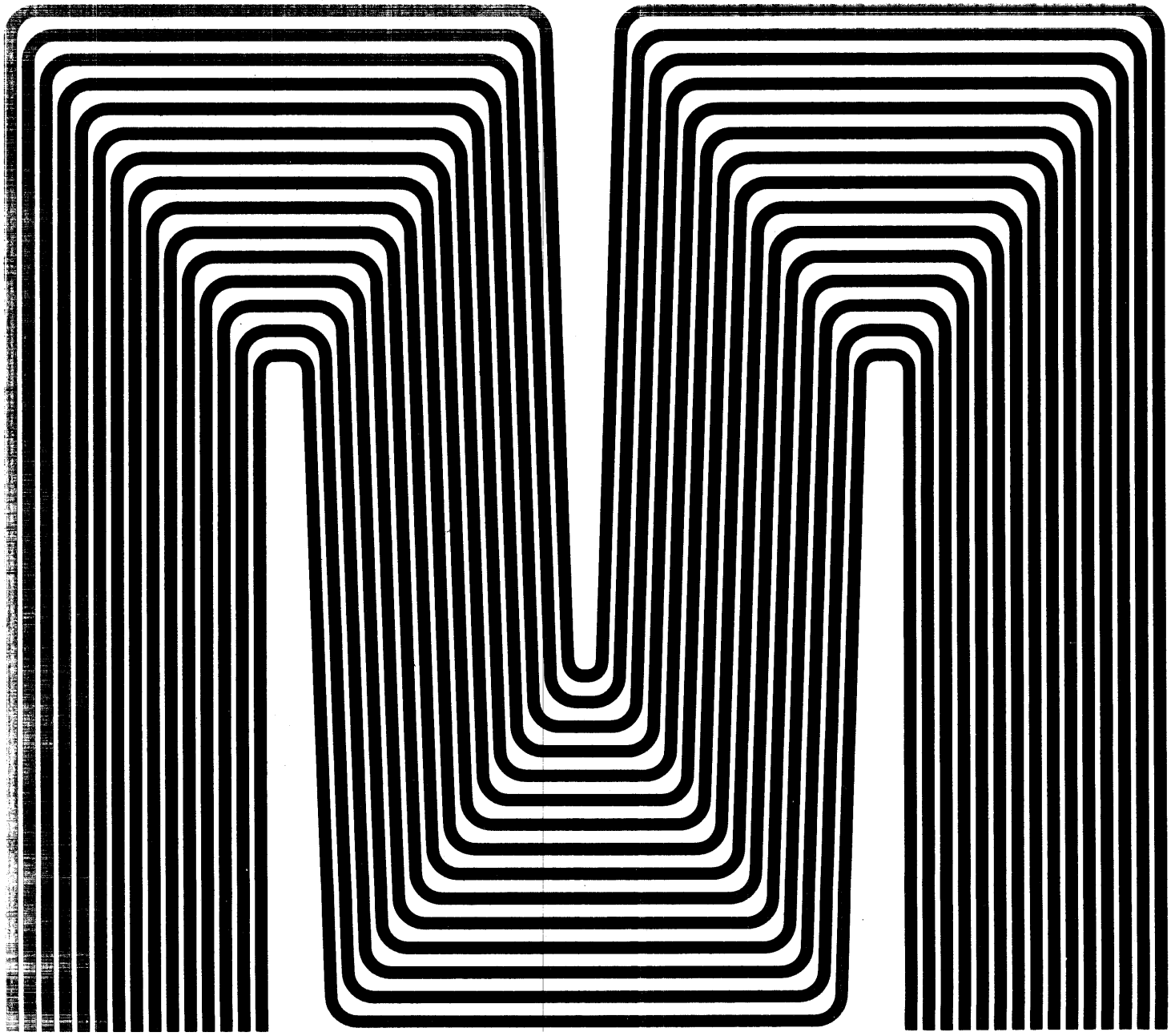
Microdata

Micro 1600 Computer Interface Manual



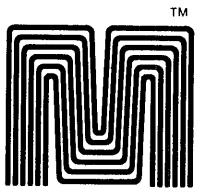
$7.50

INTERFACE AND INSTALLATION MANUAL MICRO 1600 SERIES COMPUTERS

IM20001600

September 1972

Microdata



Microdata Corporation
17481 Red Hill Avenue
Irvine, California 92714
(714) 540-6730
TWX: 910-595-1764

98820 76 1014A

PRINTED IN U.S.A.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION 1

GENERAL INFORMATION

1.1 INTRODUCTION

This manual describes the three primary input/output interfaces of the Micro 1600 series computers and provides guidelines for connecting external equipment to the interfaces. The information in this manual should not be considered the final authority for the I/O interface configuration of any particular machine. The final authority in all cases is the latest revision of all applicable engineering drawings and specifications.

Input/output operations for all Micro 1600 series computers are described in the manual. In the Micro 1600, I/O operations are performed under control of microcommands initiated by macro instructions. A standard set of I/O macro instructions is used for performing program-controlled and concurrent I/O operations.

Information in the manual is arranged in sections:

Section 1 - provides general information on the various Micro 1600 I/O interface systems.

Section 2 - provides a detailed description of the programmed byte I/O interface system, including programmed transfers, block automatic (concurrent) transfers, and external interrupts.

Section 3 - provides a detailed description of the Direct Memory Access (DMA) Port.

Section 4 - provides a detailed description of the Serial I/O Channel.

Section 5 - provides detailed information on the mechanical construction of the Micro 1600 computer systems and system installation.

Appendices - contain reference information:

A Backplane Connector Signal Lists
B I/O Interface Signal Glossary
C Teletype Modification Procedures
D I/O Controller Design Notes

1.2 I/O SYSTEM ORGANIZATION

Figure 1-1 is a block diagram of a typical Micro 1600 series computer system showing (in shaded blocks) the three primary I/O interfaces:

- Serial I/O Channel
- Parallel Byte I/O Channel
- Direct Memory Access (DMA) Port

NOTE

The integral teletype controller shown in the figure is actually interfaced to the parallel byte I/O channel but is located on a special interface board which is also used to interface the computer backplane and control panel.

These three interfaces provide the system designer with the flexibility to structure efficient I/O systems for a wide range of applications. The serial I/O interface, although most commonly used with a teletype, can be used for other bit-serial devices as well. The byte I/O interface can be used by controller circuit boards that plug into the mainframe chassis. It can also be cabled to an expansion chassis via line driver/receiver circuitry. In the configuration shown in Figure 1-1, an expansion chassis is used for additional controller and optional Priority Interrupt circuit boards. The Priority Interrupt board can also be located in the mainframe chassis. The DMA interface provides the facility for external I/O devices to communicate directly with core memory. The user can design his own interface controller for the DMA Port or use standard Microdata DMA interfaces.

1.3 SERIAL I/O INTERFACE

The serial I/O interface is designed primarily for communicating with a full duplex teletype. Character assembly and disassembly, including all timing and synchronization, are performed at the microprogram level. Two macro instructions, Input Byte Serially (IBS) and Output Byte Serially (OBS), are used for communicating with the serial channel device.

It should be noted that the Micro 1600 firmware for these instructions was originally designed to operate with Micro 800 computers for control of a 110-baud teletype. However, because of the higher speed of the Micro 1600, this standard firmware operates faster than 110 baud and can not, therefore, be used for this purpose as supplied. The system designer can alter the timing of the serial channel for compatibility with a Teletype or other serial device by performing a simple change in firmware or by slowing the Micro 1600 basic clock to the speed of the Micro 800.

1.4 BYTE I/O INTERFACE

The byte I/O interface provides the facility for transferring bytes over a party line I/O bus under microprogram control. The standard Microdata 1600 computer firmware provides both programmed I/O and concurrent I/O transfer capability, along with a priority interrupt system.

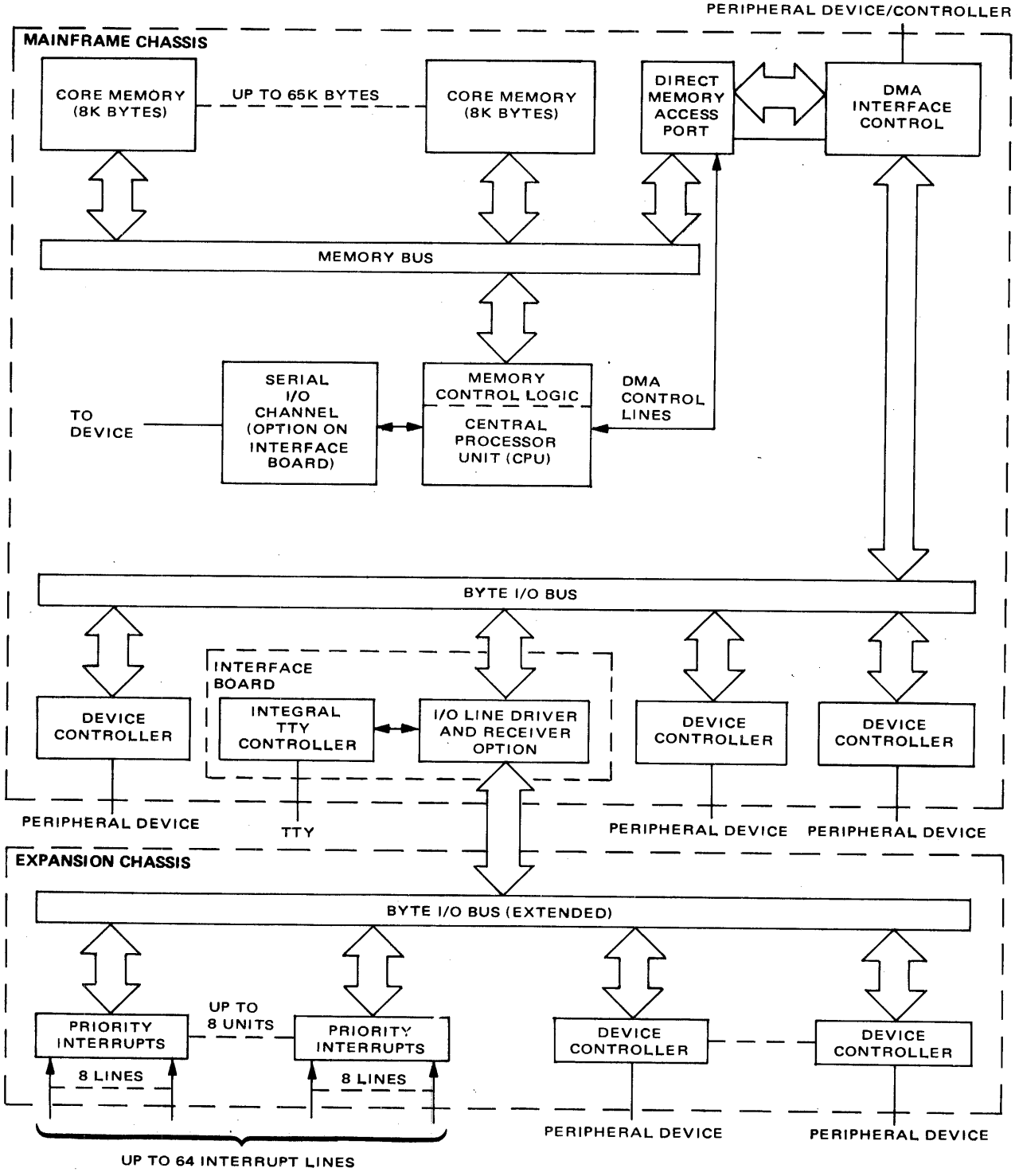


Figure 1-1. Typical Micro 1600 Series I/O Configuration

Data transfers through the byte I/O interface are basically two-phase operations. During the first phase, a control byte is placed on the byte I/O bus before the actual transfer of data. The control byte contains a device address specifying the address of one of the I/O controllers connected to the bus, and a device order code signifying the type of operation to be performed during the transfer (data transfer, status/function transfer, etc.).

All controllers on the bus examine the device number, but only the controller with a matching address accepts the control byte and logically connects itself to the bus for the subsequent data byte transfer. During the second phase of the byte I/O operation a single byte is transferred to or from the I/O controller. After each byte transfer, the controller disconnects itself from the bus.

Figure 1-1 shows that I/O controllers can be connected to the byte I/O interface at the I/O bus in the mainframe chassis or to the extended I/O bus in a separate expansion chassis. The external byte I/O bus is an extension of the internal bus brought out of the mainframe chassis through line drivers and receivers on the Interface Board.

The Interface Board is supplied with the standard computer and contains the logic to interface the operator's control panel to the CPU and the Integral Teletype Controller. If the system includes an expansion chassis to house I/O controllers, the I/O bus expansion line drivers and receivers are also located on the Interface Board.

1.4.1 PROGRAM-CONTROLLED I/O

Two basic instructions (one for input and one for output) are used for transferring information to and from controllers on the byte I/O bus under programmed control. These instructions permit transfers between the device controller and the A Register, B Register, or Memory. Up to eight types of input and eight types of output instructions may be defined for a particular controller. Generally these include function output, data output, status input, and data input, and are determined by a 3-bit device order in the control byte of the I/O instruction.

1.4.2 CONCURRENT I/O

The concurrent I/O feature provides the capability for automatic block transfers between core memory and I/O controllers connected to the byte I/O interface. The concurrent mode transfer rate is a function of the firmware set used in the computer. As an example, standard Micro 1600/20 firmware performs concurrent transfers at rates up to 20,000 bytes per second.

Once started, the transfers are fully automatic and proceed without program intervention. Concurrent I/O operations take priority over instruction execution and force a break in the execution of long instructions such as multiply, divide, and shifts to ensure that concurrent I/O servicing delays are not excessive. Concurrent I/O operations make use of pairs of two-byte address control words stored in dedicated core memory locations. One

pair of address words is used by each controller. The control words, which contain the address of the current byte being transferred and the address of the last byte in the block, are initially set by the software program and thereafter are manipulated automatically by firmware for each byte transferred.

1.4.3 EXTERNAL PRIORITY INTERRUPTS

The external interrupt system of Micro 1600 series computers operates through the byte I/O interface in both the computer mainframe and expansion chassis. Interrupts can originate from device controllers or from the optional Priority Interrupt interface board connected to the byte I/O bus. The Priority Interrupt interface board provides control of eight external interrupt signals.

The byte I/O interface contains a single external interrupt request line common to all I/O controllers on the byte I/O bus and a priority line that is carried sequentially through all controllers on the bus. Each I/O controller receives priority from the preceding controller in the priority chain and passes it along to the next controller in line if it is not ready to request an interrupt. When a controller receives priority and is ready to request an interrupt, it stops the progression of the priority signal and activates the interrupt request signal.

After receiving acknowledgment of the interrupt request, the interrupting controller places an address byte on the I/O bus that the processor uses to transfer program control to the proper interrupt servicing routine.

1.5 DIRECT MEMORY ACCESS PORT

All Micro 1600 series computers contain a Direct Memory Access (DMA) Port through which data can be transferred between core memory and I/O devices at rates up to one million bytes per second. The DMA Port provides this high transfer rate and low access latency time for use with high-speed, demand-type devices such as rotating memories.

The DMA Port consists of the memory bus and various DMA memory control lines which are available in the computer mainframe. It is up to an external DMA controller to manipulate the control lines and place data and addresses on the memory bus at the proper times when DMA transfers take place. The DMA controller may be designed and constructed by the user, or a standard Microdata DMA controller can be used.

The standard programmed I/O instructions are used to set up the DMA Controller and, if so designed, I/O device controller(s) attached to the DMA Controller with the parameters of the transfer. Program communication with these device controllers takes place over the byte I/O bus. Once the transfer is initiated, the DMA Controller and attached device controller supervise the transfer, and only minimum attention from the microprogram is required. Standard Microdata DMA Controllers can accommodate several external device controllers.

