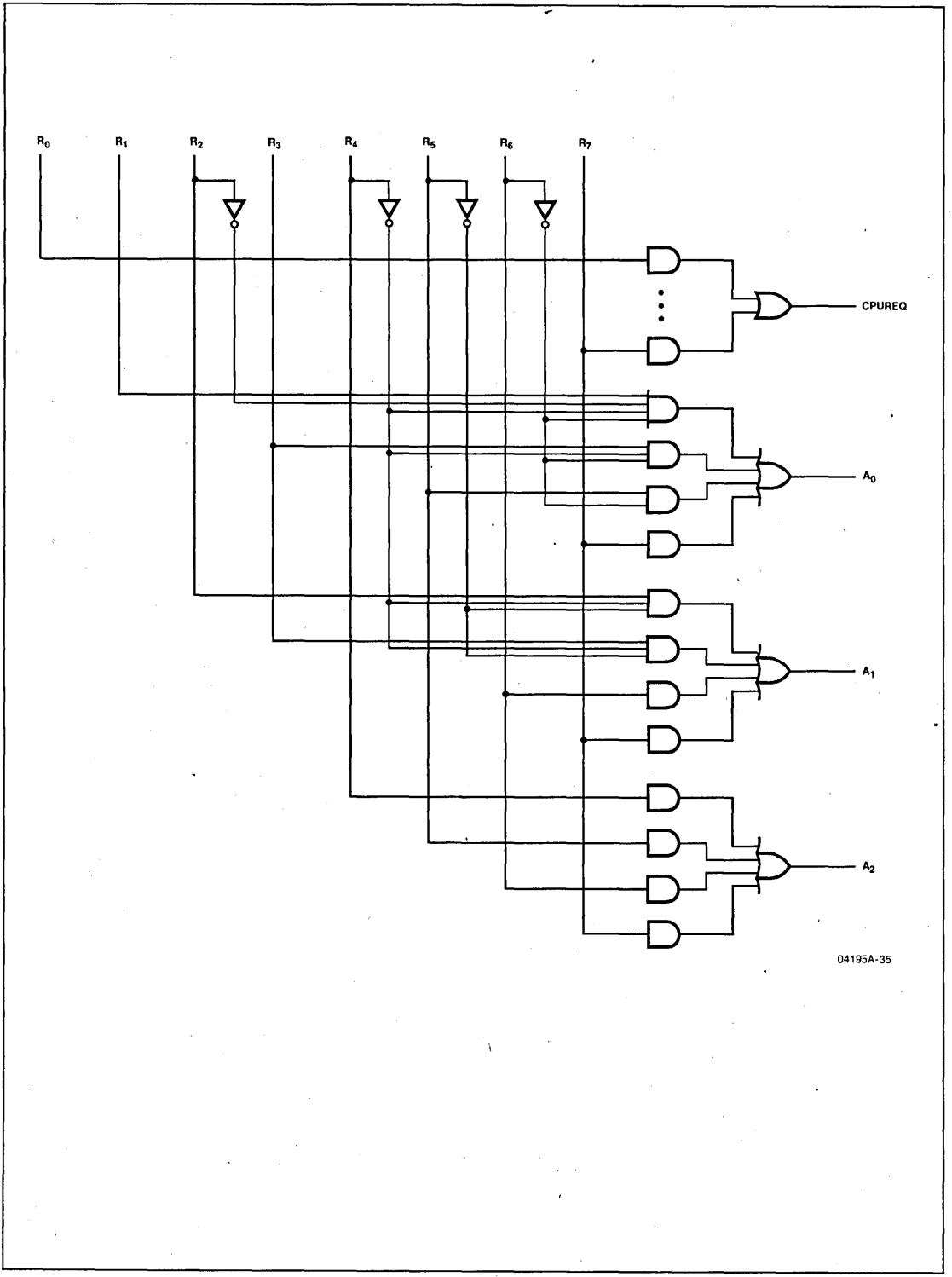


Figure D-4.

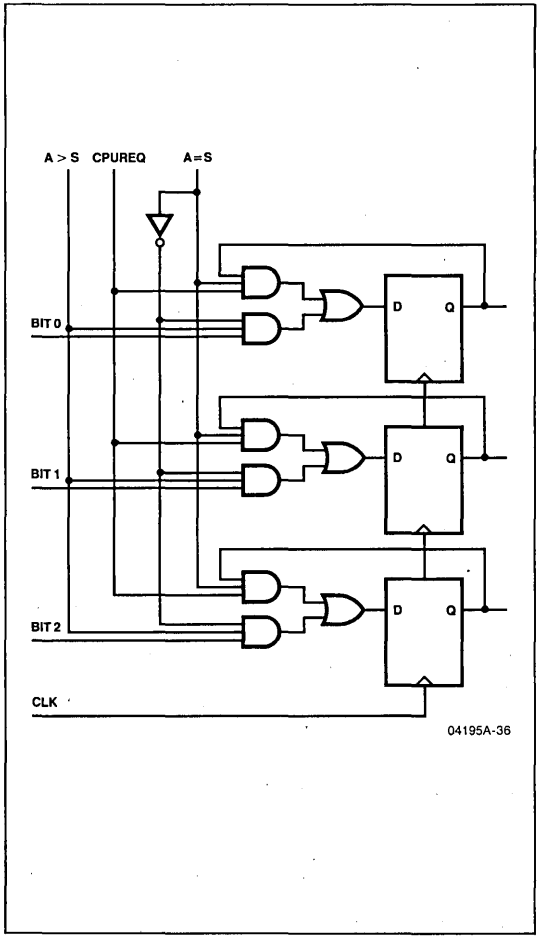


Figure D-5.

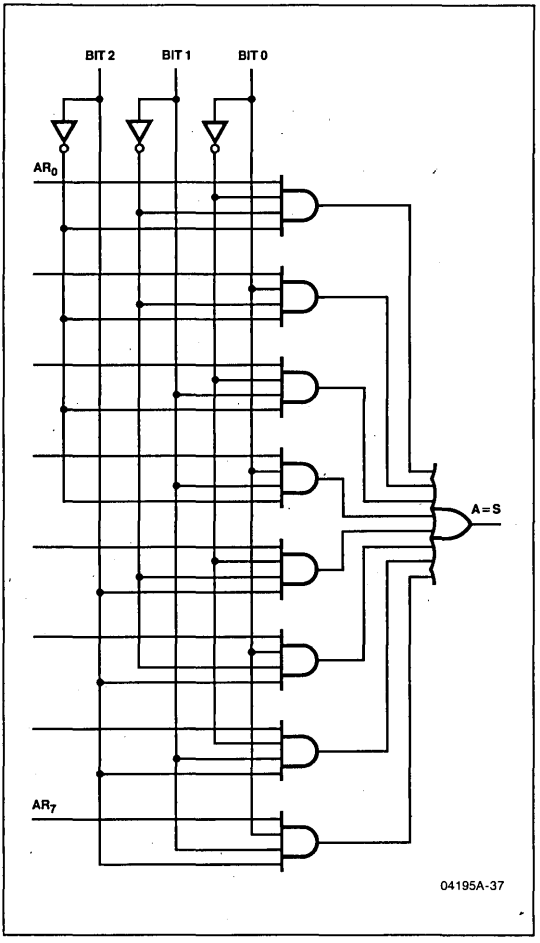


Figure D-6.

