Advanced
Micro
Computers
A subsidiary of
Advanced Micro Devices

Am95/4006 MonoBoard Computer

User's Manual

00680155

$10.00

REVISION RECORD

REVISION	DESCRIPTION
01	Preliminary Issue
(9/19/80)	
A	Manual Released
(10/31/80)	
B	Manual Updated and Reprinted
(6/3/81)	

Publication No.
00680155

REVISION LETTERS I, O, Q AND X ARE NOT USED

Address comments concerning this manual to:

ADVANCED MICRO COMPUTERS
Publications Department
3340 Scott Boulevard
Santa Clara, CA 95051

PREFACE

This manual provides general information, an installation and interface guide, programming information, principles of operation, and service information for the Advanced Micro Computers Am95/4006 MonoBoard Computer. Additional information can be obtained from the following documents.

AMD 8080A/9080A MOS Microprocessor Handbook

AMD Schottky and Low Power Schottky Data Book

Am8251-Am9551 Data Sheet

Am8255A/8255A-5 Data Sheet

Am9511 Data Sheet

Am9512 Data Sheet

Am9513 Data Sheet

Am9513 Application Note

Algorithm Details for the Am9511 Arithmetic Processor Unit

Intel Multibus† Interfacing AP-28A

Using the 8259A Programmable Interrupt Controller AP-59

This manual is intended for use by system designers familiar with microcomputer architecture that utilizes the Multibus. The information presented is sufficient to support normal installation, system interface, and programming needs; a basic theory of operation and schematic diagrams are included to assist in isolating system problems.

†Multibus is a registered trademark of Intel

TABLE OF CONTENTS

FIGURES

TABLES

CHAPTER 1
GENERAL INFORMATION

1-1. INTRODUCTION

The Am95/4006 MonoBoard Computer (MBC) is a complete microcomputer on a single board. It is fully compatible with Intel iSBC board products. The Am95/4006 offers substantially increased performance with the 4 MHz option. Major functional capabilities of the Am95/4006 include the following:

Standard 2MHz and optional 4MHz clock rate.

Either an optional Am9511 Arithmetic Processing Unit (APU) that operates concurrently with the CPU to provide both fixed-point floating-point, and transcendental computional capability, or optional Am9512 Floating-Point Processor Supports floating-point 32-bit and 64-bit operations.

Five general-purpose 16-bit counters that enhance system capability with respect to counting and timing functions.

Eight fully-programmable vectored priority interrupt channels for on board or bus vectored interrupts.

Serial priority bus master for multi-master operation.

Serial Communications Interface with RS232C capability. Baud rates are software selectable.

Parallel I/O Interface with 48 Programmable I/O lines and sockets for drivers and receivers.

4 kilobytes of on-board read/write Random Access Memory (RAM).

Sockets for up to 16 kilobytes of on-board Read Only Memory (ROM/E-PROM)

Memory can be reconfigured by PROM programming.

Bootstrap program can be placed in on-board ROM and selected by power-on or reset and then program disabled.

Optional PROM-resident monitor.

1-2. PHYSICAL DESCRIPTION

The Am95/4006 MonoBoard Computer (MBC) is a four layer printed circuit board with MSI TTL and LSI MOS circuits. Five edge connectors provide bus and peripheral interface capabilities. Physical characteristics of the Am95/4006 are:

Board Dimensions

Width	30.48 cm (12.00 inches)
Depth	17.15 cm (6.75 inches)
Thickness	1.50 cm (0.60 inches)

Environmental Requirements

Operating Temperature	0°C to 55°C
Relative Humidity	Up to 90% without condensation
Storage Temperature	-40°C to +75°C

1-3. FUNCTIONAL DESCRIPTION

The Am95/4006 MonoBoard Computer (MBC) is a complete microcomputer on a single board. The board is fully form-factor and bus compatible with Intel iSBC 80 single board products and is designed to operate with other bus masters in a serial priority multi-master environment. However, while maintaining mechanical and interface compatibility with the iSBC 80 series, the Am95/4006 offers significantly higher throughput and increased computational power over the iSBC 80 boards.

The standard board contains an Am9080A microprocessor which operates at 2MHz, a System Timing Controller, and a programmable eight level priority interrupt system capable of bus vectored interrupts. Additional features include: 4K bytes of RAM memory, sockets for up to 16K bytes of ROM and E-PROM memory, 48 programmable parallel I/O lines with sockets for line drivers or terminators, a programmable synchronous/asynchronous RS232C communication interface, programmable baud rate generator, and bus drivers for off-board memory and input/output expansion. Figure 1-1 is a block diagram of the Am95/4006.

A 4MHz version is available which substantially increases throughput. The 4MHz version provides all the features of the 2MHz board with an Am9080A-4 for 4 MHz operation. Direct addressing of up to 64K bytes of memory is supported by the 16-bit address bus of the Am9080. An external stack, located anywhere in RAM, can be used as a last-in/first-out stack to store the contents of the program counter, flags, accumulator, or any of the six general purpose registers. A 16-bit stack pointer controls the addressing of this external stack, which provides subroutine nesting that is bounded by memory size or 64K bytes, whichever is less.

The Am8224 provides an oscillator, power-up controls, and two clock signals by dividing the frequency of the oscillator by nine. These two clock signals (phase one and phase two) define the CPU minor cycle.

The Am8238 buffers the Data Bus and demultiplexes control signals to generate memory read/write and I/O read/write signals. The on-board memory system provides up to 4K bytes of read/write random access memory using eight Am9114 1024 by 4-bit static memory chips. Up to 16K bytes of read only memory can be installed using four Am9732 or equivalent 4K by 8-bit E-PROM chips. Alternatively, four Am9708 or 9716 equivalent memory chips, or pin compatable ROMs/PROMs, can be used if less on-board memory is required. All of the on-board memory can be disabled to permit off-board memory selection. The address to memory chip location relationship can be changed by programming another address decode PROM. This enables users to insert their existing ROM or PROM resident programs in any memory chip location. Discrete logic associated with the memory system supplies the MEMSEL* and MEMACK* signals. The MEMACK* signal indicates to the CPU that on-board memory has been selected. MEMSEL* indicates to the bus control PROM that on-board memory is selected.

The Am9513 is a programmable system timing controller designed to service many types of counting, sequencing, and timing applications. Five 16-bit counters, two of which can be used as a 24-hour realtime clock, are included on the chip. Each counter can be programmed to count up or count down in binary or BCD. Control of the count modulo is provided by allowing repetitive initialization of the counter from a control register. Any of the counters can be internally concatenated with an adjacent counter.

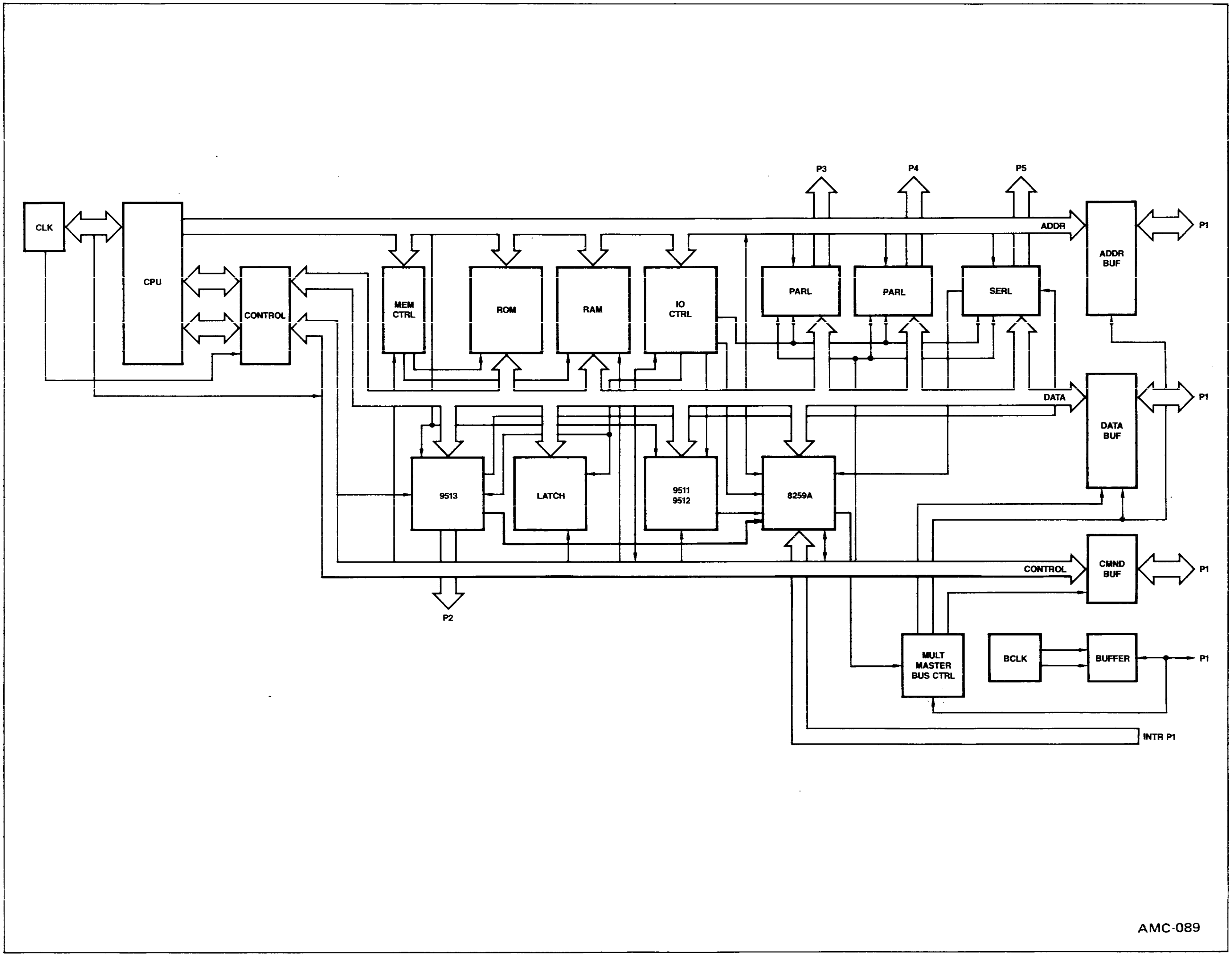


Figure 1-1. Am95/4006 MonoBoard Computer Block Diagram

The timer input, timer gate, and counter output lines are brought off-board on a 60-pin double sided edge connector; data bus input/output lines are connected to the system data bus.

An 8259A provides an eight channel bus vectored interrupt system. Each channel is associated with its own unique three byte response. The eight channels can be programmed to perform priority resolution in fully-nested, special mask, rotating, or polled modes, allowing the user flexibility to establish interrupt service priorities based upon his unique requirements.

An optional Am9511 Arithmetic Processing Unit (APU) and its associated circuitry provide a full complement of fixed and floating point arithmentic and a variety of floating point transcendental and mathematical operations. All internal APU functions can operate concurrently with the CPU. Transfers to and from the APU are handled by the CPU. An End-of-Process signal (EOP*) is issued by the APU, and can be used to produce an interrrupt to the CPU to help coordinate program execution. All transfers (including operand, result, status, and command information) are via the data bus. Operands required for APU operations are received from the data bus and stored in an internal stack. The 8-bit bytes received during a CPU write operation are assembled into 16-bit or 32-bit operands. As each successive operand is assembled in the internal stack, each previous operand is moved down one place in the stack. The last operand entered before an Am9511 command is executed resides at the top of the stack (TOS); the next to last operand entered is next on the stack (NOS). When the Am9511 executes a command, the operands are obtained from TOS and NOS. Operation results are stored on the TOS, and the contents of the TOS are placed on the data bus during a CPU read operation. Each subsequent read operation reads NOS, which becomes TOS when the stack is popped.

An optional Am9512 Floating-Point Processor Unit provides single precision (32-bit) and double precision (64-bit) add, subtract, multiply, and divide operations. The operand, result, status, and command information transfers take place over an 8bit bidirectional data bus. Operands are pushed onto an internal stack by the CPU and a command is issued to perform an operation on the data stack. The results of this operation are available to the CPU by popping the stack.

Information transfers between the Am9512 and the CPU can be handled using programmed Interrupts. After completing an operation, the Am9512 activates its End-of-Execution signal which can interrupt the CPU, via the 8259A.

A programmable communications interface, using an Am9551 USART, provides a standard RS232C communications interface. The Am9551 provides full duplex, double buffered transmit and recieve capabilities. A programmable baud rate generator provides the baud rates. The communications interface can be programmed to implement the desired synchronous or asynchronous serial data transmission protocol. Data format, control character format, parity and transmission rate are all under program control. Parity, overrun, and framing error detection are all incorporated on the programmable communications interface. Command and control lines, serial data lines, and signal ground lines are brought out to a 26-pin RS232C compatible connector.

The system contains 48 programmable parallel I/O lines implemented by using two Am8255A Programmable Peripheral Interface chips. System software can configure the I/O lines to sets of input, output or bidirectional input/output ports. To take full advantage of the large number of possible I/O configurations, sockets are provided for interchangeable I/O line drivers and terminators. Hence, the

I/O interface is further enhanced by the capability of selecting the appropriate combination of optional line drivers and termination characteristics for each application. The programmable I/O lines and signal ground lines are brought out to two 50-pin edge connectors.

1-4. SPECIFICATIONS

Specifications for the Am95/4006 MonoBoard Computer are listed in table 1-1.

TABLE 1-1. SPECIFICATIONS

Word Size	
Instruction:	8, 16 or 24 bits
Data:	8 bits
Memory Addressing	
On-Board ROM/E-PROM:	0 to 0FFFH (1K Byte Devices) 0 to 1FFFH (2K Byte Devices) 0 to 3FFFH (4K Byte Devices)
On-Board RAM:	3000H-3FFFH (1K Byte and 2K Byte Devices) 4000H to 4FFFH (4K Byte Devices)
	All on-board addresses can be disabled for off-board memory expansion.
Memory Capacity	
On-Board ROM/E-PROM:	Sockets for 16K bytes (using 4K byte devices)
On-Board RAM:	4K bytes static (eight Am9114 chips)
Off-Board Expansion:	Up to 64K bytes
Serial I/O Address (Am9551)	
Control:	EDH
Data:	ECH
Parallel I/O Address (Am8255)	
Connector P3:	CONTROL: E7H Port A: E4H Port B: E5H Port C: E6H
Connector P4:	CONTROL: EBH Port A: E8H Port B: E9H Port C: EAH
Parallel I/O Capacity:	48 Programmable lines

TABLE 1-1. SPECIFICATIONS (Cont.)

Serial Communications (Am9551)	
Characteristics	5 to 8-bit characters
Synchronous:	Internal or External Character Synchronization
Asynchronous:	5 to 8-bit characters Break Character Generation 1, 1 1/2, or 2 Stop bits False start bit detector
Serial Baud Rates:	Program Selectable (Normally 9600 Baud)
Interrupt Controller (8259A)	
Addressing	
Control:	C2H
Data:	C3H
System Timing Controller (Am9513)	
Addressing	
Control:	D9H
Data:	D8H
Programmable Latch:	E0H
Arithmetic Processing Unit or Floating-Point Processor Addressing	
Control:	C1H
Data:	C0H
Power Requirements	
V_{CC}	+5 V ±5%
V_{DD}	+12 V ±5%
V_{BB}	-5 V ±5%
V_{AA}	-12 V ±5%

	WITHOUT ROM MEMORY	
	MAX	TYPICAL
I_{CC}	3.2 A	2.0 A
I_{DD}	300.0mA	190.0mA
I_{BB}	1.0mA	1.0mA
I_{AA}	25.0mA	20.0mA

NOTE

A -5 volt regulator is used to supply -5 volts to the Am9080.

TABLE 1-1. SPECIFICATIONS (Cont.)
Am9511 Command Execution Times

COMMAND MNEMONIC	DESCRIPTION	μSec Am95/4006-2,-4
ACOS	32-bit floating-point inverse cosine	2565 - 3370
ASIN	32-bit floating-point inverse sine	2535 - 3231
ATAN	32-bit floating-point inverse tangent	2031 - 2659
CHSD	32-bit fixed-point sign change	9.8 - 11.4
CHSF	32-bit floating-point sign change	6.5 - 8.1
CHSS	16-bit fixed-point sign change	8.1 - 9.8
COS	32-bit floating-point cosine	1562 - 1986
DADD	32-bit fixed-point add	8.1 - 9.8
DDIV	32-bit fixed-point divide	80 - 86
DMUL	32-bit fixed-point multiply, lower	78 - 86
DMUU	32-bit fixed-point multiply, upper	75 - 91
DSUB	32-bit fixed-point subtract	14.6 - 16.3
EXP	32-bit floating-point exponentiation	1545 - 1986
FADD	32-bit floating-point add	23 - 150
FDIV	32-bit floating-point divide	63.5 - 74.9
FIXD	32-bit floating-point to 32-bit fixed-point conversion	37 - 137
FIXS	32-bit floating-point to 16-bit fixed-point conversion	37 - 88
FLTD	32-bit fixed-point to 32-bit floating-point conversion	23 - 140
FLTS	16-bit fixed-point to 32-bit floating-point conversion	26 - 64
FMUL	32-bit floating-point multiply	60.2 - 68.4
FSUB	32-bit floating-point subtraction	29 - 151
LOG	32-bit floating-point common logarithm	1821 - 2902
LN	32-bit floating-point natural logarithm	1742 - 2830
NOP	No operation	1.6
POPD	32-bit stack pop	4.9
POPF	32-bit stack pop	4.9
POPS	16-bit stack pop	4.1
PTOD	Push 32-bit TOS onto stack	8.1
PTOF	Push 32-bit TOS onto stack	8.1
PTOS	Push 16-bit TOS onto stack	6.5
PUPI	Push 32-bit floating-point π onto TOS	6.5
PWR	32-bit floating-point X to the Y power	3374 - 4896
SADD	16-bit fixed-point add	6.5 - 73
SDIV	16-bit fixed-point divide	34 - 39
SIN	32-bit floating-point sine	1544 - 1956
SMUL	16-bit fixed-point multiply, lower	34 - 39
SMUU	16-bit fixed-point multiply, upper	33 - 40
SQRT	32-bit floating-point square root	319 - 355
SSUB	16-bit fixed-point subtract	11.4 - 13.0
TAN	32-bit floating-point tangent	1992 - 2396
XCHD	Exchange 32-bit stack operands	10.6
XCHS	Exchange 16-bit stack operands	7.3

TABLE 1-1. SPECIFICATIONS (Cont.)
Am9512 Command Execution Times

COMMAND MNEMONIC	DESCRIPTION	µSec Am95/4006-2,-4
SADD	Add TOS to NOS Single Precision and result to NOS. Pop stack.	18.88
SSUB	Subtract TOS from NOS Single Precision and result to NOS. Pop stack.	18.23
SMUL	Multiply NOS by TOS Single Precision and result to NOS. Pop stack.	64.45
SDIV	Divide NOS by TOS Single Precision and result to NOS. Pop stack.	72.22
CHSS	Change sign to TOS Single Precision operand.	3.26
PTOS	Push Single Precision operand on TOS to NOS.	5.21
POPS	Pop Single Precision operand from TOS. NOS becomes TOS.	4.56
XCHS	Exchange TOS with NOS Single Precision.	8.46
CHSD	Change sign of TOS Double Precision operand.	7.81
PTOD	Push Double Precision operand on TOS to NOS.	13.02
POPD	Pop Double Precision operand from TOS. NOS becomes TOS.	8.46
CLR	CLR status.	1.30
DADD	Add TOS to NOS Double Precision and result to NOS. Pop stack.	188.15
DSUB	Subtract TOS from NOS Double Precision and result to NOS. Pop stack.	188.15
DMUL	Multiply NOS by TOS Double Precision and result to NOS. Pop stack.	569.01
DDIV	Divide NOS by TOS Double Precision and result to NOS. Pop stack.	1484.38

CHAPTER 2
INSTALLATION AND INTERFACE

2-1. INTRODUCTION

This section provides information for installing and interfacing the Am95/4006 MonoBoard Computer (MBC). These instructions include unpacking and inspection, power requirements, cooling requirements, user selectable options, bus interface characteristics, and connector pin assignments.

2-2. UNPACKING AND INSPECTION

Inspect the shipping carton immediately upon receipt for evidence of mishandling during transit. If the shipping carton is severely damaged or water-stained, request the carrier's agent to be present when the carton is opened. If the carrier's agent is not present when the carton is opened and the contents of the carton are damaged, keep the carton and packing material for the agent's inspection. Shipping damages should be reported immediately to the carrier.

NOTE

Do not attempt to service the board yourself as this will void the warranty.

It is suggested that salvageable shipping cartons and packing materials be saved for use in case the product must be shipped in the future.

2-3. POWER REQUIREMENTS

The Am95/4006 requires +5, +12V, and -12V power supply inputs. The current required from these supplies is listed in table 1-1. Ensure that the power supply has sufficient current to accomodate the Am95/4006 requirements.

2-4. COOLING REQUIREMENTS

The Am95/4006 dissipates approximately 500 gram-calories/minute (2.1 Btu/minute), and adequate air circulation must be provided to prevent a temperature rise above 55°C (130°F).

2-5. USER SELECTABLE OPTIONS

The Am95/4006 is designed as a general purpose microcomputer; therefore, several optional jumpers might be necessary before operation. The following paragraphs provide instructions for optional jumper configuration.

2-6. SERIAL I/O INTERFACE

The serial I/O interface is designed to interface RS232C devices. The configuration, as shipped from the factory, is jumpered for an RS232C interface as shown in table 2-1. The Am9551 can be configured to function as a data set or a data processing terminal. Connector pin assignments for connecting the P5 connector to a terminal and a modem are shown in tables 2-2 and 2-3.

2-7. BAUD RATE SELECTION

An Am9513 System Timing Controller is programmed to provide a 9600 baud clock for the serial I/O interface. A 16-bit internal software controlled scaling counter divides the on-chip oscillator frequency in binary or BCD steps. Additional information is provided in Chapter 3; Operation and Programming.

TABLE 2-1. RS232C INTERFACE JUMPERS

9551 Pin	Mnemonic	Jumper	Connection
17	CTS*	52-48	P5-7 (Request to send)
9	TxC	65-61	U72-7 (CLOCK)
25	RxC	63-62	U72-7 (CLOCK)
23	RTS	51-50	P5-9 (Clear To Send)

TABLE 2-2. WIRING LIST FOR A CABLE BETWEEN P5 AND A TERMINAL

FROM		TO		FROM		TO	
(P5) Pin	MonoBoard Signal	RS232C Signal	DB-25 Pin No.	(P5) Pin	MonoBoard Signal	RS232C Signal	DB-25 Pin No.
1	CHASSIS GND	PROTECTIVE GROUND	1	14	DATA TERM READY/ TX CLK	DATA TERM RDY	20
2	Not Used	Not Used	14	15	DATA CARRIER DET	DATA CARRIER DET	8
3	TRANSMITTED DATA	TRANSMITTED DATA	2	16	Not Used	Not Used	21
4	Not Used	Not Used	15	17	Not Used	Not Used	9
5	RECEIVED DATA	RECEIVED DATA	3	18	Not Used	Not Used	22
6	Not Used	Not Used	16	19	Not Used	Not Used	10
7	REQUEST TO SEND	REQUEST TO SEND	4	20	Not Used	Not Used	23
8	Not Used	Not Used	17	21	Not Used	Not Used	11
9	CLEAR TO SEND	CLEAR TO SEND	5	22	RX CLK	Not Used	24
10	Not Used	Not Used	18	23	Not Used	Not Used	12
11	DATA SET READY	DATA SET READY	6	24	Not Used	Not Used	25
12	Not Used	Not Used	19	25	Not Used	Not Used	13
13	SIGNAL GND	SIGNAL GND	7	26	SIGNAL GND	Not Used	--

TABLE 2-3. WIRING LIST FOR A CABLE BETWEEN P5 AND A MODEM

FROM		TO		FROM		TO	
(P5) Pin	MonoBoard Signal	RS232C Signal	DB-25 Pin No.	(P5) Pin	MonoBoard Signal	RS232C Signal	DB-25 Pin No.
1	CHASIS GND	PROTECTIVE GROUND	1	14	DATA TERM RDY/ TX CLK	DATA SET RDY/ TX CLK	6/15
2	Not Used	Not Used	--	15	DATA CARRIER DET	Not Used	--
3	TRANSMITTED DATA	RECEIVED DATA	3	16	Not Used	Not Used	--
4	Not Used	Not Used	--	17	Not Used	Not Used	--
5	RECEIVED DATA	TRANSMITTED DATA	2	18	Not Used	Not Used	--
6	Not Used	Not Used	--	19	Not Used	Not Used	--
7	REQUEST TO SEND	CLEAR TO SEND	5	20	Not Used	Not Used	--
8	Not Used	Not Used	--	21	Not Used	Not Used	--
9	CLEAR TO SEND	REQUEST TO SEND	4	22	RX CLK	RX CLK	17
10	Not Used	Not Used	--	23	Not Used	Not Used	--
11	DATA SET READY	DATA TERMINAL READY	20	24	Not Used	Not Used	--
12	Not Used	Not Used	--	25	Not Used	Not Used	--
13	SIGNAL GND	SIGNAL GND	7	26	SIGNAL GND	Not Used	--

2-8. SYSTEM TIMING CONTROLLER OUTPUT JUMPERS

Jumpers are provided to allow 9513 output signals to be connected, through a latch, to the 8259A Programmable Interrupt Controller. As shown in table 2-4, any three of the six System Timing Controller output signals can be connected to the interrupt control latch. When the latches have been cleared and then enabled, as described in chapter 3, the selected System Timing Controller output sets the interrupt control latch. The output of the interrupt control latch can be connected, through the jumpers shown in table 2-5, to the Interrupt Controller. As shipped, the board is configured with the OUT1 signal jumpered to timer interrupt 1 (123-128), the OUT2 signal jumpered to timer interrupt 2 (124-129), and the OUT3 signal jumpered to timer interrupt 3 (121-127).

2-9. INTERRUPT SELECTION JUMPERS

The priority interrupt jumper matrix provides for eight out of sixteen possible interrupts to be jumpered to the eight interrupt controller inputs. Table 2-5 shows the possible jumper configurations for the interrupt controller inputs.

2-10. PARALLEL I/O JUMPER OPTION

The parallel I/O section is configured for Am8216/8226 bidirectional bus drivers at ports E4H and E8H. As delivered from the factory, jumpers are installed between jumper pins 2 and 3, and between jumper pins 25 and 35; this ties the Am8216/8226 DIEN* inputs to pin 6 of ports E6H and EAH, thereby configuring both ports E4H and E8H as bidirectional ports. Either or both ports can be configured as input or output ports with the following changes.

CONFIGU-RATION	PORT	REMOVE	INSTALL
Input	E4H	2 and 3	1 and 2
Input	E8H	25 and 35	25 and 24
Output	E4H	2 and 3	2 and 4
Output	E8H	25 and 35	25 and 34

All lines for ports E6H and EAH are jumper connected to their line driver/terminator sockets. This allows complete flexibility for signal interchange when operating in mode 2. A description of operating modes is presented in chapter 3.

2-11. MEMORY SELECTION

To customize the Am95/4006 for the type of memory devices being used, jumper connections must be made as shown on table 2-6. The memory address to memory device relationship is controlled by the address decoder PROM at location U40. As delivered, the PROM is programmed as shown in Appendix D, table D-1. Information on how to program the PROM for unique system requirements is presented in chapter 3.

When using the standard Mapping PROM; 2K by 8 or 4K by 8 PROMs can be used in location U43, U44, U45, and U46. To use 2K by 8 PROMs, jumper pins 73 - 74 must be open. Connect a jumper between 73-74 when using 4K by 8 PROMs.

When using the optional mapping PROM (shown in Appendix D, table D-2), no jumper should be connected between jumper pins 73-74 to use 2K by 8 PROMs. A jumper should be connected between jumper pins 73 and 74 to use 1K by 8 PROMs.

TABLE 2-4. SYSTEM TIMING CONTROLLER OUTPUT JUMPERS

9513 OUTPUT	JUMPER PIN	JUMPER PIN	LATCH INPUT SIGNAL	TIMER INTERRUPT
OUT5	125			
OUT4	122	127	TINT3	TI3
OUT3	121	129	TINT2	TI2
OUT2	124	128	TINT1	TI1
OUT1	123			
FOUT	126			

TABLE 2-5. PRIORITY INTERRUPT JUMPERS

Signal	Pin No.	Column 1 Jumper Pins	Column 2 Jumper Pins	INT Input
TI0	On-Board	118		
TI1	On-Board	119		
TI2	On-Board	117		
TI3	On-Board	120		
INT 11*	On-Board	106		
INT 51A* (TxRDY)	On-Board	108	103	IR0
INT 51B* (RxRDY)	On-Board	107	97	IR1
ERROR* (9511/9512)	On-Board	105	98	IR2
IRQ7*	P1-36	109	104	IR3
IRQ6*	P1-35	110	99	IR4
IRQ5*	P1-38	111	102	IR5
IRQ4*	P1-37	112	101	IR6
IRQ3*	P1-40	113	100	IR7
IRQ2*	P1-39	114		
IRQ1*	P1-42	115		
IRQ0*	P1-41	116		
Jumper any column 1 pin to any column 2 pin.				

2-12. 9511/9512 END JUMPER

The End output on the 9511 is an active low signal and the End output on the 9512 is an active high signal. To resolve this signal inconsistancy, it is necessary to install a jumper between jumper pins 78 and 79 when a 9511 is being used on the board and to leave jumper pins 78 and 79 open when no 9511 is being used. Leave Jumper off when using an Am9512.

2-13. Am9080A READY TIMING OPTION

The Am95/4006 is capable of utilizing an advance memory acknowledge (AACK*) signal to increase program throughout. When used with the Am96/1000 Series RAM board, the AACK* signal will procude an improvement in program execution time of about 17%. Extreme care must be exercised when using the advance acknowledge capability of other boards due to the absence of a precise

TABLE 2-6. MEMORY JUMPER CONNECTIONS

FUNCTION	Am9708	Am9716	Am9732	2758
Address	73- 74	---	73- 74	73- 74
+12V to pin 19	138-140	---	---	---
GND to pin 19	---	---	---	140-141
A10	---	140-141	134-141	---
A11	---	---	133-136	---
-5V to pin 21	133-135	---	---	---
+5V to pin 21	---	136-135	---	---
To use 2708 or 2758 E-PROMS the user must program a memory mapping PROM using the optional bit mask shown in Appendix D, table D-2.				

precise advance acknowledge signal definition. To use the advance acknowledge capability, connect a jumper between jumper pins 88 and 89. When AACK* is not to be used, no jumper should be connected between pins 87, 88, or 89.

The Delay 50 AACK* jumper on the Am96/1000 Series RAM board should be selected for AACK* operation with the Am95/4006 MonoBoard.

2-14. MULTI-MASTER CONTROL

The board, as shipped, is configured for serial bus priority resolution; a jumper is connected between jumper pins 86 and 85.

2-15. INTERFACE REQUIREMENTS

The following paragraphs identify the board external connections and bus signal characteristics and timing.

2-16. SERIAL I/O INTERFACE

The serial I/O interface communicates with an external device via 26-pin PC edge connector P5. An external device can be connected to P5 using a 3M 34620001 flat cable connector or a TI H312113 or AMP 1-583715-1 solder connector. When connected to a DB-25 connector, the connector pins are numbered differently. Tables 2-2 and 2-3 are is a pin lists for connector P5 and includes a cross reference to standard RS232C pin numbering.

2-17. PARALLEL I/O INTERFACE

The parallel I/O interface communicates with external I/O devices via two 50 pin double sided edge connectors P3 and P4. External devices can be attached to P3 or P4 using one of the mating connectors listed in table 2-7. Tables 2-8 and 2-9 provide a pin list for connectors P3 and P4. TTL line drivers compatible with the I/O driver sockets interface are listed in table 2-10. Parallel I/O interface lines can be terminated by either a 220Ω/330Ω divider or a 1KΩ pull-up as shown in figure 2-1. The 220Ω/330Ω divider is stocked by distributors under Intel part number iSBC901 and National Semiconductor part number BLC-901. The 1KΩ pull-up is stocked under Intel part number iSBC902 and National Semiconductor part number BLC-902.

2-18. BCLK AND CCLK JUMPERS

For the Am95/4006 to provide the BCLK* (Bus Clock), connect a jumper from pin 83 to 84. No jumper should be connected if BCLK* is supplied from another source. Connect a jumper from 90 to 91 for CCLK* (Constant Clock) from the Am95/4006. No jumper should be connected if CCLK* is supplied from another source.

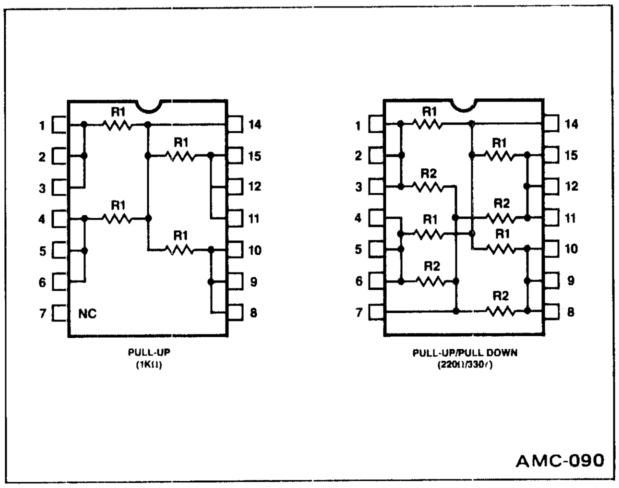


Figure 2-1. Parallel I/O Line Terminator Packs

TABLE 2-7. PARALLEL I/O MATING CONNECTORS

CONNECTOR TYPE	VENDOR	PART NUMBER
Flat Cable	3M	3415-0001
	AMP	2-86792-3
Soldered	AMP	2-583715-3
	VIKING	3VH25/1JV-5
	TI	H312125
Wire-wrap	TI	H311125
	VIKING	3VH25/1JND-5
	CDC	VPB01B25D00A1
	ITT	EC4A050A1A
Crimp	AMP	1-583717-1

TABLE 2-8. PARALLEL I/O CONNECTOR P3 PIN ASSIGNMENTS

PIN		SIGNAL	PIN	SIGNAL
1		Bit 7	2	GND
3		Bit 6	4	
5		Bit 5	6	
7		Bit 4	8	
9	Port E5	Bit 3	10	
11		Bit 2	12	
13		Bit 1	14	
15		Bit 0	16	
17		Bit 3	18	
19		Bit 2	20	
21		Bit 1	22	
23	Port E6	Bit 0	24	
25		Bit 4	26	
27		Bit 5	28	
29		Bit 6	30	
31		Bit 7	32	
33		Bit 7	34	
35		Bit 6	36	
37		Bit 5	38	
39		Bit 4	40	
41	Port E4	Bit 3	42	
43		Bit 2	44	
45		Bit 1	46	
47		Bit 0	48	
49			50	GND

2-19. AUXILIARY CONNECTOR P2

Connector P2 is a 60-pin double sided edge connector that provides interface to the System Timing Controller. Table 2-11 is a pin list for connector P2.