1.6 PHYSICAL CONSTRUCTION

A variety of Micro 1600 computer mainframes, expansion chassis, and chassis enclosures are available to permit the most convenient and effective physical configuration of each individual system. In each case, a number of I/O controllers can be installed in the mainframe along with Priority Interrupt boards and one DMA Controller. If the number of I/O interfaces exceeds the capacity of the mainframe, an expansion chassis is used to accommodate the additional device controllers.

Detailed descriptions of the numerous physical configurations are provided in Section 5 of this manual.

SECTION 2

BYTE I/O INTERFACE

2.1 INTRODUCTION

The byte I/O interface, to which the parallel-byte peripheral device controllers connect, contains input control lines, input data lines, output control lines, output data lines and spare lines. The primary byte I/O interfaces are the internal byte I/O bus (located in the Micro 1600 computer mainframe chassis) and the extended byte I/O bus (located in the expansion chassis). The buses are presented to device controllers at the I/O card connectors in the chassis backplanes. The only parallel-byte controller which does not connect into a standard I/O bus card slot is the Integral Teletype Controller (located on the interface board).

The external byte I/O bus is actually an extension of the internal bus in the mainframe. Line driver and receiver logic, mounted on the Interface Board in mainframe card slot J1 and on the I/O expansion interface board which plugs into the expansion chassis, provides the additional drive capability necessary to extend the I/O bus to the controllers in the expansion chassis. The signals on the extended bus are identical to those on the internal bus and carry the same mnemonic designations.

Figure 2-1 shows the points of origin or destination in the CPU of all lines in the byte I/O interfaces. Also shown are the relationships between the two buses and the line driver and receiver logic on the interface and expansion interface boards.

2.2 BYTE I/O BUS

The following paragraphs describe the input and output data and control lines of the byte I/O bus. Unless stated otherwise, the descriptions apply to both the internal and external buses.

2.2.1 INPUT DATA LINES

Input data lines ID00/ through ID07/ are terminated on the CPU B bus by a termination network. The lines are driven by power gates (7438 TTL, or equivalent) with uncommitted collectors on each controller. Because of this termination network, the lines swing only between ground and +3 volts. Each line is at +3 volts when all gates on the lines are off. When a gate is switched on, the line to which it connects swings to ground potential and places a logical 1 on the B bus.

The input data lines are handled the same whether the device controller is located in the mainframe or in the expansion chassis.

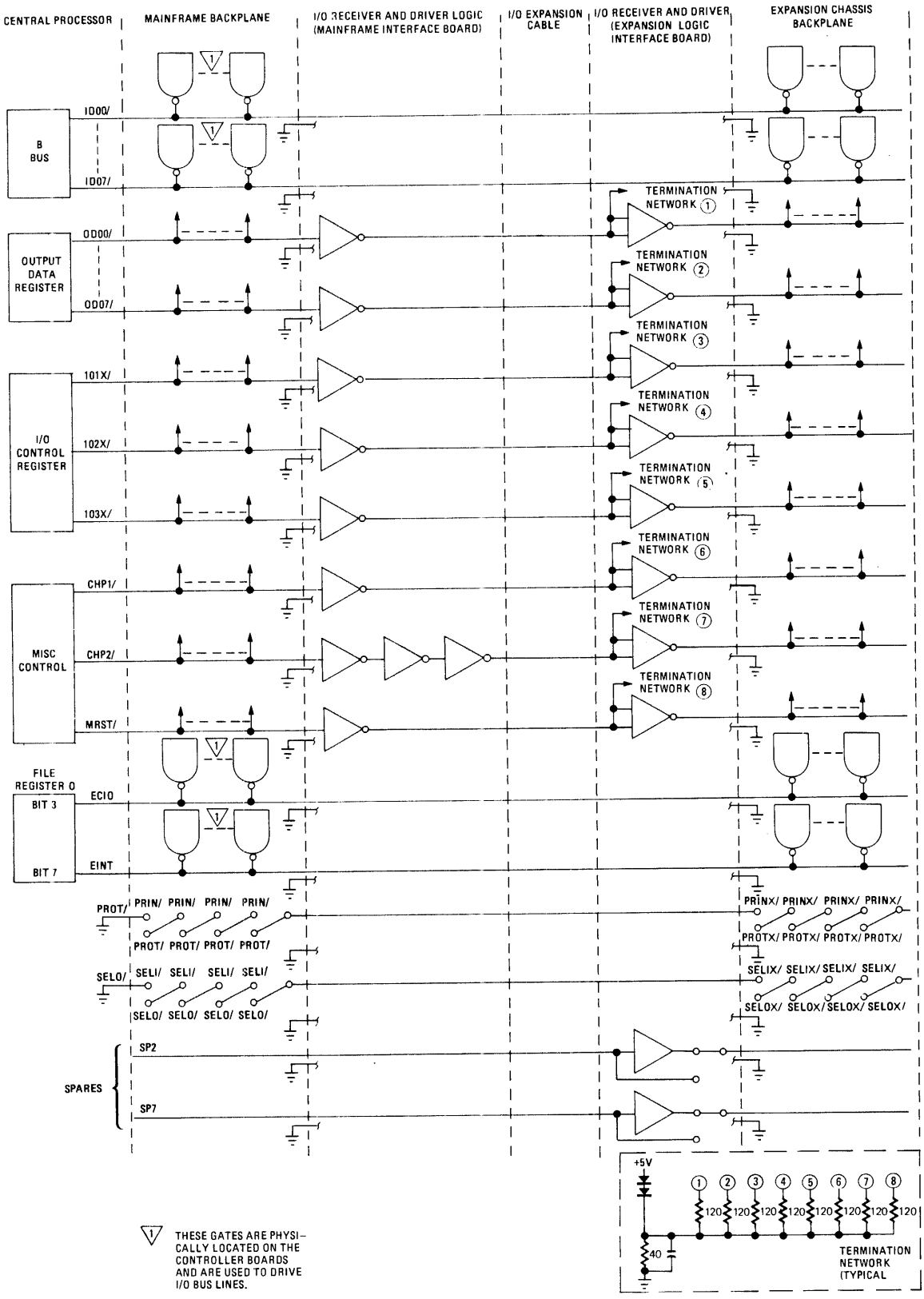


Figure 2-1. Byte I/O Interface Simplified Logic Diagram

2.2.2 OUTPUT DATA LINES

Output data lines OD00/ through OD07/ originate at the processor Output Data Register. Data or address information to be transferred over the output data lines is transferred from the A Register, the B Register, or Memory into the Output Data Register and out over the lines. Lines OD00/ through OD07/ are present at all the CPU I/O backplane connectors, are buffered on both the interface and expansion interface boards, and are sent to the expansion chassis backplane via the I/O expansion cable.

To preserve the expansion capability of the byte I/O bus, each device controller on the bus is restricted to a single unit load (one TTL gate, or 1.6 ma maximum) on each of the output data lines. Two loads are allowed if one load is a low power TTL gate (i.e., 74L04).

Each of the OD0X/ lines has the following characteristics:

Output Data Register	OC0X/ in Mainframe	OD0X/ in Expansion Chassis
Binary 1	0 V	0 V
Binary 0	+4 V (nom)	+3 V (nom)

2.2.3 INPUT CONTROL LINES

The input control lines on the byte I/O bus are:

- ECIO/ - Concurrent I/O request
- IRPY/ - I/O Reply (Spare)
- EINT/ - External Interrupt

These lines are present in the I/O card connectors in the mainframe backplane and are routed to the expansion chassis backplane by the I/O expansion cable. Line IRPY/, however, is available only in the mainframe backplane. The lines are driven by power gates (7438 TTL, or equivalent) with uncommitted collectors in each controller. A termination network for each line is included in the processor. Because of the termination network, the lines swing only between ground and +3 volts. All lines are active (indicate assertion) when they are at ground potential. For example; ground potention on the line EINT/ causes an external interrupt request. (The input control lines are defined in Appendix B.)

2.2.4 OUTPUT CONTROL LINES

The output control lines on the byte I/O bus are:

- IO1X/ -
- IO2X/ - } I/O control bits 1 through 3 from the I/O Control Register
- IO3X/ -
- CPH1 - Processor clock
- CPH2/ - Processor clock (inverted and delayed 33 nanoseconds from CPH1)
- MRST/ - Master reset
- PRIN/ - Priority in
- PROT/ - Priority out
- SELI/ - Select in
- SELO/ - Select out

The output control lines on the external byte I/O bus in the expansion chassis are identical in characteristics and mnemonic designations to the internal byte I/O bus in the mainframe. Each of these output control lines is defined in Appendix B, and the usage of each is described in the following paragraphs

2.2.4.1 CONTROL LINES IO1X/ THROUGH IO3X/. These control lines to the inverted outputs of the I/O Control Register in the CPU. This register is set and reset at the micro-command level. Device controllers connected to the I/O bus decode these lines into eight states which are assigned meanings to indicate various I/O control modes. Table 2-1 provides standard definitions of the eight control flip-flop states. Other definitions can be devised for systems not using standard Microdata firmware and I/O controllers.

Subsequent discussions in this manual refer to the conditions of IO1X/ through IO3X/ by the logic terms assigned to the eight states of these lines (COXX/, DOXX/, etc.) which are decoded in the I/O controllers.

Table 2-1. I/O Control States

I/O CONTROL REGISTER STATE	CONTROL DEFINITION	LOGIC TERM
0	None	None
1	Control output	COXX/
2	Data output	DOXX/
3	Space serial Teletype	SP1
4	Concurrent I/O acknowledge	CACK/
5	Interrupt acknowledge	IACK/
6	Data input	DIXX/
7	Spare	SP3/

2.2.4.2 LINES CPH1 AND CPH2/. These lines provide processor clock signals to device controllers. Each line can be used independently as a square wave source (5 MHz), or they may be used together in a NAND gate to produce a 33-nanosecond clock pulse (CPH1 is inverted and delayed 33 nanoseconds to form CPH2/). The relationship of the signals on the lines CPH1 and CPH2/ is shown in Figure 2-2.

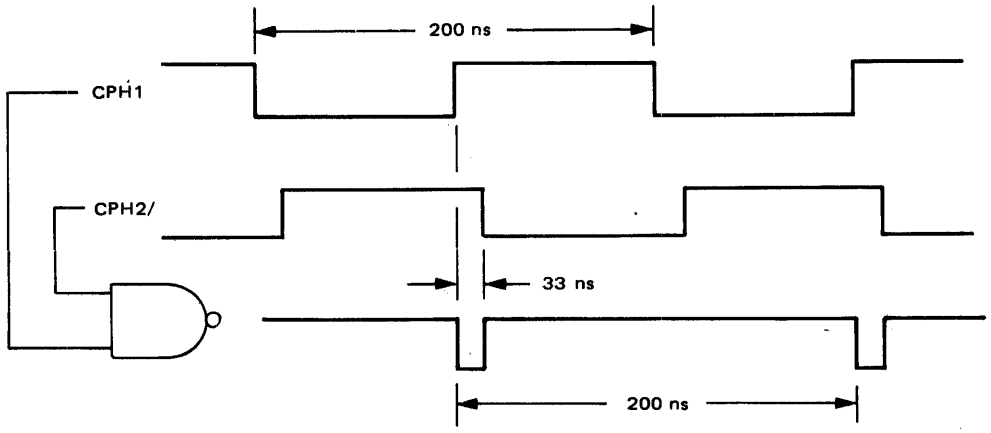


Figure 2-2. Relationship of Control Signals CPH1 and CPH2/

2.2.4.3 CONTROL LINE MRST/. This line is the master reset line. It is activated by the RESET switch on the system front panel, or by the power fail circuit during power failure or restart. It is used to clear all control flip-flops to their initialized conditions. Ground potential is applied to this line when the RESET switch is pressed.

2.2.4.4 CONTROL LINE PROT/, PRIN/. This line carries interrupt request priority from controller to controller along the mainframe backplane and then from controller to controller along the expansion chassis backplane. The line is labeled PROT/ (priority out) as it leaves the CPU and each controller, and enters each controller as term PRIN/ (priority in).

Relative priority of each controller is determined by the positions of the controller boards. The first controller in the mainframe (nearest the CPU) has highest mainframe priority with the last controller having lowest mainframe priority. Next, in decreasing order of priority, are the expansion chassis controllers from J2 down to the last controller in the chassis.

2.2.4.5 CONTROL LINE SELO/, SELI/. This line carries external interrupt and concurrent I/O select priority from controller to controller along the mainframe backplane and from controller to controller along the expansion chassis backplane. It is labeled SELO/ (select out) as it leaves the CPU and each controller, and enters each controller as term SELI/ (select in).

Selection priority of the controllers is the same as the interrupt request priority (Paragraph 2.2.4.4), determined by the positions of the controller boards in the mainframe and expansion chassis.

Note

- A controller must receive PRIN/ to make an external interrupt request. The requesting controller removes PROT/ from all lower priority controllers to lock out lower priority interrupt requests. The requesting controller must receive SELI/ from the preceding controller to respond to an interrupt acknowledgment (IACK/).
- Any controller can make a concurrent I/O request (ECIO/) at any time. Simultaneous concurrent requests are handled in order of priority, as a requesting controller must receive SELI/ in order to respond to the concurrent I/O acknowledgment (CACK/). A controller requesting a concurrent I/O transfer will not pass SELO/ to the next controller until it has transferred one data byte.
- Descriptions of the PROT/, PRIN/, SELO/, and SELI functions are provided in Paragraphs 2.5 through 2.6.3.

2.2.5 SPARE LINES

Spare lines SP2 and SP7 are applied to the internal and external byte I/O bus. They are provided only for special requirements and are not terminated in any way on the standard computer. They are buffered on the expansion interface board for transmission from the mainframe backplane to the expansion backplane. These buffers can be bypassed, by cutting etch and adding jumpers, to drive in the other direction.

2.3 BYTE I/O FUNDAMENTALS

Although the flexibility of the byte I/O system lends itself to customizing for individual applications, certain standard conventions have been adopted for byte I/O operations in the Micro 1600 series computers. These conventions are described in the following paragraphs.

Byte-programmed input/output operations provide transfers of data, control, and status information over the byte I/O bus. This multiplex channel permits intermixed program and concurrent I/O transfers. More than one device on the bus can be operating in a concurrent block transfer mode at the same time. A maximum of 32 controllers can normally be addressed on the byte I/O bus.

The second byte of an I/O instruction is a control byte containing a three-bit device order and a five-bit device address:

DEVICE ORDER (f)			DEVICE ADDRESS (d)				
7	6	5	4	3	2	1	0

Input and output operations are two-phase operations. When an I/O instruction is executed, the control byte is placed on the output data lines before the actual transfer of data. All controllers examine the transmitted device address. The controller whose assigned address is the same as that specified in the control byte, accepts the control byte and sets up for a subsequent data byte transfer according to the device order code (f). The second phase consists of the input or output of a single byte. When a device order does not require a data transfer, the second byte is disregarded by the device controller.

2.3.1 DEVICE ADDRESSES

Each I/O controller on the byte I/O bus is assigned a unique five-bit device address. On Microdata device controllers, standard addresses are assigned by printed circuitry on the controller board. Other addresses may be assigned by cutting the etch and installing jumper wires.

Each device controller on the I/O bus determines if it is being addressed by comparing its assigned address to the five-bit device number in the control byte sent to all controllers on the output data lines. The device address portion of the control byte appears on data lines OD00/ through OD04/. The assigned device address is also used to identify the I/O controller requesting an interrupt or concurrent I/O transfer. The processor acknowledges each request with signal IACK/ (for interrupts) or CACK/ (for concurrent I/O). On receiving the acknowledgment signal, the requesting controller places its address on input data lines ID01/ through ID05/. For concurrent I/O operations, the controller indicates the direction of data transfer by bit ID07/; one indicates output, zero indicates input.

Table 2-2 lists the device addresses assigned to Microdata standard interface units. Customer-designed controllers should not use the addresses assigned to standard Microdata controllers which are to be used. Each priority interrupt group listed in the table is one set of eight priority interrupt levels on the optional (8-level) Priority Interrupt board.

2.3.2 DEVICE ORDERS

Accompanying the five-bit device address in the control byte is a three-bit device order specifying the I/O operation to be performed by the controller. The device order portion of the control byte appears on output data lines OD05/ through OD07/.

