5.00

SBC 310
HIGH-SPEED MATHEMATICS UNIT
HARDWARE
REFERENCE MANUAL

Manual Order Number: 9800410A

intel®

SBC 310
HIGH-SPEED MATHEMATICS UNIT
HARDWARE
REFERENCE MANUAL

Manual Order Number: 9800410A

Intel Corporation, 3065 Bowers Avenue, Santa Clara, California 95051

PREFACE

This manual provides general information, installation, programming information, principles of operation, and service information for the Intel SBC 310 High-Speed Mathematics Unit. The information presented herein is adequate to support normal installation and programming needs. Additional system information and component part details are available in the following documents:

- *Intel Series 3000 Reference Manual,* part no. 98-221
- *Intel Series 3000 Microprogramming Manual,* part no. 98-210
- *Intel Intellec Microcomputer Development System Hardware Reference Manual,* part no. 98-132
- Hardware Reference Manuals for Intel SBC 80 Single Board Computer based systems.

The information in this manual is subject to change without notice. Intel Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel Corporation assumes no responsibility for any errors that may appear in this manual. Intel Corporation makes no commitment to update nor to keep current the information contained in this manual.

The following are trademarks of Intel Corporation and may be used only to describe Intel products:

ICE-30	MCS
ICE-80	MEGACHASSIS
INSITE	MICROMAP
INTEL	MULTIBUS
INTELLEC	PROMPT
LIBRARY MANAGER	UPI

CONTENTS

TABLES

ILLUSTRATIONS

CHAPTER 1
GENERAL INFORMATION

1-1. INTRODUCTION

The SBC 310 High-Speed Mathematics unit is a member of a complete line of Intel SBC 80 system expansion modules. In performing high-speed mathematic functions, the Math Unit acts as an intelligent processor slaved to one or more SBC 80 computer masters. The Math Unit performs its repertoire of 14 arithmetic functions an order of magnitude faster than is possible with software routines.

1-2. DESCRIPTION

The SBC 310 High-Speed Mathematics Unit (figure 1-1) is designed to be plugged into a standard SBC 604/614 Modular Backplane and Cardcage to interface directly with an SBC 80 Single Board Computer or used with an Intel Intellec Microcomputer Development System (MDS). See paragraph 2-3 for important information before installing the Math Unit in an Intellec MDS.

The Math Unit is a microprogrammed processor on a single board and includes the following standard Intel Series 3000 Schottky bipolar components: 3001 Microprogram Control Unit (MCU), 3002 Central Processing Element (CPE), 3003 Look-Ahead Carry Generator (LCG), and 3604 Electrically Programmable Read Only Memory (PROM). Also included are a pipeline register and bus interface logic. The pipeline register permits the overlapping of microinstruction fetch/execute cycles and the bus interface logic provides compatibility with the Intel Multibus.

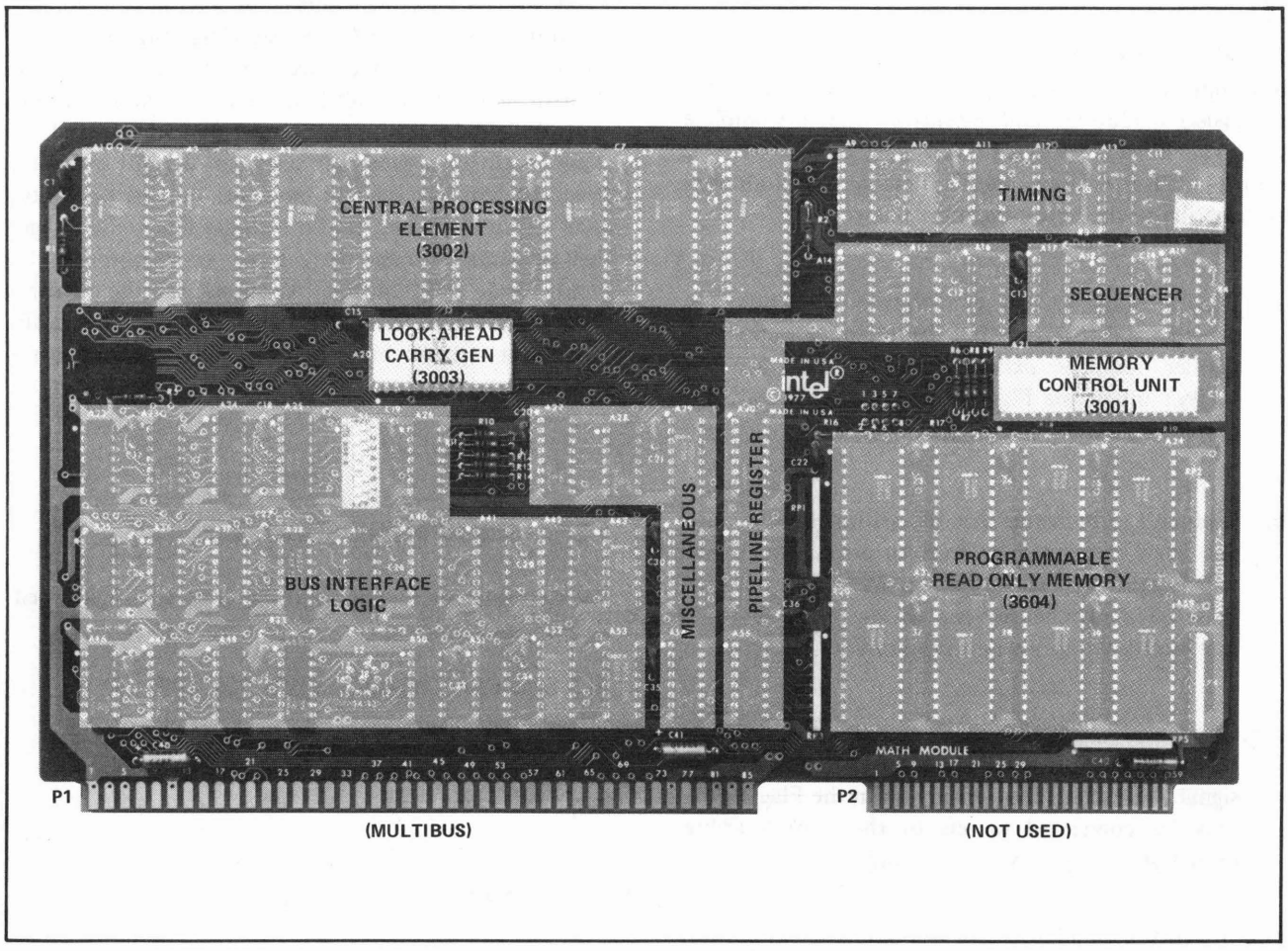


Figure 1-1. SBC 310 High-Speed Mathematics Unit

Standard operations include floating point add, subtract, multiply, divide, square, and square root; fixed point integer multiply, divide, and extended divide; conversions between fixed and floating point representations; and test, compare, and argument exchange. The Math Unit implements unbiased rounding for maximum accuracy. Unbiased rounding is the same as ordinary rounding unless the result is exactly midway between two floating point numbers. In this case, ordinary rounding always increases the result, whereas unbiased rounding rounds the result to the nearest even number. When a calculation is performed that results in either an exponent underflow or overflow, the Math Unit provides exponent wraparound to prevent loss of information.

Operation codes for invoking the arithmetic functions are passed to the Math Unit via I/O Write commands, which are also used to initialize the unit with a memory base address. I/O Read commands are used to determine the Math Unit Status. Arguments are passed to the Math Unit via Memory Write commands and the results are obtained via Memory Read commands.

The Math Unit, which can be operated either in the Interrupt or Polled mode, generates a Busy signal during processing operations and generates either a Complete signal or an Error signal after the computation is complete. These three signals convey the following information to the host computer:

a. Busy – indicates that an arithmetic operation is in progress and that the Math Unit cannot respond to further requests *except* for flag requests. The "busy" bit in the Flag buffer remains set until the current operation is complete.

b. Complete – indicates that the arithmetic operation has been completed without an error. This signal, which sets the "complete" bit in the Flag buffer, may be connected to one of the eight Multibus interrupt lines bia a wire-wrap jumper.

c. Error – indicates that the arithmetic operation has been completed with an erroneous result. This signal, which sets the "error" bit in the Flag buffer, may be connected to one of the eight Multibus interrupt lines via a wire-wrap jumper.

The memory base address and I/O base address are user selectable. The 16-bit memory base address is completely under software control and assigned by the host processor through a sequence of I/O Write commands addressed to the Math Unit. The 8-bit I/O base address is selected by a dual inline package (DIP) switch on the board.

All Math Unit operations, including arithmetic calculations, data flow between functional elements on the board, bus interface, and associated logical tasks, are controlled either directly or indirectly by a resident microprogram permanently stored in a set of eight Intel 3604 Electrically Programmable Read Only Memory (PROM) chips. This memory provides 1,024 microinstructions of 32 bits each.

The 32-bit microinstructions are accessed by a combination of a 9-bit address and a page control bit. The Microprogram Control Unit (MCU) generates the 9-bit address and the most recent microinstruction contains the page control bit. These 10 bits provide access to any 32-bit microinstruction in the 1,024-word ROM storage area. Sequencing of microinstructions is controlled by the MCU by decoding various control inputs to generate the next microprogram address. Control inputs to the MCU are taken from an address control field in the previous microinstruction and, in some cases, from processor carry out or shift out data, from the Multibus address lines, or from an operation code register. The MCU also includes flag control logic that interacts with the carry input, carry output, shift input, and shift output functions of the Central Processing Element (CPE). This logical relationship provides a versatile bit-testing capability to the control logic.

1-3. EQUIPMENT SUPPLIED

The following are supplied with the SBC 310 High-Speed Mathematics Unit:

a. Schematic Diagram, dwg no. 2001109.

b. Assembly Drawing, dwg no. 1001107.

1-4. SPECIFICATIONS

Specifications of the SBC 310 High-Speed Mathematics Unit are listed in table 1-1.

Table 1-1. Specifications

POWER REQUIREMENTS:	V_{CC} = +5V ±5% I_{CC} = 6.7A maximum; 4.9A typical
ENVIRONMENTAL REQUIREMENTS:	
Operating Temperature:	0° to 55°C (32° to 131°F)
Relative Humidity:	To 90%, without condensation
PHYSICAL CHARACTERISTICS:	
Width:	30.48 cm (12.00 inches)
Depth:	17.15 cm (6.75 inches)
Thickness:	1.27 cm (0.50 inch)
Weight:	340.5 gm (12 ounces)
BUS COMPATIBILITY:	
Interface:	TTL compatible
Connector:	86-pin, double-sided PC edge connector with 3.96-mm (0.156-inch) contact centers.
Mating Connector:	Control Data VPB01E43A00A1 or Viking 2VH43/1AV5
INTERRUPTS:	Operation Complete and Operation Error interrupts. Either one or both interrupts may be connected via jumper wire to any of the eight interrupt lines on the Multibus.

CHAPTER 2
PREPARATION FOR USE

2-1. INTRODUCTION

This chapter provides instructions for installing the SBC 310 High-Speed Mathematics Unit. These instructions include unpacking and inspection; installation considerations such as power and cooling requirements, physical dimensions, and Multibus interface requirements; switch and jumper configurations; and card installation.

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 waterstained, request that the carrier's agent 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.

For repairs to a product damaged, in shipment, contact the Intel Technical Support Center (see paragraph 5-3) to obtain a Return Authorization Number and further instructions. A purchase order will be required to complete the repair. A copy of the purchase order should be submitted to the carrier with your claim.

It is suggested that salvageable shipping cartons and packing material be saved for future use in the event the product must be reshipped.

2-3. INSTALLATION CONSIDERATIONS

The Math Unit is designed for interface with an Intel SBC 80 Single Board Computer based system or an Intel Intellec Microcomputer Development System (MDS). Important installation and interfacing criteria are presented in the following paragraphs.

CAUTION

When installing the SBC 310 in an Intel Intellec MDS, the Intellec CPU board must be reconfigured to generate a Qualified Write Signal. (Refer to *Intellec MDS Hardware Reference Manual.)* For using the SBC 310 in conjunction with MDS 016 or SBC 016 RAM boards, see paragraph 3-3 for memory addressing constraints.

2-4. POWER REQUIREMENT

The Math Unit requires +5V (±0.25V) at 6.7A maximum. For installation in an SBC 80 Single Board Computer based system, ensure that the system power supply has sufficient +5V current overhead to accommodate the additional requirement. For installation in an Intellec MDS, calculate the total current requirement for the standard modules and all installed optional modules. Ensure that the additional 6.7A maximum current requirement will not exceed the capacity of the +5V supply.

2-5. COOLING REQUIREMENT

The Math Unit dissipates 475 gram-calories/minute (1.91 BTU/minute) and adequate circulation of air must be provided to prevent a temperature rise above 55°C (131°F). The System 80 enclosures and the Intellec MDS include fans to provide adequate intake and exhaust of ventilating air.

2-6. PHYSICAL DIMENSIONS

Physical dimensions of the Math Unit are specified in table 1-1.

2-7. MULTIBUS INTERFACE REQUIREMENTS

The Math Unit is designed for installation in a standard Intel SBC 604/614 Modular Backplane and Cardcage or in the Intellec MDS motherboard. As shown in figure 1-1, the Math Unit includes two edge connectors. Connector P1, which connects to the Intel Multibus, is an 86-pin double-sided connector with 3.96-mm (0.156-inch) pin centers. Connector P1 pin assignments are listed in table 2-1, descriptions of the signal functions are given in table 2-2, and alternative mating connectors are specified in table 1-1. Connector P2 is for test only and should not be connected to the user system.

AC characteristics of the Math Unit are presented in table 2-3 and figure 2-1. DC characteristics are specified in table 2-4.

Table 2-1. Connector P1 Pin Assignments

PIN*	SIGNAL	FUNCTION	PIN*	SIGNAL	FUNCTION
1	GND	Ground	44	ADRF/	Address bus
2	GND		45	ADRC/	
3	+5 VDC	Power input	46	ADRD/	
4	+5 VDC		47	ADRA/	
5	+5 VDC		48	ADRB/	
6	+5 VDC		49	ADR8/	
7			50	ADR9/	
8			51	ADR6/	
9			52	ADR7/	
10			53	ADR4/	
11	GND	Ground	54	ADR5/	
12	GND		55	ADR2/	
13			56	ADR3/	
14	INIT/	System Initialize	57	ADR0/	
15			58	ADR1/	
16			59		
17			60		
18			61		
19	MRDC/	Memory Read Command	62		
20	MWTC/	Memory Write Command	63		
21	IORC/	I/O Read Command	64		
22	IOWC/	I/O Write Command	65		
23	XACK/	Transfer Acknowledge	66		
24	INH1/	Inhibit RAM	67	DAT6/	Data bus
25			68	DAT7/	
26	INH2/	Inhibit PROM	69	DAT4/	
27			70	DAT5/	
28			71	DAT2/	
29			72	DAT3/	
30			73	DAT0/	
31			74	DAT1/	
32			75	GND	Ground
33			76	GND	
34			77		
35	INT6/	Interrupt request on level 6	78		
36	INT7/	Interrupt request on level 7	79		
37	INT4/	Interrupt request on level 4	80		
38	INT5/	Interrupt request on level 5	81	+5 VDC	Power input
39	INT2/	Interrupt request on level 2	82	+5 VDC	
40	INT3/	Interrupt request on level 3	83	+5 VDC	
41	INT0/	Interrupt request on level 0	84	+5 VDC	
42	INT1/	Interrupt request on level 1	85	GND	Ground
43	ADRE/	Address bus	86	GND	

*All unassigned pins are reserved for system use.

Table 2-2. Multibus Signal Functions

SIGNAL	FUNCTIONAL DESCRIPTION
ADR0/-ADRF/	*Address.* These 16 address lines transmit the address of the memory location or I/O port to be accessed. ADRF/ is the most-significant bit.
DAT0/-DAT7/	*Data.* These eight bidirectional data lines transmit or receive information to or from the addressed memory location or I/O port. DAT7/ is the most-significant bit.
INH1/	*Inhibit RAM.* Generated when the Math Unit is addressed and prevents any RAM (except MDS 016 and SBC 016) sharing the same address space from responding.
INH2/	*Inhibit ROM.* Generated when the Math Unit is addressed and prevents any ROM or PROM sharing the same address space from responding.
INIT/	*Initialization.* Resets the entire system to a known internal state.
INT0/-INT7/	*Interrupt.* These eight lines are used for inputting interrupt requests to the system. The user may use the Math Unit operation Complete signal or operation Error signal (or both) to initiate an interrupt request.
IORC/	*I/O Read Command.* Indicates that the address of an I/O port is on the Multibus address lines and that the output of that port is to be read (placed) onto the Multibus data lines.
IOWC/	*I/O Write Command.* Indicates that the address of an I/O port is on the Multibus address lines and that the contents on the Multibus data lines are to be accepted by the addressed port.
MRDC/	*Memory Read Command.* Indicates that the address of a memory location is on the Multibus address lines and that the contents of that location are to be placed onto the Multibus data lines by the Math Unit.
MWTC/	*Memory Write Command.* Sent by the Math Unit to indicate that the address of a memory location is on the Multibus address lines and that the contents on the Multibus data lines are to be written into that location.
XACK/	*Transfer Acknowledge.* Indicates that the addressed memory location or I/O port has completed the specified read or write operation. That is, data has been placed onto or accepted from the Multibus data lines.