2-20. BUS INTERFACE

This section describes the signals that interface the Am95/4006 to the external system bus. signals shown with an asterisk (*) following the signal name are active-low signals.

TABLE 2-9. PARALLEL I/O CONNECTOR P4 PIN ASSIGNMENTS

PIN		SIGNAL	PIN	SIGNAL
1	Port E9	Bit 7	2	GND
3		Bit 6	4	
5		Bit 5	6	
7		Bit 4	8	
9		Bit 3	10	
11		Bit 2	12	
13		Bit 1	14	
15		Bit 0	16	
17	Port EA	Bit 3	18	
19		Bit 2	20	
21		Bit 1	22	
23		Bit 0	24	
25		Bit 4	26	
27		Bit 5	28	
29		Bit 6	30	
31		Bit 7	32	
33	Port E8	Bit 7	34	
35		Bit 6	36	
37		Bit 5	38	
39		Bit 4	40	
41		Bit 3	42	
43		Bit 2	44	
45		Bit 1	46	
47		Bit 0	48	
49		------	50	GND

TABLE 2-10. PARALLEL I/O SOCKET COMPATIBLE LINE DRIVERS

DRIVER	CHARACTERISTIC	SINK CURRENT
7438	I, OC	48mA
7437	I	48mA
7432	NI	16mA
7426	I, OC	16mA
7409	NI, OC	16mA
7408	NI	16mA
7403	I, OC	16mA
7400	I	16mA

Note: I = inverting; NI = non-inverting; OC = open collector

TABLE 2-11. CONNECTOR P2 PIN ASSIGNMENTS

PIN	MNEMONIC	FUNCTION
15	SRC5	Source (Inputs)
13	SRC4	
53	SRC3	
51	SRC2	
49	SRC1	
39	GATE5	Gate (Inputs)
41	GATE4	
43	GATE3	
45	GATE2*	
47	GATE1	
17	GATE5A	
19	GATE4A	
21	GATE3A	
23	GATE2A	
25	GATE1A	
37	OUT5*	Counter (Outputs)
35	OUT4*	
33	OUT3*	
31	OUT2*	
29	OUT1*	
27	FOUT*	Frequency Out (Output)

Connector P1 is an 86-pin double sided edge connector that provides the bus interface for the Am95/4006. Table 2-12 is a pin list for connector P1. When the MonoBoard is being used with another bus master, the BPRN* input (P1-15) to the master assigned the highest priority must be tied low. The BPRN* input to each master with the next lower priority must be connected to the BPRO* output (P1-16) of the next higher priority master. Consult the Multibus specification listed in the preface for more bus interfare information.

TABLE 2-12. SYSTEM BUS CONNECTOR P1 PIN ASSIGNMENTS

	(COMPONENT SIDE)			(CIRCUIT SIDE)		
	PIN	MNEMONIC	DESCRIPTION	PIN	MNEMONIC	DESCRIPTION
Power Supplies	1	GND	Signal GND	2	GND	Signal GND
	3	+5	+5 VDC	4	+5	+5 VDC
	5	+5	+5 VDC	6	+5	+5 VDC
	7	+12	+12 VDC	8	+12	+12 VDC
	9	-5	-5 VDC	10	-5	-5 VDC
	11	GND	Signal GND	12	GND	Signal GND
Bus Controls	13	BCLK*	Bus Clock	14	INIT*	Initialize
	15	BPRN*	Bus Priority In	16	BPRO*	Bus Priority Out
	17	BUSY*	Bus Busy	18	BREQ*	Bus Request
	19	MRDC*	Mem. Read Command	20	MWTC*	Mem. Write Command
	21	IORC*	I/O Read Command	22	IOWC*	I/O Write Command
	23	XACK*	XFER Acknowledge	24	INH1*	Inhibit 1 (RAM)
	25	AACK*	Advance Acknowledge	26	INH2*	Inhibit 2 (ROM)
	27	BHEN*	Not Used	28	ADR10*	Not Used
	29	CBRQ*	Common Bus Request	30	ADR11*	Not Used
	31	CCLK*	Constant Clock	32	ADR12*	Not Used
	33	INTA*	Interrupt Acknowledge	34	ADR13*	Not Used
Interrupts	35	IRQ6*	Interrupt Requests	36	IRQ7*	Interrupt Requests
	37	IRQ4*		38	IRQ5*	
	39	IRQ2*		40	IRQ3*	
	41	IRQ0*		42	IRQ1*	
Addresses	43	ADRE*	Address Bus	44	ADRF*	Address Bus
	45	ADRC*		46	ADRD*	
	47	ADRA*		48	ADRB*	
	49	ADR8*		50	ADR9*	
	51	ADR6*		52	ADR7*	
	53	ADR4*		54	ADR5*	
	55	ADR2*		56	ADR3*	
	57	ADR0*		58	ADR1*	
Data	59	DATE*	Not Used	60	DATF*	Not Used
	61	DATC*	Not Used	62	DATD*	Not Used
	63	DATA*	Not Used	64	DATB*	Not Used
	65	DAT8*	Not Used	66	DAT9*	Not Used
	67	DAT6*	Data Bus	68	DAT7*	Data Bus
	69	DAT4*		70	DAT5*	
	71	DAT2*		72	DAT3*	
	73	DAT0*		74	DAT1*	
Power Supplies	75	GND	Signal GND	76	GND	Signal GND
	77		Reserved	78		Reserved
	79	-12	-12 VDC	80	-12	-12 VDC
	81	+5	+5 VDC	82	+5	+5 VDC
	83	+5	+5 VDC	84	+5	+5 VDC
	85	GND	Signal GND	86	GND	Signal GND

CHAPTER 3
OPERATION AND PROGRAMMING

3-1. INTRODUCTION

This section provides operating and programming information for the Am95/4006 MonoBoard Computer (MBC) and the on-board programmable devices. The seven on-board programmable devices are:

- An Am9551 Programmable Communications Interface chip that provides serial I/O;
- Two Am9555 Programmable Peripheral Interface chips that control the 48 parallel I/O lines;
- An 8259A that provides eight fully-programmable interrupt channels for software-generated interrupts.
- An Am9513 System Timing Controller that provides five general-purpose 16-bit counters which enhance system capability with respect to counting and timing functions.
- An optional Am9511 or Am9512 Arithmetic Processing Unit that provides extended fixed and floating point arithmetic processing capabilities.
- An LS273 Latch

3-2. ADDRESS ASSIGNMENT

The CPU communicates with the programmable devices through a sequence of I/O read and I/O write commands. A summary of the I/O addresses as shipped from the factory is provided in table 3-1. Depending upon the application for which the board is to be used, users might have to alter the factory selected I/O addresses by programming their own I/O chip select PROM (U33). Bus address bits 8 through 15 correspond to the A0 through A7 PROM inputs respectively. PROM output pins 00, 01 and 02 are connected to input pins A, B, and C respectively of the three-to-eight line decoder (U34) which generates the signals used to select the I/O chip. Prom output pin 03 is used to select on-board or off-board memory (1 = on-board, 0 = off-board). It is possible to change the address of an I/O device, by changing the PROM (U33).

3-3. LATCH PROGRAMMING

A programmable latch (U42) provides dynamic control to allow memory locations to be reassigned under program control and clear and arm the interrupt control latches. As delivered, the programmable latch is accessed by address E0H. This address can be changed by reprogramming the I/O address PROM (U33). The accompanying byte (8-bits) is used to program the Boot Bit (bit 0) and the interrupt control latches (bits 3 through 6). All bits are cleared (set to logical 0) when the board is powered up or reset.

Bit 0 from the programmable latch is input as address bit 7 to the memory mapping PROM, where it forms part of the input address to the PROM. A possible use of the bit 0 latch might be to select an on-board bootstrap operation starting in the low addresses of ROM/E-PROM. Once the bootstrap has executed, the bit 0 latch could be set high to select another group of memory mapping PROM locations, and consequently overlay the memory space.

Another use of the bit 0 latch output might be to disable all on-board memory so that the bootstrap program could come from off board memory. The memory mapping PROM determines how the bit 0 latch will be used. As delivered, the memory mapping PROM (U40) is programmed to select all off-board memory from 0000H to FFFH with the boot bit high or logic 1.

Programmable latch bits 3 through 6 are used to selectively clear and arm the 4-bit timer interrupt latch (U74).

The outputs from the timer interrupt latch can be jumper-connected to the 8259A. The relationship between the programmable latch bits and the timer interrupts are as follows:

- Programmable latch bit 3 corresponds to timer interrupt latch bit 0 (TIO). TIO is always connected to the timeout signal, and is therefore the timeout interrupt signal to the interrupt jumper matrix.

TABLE 3-1. I/O PORT ADDRESSES

I/O Port Address	I/O Device	Input Function	Output Function
D8H D9H	Am9513 System Timing Controller	Data Read Status Read	Data Write Command Write
C2H C3H	8259A Interrupt Controller	Data Read Status Read	Data Write Command Write
COH C1H	Am9511 APU or Am9512 APU	Data Byte From Stack Read Status	Data Byte Onto Stack Enter Command
EOH	74LS273 Latch	Not Used	Bus Override and Boot Control
E4H E5H E6H E7H	8255A Parallel I/O Ports E4H-E6H	Read Port A Read Port B Read Port C Not Used	Write Port A Write Port B Write Port C Control Register
E8H E9H EAH EBH	8255A Parallel I/O Ports E8H-EAH	Read Port A Read Port B Read Port C	Write Port A Write Port B Write Port C Control Register
ECH EDH	Am9551 Serial I/O	Receive Data Buffer Status Register	Transmit Data Buffer Command Register

- Programmable latch bits 4 through 6 correspond to timer interrupt latch bits 1 through 3 (TI1 through TI3) respectively. These three latches can be jumper-assigned on any of the six 9513 output signals.

Once a timer interrupt latch bit is set, it remains set until it is cleared by the programmable latch. Therefore, the interrupt service routine should clear and arm the latch for the serviced interrupt. To clear a timer interrupt latch, a zero must be written to the corresponding programmable latch bit. To rearm a timer interrupt latch, a 1 must be written after being cleared. Remember, when writing to the programmable latch, every bit is affected each time address E0H is received. Therefore when only one bit in the latch is to be changed, care must be taken to send a data byte in which all bits (except the one to be changed) are in the same state as the corresponding bits in the latch.

3-4. MEMORY SELECT PROM PROGRAMMING

A memory mapping PROM located at U40 is used to map the addresses of on-board PROM and RAM memory. Four independent maps are allowed, with two of the maps under program control through latch U42 (BOOT*) and the other maps under strappable control (posts 7374). As delivered, the mapping PROM is programmed with the standard map shown in Appendix D, table D-1. When configured for use with the Am9716s, the PROM maps addresses 0 through 1FFFH to ROM and addresses 3000H through 3FFFH to on- board RAM. When configured for use with Am9732s, the PROM maps addresses 0 through 3FFFH to ROM and addresses 4000H through 4FFFH to on-board RAM. This mapping occurs only while the Boot Bit is low. When the Boot Bit is high, all on-board RAM and ROM is disabled. The Boot Bit could be used to allow a boot- strap program to execute from ROM following power-up and then map all memory off board and use the full address range. This arrangement can be particularily useful to substitute off-board RAM for on-board PROM after a system completes its initialization (boot) routines.

To use a memory configuration other than the one supported by the standard memory mapping PROM, it is necessary to replace the PROM with one customized to the specific application. An optional memory mapping PROM pattern is shown in Appendix D, table D-2. A mapping PROM programmed with the optional PROM pattern will support Am9716 and Am9708 PROMs instead of the standard Am9716 and Am9732 PROMs.

For an application other than one supported by the standard or optional configuration, it is necessary to generate a customized PROM pattern. When designing a customized PROM pattern, it is important to rememeber that three addressing schemes are being used. One address to consider is the 16-bit bus address; this is the address from the CPU that is used to access memory. Another address to consider is the one applied to the eight address lines of the PROM when the PROM is being programmed or accessed by the CPU during normal operation. This address selects a location within the PROM for PROM data. The third address is formed by the data from the mapping PROM; this address selects on-board and off-board memory.

The addressing circuitry is designed so that bus address bits 10 through 15, the Boot Bit, and a jumper option (73 to 74) combine to form the 8-bit address to the memory mapping PROM. As described previously in this chapter, the Boot Bit can be set and cleared under program control. The jumper is strictly a user option; it may or may not be used depending on the application. Due to the circuitry, 1K bus addresses (400H) access one mapping PROM location. It is the output from this one location that selects a block of 1000 memory locations (RAM or ROM chip).

The following factors need to be specified in order to determine the bit map for the U40 PROM:

- Type of PROM devices to be used 2708, 2716, or 2732
- Type of PROM device being placed in PROM sockets
- Number of maps to be used based on use of Boot Bit and Jumper Bit
- Addresses for PROM sockets
- Addresses for on-board RAM

Although there are variations to the steps that might be taken to program a PROM, the following procedure is suggested.

1. Determine how many of what type memory devices are to be used.
2. Decide where the memory devices are to be located. On-board RAM (1K x 4) can be located in pairs in U22, U26; U23, U27; U24, U28; and U25, U29. On-board ROM (1K x 8, 2K x 8, or 4K x 8) can be located in U43, U44, U45, and U46. The organization of off-board memory depends on the memory board being used.
3. Decide what range of memory (bus) addresses are to be used to access on-board RAM, on-board ROM, and off-board memory.
4. Now refer to tables 3-2 and 3-3. The Processor Address column of table 3-2 list bus addresses in 1K (400H) blocks. Only the starting address of each block is listed; all address up to, and including, one less than the next starting address are implied to be in a block. Any given block will access the same mapping PROM location, and will consequently access the same memory chip.
5. As shown on table 3-2, a decision must now be made concerning the use of the Boot Bit and PROM jumper (73-74).
6. Go down the Processor Address Column and find the starting memory address for the desired 1K block. Move accross the table to the right to the column for the Boot Bit and jumper combination being used. The number located at this intersection is the hex address to the memory mapping PROM.
7. Having determined the mapping PROM address to use for the memory address range selected, the next thing to do is determine what data to store in the mapping PROM to select the desired memory chip. Refer to table 3-3. In the right hand column, find the component location identifier (U number) for the location of the component. Note that one mapping PROM location selects only 1K bytes. Therefore, when 2K byte or 4K byte devices are used, additional mapping PROM locations have to be programmed to fully use the memory chip. The left hand column shows the hex data pattern to store in the mapping PROM.
8. Repeat this procedure until all memory addresses have been assigned to the available memory. Any memory PROM locations programmed with a hex F will select off board memory.

TABLE 3-2. MEMORY ADDRESS TO MAPPING PROM ADDRESS RELATIONSHIP

PROCESSOR ADDRESS	BOOT=0 JUMPER=0 (Jumper In)	BOOT=1 JUMPER=0 (Jumper In)	BOOT=0 JUMPER=1 (Jumper Out)	BOOT=1 JUMPER=1 (Jumper Out)
0	00	80	40	C0
400	01	81	41	C1
800	02	82	42	C2
C00	03	83	43	C3
1000	04	84	44	C4
1400	05	85	45	C5
1800	06	86	46	C6
1C00	07	87	47	C7
2000	08	88	48	C8
2400	09	89	49	C9
2800	0A	8A	4A	CA
2C00	0B	8B	4B	CB
3000	0C	8C	4C	CC
3400	0D	8D	4D	CD
3800	0E	8E	4E	CE
3C00	0F	8F	4F	CF
4000	10	90	50	D0
4400	11	91	51	D1
4800	12	92	52	D2
4C00	13	93	53	D3
5000	14	94	54	D4
5400	15	95	55	D5
5800	16	96	56	D6
5C00	17	97	57	D7
6000	18	98	58	D8
6400	19	99	59	D9
6800	1A	9A	5A	DA
6C00	1B	9B	5B	DB
7000	1C	9C	5C	DC
7400	1D	9D	5D	DD
7800	1E	9E	5E	DE
7C00	1F	9F	5F	DF

TABLE 1 PROM addresses are associated with the processor addresses listed in the left column.

The following example illustrates how to program the mapping PROM to use the board with four consecutive 1K byte on-board ROMs starting at address 0H and 4K bytes of on-board RAM starting at address 4000H. The Boot Bit and jumper will both be zero. The hex data pattern programmed into the map-mapping PROM is as follows:

PROM ADDRESS	DATA
00H	8
01H	9
02H	A
03H	B
10H	4
11H	5
12H	6
13H	7

TABLE 3-2. MEMORY ADDRESS TO MAPPING PROM ADDRESS RELATIONSHIP (Cont.)

PROCESSOR ADDRESS	BOOT=0 JUMPER=0 (Jumper In)	BOOT=1 JUMPER=0 (Jumper In)	BOOT=0 JUMPER=1 (Jumper Out)	BOOT=1 JUMPER=1 (Jumper Out)
8000	20	A0	60	E0
8400	21	A1	61	E1
8800	22	A2	62	E2
8C00	23	A3	63	E3
9000	24	A4	64	E4
9400	25	A5	65	E5
9800	26	A6	66	E6
9C00	27	A7	67	E7
A000	28	A8	68	E8
A400	29	A9	69	E9
A800	2A	AA	6A	EA
AC00	2B	AB	6B	EB
B000	2C	AC	6C	EC
B400	2D	AD	6D	ED
B800	2E	AE	6E	EE
BC00	2F	AF	6F	EF
C000	30	B0	70	F0
C400	31	B1	71	F1
C800	32	B2	72	F2
CC00	33	B3	73	F3
D000	34	B4	74	F4
D400	35	B5	75	F5
D800	36	B6	76	F6
DC00	37	B7	77	F7
E000	38	B8	78	F8
E400	39	B9	79	F9
E800	3A	BA	7A	FA
EC00	3B	BB	7B	FB
F000	3C	BC	7C	FC
F400	3D	BD	7D	FD
F800	3E	BE	7E	FE
FC00	3F	BF	7F	FF

TABLE 1 PROM addresses are associated with the processor addresses listed in the left column.

All other PROM addresse should be programmed with a hex F to select board off-board memory.

Another example shows the mapping PROM pattern to address three 2K byte ROMs starting at address 0H and 1K byte of on-board RAM starting at 3C00H. The Boot Bit is set to ONE and the jumper is ZERO. The hex data pattern programmed into the PROM is as follows:

PROM ADDRESS	DATA
80H	8
81H	8
82H	9
83H	9

PROM ADDRESS	DATA
84H	A
85H	A
8FH	4,5,6,or 7

The data in location 8FH depends on which chip locations the ROM pair is placed in. All other PROM locations should be programmed with a hex F to select off-board memory.

3-5. SERIAL I/O INTERFACE PROGRAMMING

An Am9551 Programmable Communications Interface presents a parallel 8-bit interface to the CPU via the data bus and an RS232C interface to an external device via connector P5. Programmable operating modes and format options allow the Am9551 to service a wide range of communications, disciplines and applications. Operating modes are determined by a mode instruction word and a command instruction word.

3-6. Am9551 INITIALIZATION

The Am9551 device is initialized as follows:

a) Write three nulls to address EDH (command register).

b) Write a 40 hex to I/O address EDH.

c) Write a Mode select word to I/O address EDH. See figure 3-1 or figure 3-2.

d) If synchronous mode has been programmed in step c, then write one or two sync characters to I/O address EDH.

e) Write a Command Instruction word to I/O address EDH. See figure 3-3.

f) Read I/O address ECH (receive register). Power on or a system reset may place an unknown character in the Am9551 Rx register.

TABLE 3-3. PROM DATA FOR MEMORY SELECTION

PROM DATA	WHAT WILL BE SELECTED
0-3	Illegal
4	1K byte RAM pair on-board (U22 and U26)
5	1K byte RAM pair on-board (U23 and U27)
6	1K byte RAM pair on-board (U24 and U28)
7	1K byte RAM pair on-board (U25 and U29)
8	1K byte ROM on-board (U43)
9	1K byte ROM on-board (U44)
A	1K byte ROM on-board (U45)
B	1K byte ROM on-board (U46)
C-E	Illegal
*F	1K byte memory off-board

*Note: Every byte in the PROM not programmed for on-board memory must be filled with an F to select off-board memory.

NOTE

After initialization, always check the status of the TxRDY bit prior to writing data or a new command word to the Am9551. The TxRDY bit must be true to prevent overwriting and subsequent loss of commands or data. The TxRDY is inactive until initialization has been completed.

Once initialized, it is not necessary for a command instruction to precede all data transactions--only those transmissions that require a change in the command instruction.

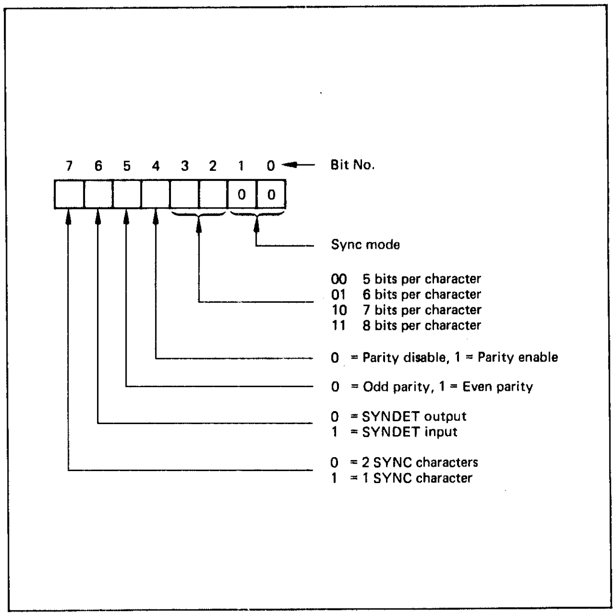


Figure 3-1. Am9551 Synchronous Mode Control Code

3-7. Am9551 MODE INSTRUCTION WORD FORMAT

The mode instruction word defines the general characteristic of the Am9551.

Once the mode instruction has been written, sync characters or command instructions may be inserted. The mode instruction word defines the following:

a) For synchronous mode:
   Character length
   Parity enable/disable
   Even/odd parity
   Character synchronization
   Single or double character sync

b) For asynchronous mode:
   Baud rate multiplier
   Character length
   Parity enable/disable
   Even/odd parity
   Number of stop bits

The mode instruction word formats for synchronous and asynchronous modes are shown in figures 3-1 and 3-2 respectively.

3-8. Am9551 SYNC CHARACTERS

In the synchronous mode, one or two sync characters must be written to address EDH. The format of the sync characters is at the option of the programmer.

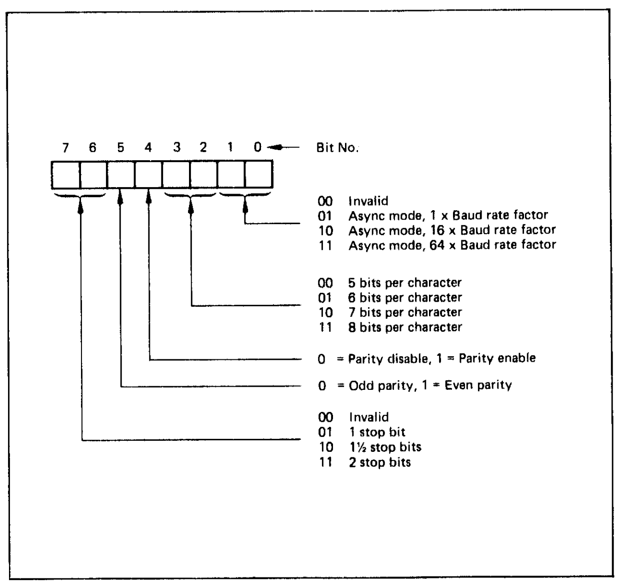


Figure 3-2. Am9551 Asynchronous Mode Control Code

3-9. COMMAND INSTRUCTION WORD FORMAT

The command instruction word must follow the mode and/or sync words. Once the command word is written, data can be transmitted or received by the Am9551. The format of the command word is shown in figure 3-3.

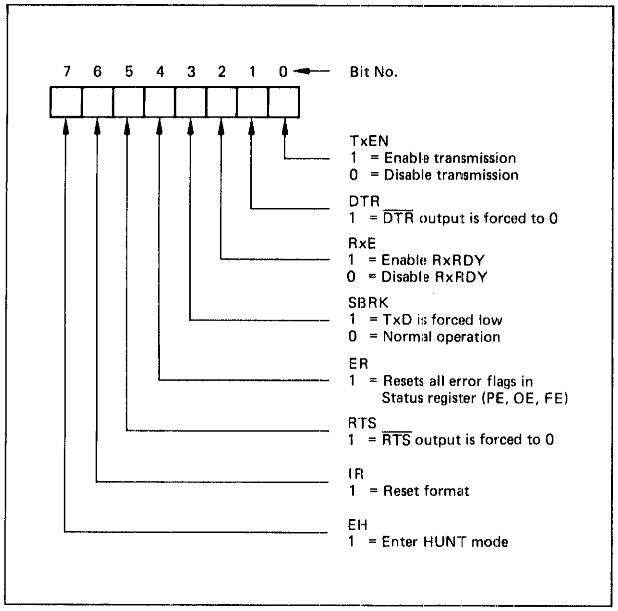


Figure 3-3. Am9551 Command Instruction Word Format

3-10. Am9551 STATUS

The CPU can determine the status of the Am9551 any time by issuing an I/O input to address EDH. The format of the status byte is shown in figure 3-4.

The definition of the status bits is as follows:

TxRDY Transmitter Ready indicates the Am9551 is ready to accept a data character or command.

RxRDY Receiver Ready indicates the Am9551 has received a character on its serial input and is ready to transfer it to the CPU.

TxE Transmitter Empty signals the processor that the transmit register is empty.

PE Parity Error indicates the character stored in the receive character buffer was received with an incorrect number of binary 1 bits.

OE Overrun flag is set when a byte stored in the receiver character register is overwritten with a new byte before being transferred to the processor.

FE Framing Error indicates the asynchronous mode byte stored in the receiver character buffer was received with incorrect character bit format.

SYNDET When Sync Detect is set for internal sync detect, this bit indicates character sync has been achieved and the Am9551 is ready for data.

DSR Data Set Ready is set by the external Data Set Ready Signal to indicate the communications data set is ready.

3-11. PARALLEL I/O INTERFACE PROGRAMMING

Two 8255A Programmable Peripheral Interface chips provide 48 parallel signal lines for the transfer and control of data to and from peripheral devices.

Each chip provides three 8-bit ports (A, B, and C). Each port can be configured as either input or output, and port C on each chip is used as control lines for ports A and B in some modes.

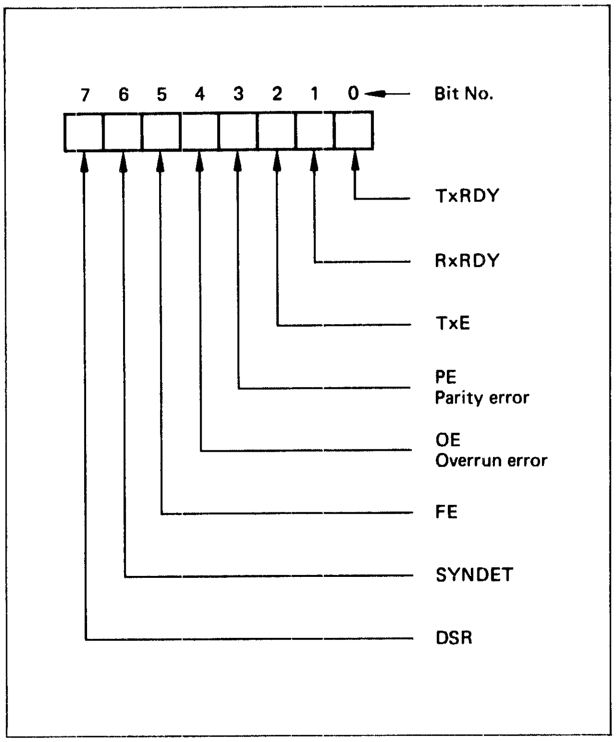


Figure 3-4. Am9551 Status Word Format

The operating modes of the ports are controlled by outputting either an operation control word or a bit set/reset control word. Table 3-4 is a complete configuration guide for the Am8255As.

3-12. 8255A ADDRESSING

Each chip uses four consecutive addresses (E4-E7H and E8-EBH) for control, data transfer, and status read. See table 3-1 for the port addresses and their function.

3-13. 8255A INITIALIZATION

The 8255A chips are initialized by writing an operation control word to address E7H and EBH to define the mode and by writing a bit set/reset control word for Port C control.

3-14. 8255A OPERATION CONTROL WORD FORMAT

The operation control word (bits 5 and 6) defines three basic modes of operation.

Mode 0: Each group of 12 I/O pins may be programmed in sets of 4 and 8 to be input or output.

Mode 1: Each group can be programmed to have 8 lines of input or output. The remaining pins are used for handshaking and interrupt control signals.

Mode 2: Is a bidirectional bus mode which uses 8 lines for a bidirectional bus, and 5 lines, borrowing one from the other group, for handshaking.

The modes for port A and B can be separately defined, while port C is divided into two 4-bit ports as required by the port A and port B definitions. Table 3-4 provides a summary of all mode definitions and port restrictions. The mode control word format is shown in figure 3-5.

3-15. 8255A BIT SET/RESET CONTROL WORD

When operating in mode 1 or 2, the bits of port C can be set or reset using the bit set/reset control word. The functions of some port C bits are defined by port A and B operations in modes 1 and 2. Refer to table 3-4 for port C bit definitions in modes 1 and 2. Figure 3-6 shows the bit set/reset control word format.

3-16. 8255A PORT C STATUS READ

The status of port C can be read at any time by an I/O read to address E6H

TABLE 3-4. PARALLEL I/O PORT CONFIGURATION SUMMARY

PORT AND MODE	CONTROL WORD	CONNECTOR POLARITY	DRIVER/TERMINATOR NETWORK	JUMPER ACTION			PORT RESTRICTIONS
				DELETE	ADD	EFFECT	
E4H 0 Input	1001XXXX	Negative True	8226s at U1 and U2.	1-3	1-2	Enable input at U1 and U2	Port E5H: None Port E6H: None unless port E5H is in mode 1
E4H 0 Output (latched)	1000XXXX	Negative True	8226s at U1 and U2.	1-2	1-3	Enable input at Ů1 and U2	Port E5H: None Port E6H: None unless port E5H is in mode 1
E4H -2Input (strobed)	1011XXXX	Negative True	8226s at U1 and U2. Termination Network at U3.	1-3	1-2	Enable input at U1 and U2	Port E5H: None Port E6H: Performs the following dedicated functions: Bits 0, 1, 2: None unless port E5H is in mode 1 Bit 3: INTR (Interrupt Request) output for port E4H Bit 4: STB* (Strobe) input for port E4H Bit 5: IBF (Input Buffer Full) output for port E4H Bits 6 and 7: Can be used for input or output Both have same direction
E4H 1 Output (latched)	1010XXXX	Negative True	8226s at U1 and U2. Termination Network at U4.	1-2	1-3	Enable output at 8226s	Port E5H: None Port E6H: Performs the following dedicated functions: Bits 0, 1, 2: None unless port E5H is in mode 1 Bit 3: INTR (Interrupt Request) output for port E4 Bits 4 and 5: Can be used for input or output; both have same direction Bit 6: ACK* (Acknowledge) input for port E4H Bit 7: OBF* (Output Buffer Full) output for port E4H
E4H 2 Bidirectional	11XXXXXX	Negative True	8226s at U1 and U2	1-2 1-3	1-4	Allows ACK_A* output of port	Port E5H: None Port E6H: Performs the following dedicated functions:

TABLE 3-4. PARALLEL I/O PORT CONFIGURATION SUMMARY (Cont.)

PORT AND MODE	CONTROL WORD	CONNECTOR POLARITY	DRIVER/TERMINATOR NETWORK	JUMPER ACTION			PORT RESTRICTIONS
				DELETE	ADD	EFFECT	
E4H 2 Bidirectional (continue)						E6H to control 8226 direction of data flow	Bit 0: Cannot be used Bits 1 and 2: Can be used as input or output if port E5H is in mode 0 Bit 3: INTR (Interrupt Request) output for port E4H Bit 4: STR* (Strobe) input for port E4H Bit 5: IBF (Input Buffer Full) output for port E4H Bit 6: ACK* (Acknowledge) input for port E4H. Data Flow direction control for 8226 via jumper 57-32.† Bit 7: OBF* (Output Buffer Full) output for port E4H.
E5H 0 Input	1XXXX01X	Positive True	Termination Networks at U5 and U6	None	None		Port E4H: None Port E6H: None unless port E4H is in mode 1 or 2
E5H 0 Output (latched)	1XXXX00X	Negative True	Driver Networks at U5 and U6	None	None		Port E4H: None Port E6H: None unless port E4H is in mode 1 or 2
E5H 1 Input (strobed)	1XXXX11X	Positive True	Termination Networks at U4, U5, and U6	Opt.	Opt.		Port E4H: None Port E6H: Performs the following dedicated functions: Bit 0: INTR (Interrupt Request) output for port E5H Bit 1: IBF: (Input Buffer Full) output for port E5H Bit 2: STB* (Strobe) input for port E5H Bit 3: Can be used as input or output if port E4H is in mode 0 Bits 4 to 7: Can be used as input or output if port E4H is in mode 0 or in some combinations where port E4H is in mode 1. These bits are always dedicated when port E4H is in mode 2.

TABLE 3-4. PARALLEL I/O PORT CONFIGURATION SUMMARY (Cont.)

PORT AND MODE	CONTROL WORD	CONNECTOR POLARITY	DRIVER/TERMINATOR NETWORK	JUMPER ACTION			PORT RESTRICTIONS
				DELETE	ADD	EFFECT	
E5H 1 Output (latched)	1XXXX10X	Negative True	Driver Networks at U4, U5, and U6	Opt.	Opt.		Port E4H: None Port E6H: Performs the following dedicated functions: Bit 0: INTR (Interrupt Request) output for port E5H Bit 1: OBF* (Output Buffer Full) output for port E5H Bit 2: ACK* (Acknowledge input for port E5H. Bit 3: Can be used as input or output if port E8H is in mode 0 Bits 4 to 7: Can be used as input or output if port E4H is in mode 0 in some combinations where port 1 is in mode 1. These bits are always dedicated when port E4H is in mode 2.
E6H High Order Bits 0 Input	100X10XX	Positive True	Termination Network at U3	Opt.	Opt.		Port E4H: Must be in mode 0 for all four bits to be available. Port E5H: Must be in mode 0 for all four bits to be available.
E6H Low Order Bits 0 Input	100XX0X1	Positive True	Termination Network at U4	Opt.	Opt.		Port E4H: Must be in mode 0 for all four bits to be available. Port E5H: Must be in mode 0 for all four bits to be available.
E6H High Order Bits 0 Output (latched)	100X00XX	Negative True	Driver Network at U3	Opt.	Opt.		Port E4H: Must be in mode 0 for all four bits to be available. Port E5H: Must be in mode for all four bits to be available.
E6H Low Order Bits 0 Output (latched)	100XX0X0	Negative True	Driver Network at U3	Opt.	Opt.		Port E4H: Must be in mode 0 for all four bits to be available.

TABLE 3-4. PARALLEL I/O PORT CONFIGURATION SUMMARY (Cont.)