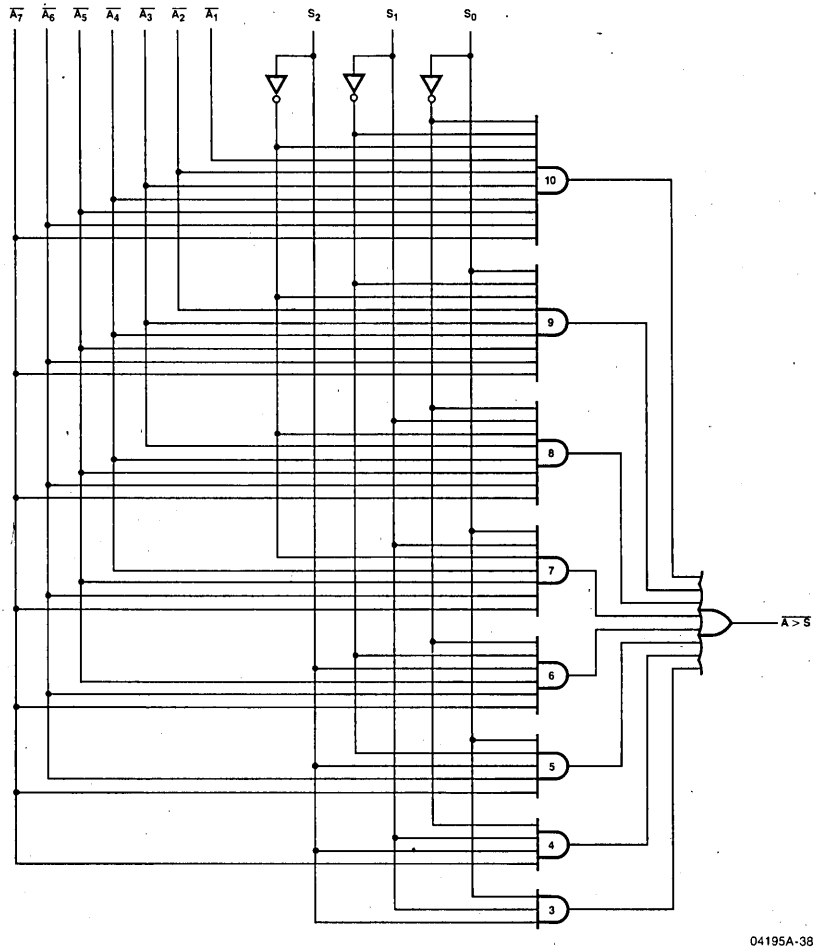


Figure D-6a.

A_{7-0} \ S_{2-0}	000	001	010	011	100	101	110	111
0000 0001	0	0	0	0	0	0	0	0
0000 001X	1	0	0	0	0	0	0	0
0000 01XX	1	1	0	0	0	0	0	0
0000 1XXX	1	1	1	0	0	0	0	0
0001 XXXX	1	1	1	1	0	0	0	0
001X XXXX	1	1	1	1	1	0	0	0
01XX XXXX	1	1	1	1	1	1	0	0
1XXX XXXX	1	1	1	1	1	1	1	0

0 = lowest priority
7 = highest priority

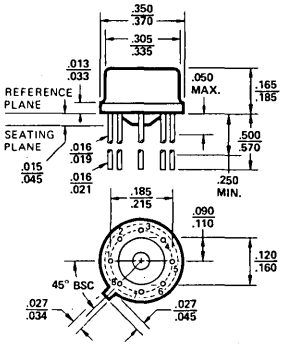
04195A-39

Figure D-6b. A > S K-Map

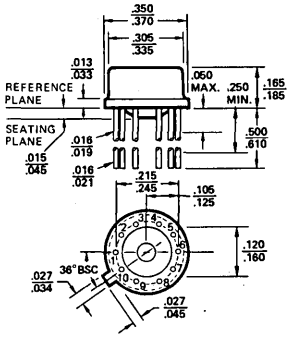
Package Outlines

METAL CAN PACKAGES

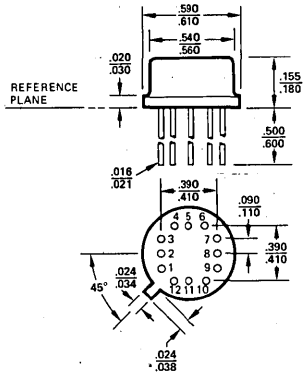
H-8-1



H-10-1



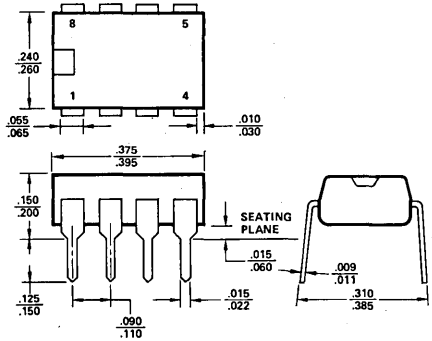
G-12-1



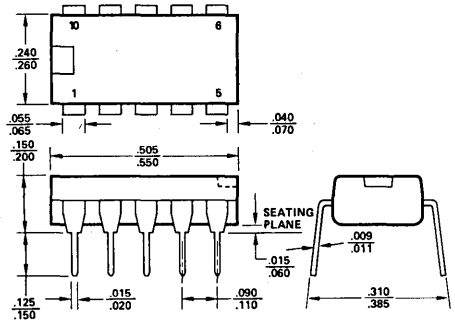
Note. Standard lead finish is bright acid tin plate or gold plate.