Table 2-3. AC Characteristics

PARAMETER	MIN. (nsec)	MAX. (nsec)	DESCRIPTION	REMARKS
t_{AS}	50		Address Setup to Command	
t_{ACK}			Command to Transfer Acknowledge	
	800	1120	Memory Read	Microcode generated ACK
	960	1280	Memory Write	Microcode generated ACK
	640	1440	I/O Read/Write	Microcode generated ACK
		120	Status Read	Hardware generated ACK
t_{CMD}	100		Command Pulse Width	
t_{AH}	50		Address Hold Time	
t_{TO}		110	Acknowledge Turnoff Delay	
t_{CI}		100	Address Stable to Inhibit	
t_{ID}		85	Inhibit Turnoff Delay	
t_{DH}	0		Write Data Hold Time	
t_{DS}	50		Write Data Setup to Command	

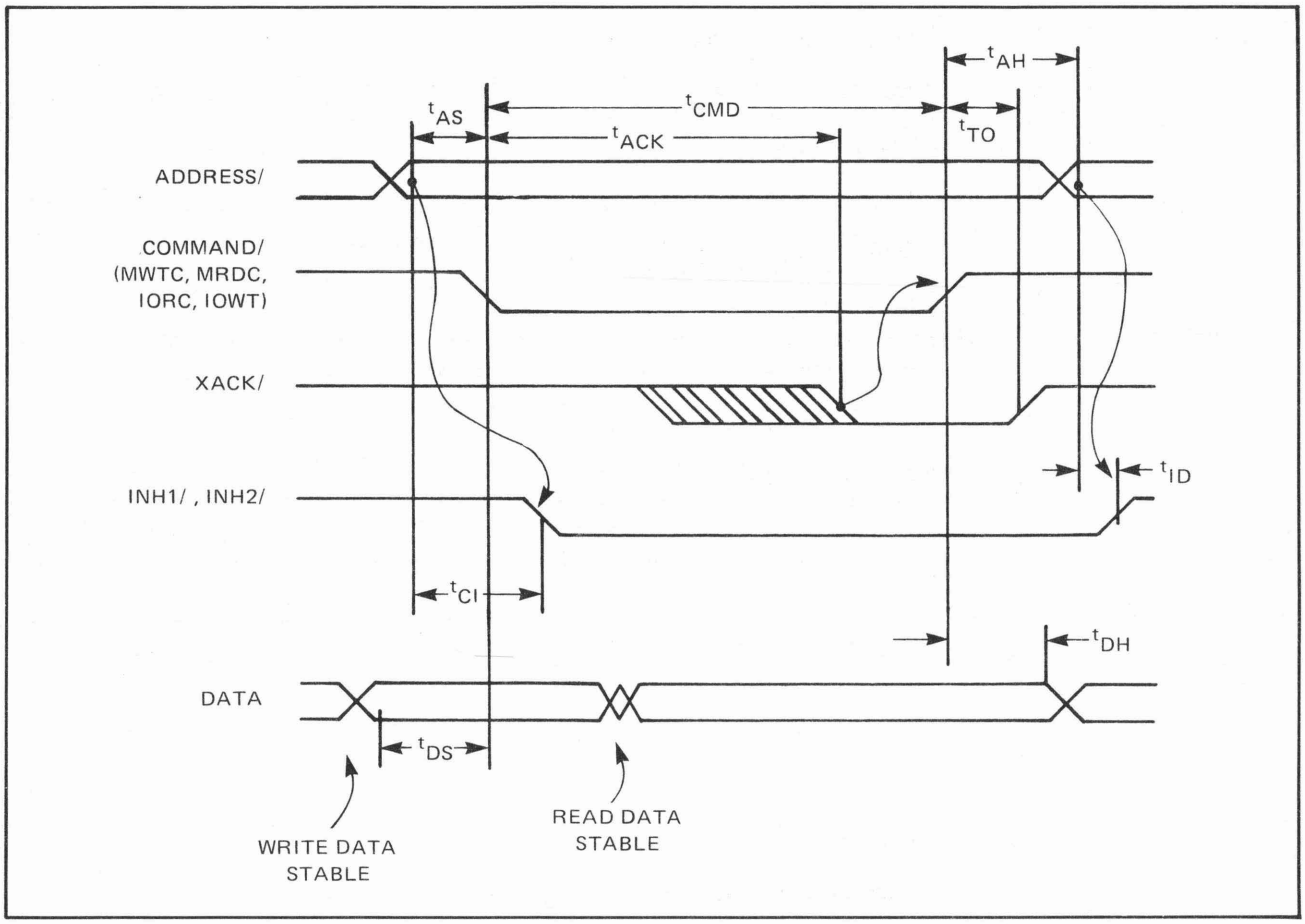


Figure 2-1. Read/Write Command Timing

Table 2-4. DC Characteristics

SIGNALS	SYMBOL	PARAMETER DESCRIPTION	TEST CONDITIONS	MIN.	MAX.	UNITS
ADR0/-ADRF/	V_{IL}	Input Low Voltage			0.8	V
IOWC/	V_{IH}	Input High Voltage		2.0		V
IORC/	I_{IL}	Input Current at Low V	V_{IN} = 0.4		-0.4	mA
MWTC/	I_{IH}	Input Current at High V	V_{IN} = 2.7		20	μA
MRDC/	*C_L	Capacitive Load			18	pF
(74LS04)						
XACK/	V_{OL}	Output Low Voltage	I_{OL} = 32 mA		0.4	V
(8097)	V_{OH}	Output High Voltage	I_{OH} = -5.2 mA	2.4		V
	I_{LH}	Output Leakage High	V_O = 2.4		40	μA
	I_{LL}	Output Leakage Low	V_O = 0.4		-40	μA
	*C_L	Capacitive Load			15	pF
DAT0/-DAT7/	V_{OL}	Output Low Voltage	I_{OL} = 55 mA		0.6	V
(8216)	V_{OH}	Output High Voltage	I_{OH} = -10 mA	2.4		V
	V_{IL}	Input Low Voltage			0.95	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.45		-0.25	mA
	I_{LH}	Output Leakage High	V_O = 5.25		100	μA
	*C_L	Capacitive Load			18	pF
INIT/	V_{IL}	Input Low Voltage			0.8	V
(7404)	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.4		-1.6	mA
	I_{IH}	Input Current at High V	V_{IN} = 2.4		400	μA
	*C_L	Capacitive Load			18	pF
INHI/ INH2/	V_{OL}	Output Low Voltage	I_{OL} = 16 mA		0.4	V
INT0/-INT7/	V_{OH}	Output High Voltage	OPEN COLLECTOR			
(7409)	V_{IL}	Input Low Voltage			0.8	V
	V_{IH}	Input High Voltage		2.0		V
	I_{IL}	Input Current at Low V	V_{IN} = 0.4		-1.6	mA
	I_{IH}	Input Current at High V	V_{IN} = 2.4		400	μA
	*C_L	Capacitive Load			22	pF

2-8. SWITCH AND JUMPER CONFIGURATIONS

The following paragraphs provide instructions for configuring the I/O base address switches and the interrupt jumpers. The memory base address assignment, which is under program control, is described in Chapter 3.

2-9. I/O BASE ADDRESS SWITCHES

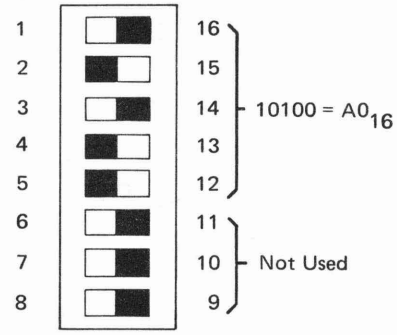
The host processor transmits control information to and receives status information from the Math Unit by issuing I/O Write and I/O Read commands, respectively. The I/O address used for these commands is relative to an 8-bit base address that must be a multiple of 8. This base address is assigned by the user by means of an 8-pole dual inline package (DIP) switch assembly (S1). Five of the eight switch poles are connected to the I/O base address detection logic; three poles are unused. Refer to table 2-5 and set the five switch poles to select the desired I/O base address. Note that the addresses are given in hexadecimal (base 16).

Table 2-5. I/O Base Address Switch Settings

I/O BASE ADDRESS (HEX)	SWITCH S1 POLE CLOSURES*					I/O BASE ADDRESS (HEX)	SWITCH S1 POLE CLOSURES*				
	1-16	2-15	3-14	4-13	5-12		1-16	2-15	3-14	4-13	5-12
00	0	0	0	0	0	80	1	0	0	0	0
08	0	0	0	0	1	88	1	0	0	0	1
10	0	0	0	1	0	90	1	0	0	1	0
18	0	0	0	1	1	98	1	0	0	1	1
20	0	0	1	0	0	A0	1	0	1	0	0
28	0	0	1	0	1	A8	1	0	1	0	1
30	0	0	1	1	0	B0	1	0	1	1	0
38	0	0	1	1	1	B8	1	0	1	1	1
40	0	1	0	0	0	C0	1	1	0	0	0
48	0	1	0	0	1	C8	1	1	0	0	1
50	0	1	0	1	0	D0	1	1	0	1	0
58	0	1	0	1	1	D8	1	1	0	1	1
60	0	1	1	0	0	E0	1	1	1	0	0
68	0	1	1	0	1	E8	1	1	1	0	1
70	0	1	1	1	0	F0	1	1	1	1	0
78	0	1	1	1	1	F8	1	1	1	1	1

*1 = ON = closed circuit to ground.

SWITCH S1



Note: Pole closures refer to DIP pin numbers. For example, pole closure 1-16 refers to pins 1 and 16 as shown at left.

2-10. INTERRUPT JUMPERS

As mentioned in Chapter 1, the Math Unit generates either a Complete signal or an Error signal after each computation is completed; either one or both of these signals may be wire-wrapped to initiate an interrupt request of the desired priority. As shown in figure 2-2, there are eight interrupt (INT) lines numbered 0 through 7. As required by the intended system application, connect either Complete signal "C" or Error signal "E" (or both) to the appropriate interrupt line on jumper pad W2. Note in figure 2-2 that the Complete and Error signals may be OR-tied to the same interrupt line or they may be connected to separate interrupt lines.

2-11. CARD INSTALLATION

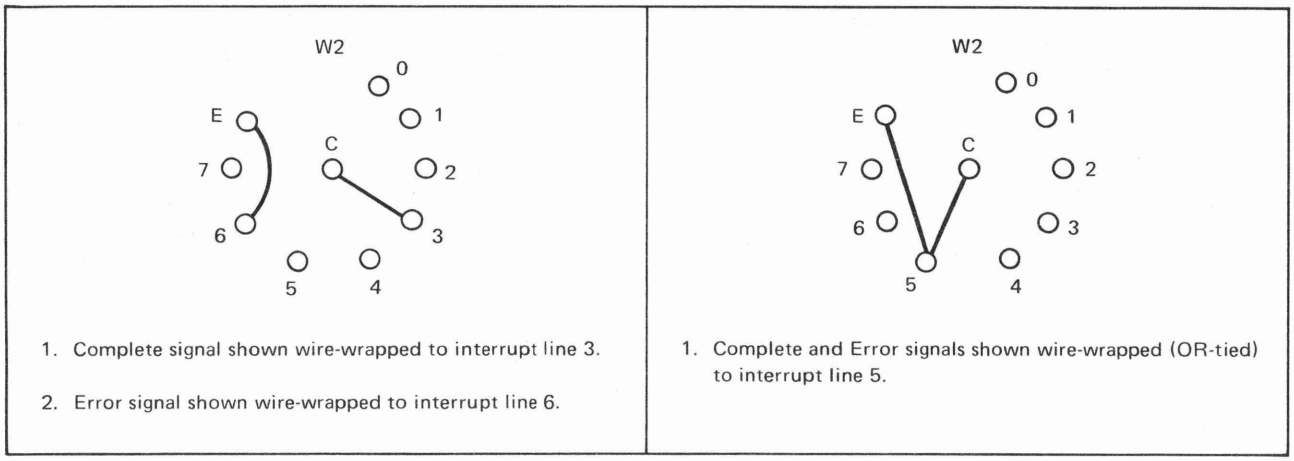
CAUTION

Always turn off the computer system power before installing or removing the Math Unit. Failure to take this precaution can result in damage.

In an SBC 80 Single Board Computer environment, install the Math Unit in any SBC 604/614 slot that has not been wired for a dedicated function. In an Intellec MDS environment, install the Math Unit in any slot except slots 1 and 2.

CAUTION

When installing the SBC 310 in an Intel Intellec MDS, the Intellec CPU board must be reconfigured to generate a Qualified Write signal. (Refer to *Intellec MDS Hardware Reference Manual.*) For using the SBC 310 in conjunction with MDS 016 or SBC 016 RAM boards, see paragraph 3-3 for memory addressing constraints.



1. Complete signal shown wire-wrapped to interrupt line 3.
2. Error signal shown wire-wrapped to interrupt line 6.

1. Complete and Error signals shown wire-wrapped (OR-tied) to interrupt line 5.

Figure 2-2. Interrupt Jumper Examples

CHAPTER 3
PROGRAMMING INFORMATION

3-1. INTRODUCTION

This chapter describes the I/O and memory base assignments; operation codes; and result, status, and flag formats. Also included are assembly language programming examples for initialization and computation.

3-2. I/O BASE ADDRESS

The I/O base address, which must be assigned by switch selection before the memory base address can be assigned, is normally performed as part of the installation procedure. (Refer to paragraph 2-9.) The I/O base address, which is used by the host processor to communicate with the Math Unit, should not be changed unless the system I/O structure is being reconfigured. As shown in table 3-1, the base address (P) and the next seven sequential memory locations (P+1 through P+7) are used to pass operation codes, memory address boundaries, and status and flag bytes between the host processor and Math Unit.

Table 3-1. I/O Addressing

I/O PORT ADDRESS	OUTPUT	INPUT
Base (P)	OP CODE	R
P+1	MEM LOW	STATUS BYTE
P+2	MEM HIGH	R
P+3	R	R
P+4	R	R
P+5	R	R
P+6	R	R
P+7	R	FLAG BYTE

P = I/O base address selected as described in paragraph 2-9.

R = reserved.

OPCODE = mathematic function; see table 3-3.

MEM LOW = memory base address (lower byte).

MEM HIGH = memory base address (upper byte).

3-3. MEMORY BASE ADDRESS

The memory base address, which is software controlled, is assigned by a sequence of two I/O Write Commands. The first command is addressed to P+1 and loads the low-order byte of the memory base address. The second command is addressed to P+2 and loads the high-order byte of the memory base address. The memory base address must be a multiple of 16; i.e., the lower byte must be in the form X0H (X is any hexadecimal digit) to accommodate the required 16 memory locations. After both bytes are output, the memory base address (M) is established and need not be reloaded during any subsequent operations. An initialization routine for establishing the memory base address is given under paragraph 3-7.

To avoid the possibility of improper operation, care must be taken in assigning the memory base address. The Math Unit generates RAM/PROM inhibit signals INH1/INH2 when accessed. Thus, system RAM/PROM (with the exception of the SBC 016 and MDS 016 RAM boards) may be overlayed by the Math Unit. In addition, the Math Unit cannot overlay memory that physically resides on an SBC 80 Single Board Computer. (Refer to paragraph 2-3.)

3-4. MATH UNIT FUNCTIONS

The Math Unit performs floating point arithmetic, fixed point integer arithmetic, compare and test operations, and float-to-fix and fix-to-float conversions. Operation codes and execution times for the various functions are listed in table 3-2. Arithmetic and conversion formats are shown in table 3-3.