PORT AND MODE	CONTROL WORD	CONNECTOR POLARITY	DRIVER/TERMINATOR NETWORK	JUMPER ACTION			PORT RESTRICTIONS
				DELETE	ADD	EFFECT	
E6H Low Order Bits 0 Output (latched) (continued)							Port E5H: Must be in mode 0 for all four bits to be available.
E8H 0 Input	1001XXXX	Negative True	8226s at U7 and U8	25-34	24-25	Enable inputs at U7 and U8	Port EAH: None unless port E9H is in mode 1
E8H 0 Output (latched)	1000XXXX	Negative True	8226s at U7 and U8	24-25	25-34	Enable outputs at U7 and U8	Port E9H: None Port EAH: None unless port E9H is in mode 1
E8H 1 Input (strobed)	1011XXXX	Negative True	8226s at U7 and U8 Termination Network at U9	25-34	24-25	Enable inputs at U7 and U8	Port E9H: None Port EAH: Performs the following dedicated functions: Bits 0, 1, 2: None unless port E9 is in mode 1. Bit 3: INTR (Interrupt Request) output for port E8H. Bit 4: STR* (Strobe) input for port E8H. Bit 5: IBF (Input Buffer Full) output for port E8H Bits 6 and 7: Can be used for input or output. Both have same direction.
E8H 1 Output (latched)	1010XXXX	Negative True	8226s at U7 and U8. Termination Network at U10	24-25	25-34	Enable outputs at U7 and U8	Port E9H: None Port EAH: Performs the following dedicated functions: Bits 0, 1, 2: None unless port E9H is in mode 1. Bit 3: INTR (Interrupt Request) output for port E84. Bits 4 and 5: Can be used for input or output; both have same direction. Bit 6: ACK* (Acknowledge) input for E8H. Bit 7: OBF* (Output Buffer Full) output for port E4H.

TABLE 3-4. PARALLEL I/O PORT CONFIGURATION SUMMARY (Cont.)

PORT AND MODE	CONTROL WORD	CONNECTOR POLARITY	DRIVER/TERMINATOR NETWORK	DELETE	JUMPER ACTION		PORT RESTRICTIONS
					ADD	EFFECT	
E8H 2 Bidirectional	11XXXXXX	Negative True	8226s at U7 and U8 Termination Network at U9 Driver Network at U10	24-25 25-34	25-37 25-35	Allows ACK_A* output of port EAH to control 8226 direction of data flow	Port E9H: None Port EAH: Performs the following dedicated functions: Bit 0: Cannot be used Bits 1 and 2: Can be used as input or if port E9H is in mode 0 Bit 3: INTR (Interrupt Request) output for port E8H Bit 4: STB* (Strobe) input for port E8H Bit 5: IBF (Input Buffer Full) output for port E8H
E9H 0 Output	1XXXX01X	Positive True	Termination Network at U11 and U12	None	None		Port E8H: None Port EAH: None unless port E8H is in mode 1 or 2
E9H 0 Output (latched)	1XXXX00X	Negative True	Driver Network at U11 and U12	None	None		Port E8H: None Port EAH: None unless port E8H is in mode 1 or 2
E9H 1 Input (strobed)	1XXXX11X	Positive True	Termination Network at U10, U11, and U12	Opt.	Opt.		Port E8H: None Port EAH: Performs the following dedicated functions: Bit 0: INTR (Interrupt Request) output for port E9H Bit 1: IBF (Input Buffer Full) output for port E9H Bit 2: STB* (Strobe) input for port E9H Bit 3: Can be used as input or output if port E8H is in mode 0 Bits 4 to 7: Can be used as input or output if port E8H is in mode 0 or in some combinations where port E8H is in mode 1. These bits are always dedicated when port E8H is in mode 2.

TABLE 3-4. PARALLEL I/O PORT CONFIGURATION SUMMARY (Cont.)

PORT AND MODE	CONTROL WORD	CONNECTOR POLARITY	DRIVER/TERMINATOR NETWORK	JUMPER ACTION			PORT RESTRICTIONS
				DELETE	ADD	EFFECT	
E9H 1 Output (latched)	1XXXX10X	Negative True	Driver Network at U10, U11, and U12	Opt.	Opt.		Port E8H: None Port EAH: Performs the following dedicated functions: Bit 0: INTR (Interrupt Request) output for port E9H Bit 1: OBF* (Output Buffer Full) output for port E9H Bit 2: ACK* (Acknowledge) input for port E9H Bit 3: Can be used as input or output if port E8H is in mode 0 or some combinations of mode 1. These bits are always reserved when port E8H is in mode 2.
EAH High Order Bits 0 Input	100XX0X1	Positive True	Termination Network at U10	Opt.	Opt.		Port E8H: Must be in mode 0 for all four bits to be available. Port E9H: Must be in mode 0 for all four bits to be available.
EAH High Order Bits 0 Output (latched)	100X00XX	Negative True	Driver Network at U9	Opt.	Opt.		Port E8H: Must be in mode 0 for all four bits to be available. Port E9H: Must be in mode 0 for all four bits to be available
EAH Low Order Bits 0 Output (latched)	100XX0X0	Negative True	Driver Network at U10	Opt.	Opt.		Port E8H: Must be in mode 0 for all four bits to be available Port E9H: Must be in mode 0 for all foru bits to be available

or EAH. The definition of port C bits are determined by the operating modes of port A and B. Refer to table 3-4 for port C bit definitions.

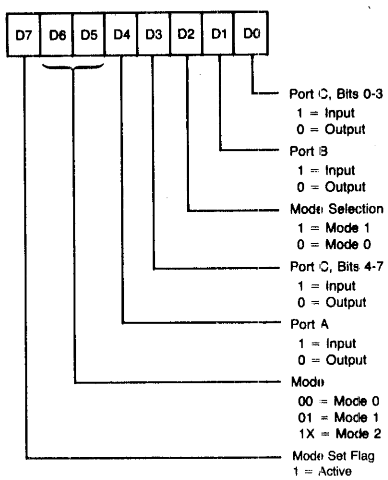


Figure 3-5. 8255A Operation Control Word Format

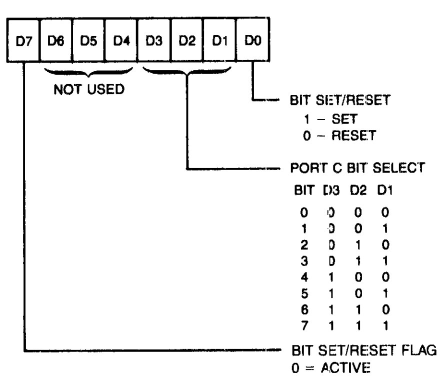


Figure 3-6. Bit Set/Reset Control Word Format

3-17. 8259A FUNCTIONAL DESCRIPTION

An 8259A Programmable Interrupt Controller (PIC) chip accepts 8 out of 16 possible interrupt requests originating in the Arithmetic Processor, Floating Point Processor, Serial I/O ports, and System Timing Controller. The 8259A processes these requests, selects the highest priority interrupt request, interrupts the CPU, and gives the CPU the address information for the interrupt processing subroutine in memory.

Each PIC is programmed by the system's software as an I/O device. The priority assignments and algorithms can be changed or reconfigured dynamically at any time during the running of the main program. Since the response bytes are programmable, any instruction or vectoring protocol appropriate for the CPU can be used.

3-18. 8259A INTERRUPT REQUEST REGISTER (IRR)

The IRR stores the interrupt level(s) requesting service. A bit is set when the corresponding interrupt request input becomes active and is automatically cleared when it is acknowledged. All bits are cleared by a reset function.

3-19. 8259A IN-SERVICE REGISTER (ISR)

The ISR is used to keep track of a pending interrupt that has been acknowledged and to mask lower priority interrupts. When a bit is set in the ISR, the corresponding IRR bit is cleared. All ISR bits are cleared by a reset function.

3-20. 8259A PRIORITY RESOLVER

The Priority Resolver determines the highest priority unmasked pending request and strobes the corresponding bit in the ISR when the request is acknowledged by the CPU.

3-21. 8259A INTERRUPT MASK REGISTER (IMR)

The IMR enables or disables the individual interrupt inputs and all eight can be enabled or disabled in parallel. A reset function sets all eight mask bits, disabling all interupt requests. A mask bit which is set does not disable the IRR, and an interrupt request which arrives while a corresponding mask bit is set will cause an interrupt later when that mask bit is cleared.

3-22. 8259A ADDRESSING

The 8259A uses two consecutive I/O addresses for writing commands and vector data and reading status. The addresses and their functions are shown in table 3-1.

3-23. 8259A PROGRAMMING

The Am95/4006 has only one 8259A Programmable Interrupt Controller (PIC) to control interrupt processing. It must be programmed as a master in the buffered mode. If off-board interrupt devices drive the on-board PIC, the Am95/4006 must be jumpered for bus-vectored interrupts. The Am95/4006 is shipped configured for non bus-vectored interrupts. For bus-vectored interrupts, connect jumpers as follows:

1. Remove jumpers 95 to 96 and 76 to 77.

2. Connect jumpers 95 to 146, 75 to 76, and 146 to 147.

3-24. 8259A INITIALIZATION

Initializing an 8259A is accomplished using two types of command words: Initialization Command Words (ICWs) shown in figure 3-8 and Operational Command Words (OCWs) shown in figure 3-9. The ICWs are issued by the CPU in a sequential format as illustrated with figure 3-7 and used to set-up the PIC in an initial state. The OCWs are issued as needed to change and control PIC operation. Both ICWs and OCWs are issued via the PIC data bus and timed with the write strobe.

3-25. ICW1 AND ICW2

Initialization Command Words 1 and 2 are the minimum programming needed for any type of PIC operation. When a command is issued with A0=0 and D4=1 (see figure 3-7), the PIC interprets it as ICW1. Initialization Command Word 1 starts the intialization sequence (figure 3-7) and during ICW1 time the following automatically occurs:

- Sequencer logic is set to accept the remaining ICWs as designated in ICW1.

- The ISR (In-Service Register) and IMR (Interrupt Mask Register) are both cleared.

- The special mask mode is reset.

- The rotate in automatic EOI mode flip-flop is cleared.

- The IRR (Interrupt Request Register) is selected for the read register command.

- If the IC4 bit equals 0 in ICW1, all functions in ICW4 are cleared.

- The fully nested mode is entered with an initial priority assignment of IR0 highest through IR7 lowest.

- The edge sense latch of each IR priority cell is cleared, thus requiring a low to high transition to generate an interrupt (edge triggered mode effected only).

Once started, the initialization sequence must be completed before the PIC can process interrupts. This applies to each PIC in a master/slave system. Initialization command word 2 supplies the most-significant bits of the address, which is used as the starting memory location of the service routine. When a PIC is operated as the case using the Am95/4006, A5 through A15 must be programmed with the desired address when ADI is set for the 4-byte interval. If ADI is set for the 8-byte interval A6 through A15 must be programmed with the desired address. Description of ICW1 and ICW2 bits are as follows:

- ICW1 Bit 0 — IC4 - Always a 1.

- ICW1 Bit 1 — The SNGL bit is used to designate whether or not the 8259A is to be used alone or in the cascade mode. When the cascade mode is desired, SNGL must equal 0. In doing this, the 8259A will accept ICW3 for further cascade mode programming. When SNGL=0, ICW4 is needed for buffered mode and M/S definition. When the 8259A is to be used as the single 8259A within a system, the SNGL bit must equal 1; ICW3 won't be accepted.

- ICW1 Bit 2 — The ADI bit is used to specify the address interval for the 8080 mode. When a 4-byte address interval is to be used, ADI must equal 1. For an 8-byte address interval, ADI must equal 0.

- ICW1 BIT3 — ICW1 Bit 3 should be zero for edge triggering.

- A5-A15: — The A5-A15 bits are used to select the interrupt vector address. There are two programming formats that can be used to do this. Which one implemented depends upon the selected address interval (ADI). When ADI is set for the 4-byte interval, the 8259A automatically insert A0-A4 (A0, A1=0 and A2, A3, A4=IR0-7).
Thus A5-A15 must be user-selected by programming the A5-A15 bits with the desired address. When ADI is set for the 8 byte interval, A0-A5 are automatically inserted (A0, A1, A2=0 and A3, A4, A5=IR0-7). This leaves A6-A15 to be selected by programming the A6-A15 bits with the desired address. The state of bit 5 is ignored in the latter format.

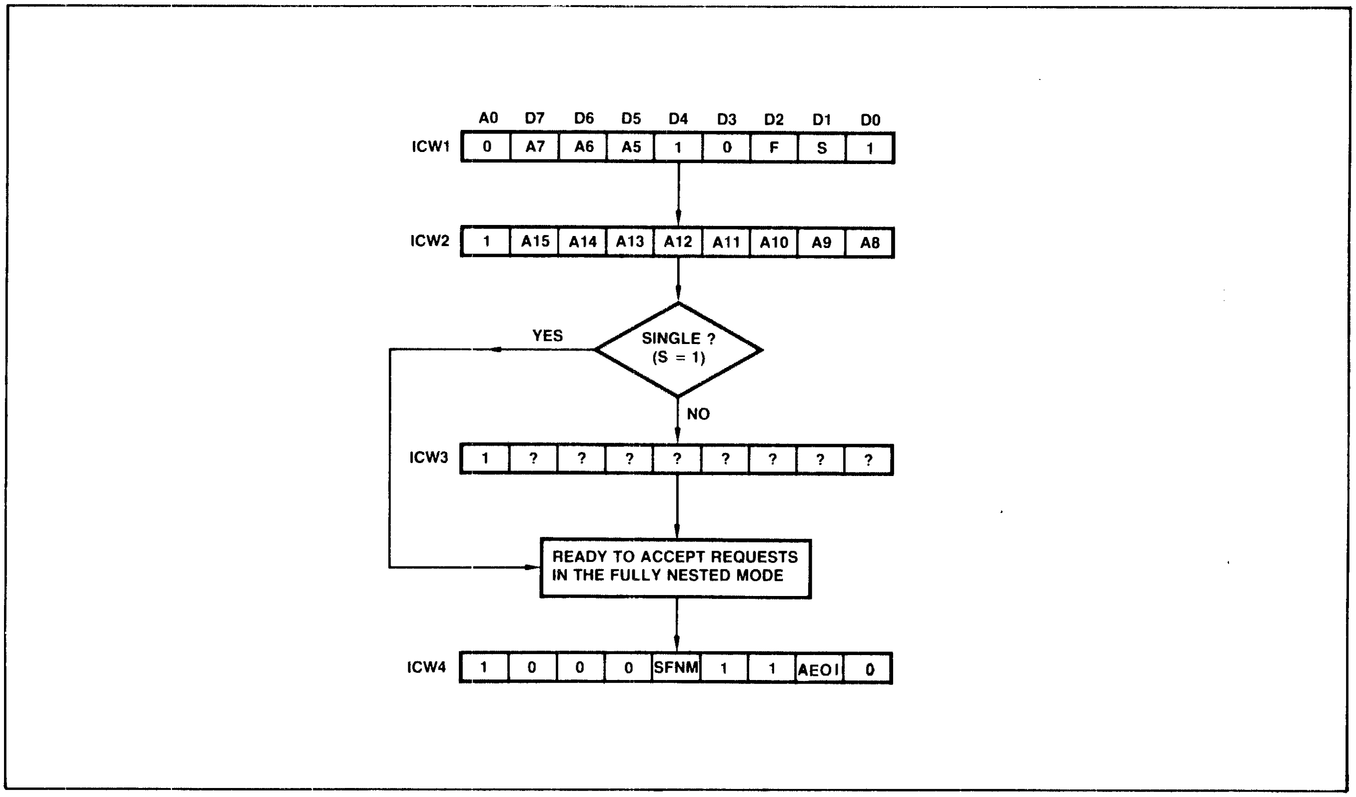


Figure 3-7. 8259A Initialization Sequence

3-26. ICW3 and ICW4

Initialization Command Word 3 is programmed when there is more than one PIC in the system (ICW1, SNGL=0) and cascading is used. This word controls the master/slave relationship to ensure the correct PIC places the service routine address onto the system bus. Initialization Command Word 4 offers choice of various modes of operation. Bit definition of ICW3 and ICW4 are as follows:

- ICW3 D0-D7 — When the 8259A is a master in the buffered mode when M/S=1 in ICW4), ICW3 bit definition is D0-7, corresponding to slave 0-7. These bits are used to establish which IR Inputs have slaves connected to them. A 1 designates a slave, a 0 no slave. For example, if a slave was connected to IR3, the D3 bit should be set to a 1. (D0) should be last choice for slave designation.

- ICW4 Bit 0 — Always a zero.

- ICW4 Bit 1 (AEOI) — The AEOI bit is used to select the automatic end of interrupt mode. When AEOI=1, the automatic end of interrupt mode is selected. When AEOI=0, it isn't selected; thus an EOI command must be used during a service routine.

- ICW4 Bit 2 — Always a 1.

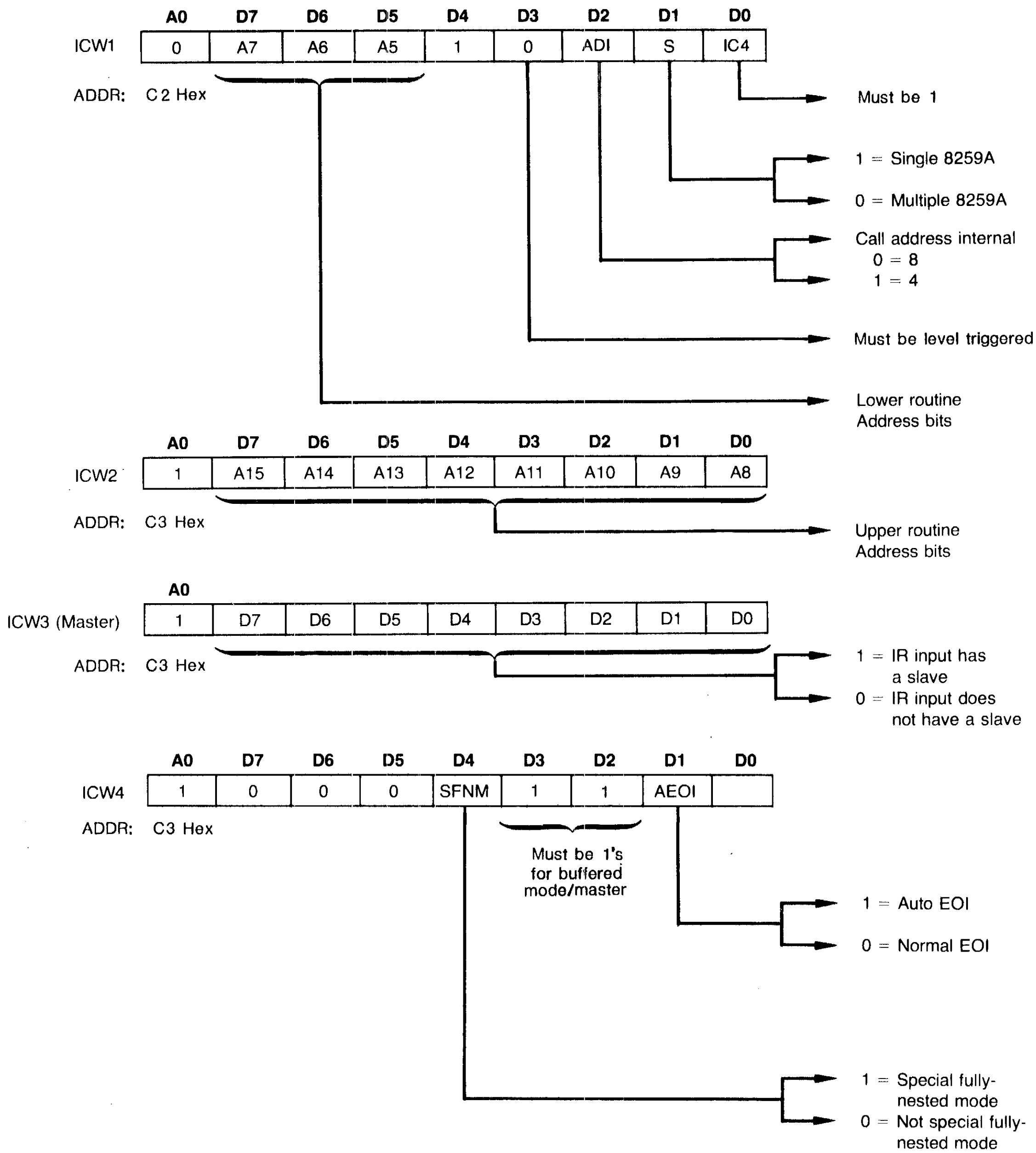


Figure 3-8. Initialization Command Word (ICW) Format

OCW1

A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0

M_7–M_0: Interrupt mask bit
1 = Masked (inhibited)
0 = Enabled

OCW2

A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	R	SEOI	EOI	0	0	L_2	L_1	L_0

IR LEVEL TO BE ACTED UPON

	0	1	2	3	4	5	6	7
L_0	0	1	0	1	0	1	0	1
L_1	0	0	1	1	0	0	1	1
L_2	0	0	0	0	1	1	1	1

R	SEOI	EOI	
0	0	0	No action
0	0	1	Non-specific end of interrupt
0	1	0	No action
0	1	1	Specific end of interrupt. L_2, L_1, L_0 is the BCD level to be reset.
1	0	0	No action
1	0	1	Rotate priority at EOI (auto mode).
1	1	0	Rotate priority, L_2, L_1, L_0 becomes bottom priority without ending of interrupt.
1	1	1	Rotate priority at EOI (specific mode), L_2, L_1, L_0 becomes bottom priority, and its corresponding IS bit is reset.

OCW3

A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	*	ESMM	SMM	0	1	P	ERIS	RIS

ESMM	SMM	
0	0	No action
0	1	No action
1	0	Reset special mask
1	1	Set special mask

ERIS	RIS	
0	0	No action
0	1	No action
1	0	Read on next RD* pulse
1	1	Read ISR on next RD* pulse

P: Polling
1 = Polling mode
0 = Interrupt mode

D_7 (*): Don't care

Figure 3-9. Operation Command Word (OCW) Formats

- ICW4 Bit 3 — Always a 1.

- ICW4 Bit 4 (SFNM) — The SFNM bit designates selection of the special fully nested mode. Only the master should be programmed in the special fully nested mode. SFNM does not mask a slave while it is servicing an interrupt. The slave can produce another interrupt from a higher priority. The interrupt will be recognized by the CPU. When the SFNM bit is set, the special fully nested mode is selected. If the SFNM bit is cleared, the special fully nested will not be selected.

When programming changes within an ICW are to be made, the ICW sequence must be reprogrammed, not just an individual ICW.

3-27. OCW1 and OCW2

After initialization by the ICWs, the PIC operation can be controlled or changed by the use of OCWs. The OCWs need not be in any type of sequential order; they can be issued as needed within a program. OCW1 (figure 3-9) is used for PIC masking operations; it provides a direct link to the IMR (Interrupt Mask Register). The CPU can write to or read from the IMR via OCW1. OCW1 bits are defined as follows:

- OCW1 M0-M7 — The M0-M7 bits are used to control the masking of IR inputs. If an M bit is set to a 1, it will mask the corresponding IR input. These bits convey the same meaning when being read by the processor for status update.

OCW2 is used for end of interrupt, automatic rotation, and specific rotation operations. OCW2 bit definition is as follows:

- L0-L2: — The L0-L2 bits are used to designate an interrupt level (0-7) to be acted upon for the operation selected by the EOI, SEOI, and R bits of OCW2. The level designated will either be used to reset a specific ISR bit or to set a specific priority. The L0-L2 bits are enabled or disabled by the SEOI bit.

- EOI: — The EOI bit is used for all end of interrupt commands (not automatic end of interrupt mode). When set to a 1, a form of an end of interrupt command will be executed depending on the state of the SEOI and R bits. When EOI is 0, an end of interrupt command won't be executed.

- SEOI: — The SEOI bit is used to select a specific level for a given operation. When SEOI is set to a 1, the L0-L2 bits are enabled. The operation selected by the EOI and R bits will be executed on the specified interrupt level. When SEOI is 0, the L0-L2 bits are disabled.

- R: — The R bit is used to control all 8259A rotation operations. When the R bit is set to a 1, a form of priority rotation will be executed depending on the state of SEOI and EOI bits. When R is 0, rotation won't be executed.

3-28. OCW3

OCW3 is used to issue various modes and commands to the 8259A. There are two main categories of operation associated with OCW3: interrupt status and interrupt masking. Bit definition of OCW3 is as follows:

- RIS: The RIS bit is used to select the ISR or IRR for the read register command. When RIS is set to 1, ISR is selected. When RIS is 0, IRR is selected. The state of the ERIS is only honored if the ERIS bit is a 1.

- ERIS: The ERIS bit is used to execute the read register command. When ERIS is set to a 1, the read register command is issued and the state of RIS determines the register to be read. When ERIS is 0, the read register command isn't issued.

- P: The P bit selects either polling mode (1) or interrupt Mode (0).

- SMM: The SMM bit is used to set the special mask mode. When in the special mask mode, setting a mask bit in OCW1 will inhibit interrupts only for those levels represented by those bits. All others will be enabled. When the SSM bit is set, the special mask mode is selected. If SSM is cleared, the special mask mode will not be set. The SMM bit is valid only if it is enabled by the ESSM bit.

- ESMM: The ESMM bit is used to enable or disable the effect of the SMM bit. When ESMM is set to a 1, SMM is enabled. When ESMM is 0, SMM is disabled. This bit is useful to prevent interference of mode and command selections in OCW3.

3-29. 8259A STATUS READ

The input status of the following internal registers can be read by issuing an OCW and reading with RD:

- In-Service Register (ISR)
- Interrupt Mask Register (IMR)

The ISR stores a logical one in the associated bit for priority inputs that are being serviced. The ISR is updated when an EOI command is issued.

3-30. ARITHMETIC PROCESSING UNIT PROGRAMMING

The optional Am9511 Arithmetic processing Unit (APU) provides high performance fixed and floating point arithmetic as well as floating point transcendental and mathematical operations.

All transfers (e.g. operands, results, and commands) take place on the data bus. Operands are pushed onto an internal 8-level, 16-bit wide data stack, and a command is issued to perform operations on the data in the stack. Results are then retrieved from the stack, or additional commands may be entered.

Transfers to and from the APU is handled using programmed I/O.

Upon completion of each command, the APU issues an end of execution signal that can be used as an interrupt to the CPU.

3-31. AM9511 ADDRESSING

The APU uses two consecutive addresses (COH and C1H) for commands, data transfers, and status read. See table 3-1 for the port addresses and their functions.

3-32. AM9511 INITIALIZATION

The APU does not require special initialization. After a power-on or system reset operation, the status register is clear and the APU is in the idle state.

3-33. AM9511 DATA FORMATS

The Am9511 APU handles operands in both fixed-point and floating-point formats. Within the internal stack, data is logically organized as 16-bit or 32-bit operands as shown in figure 3-11. The data stack operates as a true push-down first-in/last-out (FILO) stack; the data first written in is the last data read out. Within each stack entry the least significant byte is entered first and retrieved last. Since all words are entered as 8-bit bytes, data must be entered into the stack in multiples of the number of bytes appropriate to the chosen data format.

3-34. FIXED-POINT

Fixed-point operands can be represented in either single (16-bit) or double (32-bit) precision formats. They are always represented as binary, twos complement values. The single precision and double precision fixed-point word formats are shown in figure 3-12. The sign of the operand is located in the most significant bit position. Positive values are represented by a sign bit of 0; negative values are represented by a sign bit of 1. The range of values that can be expressed by the single precision format is -32,768 to +32,767. The double precision value range is from -2,147,483,648 to +2,147,483,647.

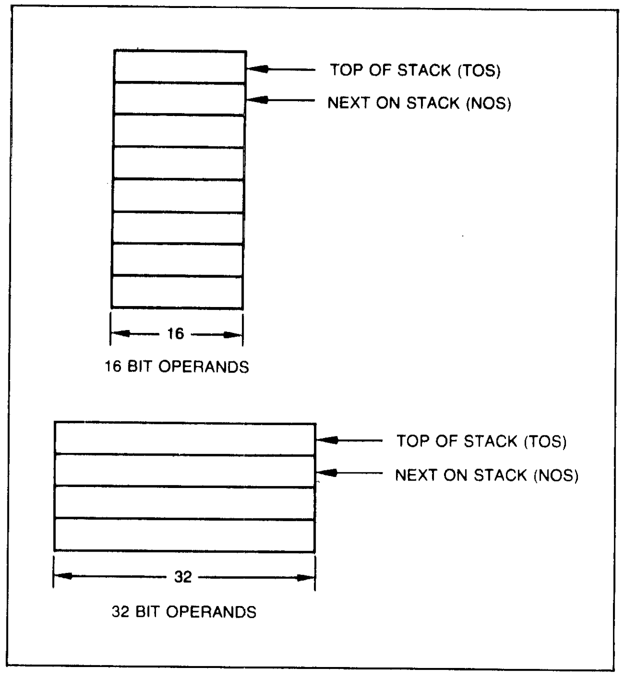


Figure 3-11. Am9511 Data Stack Configurations

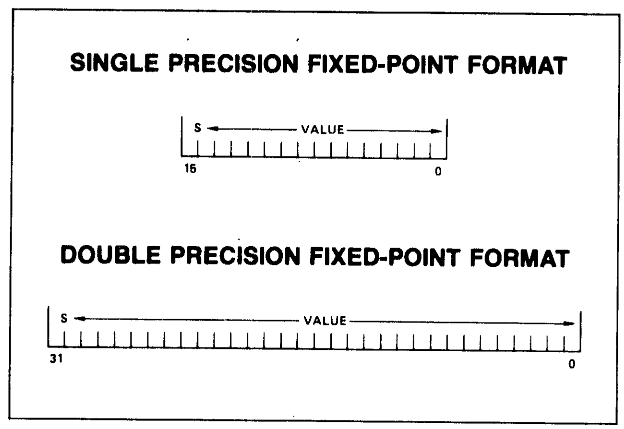


Figure 3-12. Fixed-Point Word Formats

3-35. FLOATING-POINT

The 32-bit floating-point format is shown in figure 3-13. Bit 31 indicates the sign of the mantissa. The next seven bits form the exponent, with bit 30 representing the exponent sign. Bits 0 through 23 form the mantissa value.

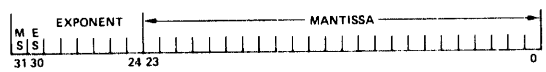


Figure 3-13. Floating-Point Word Format

The mantissa is a sign-magnitude number with an assumed binary point just to the left of the most significant mantissa bit (bit 23). The exponent is interpreted as a power of two and is expressed as a twos complement value having a range of from -64 to +63 (2^{-64} to 2^{+63}).

All floating-point values must be normalized, which makes bit 23 always equal to 1 except when representing a value of zero. The number zero is represented with binary zeros in all 32 positions.

3-36. AM9511 COMMAND DESCRIPTIONS

The following detailed description of the Am9511 commands are presented in alphabetical order by command mnemonic. In the descriptions, TOS means Top of Stack and NOS means Next on Stack. Figure 3-14 shows the command format.

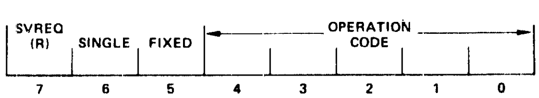


Figure 3-14. Am9511 Command Format

All derived functions except square root use Chebyshev polynomial approximating algorithms. This approach is used 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 can produce error codes in the status register as a result.

Execution times are listed in terms of clock cycles and can 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 2MHz rate translates to 22 microseconds (44 x .5us = 22us); the same 44 clock cycles translate to 18.04 microseconds when running at a 2.4576MHz rate (44 x 0.41us = 18.04 us). Variations in execution cycles reflect the data dependency of the algorithms. The 2MHz board uses a 2MHz clock, and a 2.4576MHz clock for the 4MHz board.

In some operations, exponent overflow or underflow is 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.

3-37. AM9511 STATUS READ

The APU status register is read by executing an I/O read to port C1H. When the status busy bit (bit 7) is high, the APU is processing a previously entered command and the balance of the status register is not valid. The definition of the status bits is given in figure 3-15.