MOLDED DUAL IN-LINE PACKAGES

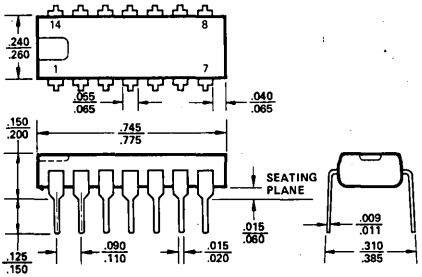
P-8-1



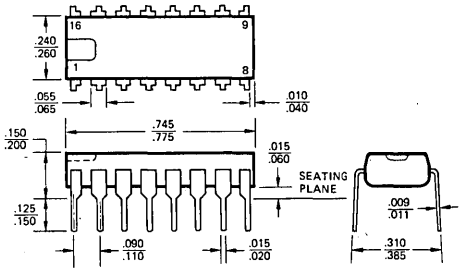
P-10-1



P-14-1



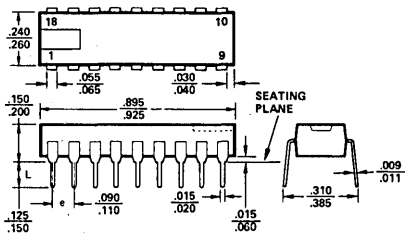
P-16-1



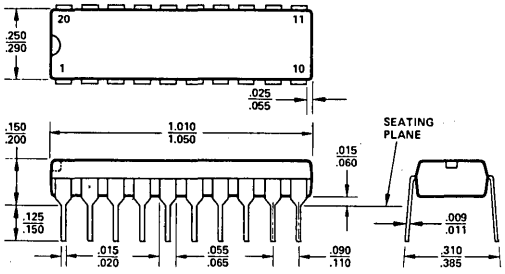
PACKAGE OUTLINES (Cont.)

MOLDED DUAL IN-LINE PACKAGES (Cont.)

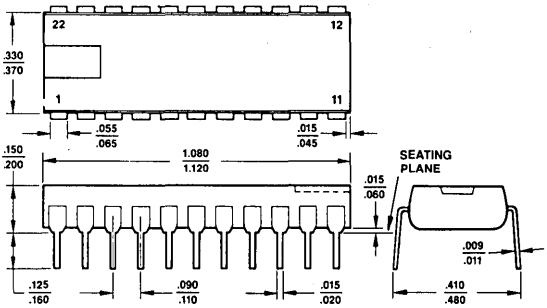
P-18-1



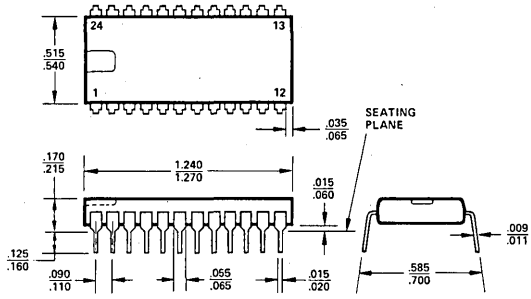
P-20-1



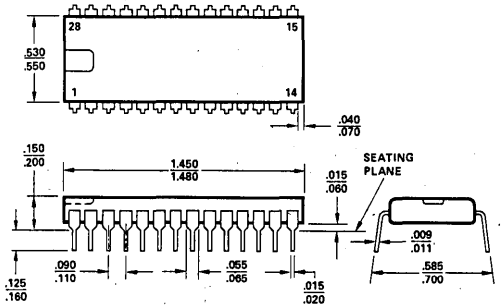
P-22-1



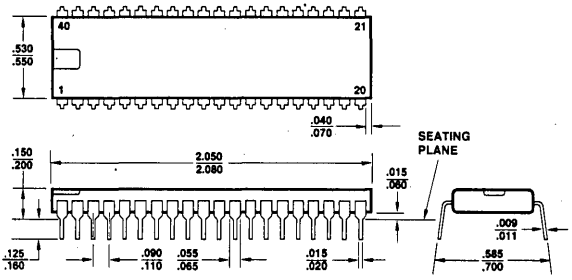
P-24-1



P-28-1



P-40-1

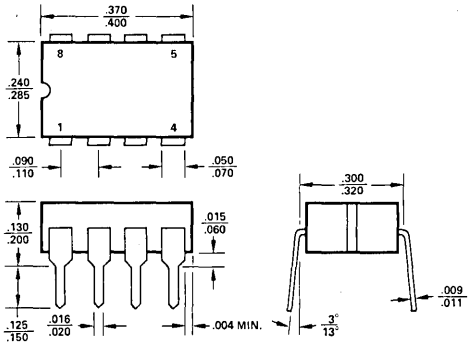


Note . Standard lead finish is tin plate or solder dip.

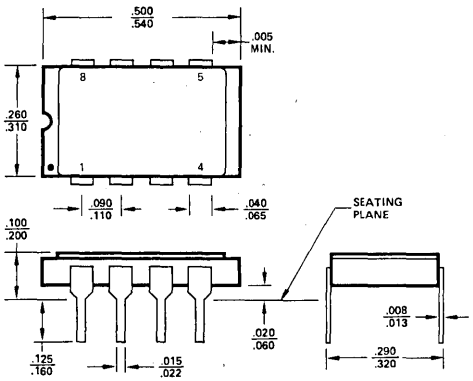
PACKAGE OUTLINES (Cont.)