3-5. ARGUMENT AND RESULT DATA FORMATS

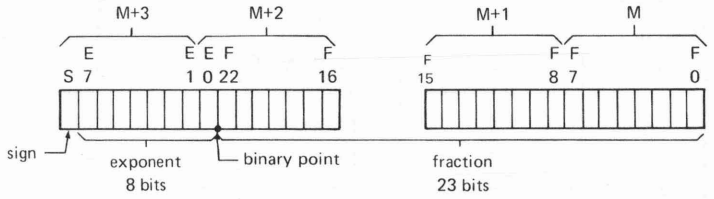
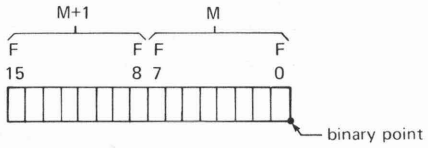
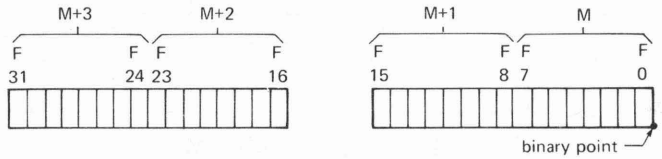
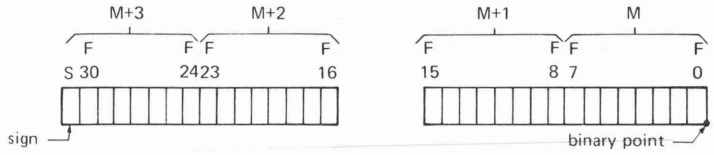
Argument and result data formats and memory locations for the various operations are presented in table 3-4. For

Table 3-2. Math Unit Functions and Execution Times

OPERATION	OP CODE	TYP. EXECUTION TIME* (μSEC)	MAX. EXECUTION TIME* (μSEC)
Fixed Point Multiply (MUL)	0	15	20
Fixed Point Divide (DIV)	1	26	30
Extended Fixed Point Divide (EDIV)	E	84	100
Floating Point Multiply (FMUL)	2	84	100
Floating Point Divide (FDIV)	3	92	110
Floating Point Add (FADD)	4	33	75
Floating Point Subtract (FSUB)	5	33	75
Floating Point Square (FSQR)	6	84	100
Floating Point Square Root (FSQRT)	7	178	205
Fixed-to-Float Conversion (FLTDS)	8	72	100
Float-to-Fixed Conversion (FIXSD)	9	42	85
Floating Point Compare (FCMPR)	A	7	7
Floating Point Test (FZTST)	B	7	7
Exchange (EXCH)	F	4	4

*This specification does not include the time requirements to pass arguments to the Math Unit and read results upon completion. This time is typically 90 microseconds.

Table 3-3. Arithmetic and Conversion Formats

FORMAT NO.	SINGLE PRECISION FLOATING POINT (SEE NOTES BELOW)
1	 where: M = memory base address S = "0" = positive; "1" = negative E7-E0 = biased exponent; bias = $7F_H$ F22-F0 = fraction; F is always normalized (i.e., a "1" is assumed in the highest bit position), yielding an effective 24-bit fraction.
2	**FIXED POINT INTEGER**
	 where: M = memory base address F15-F0 = 16-bit integer (unsigned)
2A	 where: M = memory base address F31-F0 = 32-bit integer (unsigned)
3	**CONVERSION FUNCTIONS**
	 where: M = memory base address S = "0" = positive; "1" = negative F30-F0 = two's complement integer

NOTES:

1. The floating point argument may be expressed as:
   $(-1)^S \cdot 2^{e-(bias)} \cdot (1.F)$
2. There is one unique representation for zero:
   S = 0, E7-E0 = 0, E22-F0 = 0 } All 32 bits are zero
3. The following representations are invalid:
   E7-E0 = 0 and { S ≠ 0 or, F22-F0 ≠ 0 } : Assures unique representation of zero.
   E7-E0 = 0FFH : Reserved for future enhancements.

Table 3-4. Operation Argument and Result Data Formats

OPERATION	OP CODE	ARGUMENT FORMAT*	ARGUMENTS**	RESULT FORMAT*	RESULT***
MUL	0	2 2	M, M+1 M+4, M+5	2A	M, M+1, M+2, M+3
DIV	1	2 2	M, M+1 M+4, M+5	2	M, M+1 (remainder in M+4, M+5)
EDIV	E	2A 2	M, M+1, M+2, M+3 M+4, M+5	2A	M, M+1, M+2, M+3 (remainder in M+4, M+5, M+6, M+7)
FMUL FDIV FADD FSUB	2 3 4 5	1	M, M+1, M+2, M+3 M+4, M+5, M+6, M+7	1	M, M+1, M+2, M+3
FSQR FSQRT	6 7	1	M, M+1, M+2, M+3	1	M, M+1, M+2, M+3
FLTDS	8	3	M, M+1, M+2, M+3	1	M, M+1, M+2, M+3
FIXSD	9	1	M, M+1, M+2, M+3	3	M, M+1, M+2, M+3
FCMPR	A	1	M, M+1, M+2, M+3 M+4, M+5, M+6, M+7	–	STATUS byte
FZTST†	B	1	M, M+1, M+2, M+3	–	STATUS byte
EXCH	F	Any	M, M+1, M+2, M+3 M+4, M+5, M+6, M+7	–	

*Refer to appropriate FORMAT NO. column in table 3-3.

**Second argument is always the operator (i.e., multiplier, divisor, addend, or subtrahend) and may be destroyed during the operation.

***Results of all operations, except FIXSD, are rounded; FIXSD truncates the value.

†Tests argument against 0.0.

each argument and result, this table includes a FORMAT number cross-referenced to one of the four formats shown in table 3-3. Table 3-4 also includes the OP CODE for each operation. It is important to note that the result of an operation replaces the first argument in memory, and that the second argument may be destroyed in the course of the computation. Error conditions for each operation are described in paragraph 3-6.

3-6. STATUS AND FLAGS

The Math Unit may be operated in the interrupt mode or polled mode. As described in paragraph 2-10, the Math Unit may be wire-wrapped to initiate an interrupt request under one or both of the following conditions: (1) operation complete without an error and (2) operation complete with an error. These "completion" signals may be individually wire-wrapped to separate interrupt lines or both "completion" signals may be wire-wrapped to the same interrupt line.

The condition of the Math Unit is continuously updated and stored. The flag byte shown in table 3-5 may be obtained by performing an I/O Read Command to P+7.

Table 3-5. Flag Byte Format

7	6	5	4	3	2	1	0
R	R	R	R	R	E	C	B*

where: R is reserved for future use
B is busy
C is operation complete without error
E is operation complete with error

*When B = 1, the Math Unit is busy and cannot respond to further requests *except* requests for flags.

After an operation is completed, the status byte may be obtained by performing an I/O Read Command to P+1. As shown in table 3-6, the status byte indicates error conditions where applicable and the results of Compare (FCMPR) and Test (FZTST) operations. Each of the six error conditions are defined as follows:

a. Divide by Zero (001) – This error condition is returned by either DIV, EDIV, or FDIV to indicate that an attempt was made to divide by zero.

b. Domain Error (010) – This error condition is returned by FSQRT to indicate that the argument was not in the domain of the function; i.e., an attempt was made to take the square root of a negative number.

Table 3-6. Status Byte Format

7	6	5	4	3	2 1 0
=	>	<	R	R	ERR

where: R is reserved for future use
= is equal (for FCMPR and FZTST)
> is greater than (for FCMPR and FZTST)
< is less than (for FCMPR and FZTST)
ERR is a 3-bit error code specifying one of the following error conditions:

000 No error
001 Divide by zero
010 Domain error
011 Overflow
100 Underflow
101 First argument invalid
110 Second argument invalid
111 Reserved

c. Overflow (011) – This error condition is returned by FADD, FSUB, FMUL, FDIV, FSQR, and FIXSD. In the case of FIXSD, this error indicates that the floating point number is too large to be converted to a 32-bit two's complement signed integer. If an overflow error occurs during FIXSD, the floating point argument is left unchanged and may be read from the Math Unit.

In all other cases, this error condition signifies that the exponent of the result is too large to be represented in eight bits. In this case, 0BEH is subtracted from the resulting exponent (bringing it back into range for other computations and ensuring a valid result), and the lower eight bits of the exponent are returned in the exponent field of the result.

d. Underflow (100) – This error condition is returned by FADD, FSUB, FMUL, FDIV, and FSQR to indicate that the exponent of the result is too small to be represented in eight bits. In this case, 0BEH is added to the resulting exponent (bringing it back into range for other computations and ensuring a

valid result), and the lower eight bits of the exponent are returned in the exponent field of the result.

e. First Argument Invalid (101) – This error condition is returned by FADD, FSUB, FMUL, FDIV, FSQR, FSQRT, FIXSD, FCMPR, and FZTST to indicate that the first (or only) argument for the specified function is invalid. (Refer to notes in table 3-3 for invalid representations.) The second argument (if applicable) is not checked if this error is encountered. The invalid argument is left unchanged and may be read from the Math Unit.

f. Second Argument Invalid (110) – This error condition is returned by FADD, FSUB, FMUL, FDIV, and FCMPR to indicate that the second argument for the specified function is invalid. (Refer to notes in table 3-3 for invalid representations.) This error condition occurs only after the first argument is checked and found valid. The invalid argument is left unchanged and may be read from the Math Unit.

3-7. PROGRAMMING EXAMPLES

An initialization routine for assigning the memory base address is presented in table 3-7. In this example, the I/O base address 98H is presumed to be selected during the Math Unit installation, and that memory addresses FFF0-FFFF do not overlay any excluded portion of system memory. (Refer to paragraph 3-3.) After this routine is executed, the memory base address (FFF0) remains established until re-initialized with a new address or until the system is powered down.

A computation routine is given in table 3-8. This routine can call the polled mode subroutine presented in table 3-9 or the interrupt mode subroutine presented in table 3-10.

The polled mode subroutine checks both the status byte and the flag byte. The polled mode subroutine loops on testing the busy bit until the busy bit is clear, and then checks the error bits. If no error exists, the subroutine returns to the next computation. If an error exists, the error code is input from the Math Unit and stored in Register A of the host computer.

The subroutine example presented in table 3-10 provides subroutines for handling the following interrupts: (1) completion without error and (2) completion with error.

Table 3-7. Math Unit Memory Initialization

```
IOBAS     EQU   98H            ;  I/O BASE ADDRESS
MEBAS     EQU   0FFF0H         ;  MEMORY BASE ADDRESS
MINIT:    MVI   A,LOW(MEBAS)   ;  SET UP LOW ADDRESS BYTE
          OUT   IOBAS+1
          MVI   A,HIGH(MEBAS)  ;  SET UP HIGH ADDRESS BYTE
          OUT   IOBAS+2
```

Table 3-8. Computation Routine Example

```
- EXAMPLE OF COMPUTING  R = √Z/(X-Y)

FSUB        EQU 5
FDIV        EQU 3
FSQRT       EQU 7
EXCH        EQU 0FH

COMP:       LHLD    X           ; LOAD X INTO M, M+1, M+2, M+3
                                ;       (FFF0-FFF3)
            SHLD    MEBAS
            LHLD    X+2
            SHLD    MEBAS+2
            LHLD    Y           ; LOAD Y INTO M+4, M+5, M+6, M+7
                                ;       (FFF4-FFF7)
            SHLD    MEBAS+4
            LHLD    Y+2
            SHLD    MEBAS+6
            MVI     A,FSUB      ; FSUB OP CODE
            CALL    OPCODE      ; COMPUTE X-Y, RESULT IN M, M+1,
                                ;       M+2, M+3 (FFF0-FFF3)
            LHLD    Z           ; LOAD Z INTO M+4, M+5, M+6, M+7
                                ;       (FFF4-FFF7)
            SHLD    MEBAS+4
            LHLD    Z+2
            SHLD    MEBAS+6
            MVI     A,EXCH      ; EXCHANGE (X-Y) AND Z FOR DIVIDE
            CALL    OPCODE
            MVI     A,FDIV      ; FDIV OP CODE
            CALL    OPCODE      ; COMPUTE Z/(X-Y), RESULT IN M, M+1,
                                ;       M+2, M+3 (FFF0-FFF3)
            MVI     A,FSQRT     ; FSQRT OP CODE
            CALL    OP CODE     ; COMPUTE SQUARE ROOT, RESULT
                                ;       IN M, M+1, M+2, M+3 (FFF0-FFF3)
            LHLD    MEBAS       ; STORE RESULT INTO R
            SHLD    R
            LHLD    MEBAS+2
            SHLD    R+2
                .
                .
                .
```

Note: The above routine can call either a polled mode subroutine or an interrupt handling subroutine. Examples of these subroutines are given in table 3-9 and 3-10, respectively.

Table 3-9. Polled Mode Subroutine Example

```
BUSY        EQU   1              ; BUSY BIT MASK
ECODE       EQU   7              ; ERROR CODE MASK
OPCODE:     OUT   IOBAS          ; COMMAND MATH UNIT TO PERFORM
                                 ;      CALCULATION
WAIT:       IN    IOBAS+7        ; GET FLAG BYTE
            ANI   BUSY           ; CHECK BUSY BIT
            JNZ   WAIT           ; STAY IN LOOP UNTIL NOT BUSY
            IN    IOBAS+1        ; READ ERROR CODE
            ANI   ECODE          ; MASK OUT CODE. STORE ERROR
                                 ;      CODE INTO A
            RZ                   ; RETURN FOR NEXT COMPUTATION
                                 ;      IF NO ERROR
              .                  ; USER DEFINED ERROR
              .                  ;      FUNCTION
              .
              .
```

Table 3-10. Interrupt Mode Subroutine Example

```
– SUBROUTINES FOR COMPLETION WITHOUT ERROR AND COMPLETION WITH ERROR
  INTERRUPTS

OPCODE:     OUT    IOBAS          ; COMMAND MATH UNIT TO PERFORM
                                  ;       CALCULATION
                .
                .                 ; SET UP WAIT FOR COMPLETION
                .                 ;       MESSAGE
            CALL   WAIT           ; WAIT FOR COMPLETION MESSAGE
                .                 ; CHECK FOR ABORT COMMAND
                .
                .
            RET
```

Note: WAIT is a system function which allows other programs to execute while this subroutine is waiting for an interrupt from the Math Unit.

```
– THE FOLLOWING SUBROUTINE IS EXECUTED WHEN A "COMPLETION WITHOUT ERROR"
  INTERRUPT IS RECEIVED:

CINT:           .                 ; SET UP COMPLETION MESSAGE
                .
                .
            CALL   SEND           ; INDICATE GOOD RESULT TO
                                  ;       "OPCODE"
            CALL   EXIT           ; EXIT TO SYSTEM DISPATCHER
```

Note: SEND is a system routine that transfers a message to a waiting subroutine. EXIT is a system routine that permits a program to stop executing and causes the system to start running another program.

```
– THE FOLLOWING SUBROUTINE IS EXECUTED WHEN A "COMPLETION WITH ERROR"
  INTERRUPT IS RECEIVED:

ECODE       EQU    7              ; ERROR CODE MASK
EINT:       IN     IOBAS+1        ; READ RESULT BYTE
            ANI    ECODE          ; MASK OUT ERROR CODE
                .                 ; PERFORM STANDARD ERROR
                .                 ;       FIX IF APPLICABLE, IF NONE,
                .                 ;       GO TO "EABORT"
                .
                .                 ; SET UP COMPLETION MESSAGE
            CALL   SEND           ; INDICATE GOOD RESULT TO
                                  ;       "OPCODE"
            CALL   EXIT           ; EXIT TO SYSTEM DISPATCHER
EABORT:         .                 ; SET UP ABORT MESSAGE
                .
                .
            CALL   SEND           ; SEND ABORT MESSAGE TO "OPCODE"
            CALL   EXIT           ; EXIT TO SYSTEM DISPATCHER
```

CHAPTER 4
PRINCIPLES OF OPERATION

4-1. INTRODUCTION

This chapter provides a functional description and a circuit analysis of the SBC 310 High-Speed Mathematics Unit. The circuit analysis is presented with the assumption that the reader is familiar with the architecture of the host computer and has access to literature describing in detail the Intel Series 3000 components. This chapter also presents in flowchart form an overview of how the microprogram controls the various hardware functions.

4-2. FUNCTIONAL DESCRIPTION

As shown in figure 4-1, the Math Unit is composed of the following six major functional blocks:

a. Bus Interface

b. Microprogram Control Unit (MCU)

c. Microprogram Storage

d. Pipeline Register and Output Control Logic

e. Central Processing Element (CPE)

f. Look-Ahead Carry Generator

The *Bus Interface* logic consists of those circuit elements most directly involved with the Multibus. These include memory base address and I/O base address detection logic, memory and I/O read/write command receivers, transfer and acknowledge generation logic, bidirectional data-buffers, operation status drivers, bus interrupt drivers, and RAM and PROM inhibit logic.

The *Microprogram Control Unit* (MCU) controls the sequence in which microinstructions are fetched from the microprogram memory and, for this purpose, contains a microprogram address register and associated logic for selecting the next microinstruction address. The MCU also controls two flag flip-flops included for interaction with the carry input and carry output of the *Central Processing Element* (CPE).