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

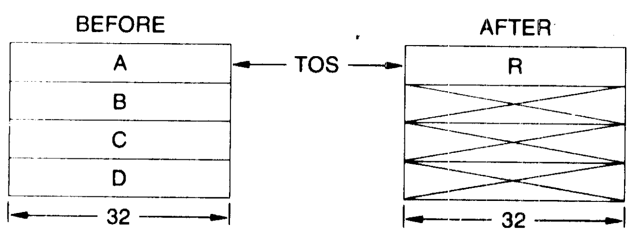
Execution Time: 6304 to 8284 clock cycles

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 2.0×10^{-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

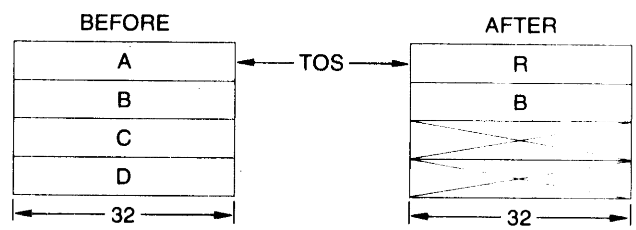
Execution Time: 4992 to 6536 clock cycles

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 3.0×10^{-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

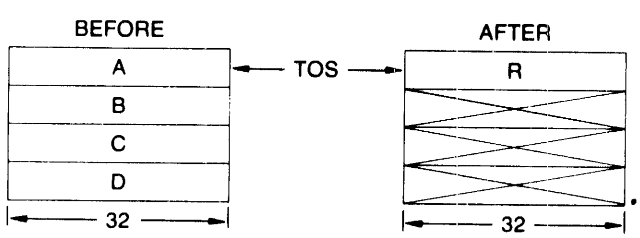
Execution Time: 6230 to 7938 clock cycles

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 4.0×10^{-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

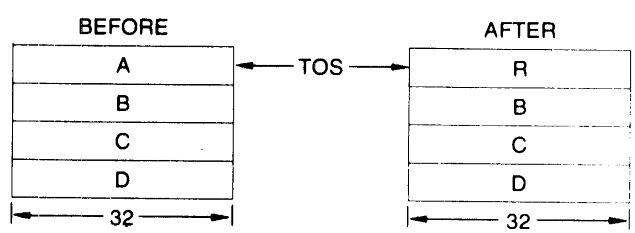
Hex Coding: B4 with sr = 1
34 with sr = 0

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

Binary Coding:

7	6	5	4	3	2	1	0
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

BEFORE		AFTER
A	← TOS →	R
B		B
C		C
D		D
← 32 →		← 32 →

CHSS

16-BIT FIXED-POINT SIGN CHANGE

Binary Coding:

7	6	5	4	3	2	1	0
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

BEFORE		AFTER
A	← TOS →	R
B		B
C		C
D		D
E		E
F		F
G		G
H		H
← 16 →		← 16 →

COS

32-BIT FLOATING-POINT COSINE

Binary Coding:

7	6	5	4	3	2	1	0
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

BEFORE		AFTER
A	← TOS →	R
B		B
C		
D		
← 32 →		← 32 →

DADD

32-BIT FIXED-POINT ADD

Binary Coding:

7	6	5	4	3	2	1	0
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

BEFORE		AFTER
A	← TOS →	R
B		C
C		D
D		A
← 32 →		← 32 →

DDIV

32-BIT FIXED-POINT DIVIDE

Binary Coding:

7	6	5	4	3	2	1	0
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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		C
C		D
D		╳
← 32 →		← 32 →

DMUL

32-BIT FIXED-POINT MULTIPLY, LOWER

Binary Coding:

7	6	5	4	3	2	1	0
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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		C
C		D
D		╳
← 32 →		← 32 →

DMUU

32-BIT FIXED-POINT MULTIPLY, UPPER

Binary Coding:

7	6	5	4	3	2	1	0
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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		C
C		D
D		╳
← 32 →		← 32 →

DSUB

32-BIT FIXED-POINT SUBTRACT

Binary Coding:

7	6	5	4	3	2	1	0
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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		C
C		D
D		A
← 32 →		← 32 →

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

BEFORE		AFTER
A	←TOS→	R
B		B
C		
D		
←32→		←32→

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

BEFORE		AFTER
A	←TOS→	R
B		C
C		D
D		
←32→		←32→

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

BEFORE		AFTER
A	←TOS→	R
B		C
C		D
D		
←32→		←32→

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

BEFORE		AFTER
A	←TOS→	R
B		B
C		C
D		
←32→		←32→

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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		Bu
C		Bl
D		Cu
← 32 →		Cl
		╳
		╳
		╳
		╳
		← 16 →

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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		B
C		C
D		╳
← 32 →		← 32 →

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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	Ru
B		Rl
C		B
D		C
E		D
F		E
G		╳
H		╳
← 16 →		← 16 →

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

STACK CONTENTS

BEFORE		AFTER
A	← TOS →	R
B		C
C		D
D		╳
← 32 →		← 32 →

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 $\neq$ 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)

BEFORE	STACK CONTENTS	AFTER
A	←TOS→	R
B		C
C		D
D		X
←32→		←32→

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 $\leq$ 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

BEFORE	STACK CONTENTS	AFTER
A	←TOS→	R
B		B
C		X
D		X
←32→		←32→

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$\leq$ 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

BEFORE	STACK CONTENTS	AFTER
A	←TOS→	R
B		B
C		X
D		X
←32→		←32→

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

BEFORE		AFTER
A	←TOS→	B
B		C
C		D
D		A
← 32 →		← 32 →

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

BEFORE		AFTER
A	←TOS→	B
B		C
C		D
D		A
← 32 →		← 32 →

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

BEFORE		AFTER
A	←TOS→	B
B		C
C		D
D		E
E		F
F		G
G		H
H		A
← 16 →		← 16 →

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

BEFORE		AFTER
A	←TOS→	A
B		A
C		B
D		C
← 32 →		← 32 →

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

BEFORE		AFTER
A	← TOS →	A
B		A
C		B
D		C
← 32 →		← 32 →

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

BEFORE		AFTER
A	← TOS →	A
B		A
C		B
D		C
E		D
F		E
G		F
H		G
← 16 →		← 16 →

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

BEFORE		AFTER
A	← TOS →	π
B		A
C		B
D		C
← 32 →		← 32 →

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 $B^A = \text{EXP}\ [A(\text{LN } B)]$. Thus if the term $[A(\text{LN } B)]$ 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 $[A(\text{LN } B)]$ 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

BEFORE		AFTER
A	←— TOS —→	R
B		C
C		╳
D		╳
←— 32 —→		←— 32 —→

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

BEFORE		AFTER
A	←— TOS —→	R
B		C
C		D
D		E
E		F
F		G
G		H
H		A
←16→		←16→

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 $A \neq 0$
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

BEFORE		AFTER
A	←TOS→	R
B		C
C		D
D		E
E		F
F		G
G		H
H		X
←16→		←16→

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 x 10^{-7} for input values in the range of -2π to $+2\pi$ radians.

Status Affected: Sign, Zero

STACK CONTENTS

BEFORE		AFTER
A	←TOS→	R
B		B
C		X
D		X
←32→		←32→

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

BEFORE		AFTER
A	←TOS→	R
B		C
C		D
D		E
E		F
F		G
G		H
H		
←16→		←16→

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

BEFORE		AFTER
A	←TOS→	R
B		C
C		D
D		E
E		F
F		G
G		H
H		
←16→		←16→

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

BEFORE	**STACK CONTENTS**	AFTER
A	←TOS→	R
B		B
C		C
D		
← 32 →		← 32 →

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

BEFORE	**STACK CONTENTS**	AFTER
A	←TOS→	R
B		C
C		D
D		E
E		F
F		G
G		H
H		A
← 16 →		← 16 →

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)

BEFORE	**STACK CONTENTS**	AFTER
A	←TOS→	R
B		B
C		
D		
← 32 →		← 32 →

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

BEFORE	**STACK CONTENTS**	AFTER
A	←TOS→	B
B		A
C		C
D		D
← 32 →		← 32 →

XCHF

EXCHANGE 32-BIT
STACK OPERANDS

Binary Coding:

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

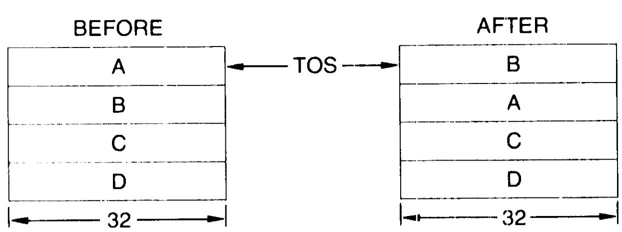
Hex Coding: 99 with sr = 1
19 with sr = 0

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

Binary Coding:

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

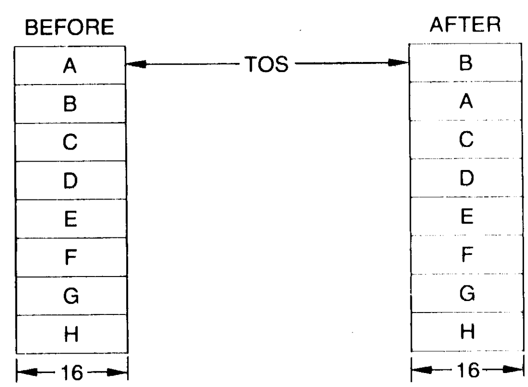
Hex Coding: F9 with sr = 1
79 with sr = 0

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



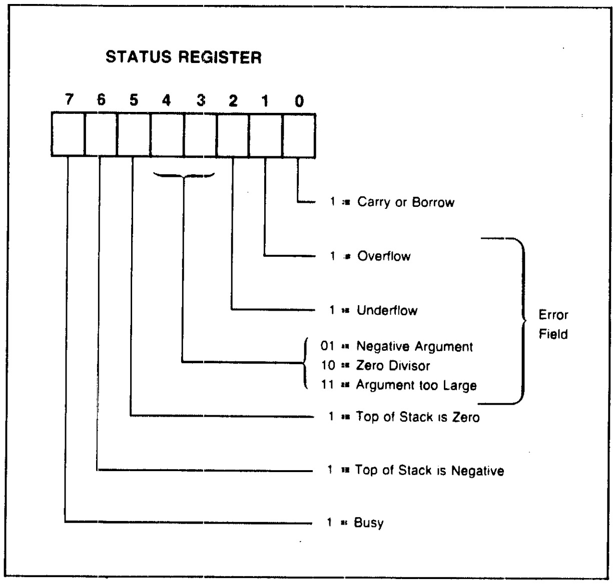


Figure 3-15. Am9511 Status Register Bit Definitions

3-38. FLOATING-POINT PROCESSOR

An optional Am9512 Floating-Point Processor performs single (32 bits) and double (64 bits) precision add, subtract, multiply, and divide floating-point operations. In addition, the Am9512 is capable of changing the sign of a single or double precision operand at the Top-Of-Stack (TOS) and Next-On-Stack (NOS), exchanging single or double precision operand located at TOS and NOS, as well as copying and popping single or double precision operands.

All transfers (e.g. operands, results, and commands) take place over an 8-bit bidirectional data bus. Operands are pushed onto an 8x17 LIFO stack and a command is issued to perform an operation on the stack. The results of this operation are retrieved by popping the stack. After completing an operation, the Am9512 activates an End-Of-Execution (END) signal that interrupts the CPU. The results from an operation are the same precsiion and format as the operands, and are rounded to maintain the accuracy.

3-39. Am9512 ADDRESSING

The Am9512 uses two consecutive addresses C0H and C1H for commands, data transfers, and status read. See table 3-1 for the port addresses and their functions.

3-40. Am9512 INITIALIZATION

The Am9512 does not require special initialization. After a power-on or system reset operation, the status register is cleared, the internal stack pointer is initialized (the contents of the stack may be affected), and the Am9512 is in the idle state.

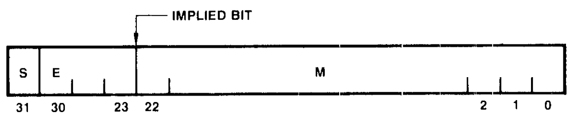
3-41. Am9512 DATA FORMATS

The Am9512 handles floating-point quantities in single precision or double precision formats:

The single precision quantities are 32 bits long as shown in figure 3-16. 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 shown in figure 3-16. The data stack operates as a push-down last-in/first-out (LIFO) stack; the data first written in is the last data read out.

When pushing operands on the stack, the least significant byte must be pushed first and the most significant byte pushed last. When popping the stack, the most significant byte will be available on the data bus first and the least significant byte will be last. In addition to pushing operands and popping results, the number of transactions must be equal to the proper number of bytes for the chosen format. Otherwise, the internal byte pointer will not be aligned properly.

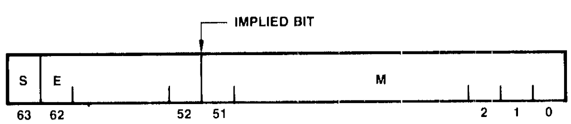
The Am9512 single precision format requires 4 bytes and the double precision format requires 8 bytes.



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.



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.

Figure 3-16. Am9512 Data Formats

3-42. Am9512 COMMAND DESCRIPTIONS

The following description of the Am9511 commands are presented in alphabetical order by command mnemonic. The command format is shown in figure 3-17.

The command consists of 8 bits; the least significant 7 bits specify the operation to be performed. 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 Execution times of the Am9512 command are all data dependent. Table 1-1 shows each command execution time.

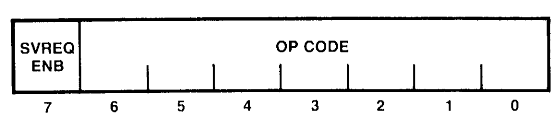


Figure 3-17. Am9512 Command Format

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

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.

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

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

BEFORE		AFTER
A	◄— TOS —►	R
B	◄— NOS —►	Undefined

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

BEFORE		AFTER
A	← TOS →	R (see note)
B	← NOS →	Undefined

Note: If A is zero, then R = B (Divide exception).

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

BEFORE		AFTER
A	← TOS →	R
B	← NOS →	Undefined

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

BEFORE		AFTER
A	← TOS →	R
B	← NOS →	Undefined

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

BEFORE		AFTER
A	← TOS →	B
B	← NOS →	C
C		D
D		A

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

BEFORE		AFTER
A	◄— TOS —►	A
B	◄— NOS —►	A

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

BEFORE		AFTER
A	◄— TOS —►	A* See note
B	◄— NOS —►	A
C		B
D		C

Note: A* = A if Exponent field of A is not zero.
A* = 0 if Exponent field of A is zero.

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

BEFORE		AFTER
A	◄— TOS —►	R
B	◄— NOS —►	C
C		D
D		Undefined

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

BEFORE		AFTER
A	◄— TOS —►	R (see note)
B	◄— NOS —►	C
C		D
D		Undefined

Note: If exponent field of A is zero then R = B (Divide exception).

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

BEFORE		AFTER
A	← TOS →	R
B	← NOS →	C
C		D
D		Undefined

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

BEFORE		AFTER
A	← TOS →	R
B	← NOS →	C
C		D
D		Undefined

3-43. Am9512 STATUS READ

The Am9512 contains an 8 bit status register as shown in figure 3-18. The status register can be read with an I/O read to port C1H, without regard to whether a command is in progress or not. When the status busy bit (bit 7) is high, the Am9512 is processing a command and the remainder of the status register bit indications are not valid unless the ERR occurs.

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 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.

Figure 3-18. Am9512 Status Register Formats

3-44. SYSTEM TIMING CONTROLLER PROGRAMMING

An Am9513 System Timing Controller is used to provide the timing and counting functions performed by the

TABLE 3-5. CONTROL ELEMENT SUMMARY

Control Element	Bit Size	Quantity
Output Control Bit	1	5
FOUT Divider Counter	4	1
Data Pointer Counter	6	1
Status Register	8	1
Command Register	8	1
Frequency Scaling Counter	16	1
Master Mode Register	16	1
Alarm Register	16	2
Comparator	16	2
Counter Mode Register	16	5
Counter Load Register	16	5
Counter Hold Register	16	5
General Counter	16	5

TABLE 3-6. Am9513 COMMAND SUMMARY

Arm Counters

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	1	S5	S4	S3	S2	S1

Description: Any combination of counters, as specified by the S field, will be enabled for counting. A counter must be armed before counting can commence. Once armed, the counting process may be further enabled or disabled using the hardware gating facilities. This command can only arm or do nothing for a given counter; a zero in the S field does not disarm the counter.

Load Counters

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	0	S5	S4	S3	S2	S1

Description: Any combination of counters, as specified in the S field, will be loaded with previously entered values. The source of information for each counter will be either the associated Load register or the associated Hold register, as determined by the operating configuration in the Mode register. The Load/Hold contents are not changed. This command will cause a transfer independent of any current operating configuration for the counter. It will often be used as a software retrigger, or as counter initialization prior to active hardware gating.

Load and Arm Counters

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	1	1	S5	S4	S3	S2	S1

Description: Any combination of counters, as specified in the S field, will be first loaded and then armed. This command is equivalent to issuing a LOAD command and then an ARM command.

Disarm Counters

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	0	S5	S4	S3	S2	S1

Description: Any combination of counters, as specified by the S field, will be disabled from counting. A disarmed counter will cease all counting independent of other control conditions. The only exception to this is that a counter in the TC state will always count once, in order to leave TC, before DISARMing. This count may be generated by a source edge, by a LOAD or LOAD-and-ARM command (the LOAD-and-ARM command will negate the DISARM command) or by a STEP command. A disarmed counter may be updated using the LOAD command and may be read using the SAVE command. A count process may be resumed using an ARM command.

Save Counters

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	0	1	S5	S4	S3	S2	S1

Description: Any combination of counters, as specified by the S field, will have their contents transferred into their associated Hold register. The transfer takes place without interfering with any counting that may be underway. This command will overwrite any previous Hold register contents. The SAVE command is designed to allow an accumulated count to be preserved so that it can be read by the host CPU at some later time.

TABLE 3-6. Am9513 COMMAND SUMMARY (Cont.)

Disarm and Save Counters

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	0	0	S5	S4	S3	S2	S1

Description: Any combination of counters, as specified by the S field, will be disarmed and the contents of the counter will be transferred into the associated Hold registers. This command is identical to issuing a DISARM command followed by a SAVE command.

Set Output

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	1	N4	N2	N1

(001 ≤ N ≤ 101)

Description: The output toggle for counter N is set. The OUTN signal will be driven high unless a TC output is specified.

Clear Output

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	0	N4	N2	N1

(001 ≤ N ≤ 101)

Description: The output toggle for counter N is reset. The OUTN signal will be driven low unless a TC output is specified.

Step Counter

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	1	0	N4	N2	N1

(001 ≤ N ≤ 101)

Description: Counter N is incremented or decremented by one, depending on its operating configuration. If the Counter Mode register associated with the selected counter has its CM3 bit cleared to zero, this command will cause the counter to decrement by one. If CM3 is set to a logic high, this command will increment the counter by one. The STEP command will take effect even on a disarmed counter.

Disable Data Pointer Sequencing

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	1	0	0	0

Description: This command sets Master Mode bit 14 without affecting other bits in the Master Mode register. MM14 controls the automatic sequencing of the Data Pointer register. Disabling the sequencing allows repetitive host processor access to a given internal location without repetitive updating of the Data Pointer. MM14 may also be controlled by loading a full word into the Master Mode register.

Enable Data Pointer Sequencing

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	0	0	0	0

Description: This command clears Master Mode bit 14 without affecting other bits in the Master Mode register. MM14 controls the automatic sequencing of the Data Pointer register. Enabling the sequencing allows sequential host processor access to several internal locations without repetitive updating of the Data Pointer. MM14 may also be controlled by loading a full word into the Master Mode register. See the "Data Pointer Register" section of this document for additional information on Data Pointer sequencing.

Enable 8-Bit Data Bus

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	0	1	1	1

Description: This command clears Master Mode bit 13 without affecting other bits in the Master Mode register. MM13 controls the multiplexer in the data bus buffer. When MM13 is cleared, the multiplexer is enabled and 16-bit internal information is transferred eight bits at a time to the eight low-order external data bus lines. MM13 may also be controlled by loading the full Master Mode register in parallel.

Gate Off FOUT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	1	1	1	0

Description: This command sets Master Mode bit 12 without affecting other bits in the Master Mode register. MM12 controls the output state of the FOUT signal. When gated off, the FOUT line will exhibit a low impedance to ground. MM12 may also be controlled by loading the full Master Mode register in parallel.

Gate On FOUT

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	0	0	1	1	0

Description: This command clears Master Mode bit 12 without affecting other bits in the Master Mode register. MM12 controls the output status of the FOUT signal. When MM12 is cleared, FOUT will become active and will drive out the selected and divided FOUT signal. MM12 may also be controlled by loading the full Master Mode register in parallel. When FOUT is gated on or off, a transient pulse may be generated on the FOUT signal.

Load Data Pointer Register

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
0	0	0	E2	E1	G4	G2	G1

(G4, G2, G1 ≠ 000, ≠ 110)

Description: Bits in the E and G fields will be transferred into the corresponding Element and Group fields of the Data Pointer

Master Reset

Coding:

C7	C6	C5	C4	C3	C2	C1	C0
1	1	1	1	1	1	1	1

Description: The Master Reset command duplicates the action of the power-on reset circuitry. It disarms all counters, enters 0000 in the Master Mode, Load and Hold registers and enters 0B00 (hex) in the Counter Mode registers.

Am95/4006. The system timing controller contains many control elements that allow it to be customized for particular applications as well as dynamically reconfigured under program control. These control elements, which include comparitors, registers, and counters, are summarized in table 3-5. Most of the registers can be both written into and read out of under CPU control. As delivered, control elements within the system timing controller are accessed by hexadecimal address D8H and D9H.

3-45. COMMAND REGISTER

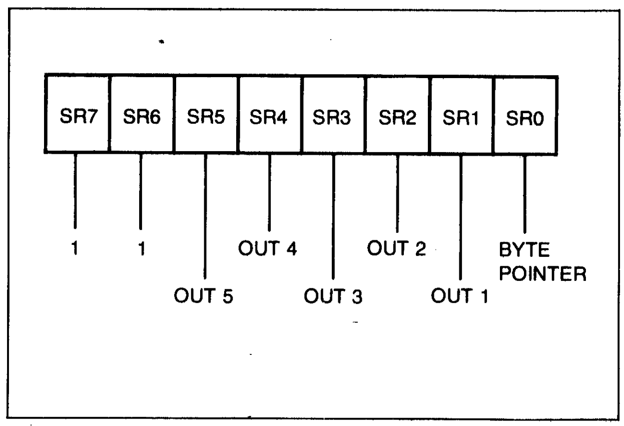
Direct CPU software control over many general counter functions is provided by the 8-bit command register. The CPU accesses the command register by performing an I/O write to address D8H. Command register codes and a brief description of each function is presented in table 3-6. Six of the command types are used for direct software control of the counting process and they each contain a five-bit S field. In a linear-select fashion, each bit in the S field corresponds to one of the five general counters (S1= Counter 1, S2= Counter 2, etc.). When an S bit is a one, the specified operation is performed on the counter so designated; when an S bit is a zero, no operation occurs for the corresponding counter. This type of command format has three basic advantages:

- Host software is conserved by allowing any combination of counters to be acted on by a single command.
- Facilitates simultaneous action of multiple counters where synchronization of commands is important.
- Allows counter specific service routine to control individual counters regardless of the operating context of other counters.

3-46. STATUS REGISTER

The 8-bit read-only status register, shown in table 3-7, indicates the state of the Byte Pointer bit in the Data Pointer register and the state of the OUT signal for each of the general counters. The OUT signals reported are those internal to the chip just before the three state interface buffer circuitry. Thus, the status register reflects the results of polarity control over the OUT signals, and continues to indicate the counter condition even when the output is off.

TABLE 3-7. STATUS REGISTER BITS



3-47. MASTER MODE REGISTER

The 16-bit Master Mode (MM) register is used to control those internal activities that are not controlled by the individual Counter Mode registers. This includes frequency control, time-of-day operation, comparator controls, data bus width and data pointer sequencing. Table 3-8 shows the bit assignments for the Master Mode register. Each control field is described in the following paragraphs.

After power-on reset or a Master Reset command, the Master Mode register is cleared to an all zero condition. This results in the following configuration:

- Time-of-day disabled
- Both Comparators disabled
- FOUT Source is frequency F1
- FOUT Divider set for divide-by-16
- FOUT gated on
- Data Bus 8 bits wide
- Data Pointer Sequencing enabled
- Frequency Scaler divides in binary

3-48. Time Of Day

Bits MM0 and MM1 of the Master Mode register specify the time-of-day (TOD) options. When MM0 = 0 and MM1 = 0, the special logic used to implement TOD is disabled and counters 1 and 2 will operate in exactly the same ways as counters 3, 4, and 5. When MM0 = 1 or MM1 = 1, additional counter decoding and control logic is enabled on counters 1 and 2 which causes them to turn-over at the counts that generate appropriate 24-hour TOD accumulations.

Table 3-9 shows the counter configurations for TOD operation. The two most significant decades of Counter 2 contain the hours digits and they can hold a maximum count of 23 hours. The two least significant decades of Counter 2 indicate minutes and will hold values up to 59. The three most significant decades of Counter 1 indicate seconds and will contain values up to 59.9. The least significant decade of Counter 1 is used to scale the input frequency in order to output tenth-of-second periods into the next decade. It can be set up to divide by five (MM0 = 1, MM1 = 0, divide by six (MM0 = 0, MM1 = 1), or divide by ten (MM0 = 1, MM1 = 1). The input frequency, therefore, for real-time clocking can be, respectively, 50Hz, 60Hz or 100Hz. These frequencies can be obtained by setting up counter 5 to divide the 2MHz or 4MHz oscillator frequency by the appropriate number and cascading the output of counter 5 to the input of counter 1. With divide-by-ten specified and with 100Hz input, the least significant decade of Counter 1 accumulates time in hundredths of seconds (tens of milliseconds). For accelerated time application other input frequencies may be useful.

3-49. Comparator Enable

Bits MM2 and MM3 control the Comparators associated with Counters 1 and 2. When a Comparator is enabled, its output is substituted for the normal counter output on the associated OUT1 or OUT2 pin. The comparator output will be active-high if the output control field of the counter mode register is code 001 or 010 and active-low for a code of 101. Once the compare output is true, it remains true until the count changes and the comparison therefore goes false.

The two Comparators can always be used individually in any operating mode. One special case occurs when the time-of-day option is selected and both Comparators are enabled. The operation of Comparator 2 is then conditioned by Comparator 1 so that a full 32-bit compare must be true in order to generate a true signal on OUT2. OUT1 continues, as usual, to reflect the state of the 16-bit comparison between Alarm 1 and Counter 1.

TABLE 3-8. MASTER MODE REGISTER BITS

MM15	MM14	MM13	MM12	MM11	MM10	MM9	MM8	MM7	MM6	MM5	MM4	MM3	MM2	MM1	MM0

FOUT Divider (MM11–MM8)
- 0000 = Divide by 16
- 0001 = Divide by 1
- 0010 = Divide by 2
- 0011 = Divide by 3
- 0100 = Divide by 4
- 0101 = Divide by 5
- 0110 = Divide by 6
- 0111 = Divide by 7
- 1000 = Divide by 8
- 1001 = Divide by 9
- 1010 = Divide by 10
- 1011 = Divide by 11
- 1100 = Divide by 12
- 1101 = Divide by 13
- 1110 = Divide by 14
- 1111 = Divide by 15

FOUT Source (MM7–MM4)
- 0000 = F1
- 0001 = SRC 1
- 0010 = SRC 2
- 0011 = SRC 3
- 0100 = SRC 4
- 0101 = SRC 5
- 0110 = GATE 1
- 0111 = GATE 2
- 1000 = GATE 3
- 1001 = GATE 4
- 1010 = GATE 5
- 1011 = F1
- 1100 = F2
- 1101 = F3
- 1110 = F4
- 1111 = F5

FOUT Gate (MM12)
- 0 = FOUT On
- 1 = FOUT Off (Low Z to GND)

Data Bus Width (MM13)
- 0 = 8-Bit Bus
- 1 =16-Bit Bus

Data Pointer Control (MM14)
- 0 = Enable Increment
- 1 = Disable Increment

Scaler Control (MM15)
- 0 = Binary Division
- 1 = BCD Division

Compare 2 Enable (MM3)
- 0 = Disabled
- 1 = Enabled

Compare 1 Enable (MM2)
- 0 = Disabled
- 1 = Enabled

Time-of-Day Mode (MM1–MM0)
- 00 = TOD Disabled
- 01 = TOD Enabled; ÷ 5 Input
- 10 = TOD Enabled; ÷ 6 Input
- 11 = TOD Enabled; ÷ 10 Input

TABLE 3-9. TIME-OF-DAY CONFIGURATIONS

Counter 2

C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
(2)				(3)				(5)				(9)			
Hours								Minutes							

Counter 1

C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
(5)				(9)				(9)							
Seconds								1/10 Sec.				÷5, 6, 10			

3-50. FOUT Source

Master Mode bits MM4 through MM7 specify the source input for the FOUT Divider. Fifteen inputs are available for selection and they include the five Source pins, the five Gate pins and the five internal frequencies derived from the oscillator. The 16th combination of the four control bits (all zeros) is used to assure that an active frequency is available on FOUT following a reset.

3-51. FOUT Divider

Bits MM8 through MM11 specify the dividing ratio for the FOUT Divider. The FOUT source (selected by bits MM4 through MM7) is divided by an integer value between 1 and 16, inclusive, and is then passed to the FOUT output buffer. The undivided oscillator frequency is available at FOUT simply by selecting the F1 source and a dividing ratio of one. After power-on or reset, the FOUT Divider is set to divide by sixteen.

3-52. FOUT Gate

Master Mode bit MM12 provides a software gating capability for the FOUT signal. When MM12 = 1, FOUT is off and in a low impedance state to ground. MM12 can be set or cleared in conjunction with the loading of the other bits in the Master Mode register; alternatively, there are commands that allow MM12 to be individually set or cleared directly without changing any other Master Mode bits. After power-up or reset, FOUT is gated on.

3-53. Bus Width

Bit MM13 controls the multiplexer at the data bus interface in order to configure the part for an 8-bit or 16-bit external bus. The internal bus is always 16-bits wide. Always set MM13 = 0 so that 16-bit internal data is transferred, a byte at a time, to and from the eight low-order external data bus lines. The Byte Pointer bit toggles with each byte transfer in this mode.

3-54. Data Pointer Sequencing

Bit MM14 controls the Data Pointer logic to enable or disable the automatic sequencing functions. When MM14 = 1, the contents of the Data Pointer can be changed only directly by entering a command. When MM14 = 0, several types of automatic Data Pointer sequencing are available. Thus the host processor, by controlling MM14, can repetitively read/write a single internal location, or can sequentially read/write groups of locations.

3-55. Scaler Ratios

Master Mode bit MM15 controls the counting configuration of the Frequency Scaler counter. When MM15 = 0, the Scaler divides the oscillator frequency in binary steps so that each subfrequency is 1/16 of the preceding frequency. When MM15 = 1, the Scaler divides in BCD steps so that adjacent frequencies are related by ratios of 10 instead of 16.

3-56. Frequency Scaling Counter

A 16-bit scaling counter divides the output of the on-chip oscillator into four additional sub-frequencies. This provides a total of five internal frequencies that can be routed to any of the general counters and to the FOUT divider. The scaler is tapped every four bits and can be programmed to divide in binary or in BCD. The combinations of frequencies thus available are shown in figure 3-19. For example, if the base oscillator frequency is 8MHz, the F4 frequency will be 8KHz when BCD scaling is selected. If the base oscillator frequency is 8MHz, the F3 frequency will be 31.25 KHz when binary scaling is selected. The control bit that selects BCD or binary scaling is located in the Master Mode register.

3-57. FOUT DIVIDER COUNTER

The 4-bit FOUT Divider is used to subdivide the source selected for the Frequency Out pin. It provides for division by an integer from 1 to 16, inclusive. The dividing ratio is selected by a 4-bit field in the Master Mode register. The FOUT Divider is intended to allow a relatively low FOUT frequency for use as a system clock while still permitting higher resolution internal frequencies from the oscillator. In applications that do not use FOUT as a frequency source or as a clock, the Divider forms a simple generalpurpose programmable 4-bit divider that can use any of the general input pins.

3-58. DATA POINTER COUNTER

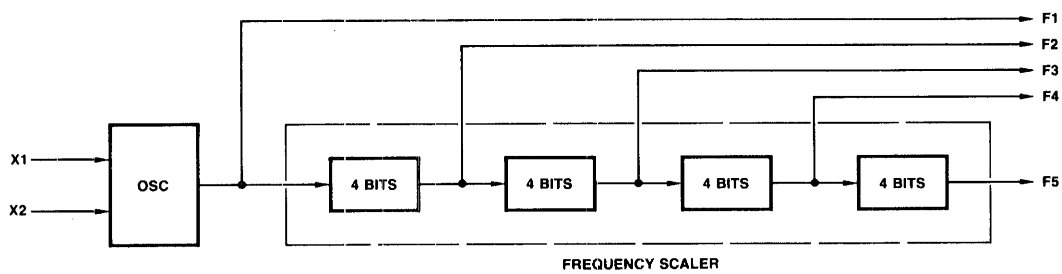
The 6-bit Data Pointer register is loaded by issuing the appropriate command through the control port to the Command register. The contents of the Data Pointer register are used to control the Data Port multiplexer, selecting which internal register is to accessible through the Data Port.

The Data Pointer consists of a 3-bit Group Pointer, a 2-bit Element Pointer, and a 1-bit Byte Pointer, depicted in figure 3-20. The Byte Pointer bit indicates which byte of a 16-bit register is to be transferred on the next access through the data port. Whenever the Data Pointer is loaded, the Byte Pointer bit is set to one, indicating a least-significant byte is expected. The Byte Pointer toggles following each 8-bit data transfer with an 8-bit data bus (MM13=0). The Element and Group pointers are used to select which internal register is to be accessible through the Data Port. Although the contents of the Element and Group Pointer in the Data Pointer register cannot be read by the host processor, the Byte Pointer is available as a bit in the Status register.

Random Access to any available internal data location can be accomplished by simply loading the Data Pointer using the command shown in figure 3-21 and then initiating a data read or data write. This procedure can be used at any time, regardless of the setting of the Data Pointer Control bit (MM14). When the 8-bit data bus configuration is being used (MM13=0), two bytes of data would normally be transferred following the issuing of the Load Data Pointer command.

To permit the host processor to rapidly access the various internal registers, automatic sequencing of the Data Pointer is provided. Sequencing is enabled by clearing Master Mode bit 14 (MM14) to zero.

When E1=0 or E2=0 and G4, G2, G1 point to a Counter Group, the Data Pointer will proceed through the Element with associated control and output logic, a 16-bit Load register, a 16bit Hold register and a 16-bit Mode register. In addition, Counter Groups 1 and 2 also include 16-bit comparators and 16-bit Alarm registers. The comparator/alarm functions are controlled by the Master Mode register. The operation of the Counter Mode registers is the same for all five counters. The host CPU has both read and write access to all registers in values 00,



Frequency	BCD Scaling MM15 = 1	Binary Scaling MM15 = 0
F1	OSC	OSC
F2	F1 ÷ 10	F1 ÷ 16
F3	F1 ÷ 100	F1 ÷ 256
F4	F1 ÷ 1,000	F1 ÷ 4,096
F5	F1 ÷ 10,000	F1 ÷ 65,536

AMC-092

Figure 3-19. Frequency Scaler Ratios

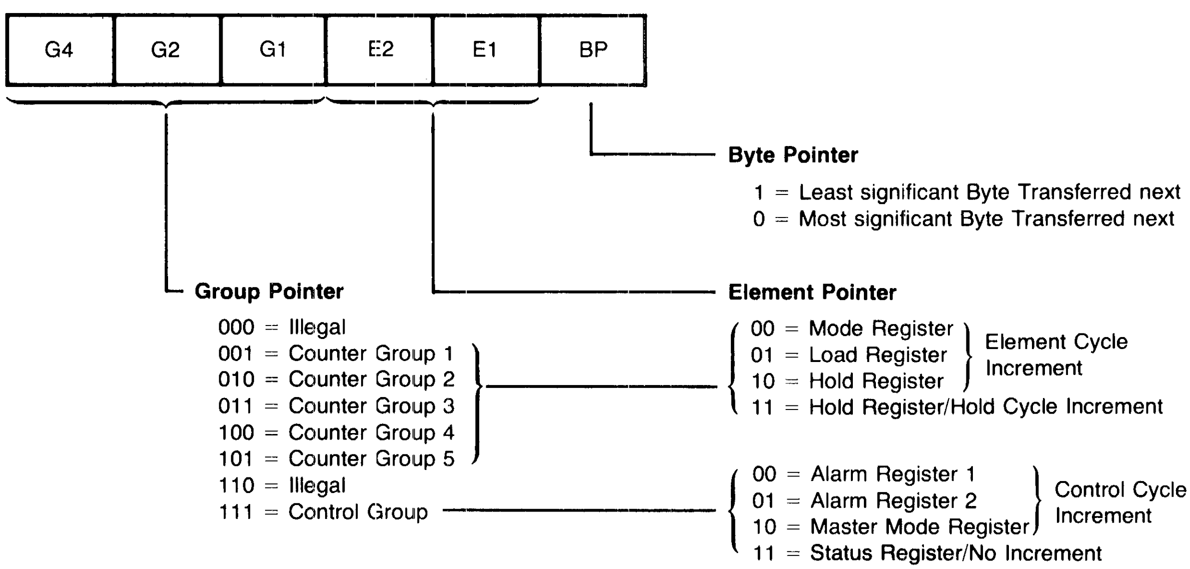


Figure 3-20. Data Pointer Counter

	ELEMENT CYCLE			HOLD CYCLE
	MODE REGISTER	LOAD REGISTER	HOLD REGISTER	HOLD REGISTER
Counter 1	FF01	FF09	FF11	FF19
Counter 2	FF02	FF0A	FF12	FF1A
Counter 3	FF03	FF0B	FF13	FF1B
Counter 4	FF04	FF0C	FF14	FF1C
Counter 5	FF05	FF0D	FF15	FF1D
Master Mode Register = FF17 Alarm 1 Register = FF07 Alarm 2 Register = FF0F				

NOTES

1. All codes are in hex.
2. When used with an 8-bit bus, onlythe two low order hex digits should be written to the command port; the FF prefix should be used only for a 16-bit data bus interface.