Table 2-3 provides a list of standard device orders. Not all device controllers use all orders listed in the table. Their use is dictated by controller design and device requirements. Many controllers have unique device order definitions.

Table 2-2. Standard I/O Device Addresses

ADDRESS (HEXADECIMAL)	I/O DEVICE
00	Teletype (Model 2610 or Integral)
01	Asynchronous Modem or TTY/CRT Controller
02	High Speed Paper Tape Reader
03	High Speed Paper Tape Punch
04	Card Reader
05	Line Printer
06	Unassigned
07	Unassigned
08	Input/Output Expander (32 X 32)
09	Magnetic Tape (1 to 4 Drives)
0A	Magnetic Tape (1 to 4 Drives)
0B	8 Channel Asynchronous Modem Controller
0C	8 Channel Asynchronous Modem Controller
0D	8 Channel Asynchronous Modem Controller
0E	8 Channel Asynchronous Modem Controller
0F	Unassigned
10	Synchronous Modem Controller
11	Synchronous Modem Controller
12	Automatic Call Unit
13	Unassigned
14	Disc Controller (1 to 4 Drives)
15	Disc Controller (1 to 4 Drives)
16	DMA Channel Controller
17	Unassigned
18	Unassigned
19	Unassigned
1A	4 or 8 Channel Communications Controller
1B	4 or 8 Channel Communications Controller
1C	4 or 8 Channel Communications Controller
1D	4 or 8 Channel Communications Controller
1E	4 or 8 Channel Communications Controller
1F	ACM or Priority Interrupt Group 1

Table 2-3. Standard Device Orders

DEVICE ORDER (Value of f)	OPERATION	DESCRIPTION
0	Data	The data order causes a data byte to be transferred between the processor and addressed controller. The direction of transfer depends on the type of instruction (input or output).
1	Status/function	The status/function order causes a status byte to be transferred from the addressed controller to the processor, or a function byte to be transferred from the processor to the controller depending on the type of instruction (input or output).
2	Block input with interrupt	The block input with interrupt order notifies the addressed controller to proceed with a concurrent block input to memory, and to generate an interrupt at the end of the transfer unless the controller has been subsequently disarmed. This order is sent with an output instruction.
3	Arm interrupt	The arm interrupt order permits the addressed controller to make an external interrupt request on satisfying the interrupt condition. This order is sent with an output instruction.
4	Disconnect	The disconnect order causes the addressed controller to stop the block transfer operation in progress, and to generate an end-of-block interrupt unless the interrupt has been disarmed. This order is sent with an output instruction.
5	Disarm interrupt	The disarm interrupt order inhibits the addressed controller from making an external interrupt request under any condition and releases priority to lower devices. This order is sent with an output instruction.

Table 2-3. Standard Device Orders (continued)

DEVICE ORDER (Value of f)	OPERATION	DESCRIPTION
6	Block output with interrupt	The block output with interrupt order notifies the addressed controller to proceed with a concurrent block output from memory, and to generate an interrupt at the end of the transfer unless the controller has been subsequently disarmed. This order is sent with an output instruction.
7	Not assigned	This order, if assigned, may perform any required function as interpreted by the individual controller. The order can be sent with either an input or output instruction to cause a corresponding byte transfer.

2.3.3 STATUS BYTES

In response to a status order from the processor (specified by the device order), the addressed I/O controller places a status byte on input data lines ID00/ through ID07/. Four of the status bits are common to most device controllers; the other four are device dependent and differ from controller to controller. The status byte is transferred into the A Register, the B Register, or Memory by an input instruction, generally with device order 1. The significance of each bit in a typical status byte is described in Table 2-4.

2.3.4 FUNCTION BYTES

Some device controllers perform a greater number of operations than are specified by the three-bit device order. In these cases a function byte is output to the controller by an I/O command to further specify the desired operation. The byte is defined as a function byte by proper coding of the device order field (f).

2.4 BYTE I/O OPERATIONS AND TIMING

The following paragraphs describe the program-controlled and concurrent I/O operations of Micro 1600 series computers. Timing diagrams are included for each operation, and typical controller logic for performing the operations is shown. For a description of the I/O instructions that pertain to the byte I/O operations, refer to Micro 1600 Computer Reference Manual, publication No. 71-1-1600-001.

Table 2-4. Typical Status Byte Definition

BIT NO.	CONDITION	DESCRIPTION
0	Ready	Set to 1 when I/O controller is in a ready state and not performing a concurrent I/O operation
1	Input flag	Set to 1 when I/O controller has a byte ready for input to the processor
2	Output flag	Set to 1 when I/O controller is ready to receive a byte from the processor
3	Error	Set to 1 when an error has occurred during a transfer operation. The error may be the result of timing, parity, or a device malfunction. The bit is cleared when the status byte is transferred.
4-7	Undefined	Unique for each I/O controller

2.4.1 PROGRAM CONTROLLED I/O OPERATIONS

The four basic types of program controlled I/O operations are:

- Output of data to I/O controller
- Output of function (control) information to I/O controller
- Input of data to computer
- Input of controller status information to computer

Each type of I/O operation consists of two phases; the address/order phase and the transfer phase. The address/order phase is the portion of the operation during which the desired controller is addressed and the device order bits are sent to the controller. The second phase transfers one byte of information between the I/O controller and the computer. Timing diagrams for output and input operations are shown in Figures 2-3 and 2-4, respectively.

2.4.1.1 ADDRESS/ORDER PHASE. The address/order phase is identical for all types of I/O operations regardless of whether the information is an input command (IBA, IBB, IBM) or an output command (OBA, OBB, OBM), or if the type of information transferred is data, a controller function, or controller status.

All input and output instructions consist of two or four bytes. The first byte specifies the direction of the transfer and the source/destination within the CPU (A or B Register, or Memory). The second byte contains a five-bit device address specifying the desired controller, and a three-bit device order code containing control information for the controller. During the address/order phase, this second instruction byte is placed on the I/O bus and applied to the inputs of all device controllers. If the source/destination is memory, the instruction consists of four bytes, with the last two bytes specifying a core memory address.

Two hundred nanoseconds after the second byte is placed on the I/O bus, the I/O Control Register changes from state 0 to state 1. The controllers decode state 1 of I01X/ through I03X/ to produce control output signal COXX/. During COXX/, each controller examines lines OD00/ through OD04/ to determine which controller is being addressed. The controller whose address is on the lines connects itself for service and decodes and stores the device order code on lines OD05/ through OD07/. The I/O Control Register remains in the COXX/ state for 800 nanoseconds, after which it returns to state 0 (no function) and the address/order byte is removed from the output data lines. This marks the end of the address/order phase.

At this point, the addressed controller has been connected to the I/O bus and has received, decoded, and (in most cases) stored the function information contained in the device order code. The device order code may specify:

a. Transfer of data byte between the processor and controller.

b. Transfer of a status/function byte between the processor and controller.

c. One of several possible control functions to be performed by the controller (i.e., rewind magnetic tape, set up for concurrent transfer, enable interrupt, etc.).

If the device order code specifies transfer of data, the controller readies itself for a transfer to or from its data register. The direction of transfer is specified by the computer during the second phase.

A device order specifying a status/function transfer causes the controller to ready itself for transfer to or from its status/function register. In most standard controllers, the status/function register is used only to hold controller status information for transfer tothe computer. In these controllers, all necessary control functions can be specified in the device order alone. Some controllers, however, require a wider range of control than is provided by the three-bit device order. In these cases, the next byte output to the controller is sent to the status/function register where it specifies controller and/or device actions. When the device order specifies a device/controller function, the byte transferred during the second phase may be ignored by the controller.

2.4.1.2 TRANSFER PHASE. Transfer of the data, function, or status byte takes place during the second (transfer) phase of the I/O operation. During the transfer phase, the I/O Control Register changes to one of two states to specify the direction of transfer. For input operations, the controller places the byte being transferred onto the input data lines ID00/ through ID07/ and the CPU strobes the byte into the destination register or memory location.

During output operations, the byte is applied to output data lines OD00/ through OD07/ by the computer and is strobed into the appropriate controller register. At the end of the transfer phase, the I/O Control Register returns to state 0.

2.4.1.2.1 Data Output Operations. The transfer phase of a data output transfer begins when the computer places the data byte on the output data lines. Two hundred nanoseconds later, the I/O Control Register changes to state 2. The controller decodes the state of IO1X/ through IO3X/ to produce data output signal DOXX/, indicating the presence of data on the lines. The controller then strobes the data byte into its data register. When the I/O Control Register returns to state 0, the controller disconnects itself from further service. A timing diagram for data output operations is shown in Figure 2-3.

2.4.1.2.2 Function Output Operations. The timing diagram shown in Figure 2-3 for a data output operation is also valid for a function output operation. The function output operation is typically used to control a discrete action in an I/O device for which data transfer is not required. Rewinding tape is an example of such an action. The most efficient way to perform this operation is to issue a single instruction containing all the information necessary to alert the device and cause the tape to rewind.

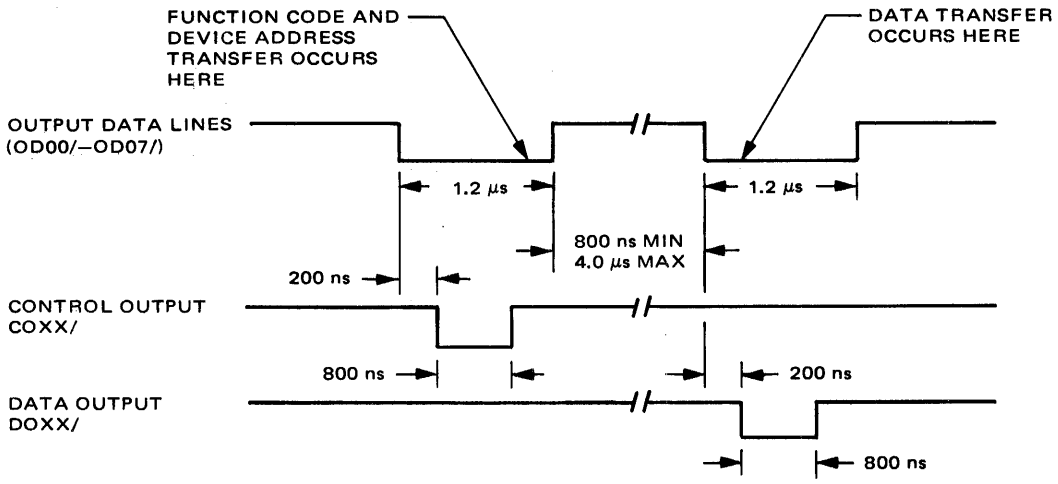


Figure 2-3. Data or Function Output Timing

In Micro 1600 computers, the output byte instructions (OBA, OBB, OBM) are also used to perform the function output operation. The only difference in the instructions is the value of the device order (bits 5 through 7) in the second byte of the instruction.

When an output byte instruction is used for function output, the device order code of the second byte designates the unique function in the I/O device to be controlled. The assignment of device order codes for function operations precludes the use of the same codes for data transfer operations.

The function output operation is executed exactly like the data output operation previously described. A data byte is transferred from either the A Register, the B Register, or Memory depending on the output instruction used. This data byte is usually ignored by the device controller, since the device order code of the control byte contains enough information to describe most function operations. However, should a controller require more function definition than is possible in the control byte, the data byte transferred during the function operation could be used to carry additional function information, as described in Paragraph 2.3.4.

2.4.1.2.3 Data Input Operations. The timing diagram for a typical data input operation is shown in Figure 2-4. Four hundred nanoseconds after the address/order byte is removed from the data output lines, the I/O Control Register changes from state 0 to state 6. The controller decodes the state of IO1X/ through IO3X/ to produce data input control signal DIXX/. DIXX/ causes the controller to place the data byte (from the controller data register) on input data lines ID00/ through ID07/. The data must be settled no later than 400 nanoseconds after DIXX/ goes low, and remain stable until DIXX/ again goes high.

Note

The controller may place the data byte on the lines as early as the beginning of signal COXX/.

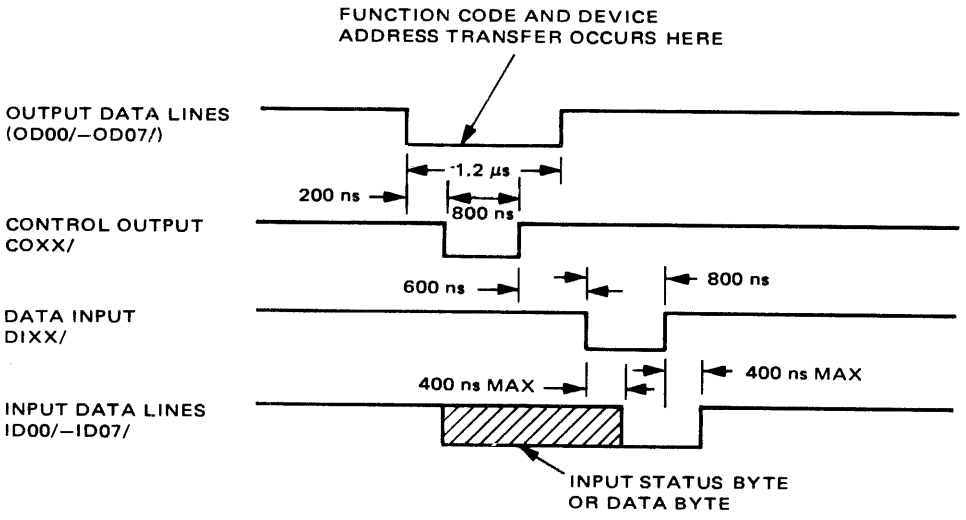


Figure 2-4. Data or Status Input Timing

Input data must be removed from the input data lines no later than 400 nanoseconds after line DIXX/ goes high. For normal operation on the external byte I/O bus with less than a 30-foot twisted pair cable, it is feasible to use the DIXX/ signal itself for gating or qualifying the application of input data to the input data lines.

The timing diagram shown in Figure 2-4 is also valid for a status input operation. A similar relationship exists between status and data input operations as exists between data and function output operations. In Micro 1600 computers, the input byte instructions (IBA, IBB, IBM) are used both for data and status input operations. The only difference is the source of the data byte in the controller. In status input operations, the source of the input byte is the status/function register. To differentiate between the two operations, a device order code of 000 is used in the control byte for data transfer and a code of 001 is normally used for status transfer.

2.4.2 TYPICAL CONTROLLER LOGIC

Typical logic used in Microdata designed, and user designed, I/O controllers to perform the various input and output operations is described in the following paragraphs.

2.4.2.1 ADDRESS/ORDER PHASE LOGIC. Controller logic active in this phase is shown in Figures 2-5 and 2-6. The eight I/O bus output lines are connected to TTL inverters to minimize the loading on these lines. Buffered lines OD00/ through OD04/ are applied to an address comparison circuit. As shown in Figures 2-5 and 2-6, this circuit is designed to match on device address 00. Buffered lines OD05/ through OD07/ carry device order information and are stored for subsequent use in the function storage register.

The I/O control lines (101X/, 102X/, and 103X/) are decoded in the controller to generate control lines COXX/, DOXX/, and DIXX/ (logic not shown). The occurrence of COXX/ enables the device address comparison circuit to generate DA00 if the controller has been addressed. Generating DA00 sets the connect-for-service flip-flop and stores the device order bits.

The connect-for-service flip-flop remains set through the subsequent transfer phase. If this flip-flop is set when DOXX/ occurs, OSTB is generated to strobe out a data or function byte. If it is set when a subsequent DIXX/ occurs, IDEN is generated to strobe in a data or status byte. Either DOXX or DIXX generates data strobe DSTR (if the controller is connected for service). Signal DSTR strobes the device order bits out of the function storage register into the function decoder where they are interpreted.

The connect for service flip-flop is set during COXX time or during ANXX time for concurrent transfers. Connect for service remains set until the trailing edge of following DOXX or DIXX.