HERMETIC DUAL IN-LINE PACKAGES

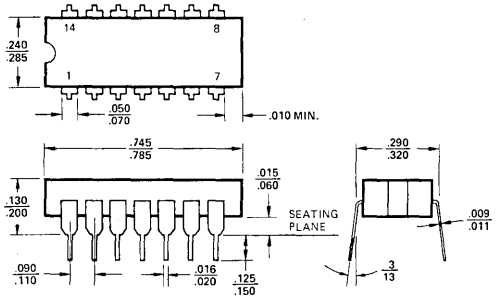
D-8-1



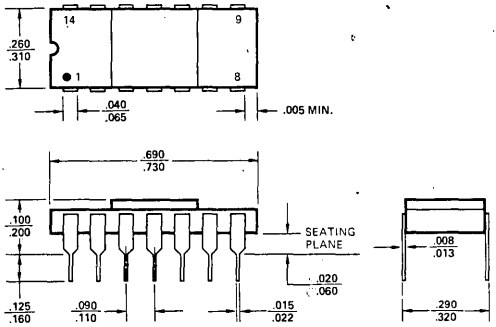
D-8-2



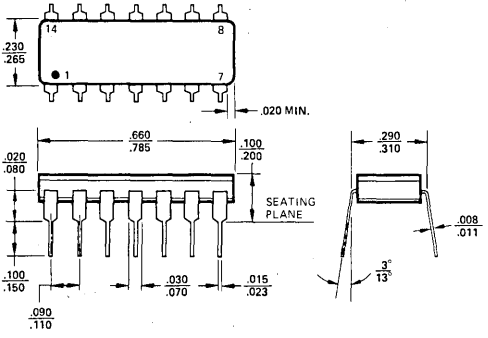
D-14-1



D-14-2



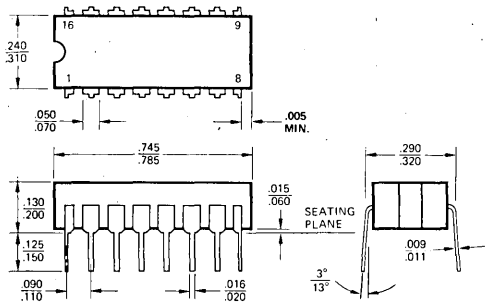
D-14-3



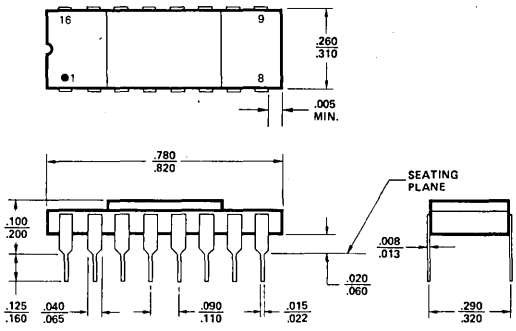
PACKAGE OUTLINES (Cont.)

HERMETIC DUAL IN-LINE PACKAGES (Cont.)

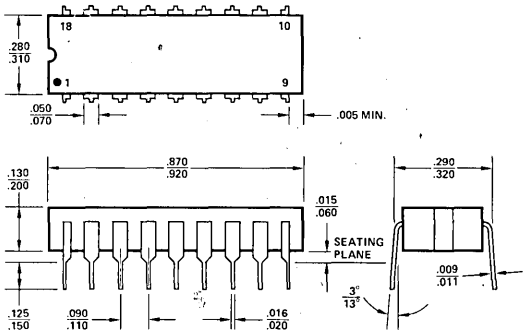
D-16-1



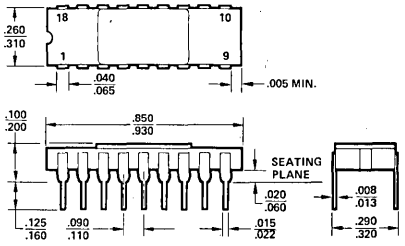
D-16-2



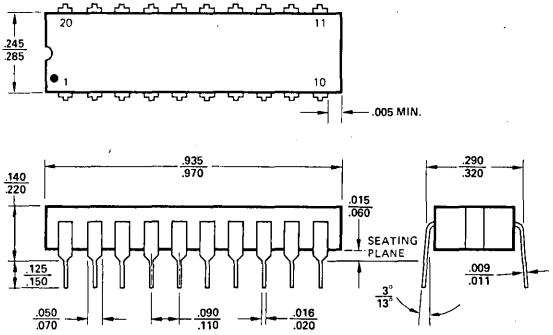
D-18-1



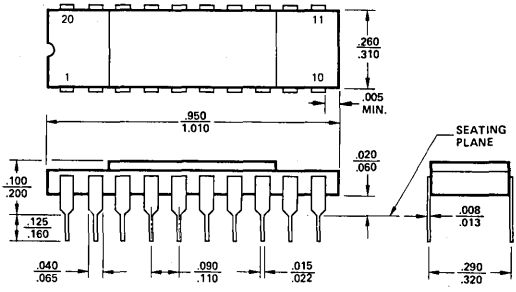
D-18-2



D-20-1



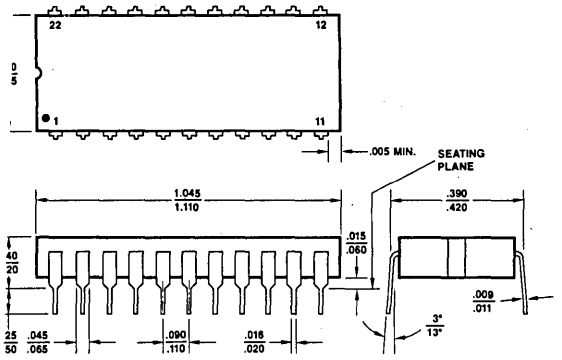
D-20-2



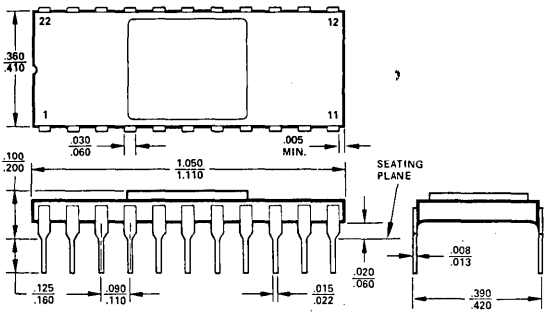
PACKAGE OUTLINES (Cont.)

HERMETIC DUAL IN-LINE PACKAGES (Cont.)