Figure 3-21. Load Data Pointer Commands

01, and 10 starting with the value entered. When the transition from 10 to 00 occurs, the Group field will also be incremented by one. Note that the Element field circulates only within the five Counter Group codes.

If E2, E1=11 and a Counter Group is selected, then only the Group field is sequenced. This is the Hold cycle. It allows the Hold registers to be sequentially accessed while bypassing the Mode and Load registers. The third type of sequencing is th Control cycle. If G4, G2, G1=111 and E2, E1≠11, the Element Pointer will be incremented through the values 00, 01, and 10, with no change to the Group Pointer.

When the G4, G2, G1=111, no incrementing takes place and only the Status register will be available through the data port. Note that the Status register can also always be read directly through the Control port.

For all of these auto-sequence modes, since an 8-bit data bus is being used, the Byte pointer must toggle after every data transfer to allow the least and most significant bytes to be transferred before the Element or Group Fields are incremented.

3-59. PREFETCH CIRCUIT

In order to minimize the read access time to internal Am9513 registers, a prefetch circuit is used for all read operations through the Data Port.

Following each read or write operation through the Data Port, the Data Pointer register is updated to point to the next register to be accessed. Immediately following this update, the new register data is transferred to a special prefetch latch at the interface pad logic. When the user performs a subsequent read of the Data Port, the data bus drivers are enabled, outputting the prefetched data on the bus. Since the internal data register is accessed prior to the start of the read operation, its access time is transparent to the user. In order to keep the prefetched data consistent with the data pointer, prefetches are also performed after each write to the Data Port and after execution at the Load Data Pointer command. The following rules should be kept in mind regarding Data Port Transfers.

1. The Data Pointer register should always be reloaded before reading from the Data Port if a command other than Load Data Pointer was issued to the Am9513 following the last Data Port read or write. The Data Pointer does not have to be loaded again if the first Data Port transaction after a command entry is a write, since the Data Port write will automatically cause a new prefetch to occur.

2. Operating modes N, O, Q, and R allow the user to save the counter contents in the Hold register by applying an active-going gate edge. If the Data Pointer register had been pointing to the Hold register in question, the prefetched value will not correspond to the new value saved in the Hold register. To avoid reading an incorrect value, a new Load Data Pointer command should be issued before attempting to read the saved data. A Data Port write (to another register) will also initiate a prefetch; subsequent reads will access the recently Saved Hold register data.

3-60. COUNTER LOGIC GROUPS

Each of the five Counter Logic Groups consists of a 16-bit general counter with associated control and output logic, a 16-bit Load register and a 16-bit Mode register. In addition, Counter Groups 1 and 2 also include 16-bit comparators and 16-bit Alarm registers. The comparator/alarm functions are controlled by the Master Mode register. The operation of the Counter Mode registers is the same for all five counters. The host CPU has both read and write access to all registers in the Counter Logic Groups. The counter itself is never directly accessed.

3-61. Counter Load Register

The 16-bit read/write Load register is used to control the effective length of the general counter. Any 16-bit value can be written into the Load register. That value can then be transferred into the counter each time the Terminal Count (TC) occurs. Terminal Count is defined as that period of time when the counter contents would have been zero if an external value had not been transferred into the counter. Thus, the terminal count frequency can be the input frequency divided by the value in the Load register. In all operating modes, either Load or Hold will be transferred into the counter when TC occurs. In cases where values are being accumulated in the counter, the Load register action can become transparent by filling the Load register with all zeros.

3-62. Counter Hold Register

The 16-bit read/write Hold register is dual-purpose. It can be used in the same way as the Load register, thus offering an alternate source for modulo definition for the counter. The Hold register can also be used to store accumulated counter values for later transfer to the CPU. This allows the count to be sampled while the counting process proceeds without interruption. Transfer of the counter contents into the Hold register is accomplished by the hardware interface in some operating modes or the by software save commands at any time.

3-63. Counter Mode Register

Each Counter Logic Group includes a Counter Mode (CM) register used to control all of the individual options available with its associated general counter. These options include output configuration, count control, count source and gating control. Figure 3-22 shows the bit assignments for the Count Mode registers. The following paragraphs describe the control options in detail. Note that generally each counter is independently configured and does not depend on information outside its Counter Logic Group.

The Counter Mode register should be loaded only when the counter is disarmed. Attempts to load the Counter Mode register when the counter is armed can result in erratic counter operation.

After power-on reset or a Master Reset command, the Counter Mode registers are initialized to a preset condition. The value entered is 0B00 hex and results in the following control configuration:

Output low impedance to ground

Count down

Count binary

Count once

Load register selected

No retriggering

F1 input source selected

Positive-true input polarity

No gating

3-64. Output Control

Counter Mode bits CM0 through CM2 specify the output control configuration. The OUT pin can be off and in a high impedance state or it can be off with a low impedance to ground. The six remaining combinations are split into active-high and active-low versions of the three basic output waveforms.

One output form available is called Terminal Count (TC) and represents the period in time that the counter reaches an equivalent value of zero. TX will occur on the next count when the counter is at 0001 for down counting, at 9999 (BCD) for BCD up counting or at FFFF (hex) for binary up counting. Figure 3-23 shows a Terminal Count pulse and an example context that generated it. The TC width is determined by the period of the counting source. Regardless of any gating input or whether the counter is Armed or Disarmed, the terminal count will go active for only one clock cycle. Figure 3-23 assumes active-high source polarity, counter armed, counter decrementing, and an external reload value of K.

The counter will always be loaded from an external location when TC occurs; the user can choose the source location and the value. If a non-zero value is picked, the counter will never really attain a zero state and TC will indicate the counter state that would have been zero had no parallel transfer occurred.

The other output form, TC Toggled, uses the trailing edge of TC to toggle a flip-flop to generate an output level instead of a pulse.

The toggle output is 1/2 the frequency of TC. The TC Toggled output will frequency be used to generate variable duty-cycle square waves in Operating Modes G through K.

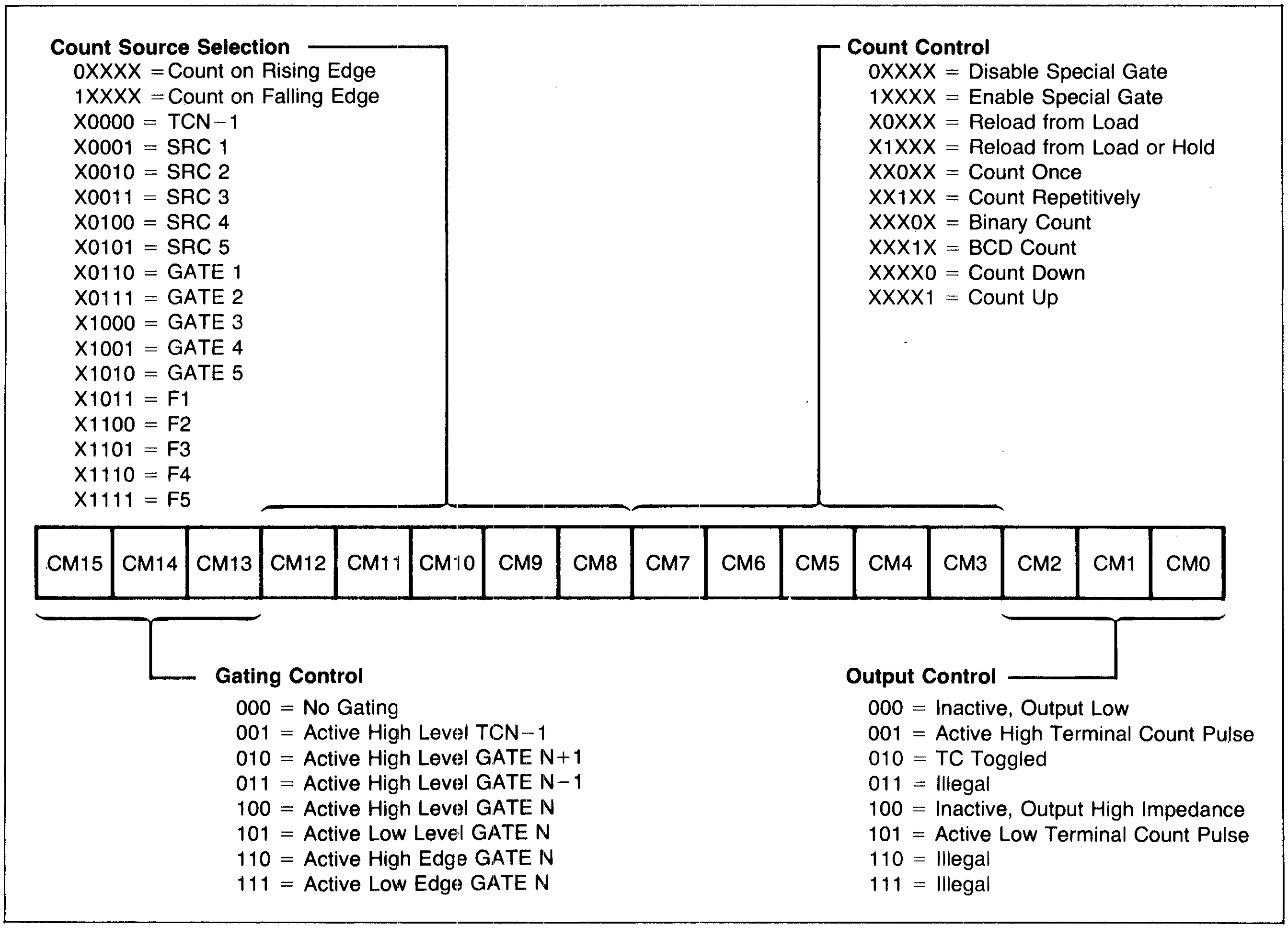


Figure 3-22. Counter Mode Register Bit Assignments

In Mode L the TC Toggled output can be used to generate a one-shot function, with the delay to the start of the output pulse and the width of the output pulse separately programmable. With selection of the minimum delay to the start of the pulse, the output will toggle on the source pulse following application of the triggering Gate edge.

Note that the TC Toggled output form contains no implication about whether the output is active-high or active-low. Unlike the TC output, which generates a transient pulse which can clearly be active-high or active-low, the TC Toggled ouput waveform only flips the state of the output on each TC. The sole criteria of whether the TC Toggled output is active-high or active-low is the level of the output at the start of the count cycle. This can be controlled by the Set and Clear Output commands.

3-65. Count Control

Counter Mode bits CM3 through CM7 specify the various options available for direct control of the counting process. CM3 and CM4 operate independently of the others and control up/down and BCD/binary counting. They may be combined freely with other control bits to form many types of counting configurations. The other three bits and the Gating Control field interact in complex ways. Bit CM5 controls the repetition of the count process. When CM5 = 1, counting will proceed in the specified mode until the counter is

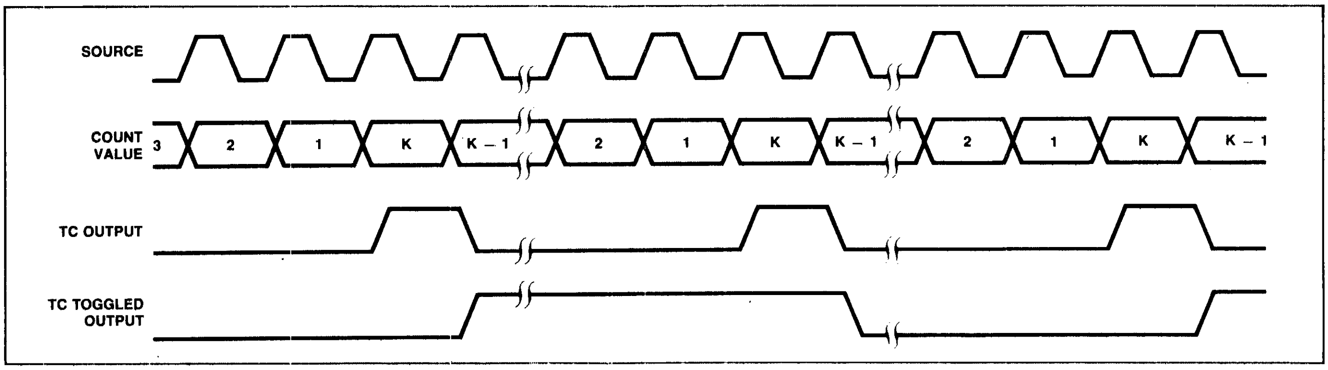


Figure 3-23. TC Waveform Format

disarmed. When CM5 = 0, the count process will proceed only until one full cycle of operations occurs. This may occur after one or two TC events. The counter is then disarmed automatically. The single or double TC requirements will depend on the state of other control bits. Note that even if the counter is automatically disarmend upon a TC, it always counts the count source edge which generates the trailing TC edge.

When TC occurs, the counter is always reloaded with a value from either the Load register or the Hold register.

Bit CM6 specifies the source options for reloading the counter. When CM6 = 0, the contents of the Load register will be transferred into the counter at every occurrence of TC. When CM6 = 1, the counter reload location will be either the Load or Hold Register. The reload location in this case can be controlled externally by using a GATE pin (Modes S and V) or can alternate on each TC (Modes G through L). With alternating sources and with the TC Toggled output selected, the duty cycle of the output waveform is controlled by the relative Load and Hold values and very fine resolution of duty cycle ratios can be achieved.

Bit CM7 controls the special gating functions that allow retriggering and the selection of Load or Hold sources for counter reloading. The use and definition of CM7 will depend on the status of the Gating Control field and bits CM5 and CM6.

When some form of Gating is specified, CM7 controls hardware retriggering. In this case, when CM7 = 0, hardware retriggering does not occur; when CM7 = 1, the counter is retriggered any time an active-going Gate edge occurs. Retriggering causes the counter value to be saved in the Hold register and the Load Register contents to be transferred into the counter.

Whenever hardware retriggering is enabled (Modes N, O, Q, and R) all active going Gate edges initiate retrigger operations. On application of the Gate edge, the counter contents will be transferred to the Hold register. On the first qualified source edge after application of the retriggering Gate edge, the Load register contents will be transferred into the counter. (Qualified source edges are edges which occur while the counter is gated on and Armed.)

This means that if level gating is used, the edge occurring on active-going gate transitions will initiate a retrigger. Similarly, when edge gating is enabled, an edge used to start the counter will also initiate a retrigger. The first count source edge applied after the Gate edge will not increment/decrement the counter but reload it.

When No Gating is specified, the definition of CM7 changes. In this case, when CM7=0, the Gate input has no effect on the counting; when CM7=1, the Gate N input specifies the reload source (either the Load or Hold register) used to reload the counter when TC occurs.

3-66. Count Source Selection

Counter Mode bits CM8 through CM12 specify the source used as input to the counter and the active edge that is counted. Bit CM12 controls the polarity for all the sources; logic zero counts rising edges and logic one counts falling edges. Bits CM8 through CM11 select one of sixteen counting sources to route to the counter input. Five of the available inputs are internal frequencies derived from the internal oscillator. Ten of the available inputs are interface pins; five are labeled SRC and five are labeled GATE.

The sixteenth available input is the TC output from the adjacent lower numbered counter. (The Counter 5 TC wraps around to the Counter 1 input.) This option allows internal concatenating that permits very long counts to be accumulated. Since all five counters can be concatenated, it is possible to configure a counter that is 80 bits long on one Am9513 chip. When TCN-1 is the source, the count ripples between the connected counters. External connections can also be made, and can use the toggle bit for even longer counts. This is easily accomplished by selecting a TOG output mode and wiring OUTN to one of the SRC inputs.

3-67. Gating Control

Counter Mode bits CM13 through CM15 specify the hardware gating options. When no gating is selected (000), the counter will proceed unconditionally as long as it is armed. For any other gating mode, the count process is conditioned by the specified gating configuration.

For a code of 100 in this field, counting can proceed only when the pin labeled GATEN associated with Counter N is at a logic high level. When it goes low, counting is simply suspended until Gate goes high again. A code of 101 performs the same function with an opposite active polarity. Codes 010 and 011 offer the same functions as 100, but specify alternate input pins as Gating Sources. This allows any of three interface pins to be used as gates for a given counter. On Counter 4, for example, pin 34, pin 35 or pin 36 can be used to perform the gating function. This also allows a single Gate pin to simultaneously control up to three counters.

For codes of 110 or 111 in this field, counting proceeds after the specified active Gate edge until one or two TC events occur. Within this interval the Gate input is ignored, except for the retriggering option. When repetition is selected, a cycle is repeated as soon as another Gate edge occurs. This mode is useful when implementing a digital single-shot since the gate can serve as a convenient firing trigger.

A 001 code in this field selects the TC output from the adjacent lower/numbered counter as the gate. Thus, one counter can be configured to generate a counting window for another counter.

3-68. Operating Modes

Counter Mode register bits CM15-CM13 and CM7-CM5 select the operating mode for each counter (see figure 3-24). To simplify references to a particular mode, each mode is assigned a letter from A through X. Operating Modes are described in table 3-10.

To keep the mode descriptions concise and to the point, the phrase source edges is used to refer to active-going source edges only, not to inactive-going edges. Similarily, the phrase gate edges refers only to active-going gate edges. Also, again to avoid verbosity and euphuism, the descriptions of some modes states that a counter is stopped or disarmed on a TC, inhibiting further counting. As is fully explained in the TC section for these modes the counter is actually stopped or disarmed following the active-going source edge which drives the counter out of TC. In other words, since a counter in the TC state always counts, irrespective of its gating or arming status, the stopping or disarming of the count sequence is delayed until TC is terminated.

3-69. TC (Terminal Count)

On each Terminal Count (TC), the counter will reload itself from the Load or Hold register. TC is defined as that period of time when the counter contents would have been zero had no reload occurred. Some special conditions apply to counter operation immediately before and during TC.

1. In the clock cycle before TC, an internal signal is generated that commits the counter to go to TC on the next count, and retriggering by a hardware Gate edge (Modes N, O, Q, and R) or a software Load or Load-and-Arm command will not extend the time to TC. Note that the next count driving the counter to TC can be caused by the application of a count source edge (in level gating modes, the edge must occur while the gate is active, or it will be disregarded), by the application of a Load or Load-and-Arm command (see 2 below) or by the application of a Step command.

2. If a Load or Load-and-Arm command is executed during the cycle preceding TC, the counter will immediately go to TC. If these commands are issued during TC, the TC state will immediately terminate.

3. When TC is active, the counter will always count the next source edge issued to it, even if it is disarmed or gated off during TC. This means that TC will never be active for longer than one count period and it may, in fact, be shorter if a Step command or a Load or Load-and-Arm command is applied during TC (see item 2 above). This also means that a counter that is disarmed or stopped on TC is actually disarmed/stopped immediately following TC.

This may cause count sequence different from what a user might expect. Since the counter is always reloaded at the start of TC, and since it always counts at the end of TC, the counter contents follwoing TC will differ by one from the reloaded value, irrespective of the operating mode used.

If the reloaded value was 0001 for down counting, 9999 (BCD) for BCD up counting or FFF (hex) for binary up counting, the count at the end of TC will drive the counter into TC again regardless of whether the counter is gated off or disarmed. As long as these values are reloaded, the TC output will stay active. If a TC Toggled output is selected, it will

Operating Mode	A	B	C	D	E	F	G	H	I	J	K	L
Special Gate (CM7)	0	0	0	0	0	0	0	0	0	0	0	0
Reload Source (CM6)	0	0	0	0	0	0	1	1	1	1	1	1
Repetition (CM5)	0	0	0	1	1	1	0	0	0	1	1	1
Gate Control (CM15-CM13)	000	LEVEL	EDGE	000	LEVEL	EDGE	000	LEVEL	EDGE	000	LEVEL	EDGE
Count to TC once, then disarm	X	X	X									
Count to TC twice, then disarm							X	X	X			
Count to TC repeatedly				X	X	X				X	X	X
Gate input does not gate counter input	X			X			X			X		
Count only during active gate level		X			X			X			X	
Start count on active gate edge and stop count on next TC.			X			X						
Start count on active gate edge and stop count on second TC.									X			X
No hardware retriggering	X	X	X	X	X	X	X	X	X	X	X	X
Reload counter from Load Register on TC	X	X	X	X	X	X						
Reload counter on each TC, alternating reload source between Load and Hold Registers.							X	X	X	X	X	X
Transfer Load Register into counter on each TC that gate is LOW; transfer Hold Register into counter on each TC that gate is HIGH.												
On active gate edge transfer counter into Hold Register and then reload counter from Load Register.												

Operating Mode	M	N	O	P	Q	R	S	T	U	V	W	X
Special Gate (CM7)	1	1	1	1	1	1	1	1	1	1	1	1
Reload Source (CM6)	0	0	0	0	0	0	1	1	1	1	1	1
Repetition (CM5)	0	0	0	1	1	1	0	0	0	1	1	1
Gate Control (CM15-CM13)	000	LEVEL	EDGE	000	LEVEL	EDGE	000	LEVEL	EDGE	000	LEVEL	EDGE
Count to TC once, then disarm		X	X									
Count to TC twice, then disarm							X					
Count to TC repeatedly					X	X				X		
Gate input does not gate counter input							X			X		
Count only during active gate level		X			X							
Start count on active gate edge and stop count on next TC.			X			X						
Start count on active gate edge and stop count on second TC.												
No hardware retriggering							X			X		
Reload counter from Load Register on TC		X	X		X	X						
Reload counter on each TC, alternating reload source between Load and Hold Registers.												
Transfer Load Register into counter on each TC that gate is LOW; transfer Hold Register into counter on each TC that gate is HIGH.							X			X		
On active gate edge transfer counter into Hold Register and then reload counter from Load Register.		X	X		X	X						

Note: Operating modes M, P, T, U, W and X are reserved and should not be used.

Figure 3-24. Counter Control Interaction

TABLE 3-10. OPERATING MODES

MODE A
Software-Triggered Strobe with No Hardware Gating

Mode A is one of the simplest operating modes. The counter will be available for counting source edges when it is issued an ARM command. On each TC the counter will reload from the Load register and automatically disarm itself, inhibiting further counting. Counting will resume when a new ARM command is issued.

MODE B
Software-Triggered Strobe with Level Gating

Mode B is identical to Mode A except that source edges are counted only when the assigned Gate is active. The counter must be armed before counting can occur. Once armed, the counter will count all source edges which occur while the Gate is active and disregard those edges which occur while the Gate is inactive. This permits the Gate to turn the count process on and off. On each TC the counter will reload from the Load register and automatically disarm itself, inhibiting further counting until a new ARM command is issued.

MODE C
Hardware-Triggered Strobe

Mode C is identical to Mode A, except that counting will not begin until a Gate edge is applied to the armed counter. The counter must be armed before application of the triggering Gate edge; Gate edges applied to a disarmed counter are disregarded. The counter will start counting on the first source edge after the triggering Gate edge and will continue counting until TC. At TC, the counter will reload from the Load register and automatically disarm itself. Counting will then remain inhibited until a new ARM command and a new Gate edge are applied in that order. Note that after application of a triggering Gate edge, the Gate input will be disregarded for the remainder of the count cycle. This differs from Mode B, where the Gate can be modulated throughout the count cycle to stop and start the counter.

MODE D
Rate Generator with No Hardware Gating

Mode D is typically used in frequency generation applications. In this mode, the Gate input does not affect counter operation. Once armed, the counter will count to TC repetitively. On each TC the counter will reload itself from the Load register; hence the Load register value determines the time between TCs. A square wave rate generator may be obtained by specifying the TC Toggled output mode in the Counter Mode register.

MODE E
Rate Generator with Level Gating

Mode E is identical to Mode D, except the counter will only count those source edges which occur while the Gate input is active. This feature allows the counting process to be enabled and disabled under hardware control. A square wave rate generator may be obtained by specifying the TC Toggled output mode.

MODE F
Non-Retriggerable One-Shot

Mode F provides a non-retriggerable one-shot timing function. The counter must be armed before it will function. Application of a Gate edge to the armed counter will enable counting. When the counter reaches TC, it will reload itself from the Load register. The counter will then stop counting, awaiting a new Gate edge. Note that unlike Mode C, a new ARM command is not needed after TC, only a new Gate edge. After application of a triggering Gate edge, the Gate input is disregarded until TC.

MODE G
Software-Triggered Delayed Pulse One-Shot

In Mode G, the Gate does not affect the counter's operation. Once armed, the counter will count to TC twice and then automatically disarm itself. For most applications, the counter will initially be loaded from the Load register either by a LOAD command or by the last TC of an earlier timing cycle. Upon counting to the first TC, the counter will reload itself from the Hold register. Counting will proceed until the second TC, when the counter will reload itself from the Load register and automatically disarm itself, inhibiting further counting. Counting can be resumed by issuing a new ARM command. A software-triggered delayed pulse one-shot may be generated by specifying the TC Toggled output mode in the Counter Mode register. The initial counter contents control the delay from the ARM command until the output pulse starts. The Hold register contents control the pulse duration.

MODE H
Software-Triggered Delayed Pulse One-Shot with Hardware Gating

Mode H is identical to Mode G except that the Gate input is used to qualify which source edges are to be counted. The counter must be armed for counting to occur. Once armed, the counter will count all source edges that occur while the Gate is active and disregard those source edges that occur while the Gate is inactive. This permits the Gate to turn the count process on and off. As with Mode G, the counter will be reloaded from the Hold register on the first TC and reloaded from the Load register and disarmed on the second TC. This mode allows the Gate to control the extension of both the initial output delay time and the pulse width.

MODE I
Hardware-Triggered Delayed Pulse Strobe

Mode I is identical to Mode G, except that counting will not begin until a Gate edge is applied to an armed counter. The counter must be armed before application of the triggering Gate edge; Gate edges applied to a disarmed counter are disregarded. An armed counter will start counting on the first source edge after the triggering Gate edge. Counting will then proceed in the same manner as in Mode G. After the second TC, the counter will disarm itself. An ARM command and Gate edge must be issued in this order to restart counting. Note that after application of a triggering Gate edge, the Gate input will be disregarded until the second TC. This differs from Mode H, where the Gate can be modulated throughout the count cycle to stop and start the counter.

MODE J

Variable Duty Cycle Rate Generator with No Hardware Gating

Mode J will find the greatest usage in frequency generation applications with variable duty cycle requirements. Once armed, the counter will count continuously until it is issued a DISARM command. On the first TC, the counter will be reloaded from the Hold register. Counting will then proceed until the second TC at which time the counter will be reloaded from the Load register. Counting will continue, with the reload source alternating on each TC, until a DISARM command is issued to the counter. (The third TC reloads from the Hold register, the fourth TC reloads from the Load register, etc.) A variable duty cycle output can be generated by specifying the TC Toggled output in the Counter Mode register. The Load and Hold values then directly control the output duty cycle, with high resolution available when relatively high count values are used.

MODE K

Variable Duty Cycle Rate Generator with Level Gating

Mode K is identical to Mode J except that source edges are only counted when the Gate is active. The counter must be armed for counting to occur. Once armed, the counter will count all source edges which occur while the Gate is active and disregard those source edges which occur while the Gate is inactive. This permits the Gate to turn the count process on and off. As with Mode J, the reload source used will alternate on each TC, starting with the Hold register on the first TC after any ARM command. When the TC Toggled output is used, this mode allows the Gate to modulate the duty cycle of the output waveform. It can affect both the high and low portions of the output waveform.

MODE L

Hardware-Triggered Delayed Pulse One-Shot

Mode L is similar to Mode J except that counting will not begin until a Gate edge is applied to an armed counter. The counter must be armed before application of the triggering Gate edge; Gate edges applied to a disarmed counter are disregarded. The counter will start counting source edges after the triggering Gate edge and counting will proceed until the second TC. Note that after application of a triggering Gate edge, the Gate input will be disregarded for the remainder of the count cycle. This differs from Mode K, where the gate can be modulated throughout the count cycle to stop and start the counter. On the first TC after application of the triggering Gate edge, the counter will be reloaded from the Hold register. On the second TC, the counter will be reloaded from the Load register and counting will stop until a new gate edge is issued to the counter. Note that unlike Mode K, new Gate edges are required after every second TC to continue counting.

MODE N

Software-Triggered Strobe with Level Gating and Hardware Retriggering

Mode N provides a software-triggered strobe with level gating that is also hardware retriggerable. The counter must first be issued an ARM command before counting can occur. Once armed, the counter will count all source edges which occur while the gate is active and disregard those source edges which occur while the Gate is inactive. This permits the Gate to turn the count process on and off. After the issuance of an ARM command and the application of an active Gate, the counter will count to TC. Upon reaching TC, the counter will reload from the Load register and automatically disarm itself, inhibiting further counting. Counting will resume upon the issuance of a new ARM command. All active-going Gate edges issued to an armed counter will cause a retrigger operation. Upon application of the Gate edge, the counter contents will be saved in the Hold register. On the first qualified source edge after application of the retriggering gate edge the contents of the Load register will be transferred into the counter. Counting will resume on the second qualified source edge after the retriggering Gate edge. Qualified source edges are active-going edges which occur while the Gate is active.

MODE O

Software-Triggered Strobe with Edge Gating and Hardware Retriggering

Mode O is similar to Mode N, except that counting will not begin until an active-going Gate edge is applied to an armed counter and the Gate level is not used to modulate counting. The counter must be armed before application of the triggering Gate edge; Gate edges applied to a disarmed counter are disregarded. Irrespective of the Gate level, the counter will count all source edges after the triggering Gate edge until the first TC. On the first TC the counter will be reloaded from the Load register and disarmed. A new ARM command and a new Gate edge must be applied in that order to initiate a new counting cycle. Unlike Modes C, F, I and L, which disregard the Gate input once counting starts, in Mode O the count process will be retriggered on all active-going Gate edges, including the first Gate edge used to start the counter. On each retriggering Gate edge, the counter contents will be transferred into the Hold register. On the first source edge after the retriggering Gate edge the Load register contents will be transferred into the counter. Counting will resume on the second source edge after a retrigger.

MODE Q

Rate Generator with Synchronization
(Event Counter with Auto-Read/Reset)

Mode Q provides a rate generator with synchronization or an event counter with auto-read/reset. The counter must first be issued an ARM command before counting can occur. Once armed, the counter will count all source edges which occur while the Gate is active and disregard those edges which occur while the Gate is inactive. This permits the Gate to turn the count process on and off. After the issuance of an ARM command and the application of an active Gate, the counter will count to TC repetitively. On each TC the counter will reload itself from the Load register. The counter may be retriggered at any time by presenting an active-going Gate edge to the Gate input. The retriggering Gate edge will transfer the contents of the counter into the Hold register. The first qualified source edge after the retriggering Gate edge will transfer the contents of the Load register into the counter. Counting will resume on the second qualified source edge after the retriggering gate edge. Qualified source edges are active-going edges which occur while the Gate is active.

TABLE 3-10. OPERATING MODES (continued)

MODE R

Retriggerable One-Shot

Mode R is similar to Mode Q, except that edge gating rather than level gating is used. In other words, rather than use the Gate level to qualify which source edges to count, Gate edges are used to start the counting operation. The counter must be armed before application of the triggering Gate edge; Gate edges applied to a disarmed counter are disregarded. After application at a Gate edge, an armed counter will count all source edges until TC, irrespective of the Gate level. On the first TC the counter will be reloaded from the Load register and stopped. Subsequent counting will not occur until a new Gate edge is applied. All Gate edges applied to the counter, including the first used to trigger counting, initiate a retrigger operation. Upon application of a Gate edge, the counter contents are saved in the Hold register. On the first source edge after the retriggering Gate edge, the Load register contents will be transferred into the counter. Counting will resume on the second source edge after the retriggering Gate edge.

MODE S

In this mode, the reload source for LOAD commands (irrespective of whether the counter is armed or disarmed) and for TC-initiated reloads is determined by the Gate input. The Gate input in Mode S is used only to select the reload source, not to start or modulate counting. When the Gate is Low, the Load register is used; when the Gate is High, the Hold register is used. Note the Low-Load, High-Hold mnemonic convention. Once armed, the counter will count to TC twice and then disarm itself. On each TC the counter will be reloaded from the reload source selected by the Gate. Following the second TC, an ARM command is required to start a new counting cycle.

MODE V

Frequency-Shift Keying

Mode V provides frequency-shift keying modulation capability. Gate operation in this mode is identical to that in Mode S. If the Gate is Low, a LOAD command or a TC-induced reload will reload the counter from the Load register. If the Gate is High, LOADs and reloads will occur from the Hold register. The polarity of the Gate only selects the reload source; it does not start or modulate counting. Once armed, the counter will count repetitively to TC. On each TC the counter will reload itself from the register determined by the polarity of the Gate. Counting will continue in this manner until a DISARM command is issued to the counter. Frequency shift keying may be obtained by specifying a TC Toggled output mode in the Counter Mode register. The switching of frequencies is achieved by modulating the Gate.

toggle on each count. Execution of a Load, Load-and-Arm or Step command with these counter contents will act the same as application of a source pulse, causing TC to remain active and a TC Toggled output to toggle.

3-70. Alarm Registers and Comparators

Added funcitons are available in the Counter Logic Groups for counters 1 and 2. Each contains a 16-bit Alarm register and a 16-bit Comparator. When the value in the counter reaches the value in the Alarm register, the Comparator output will go true. The Master Mode register contains control bits to individually enable/disable the comparators. When enabled, the comparator output appears on the OUT pin of the associated counter in place of the normal counter output. The output will remain true as long as the comparison is true, that is, until the next input causes the count to change. The Comparator output is subject to the same polarity control logic as the counter out.

In some applications, values are being accumulated and recognition of a particular count event is desired. This might be accomplished by preloading the counter with the desired value and then counting it down toward zero.

Alternatively, the counter can be initialized with zero and the desired value entered into the Alarm register.

With the Comparator enabled, an OUT transition will then occur when the value is reached, plus counts will continue to accumulate.

3-71. Output Control Bit

The output control circuitry for each of the five counter groups includes a toggle flip-flop that can be used to generate various output waveforms. TC from the counter is the toggling signal and the output frequency can thus be half the TC frequency. The toggle can be initialized to either high or low state by command. More details of the toggle activity are included in the Counter Mode Register description.

CHAPTER 4
THEORY OF OPERATION

4-1. INTRODUCTION

This chapter provides a detailed functional description of the Am95/4006 MonoBoard Computer. The MonoBoard Computer consists of the functional blocks shown in figure 4-1.

Both active-high (positive true) and active-low (negative true) signals appear on the schematics. To eliminate confusion, and simplify presentation, the following convention is adhered to within this manual: whenever a signal is active-low (negative true), its mnemonic is followed by an asterisk (*). For example, MEMR* denotes an active-low signal. When a signal is active-high, the asterisk is omitted.

4-2. CENTRAL PROCESSING UNIT (CPU) GROUP

The CPU group consists of an Am9080A CPU, an Am8224 Clock Generator, an Am8238 High Speed System Controller and associated control ciruits.