The major control inputs to the MCU, AC0-6, are obtained from the 7-bit address control field in the microinstruction. Two other inputs, SX0-3 and PX4-7, are controlled by different sources. The SX0-3 inputs are normally obtained from the four least-significant bits of the microprogram address. This path is used to implement a single-level subroutine scheme. When the Math Unit is initialized with a memory base address, the SX inputs are forced to all zeros. The PX4-7 inputs have three sources of control. When the Math Unit is initialized, the four PX inputs are forced to all zeros. When an operation code is received from the host processor, the contents of the operation code register are selected as the source. When a memory command is received from the host processor, the four least-significant system address bits are connected to the PX inputs. A quad 2:1 multiplexer is used to control which source is used for the PX inputs.

Microprogram Storage is provided by eight Intel 3604 PROM chips organized to form a 1K X 32-bit read-only memory array. The 32-bit microinstructions are accessed by a 9-bit address generated by the MCU and by a page select bit contained in the previously fetched microinstruction

The *Pipeline Register* permits the overlapping of microinstruction fetch/execute cycles and effectively doubles the microprogram execution speed. As each microinstruction is loaded into the Pipeline Register, the address of the next microinstruction (which is contained in the current microinstruction) is routed through the MCU to the Microprogram Storage area. Then, while the current microinstruction is being executed, the next microinstruction is fetched and waiting to be loaded into the Pipeline Register.

The *Central Processing Element* (CPE) is a 16-bit processor that performs the required arithmetic operations as well as various logical tasks that support the overall operation. The CPE features two input data buses that are interconnected to allow high-low byte exchanges; a mask/constant input bus; independent carry input, carry output, shift input, and shift output lines; as well as complete look-ahead carry outputs for use by the *Look-Ahead Carry Generator.* All CPE

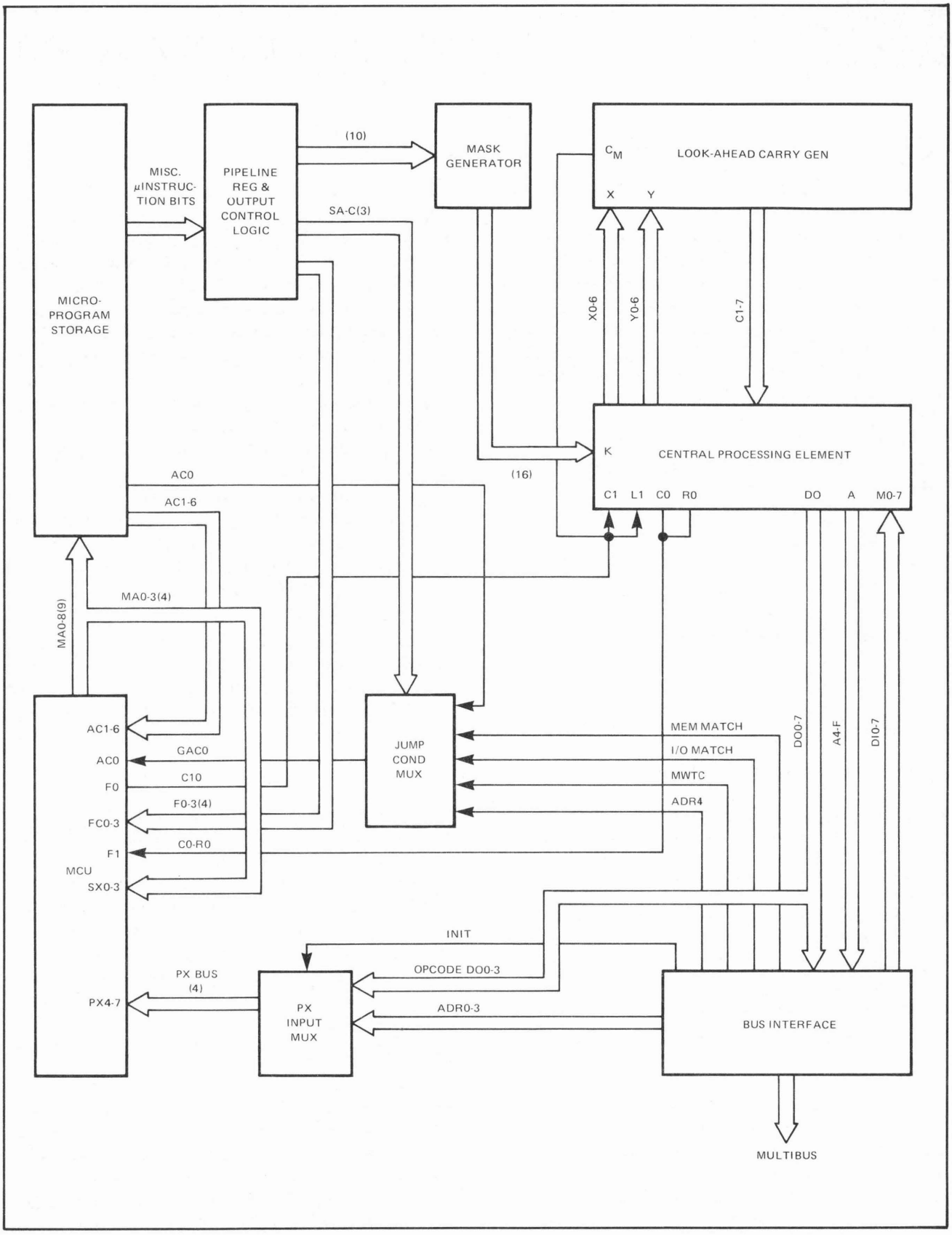


Figure 4-1. Math Unit Functional Block Diagram

operations are controlled by the resident microprogram via a 7-bit function field contained in each micro-instruction.

The Intel 3003 *Look-Ahead Carry Generator* is a high-speed circuit capable of anticipating a carry across the full width of the CPE. It uses the seven pairs of cascade outputs from the CPE (X0-6, Y0-6) to generate seven carry inputs to the CPE.

4-3. CIRCUIT ANALYSIS

The schematic diagram for the Math Unit is given in figure 5-2. The schematic diagram consists of five sheets, each of which includes grid coordinates. Signals that transverse from one sheet to another are assigned grid coordinates at both the signal source and signal destination. For example, the grid coordinates 5ZB1 locate a signal source (or signal destination as the case may be) on sheet 5 in zone B1.

Both active-high and active low signals are used. A signal mnemonic that ends with a virgule (e.g., DAT7/) denotes that the signal is active low ($\leqslant$ 0.4V). Conversely, a signal mnemonic without a virgule (e.g., MWTC) denotes that the signal is active high ($\geqslant$ 2.4V).

4-4. BUS INTERFACE

The Bus Interface includes the logic to participate in the following bus activities: (1) address detection, (2) command decoding, (3) bidirectional data buffer, (4) transfer acknowledge generation, (5) status and interrupt generation, and (6) RAM/PROM inhibit generation.

4-5. ADDRESS DETECTION. The address detection logic recognizes memory or I/O commands addressed to the Math Unit. The principal address detection logic elements (A25, A40, A41, and A42) are shown on schematic diagram sheets 1 and 3.

Memory Write Commands are used to pass arithmetic operation parameters to the Central Processing Element (CPE) and Memory Read Commands are used to read the results of the arithmetic operations. I/O Write Commands are used to initialize the CPE with its memory base address and to transfer operation codes to the Math Unit. I/O Read Commands are used to read operating status and result bytes.

The host processor outputs the memory base address, which is assigned under program control, to the CPE using I/O Write Commands. The Math Unit microprogram loads the memory base address into the CPE memory address register. Address bits 4-F form the memory base address. The base address assigned to the Math Unit memory functions must be placed on address lines ADR4/-F/ in order for the Math Unit to accept a memory command. Address bits 0-3 are not part of the base address. For memory Read/Write Commands, these four bits are decoded by the MCU into 16 memory byte addresses.

Memory address detection is performed by 4-bit magnitude comparators A25, A40, and A41. These devices compare system address bits ADR4-F with the corresponding bits stored in the CPE memory address register. If the two sets of bits match and the EN MEM MATCH signal is active, AND gate A47-8 is enabled. The output of this gate is ANDed with the signal MRDC or MWTC. (EN MEM MATCH is one of the microinstruction control bits and is controlled by the microprogram; MRDC and MWTC, respectively, are the Memory Read and Memory Write Commands.) If the second AND gate is satisfied, the signal MEM MATCH is generated.

The signal MEM MATCH performs the following three functions:

a. Enables the 3-state transfer acknowledge gate via NOR gate A9-10. See paragraph 4-8 for additional transfer acknowledge information.

b. Causes the system RAM and PROM inhibits (INH1/ and INH2/) to be generated. This prevents other memory devices on the bus from responding to Memory Read and/or Memory Write Commands.

c. Applied to input control multiplexer A27 (5ZD1) to allow MEM MATCH to be sampled by the microprogram.

The I/O base address, which is formed by address bits 3-7, is switch selectable on the board. The I/O address select switch (S1) and the I/O address detection logic are shown on schematic sheet 3.

Four-bit magnitude comparator A42 compares address bits ADR4-7 with the corresponding bits set into S1. Exclusive-OR gate A28-8 compares address bit ADR3 with the bit 3 selection of S1. If all five pairs of I/O address bits match, two of the three inputs to AND gate A36-8 are satisfied. The third input is satisfied by either an I/O Read Command (IORC) or an I/O Write Command (IOWC). When all three inputs are satisfied,

A36-8 generates an I/O command request signal, which is blocked by the READ STAT signal via A35. This ensures that the microprogram will respond only to I/O commands which are not hardware acknowledged.

The signal I/O MATCH performs the following three functions:

a. Enables the 3-state transfer acknowledge gate via NOR gate A9-10.

b. Logically ANDed with IORC and READ STAT to enable the three status gates and to generate transfer acknowledge. Refer to paragraphs 4-8 and 4-9 for additional information regarding transfer acknowledge and the three status gates.

c. Applied to input control multiplexer A27 to allow I/O MATCH to be sampled by the microprogram.

4-6. COMMAND DECODING. Memory and I/O Read/Write Commands are received from the Multibus hex inverters A48 and A49 (3ZC8). The MRDC input is ORed with IORC at gate A38-11 to produce the signal ANY READ. ANY READ is gated by the inactive (high) state of BUS IN/ (set by the microprogram) to control the DIEN/ input (direction control) to bidirectional data buffer A52 and A53 (2ZD2, 2ZB2). If BUS IN/ is false and BUS DRV/ (set by the microprogram) is true, the high level on ANY READ will cause the bidirectional data buffer to operate as a bus driver. The BUS DRV/ signal provides chip select inputs for the bidirectional data buffer. The MRDC input is also ORed with MWTC at gate A38-8 to produce an enable for the memory match detection logic. When the memory base address is detected on the system bus, a Memory Read or a Memory Write Command will permit the memory address detection logic to generate the signal MEM MATCH. Refer to paragraph 4-5 for details regarding the distribution of MEM MATCH.

The MWTC input is applied to control multiplexer A27 (5ZD1). This makes MWTC available to the microprogram for use as a conditional jump test bit.

The IORC input performs the following four functions:

a. Logically ORed with MRDC to produce ANY READ.

b. Logically ORed with IOWC at gate A38-6 to enable the I/O base address detection logic. When the I/O base address is detected on the system bus, an I/O Read or an I/O Write Command will permit the I/O address detection logic to generate the signal I/O MATCH. (Refer to paragraph 4-5.)

c. Inverted a second time and ANDed with the output of OR gate A38-3 to produce modified address bit AD3P. (The output of gate A38-3 represents the ORed combination of IOWC and ADR3.) When an I/O Read Command is received (i.e., IORC/ is true), AD3P is forced low and, as a result, the microprogram transfers data to the system data bus. Bus address bits 0-2 are used by the microprogram to determine the appropriate data. When IORC/ is false, the state of AD3P is controlled by the ORed combination of ADR3 and IOWC.

d. ANDed with I/O MATCH and READ STAT to provide an enable for the 3-state status gates and for the transfer acknowledge generation circuit.

The IOWC input, which is ORed with IORC at gate A38-6 as described above, is also ORed with ADR3 at gate A38-3. This combination is ANDed with IORC/ (inverted twice) to produce AD3P. When an I/O Write Command is received, IOWC causes AD3P to go high. As a result, the microprogram loads data from the bus into the CPE. Address bits 0-2 are used by the microprogram to determine the data destination.

4-7. BIDIRECTIONAL DATA BUFFER. Two Intel 8216 4-Bit Bidirectional Bus Drivers (A52 and A53) comprise an 8-bit bus interface data buffer. The data bus is connected to the buffer DB pins. The buffer DO pins are connected to the M-bus; this is the path by which the memory base address and arguments are written into the CPE. The buffer DI pins are connected to the D-bus; this is the path by which the host processor reads the contents of the CPE (i.e., operation result and status information).

Direction control (DIEN/) for A52 and A53 is exercised by the product of NANDing ANY READ and BUS IN/ at gate A22-3. The data buffer operates as a bus driver when an I/O Read or Memory Read Command is received (ANY READ is high) and the BUS IN/ signal is false. If no read command is active (ANY READ is false) or if BUS IN/ is true, the data buffer operates in the receiver mode. In either case, a low level must be present at the buffer chip select (CS/) input. This input is controlled by the microprogram via the BUS DRV/ signal, which must be true for the data buffer to be selected.

4-8. TRANSFER ACKNOWLEDGE GENERATION. Multibus protocol requires that a Transfer Acknowledge signal (XACK/) be generated in response to a Memory or I/O Command to indicate that write data has been accepted from the bus or that the requested read data is available on the bus. The logic for generating XACK/ is shown on schematic sheet 3.

There are two sources of control for the generation of XACK/: Read Status Commands and the microprogram. When a Read Status Command is decoded, together with an I/O Read Command (IORC) and an I/O Address Match (I/O MATCH), AND gate A47-6 is enabled. The output of A47-4 is inverted by NOR gate A9-4 and is driven onto the system bus as XACK/ by 3-state buffer A24-11. This gate is enabled by the I/O MATCH signal through NOR gate A9-10.

The microprogram controls the generation of XACK/ in response to all other memory and I/O commands. This control is exercised through the microprogram use of the SET ACK/ signal, which sets D-type flip-flop A37-9. The A37-9 output is inverted by NOR gate A9-4 and then driven onto the Multibus as XACK/ by 3-state buffer A24-11. An I/O or memory address match condition is required to enable the gate.

4-9. FLAG/INTERRUPT GENERATION. Three flag bits are made available to the Multibus to indicate whether or not the Math Unit is busy performing calculations and, if not, whether the most recent arithmetic operation produced a valid or invalid result. These three flag bits are derived from the microprogram to convey the following information:

a. BUSY/(DAT0/) indicates that an arithmetic operation is in progress and that the Math Unit cannot respond to further requests except further requests for status.

b. DONE G/(DAT1/) indicates that the arithmetic operation has been completed without an error.

c. DONE B/(DAT2/) indicates that the arithmetic operation has been completed with an erroneous result.

These three flag bits are gated onto the system bus by 3-state buffer A24, which is enabled by the NAND product of I/O MATCH, IORC, and READ STAT. The DONE G/ and DONE B/ signals are also buffered by A49 to produce COMP/ and ERR/, respectively. These two signals may be wire-wrapped to jumper pad W2 to the system bus interrupt lines.

4-10. INHIBIT GENERATION. This logic generates RAM and PROM inhibit signals INH1/ and INH2/, respectively, when the Math Unit memory base address is detected. The INH1/ and INH2/ generation logic, which is shown on schematic sheet 3, consists of exclusive-OR gate A28-3 and NAND gates A49-8 and A49-11.

4-11. MICROPROGRAM CONTROL UNIT AND INPUT CONTROL LOGIC

The functional elements described in this section are shown on schematic sheet 4. The principal elements involved include an Intel 3001 MCU (A21), an Intel 3404 High-Speed 6-Bit Latch (A15), a four-channel 2:1 multiplexer (A16), and an assortment of gates and hex inverters.

The MCU performs two major control functions. The first function is to control the sequence in which microinstructions are fetched from the microprogram memory. For this purpose, the MCU contains a microprogram address register and associated logic for selecting the next microinstruction address. The second function is the control of the two flag flip-flops that are included for interaction with the carry input and carry output logic of the CPE array. The logical organization, address control, and input/output signal handling of the MCU are described in the Intel Series 3000 documentation referenced in the preface of this manual.

4-12. MICROPROGRAM STORAGE

Microprogram storage is provided by eight Intel 3604 PROM chips in socket positions A31 through A34 and A56 through A59. Figure 4-2 shows the bit organization of a single 32-bit microinstruction and the bit functions of the chip array.