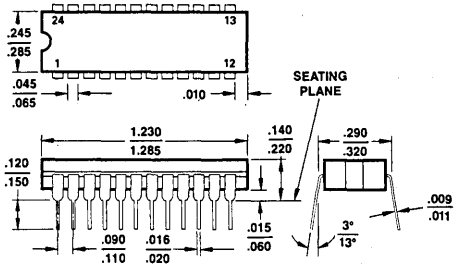
D-22-1



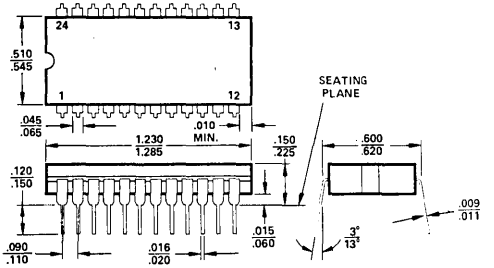
D-22-2



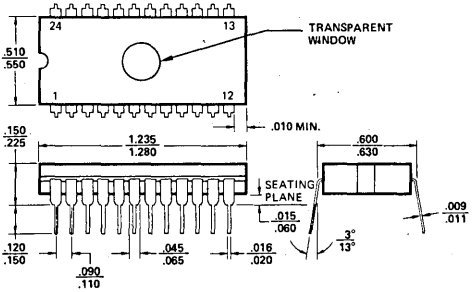
D-24-SLIM



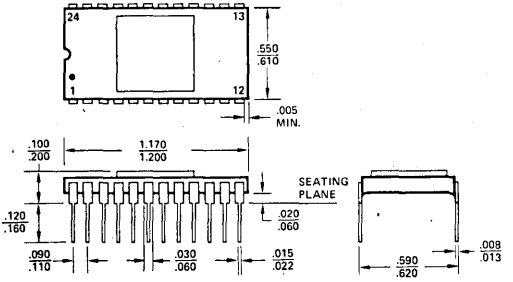
D-24-1 and D-24-4



D-24-4*



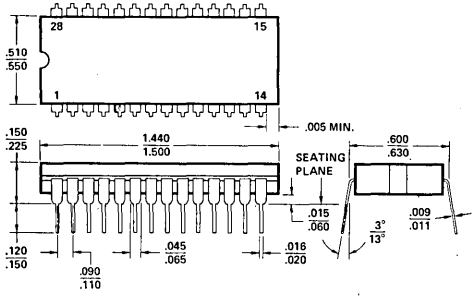
D-24-2



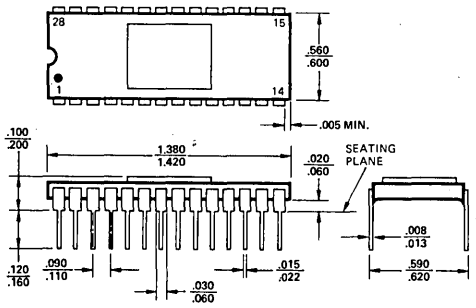
PACKAGE OUTLINES (Cont.)

HERMETIC DUAL IN-LINE PACKAGES (Cont.)

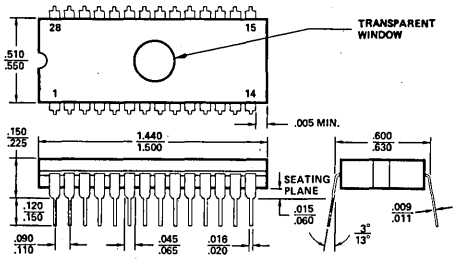
D-28-1



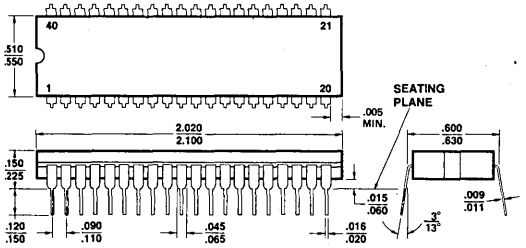
D-28-2



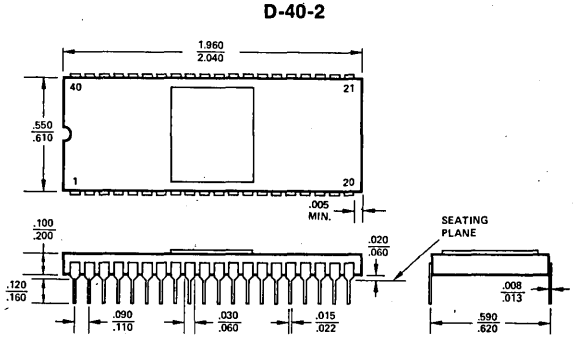
D-28-3



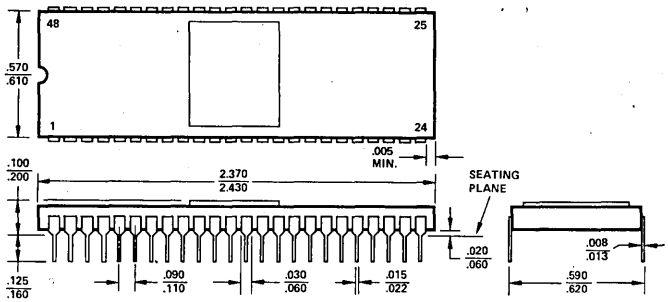
D-40-1



D-40-2



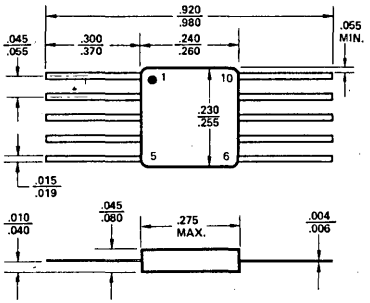
D-48-2



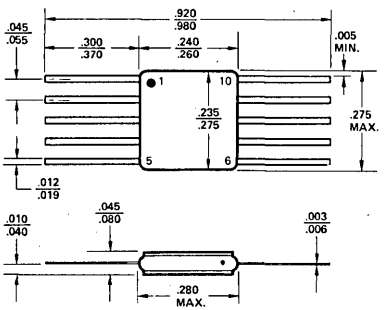
PACKAGE OUTLINES (Cont.)

FLAT PACKAGES

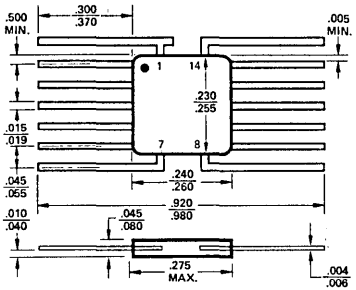
F-10-1



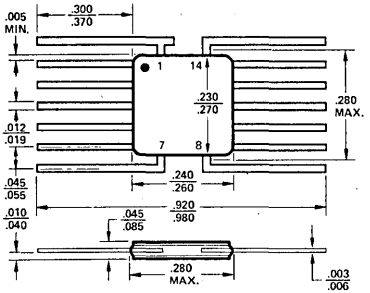
F-10-2



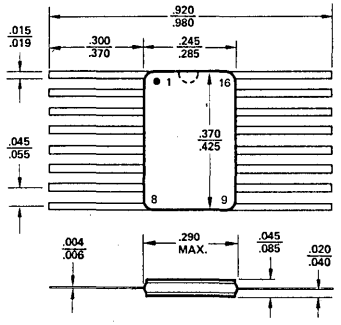
F-14-1



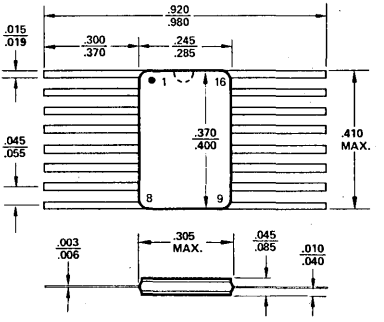
F-14-2



F-16-1

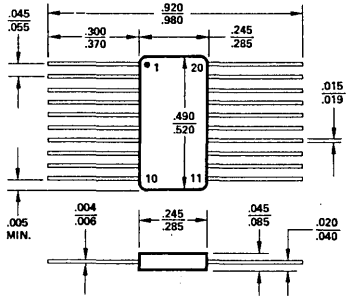


F-16-2

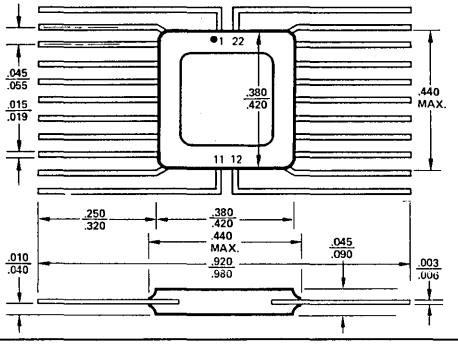


Note: Notch is pin 1 index on cerpack.