The Am8224 Clock Generator (U35) and an 18MHz oscillator provide the primary timing reference for 2MHz board operation. The 18MHz oscillator establishes the frequency of oscillation for the Am8224 via a .01 μF capacitor. The Am8224 divides the oscillator frequency by nine to produce the 2MHz phase one and phase two timing inputs to the CPU. A TTL level phase two TTL signal is also derived and made available for TTL gating. All processing activities are referenced to the phase one and phase two clocks.

The 4MHz MonoBoard contains a 36MHz oscillator which is divided by nine to produce the 4MHz phase one and phase two timing inputs for the CPU.

A power up reset circuit provides a reset to the Am8224 upon power turn-on. The Am8224 in turn provides a reset (RST) output to the CPU and other programmable devices on the board. The reset output also generates the INIT* signal that is used to initialize other boards.

A Ready In (RDYIN) input to the Am8224 is gated from the on-board and off-board ready and acknowledge outputs. When an addressed device responds with its ready or acknowledge signal, it is gated to the RDYIN input of the Am8224. The RDYIN input provides the D input for an internal flip-flop. After RDYIN goes high, the leading edge of the next phase two clock pulse clocks the flip-flop to produce a high READY signal to the CPU. A failsafe timer circuit produces a TIME OUT* signal that is also gated to the Am8224 RYDIN input. The failsafe timer is designed to prevent hanging up the CPU if memory or a device does not respond within approximately 1ms.

NOTE

It is recommended that the failsafe timer be used to generate a timeout interrupt to inform the program that the operation has not completed. Otherwise, invalid data could cause the program to go to unknown locations.

The failsafe timer signal goes high when a one-shot is triggered by the Status Strobe signal (STSTB*) from the Am8224 at the beginning of each machine cycle. If the one-shot is not retriggered within approximately 25ms, its output goes low and is gated through to the RDYIN pin on the Am8224, therby driving the READY line

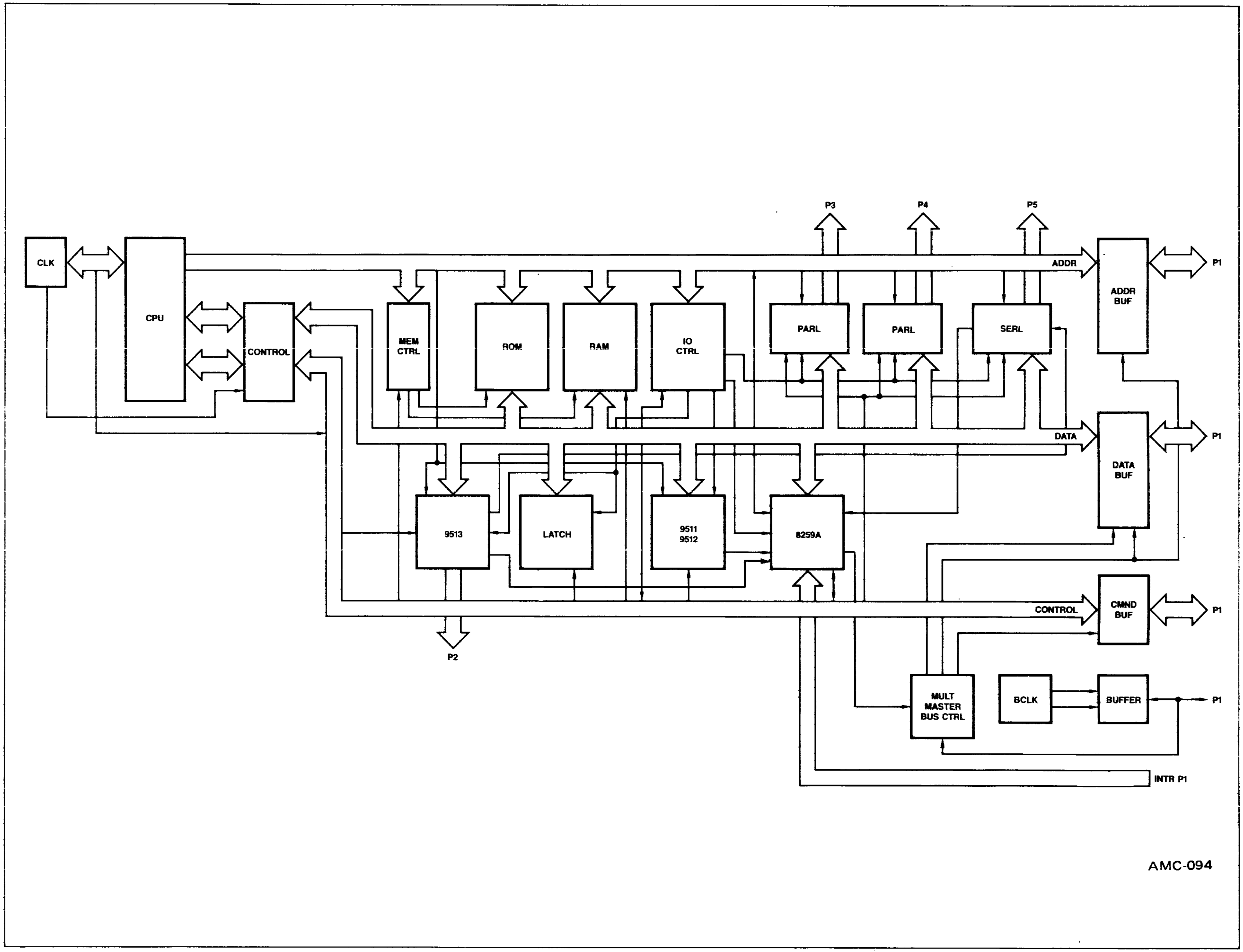


Figure 4-1. Am95/4006 MonoBoard Computer Block Diagram.

high and allowing the CPU to exit the wait states.

Three groups of signals are provided by the Am9080A CPU to transmit data and controls between the CPU, memory, and peripheral devices. An 8-bit parallel data bus transmits instructions, data, and status to and from the CPU. A 16-bit address bus identifies any of 65,536 (64K) bytes of memory location during a memory access operation. The address bus also addresses peripheral devices during I/O read or write instructions. The third group of signals includes ten control signals to synchronize CPU operations with memory and peripheral devices.

Within the CPU, an instruction cycle is defined as the time required to fetch and execute an instruction. During the fetch, a selected instruction is extracted from memory and stored in the CPU operating registers. During the execution part, the instruction is decoded and translated into specific processing activities.

The instruction cycle consists of from one to five machine cycles. A machine cycle is required each time the CPU accesses memory or an I/O port. The fetch portion of an instruction cycle requires one machine cycle for each byte to be fetched. The number of machine cycles required for instruction execution depends on the kind of instruction that has been fetched. Some instructions do not require additional machine cycles for execution; others require additional cycles to write or read data to/from memory or I/O devices. An instruction could require from one to five machine cycles.

Each machine cycle consists of from three to five clock pulses. The clock pulses are identified as states (T1 through T5). A state is identified as the interval between two successive positive-going transitions of the phase one clock.

In summary, a clock period defines a state. Three to five states constitute a machine cycle, and one to five machine cycles comprise an instruction cycle.

There are three exceptions to the defined duration of a state. These are the Wait state, the Hold state, and the Halt state. Because the duration of these states depends upon external events, they are of indeterminate length. These states must be synchronized with the phase one clock pulses. Their duration is therefore an integral multiple of the phase one clock.

At the beginning of each machine cycle (in state T1), the CPU activates its SYNC output and issues status information on its data bus. The Am8224 accepts SYNC and generates an active-low status strobe (STSTB*) signal to the Am8238 and the failsafe timer. The status information indicates the type of machine cycle in progress.

The rising edge of the phase two clock during T1 loads of the address lines of the processor. These lines become stable during the phase two clock and remain stable until the first phase two pulse after state T3. This gives the CPU time to read the data returned from memory. Once the processor has sent an address to memory, the processor samples the READY input. If the READY input is low, the processor will idle. This gives the memory time to respond to the addressed data request. The CPU remains in the idle condition until its READY line again goes high. When the READY line does high, the cycle proceeds, beginning with the rising edge of the next phase one clock.

The events that occur during the T3 through T5 states are determined by the machine cycle in progress. During a fetch memory cycle the CPU interprets the data on the data bus as a data word. The CPU outputs data on the data bus during a memory write.

During an I/O operation the CPU can either receive or transmit data, depending upon whether the machine cycle is an input or output operation.

An Am8238 eight bit, bidirectional bus driver, buffers the data bus from memory and devices. At the beginning of each machine cycle the CPU issues status on the data bus that indicates the type of function that will occur during the cycle. The Am8238 stores this information in a status latch when the STSTB input from the Am8224 goes low. The outputs of the status latch are connected to a gating array which is used for control signal generation. The gating array generates control signals MEMW*, MEMR*, IOW*, IOR*, and INTA* by gating the status latch with the DBIN, WR*, and HLDA signals from the CPU.

4-3. MULTI-MASTER BUS CONTROL LOGIC

Multi-Master bus operation is made possible by the bus control logic shown on sheet 6 of the schematics. The Am95/4006 MonoBoard Computer can be configured for serial parallel bus priority resolution. As shipped, a jumper is connected between jumper pins 86 and 85 to configure the board for serial priority operation. Therefore, the BPRO* signal is low when no higher priority master (including this master) is using the bus.

When an off-board memory or I/O operation is to take place, the XFER REQ signal is generated by the PROM at U54 to initiate a Bus Request. The flip-flop at U55 pin 14 is set, which in turn generates the Common Bus Request (CBRQ*) signals.

When no higher priority master is requesting the bus (BPRN* high) and no master has control of the bus (BUSY* high), the flip-flop at U58 pin 14 is set, which causes the CBRQ* signal to go high. One BCLK* clock cycle later, the flip-flop at U55 pin 2 is set to turn on the bus command drivers. The Am95/4006 will relinquish control of the bus after each bus transfer.

4-4. ON-BOARD MEMORY

MonoBoard memory addressing is controlled by a mapping PROM (U40) and the 74LS139 decoder at U41. Address bits 0 through 9 specify a single storage location within each memory device. The decoder output determines which memory chips are enabled.

Inputs to the memory select PROM can be program and jumper selected, as described in chapters 2 and 3. A customized PROM might also be inserted to satisfy unique application requirements. PROM output bit 3 is used to enable the decoder to select a pair of RAM memory chips. PROM output bit 2 is used to enable the ROM memory chip select decoder. When either the RAM or ROM is selected, the resultant RAMSEL* or ROMSEL* is used to generate the MEMSEL* to indicate that an on-board memory read or write operation is in progress.

When this master is already in control of the bus, data and addresses are placed on the bus during an on-board or off-board device selection. When the MEMSEL* signal is not present during a memory read operation, or the IORDY* signal is not present during an I/O read operation, an off-board memory read or off-board I/O read is taking place; the DBOUT* signal causes the bus drivers to read into the board. During a read or write to an on-board device, bus master logic determines whether data and address information is placed onto the bus, but read and write control signals are not placed on the bus.

When this master is not already in control of the bus during an on-board memory reference or on-board I/O operation, the Am8226 address and data bus drivers are held in the high impedance state by the high BUSEN* signal. The control drivers are placed in the high impedence state by the CMEN* signal. Therefore, although the direction control for the data bus drivers is out, the data bus driver/receivers are disabled during an on-board memory reference.

The memory acknowledge (MEMACK*) signal is input to the Am8224 control circuit, which in turn generates the READY signal to the Am9080 CPU to indicate that the memory access is done.

4-5. READ ONLY MEMORY (ROM/E-PROM)

The Read Only Memory (ROM/E-PROM) section provides for installing up to 16K bytes of ROM using four Am9208, Am9218, or Am9233 compatable ROMs or four Am9708, Am9716, Am9732, or 2758 compatable E-PROMs. Jumpers are used to customize the memory socket voltages to the ROM/E-PROM being used as shown in table 2-7.

Memory addressing is controlled by a mapping PROM at U40 and the 74LS139 decoder at U41. Address bits 0 through 11 specify a single storage location within each of the four ROM/EPROM devices; the decoder output selects one of the four chip locations.

4-6. RANDOM ACCESS MEMORY (RAM)

The Random Access Memory (RAM) section provides the user with 4096 (4K) by 8-bits of read/write storage. The RAM memory consists of eight Am9114 static RAM chips which store 1024 by 4-bits each. Address decoding for chip selection is accomplished by using Am25LS139 one-of-four decoder and assorted control gates.

Each Am9114 chip has 10 address inputs (0 through 9) and four common data input/output pins I/01 through I/04, as well as active low chip select CS* and write enable WE* inputs. The data input/output pins I/01 through I/04 from two Am9114 chips are connected to the data bus as bits DB0 through DB7.

When on-board RAM is accessed, the ten least significant bits (0 through 9) of the address bus specify the 4-bit segment to be accessed on the selected RAMs. Address bits 10 and 11 specify which pair of RAMs are to be selected. Access to the RAMs is controlled by bit 3 from the memory select PROM at U40. RAM read/write control is provided by the ADVMW* signal.

4-7. I/O ADDRESS DECODING

The Am95/4006 includes six programmable devices and one 8-bit latch (U42) that are accessed by I/O commands.

- Two Am8255 Programmable Peripheral Interfaces
- An Am9551 Programmable Communications Interface
- An 8259A Programmable Interrupt Controller
- An Optional Am9512 Floating-Point Processor
- An Am9511 Arithmetic Processing Unit or an Am9512 Floating Point Processor
- An Am9513 System Timing Controller

The I/O Address Decode logic consists of a 256 by 4-bit PROM (U33) and an Am27LS138 one-of-eight decoder (U34). The PROM is programmed with the bit pattern required by the Am25LS138 to select one of eight active low outputs. Seven of these outputs select the six programmable and the latch devices. One output is not used and is never selected. Table D-5 in Appendix D shows the bit pattern to memory location relatonship for the I/O addresses.

4-8. SERIAL I/O INTERFACE

The Serial I/O Interface provides the MonoBoard Computer with a serial data communications channel that can be programmed to operate with most of the current serial data transmission protocols: synchronous or asynchronous. Character length, number of stop bits, and even/odd parity are program selectable. Baud rate is software controlled. The serial I/O Interface consists of an Am9551 Programmable Communications Interface and driver/ receiver circuits.

The least significant address bit A0 is applied to the C/D input of the Am9551. An output instruction (IOW* true, CS51* true, and C/D high) causes the Am9551 to accept a control byte through its data bus pins. The control byte can be either a mode instruction or a command instruction, depending on the sequence in which it is sent. The mode instruction specifies the baud rate multiplier, character length, parity, and the number of stop bits. The actual baud rate selection is determined by the baud rate input from the Am9513 and the baud rate factor selected in the Am9551. The command word instructs the Am9551 to enable/disable the receiver and transmitter, to reset errors, to return to Idle mode, and to set/clear the Data Terminal Ready Signal output. An output instruction also causes the Am9551 to accept a data byte through its data bus pins. Bit 0 is the least significant bit and bit 7 is the most significant bit. The Am9551 will subsequently transmit serial data to an external device if the transmitter is enabled.

An input instruction (IOR* true, CS51* true, and C/D high) causes the Am9551 to place a status byte on the data bus. The status bits are the result of status and error checking functions performed within the Am9551. An input instruction also causes the Am9551 to output a data byte onto the data bus. Bit 0 is the least significant bit and bit 7 is the most significant bit.

Timing for the Am9551 internal functions is provided by the FOUT output of the Am9513. The TXC and RXC signals can also be supplied externally.

A high on the Am9551 reset (RST) line forces the Am9551 into an idle mode. After a high RST input, the device remains idle until a new set of control words are written into the Am9551 to define its function.

In addition to the above control lines, the Am9551 also has a set of control inputs and outputs that can be used to simplify the interface to almost any serial data device. These control signals are general purpose in nature.

4-9. PARALLEL I/O INTERFACE

The Parallel I/O Interface (U20 and U21) on the MonoBoard provides 48 lines for the transfer and control of data to and from peripheral devices. Sixteen lines have bidirectional driver/terminator networks installed. The remaining 32 lines are uncommitted. Sockets are provided for installing driver/terminator networks in groups of four lines per network.

The two 8255As are operationally identical except for addressing.

Each 8255A contains three 8-bit ports (A, B, and C). All ports can be configured by software for a variety of functions. Each port has the following special characteristics.

Port A: Provides an output latch, an input latch, an input buffer or a bidirectional bus for eight data bits.

Port B: Provides an output latch, an input latch or an input buffer for eight data bits.

Port C: Operates as an output latch for eight bits, an input buffer for eight bits, or as two four bit control ports for ports A and B.

Communication between the CPU and a 8255A is via the data bus and six control lines. Control bytes and data bytes are transmitted to a 8255A; status bytes and data bytes are transmitted from a 8255A on the data bus. The six control lines provide the necessary controls for all data bus operations.

The chip select (SC55A* and CS55B*) to the CS* inputs allow the chips to be individually addressed. When CS* is true (logical 0), the 8255As accept or output data (or control bytes) from the data bus. The direction of data flow is determined by the RD and WR (IOR* and IOW*) inputs. A low (true) IOR* allows the CPU to read data or status from a 8255A. A low (true) on the IOW* allows the CPU to write data or control words to a 8255A.

Address lines 0 and 1 allow selection, by the CPU, of a specific port or control word register. When the chip select (CS*) is true and IOR* or IOW* is true, the CPU can read or write to the ports or control registers identified by 0 and 1 as follows:

A1	A0	SELECTION
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Control Register

All internal registers are cleared and the ports are set to the high impedance input mode when a high level is presented to the Reset input.

4-10. INTERRUPT CONTROLLER

The interrupt controller logic consists of an 8259A Interrupt Controller and a jumper pad that allows the user to connect any of 16 possible interrupt requests to the 8259A eight interrupt inputs.

The 8259A resolves priorities among the eight levels. Priority resolution can be changed at any time. Operation of the 8259A is controlled by five control lines and the data bus. The five control lines are decoded to provide controls for programming and reading status. Control words and status information are transferred through the data bus.

The 8259A must be programmed as a master in the buffered mode. The onboard 8259A may also be programmed to have off-board slave 8259s. If off-board slaves are to be used, the Am95/4006 must be configured for bus-vectored interrupts. Refer to Chapter 3 for configuration information.

When the 8259A is operated without slave inputs, the following sequence occurs to process interrupts:

1. 8259A INT output to the processor goes high.

2. The INTA* input to 8259A goes low, then the 8259A SP/EN output goes low and the 8259A outputs a CALL (CDH) on the data bus.

3. When the INTA* line goes high, the processor reads the data bus and forces two more INTA* pulses to the 8259A.

4. The two INTA* pulses cause the 8259A to output the address of the interrupt subroutine, low byte first. Each time INTA* goes low, the SP/EN output goes low.

When the on-board 8259A is operated as master with slave inputs from offboard 8259A devices, operation is similar. The main difference is that the subroutine addresses are obtained from the off-board slave devices via the Multibus. In order for the processor to take over the bus, the Am95/4006 must be configured for bus- vectored interrupts. All three INTA* go onto Multibus. The first is answered by the master placing a CALL on the data bus. The last two INTA* pulses are answered by the slave outputting address of the interrupt service routine to the Multibus.

4-11. ARITHMETIC PROCESSING UNIT (APU)

An optional Am9511 Arithmetic Processing Unit (APU) performs both single (16-bit) and double (32-bit) precision fixedpoint arithmetic as well as floating-point addition, subtraction, multiplication, and division. Transcendental derived functions and data manipulation and conversion commands can also be executed by the APU. The following paragraphs are intended to familiarize the user with the basic theory of the Am9511 operation.

Operands and commands are written into the APU, and results and status are read from the APU via the data bus. Four control signals identify these operations to the APU. The Chip Select (CS) input to the APU selects a read or write operation; the RD and WR inputs define the direction of data flow to (ADVIOW* true) or from (IOR* true) the APU. Data bus information is defined as data (C/D low and ADVIOW* low) command (C/D low and IOR* low) by the C/D input.

Data (operands) are stored in the Am9551 in an eight level 16-bit wide data stack. Since single precision fixed-point operations are 16-bits wide, eight such values can be concurrently maintained in the stack. When using either double precision fixed-point or floating-point formats, four values can be concurrently stored. Data is written into the stack eight bits at a time. The least significant byte is written first. Data is removed from the stack in the reverse byte order. Data must be entered onto the stack in multiples of the number of bytes appropriate to the chosen format.

Data is removed from the stack in the reverse order of entry. That is, the first byte in is the last byte out.

The removal of each data word redefines the top of stack (TOS) so that the next successive byte to be removed becomes TOS. Data removed from the stack rotates to the bottom of the stack.

During data bus operations, an active low output (PAUSE) from the APU to the CPU indicates the APU has not completed its information transfer over the data bus. Whenever a data read or status read operation is requested, PAUSE goes low. It returns high only after the data bus contains valid data. When an existing command is in the process of execution, any read or write request will cause the PAUSE output to go low for the remaining duration of the command plus any time needed for initiating a data bus operation.

When any command has completed execution, an active low (open drain) output (END) indicates that execution of the previous entered command is complete. The END output is applied to the interrupt controller jumper matrix as INT11*.

4-12. FLOATING-POINT PROCESSOR

An optional Am9512 Floating-Point Processor performs single (32 bits) and double (64 bits) precision add, subtract, multiply, and divide floating-point operations. In addition, the Am9512 is capable of changing the sign of a single or double precision operand at the Top-Of-Stack (TOS) and Next-On-Stack (NOS), exchanging single or double precision operand located at TOS and NOS, as well as copying and popping single or double precision operands.

Data transfers between the Am9512 and the CPU are handled using Programmed I/O instructions. Data transfers within the Am9512 are over an 8 bit bidirectional data bus. Operands are pushed onto an 8x17 LIFO stack and a command is issued to perform an operation on the stack. The results of this operation are retrieved by popping the stack. After completing an operation, the Am9512 activates an End-Of-Execution (EOE) signal that interrupts the CPU. The results from an operation are the same precision and format as the operands, and are rounded to perserve the accuracy.

To perform a write operation, data is presented on DBO theough DB7 lines, logic level on the C/D input and the CS input is made LOW. Whenever WR and RD inputs are both HIGH and CS is LOW, PAUSE goes LOW. However actual writing into the Am9512 cannot start until WR is made LOW. After initiating the write operation by the HIGH to LOW transition on the WR input, the PAUSE output will go HIGH indicating the write operation has been acknowledged. The WR input can go HIGH after PAUSE goes HIGH. The data lines, C/D input and the CS input can change when the hold time requirements are satisfied.

To perform a read operation, the logic level is established on the C/D input and CS is made LOW. The PAUSE output goes LOW because WR and RD inputs are HIGH. The read operation does not start until the RD input goes LOW. PAUSE will go HIGH indicating that read operation is complete and the required information is available on the DBO through DB7 lines. This information will remain on the data lines as long as RD is low. The RD input can return HIGH anytime after PAUSE goes HIGH. The CS input and C/D input can change anytime after RD returns HIGH. If the CS is tied LOW permanently, PAUSE will remain LOW until the next Am9512 read or write access.

4-13. SYSTEM TIMING CONTROLLER

The Am9513 system timing controller provides the counting, sequencing, and timing functions for the Am95/4006.

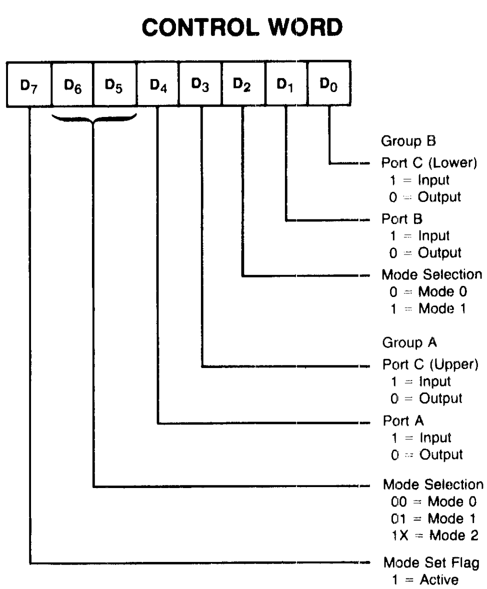
It is accessed by I/O port addresses D8H and D9H, and the IORC* and IOWC* signals are used to control timer data input/output. Addresses D8H and D9H are decoded by the address decode PROM (U33) to produce the Chip Select (CS13*) signal for the Am9513 system timing controller at U72. Address bit A0 is connected to the C/D input to select a data or control operation. The D0 through D7 inputs to the Am9513 are connected to the data bus through Bus Drivers U68 and U69. An external crystal is connected to the X2 input to control the frequency of the Am9513 internal oscillator. The IN, GATE, OUT, and FREQ OUT (FOUT) signals from the Am9513 are available to the user on connector P2.

APPENDIX A
COMMAND SUMMARIES

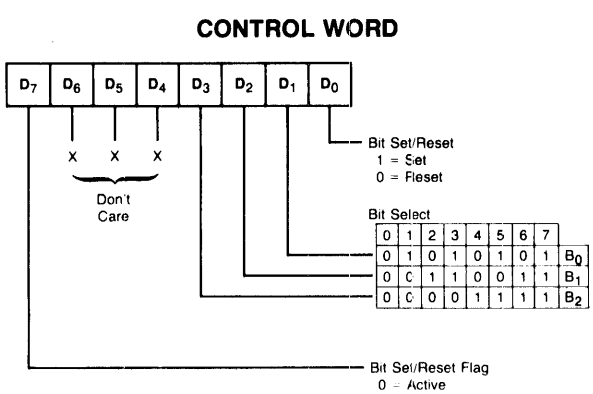
A-1. INTRODUCTION

The following are command summaries for the programmable devices in the Am95/4006.

A-2. Am8255 PROGRAMMABLE PERIPHERAL INTERFACE

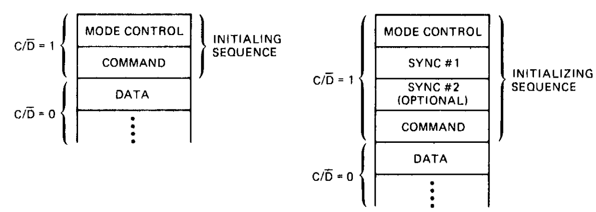


Mode Definition Format

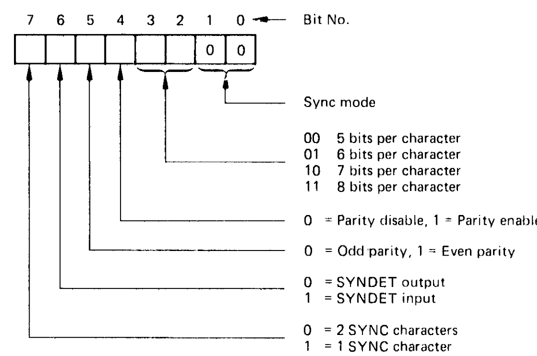


Bit Set/Reset Format

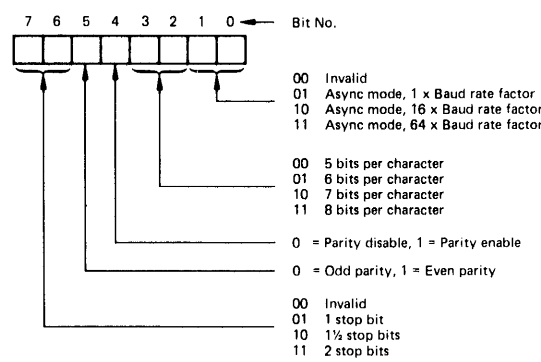
A-3. Am9551 PROGRAMMABLE COMMUNICATIONS INTERFACE



Control Word Sequence for Initialization



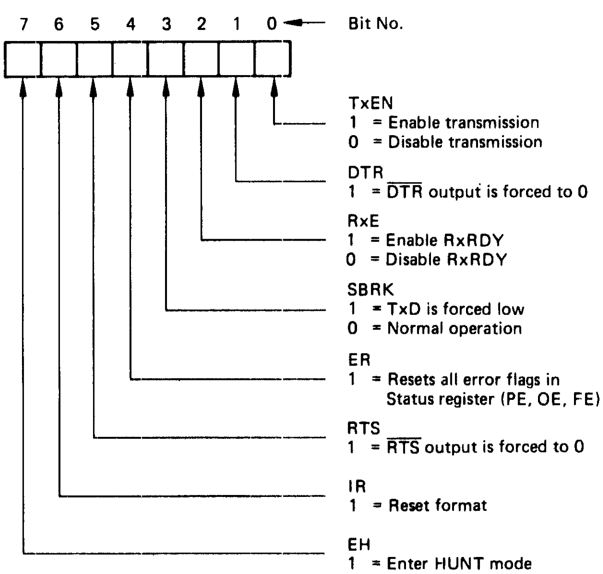
Synchronous Mode Control Code



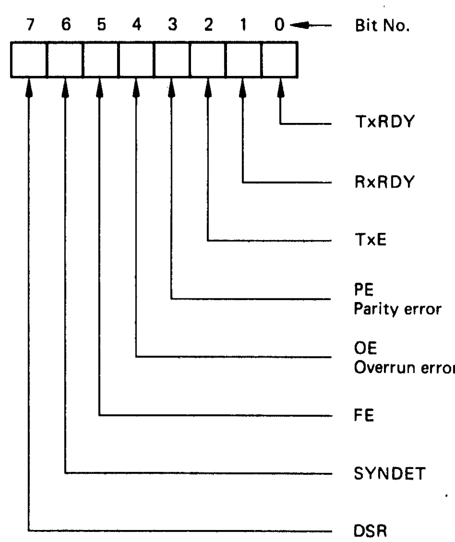
Asynchronous Mode Control Code

TxEN	TxE	TxRDY	
1	1	1	Transmit Output Register and Transmit Character Buffer empty. TxD continues to mark if Am9551 is in the asynchronous mode. TxD will send Sync pattern if Am9551 is in the Synchronous Mode. Data can be entered into Buffer.
1	0	1	Transmit Output Register is shifting a character. Transmit Character Buffer from the processor.
1	1	0	Transmitt Register had 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.

Operation of the Transmitter Section as a Function of TxE, TxRDY and TxEN



The Am9551 Status Register



Am9551 Control Command

A-4. Am9511 ARITHMETIC PROCESSING UNIT

COMMAND CODE

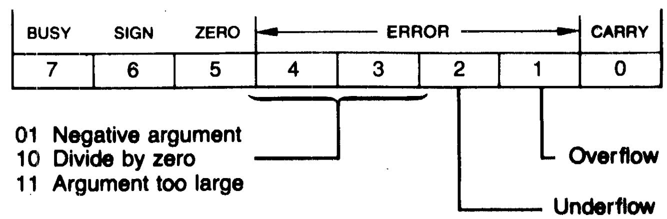
MNEMONIC	SERVICE REQUEST (HEX OPCODE)	NO SERVICE REQUEST (HEX OPCODE)	CYCLES
FIXED POINT 16-BIT			
SADD	EC	6C	17
SSUB	ED	6D	31
SMUL	EE	6E	88
SDIV	EF	6F	89
FIXED POINT 32-BIT			
DADD	AC	2C	21
DSUB	AD	2D	40
DMUL	AE	2E	194-210
DMUU	B6	36	182-218
DDIV	AF	2F	196-210
FLOATING POINT 32-BIT			
FADD	90	10	54-368
FSUB	91	11	70-370
FMUL	92	12	146-168
FDIV	93	13	154-184
SQRT	81	01	800 *
SIN	82	02	4464 *
COS	83	03	4118 *
TAN	84	04	5754 *
ASIN	85	05	7668 *
ACOS	86	06	7734 *
ATAN	87	07	6006 *
LOG	88	08	4490 *
LN	89	09	4478 *
EXP	8A	0A	4616 *
PWR	8B	0B	9292 *
DATA MANIPULATION			
NOP	80	00	4
FIXS	9F	1F	90-214
FIXD	9E	1E	90-336
FLTS	9D	1D	62-156
FLTD	9C	1C	56-342
CHSS	F4	74	23 *
CHSD	B4	34	27 *
CHSF	95	15	18 *
PTOS	F7	77	16
PTOD	B7	37	20
PTOF	97	17	20
POPS	F8	78	10
POPD	B8	38	12
POPF	98	18	12
XCHS	F9	79	18
XCHD	B9	39	26
XCHF	99	19	26
PUPI	9A	1A	16

*Weighted average execution cycle

PORT ADDRESSING

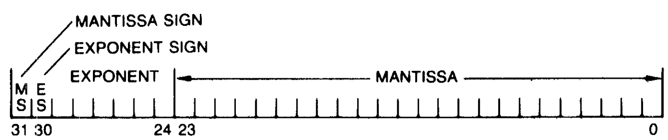
C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	Operation
0	1	0	Enter data byte into stack
0	0	1	Read data byte from stack
1	1	0	Enter command
1	0	1	Read status

STATUS REGISTER



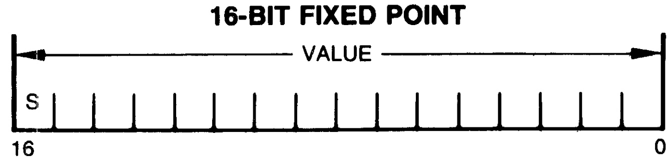
DATA FORMATS

32-BIT FLOATING POINT



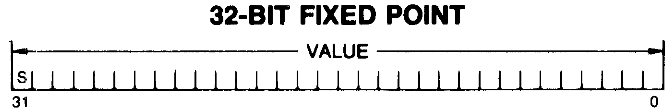
SIGN-MAGNITUDE MANTISSA
UNBIASED TWO'S COMPLEMENT EXPONENT

16-BIT FIXED POINT



TWO'S COMPLEMENT

32-BIT FIXED POINT



TWO'S COMPLEMENT

A-5. Am9512 FLOATING-POINT PROCESSOR

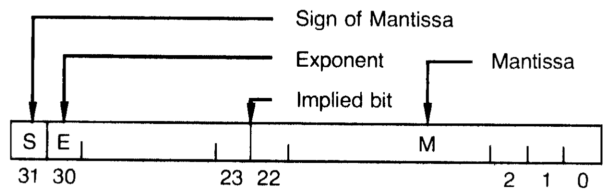
Command Code			
Mnemonic	Service Request	No Service Request	Cycles
	Hex Opcode		
Floating-Point 32-Bit			
SADD	81	01	58
SDIV	84	04	228
SMUL	83	03	198
SSUB	82	02	56
Floating-Point 64-Bit			
DADD	A9	29	578
DDIV	AC	2C	4560
DMUL	AB	2B	1748
DSUB	AA	2A	578
Data Manipulation (32-Bit)			
CHSS	85	05	10
CLR	80	00	4
POPS	87	07	14
PTOS	86	06	16
Data Manipulation (64-Bit)			
CHSD	AD	2D	24
CLR	80	00	4
POPD	AF	2F	26
PTOD	AE	2E	40

PORT ADDRESSING

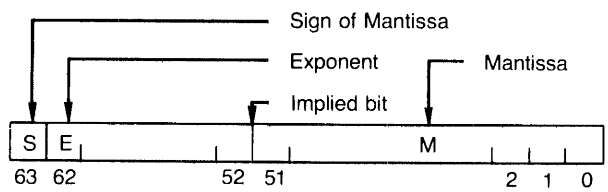
$C/\overline{D}$	$\overline{RD}$	$\overline{WR}$	Function
L	H	L	Push data byte into the stack
L	L	H	Pop data byte from the stack
H	H	L	Enter command
H	L	H	Read Status
X	L	L	Undefined

L = Low
H = High
X = Don't Care

32-BIT FLOATING POINT



64-BIT FLOATING-POINT



STATUS REGISTER

Busy	Sign S	Zero Z	Reserved	Divide Exception D	Exponent Underflow U	Exponent Overflow V	Reserved
7	6	5	4	3	2	1	0

APPENDIX B
Am9080A INSTRUCTION SET SUMMARY

B-1. INTRODUCTION

This appendix summarizes all the instruction opcodes for the Am9080A Microprocessor. Instructions are listed by functional categories, in ascending order of hex opcode value, and alphabetically by mnemonic.