Note

An alternative method of designing the address/order phase logic would be to decode the function bits first (during DAXX time) and store the decoded results in individual flip-flops. This method could be advantageous for controllers using less than the full complement of functions. Using this method eliminates the need for a connect for service flip-flop since the individual flip-flops provide this function.

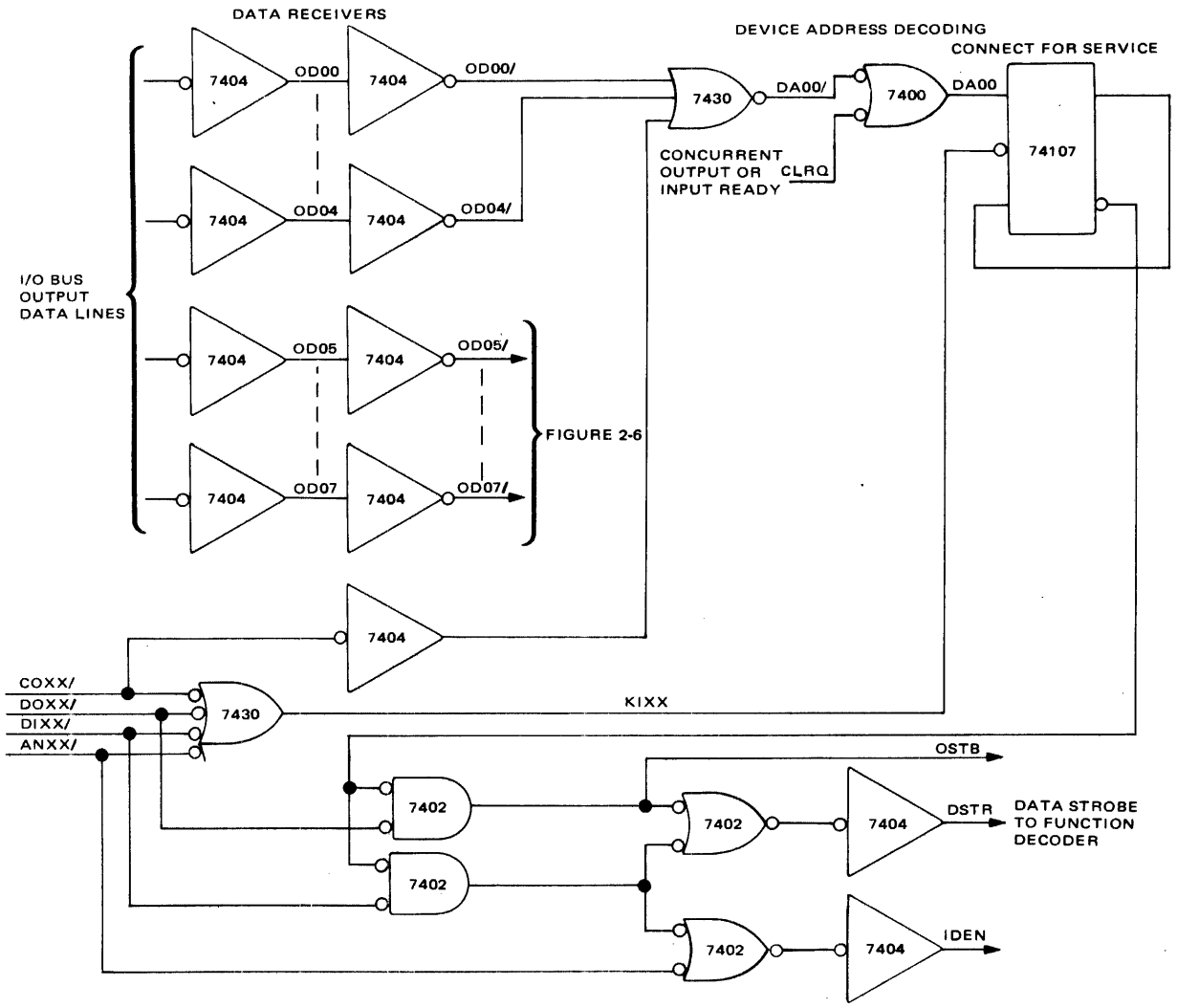


Figure 2-5. Data Output Logic

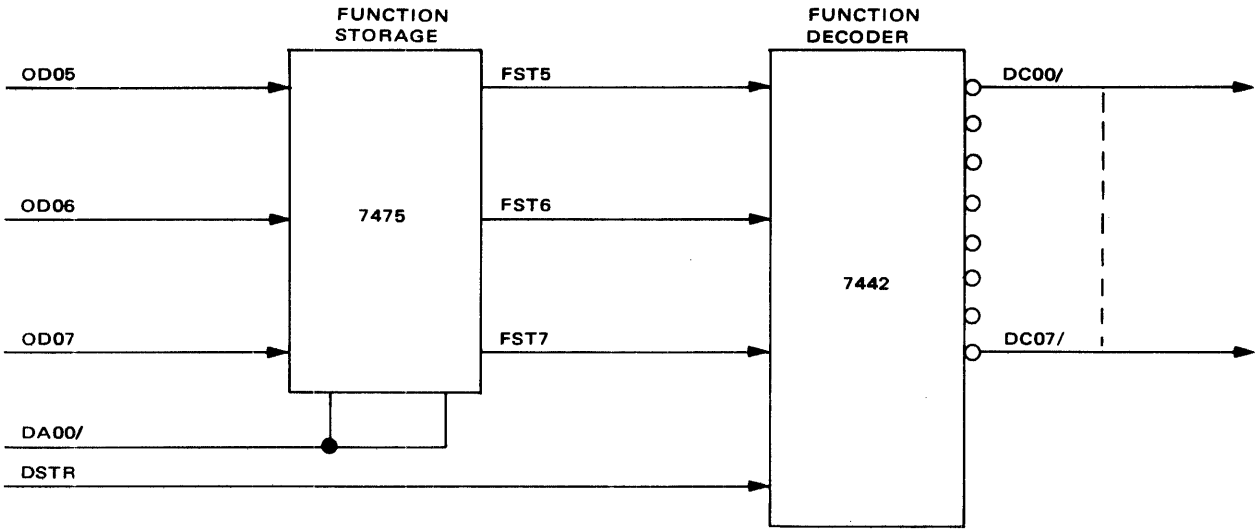


Figure 2-6. Function Output Storage and Decoding Logic

2.4.2.2 INPUT TRANSFER PHASE LOGIC. Eight power drivers are used to drive the eight I/O bus data lines (ID00/ through ID07/). The power drivers are type 7438 TTL (or equivalent) power gates with open collectors. A termination resistor network is contained in the processor. These power drivers must be held off when the controller is not addressed, and should only be turned on when the controller is in the connect for service state. This function is accomplished by signal IDEN (see Figure 2-7) which is the AND of DIXX/ and the connect-for-service flip-flop (or ANXX if an interrupt or concurrent request is being answered).

The three device order bits stored during the address/order phase determine the function to be performed by the controller during the transfer phase. If a DIXX/ occurs, these bits specify a particular type of byte to be input (e.g., data byte or status byte). In Figure 2-7 the DC00 output from the function decoder means data input and DC01 means status input.

2.4.2.3 OUTPUT TRANSFER PHASE LOGIC. The data or function byte transferred during DOXX/ time is received by the same TTL buffers that receive the second byte of the instruction during COXX/ time (see Figure 2-5). The destination of the byte may be either a data register or a control register in the controller. Signal OSTB, which is the AND of DOXX/ and the connect-for-service flip-flop, is used to clock these registers. The three device order bits stored during the address/order phase determine the destination. Function decoder outputs (see Figure 2-6) select the register which receives the byte.

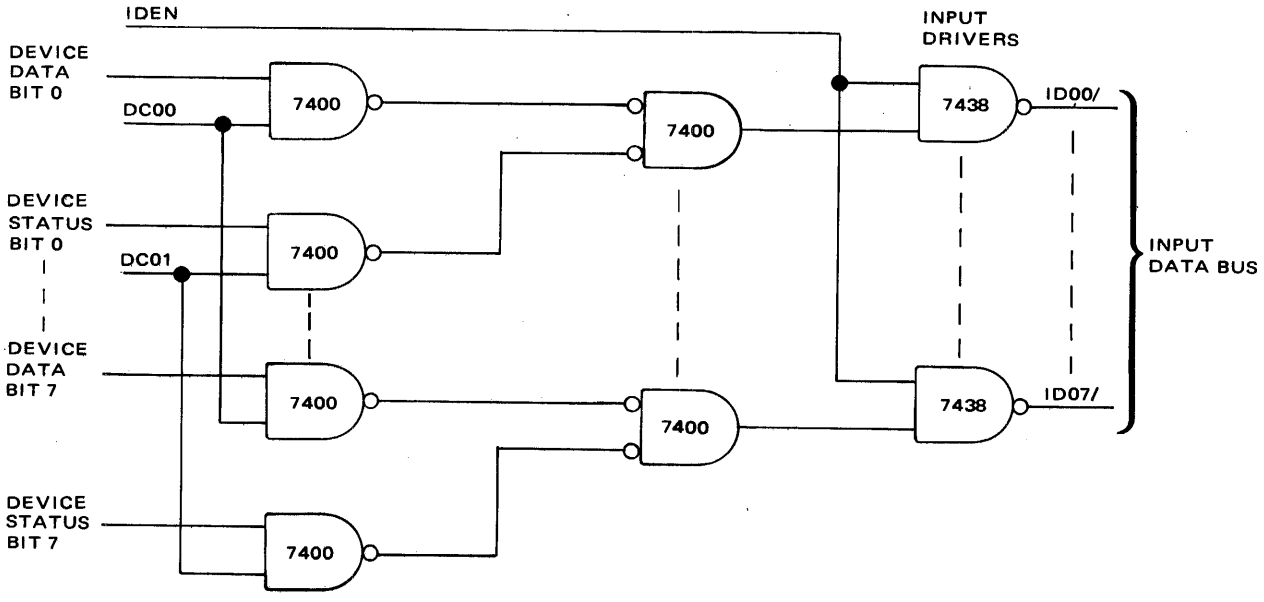


Figure 2-7. Data Input Logic

2.4.3 TYPICAL PROGRAM-CONTROLLED TRANSFER SEQUENCE

Figure 2-8 is a flow chart showing the sequence and timing for a typical program-controlled transfer of a block of data that could be handled using the typical device controller logic previously described. The sequence consists of:

a. Sampling and determining the status of the device controller.

b. Transferring data from memory to the controller when the controller is ready to accept it.

c. Updating the index register (used as a data pointer) and checking for the end of the block.

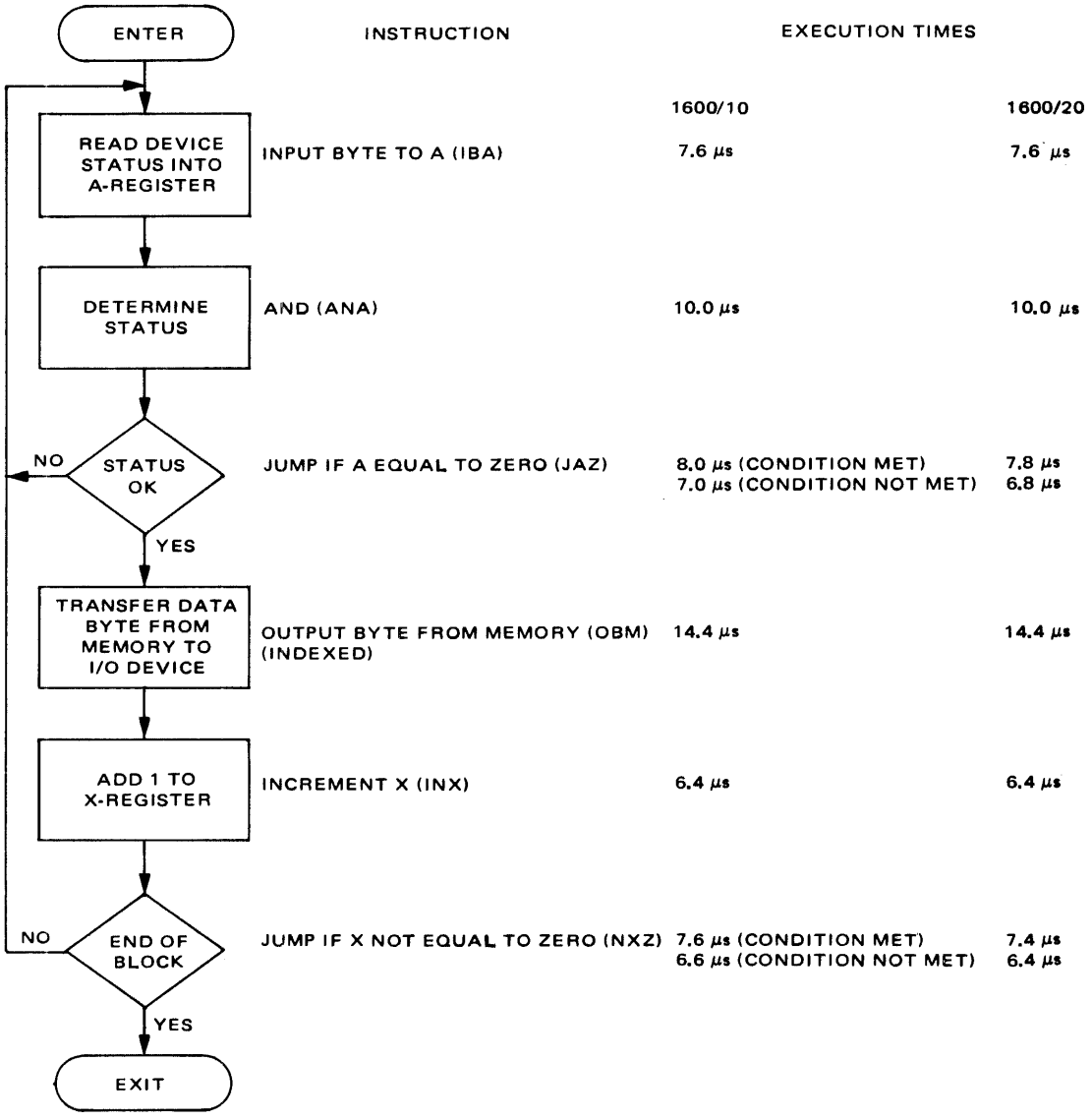


Figure 2-8. Typical Program-Controlled Transfer Sequence

The typical transfer rate using the sequence of instructions shown in Figure 2-8 is approximately 19 kilobytes per second. This rate is determined from the sum of execution times of all instructions (approximately 53 nanoseconds). The rate is valid only if the device is ready when the first status check is made.

The first operation performed in the sequence is status input to the A Register. An Input Byte to A (IBA) instruction with a suitable device address is used for this purpose. Next, the status bits are checked using AND and JAZ instructions to determine if the device is ready for the data transfer. This check is important in any program-controlled data transfer. Device data transfers typically cannot be performed without reference to some condition of the device (a status bit) that indicates its readiness to transfer or accept data.

Status checking continues until the controller indicates it is ready to receive data. A data byte is then transferred to the controller by an Output Byte from Memory (OBM) instruction. Following the transfer, the X Register (acting as a data pointer) is updated and tested for zero by INX and NXZ instructions. In this example, the X Register initially holds a negative count specifying one more than the number of bytes to be transferred. As each byte is transferred, one count is added to the register. When the last byte has been transferred, the X Register contains zero and the subroutine exits to the main program.

The data rate of this sequence can be increased if the device status checking steps are eliminated (assuming the device is always ready to accept data). The maximum data transfer rate for a sequence of this kind is approximately 37 kilobytes per second, based on instruction execution times totalling 27.2 microseconds. The instruction sequence would consist of OBM, INX and jump (NXZ) instructions.