F-20-1



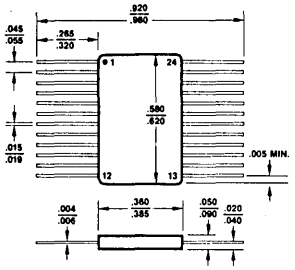
F-22-1



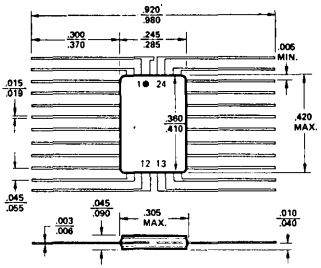
PACKAGE OUTLINES (Cont.)

FLAT PACKAGES (Cont.)

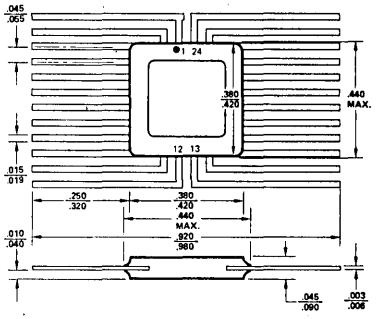
F-24-1



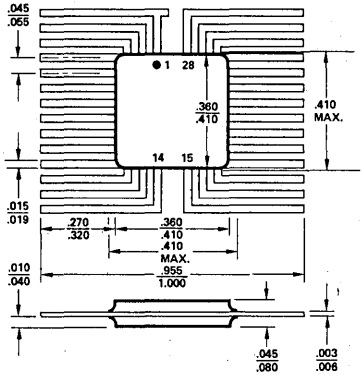
F-24-2



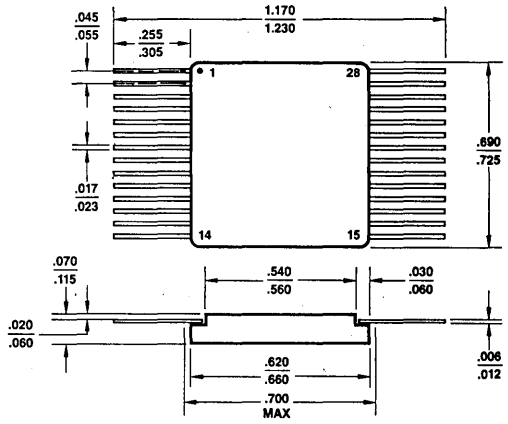
F-24-3



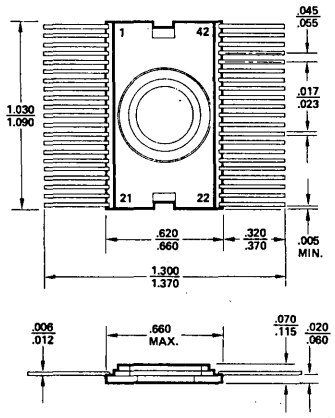
F-28-1



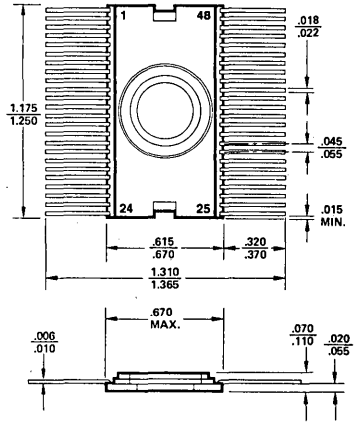
F-28-2 and F-28-3



F-42-1



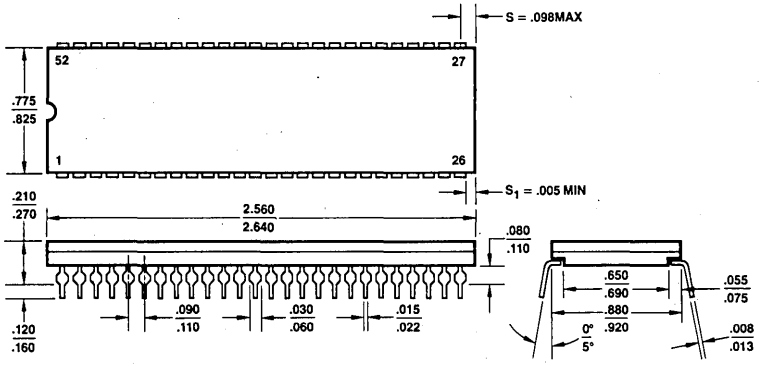
F-48-2



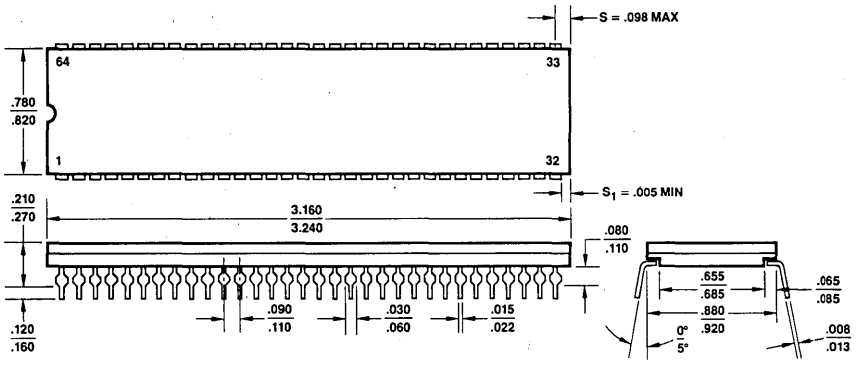
PACKAGE OUTLINES (Cont.)

HERMETIC DUAL IN-LINE PACKAGES (Cont.)