B-2. INSTRUCTION SUMMARY BY FUNCTIONAL ORDER

Data Transfer

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
40	MOV B,B	58	MOV E,B	70	MOV M,B	1A	LDAX D
41	MOV B,C	59	MOV E,C	71	MOV M,C	2A	1HLD
42	MOV B,D	5A	MOV E,D	72	MOV M,D	3A	LDA
43	MOV B,E	5B	MOV E,E	73	MOV M,E	02	STAX B
44	MOV B,H	5C	MOV E,H	74	MOV M,H	12	STAX D
45	MOV B,L	5D	MOV E,L	75	MOV M,L	22	SHLD
46	MOV B,M	5E	MOV E,M	77	MOV M,A	32	STA
47	MOV B,A	5F	MOV E,A	78	MOV A,B	01	LXI B
48	MOV C,B	60	MOV H,B	79	MOV A,E	11	LXI D
49	MOV C,C	61	MOV H,C	7A	MOV A,D	21	LXI H
4A	MOV C,D	62	MOV H,D	7B	MOV A,E	31	LXI SP
4B	MOV C,E	63	MOV H,E	7C	MOV A,H	F9	SPHL
4C	MOV C,H	64	MOV H,H	7D	MOV A,L	E3	XTHL
4D	MOV C,L	65	MOV H,L	7E	MOV A,M	EB	XCHG
4E	MOV C,M	66	MOV H,M	7F	MOV A,A	D3	OUT
4F	MOV C,A	67	MOV H,A	06	MVI B	DB	IN
50	MOV D,B	68	MOV L,B	0E	MVI C	C5	PUSH B
51	MOV D,C	69	MOV L,C	16	MVI D	D5	PUSH D
52	MOV D,D	6A	MOV L,D	IE	MVI E	E5	PUSH H
53	MOV D,E	6B	MOV L,E	26	MVI H	F5	PUSH PSW
54	MOV D,H	60	MOV L,H	2E	MVI L	C1	POP B
55	MOV D,L	6D	MOV L,L	36	MVI M	D1	POP D
56	MOV D,M	6E	MOV L,M	3E	MVI A	E1	POP H
57	MOV D,A	6F	MOV L,A	0A	LDAX B	F1	POP PSW

Arithmetic

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
80	ADD B	C6	ADI	9E	SBB M	3C	INR A
81	ADD C	CE	ACI	9F	SBB A	03	INX B
82	ADD D	90	SUB B	D6	SUI	13	INX D
83	ADD E	91	SUB C	DE	SBI	23	INX H
84	ADD H	92	SUB D	09	DAD B	33	INX SP
85	ADD L	93	SUB E	19	DAD D	05	DCR B
86	ADD M	94	SUB H	29	DAD H	0D	DCR C
87	ADD A	95	SUB L	39	DAD SP	15	DCR D
88	ADC B	96	SUB M	27	DAA	ID	DCR E
89	ADC C	97	SUB A	04	INR B	25	DCR H
8A	ADC D	98	SBB B	0C	INR C	2D	DCR L
8B	ADC E	99	SBB C	14	INR D	35	DCR M
8C	ADC H	9A	SBB D	1C	INR E	3D	DCR A
8D	ADC L	9B	SBB E	24	INR H	0B	DCX B
8E	ADC M	9C	SBB H	2C	INR L	1B	DCX D
8F	ADC A	9D	SBB L	34	INR M	2B	DCX H
						3B	DCX SP

Logical

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
A0	ANA B	A9	XRA C	B2	ORA D	BB	CMP E
A1	ANA C	AA	XRA D	B3	ORA E	BC	CMP H
A2	ANA D	AB	XRA E	B4	ORA H	BD	CMP L
A3	ANA E	AC	XRA H	B5	ORA L	BE	CMP M
A4	ANA H	AD	XRA L	B6	ORA M	BF	CMP A
A5	ANA L	AE	XRA M	B7	ORA A	FE	CPI
A6	ANA M	AF	XRA A	F6	ORI	07	RLC
A7	ANA A	EE	XRI	BB	CMP B	0F	RRC
E6	ANI	B0	ORA B	B9	CMP C	17	RAL
A8	XRA B	B1	ORA C	BA	CMP D	1F	RAR
						2F	CMA

Branching

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
C3	JMP	D7	RST 2	EC	CPE
C2	JNZ	DF	RST 3	F4	CP
CA	JZ	E7	RST 4	FC	CM
D2	JNC	EF	RST 5	C9	RET
DA	JC	F7	RST 6	C0	RNZ
E2	JPO	FF	RST 7	C8	RZ
EA	JPE	CD	CALL	D0	RNC
F2	JP	C4	CNZ	D8	RC
FA	JM	CC	CZ	E0	RPO
E9	PCHL	D4	CNC	E8	RPE
C7	RST 0	DC	CC	F0	RP
CF	RST 1	E4	CPO	F8	RM

Control

Hex	Mnemonic
00	NOP
76	HLT
F3	DI
FB	EI
37	STC
3F	CMC

B-3. INSTRUCTION SUMMARY IN HEXADECIMAL ORDER

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
00	NOP	37	STC	69	MOV L,C	9A	SBB D
01	LXI B	39	DAD SP	6A	MOV L,D	9B	SBB E
02	STAX B	3A	LDA	6B	MOV L,E	9C	SBB H
03	INX B	3B	DCX SP	6C	MOV L,H	9D	SBB L
04	INR B	3C	INR A	6D	MOV L,L	9E	SBB M
05	DCR B	3D	DCR A	6E	MOV L,M	9F	SBB A
06	MVI B	3E	MVI A	6F	MOV L,A	A0	ANA B
07	RLC	3F	CMC	70	MOV M,B	A1	ANA C
09	DAD B	40	MOV B,B	71	MOV M,C	A2	ANA D
0A	LDAX B	41	MOV B,C	72	MOV M,D	A3	ANA E
0B	DCX B	42	MOV B,D	73	MOV M,E	A4	ANA F
0C	INR C	43	MOV B,E	74	MOV M,H	A5	ANA L
0D	DCR C	44	MOV B,H	75	MOV M,L	A6	ANA M
0E	MVI C	45	MOV B,L	76	HLT	A7	ANA A
0F	RRC	46	MOV B,M	77	MOV M,A	A8	XRA B
11	LXI D	47	MOV B,A	78	MOV A,B	A9	XRA C
12	STAX D	48	MOV C,B	79	MOV A,C	AA	XRA D
13	INX D	49	MOV C,C	7A	MOV A,D	AB	XRA E
14	INR D	4A	MOV C,D	7B	MOV A,E	AC	XRA H
15	DCR D	4B	MOV C,E	7C	MOV A,H	AD	XRA L
16	MVI D	4C	MOV C,H	7D	MOV A,L	AE	XRA M
17	RAL	4D	MOV C,L	7E	MOV A,M	AF	XRA A
19	DAD D	4E	MOV C,M	7F	MOV A,A	B0	ORA B
1A	LDAX D	4F	MOV C,A	80	ADD B	B1	ORA C
1B	DCX D	50	MOV D,B	81	ADD C	B2	ORA D
1C	INR E	51	MOV D,C	82	ADD D	B3	ORA E
1D	DCR E	52	MOV D,D	83	ADD E	B4	ORA H
1E	MVI E	53	MOV D,E	84	ADD H	B5	ORA L
1F	RAR	54	MOV D,H	85	ADD L	B6	ORA M
21	LXI H	55	MOV D,L	86	ADD M	B7	ORA A
22	SHLD	56	MOV D,M	87	ADD A	B8	CMP B
23	INX H	57	MOV D,A	88	ADC B	B9	CMP C
24	INR H	58	MOV E,B	89	ADC C	BA	CMP D
25	DCR H	59	MOV E,C	8A	ADC D	BB	CMP E
26	MVI H	5A	MOV E,D	8B	ADC E	BC	CMP H
27	DAA	5B	MOV E,E	8C	ADC H	BD	CMP L
29	DAD H	5C	MOV E,H	8D	ADC L	BE	CMP M
2A	LHLD	5D	MOV E,L	8E	ADC M	BF	CMP A
2B	DCX H	5E	MOV E,M	8F	ADC A	C0	RNZ
2C	INR L	5F	MOV E,A	90	SUB B	C1	POP B
2D	DCR L	60	MOV H,B	91	SUB C	C2	JNZ
2E	MVI L	61	MOV H,C	92	SUB D	C3	JMP
2F	CMA	62	MOV H,D	93	SUB E	C4	CNZ
31	LXI SP	63	MOV H,E	94	SUB H	C5	PUSH B
32	STA	64	MOV H,H	95	SUB L	C6	ADI
33	INX SP	65	MOV H,L	96	SUB M	C7	RST 0
34	INR M	66	MOV H,M	97	SUB A	C8	RZ
35	DCR M	67	MOV H,A	98	SBB B	C9	RET
36	MVI M	68	MOV L,B	99	SBB C	CA	JZ

B-3. INSTRUCTION SUMMARY IN HEXADECIMAL ORDER (Cont.)

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
CC	CZ	D8	RC	E6	ANI	F3	DI
CD	CALL	DA	JC	E7	RST 4	F4	CP
CE	ACI	DB	IN	E8	RPE	F5	PUSH PSW
CF	RST 1	DC	CC	E9	PCHL	F6	ORI
D0	RNC	DE	SBI	EA	JPE	F7	RST 6
D1	POP D	DF	RST 3	EB	XCHG	F8	RM
D2	JUN	E0	RPO	EC	CPE	F9	SPHL
D3	OUT	E1	POP H	EE	XRI	FA	JM
D4	CNC	E2	JPO	EF	RST 5	FB	EI
D5	PUSH D	E3	XTHL	F0	RP	FC	CM
D6	SUI	E4	CPO	F1	POP PSW	FE	CPI
D7	RST 2	E5	PUSH H	F2	JP	FF	RST 7

B-4. INSTRUCTION SUMMARY IN ALPHABETICAL ORDER

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
CE	ACI	19	DAD D	78	MOV A,B	69	MOV L,C
8F	ADC A	29	DAD H	79	MOV A,C	6A	MOV L,D
88	ADC B	39	DAD SP	7A	MOV A,D	6B	MOV L,E
89	ADC C	3D	DCR A	7B	MOV A,E	6C	MOV L,H
8A	ADC D	05	DCR B	7C	MOV A,H	6D	MOV L,L
8B	ADC E	0D	DCR C	7D	MOV A,L	6E	MOV L,M
8C	ADC H	15	DCR D	7E	MOV A,M	77	MOV M,A
8D	ADC L	1D	DCR E	47	MOV B,A	70	MOV M,B
8E	ADC M	25	DCR H	40	MOV B,B	71	MOV M,C
87	ADD A	2D	DCR L	41	MOV B,C	72	MOV M,D
80	ADD B	35	DCR M	42	MOV B,D	73	MOV M,E
81	ADD C	05	DCR B	43	MOV B,E	74	MOV M,H
82	ADD D	1B	DCX D	44	MOV B,H	75	MOV M,L
83	ADD E	2B	DCX H	45	MOV B,L	3E	MVI A
84	ADD H	3B	DCX SP	46	MOV B,M	06	MVI B
85	ADD L	F3	DI	4F	MOV C,A	0E	MVI C
86	ADD M	FB	EI	48	MOV C,B	16	MVI D
C6	ADI	76	HLT	49	MOV C,C	1E	MVI E
A7	ANA A	DB	IN	4A	MOV C,D	26	MVI H
A0	ANA B	3C	INR A	4B	MOV C,E	2E	MVI L
A1	ANA C	04	INR B	4C	MOV C,H	36	MVI M
A2	ANA D	0C	INR C	4D	MOV C,L	00	NOP
A3	ANA E	14	INR D	4E	MOV C,M	B7	ORA A
A4	ANA H	1C	INR E	57	MOV D,A	B0	ORA B
A5	ANA L	24	INR H	50	MOV D,B	B1	ORA C
A6	ANA M	2C	INR L	51	MOV D,C	B2	ORA D
E6	ANI	34	INR M	52	MOV D,D	B3	ORA E
CD	CALL	03	INX B	53	MOV D,E	B4	ORA H
DC	CC	13	INX D	54	MOV D,H	B5	ORA L
FC	CM	23	INX H	55	MOV D,L	B6	ORA M
2F	CMA	33	INX SP	56	MOV D,M	F6	ORI
3F	CMC	DA	JC	5F	MOV E,A	D3	OUT
BF	CMP A	FA	JM	58	MOV E,B	E9	PCHL
B8	CMP B	C3	JMP	59	MOV E,C	C1	POP B
B9	CMP C	D2	JNC	5A	MOV E,D	D1	POP D
BA	CMP D	C2	JNZ	5B	MOV E,E	E1	POP H
BB	CMP E	F2	JP	5C	MOV E,H	F1	POP PSW
BC	CMP H	EA	JPE	5D	MOV E,L	C5	PUSH B
BD	CMP L	E2	JPO	5E	MOV E,M	D5	PUSH D
BE	CMP M	CA	JZ	67	MOV H,A	E5	PUSH H
D4	CNC	3A	LDA	60	MOV H,B	F5	PUSH PSW
C4	CNZ	0A	LDAX B	61	MOV H,C	17	RAL
F4	CP	1A	LDAX D	62	MOV H,D	1F	RAR
EC	CPE	2A	LHLD	63	MOV H,E	D8	RC
FE	CPI	01	LXI B	64	MOV H,H	C9	RET
E4	CPO	11	LXI D	65	MOV H,L	07	RLC
CC	CZ	21	LXI H	66	MOV H,M	F8	RM
27	DAA	31	LXI SP	6F	MOV L,A	D0	RNC
09	DAD B	7F	MOV A,A	68	MOV L,B	C0	RNZ

B-4. INSTRUCTION SUMMARY IN ALPHABETICAL ORDER (Cont.)

Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic	Hex	Mnemonic
F0	RP	C8	RZ	32	STA	D6	SUI
E8	RPE	9F	SBB A	02	STAX B	EB	XCHG
E0	RPO	98	SBB B	12	STAX D	AF	XRA A
0F	RRC	99	SBB C	37	STC	A8	XRA B
C7	RST 0	9A	SBB D	97	SUB A	A9	XRA C
CF	RST 1	9B	SBB E	90	SUB B	AA	XRA D
D7	RST 2	9C	SBB H	91	SUB C	AB	XRA E
DF	RST 3	9D	SBB L	92	SUB D	AC	XRA H
E7	RST 4	9E	SBB M	93	SUB E	AD	XRA L
EF	RST 5	DE	SBI	94	SUB H	AE	XRA M
F7	RST 6	22	SHLD	95	SUB L	EE	XRI
FF	RST 7	F9	SPHL	96	SUB M	E3	XTHL

APPENDIX C
ASCII CHARACTER SET

C-1. INTRODUCTION

The ASCII internal character set used with the Am95/4006 is the ANSI X3.4 1968 Version. The following is a summary of the AMSI X3.4 1968.

ASCII CODES

GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)	GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)	GRAPHIC OR CONTROL	ASCII (HEXADECIMAL)
NUL	00	+	2B	V	56
SOH	01	,	2C	W	57
STX	02	-	2D	X	58
ETX	03	.	2E	Y	59
EOT	04	/	2F	Z	5A
ENQ	05	0	30	[	5B
ACK	06	1	31	\	5C
BEL	07	2	32	]	5D
BS	08	3	33	^ (↑)	5E
HT	09	4	34	_ (←)	5F
LF	0A	5	35	`	60
VT	0B	6	36	a	61
FF	0C	7	37	b	62
CR	0D	8	38	c	63
SO	0E	9	39	d	64
SI	0F	:	3A	e	65
DLE	10	;	3B	f	66
DC1 (X-ON)	11	<	3C	g	67
DC2 (TAPE)	12	=	3D	h	68
DC3 (X-OFF)	13	>	3E	i	69
DC4 (TAPE)	14	?	3F	j	6A
NAK	15	@	40	k	6B
SYN	16	A	41	l	6C
ETB	17	B	42	m	6D
CAN	18	C	43	n	6E
EM	19	D	44	o	6F
SUB	1A	E	45	p	70
ESC	1B	F	46	q	71
FS	1C	G	47	r	72
GS	1D	H	48	s	73
RS	1E	I	49	t	74
US	1F	J	4A	u	75
SP	20	K	4B	v	76
!	21	L	4C	w	77
"	22	M	4D	x	78
#	23	N	4E	y	79
$	24	O	4F	z	7A
%	25	P	50	{	7B
&	26	Q	51	|	7C
'	27	R	52	} (ALT MODE)	7D
(	28	S	53	~	7E
)	29	T	54	DEL (RUB OUT)	7F
*	2A	U	55		

APPENDIX D
SERVICE INFORMATION

D-1. INTRODUCTION

This chapter provides service diagrams and information on service and repair assistance for AMC product lines.

D-2. SERVICE AND REPAIR ASSISTANCE

Service and repair assistance can be obtained from Advanced Micro Computers by contacting the AMC Field Service Department in Santa Clara, California at one of the following numbers:

Telephone:	(408) 988-7777
Toll Free:	(800) 672-3548 California
	(800) 538-9791 U.S.A. (except California)

If it is necessary to return a product to Advanced Micro Computers for service or repair, contact the Field Service Department at the previously listed telephone number. A Return Material Authorization number will be provided along with shipping instructions and other important information that will help AMC provide you with fast, efficient service. When reshipment is due to the product being damaged during shipment from AMC, or when the product is out of warranty, a purchase order is required for the AMC Field Service Department to initiate the repair.

Prepare the product for shipment by repackaging it in the original factory packaging material, if available. When the original packaging is not available, wrap the product in a cushioning material (such as Air Cap TH-240, manufactured by the Sealed Air Corporation, Hawthorne, New Jersey) and enclose in a heavy-duty corrugated shipping carton. Seal the shipping carton securely, mark it FRAGILE, and ship it to the address specified by the AMC Field Service Department.

Customers outside of the United States can contact an AMC Sales Office or Authorized AMC Distributor for directions on obtaining service or repair assistance.

D-3. SERVICE DIAGRAMS

The Am95/4006 assembly drawing is shown as figure D-1 and the programmed bit patterns for the memory and I/O PROMs are given in tables D-1 through D-5 respectively.

Schematic diagrams of the Am95/4006 are shown in figures D-2 through D-10. Active-low (logical 0) signals are indicated by an asterisk (*) following the signal name.

TABLE D-1. STANDARD MEMORY MAPPING PROM BIT PATTERN
(For use with 2716/2732 E-PROMs)

00: 8	20: F	40: 8	60: F	80: F	A0: F	C0: F	E0: F
01: 8	21: ↑	41: 8	61: ↑	81: ↑	A1: ↑	C1: ↑	E1: ↑
02: 8	22:	42: 9	62:	82:	A2:	C2:	E2:
03: 8	23:	43: 9	63:	83:	A3:	C3:	E3:
04: 9	24:	44: A	64:	84:	A4:	C4:	E4:
05: 9	25:	45: A	65:	85:	A5:	C5:	E5:
06: 9	26:	46: B	66:	86:	A6:	C6:	E6:
07: 9	27:	47: B	67:	87:	A7:	C7:	E7:
08: A	28:	48: F	68:	88:	A8:	C8:	E8:
09: A	29:	49: F	69:	89:	A9:	C9:	E9:
0A: A	2A:	4A: F	6A:	8A:	AA:	CA:	EA:
0B: A	2B:	4B: F	6B:	8B:	AB:	CB:	EB:
0C: B	2C:	4C: 4	6C:	8C:	AC:	CC:	EC:
0D: B	2D:	4D: 5	6D:	8D:	AD:	CD:	ED:
0E: B	2E:	4E: 6	6E:	8E:	AE:	CE:	EE:
0F: B	2F:	4F: 7	6F:	8F:	AF:	CF:	EF:
10: 4	30:	50: F	70:	90:	B0:	D0:	F0:
11: 5	31:	51: ↑	71:	91:	B1:	D1:	F1:
12: 6	32:	52:	72:	92:	B2:	D2:	F2:
13: 7	33:	53:	73:	93:	B3:	D3:	F3:
14: F	34:	54:	74:	94:	B4:	D4:	F4:
15: ↑	35:	55:	75:	95:	B5:	D5:	F5:
16:	36:	56:	76:	96:	B6:	D6:	F6:
17:	37:	57:	77:	97:	B7:	D7:	F7:
18:	38:	58:	78:	98:	B8:	D8:	F8:
19:	39:	59:	79:	99:	B9:	D9:	F9:
1A:	3A:	5A:	7A:	9A:	BA:	DA:	FA:
1B:	3B:	5B:	7B:	9B:	BB:	DB:	FB:
1C:	3C:	5C:	7C:	9C:	BC:	DC:	FC:
1D:	3D:	5D:	7D:	9D:	BD:	DD:	FD:
1E: ↓	3E: ↓	5E: ↓	7E: ↓	9E: ↓	BE: ↓	DE: ↓	FE: ↓
1F: F	3F: F	5F: F	7F: F	9F: F	BF: F	DF: F	FF: F

Address: MSB = A7
LSB = A0

Data: MSB = 03
LSB = 00

TABLE D-2. OPTIONAL MEMORY MAPPING PROM BIT PATTERN
(For use with 2708/2716 E-PROMs)

00: 8	20: F	40: 8	60: F	80: F	A0: F	C0: F	E0: F
01: 9	21: ↑	41: 8	61: ↑	81: ↑	A1: ↑	C1: ↑	E1: ↑
02: A	22:	42: 9	62:	82:	A2:	C2:	E2:
03: B	23:	43: 9	63:	83:	A3:	C3:	E3:
04: F	24:	44: A	64:	84:	A4:	C4:	E4:
05: ↑	25:	45: A	65:	85:	A5:	C5:	E5:
06:	26:	46: B	66:	86:	A6:	C6:	E6:
07:	27:	47: B	67:	87:	A7:	C7:	E7:
08:	28:	48: F	68:	88:	A8:	C8:	E8:
09:	29:	49: F	69:	89:	A9:	C9:	E9:
0A: ↓	2A:	4A: F	6A:	8A:	AA:	CA:	EA:
0B: F	2B:	4B: F	6B:	8B:	AB:	CB:	EB:
0C: 4	2C:	4C: 4	6C:	8C:	AC:	CC:	EC:
0D: 5	2D:	4D: 5	6D:	8D:	AD:	CD:	ED:
0E: 6	2E:	4E: 6	6E:	8E:	AE:	CE:	EE:
0F: 7	2F:	4F: 7	6F:	8F:	AF:	CF:	EF:
10: F	30:	50: F	70:	90:	B0:	D0:	F0:
11: ↑	31:	51: ↑	71:	91:	B1:	D1:	F1:
12:	32:	52:	72:	92:	B2:	D2:	F2:
13:	33:	53:	73:	93:	B3:	D3:	F3:
14:	34:	54:	74:	94:	B4:	D4:	F4:
15:	35:	55:	75:	95:	B5:	D5:	F5:
16:	36:	56:	76:	96:	B6:	D6:	F6:
17:	37:	57:	77:	97:	B7:	D7:	F7:
18:	38:	58:	78:	98:	B8:	D8:	F8:
19:	39:	59:	79:	99:	B9:	D9:	F9:
1A:	3A:	5A:	7A:	9A:	BA:	DA:	FA:
1B:	3B:	5B:	7B:	9B:	BB:	DB:	FB:
1C:	3C:	5C:	7C:	9C:	BC:	DC:	FC:
1D:	3D:	5D:	7D:	9D:	BD:	DD:	FD:
1E: ↓	3E: ↓	5E: ↓	7E: ↓	9E: ↓	BE: ↓	DE: ↓	FE: ↓
1F: F	3F: F	5F: F	7F: F	9F: F	BF: F	DF: F	FF: F

Address: MSB = A7
LSB = A0

Data: MSB = 03
LSB = 00

TABLE D-3. AmSYS8/8 MEMORY MAPPING PROM BIT PATTERN

00: 8	20: F	40: 8	60: F	80: F	A0: F	C0: F	E0: F
01: 9	21: ↑	41: 8	61: ↑	81: ↑	A1: ↑	C1: ↑	E1: ↑
02: A	22:	42: 9	62:	82:	A2:	C2:	E2:
03: B	23:	43: 9	63:	83:	A3:	C3:	E3:
04: F	24:	44: F	64:	84:	A4:	C4:	E4:
05: ↑	25:	45: ↑	65:	85:	A5:	C5:	E5:
06:	26:	46:	66:	86:	A6:	C6:	E6:
07:	27:	47:	67:	87:	A7:	C7:	E7:
08:	28:	48:	68:	88:	A8:	C8:	E8:
09:	29:	49:	69:	89:	A9:	C9:	E9:
0A:	2A:	4A:	6A:	8A:	AA:	CA:	EA:
0B:	2B:	4B:	6B:	8B:	AB:	CB:	EB:
0C:	2C:	4C:	6C:	8C:	AC:	CC:	EC:
0D:	2D:	4D:	6D:	8D.	AD:	CD:	ED:
0E:	2E:	4E:	6E:	8E:	AE:	CE:	EE:
0F:	2F:	4F:	6F:	8F:	AF:	CF:	EF:
10:	30:	50:	70:	90:	B0:	D0:	F0:
11:	31:	51:	71:	91:	B1:	D1:	F1:
12:	32:	52:	72:	92:	B2:	D2:	F2:
13:	33:	53:	73:	93:	B3:	D3:	F3:
14:	34:	54:	74:	94:	B4:	D4:	F4:
15:	35:	55:	75:	95:	B5:	D5:	F5:
16:	36:	56:	76:	96:	B6:	D6:	F6:
17:	37:	57:	77:	97:	B7:	D7:	F7:
18:	38:	58:	78:	98:	B8:	D8:	F8:
19:	39:	59:	79:	99:	B9:	D9:	F9:
1A:	3A:	5A:	7A:	9A:	BA:	DA:	FA:
1B:	3B:	5B:	7B:	9B:	BB:	DB:	FB:
1C:	3C:	5C:	7C:	9C:	BC:	DC:	FC:
1D:	3D:	5D:	7D:	9D:	BD:	DD:	FD:
1E: ↓	3E: ↓	5E: ↓	7E: ↓	9E: ↓	BE: ↓	DE: ↓	FE: ↓
1F: F	3F: F	5F: F	7F: F	9F: F	BF: F	DF: F	FF: F

Address: MSB = A7
LSB = A0

Data: MSB = 03
LSB = 00

TABLE D-4. STANDARD BUS CONTROL PROM BIT PATTERN

00: 0	20: 0	40: 0	60: 0	80: 0	A0: 0	C0: 0	E0: 0
01:	21:	41:	61:	81:	A1:	C1:	E1:
02:	22:	42:	62:	82:	A2:	C2:	E2:
03:	23:	43:	63:	83:	A3:	C3:	E3:
04:	24:	44:	64:	84:	A4:	C4:	E4:
05:	25:	45:	65:	85:	A5:	C5:	E5:
06:	26:	46:	66:	86:	A6:	C6:	E6:
07:	27:	47:	67:	87:	A7:	C7:	E7:
08:	28:	48:	68:	88:	A8:	C8:	E8:
09:	29:	49:	69:	89:	A9:	C9:	E9:
0A:	2A:	4A:	6A:	8A:	AA:	CA:	EA:
0B:	2B:	4B:	6B:	8B:	AB:	CB:	EB:
0C:	2C:	4C:	6C:	8C:	AC:	CC:	EC:
0D:	2D:	4D:	6D:	8D:	AD:	CD:	ED:
0E:	2E:	4E:	6E:	8E:	AE:	CE:	EE:
0F:	2F:	4F:	6F:	8F:	AF:	CF:	EF:
10:	30:	50:	70:	90:	B0:	D0:	F0:
11:	31:	51:	71:	91:	B1:	D1:	F1:
12:	32:	52:	72:	92:	B2:	D2:	F2:
13:	33:	53:	73:	93:	B3:	D3:	F3:
14:	34:	54:	74:	94:	B4:	D4:	F4:
15:	35:	55:	75:	95:	B5:	D5:	F5:
16:	36:	56:	76:	96:	B6:	D6:	F6:
17:	37:	57:	77:	97:	B7:	D7:	F7:
18:	38:	58:	78:	98:	B8:	D8:	F8:
19:	39:	59:	79:	99:	B9:	D9:	F9:
1A:	3A:	5A:	7A:	9A:	BA:	DA:	FA: 0
1B:	3B:	5B:	7B:	9B:	BB:	DB:	FB: 2
1C:	3C:	5C:	7C:	9C:	BC:	DC:	FC: 0
1D:	3D:	5D:	7D:	9D:	BD:	DD:	FD: 2
1E:	3E: 0	5E:	7E: 0	9E:	BE:	DE: 0	FE: 3
1F: 0	3F: 2	5F: 0	7F: 3	9F: 0	BF: 0	DF: 3	FF: 0

Address: MSB = A7
LSB = A0

Data: MSB = 03
LSB = 00

TABLE D-5. STANDARD I/O MAPPING PROM BIT PATTERN

00: 7	20: 7	40: 7	60: 7	80: 7	A0: 7	C0: 8	E0: E
01: ↑	21: ↑	41: ↑	61: ↑	81: ↑	A1: ↑	C1: 8	E1: 7
02:	22:	42:	62:	82:	A2:	C2: A	E2: 7
03:	23:	43:	63:	83:	A3:	C3: A	E3: 7
04:	24:	44:	64:	84:	A4:	C4: 7	E4: C
05:	25:	45:	65:	85:	A5:	C5: ↑	E5: C
06:	26:	46:	66:	86:	A6:	C6:	E6: C
07:	27:	47:	67:	87:	A7:	C7:	E7: C
08:	28:	48:	68:	88:	A8:	C8:	E8: D
09:	29:	49:	69:	89:	A9:	C9:	E9: D
0A:	2A:	4A:	6A:	8A:	AA:	CA:	EA: D
0B:	2B:	4B:	6B:	8B:	AB:	CB:	EB: D
0C:	2C:	4C:	6C:	8C:	AC:	CC:	EC: B
0D:	2D:	4D:	6D:	8D:	AD:	CD:	ED: B
0E:	2E:	4E:	6E:	8E:	AE:	CE:	EE: 7
0F:	2F:	4F:	6F:	8F:	AF:	CF:	EF: ↑
10:	30:	50:	70:	90:	B0:	D0:	F0:
11:	31:	51:	71:	91:	B1:	D1:	F1:
12:	32:	52:	72:	92:	B2:	D2:	F2:
13:	33:	53:	73:	93:	B3:	D3:	F3:
14:	34:	54:	74:	94:	B4:	D4:	F4:
15:	35:	55:	75:	95:	B5:	D5:	F5:
16:	36:	56:	76:	96:	B6:	D6: ↓	F6:
17:	37:	57:	77:	97:	B7:	D7: 7	F7:
18:	38:	58:	78:	98:	B8:	D8: 9	F8:
19:	39:	59:	79:	99:	B9:	D9: 9	F9:
1A:	3A:	5A:	7A:	9A:	BA:	DA: 7	FA:
1B:	3B:	5B:	7B:	9B:	BB:	DB: ↑	FB:
1C:	3C:	5C:	7C:	9C:	BC:	DC:	FC:
1D:	3D:	5D:	7D:	9D:	BD:	DD:	FD:
1E: ↓	3E: ↓	5E: ↓	7E: ↓	9E: ↓	BE: ↓	DE: ↓	FE: ↓
1F: 7	3F: 7	5F: 7	7F: 7	9F: 7	BF: 7	DF: 7	FF: 7

Address: MSB = A7
LSB = A0

Data: MSB = 03
LSB = 00

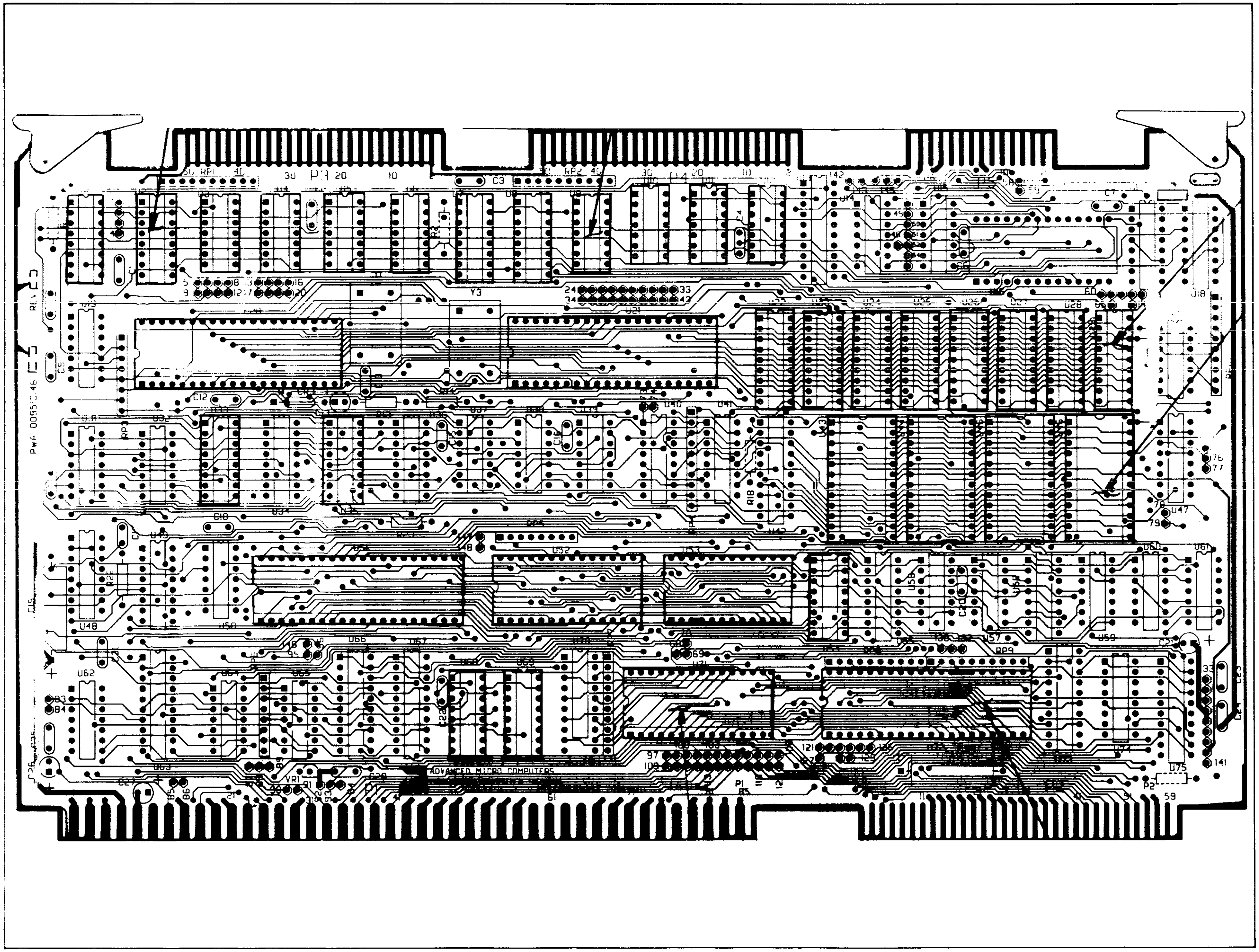
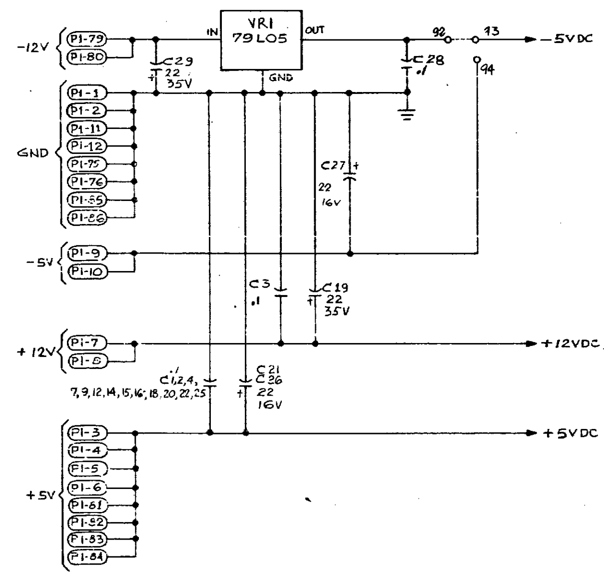


Figure D-1. Component Location Diagram

REVISIONS				
ZONE	REV	DESCRIPTION	DATE	APPROVED
	A	PROD REL ECN 0707	1/7/81	
	B	SEE ECN 0774 KP	2/24/81	

CHIPPING CONFIGURATION TABLE			
ASSEM	-001 &-002	-003	TYPE
FROM	TO	TO	
2	3	4	BLOCK
5	9	9	
6	10	10	
7	11	11	
8	12	12	
13	17	17	
14	18	18	
15	19	19	
16	20	20	
25	35	24	
26	36	36	
27	37	37	
28	38	38	
29	39	39	
30	40	40	
31	41	41	
32	42	42	
33	43	43	
50	51	49	
52	48	51	
62	63	63	
65	61	61	
76	77	77	
83	84	84	
85	86	86	
90	91	91	
92	93	93	
121	127	127	
123	128	128	
124	129	129	
130	131	131	
135	136	136	
141	140	140	
143	144	144	
69	70	70	
95	96	96	BLOCK



NOTES: UNLESS OTHERWISE SPECIFIED
1. RESISTANCE VALUES ARE IN OHMS 1/4 W, ± 5%
2. CAPACITANCE VALUES ARE IN μF.
3. ALL EVEN NUMBERED PINS ON P3 AND P4 ARE CONNECTED TO GROUND
4. THERE ARE SEVERAL SPARE RESISTOR & CAPACITOR LOCATIONS ON BD FOR ENGINEERING USE THESE ARE DENOTED BY DASHED LINES ON COMPONENT OUTLINES. INCLUDES R1,2,18.