2.5 CONCURRENT I/O OPERATION

Concurrent I/O operation is the name given to the block transfer technique used in the standard Micro 1600 computer. The software program sets up starting and stopping addresses for the block transfer in dedicated memory locations and executes an output instruction to initiate the transfer; thereafter, firmware controls the data transfer operation automatically on demand from the device controller. In terms of controller design, the additional logic required for concurrent I/O operations can be thought of as an overlay to the program-controlled logic discussed earlier. All that is required is additional logic to recognize a data ready condition and to assert a concurrent I/O request at that time instead of waiting for an input or output instruction, as is the case with program-controlled operations. A device performing a concurrent I/O operation initiates its own data transfers when it is ready.

2.5.1 CONCURRENT I/O TIMING

The timing diagram for a typical concurrent I/O operation is shown in Figure 2-9. When the device controller is ready to transfer data, it causes the concurrent I/O request line (ECIO/) to go low. After recognizing the request, firmware in the processor causes it to respond by setting the I/O Control Register to state 4, which produces the I/O acknowledge

signal (CACK/) in the controller. While line CACK/ is low, the device controller applies an address byte containing its own address and a bit indicating the direction of transfer (input or output) to input data lines ID01/ through ID07/.

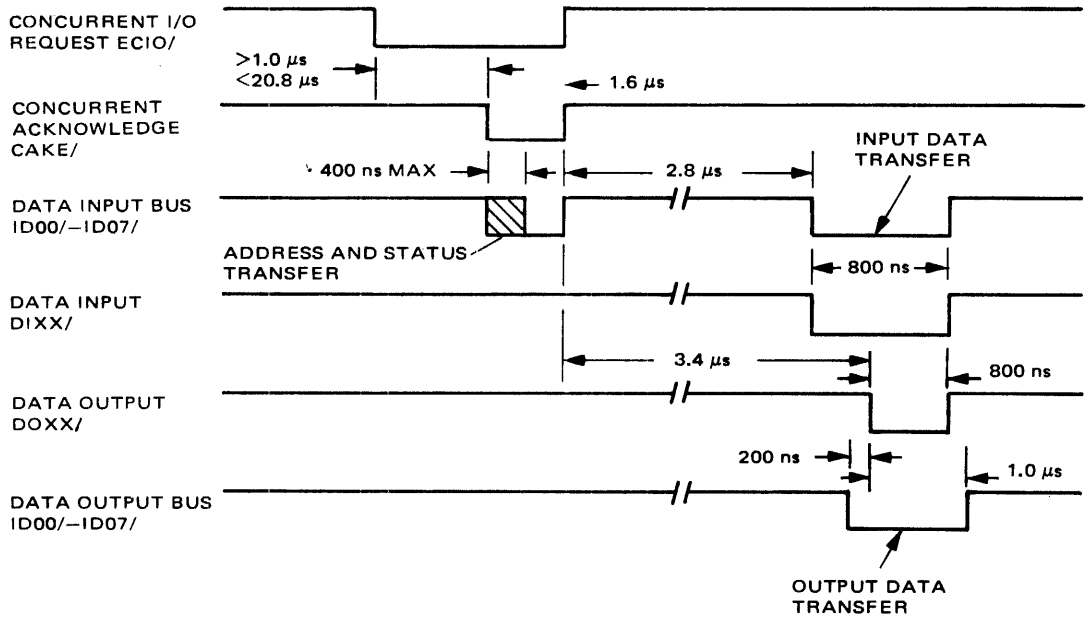
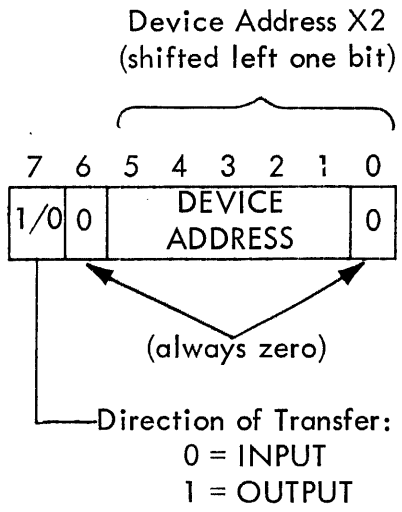


Figure 2-9. Concurrent I/O Timing

Concurrent I/O operations utilize four dedicated memory locations to hold the 16-bit starting and final addresses of the block to be transferred. In Micro 1600 computers, the first of these four bytes is at the location specified by the controller device address times four. When the processor responds to a concurrent I/O request with acknowledge signal CACK/, the controller places a byte on data input lines ID00/ through ID07/. This byte contains the device address times two (shifted left one bit) in the lower six bits with bit 7 set or reset to specify the direction of the transfer. This byte is shown below:



When the computer receives this byte, it obtains the actual dedicated memory address (four times device address) by shifting the 2X address (supplied by the controller) left one more bit position. The direction bit (7) is shifted out and used by the computer to specify an input or output operation. Examples of device addresses, addresses supplied by the controller, and dedicated memory addresses are as follows:

Actual Device Address (Hex)	Address Supplied By Controller (Hex)	Dedicated Memory Address (Hex)
00	00	00 through 03
01	02	04 through 07
02	04	08 through OB
.	.	.
.	.	.
.	.	.
.	.	.
1F	3F	7C through 7F

The firmware uses the address byte to initiate a normal input or output operation to or from memory. If an input operation is specified, the firmware drops DIXX/ and the controller responds by placing a data byte on ID00/ through ID07/. If an output operation is specified, the firmware drops DOXX/ and the controller strobes in the data byte from OD00/ through OD07/.

The addresses of the block of memory locations to or from which the transfer takes place is specified by software. Prior to starting the block transfer, the program loads the four dedicated memory locations with a 16-bit starting (current) address and a 16-bit ending address. After each byte is transferred, the starting (current) address is incremented by the firmware. Thus, each byte is transferred to or from the next sequential location. After each transfer (but before the current address is incremented) firmware compares the current address with the final address. If the two are equal, the firmware tells the controller that the transfer is complete by essentially executing an output instruction with a device order of 4. Upon receipt of the second byte of this instruction (strobed by DOXX/), the controller issues an external interrupt to indicate to the CPU that the operation is complete.

2.5.2 TYPICAL CONCURRENT I/O LOGIC

Since there is no transfer of control bytes to a controller during the data transfer phase of concurrent I/O operations, logic must be included in the controller to force it into the data transfer connected state whenever it receives a concurrent I/O acknowledge (CACK/). Once the forced connection is made, the second phase of the data transfer operation occurs normally.

Figure 2-10 shows the additional logic required for concurrent I/O operations. The concurrent I/O request line (ECIO/) and the concurrent acknowledge line (CACK/) were described in the preceding paragraphs.

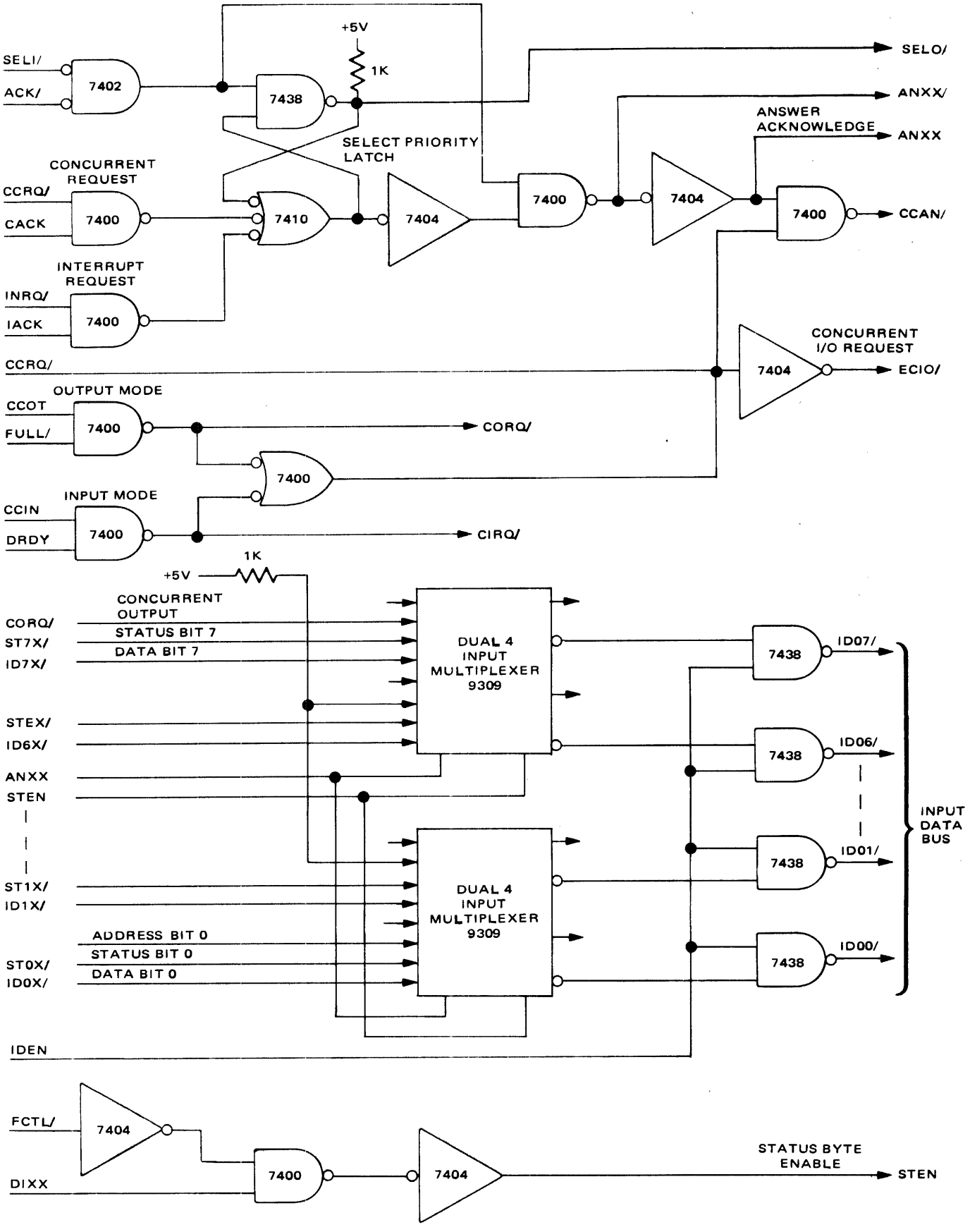


Figure 2-10. Concurrent I/O Logic

The addition of select in (SELI/) and select out (SELO/) lines are shown in the logic. Line SELI/ permits any device requesting either an external interrupt (EINT/) or a concurrent I/O request (ECIO/) to respond to an acknowledgment signal (CACK/ or IACK/). The select logic is active only during an acknowledgment signal. The device having the highest select priority responds to the signal.

Acknowledge signal CACK/ (issued by the CPU) is sent to all controllers on the I/O bux. Only the controller which issued the concurrent I/O request can answer the acknowledge signal, if two or more controllers issued a request simultaneously, only the requesting controller with the highest priority can answer. Three conditions must be met before a controller can answer CACK/ by connecting itself to the I/O bus for the data transfer:

a. Concurrent I/O acknowledge (CACK/) must be received.

b. The controller must have requested a concurrent transfer (CCRQ/ must be true).

c. A higher priority controller must not have requested a transfer (SELI/ must be present at the controller).

The first controller in the chain always has select priority (SELI/). When CACK/ is received, the first controller combines SELI/, CACK/, and CCRQ/. The combination of these signals is applied to select priority latch SELO/. If CCRQ/ is false, indicating that the controller did not request the transfer, the latch is set and SELI/ is passed to the next controller. When SELI/ is received by the next controller, the combination of CACK/, CCRQ/, and SELI/ is performed. If the second controller did not initiate the request, SELI/ is passed to the third controller, and so on until the requesting controller is reached. The requesting controller will not set latch SELO/, preventing SELI/ from being passed to controllers lower on the priority chain. The combination of CACK/, CCRQ/, and SELI/ is also applied to logic in the controller which connects the controller to the I/O bus for the data transfer.

The driving logic for the input data lines has been expanded in the example to include address transfer capability and a flag (ID07/) for indicating direction of transfer.

Disconnect signal DISC from the CPU is received by the controller. DISC advances the interrupt sequencer to the wait state. When the controller receives priority, an interrupt request is generated indicating to the CPU that the block transfer is complete (concurrent end of block).

2.6 EXTERNAL INTERRUPT OPERATION

Interface lines PROT/, PRIN/, SELO/, SELI/, EINT/, IACK/, and ID00/ through ID07/ are used by device controllers or the optional Priority Interrupt board on the byte I/O bus for external interrupt operations. Lines PROT/ and PRIN/ (Paragraph 2.2.4.4) make up the hard-wired priority chain that determines the relative priority of each controller

and Priority Interrupt board on the byte I/O bus. These lines determine priority for interrupt requests. Line EINT/ (Paragraph 2.2.3) carries the interrupt request from the controller to the processor. I/O control register state 5 is decoded in the controller as interrupt acknowledge IACK/. Input data lines ID00/ through ID07/ carry an interrupt address byte from the interrupting controller to the processor in response to the interrupt acknowledge signal on line IACK/. The interrupt address byte is used by the processor to locate the entry address of the interrupt servicing subroutine.

2.6.1 PRIORITY DETERMINATION

Interface units on the byte I/O bus are assigned priority for control of external interrupt and concurrent I/O request operations. The priority is achieved by the manner in which lines PRIN/, PROT/, SELI/, and SELO/ are used to link the interface units together. A typical example of priority wiring is shown in Figures 2-11 and 2-12. In these examples three device controllers in the mainframe chassis and four controllers in the expansion chassis are connected in the priority chain. The I/O Line Driver and Receiver board serves to pass signals PROT/ and SELO/ to the expansion chassis. The figures show that the priority of an interface unit is the same as the physical location of that interface on the byte I/O bus. With special priority wiring, however, the relative priorities can be independent of backplane positioning.

A device may make a concurrent I/O request at any time. However, to make an external interrupt request, the device must have priority in (PRIN/). Signal IS02 (Figure 2-11) on each interface unit is used to inhibit propagation of signal PRIN/ if the interrupt servicing routine is not complete. This establishes a true-level priority among all interface units for generating an external interrupt. A controller never passes a low signal on line PROT/ if it is making a request or until the interrupt servicing routine is complete.

2.6.2 INTERRUPT REQUESTS

External interrupt requests from interface units are carried on line EINT/ to the processor. The internal microprogram recognizes the presence of an external interrupt request and responds as dictated by interrupt handling firmware. External interrupt line EINT/ can be used both by device controllers and by the optional Priority Interrupt interface board. The Priority Interrupt option provides the proper interface to the I/O bus, contains priority logic for each interrupt level, and permits processor control over the handling of interrupts. This standard option provides, on one circuit board, convenient hardware for 8 levels of system interrupts. Because the basic interrupt facility makes use of the byte I/O bus, all device controllers have access to the interrupt request line and can react to the firmware interrupt handling sequences in the processor (provided they operate according to the design guidelines giveniin Paragraph 2.6.3).

Note

Requesting an interrupt removes priority for interrupt operations from all controllers lower on the priority chain.