The lower 512 microinstructions are stored in chips A31, A33, A56, and A58; the upper 512 words are located in chips A32, A34, A57, and A59. Access to individual microinstructions is controlled by memory address bits MA0 through MA8 (from the MCU) and by the current microinstruction PAGE bit. Address bits MA0-MA8 access one of 512 eight-bit storage locations in each selected chip. The PROM chips are selected according to the state of the PAGE bit. When the PAGE bit is clear, the PROM chips containing the lower 512 words are selected. When the PAGE bit is set, the PROM chips containing the upper 512 words are selected.

The PAGE bit decode circuit, which is shown on schematic sheet 4, consists of NAND gate A22-8 and hex inverters A46-2 and A23-10. The NAND gate allows the chip select function to also be controlled by the INIT/ input (via two inverters). During board initialization, the low level on INIT/ disables A22-8, which causes the lower-order PROM chips to be selected. The hex inverters provide the correct complementary logic states for the two chip select paths. These paths are designated CS1L/(lower) and CS1U/(upper).

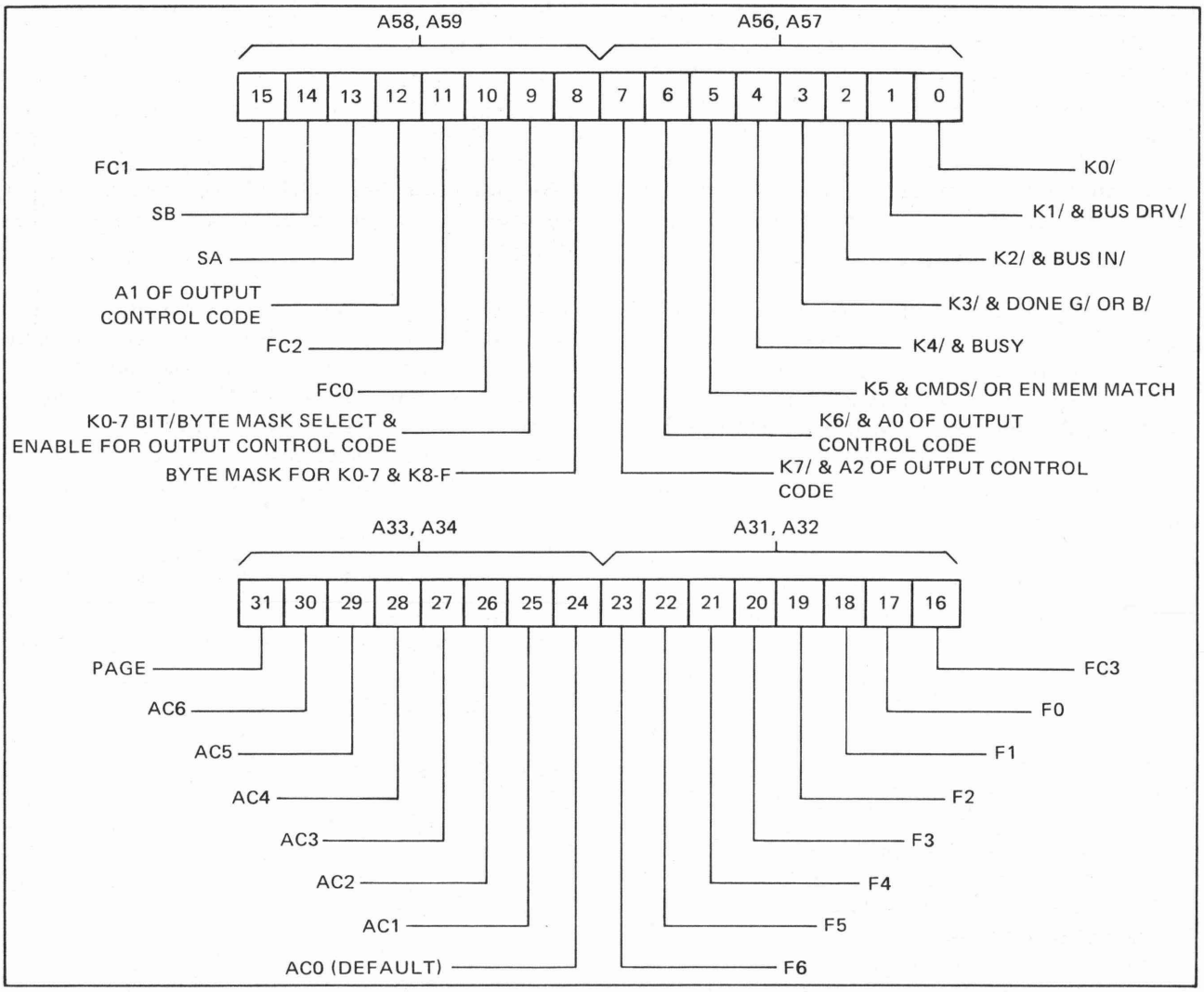


Figure 4-2. Microinstruction Bit Functions

4-13. MICROINSTRUCTION DISTRIBUTION

All PROM output bits except bits 24-30 are routed to their destinations via the pipeline register, which allows fetch operations to be overlapped with the execution of microinstructions. The pipeline register and associated microinstruction interface logic are shown on schematic sheets 4 and 5.

The pipeline register consists of five flip-flops (A14, 15, 30, 45, and 55). At each low-to-high transition on the CLK1/ line, the data present at the register D-inputs is transferred to the register outputs. Depending on the functional requirements for a particular bit, its true or complementary form is output from the register to the appropriate destination.

Microinstruction bits 24-30 form the address control inputs (AC0-AC6) to the MCU. Bits 25-30 are applied directly to inputs MC1-MC6. Bit 24 is routed through GAC0 control multiplexer A27 together with four bits of information taken from other sources. Three other microinstruction bits (7, 13, and 14) control which multiplex input is used as the AC0 bit. Normally, microinstruction bit 24 is selected. However, while in an idle loop, the microprogram polls the other multiplexer inputs to detect the need for a branch (e.g., in response to a Memory Write Command).

Microinstruction bits 0-8 are applied to the A- and B-inputs of bit/byte mask multiplexer A44 and A54. This multiplexer provides either eight individually controlled bits or a solid byte (either all zeros or all ones) to be used as a mask for bits 0-7 of the CPE arithmetic logic section (ALS). Microinstruction bits 0-7 are applied separately to the eight A-inputs of the multiplexer to form the individual bit mask. Bit 8 is fanned out to all eight B-inputs fo form the byte mask.

The multiplexer select function is controlled by microinstruction bit 9. With bit 9 clear (inactive), the following relationship exists between microinstruction bits 0-7 and the lower half of the CPE K-bus (mask) inputs:

NOTE

The multiplexer complements the selected inputs to satisfy the CPE active-low requirements for all data inputs.

MICROINSTRUCTION BIT	K-BUS BIT
0	K0/
1	K1/
.	.
.	.
.	.
6	K6/
7	K7/

When bit 9 is set (active), the K0/-7/ byte will be the complement of bit 8. Bit 8 also bypasses the multiplexer and is routed to the eight higher order K-bus inputs of the CPE via inverter A23-12. This configuration allows bit 8 to be used as a byte mask for all 16 ALS bits or for just the high-ordered byte.

When the bit mask is not selected (i.e., bit 9 is set), microinstruction bits 1-7 can be used for other control functions. These alternative control functions include: (1) selecting (or not selecting) the bidirectional data buffer (BUS DRV/), (2) enabling the memory base address detection logic (EN MEM MATCH), or (3) changing the CPE operation status (from BUSY/ to DONE G/ or DONE B/). A complete list of the alternative and control functions is provided in table 4-1.

Control of these alternative bit functions is exercised by the microprogram through an Intel 3205 1-of-8 Binary Decoder (A29). When this device is enabled, it decodes a set of three microinstruction bits (6, 7, and 12) into one of eight active-low outputs. Decoder A29 is enabled by the clear state of bit 9. This prevents mask bits that have alternative functions from being used for two different purposes at the same time. Decoder A29 is also enabled by a clock pulse that is 180 degrees out of phase with main clock CLK1/. This ensures that any changes in the bit functions controlled by A29 do not occur while the pipeline register, MCU, or CPE is being clocked.

Two of the decoded outputs, SET ACK/ and LDOPC/, are used directly to control certain functions. Refer to table 4-1 for details. Four other outputs of A29 are used to selectively strobe microinstruction bits 1-5 into a set of latches. For example, when bits 6, 7, and 12 are all ones, the 07 output of A29 enables bits 1, 2, and 5 through device A43. This is an Intel 3404 High-Speed 6-Bit Latch. Its outputs are BUS DRV/, BUS IN/ and CMDS/. When bits 6, 7, and 12 contain a 010, respectively, the 04 output of A29 enables bits 3 and 5 through a different section of A43 to produce DONE G/ and EN MEM MATCH. Bit 5 has three different functions, depending on the state of bit 9 and the code contained in bits 6, 7, and 12. See table 4-1 for details.

4-14. CENTRAL PROCESSING ELEMENT

Arithmetic operations of the Math Unit are performed by the Central Processing Element (CPE), which is composed of eight Intel 3002 Central Processing Element chips. These devices are shown on schematic sheets 1 and 2 as A1-A8. The logical organization, basic timing, and input/output signal handling of the CPE are described in the Intel Series 3000 documentation referenced in the preface to this manual.

Each CPE chip contributes a 2-bit slice to the processing logic and the associated registers to form a 16-bit central processing array. Bit organization among the eight CPE chips is as follows:

CHIP DESIGNATION	BIT ASSIGNMENT
A1	14, 15
A2	12, 13
A3	10, 11
A4	8, 9
A5	6, 7
A6	4, 5
A7	2, 3
A8	0, 1

The host processor transfers data to and from the CPE via the 8-bit system data bus (DAT0/-7/). Data from the system bus enters the CPE via its M-bus; data is output to the system bus via the CPE D-bus. The CPE D-bus outputs and I-bus are interconnected to permit byte exchanges. This configuration allows an 8-bit shift through the accumulator to be executed in a single cycle.

Table 4-1. Microinstruction Output Control Functions

MICRO-INSTRUCTION BIT	OUTPUT ENABLE CODE BITS* 7	12	6	OUTPUT FUNCTION
–	0	0	0	SET ACK/ – An output enable code of 000 generates SET ACK/, which causes XACK/ to be output onto system bus.
–	0	1	1	LDOPC/ – An output enable code of 011 generates LDOPC/, which causes operation code to be set into Op Code Register.
1	1	1	1	BUS DRV/ – When bit 1 is set, BUS DRV/ enables bidirectional data buffer chip select input.
2	1	1	1	BUS IN/ – When bit 2 is set, BUS IN/ enables bidirectional data buffer receiver circuits. When bit 2 is clear, the buffer driver circuits are enabled. In either case, bit 1 must be set.
3	1	0	0	DONE G/ – With output enable code of 100, microinstruction bit 3 indicates that most recent arithmetic operation is complete and that result is valid.
3	1	1	0	DONE B/ – With output enable code of 110, bit 3 indicates that most recent arithmetic operation is complete but that result is invalid.
4	1	0	1	BUSY/ – Bit 4 indicates that CPE is currently engaged in an operation so that its registers must not be accessed at this time.
5	1	0	0	EN MEM MATCH – With output enable code of 100, a logical 0 at bit 5 will enable memory base address detection logic. Setting bit 5 will prevent a memory match from being detected.
5	1	1	1	CMDS/ – With output enable code of 111, bit 5 generates Command Select signal. CMDS/ connects system address bits 0-3 to MCU PX bus. When CMDS/ is inactive (high), contents of the Op Code Register are applied to PX bus.

*Microinstruction bits 6, 7, and 12 are shown in conventional octal code order; that is, bit 6 = A0, bit 12 = A1, and bit 7 = A2.

Data inputs to the CPE can be masked by the contents of the CPE K-bus inputs. Individually controlled bit masks are available to the low-order data byte or a solid byte mask can be applied to both low- and high-order data bytes. The single-bit masks are controlled by microinstruction bits 0-7. The solid byte mask is controlled by bit 8 of the microinstruction. The K-bus is used for a variety of purposes. During arithmetic operations, the K-bus can be used to mask portions of the field being executed. It can also be used to supply constants to the CPE from the microprogram. During non-arithmetic operations, the K-bus can be used to mask the contents of a selected register or bus as part of bit-testing or zero-detection operations.

The memory base address is stored in the CPE memory address register. The host processor assigns the memory base address to the Math Unit by outputting the low-order memory address byte to the I/O base address +1. The high-order byte is output to I/O base address +2.

The contents of the CPE memory address register are applied to a set of magnitude comparators via address lines A04-A15. The comparators test the contents of the system address bus against the assigned memory base address to detect a memory command addressed to the Math Unit.

All arithmetic and logical operations performed by the CPE are specified by a 7-bit microfunction code. This code, which is contained in bits 17-23 of the micro-instruction, is applied to the F0-F6 inputs of each CPE chip. These bits are decoded within the CPE to select the appropriate ALS function, generate the desired register address, and control internal multiplexers.

4-15. LOOK-AHEAD CARRY GENERATOR

The Intel 3003 Look-Ahead Carry Generator (A20) shown on schematic sheet 2 is used by the CPE to increase processing speed. The CPE supplies the Look-Ahead Carry Generator (LCG) with seven pairs of active-high cascade outputs (X0-X6 and Y0-Y6). The LCG carry input, Cn/, is controlled by the MCU Flag Out (FO) logic. The LCG uses the Cn/, X, and Y inputs to generate seven active-low carries, which are applied to the CI inputs of all the CPE chips except the least-significant chip.

4-16. CLOCK GENERATOR

Timing control for all synchronous logic is provided by an Intel 8224 Clock Generator (A13), a binary counter (A19), and assorted gates and inverters. Figure 4-3 illustrates the timing waveforms for the clock generator.

Synchronous clocks, CLK1/ and CLK1A/ synchronize all time-related functions except output control decoder A29. This device is clocked in mid-cycle to ensure that all other operating conditions are stable at the time any output control functions change state.

Clock generator A13, which is controlled by 25-MHz crystal oscillator Y1, produces a square wave with a 40-nanosecond pulse width. Binary counter A19 divides this frequency to produce 12.5-MHz and 6.25-MHz square wave outputs. NOR gate A9-1 combines these two square waves to produce a stream of active-high clock pulses, with a 40-nanosecond pulse width and a 160-nanosecond period. This clock stream is inverted by A23-4 to produce CLK1/ and by A12-6 to produce CLK1A/. These active-low clocks are distributed to the pipeline register, MCU, and CPE.

The 12.5-MHz and the 6.25-MHz square waves are also applied to the circuit formed by A18-3 and A18-6. This current produces an active-low clock stream that is 180 degrees out of phase with CLK1/. This out-of-phase clock is used to complete the enable for A29.

4-17. MICROPROGRAM OVERVIEW

Figures 4-4 through 4-10 present in flowchart form the implementation of the microprogram to control the various hardware functions discussed in preceding paragraphs.