NEXT ASSY	USED ON
503510146-003	95/4006
503510146-002	95/4006
503510146-001	95/4006
APPLICATION	

Advanced Micro Computers

SCHEMATIC MONO BOARD 95/4006

DRAWN: PHAM 3/13/80
CHECKED: 1/7/81
ISSUED: MEH 1/2/81

SIZE D | DWG. NO. 109510146-XXX | REV B | SHEET 1 OF 2

DO NOT SCALE DRAWING

Figure D-2. 95/4006 Schematic Diagram Sheet 1

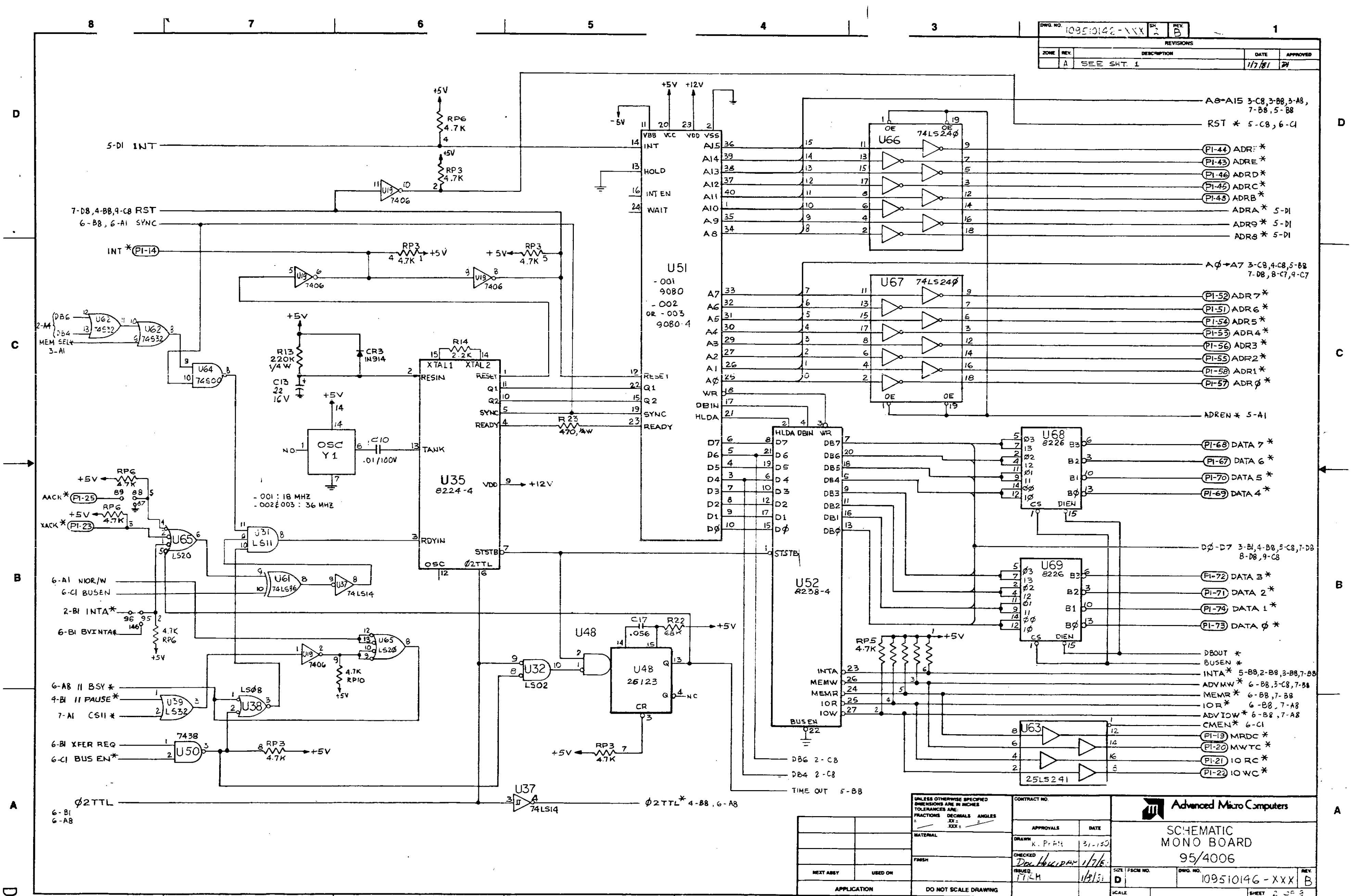


Figure D-3. 95/4006 Schematic Diagram Sheet 2

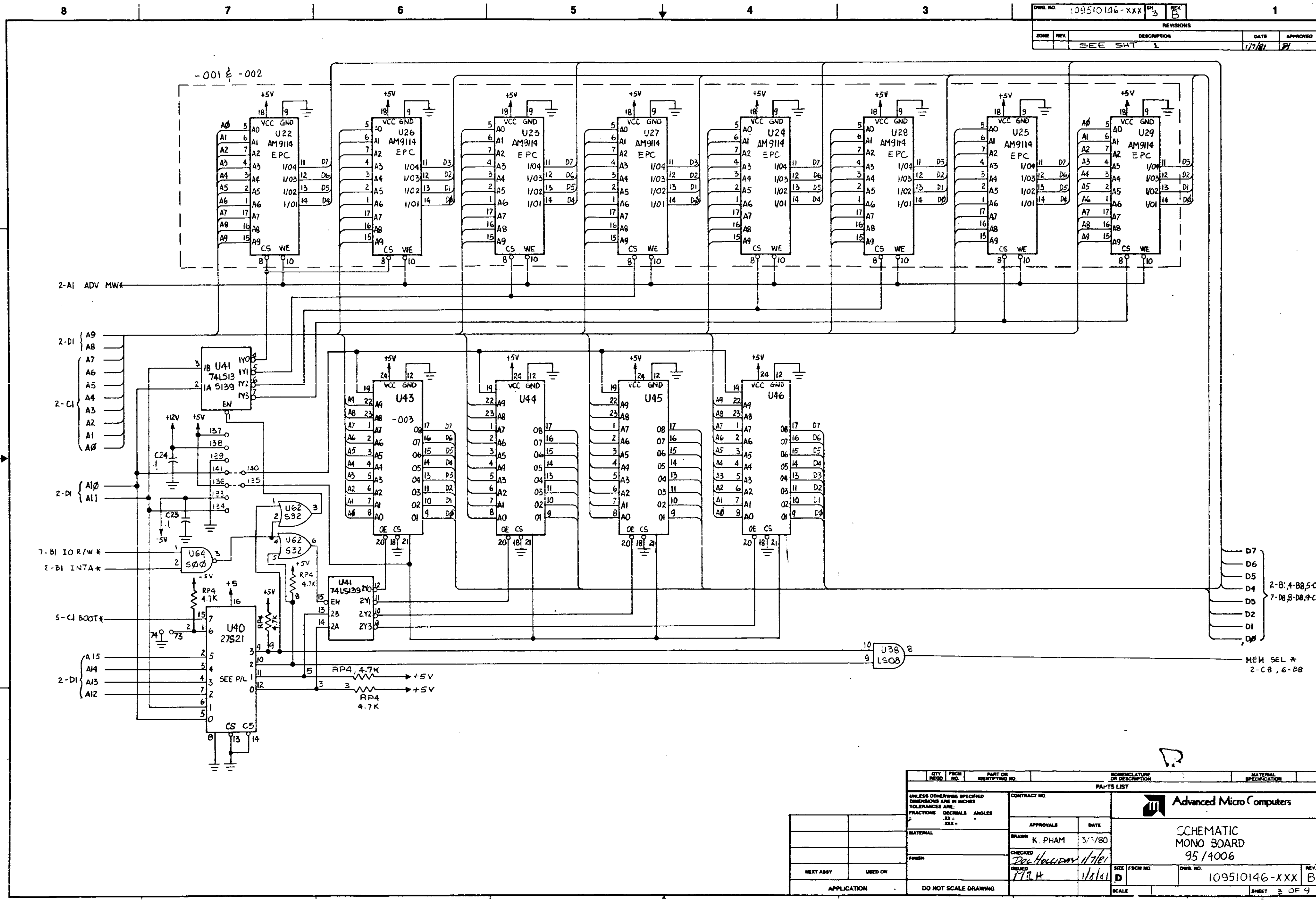


Figure D-4. 95/4006 Schematic Diagram Sheet 3

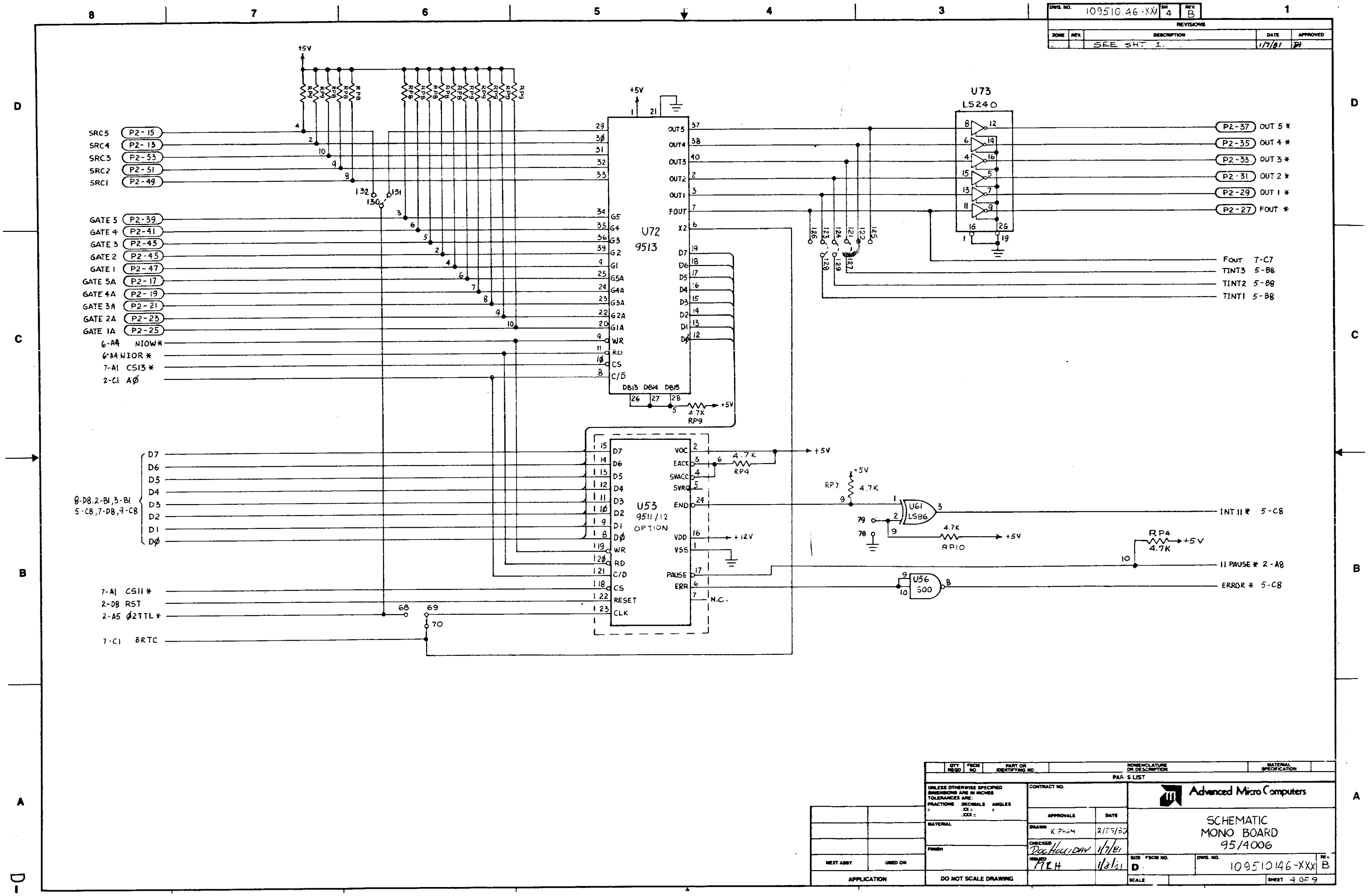


Figure D-5. 95/4006 Schematic Diagram Sheet 4

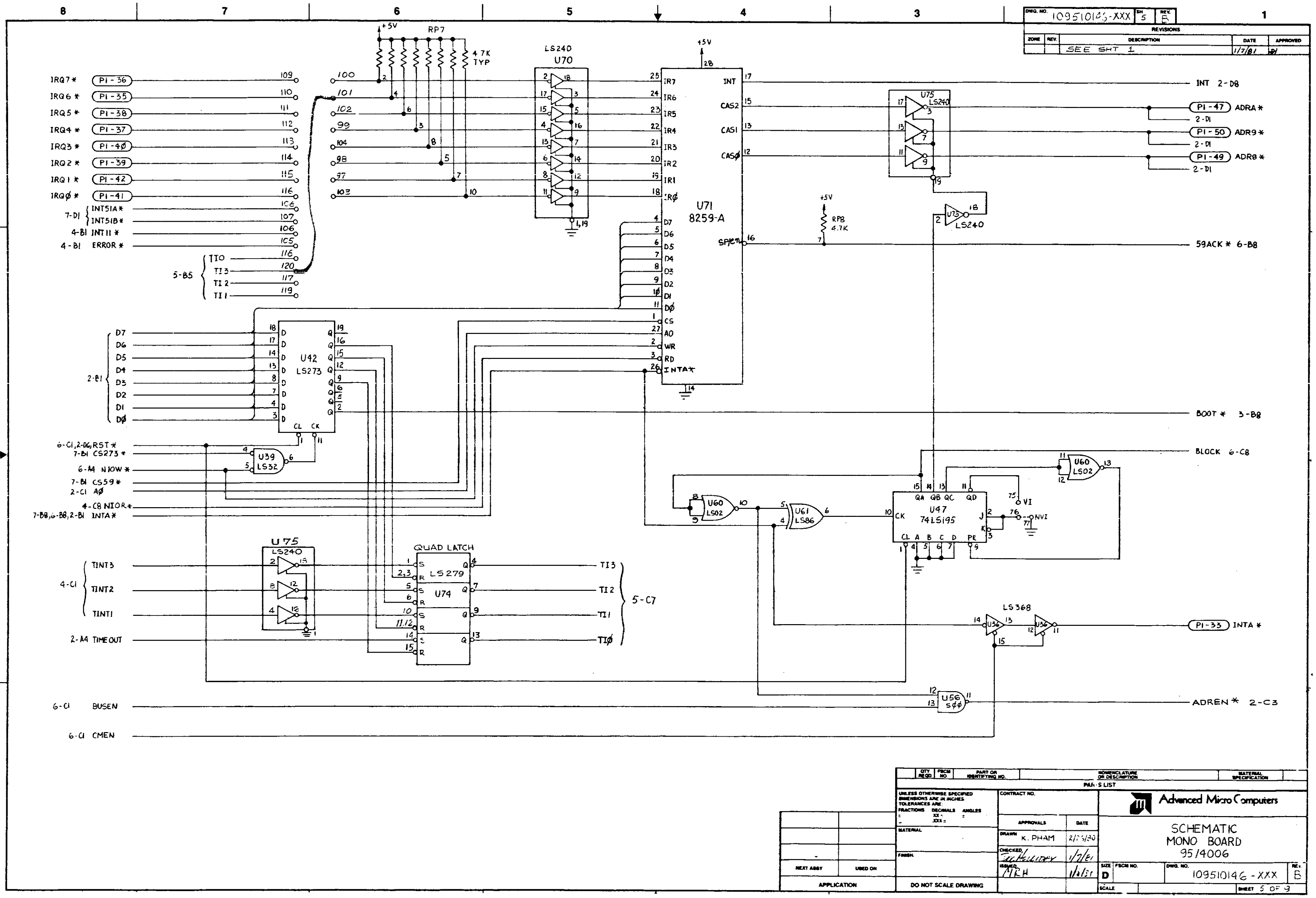


Figure D-6. 95/4006 Schematic Diagram Sheet 5

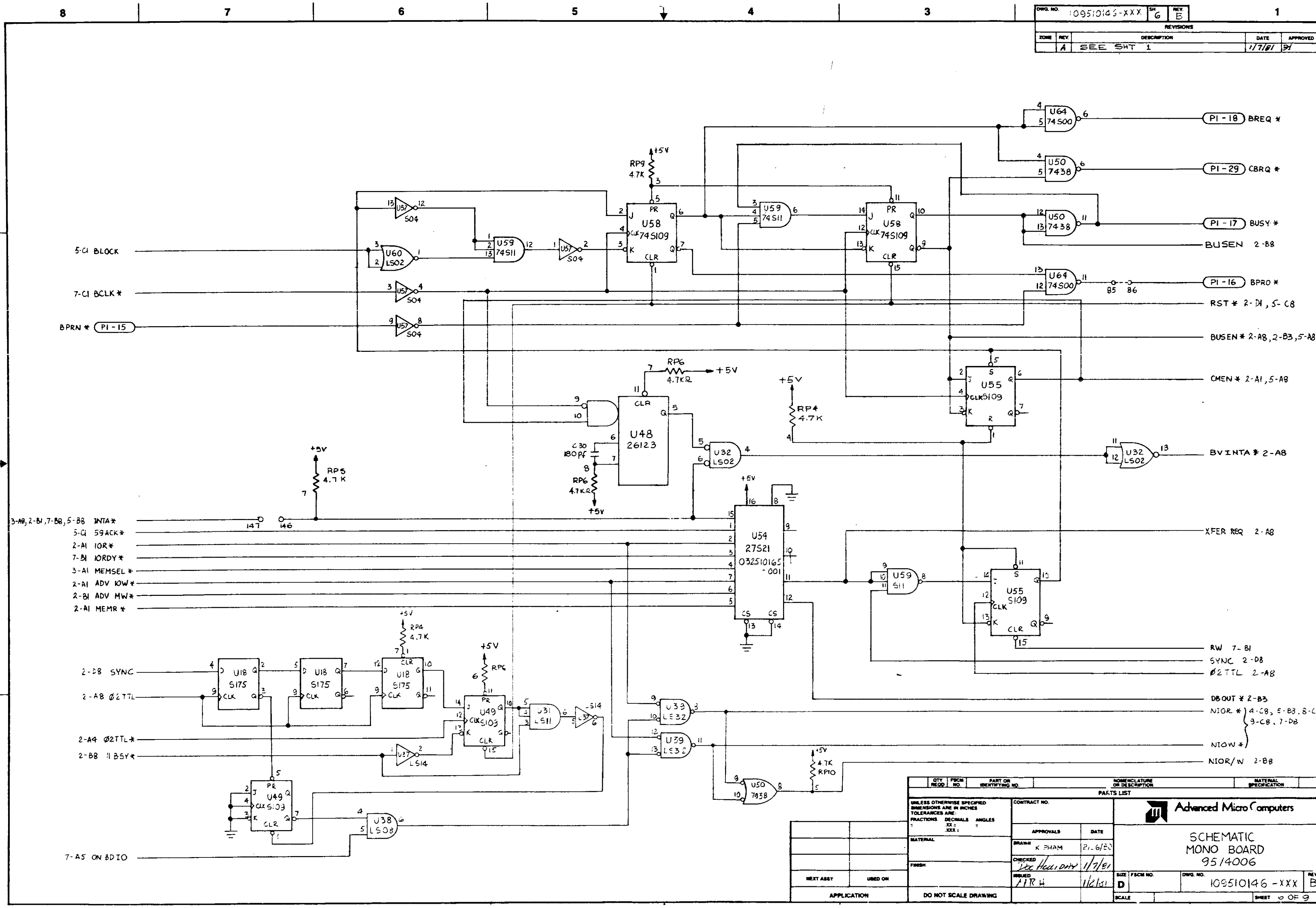


Figure D-7. 95/4006 Schematic Diagram Sheet 6

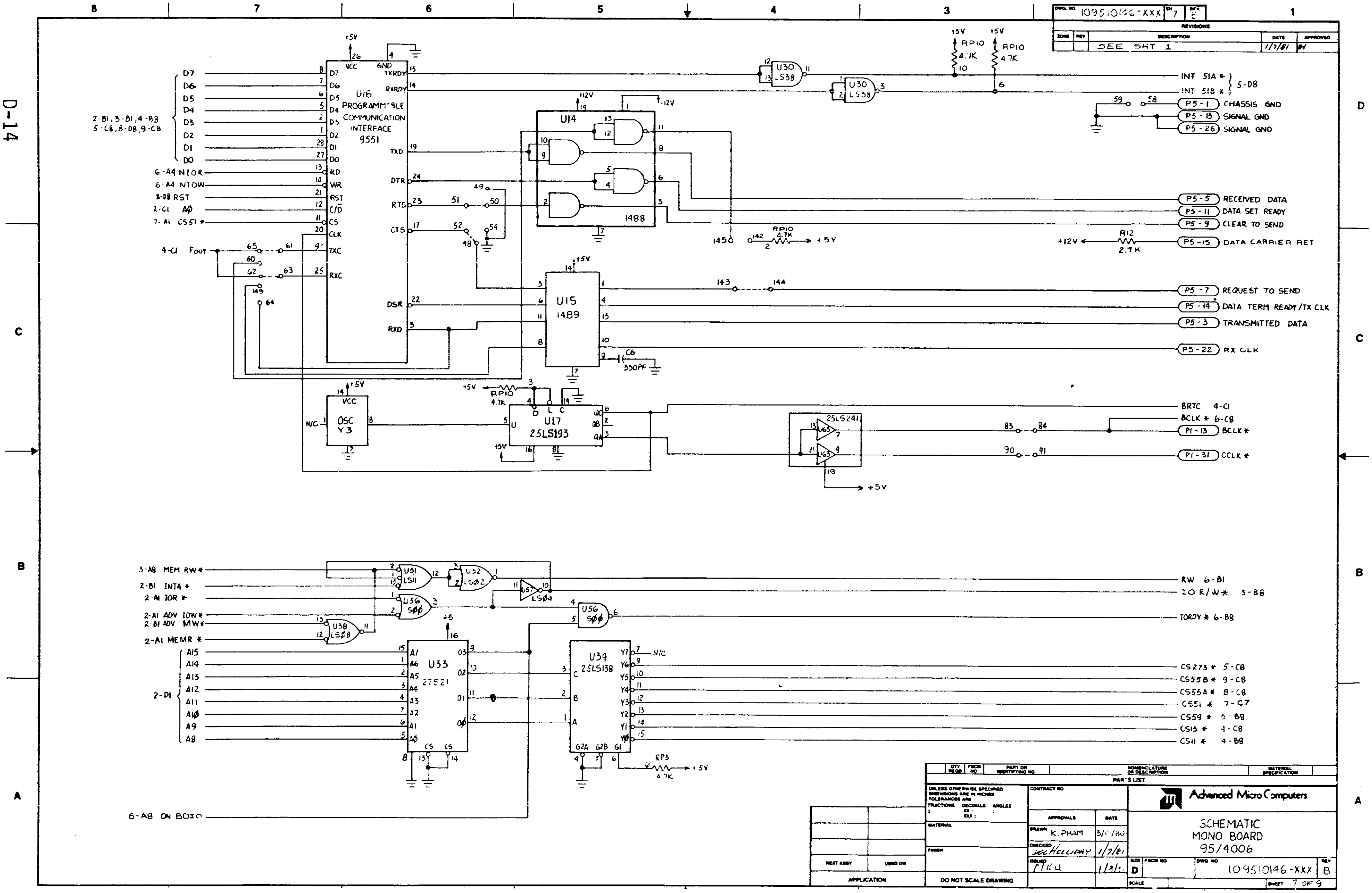


Figure D-8. 95/4006 Schematic Diagram Sheet 7

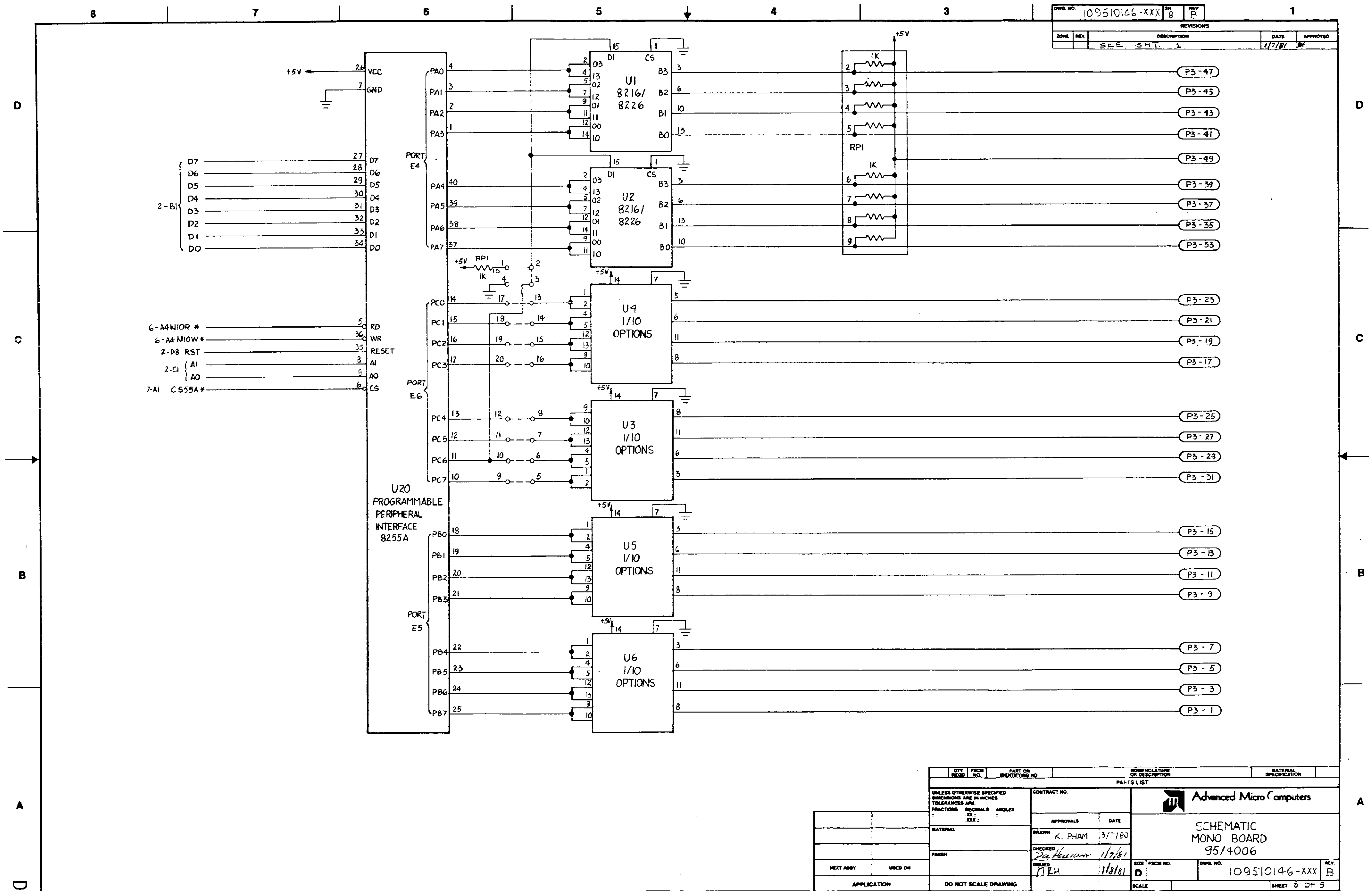


Figure D-9. 95/4006 Schematic Diagram Sheet 8

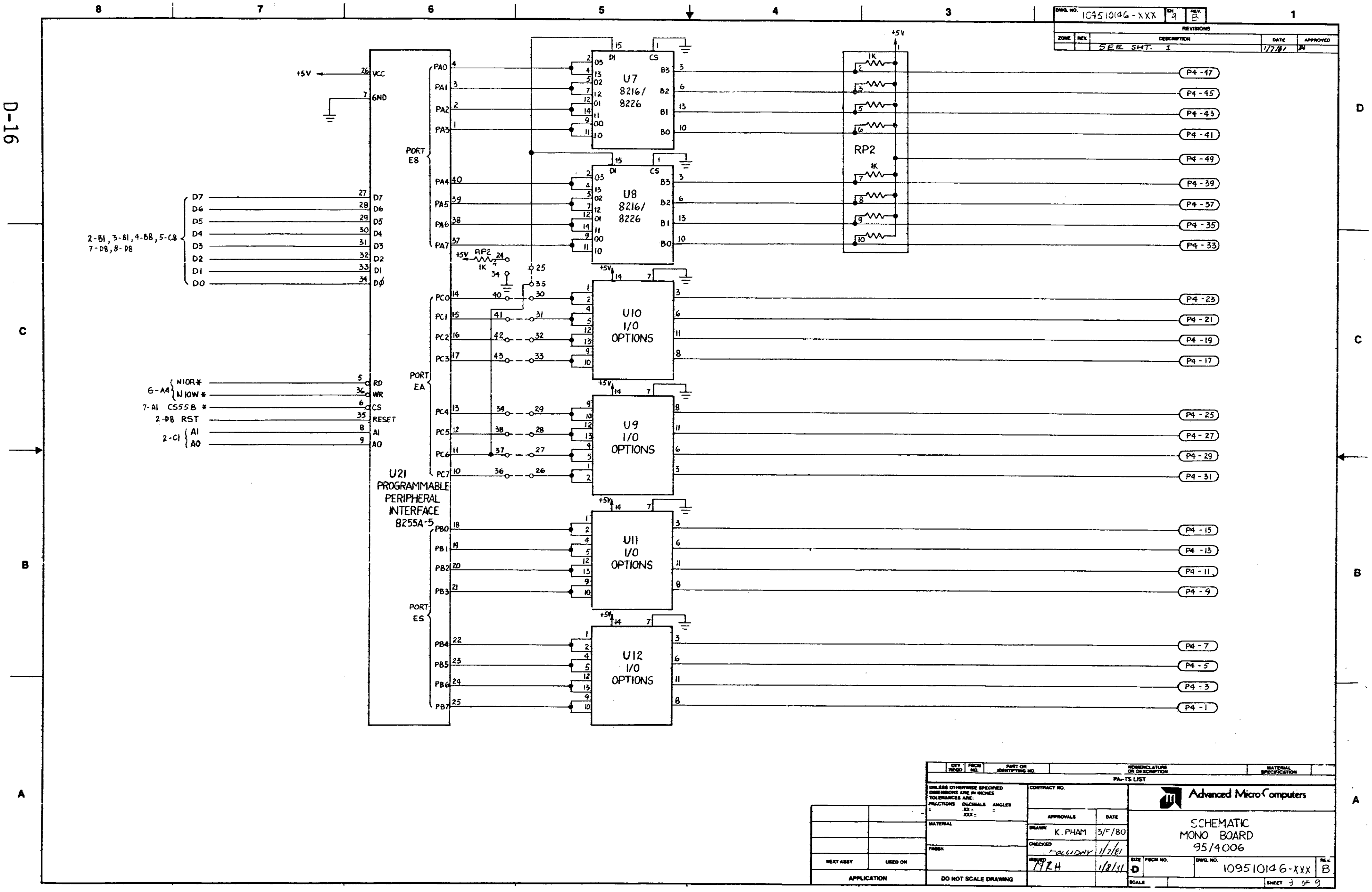


Figure D-10. 95/4006 Schematic Diagram Sheet 9

COMMENT SHEET

Address comments to:

Advanced Micro Computers
Publications Department
3340 Scott Boulevard
Santa Clara, CA 95051

TITLE: 95/4006 MonoBoard Computer
PUBLICATION NO. 00680155 Revision B

COMMENTS: (Describe errors, suggested additions or deletions, and include page numbers, etc.)

From: Name: ______________________ Position: ______________________

Company: ______________________

Address: ______________________

Advanced Micro Computers

A subsidiary of
Advanced Micro Devices
3340 Scott Boulevard
Santa Clara,
California 95051
(408) 988-7777
TELEX: 171 142