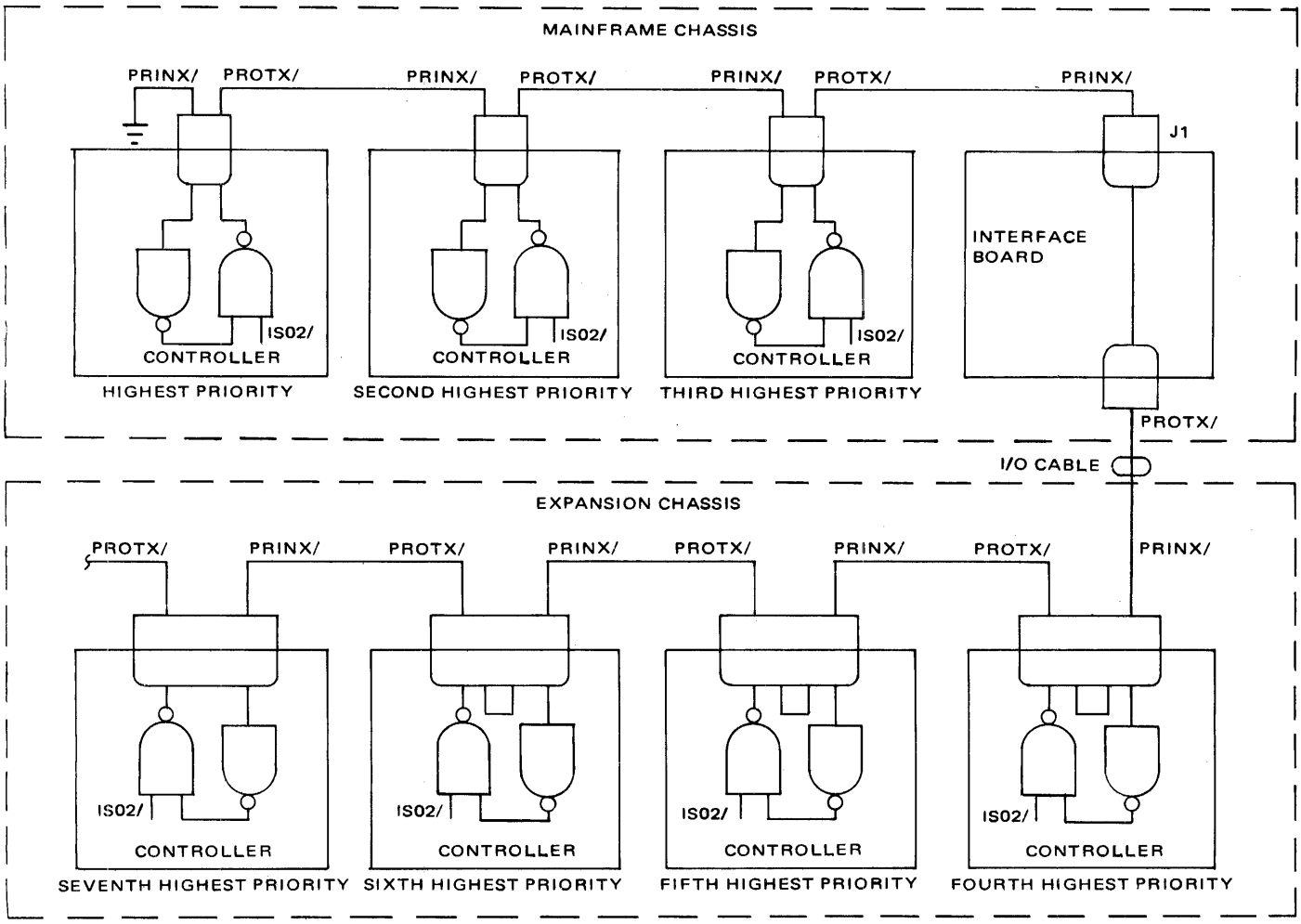


Figure 2-11. Typical Priority Scheme (Standard Backplane)

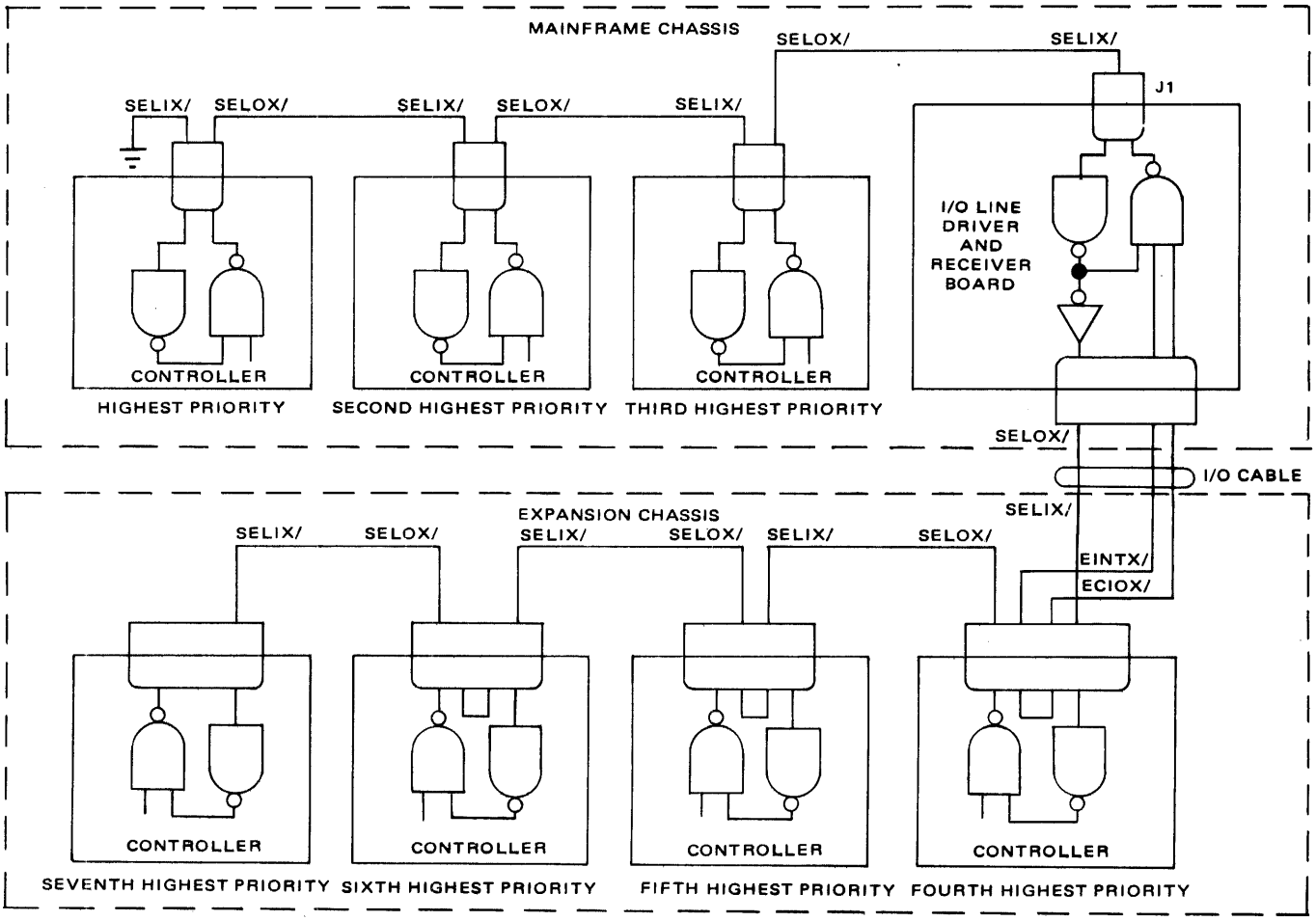


Figure 2-12. Typical Selection Acknowledgment Scheme (Standard Backplane)

2.6.3 INTERRUPT SEQUENCE AND TIMING

Figure 2-13 shows the timing for a typical external interrupt sequence. The firmware, processor, and I/O device operation during the sequence is as follows:

a. The I/O device controller lowers line EINT/ to signal a request for microprogram attention. The controller must receive priority signal PRIN/ from the higher priority controllers. The requesting controller does not pass the priority signal to lower controllers.

b. At the end of the macro instruction currently being executed (if not a privileged instruction like I/O or jump), the microprogram senses the interrupt request and jumps to a firmware subroutine to handle the request.

c. The microprogram causes line IACK/ to go true to acknowledge the request. All controllers in the priority chain decode IACK/ and each requesting controller passes SELO/ down the chain to the lower priority controllers (SELO/ becomes SELI/ at the input to each controller).

d. The controller that issued the interrupt request does not pass SELO/ to the next controller. This prevents any lower priority controller that may have simultaneously requested an interrupt from responding to signal IACK/.

e. In response to the acknowledgment and receipt of SELI/, the requesting controller places a six-bit interrupt address on input data lines ID01/ through ID06/. The interrupt address specifies the location (in core memory page 1) of the two-byte entry address for the interrupt servicing subroutine.

f. The processor accepts the six-bit interrupt address and causes line IACK/ to go high.

g. The processor fetches the two-byte interrupt subroutine entry address from the first 256-word page of memory using the interrupt address supplied by the controller as the lower six bits, and 01 as the upper two bits.

h. Using the two-byte entry address, the microprogram executes a pseudo return jump or call instruction to the interrupt servicing subroutine.

i. The interrupt servicing subroutine then proceeds to service the interrupt according to the macroprogram.

i. At the end of the servicing routine, priority is released by any of the three actions listed below. Any of these will cause the requesting controller to pass signal PROT/ to lower priority controllers.

1. rearming (if another request is expected)
2. a concurrent I/O request
3. disarming (if no further requests are desired)

Note

The logic for generating acknowledge select (SELO/) on external interrupts is the same as that used on concurrent I/O requests. The controlling signals, as shown in Figure 2-10 are:

INRQ - this device is requesting an interrupt
IACK - external interrupt is acknowledged
SELI - this device has select priority

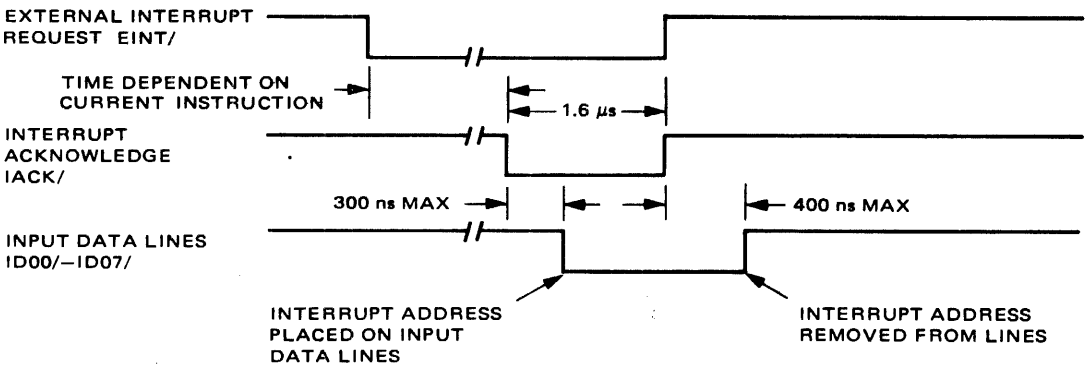


Figure 2-13. External Interrupt Timing

The interrupt sequencer in the controller contains two JK flip-flops (and associated circuits) which generate the interrupt request (EINT/) and control the priority line (PROT/) to the next controller. The four states of the flip-flops determine the priority interrupt status of the controller. These four states are illustrated in Figure 2-14 and described in Table 2-5.

When the controller is initialized, the sequencer is set to state zero (disarmed). When disarmed, the controller cannot generate an interrupt request and always passes PROT/ to the next controller.

In order to allow an interrupt, the program must execute an instruction to arm the controller interrupt, setting the sequencer to state one (armed). In this state, the controller can generate EINT/ providing that it has priority from the preceding controller on the priority chain.

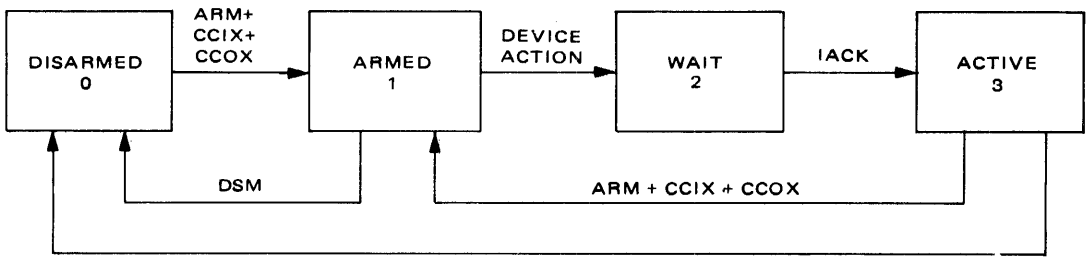


Figure 2-14. Interrupt Sequencer States

Table 2-5. Interrupt Sequencer States

State	Flip-Flop States		Function
	IS02	IS01	
0	0	0	Disarmed state. Disregards any received interrupt and does not move to requesting state
1	0	1	Armed state. Allows system to move to requesting state when peripheral conditions are met
2	1	1	Wait state. Generates an external interrupt to the processor when priority in is received
3	1	0	Active state. Inhibits propagation of priority to lower level priority controllers

When the controller is ready to interrupt the CPU, the sequencer advances to state two (wait), the priority line (PROT/) is removed from the next controller, and interrupt request EINT/ is generated. The firmware responds to the interrupt with the acknowledgment (IACK/). The first interrupting controller in the string that has SELI places its address on the input data lines. Its sequencer then advances to state three (active) and removes EINT/.

While the sequencer is in the active state, PROT/ is not passed to the next controller. This prevents a lower priority controller from generating an interrupt while the interrupt handling subroutine is in process. One of the functions of the interrupt subroutine is to execute an instruction to rearm the controller if another interrupt is expected, or to disarm the controller if no further interrupts are desired. The rearming or disarming normally takes place near the end of the subroutine and restores priority (PROT/) to the lower priority controllers.

Concurrent I/O operations are normally terminated with an end-of-operating interrupt to inform the CPU that the block of data has been transferred. The concurrent I/O request automatically arms the controller interrupt.

SECTION 3

DIRECT MEMORY ACCESS PORT

3.1 INTRODUCTION

The Direct Memory Access (DMA) Port is a channel to/from the Micro 1600 core memory through which data may be transferred at very high speeds without involving the Micro 1600 CPU. DMA transfers are controlled entirely by an external DMA interface and occur at the rate of the external device up to full memory speed. At maximum memory speed, transfers take place at 1-million bytes per second using back-to-back, 1.0 microsecond memory cycles.

The DMA Port consists of the set of lines used for DMA control and data transfer. These lines are available at mainframe backplane card slots J4 through J13 in the basic backplane, and J4 through J17 in the extended backplane.

A simplified functional block diagram of the Micro 1600 DMA system is provided in Figure 3-1. The external peripheral device is controlled in the conventional manner by a device controller connected to the byte I/O bus. Function output and status input are accomplished via the byte I/O bus. This includes commands to initiate the DMA transfer. As indicated by the dotted line, these latter functions may be performed within the DMA interface.

The DMA interface logic/buffers and the device controller may be on one printed circuit board or on separate boards. If separate boards are used, they must be interconnected via cables attached at the rear of the boards. A separate DMA interface is available from Microdata.

3.2 FUNCTIONAL DESCRIPTION

In systems which utilize the DMA capability of the Micro 1600, core memory is shared by the DMA interface and the central processor. When either element requires a read or clear/write memory cycle, it issues a request for memory service. To ensure minimum latency time in answering DMA requests and to prevent the CPU from interrupting DMA operations, the DMA interface always has priority over the CPU for memory service. When the DMA interface requests memory service, it essentially freezes any CPU activities from occuring if during this time the CPU requests memory service.

The memory responds to a DMA interface request as soon as the CPU memory cycle in process is completed. The DMA interface then selects the read or clear/write mode and initiates gating of a 16-bit address onto the memory address bus from an address counter in the DMA interface or the device controller. The data byte is then transferred.