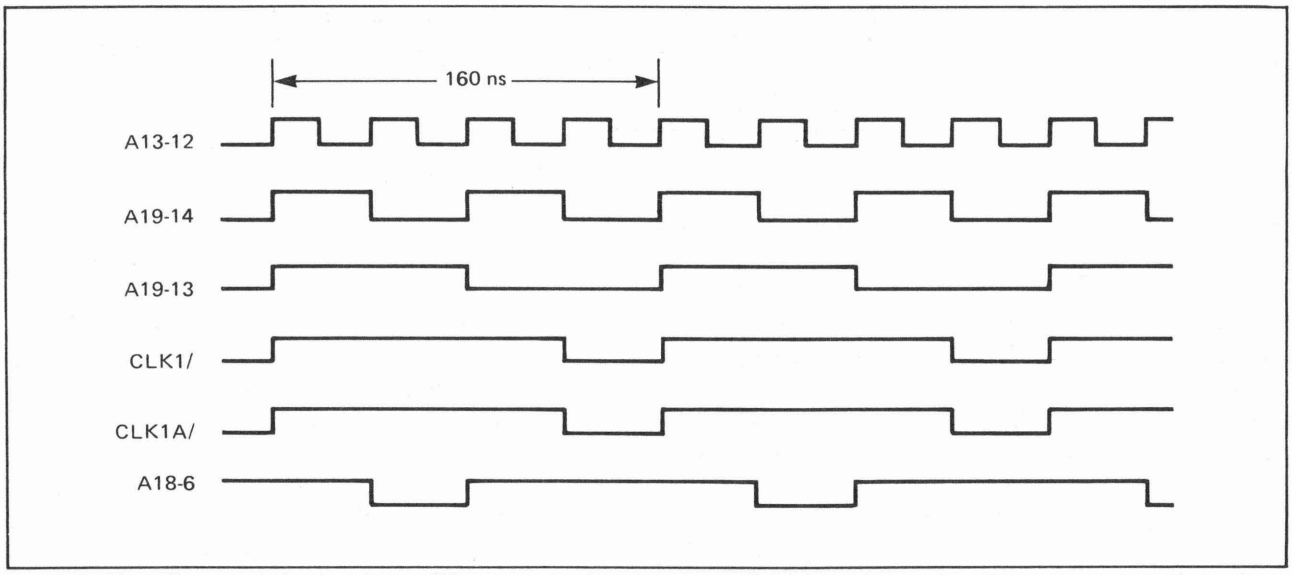


Figure 4-3. Math Unit Clock Generator Timing

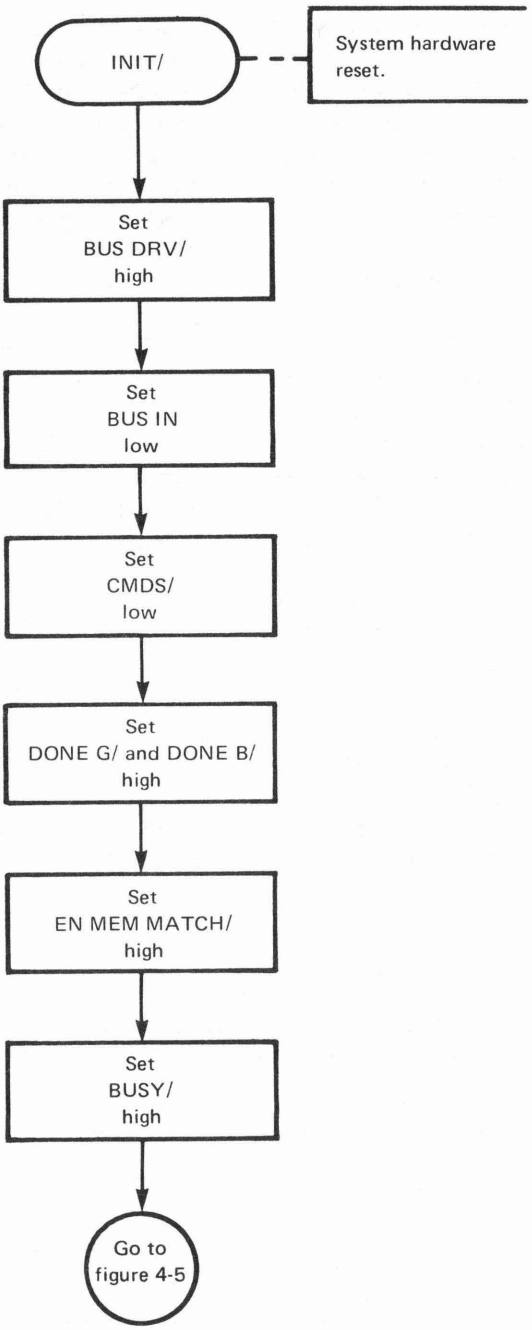


Figure 4-4. Hardware Initialization from System Reset

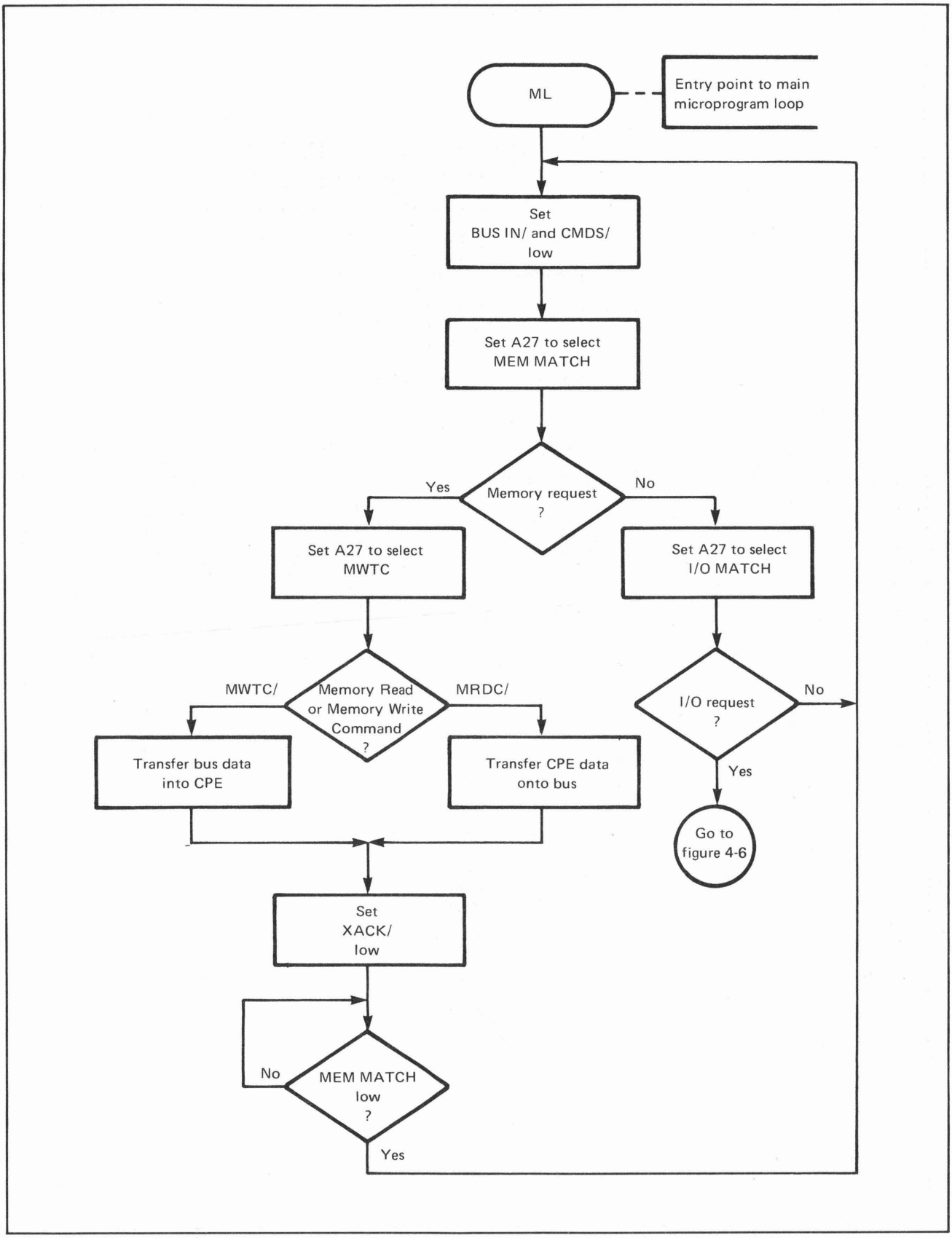


Figure 4-5. Main Microprogram Loop

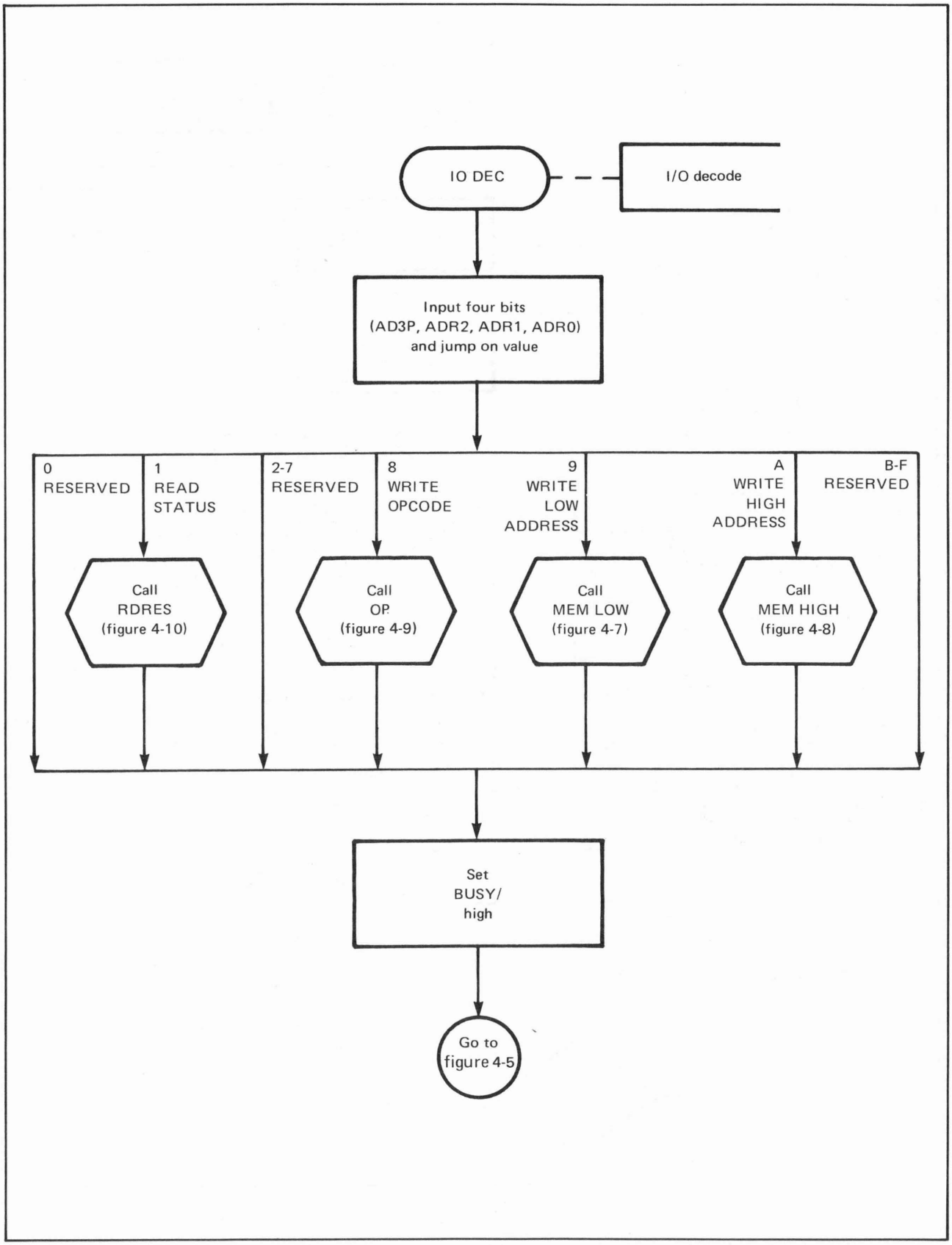


Figure 4-6. I/O Input Decode

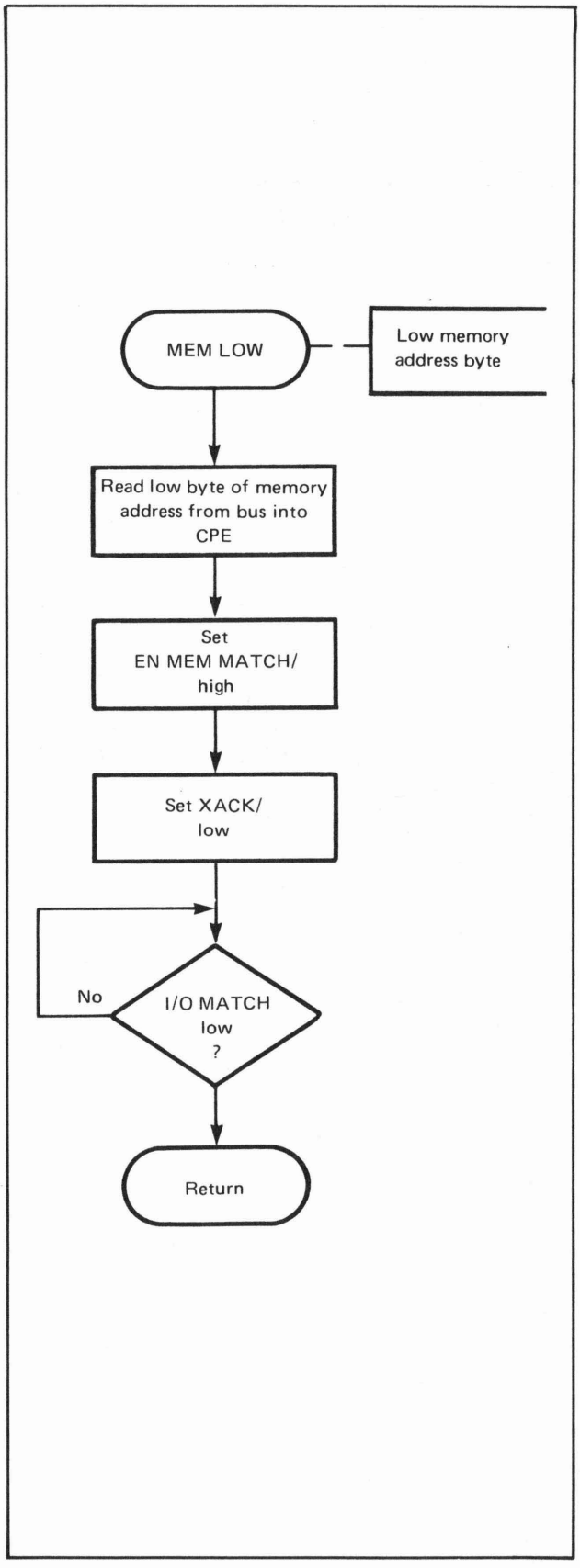


Figure 4-7. Accept Low Byte of Memory Address

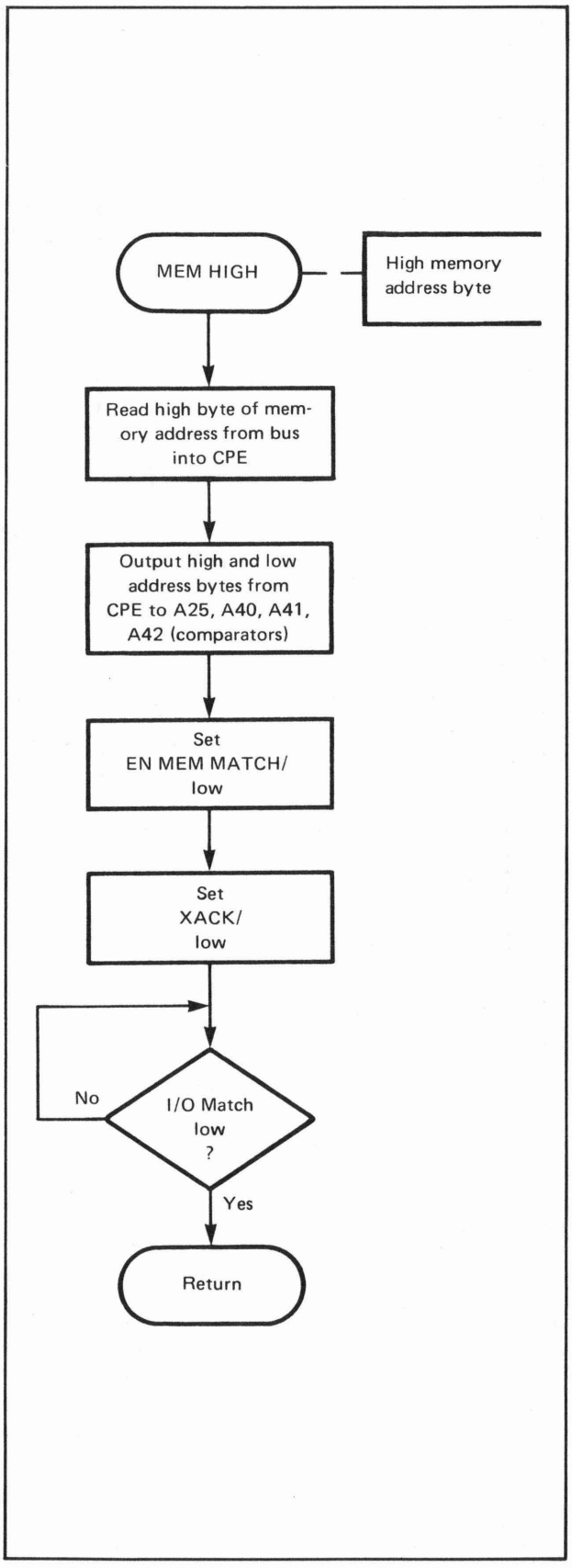


Figure 4-8. Accept High Byte of Memory Address

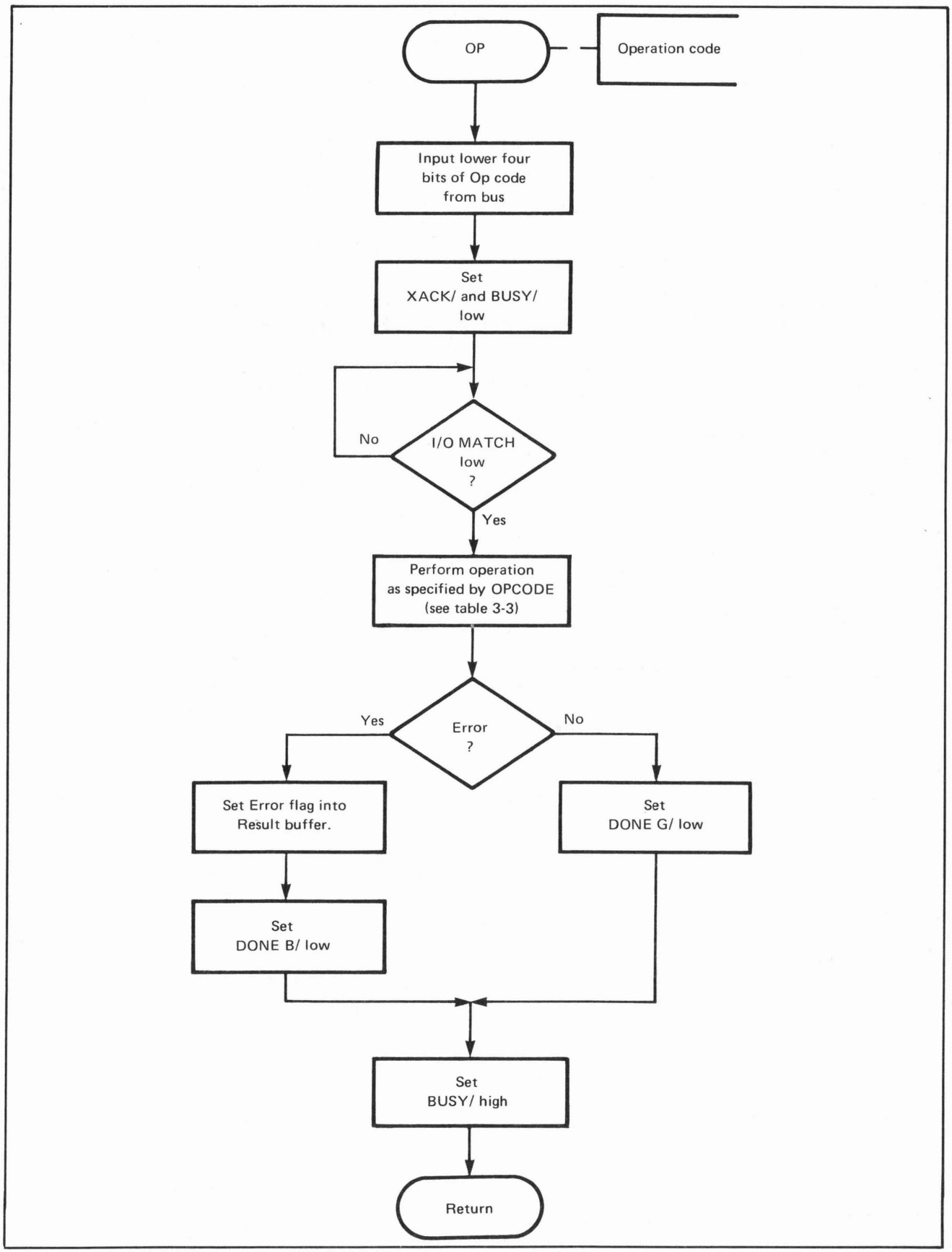


Figure 4-9. Execute Operation (OP) Code

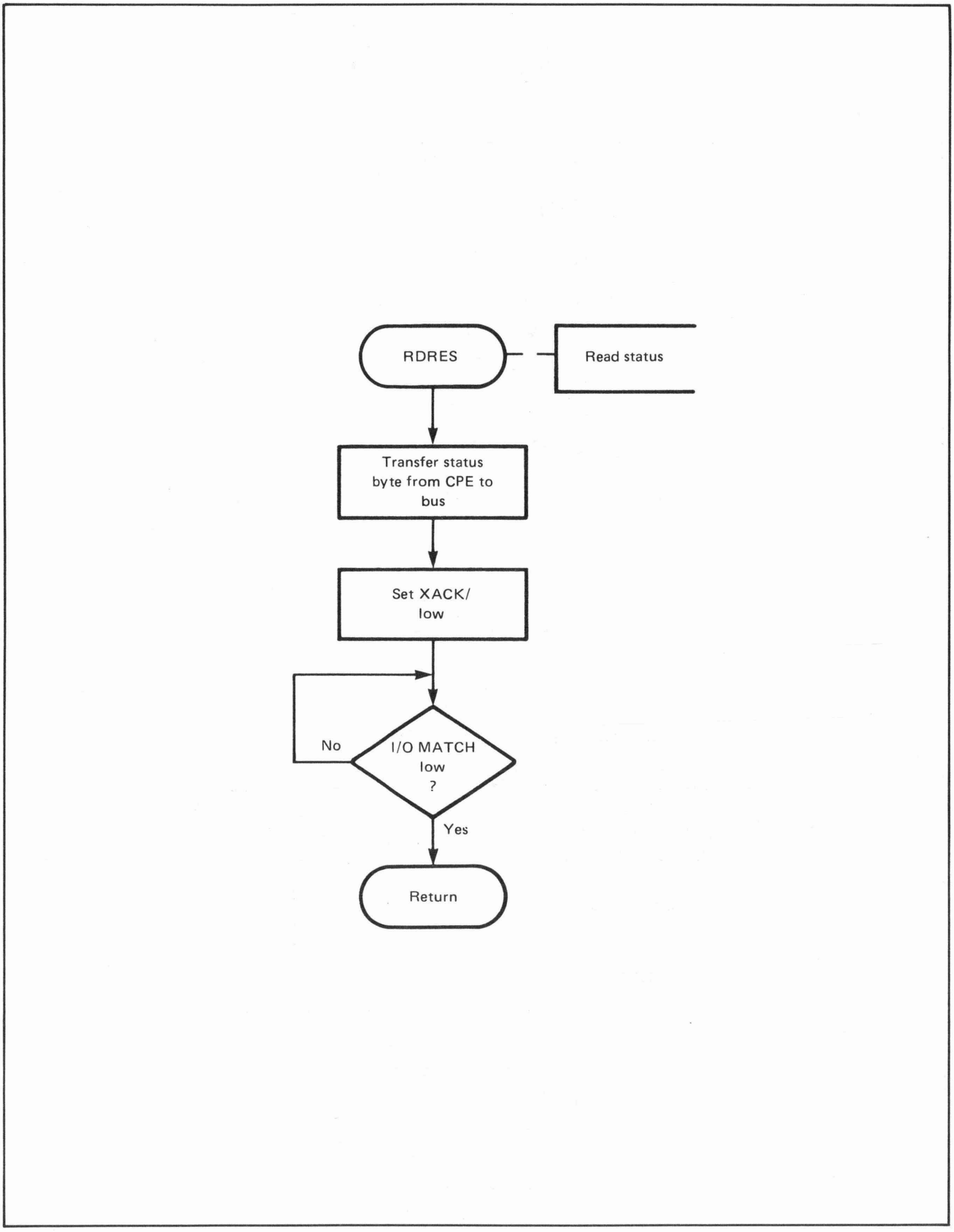


Figure 4-10. Read Status Byte

CHAPTER 5
SERVICE INFORMATION

5-1. INTRODUCTION

This chapter provides service diagrams and reshipment instructions.

5-2. SERVICE DIAGRAMS

The Math Unit parts location diagram and schematic diagram are given in figures 5-1 and 5-2, respectively. The schematic diagram consists of five sheets, each of which includes grid coordinates. Signals that transverse from one sheet to another are assigned grid coordinates at both the signal source and signal destination. For example, the grid coordinates 5ZB1 locate a signal source (or signal destination) on sheet 5 in zone B1.