D-52-3



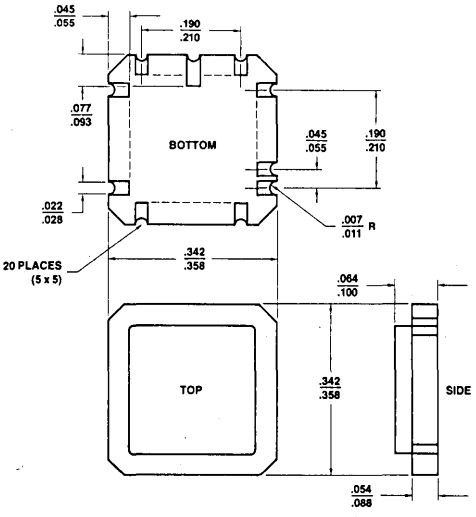
D-64-3



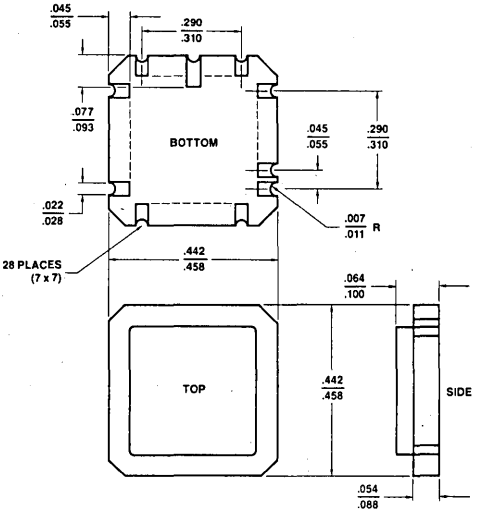
PACKAGE OUTLINES (Cont.)

SQUARE CHIP CARRIER FAMILY

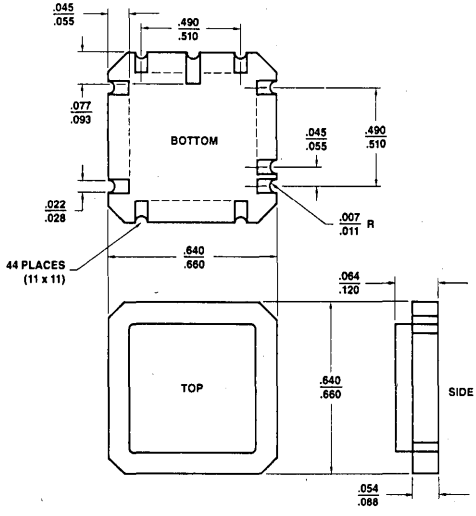
L-20-1



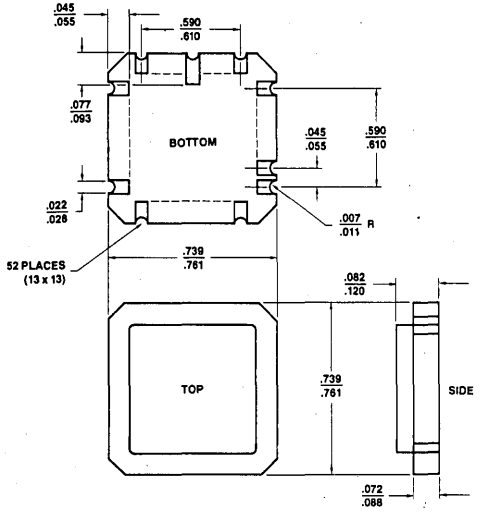
L-28-1



L-44-1



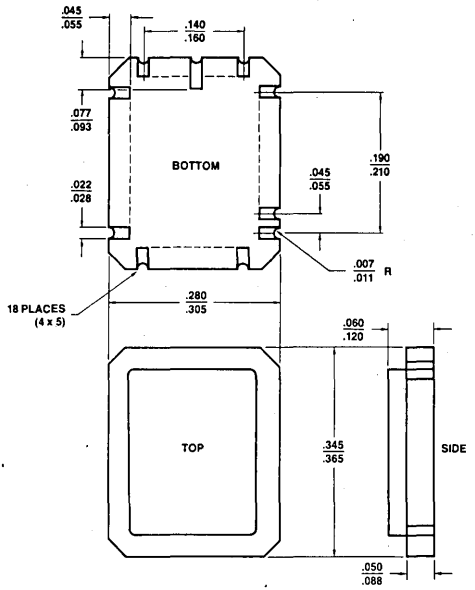
L-52-1



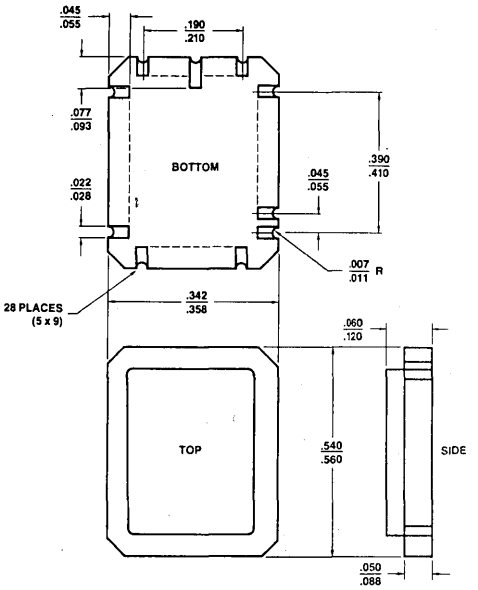
PACKAGE OUTLINES (Cont.)

RECTANGULAR CHIP CARRIER FAMILY

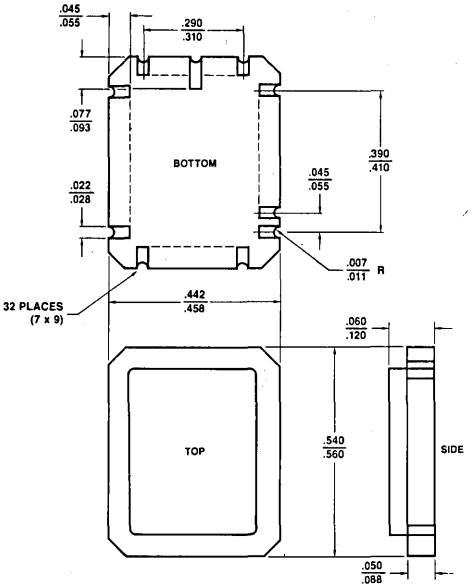
L-18-2



L-28-2



L-32-2



DICE POLICY

Advanced Micro Devices, interface and linear products are all available in dice form.

ELECTRICAL CHARACTERISTICS

Each die is electrically tested to the commercial or military grade DC parameters to guardbanded limits at 25°C to ensure performance over the temperature range.

QUALITY ASSURANCE

All dice are glass passivated with only the bonding pads exposed to provide scratch protection. All dice are provided without gold backing.

SHIPPING PACKAGES/ORDER INFORMATION

All dice are packaged in containers with individual compartments which prevent damage to the die during shipping.

Minimum order for AMD dice is 10 pcs.

SPECIAL CHIP PROCESSING

If there is a need for additional testing or processing, contact AMD for detailed information.

See following pages on ordering information for detail ordering number.