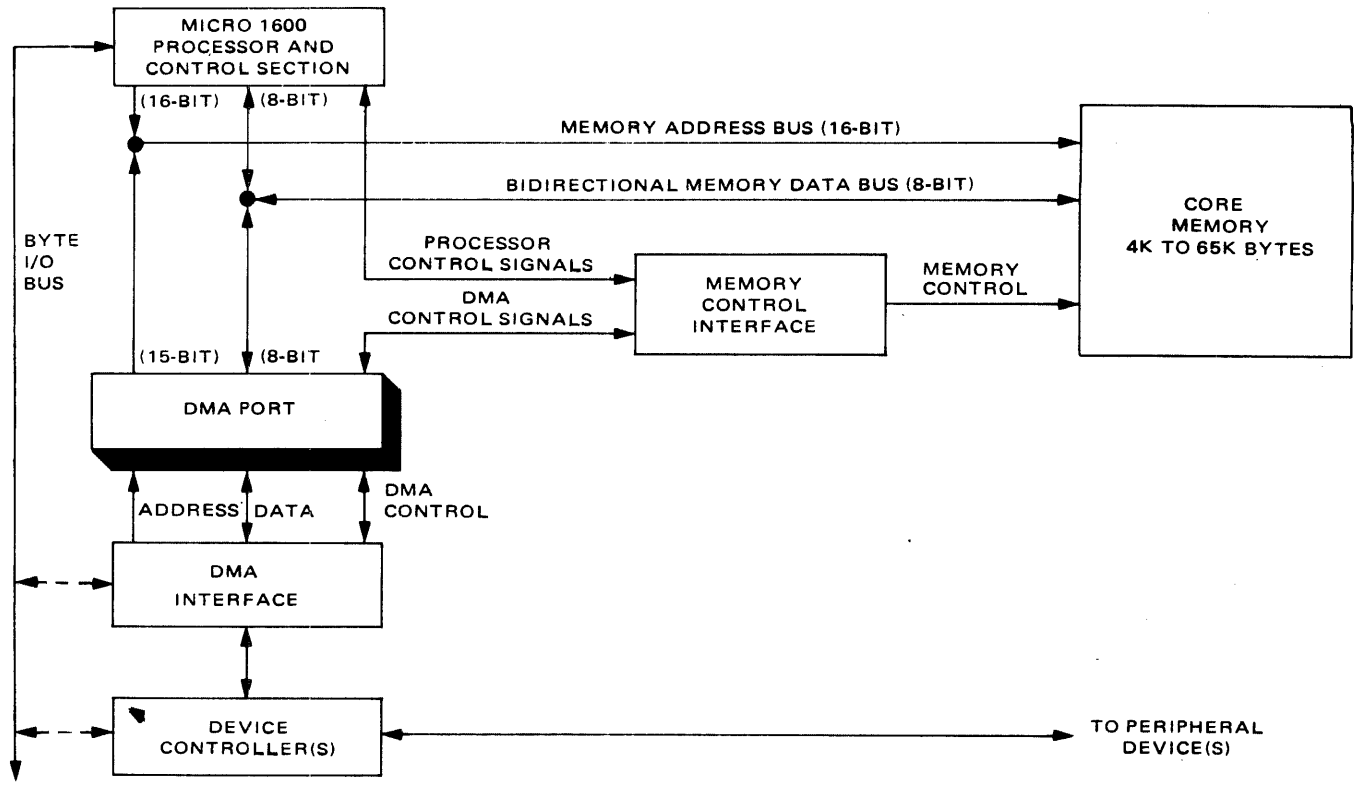


Figure 3-1. DMA/Processor Core Memory Interface (Simplified Block Diagram)

This process is repeated every time the device controller is ready to do a data transfer. Sequential addresses are gated onto the lines until an entire, predetermined block has been transferred. The determination of the end of the DMA block transfer is accomplished in the interface or device controller by a pair of 16-bit registers, the starting (current) address register, and the final address register, which are loaded with the desired limits of the block prior to the transfer. Following the transfer of each byte, the current address register is incremented to the next sequential address and compared to the final address register.

An end-of-block interrupt should be generated in the DMA interface or device controller. This may be either an external interrupt via the I/O bus or an internal DMA interrupt which sets bit 1 of the CPUs internal status register.

The following paragraphs describe the various elements in the DMA system and their functions in DMA operations. Figure 3-2 shows a detailed functional block diagram of the CPU, memory, and DMA interface elements in the DMA system

3.2.1 DMA INTERFACE

The DMA interface provides the interface and control required for utilization of the Micro 1600 DMA capability. The user may employ a standard Microdata DMA interface, or he may use a DMA interface of his own design to meet the special requirements of his individual system.

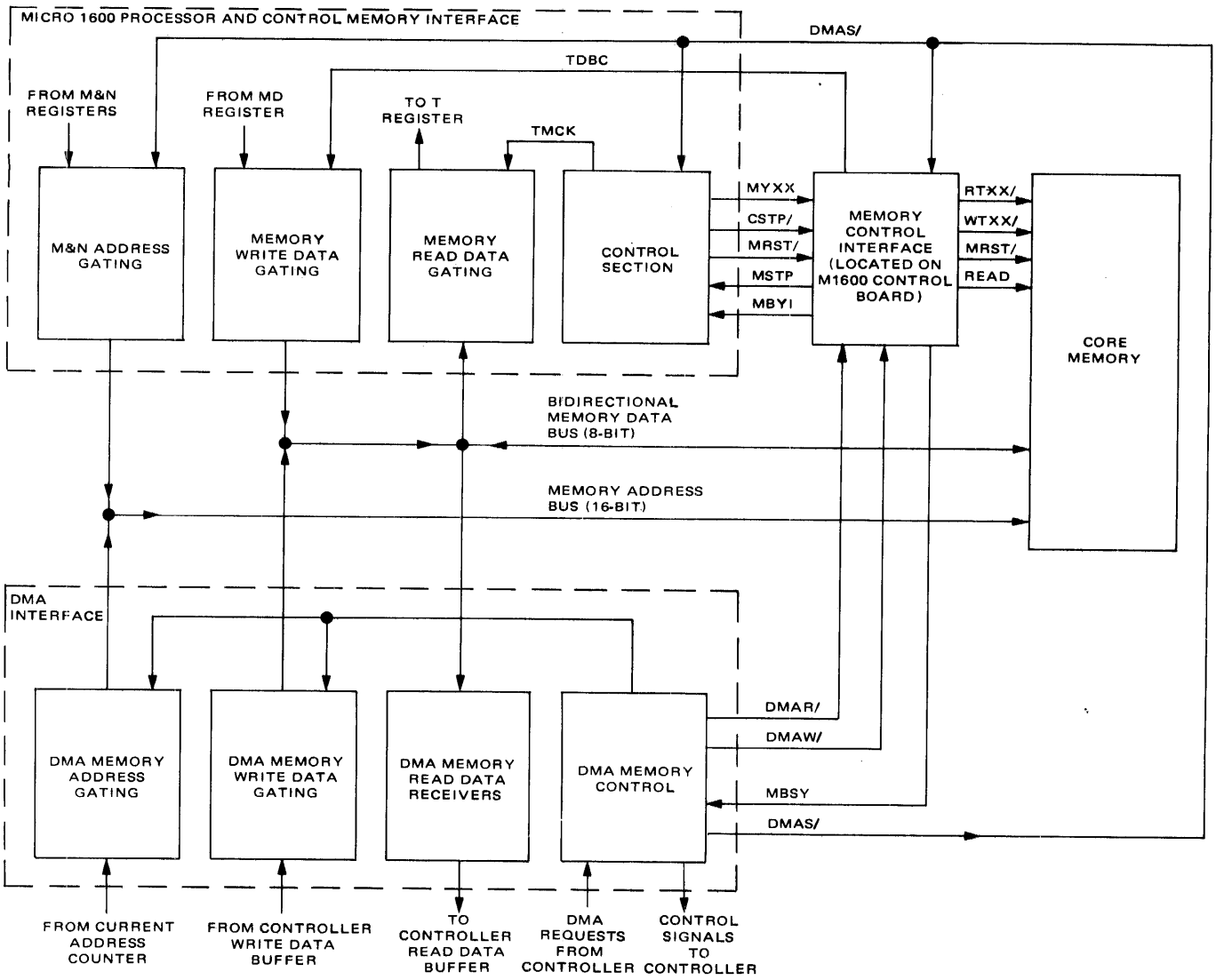


Figure 3-2. DMA Port Interface (Detailed Block Diagram)

The elements of the DMA interface are:

- DMA Memory Control Logic
- DMA Memory Read Data Receivers
- DMA Memory Write Gating Logic
- DMA Memory Address Gating Logic

3.2.1.1 DMA MEMORY CONTROL LOGIC. When the DMA interface has been set up via the I/O bus or the controller for DMA operation, this element monitors controller requests for DMA transfers. After receiving a controller request, the DMA Memory Control Logic initiates a DMA memory cycle sequence as soon as the CPU memory cycle is completed (MBSY = low level). The DMA memory cycle sequence consists of a DMA request (DMAR/), DMA selection (DMAS/), and read or write mode selection (DMAW or DMAW/, respectively). These signals originate in the DMA Memory Control section and are sent to the Memory Control Interface in the Micro 1600.

Signal DMAS/ performs the following functions in both the DMA interface and the Memory Control Interface:

- Gates memory address from current address counter onto memory address bus.
- Gates address through Memory Control Interface to address matrix in core memory.
- For memory write operations, gates write data onto bidirectional memory data bus.

Signal DMAW is set to the low state (DMAW/) by the DMA Memory Control if a clear/write cycle is being requested. It is set high (DMAW) if a read cycle is requested.

3.2.1.2 DMA MEMORY READ DATA RECEIVERS. Data read from memory during a DMA transfer is buffered from the bidirectional data bus by eight receivers in the DMA interface. The outputs of the receivers are applied to the controller.

3.2.1.3 DMA MEMORY WRITE GATING LOGIC. This element gates the eight-bit data byte from the controller onto the bidirectional data bus and is written into memory during memory write operations. The data byte is gated onto the bus by DMA Selection signal DMAS.

3.2.1.4 DMA MEMORY ADDRESS GATING LOGIC. This element gates the 16-bit address stored in the current address register (contained in either the DMA interface or the controller) onto the memory address bus. Address gating occurs at DMAS time.

3.2.2 MICRO 1600 PROCESSOR AND CONTROL MEMORY INTERFACE

As shown in Figure 3-2, the portion of the Processor and Control Memory interface which is involved in DMA operations consists of the following elements.

- Control Section
- Memory Read Data Gating Logic
- Memory Write Data Gating Logic
- M and N Register Gating Logic

3.2.2.1 CONTROL SECTION. The Control Section shown in Figure 3-2 is the main control element of the Micro 1600 CPU. However, only that part of the Control Section which pertains to DMA operation is discussed.

One function of the Control Section is decoding memory type microcommand MYXX. Microcommand MYXX requests memory service for the CPU, and eventually results in a memory busy (MBSY) condition while the memory services the CPU. While MBSY is high, due to memory use by the CPU, the DMA is prevented from issuing a DMA request for memory service. While memory is being used by the DMA channel, the Control Section is inhibited from executing MYXX.

The Control Section also generates Clock Stop signal CSTP/. In a DMA operation, CSTP/ is used to stop the main CPU clocks while a DMA memory cycle is occurring. This essentially freezes execution of microcommands and prevents CPU memory requests while memory is servicing the DMA channel. Signal CSTP/ is generated by either of two sets of conditions:

a. 1. MBYI = high level
   2. DMAS/ = high level
   3. M or N Register selected for data transfer

b. 1. MSTP = high level
   2. MYXX = high level

The third function of the Control Section which relates to DMA operations, is generation of Transfer Memory Clock signal TMCK. Signal TMCK clocks the memory read data through the Memory Read Data Gating Logic to the CPU T Register. When the DMA interface is using memory, TMCK is inhibited. Inhibiting TMCK prevents the data being transferred from memory to the controller from entering the CPU T Register.

3.2.2.2 MEMORY READ DATA GATING LOGIC. During CPU memory accesses (read mode), this element gates data read from memory into the CPU T Register. As explained in the preceding paragraph, the Transfer Memory Clock (TMCK) signal from the Control Section initiates the gating operation. To prevent read data gating to the CPU during DMA operations, DMAS/ inhibits generation of TMCK.

3.2.2.3 MEMORY WRITE DATA GATING LOGIC. During CPU memory write operations, this element gates the write data from the MD Register onto the Memory Data bus. Gating signal TDBC, which originates in the Memory Control Interface, is inhibited during DMA operations by DMAS/.

3.2.2.4 M AND N REGISTER ADDRESS GATING LOGIC. During CPU memory accesses (read or write), this element gates the 16-bit memory address onto the Memory Address bus from the M and N Registers. During DMA operations, when the memory address is supplied by the DMA interface, this function is inhibited by DMAS/.

3.2.3 MEMORY CONTROL INTERFACE

The Memory Control Interface is the principal memory controlling element of the system. Its primary functions are:

- Generation of memory status and control signals.
- Monitoring of requests for memory service from the DMA and CPU.
- When memory is not busy, determination of which requesting device (CPU or DMA) is to receive access. (Determination is based on DMA having highest priority and the CPU having second priority.)
- Initiation of memory read or write operations and timing out of the memory cycles.

The Memory Control Interface generates the memory busy signals to the CPU (MBYI) and to the DMA interface (MBSY). When memory is available, both of these signals go low and either or both the DMA and CPU can request memory service. If both elements request service simultaneously, the DMA has priority and will receive memory service. The DMA memory request signal is DMAR/; CPU memory request signal is MYXX.

When the DMA or CPU request sequence occurs, the Memory Control Interface generates the appropriate memory control signals (listed below) and times out the memory cycles.

- RTXX - Start read portion of cycle
- WTXX - Start write portion of cycle
- READ - Low level = clear/write or half-cycle write;
  High level = read/restore or half-cycle read

3.3 DMA PORT/MEMORY CONTROL INTERFACE TIMING

For the purpose of explaining the timing of the essential DMA Port/Memory Control Interface signals, Figure 3-3 illustrates a DMA clear/write memory cycle, followed immediately by a DMA read/restore cycle, followed by a Micro 1600 processor read/restore memory cycle. The sequences shown are not necessarily typical, but serve to define all the DMA signal and timing requirements.

At the start of the timing sequence the memory is not busy, and the DMA request (DMAR/) and CPU request for memory service (MYXX) occur during the same clock period (t_0 to t_1). Since the DMA channel has priority over the CPU for memory service, the DMA is granted service before the CPU in cases of simultaneous requests. The DMA receives memory service on the SCK6 computer clock pulse t_1, provided that the timing requirements defined in Paragraphs 3.3.1 and 3.3.2 are met.

The SCK6 clock pulses occur every 200 nanoseconds, and five pulses (one microsecond total) occur during each complete read/restore or clear/write memory cycle. Therefore, the DMA is granted service for the second memory cycle (read/restore) on clock pulse t_6 which occurs one microsecond after time t_1 (start of the first memory cycle). The processor memory service request is not answered until the end of the second DMA cycle (t_{11}). The CPU has been locked out of memory service for two consecutive memory cycles by the DMA. Since the DMA is not requesting memory service for the third memory cycle starting at time t_{11}, the processor is allowed to perform its read/restore operation at this time.

During the two DMA cycles, the processor operation freezes. This is accomplished by stopping or inhibiting certain computer clocks with Clock Stop signal CSTP/ so that the memory type microcommand (MYXX) is not executed. Microcommand MYXX remains true until CSTP/ is removed at times t_{10} to t_{11}, allowing DMA use of memory during CSTP/. It is assumed that a non-memory type microcommand will be decoded and executed during clock period t_{11} to t_{12}.

3.3.1 CLOCK SIGNALS

The clock signals SCK6, CPH1, and CPH2/ are used by the Micro 1600 processor, Memory Control Interface, and the DMA interface to time and synchronize all memory operations. These clocks are generated by a single crystal oscillator clock generator located on the CPU control board.

SCK6 is the main CPU clock used for the reference clock signal in all timing requirements and is used to generate CPH1 and CPH2/. Signal SCK6 has a pulse width of 33 nanoseconds and a pulse period of 200 $\pm$0.02 nanoseconds.

CPH1 and CPH2/ are phased 50/50 square wave clock signals. They are made available to the DMA interface via any of the backplane connectors into which the DMA interface board may be installed. Combining CPH1 and CPH2/ produces a clock pulse similar to and almost in phase with SCK6. (The low-to-high transition of CPH2/ occurs 5 to 17

nanoseconds after the high-to-low transition of SCK6.) The clock developed from CPH1 and CPH2/ is used in the DMA interface to time the channel's control functions in synchronism with the processor and Memory Control Interface.