A signal mnemonic that ends with a virgule (slash, slant, solidus) denotes that the signal is active low ($\leqslant$ 0.4V). Conversely, a signal mnemonic without a virgule denotes that the signal is active high ($\geqslant$ 2.4V).

5-3. RESHIPMENT

If the product is being shipped to Intel for service or repair, contact the Technical Support Center at the telephone number listed below to obtain a Return Authorization Number and further instructions. If the product was damaged during shipment from Intel, or if the product is out of warranty, a purchase order will be required to complete the repair.

Use original factory packaging material, if available, to ship the product to the Technical Support Center. If original packaging is not available, wrap the product in a cushioning material such as Air Cap TH-240 (or equivalent) manufactured by the Sealed Air Corporation, Hawthorne, N.J., and enclose in a corrugated carton suitable for shipping. Seal the shipping carton securely, mark it "FRAGILE" to ensure careful handling, and address it to:

Intel Corporation
Technical Support Center
3065 Bowers Avenue
Santa Clara, California 95051
Telephone: (408) 246-7501
TWX: 910-338-0026
TELEX: 34-6372

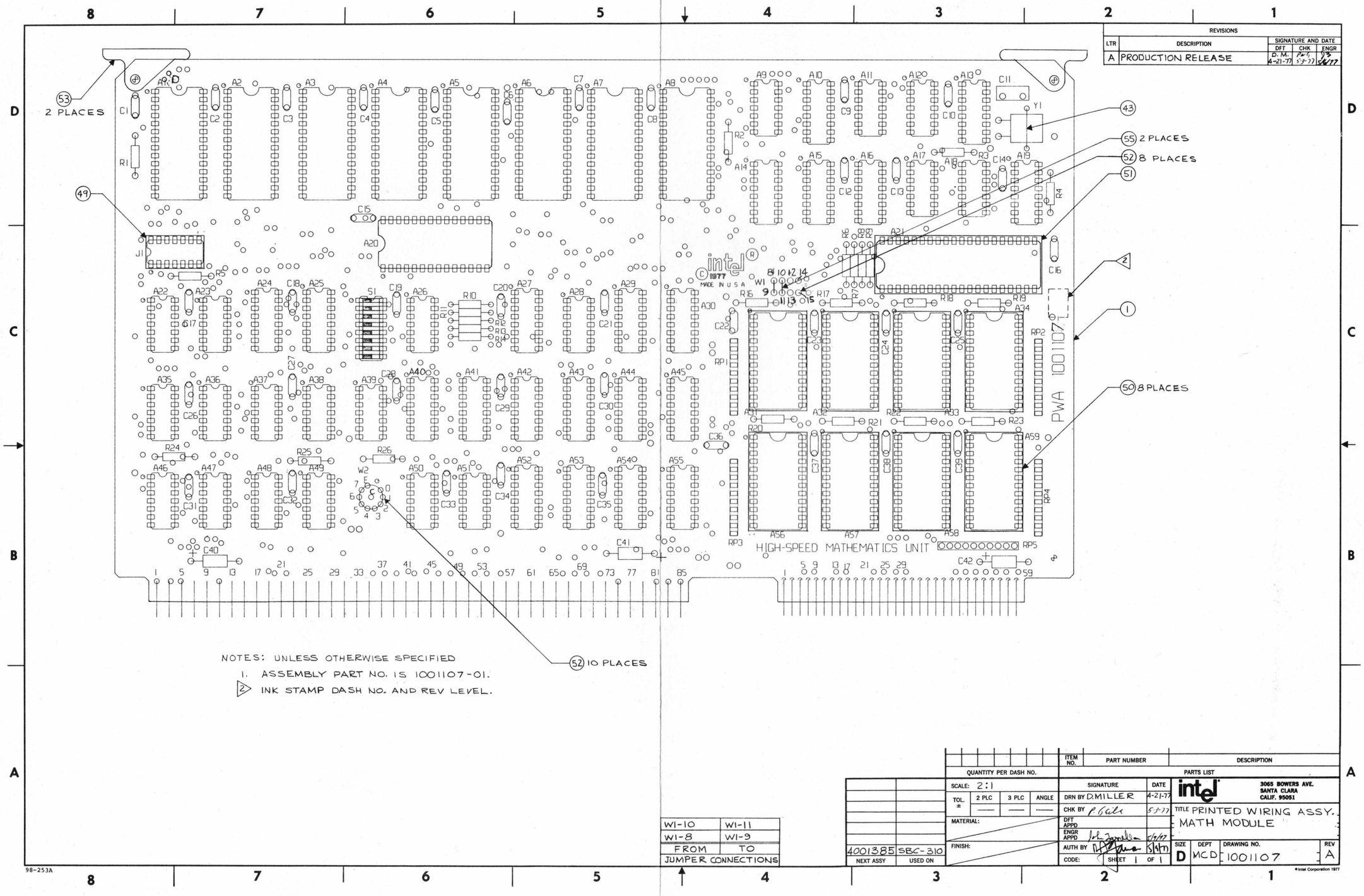


Figure 5-1. SBC 310 Parts Location Diagram

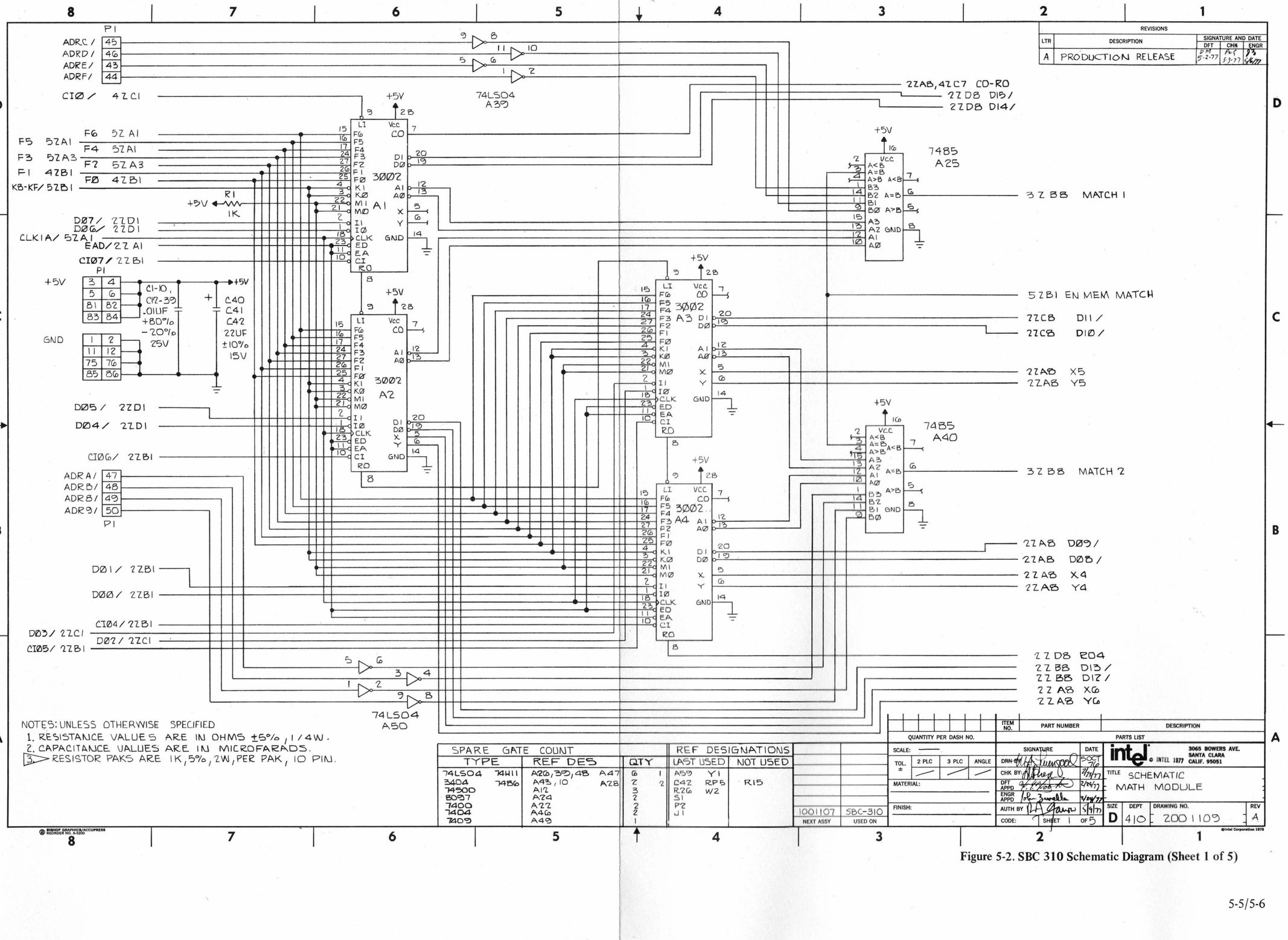


Figure 5-2. SBC 310 Schematic Diagram (Sheet 1 of 5)