World-Wide Sales Offices

Advanced Micro Devices

Advanced Micro Devices maintains a network of representatives and distributors in the U.S. and around the world. For a sales agent nearest you, call one of the AMD offices listed below.

U.S. AND CANADIAN SALES OFFICES

SOUTHWEST AREA

Advanced Micro Devices
360 N. Sepulveda, Suite 2075
El Segundo, California 90245
Tel: (213) 640-3210

Advanced Micro Devices
10050 N. 25th Street
Suite 235
Phoenix, Arizona 85021
Tel: (602) 242-4400

Advanced Micro Devices
4000 MacArthur Boulevard
Suite 5000
Newport Beach, California 92660
Tel: (714) 752-6262

Advanced Micro Devices
5955 De Soto Avenue, Suite 241
Woodland Hills, California 91367
Tel: (213) 992-4155

Advanced Micro Devices
9455 Ridgehaven Court
Suite 230
San Diego, California 92123
Tel: (619) 560-7030

NORTHWEST AREA

Advanced Micro Devices
2700 Augustine Drive, Suite 109
Santa Clara, California 95051
Tel: (408) 727-1300

Advanced Micro Devices
1873 South Bellaire Street
Suite 920
Denver, Colorado 80222
Tel: (303) 691-5100

NORTHWEST AREA (Cont.)

Advanced Micro Devices
One Lincoln Center, Suite 230
10300 Southwest Greenburg Road
Portland, Oregon 97223
Tel: (503) 245-0080

Advanced Micro Devices
Honeywell Ctr., Suite 1002
600 108th Avenue N.E.
Bellevue, Washington 98004
Tel: (206) 455-3600

MID-AMERICA AREA

Advanced Micro Devices
500 Park Boulevard, Suite 940
Itasca, Illinois 60143
Tel: (312) 773-4422

Advanced Micro Devices
9900 Bren Road East, Suite 601
Minnetonka, Minnesota 55343
Tel: (612) 938-0001

Advanced Micro Devices
3592 Corporate Drive, Suite 108
Columbus, Ohio 43229
Tel: (614) 891-6455

Advanced Micro Devices
8240 MoPac Expressway
Two Park North, Suite 385
Austin, Texas 78759
Tel: (512) 346-7830

Advanced Micro Devices
6750 LBJ Freeway, Suite 1160
Dallas, Texas 75240
Tel: (214) 934-9099

MID-ATLANTIC AREA

Advanced Micro Devices
40 Crossways Park Way
Woodbury, New York 11797
Tel: (516) 364-8020

Advanced Micro Devices
290 Elwood Davis Road
Suite 316
Liverpool, New York 13088
Tel: (315) 457-5400

Advanced Micro Devices
2 Kilmer Road
Edison, New Jersey 08817
Tel: (201) 985-6800

Advanced Micro Devices
107 Lakeside Drive
Horsham, Pennsylvania 19044
Tel: (215) 441-8210
TWX: 510-665-7572

Advanced Micro Devices
205 South Avenue
Poughkeepsie, New York 12601
Tel: (914) 471-8180
TWX: 510-248-4219

NORTHEAST AREA

Advanced Micro Devices
6 New England Executive Park
Burlington, Massachusetts 01803
Tel: (617) 273-3970

Advanced Micro Devices (Canada) Ltd.
2 Sheppard Avenue East
Suite 1610
Willowdale, Ontario
Canada M2N5Y7
Tel: (416) 224-5193

SOUTHEAST AREA

Advanced Micro Devices
Parkway Center
One Parkway Drive Building
7257 Parkway Drive, Suite 204
Dorsey, Maryland 21076
Tel: (301) 796-9310

Advanced Micro Devices
7850 Ulmerton Road, Suite 1A
Largo, Florida 33541
Tel: (813) 535-9811

Advanced Micro Devices
4740 North State Road #7
Suite 102
Ft. Lauderdale, Florida 33319
Tel: (305) 484-8600

Advanced Micro Devices
6755 Peachtree Industrial Boulevard
Suite 104
Atlanta, Georgia 30360
Tel: (404) 449-7920

Advanced Micro Devices
8 Woodlawn Green, Suite 220
Woodlawn Road
Charlotte, North Carolina 28210
Tel: (704) 525-1875

INTERNATIONAL SALES OFFICES

BELGIUM
Advanced Micro Devices
Overseas Corporation
Avenue de Tervueren, 412, bte 9
B-1150 Bruxelles
Tel: (02) 771 99 93
TELEX: 61028

FRANCE
Advanced Micro Devices, S.A.
Silic 314, Immeuble Helsinki
74, rue d'Arcueil
F-94588 Rungis Cedex
Tél: (01) 687.36.66
TELEX: 202053

GERMANY
Advanced Micro Devices GmbH
Rosenheimer Str. 139
D-8000 Muenchen 80
Tel: (089) 40 19 76
TELEX: 05-23883

Advanced Micro Devices GmbH
Harthaeuser Hauptstrasse 4
D-7024 Filderstadt 3
Tel: (07158) 30 60
TELEX: 07-21211

Advanced Micro Devices GmbH
Zur Worth 6
D-3108 Winsen/Aller
Tel: (05143) 53 62
TELEX: 925287

HONG KONG
Advanced Micro Devices
1303 World Commerce Centre
Harbour City
11 Canton Road
Tsimshatsui, Kowloon
Tel: (852) 3 695377
TELEX: 50426
FAX: (852) 123 4276

ITALY
Advanced Micro Devices S.R.L.
Centro Direzionale
Palazzo Vasari, 3° Piano
I-20090 MI2 – Segrate (MI)
Tel: (02) 215 4913-4-5
TELEX: 315286

JAPAN
Advanced Micro Devices, K.K.
Dai 3 Hoya Building
8-17, Kamitakaido 1 chome
Suginami-ku, Tokyo 168
Tel: (03) 329-2751
TELEX: 2324064
FAX: (03) 326 0262

SWEDEN
Advanced Micro Devices AB
Box 7013
S-172 07 Sundbyberg
Tel: (08) 98 12 35
TELEX: 11602

UNITED KINGDOM
Advanced Micro Devices (U.K.) Ltd.
A.M.D. House,
Goldsworth Road,
Woking,
Surrey GU21 1JT
Tel: Woking (04862) 22121
TELEX: 859103

Am Z 8078
Am Z 8060

ADVANCED
MICRO
DEVICES, INC.
901 Thompson Place
P.O. Box 3453
Sunnyvale,
California 94088
(408) 732-2400
TWX: 910-339-9280
TELEX: 34-6306
TOLL FREE
(800) 538-8450

03845A-MMP

LB-RRD-80M-6/83-0