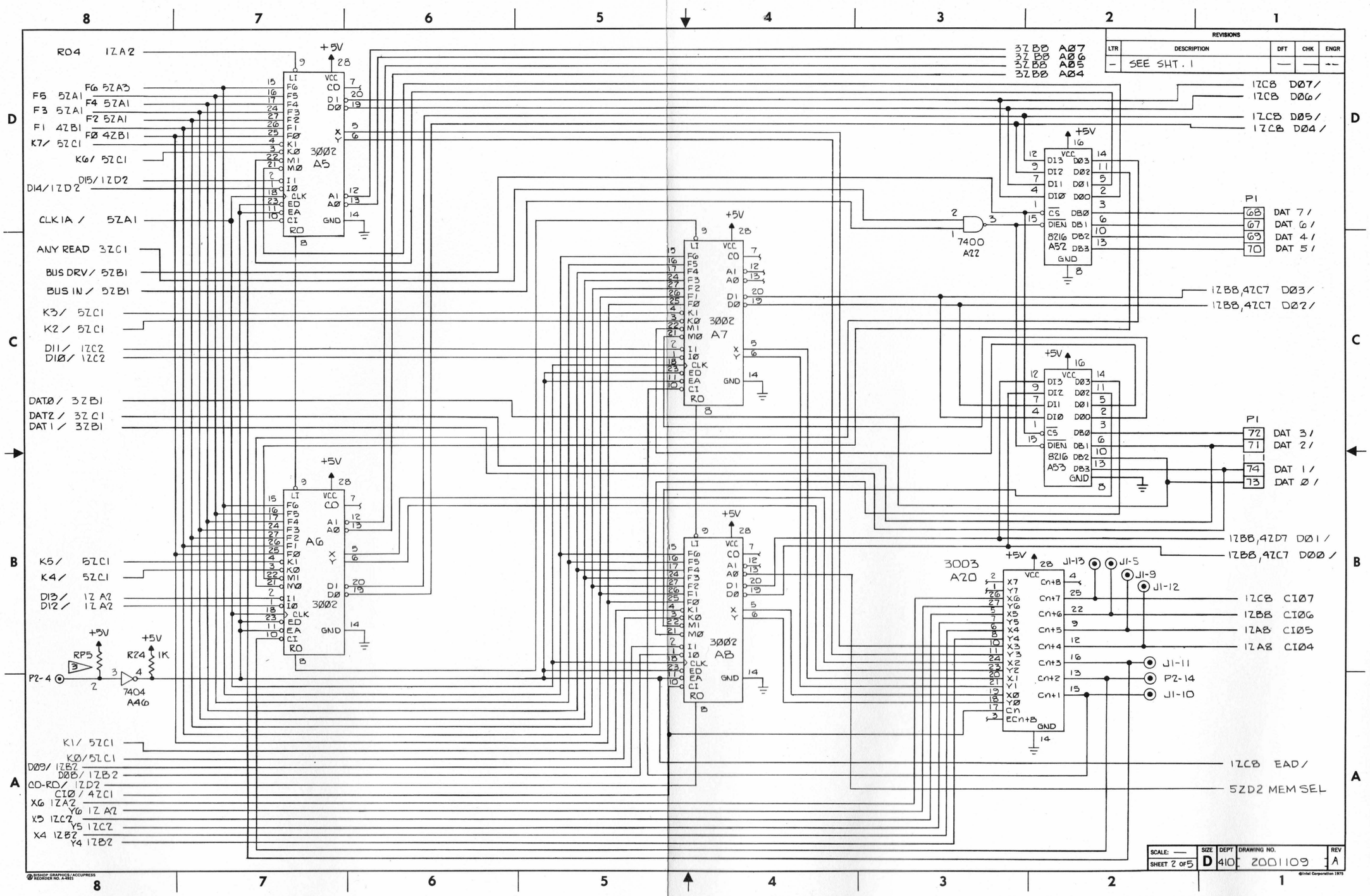


Figure 5-2. SBC 310 Schematic Diagram (Sheet 2 of 5)

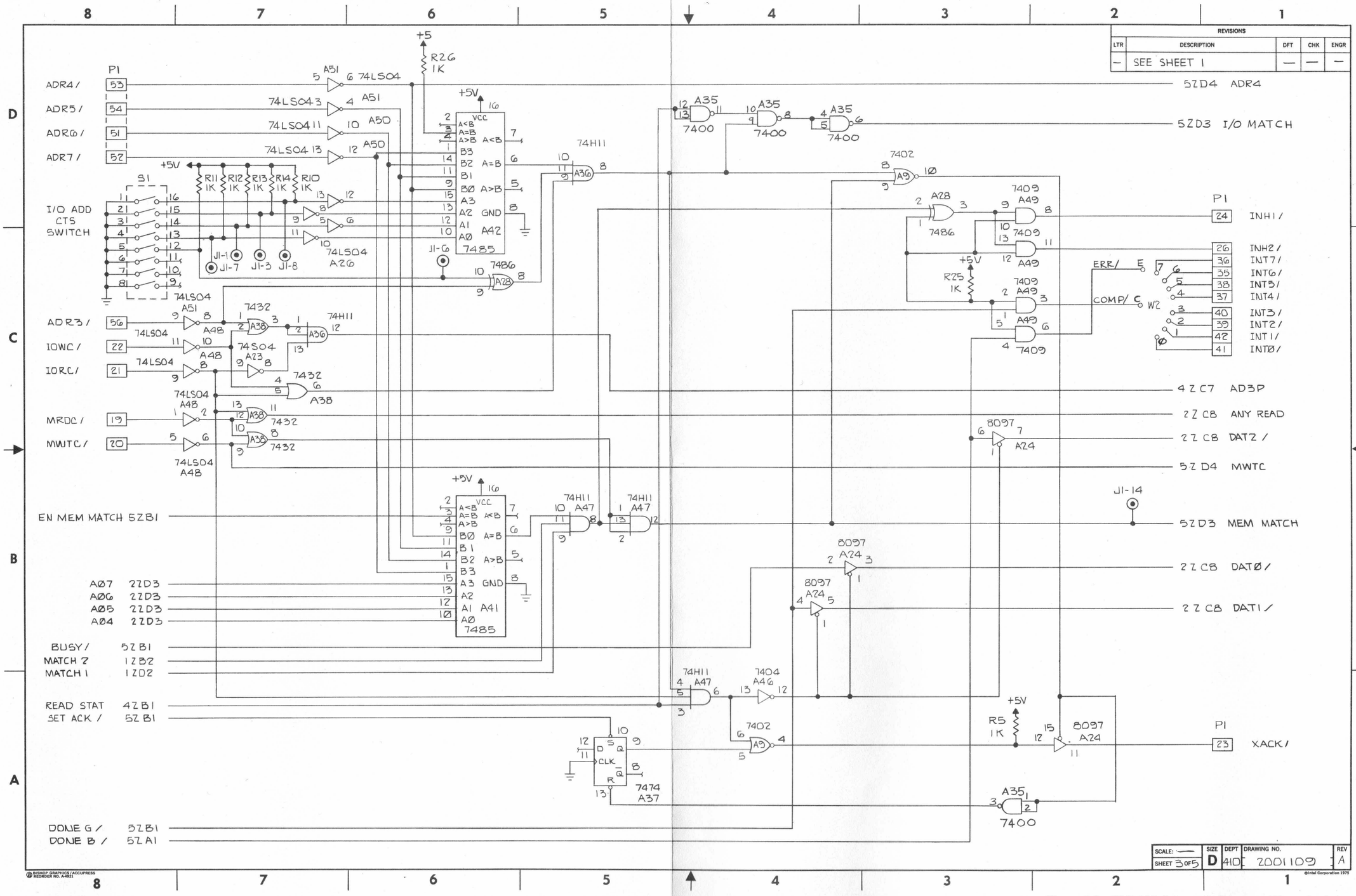


Figure 5-2. SBC 310 Schematic Diagram (Sheet 3 of 5)

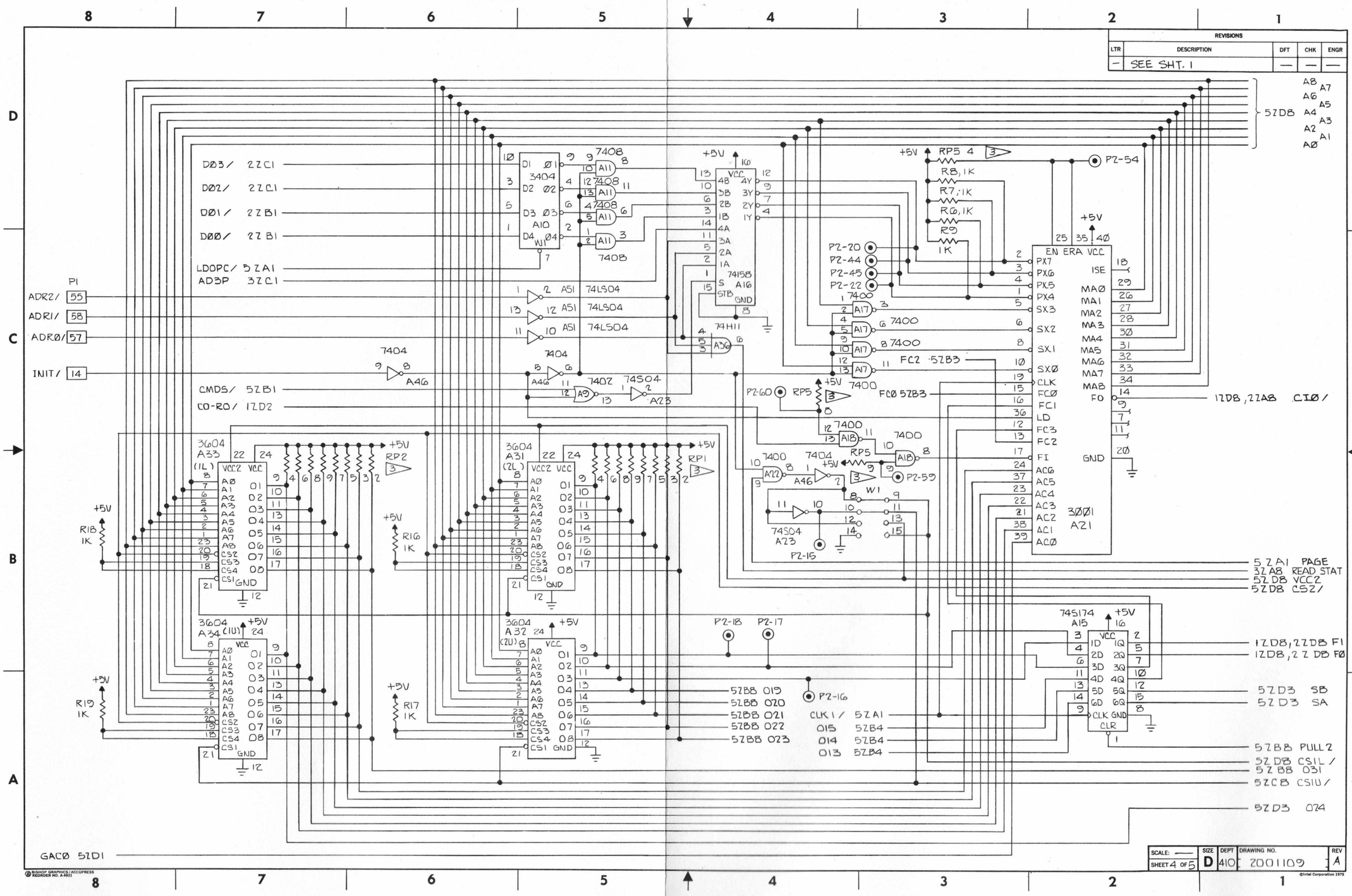


Figure 5-2. SBC 310 Schematic Diagram (Sheet 4 of 5)

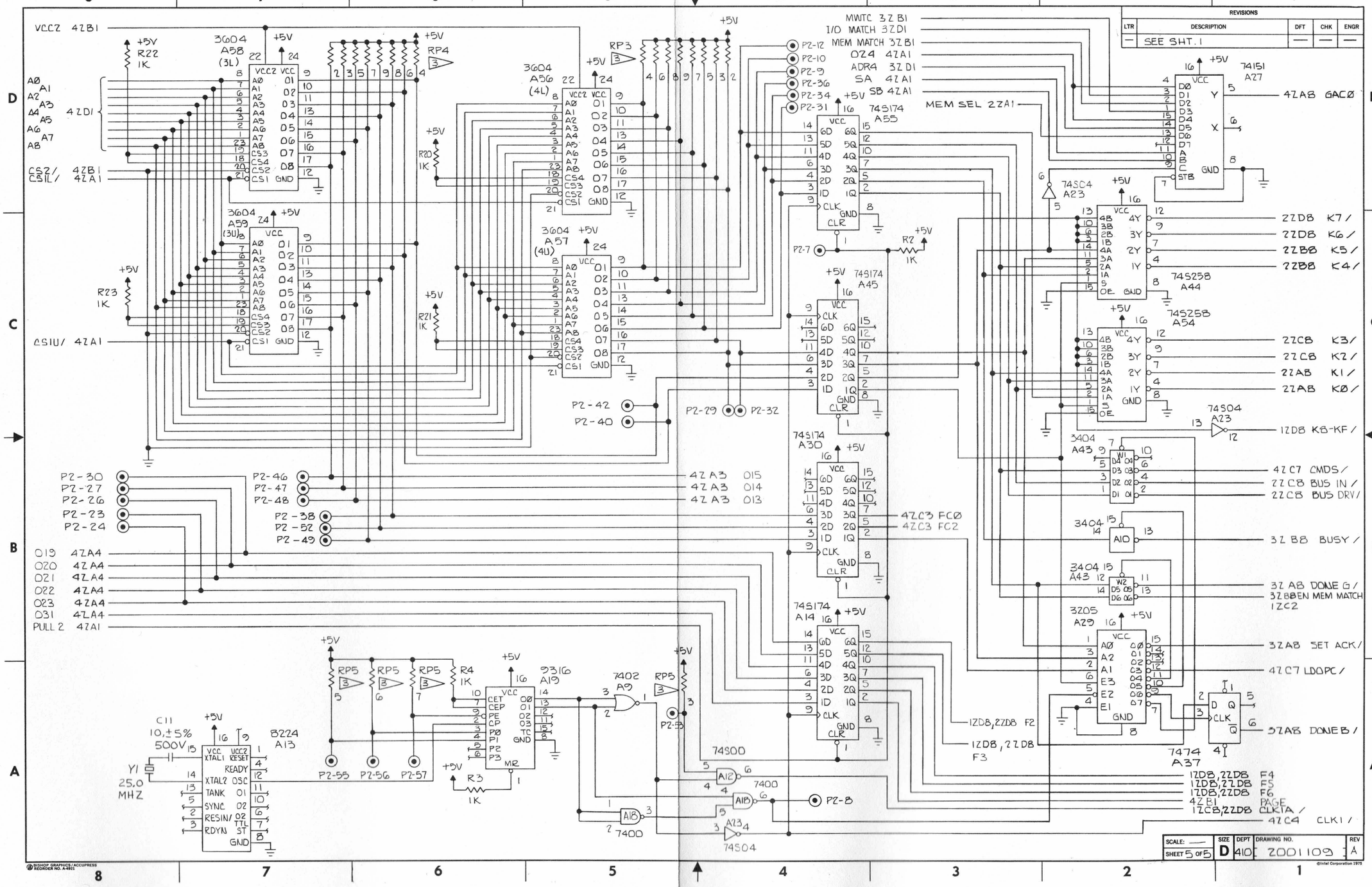


Figure 5-2. SBC 310 Schematic Diagram (Sheet 5 of 5)

intel®

REQUEST FOR READER'S COMMENTS

The Microcomputer Division Technical Publications Department attempts to provide documents that meet the needs of all Intel product users. This form lets you participate directly in the documentation process.

Please restrict your comments to the usability, accuracy, readability, organization, and completeness of this document.

1. Please specify by page any errors you found in this manual.

2. Does the document cover the information you expected or required? Please make suggestions for improvement.

3. Is this the right type of document for your needs? Is it at the right level? What other types of documents are needed?

4. Did you have any difficulty understanding descriptions or wording? Where?

5. Please rate this document on a scale of 1 to 10 with 10 being the best rating. _____

NAME _____ DATE _____

TITLE _____

COMPANY NAME/DEPARTMENT _____

ADDRESS _____

CITY _____ STATE _____ ZIP CODE _____

Please check here if you require a written reply. ☐

WE'D LIKE YOUR COMMENTS . . .

This document is one of a series describing Intel software products. Your comments on the back of this form will help us produce better software and manuals. Each reply will be carefully reviewed by the responsible person. All comments and suggestions become the property of Intel Corporation.

First Class
Permit No. 1040
Santa Clara, CA

BUSINESS REPLY MAIL
No Postage Stamp Necessary if Mailed in U.S.A.

Postage will be paid by:

Intel Corporation
3065 Bowers Avenue
Santa Clara, CA 95051

Attention: MCD Technical Publications

INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, California 95051 (408) 246-7501

Printed in U.S.A./S272/0877/2K CP