3.3.2 DMA PORT SIGNALS

The following paragraphs contain detailed descriptions and timing requirements of DMA Port signals generated in the DMA interface. Figure 3-3 is referenced as an aid in conveying the timing relations. The signal list in Appendix A of this manual lists the backplane connector pin location for each DMA Port signal. Appendix B provides a summary definition of each signal.

3.3.2.1 DMA REQUEST (DMAR/). Signal DMAR/ directly generates Memory Stop signal MSTP during clock pulse t_1. Signal MSTP is combined with MYXX to product Clock Stop signal CSTP/. Signal CSTP/ in turn, inhibits processor execution of MYXX and fetching of another microcommand. Signal DMAR/ also enables setting of Memory Control Interface flip-flops RTAX, MRTX, and MWTX on clock periods t_1 and t_6 to begin each DMA memory cycle. The Memory Busy (MBSY) and Memory Stop (MSTP) signals are held true during the DMA cycles by the true states of RTAX and MWTX.

DMAR/ must go low not later than 60 nanoseconds before the leading edge (low-to-high) of SCK6 pulse t_1, and remain true until 20 nanoseconds after the trailing edge of t_6 in order to initiate a DMA cycle of time t_1. It must return to the false (high) state before the leading edge of time t_2. Once DMAR/ goes high after initiating a DMA memory cycle, it must remain high until after memory becomes not busy (MBSY = low level) at the end of the current DMA cycle.

The DMAR/ line must be driven by a power gate (SN7438 or equivalent) with an uncommitted collector. The termination resistor is located in the processor which allows the line to swing between +5 volts and virtually 0 volt.

3.3.2.2 DMA SELECTION (DMAS/). Signal DMAS/ directly disables the CPU M and N Register address gating and the memory read and write data gates, and inhibits selection of the M and N Registers for data transfers during the time the Memory Busy Internal signal (MBYI) is true. DMAS/ also inhibits generation of Clock Stop (CSTP/) if any microcommand other than the memory type (MYXX) is decoded.

The DMAS/ signal must go low no later than 70 nanoseconds before the leading edge of SCK6 clock pulse t_1. It must remain true until the memory goes not busy (MBSY = low level) after the trailing edge of clock pulse t_5.

The DMAS/ line must be driven by a power gate (SN7438 or equivalent) with an uncommitted collector. The termination resistor is located in the processor and allows the line to swing between +5 volts and virtually 0 volt.

3.3.2.3 DMA WRITE (DMAW/). When DMAW/ is true (low level), it is combined with the DMAR signal (DMAR/ inverted) in the Memory Control Interface to enable a clear/write memory cycle. (See Figure 3-3, time t_0.) A false (high level) state of DMAW/ enables a read/restore DMA memory cycle, as shown at time t_6.

Signal DMAW/ must be stable in the desired state no later than 55 nanoseconds before the trailing edge of SCK6 clock pulse t_1. It must remain true until the memory goes not busy (MBSY = low level) after the trailing edge of clock pulse t_5.

The DMAW/ line must be driven by a power gate (SN7438 or equivalent) with an uncommitted collector. The termination resistor is located in the processor which allows the line to swing between +5 volts and virtually 0 volt.

3.3.2.4 MEMORY BUSY (MBSY). MBSY is generated by the Memory Control Interface to inform the DMA Interface when memory is not busy so a DMA request can be issued. MBSY is used in the DMA Interface to inhibit DMAR/ and DMAS/ until MBSY goes false (low level).

MBSY will go true (high) at the DMA interface 77 nanoseconds after the leading edge of the SCK6 clock pulse on which the memory cycle begins (t_1, t_6, and t_{11} of Figure 3-3). It remains true for four clock periods during a full memory cycle and goes false no later than 77 nanoseconds after the fifth clock pulse. During half memory cycles (initiated and controlled by the processor only), MBSY remains true for two clock periods, going false not more than 77 nanoseconds after the thrid clock pulse.

The MBSY line must be terminated in a single unit load (one TTL gate input) which is equivalent to a maximum load of 2 ma.

3.3.2.5 MEMORY ADDRESSES. The DMA Interface must have the current address on the Memory Address bus at least 15 nanoseconds before the leading edge of the SCK6 clock pulse on which the DMA memory cycle begins (t_1 and t_6). The address must remain stable on the bus until MBSY goes false (low).

Memory address lines M07A/ through M00A/ and N07A/ through N00A/ are terminated on the computer backplane. The lines must be driven by power gates (type SN7438 or equivalent) with uncommitted collectors. Because of the termination network, the lines are allowed to swing between virtually 0 volt and +3.6 volts.

3.3.2.6 WRITE DATA. During clear/write operations, the DMA Interface must place the desired write data on the bidirectional memory data bus no later than the leading edge of the t_3 clock pulse (Figure 3-3). The data must remain stable until MBSY goes false (low).

Bidirectional memory data lines MD07 through MD00 are terminated on the computer backplane. The memory write data must be driven to the memory data bus by power gates (type SN7438 or equivalent) with uncommitted collectors. Because of the termination network, the lines are allowed to swing between virtually 0 volt and +3.6 volts.

3.3.2.7 READ DATA. The data read from memory during a read/restore cycle will be available on the bidirectional memory data bus 375 nanoseconds after the leading edge of the SCK6 clock pulse on which the cycle started (t_6 of Figure 3-3). The data will remain stable for gating into the DMA interface until the trailing edge of the DMA clock pulse corresponding to SCK6 clock pulse t_9.

Read data, received from the bidirectional memory data bus, must be buffered by gates whose inputs load each line with only one unit load (one TTL gate input) that is equivalent to a maximum load of 2 ma.

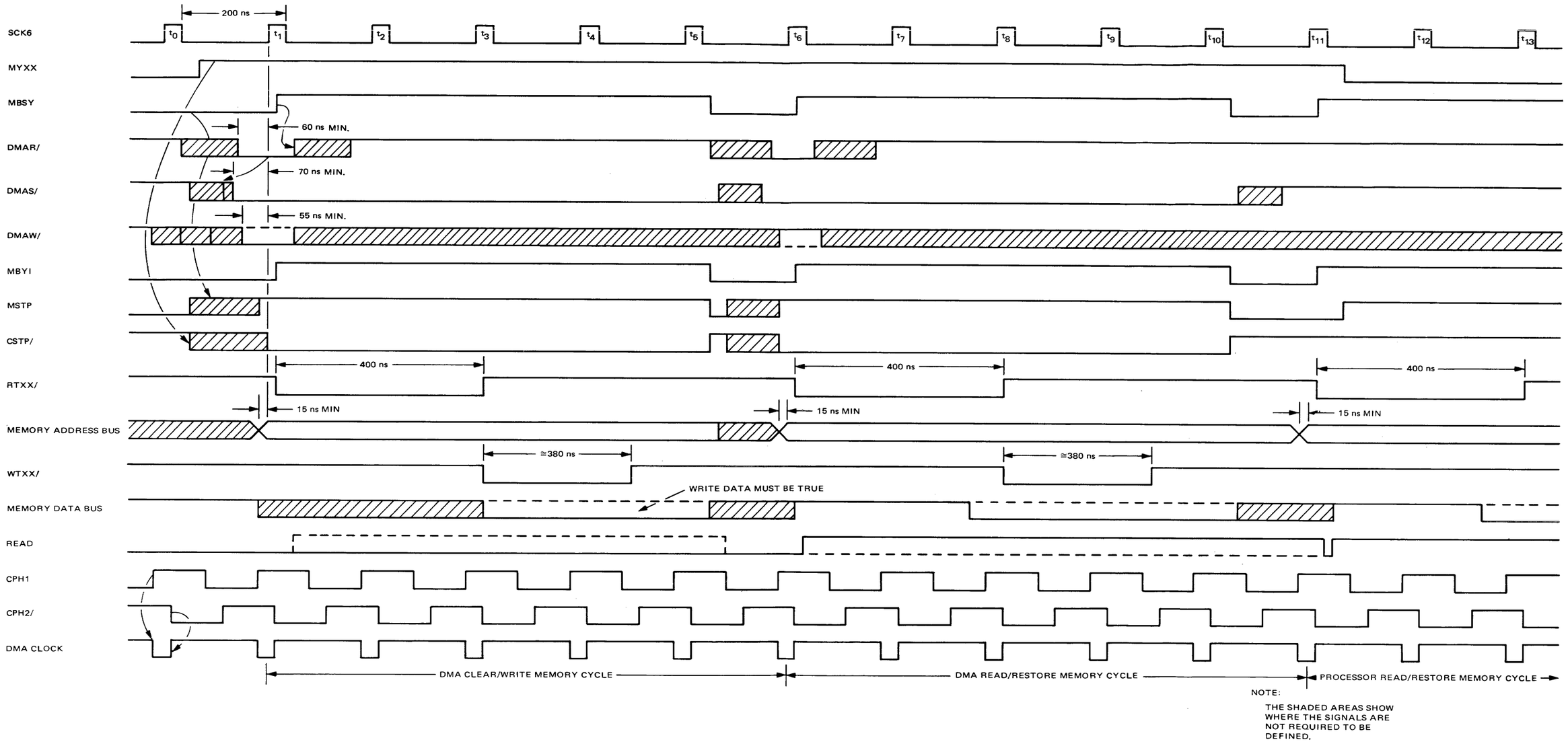


Figure 3-3. DMA Port/Memory Control Timing

SECTION 4

SERIAL I/O INTERFACE

4.1 INTRODUCTION

The Serial I/O Interface is an optional feature of Micro 1600 series computers. It is a hardware/firmware option which, through microprogramming, can be used to control a serial device such as a Teletype or modem.

The serial channel is mounted on the Interface Board containing the front panel interface circuitry. The serial channel circuitry occupies the portion of the card which normally contains the standard Integral Teletype Controller. The user may order either the serial channel or the Integral Teletype Controller, but not both.

4.2 MICRO 800/1600 COMPATIBILITY

The serial I/O channel was originally designed as a low-cost teletype controller for Micro 800 computers. It has been replaced in Micro 1600 computers by the standard Integral Teletype Controller, but is still available for Micro 1600 systems where total 800/1600 compatibility is required.

The Micro 1600 firmware is identical to the firmware in Micro 800 series computers. The microprogram which controls the serial channel for operation of a 110-baud teletype with the Micro 800 is also present in the Micro 1600. Because of the faster clock in the Micro 1600, the standard firmware produces a transfer rate which is not teletype compatible. Therefore, use of the serial channel necessitates firmware changes, or slowing the Micro 1600 master clock to the rate of the Micro 800 clock.

4.3 USE AS TTY CONTROLLER

The following paragraphs describe operation of the serial I/O channel with a four-wire, full-duplex, 20-ma teletype. A cable is provided with the serial channel to connect directly to the Teletype.

4.3.1 GENERAL OPERATION

The four-wire I/O interface circuit is shown in Figure 4-1. The transmit portion of the circuit contains a 20-ma current source that can be turned on or off depending on the state of the I/O control register. When the I/O control register is in any state other than state 3, output of gate Z2 is high, emitter follower Q1 conducts, and approximately 20 ma of current flows through resistor R8. This current holds the Teletype in the mark condition. When the I/O control register is set to state 3 by a microcommand, the output of gate Z2 is low, emitter follower Q1 cuts off, and no current flows to the Teletype.

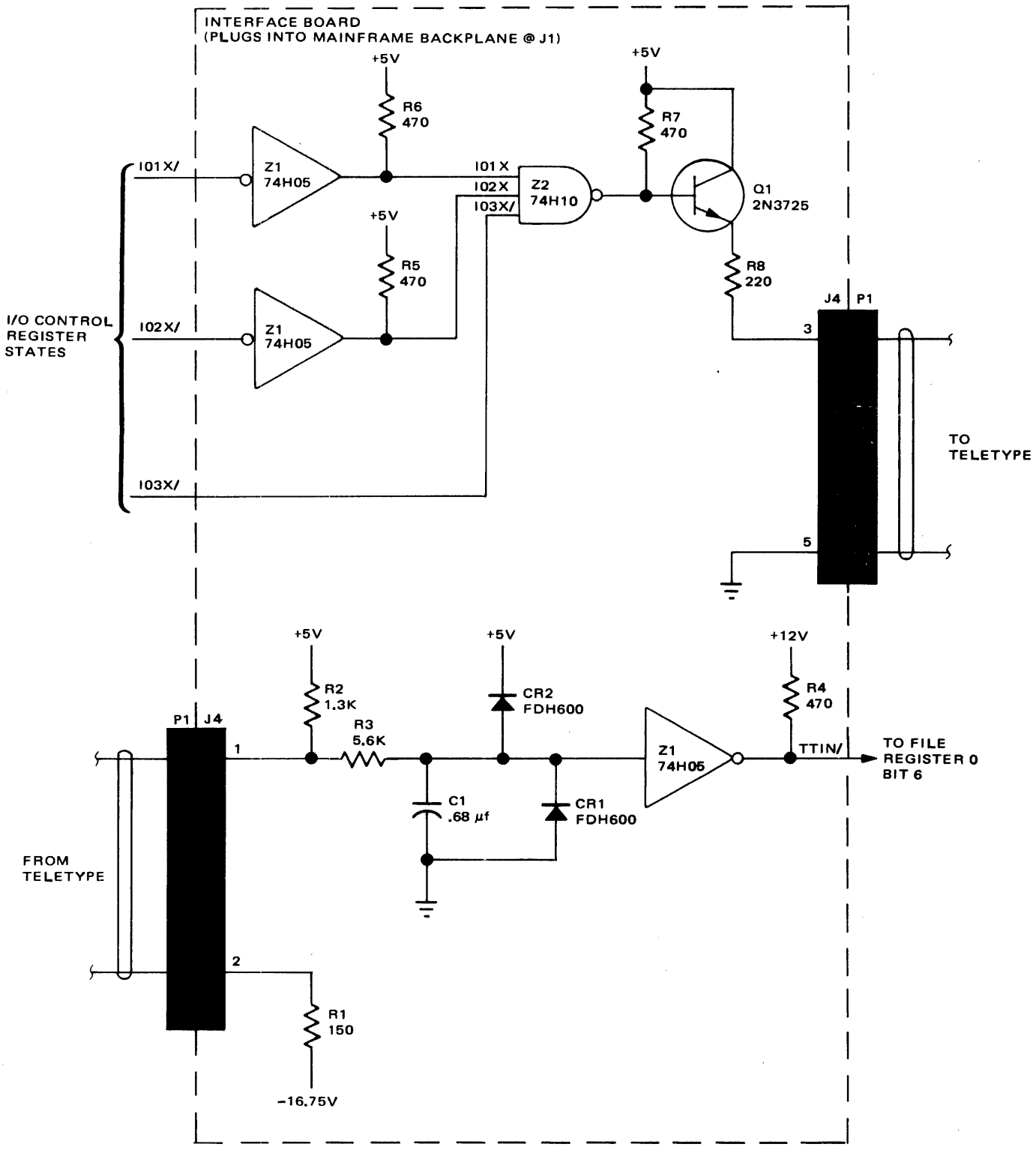


Figure 4-1. Serial I/O Interface Circuit (For ASR 33 TTY)

The receive portion of the interface circuit contains a low-pass filter network connecting the teletype distributor to bit 6 of File Register 0 where it may be sensed by microcommands. One side of the teletype distributor is connected to -16.75 volts through resistor R1. The other side of the distributor is connected to Z1, which forms bit 6 of File Register 0. When the Teletype sends a mark signal, the output of Z1 is held low and a zero bit appears in bit 6 of File Register 0. When the Teletype sends a space signal, a one bit appears in bit 6 of File Register 0.

4.3.2 CHARACTER ASSEMBLY AND DISASSEMBLY

Teletype character assembly, disassembly, synchronization, and timing is accomplished by a firmware routine initiated by the macro instructions for the serial I/O interface.* Figure 4-2 shows the timing for transmitting or receiving 110-baud teletype characters.

*The Micro 1600/10 and 1600/13 are the only standard firmware which have these serial I/O macro instructions.

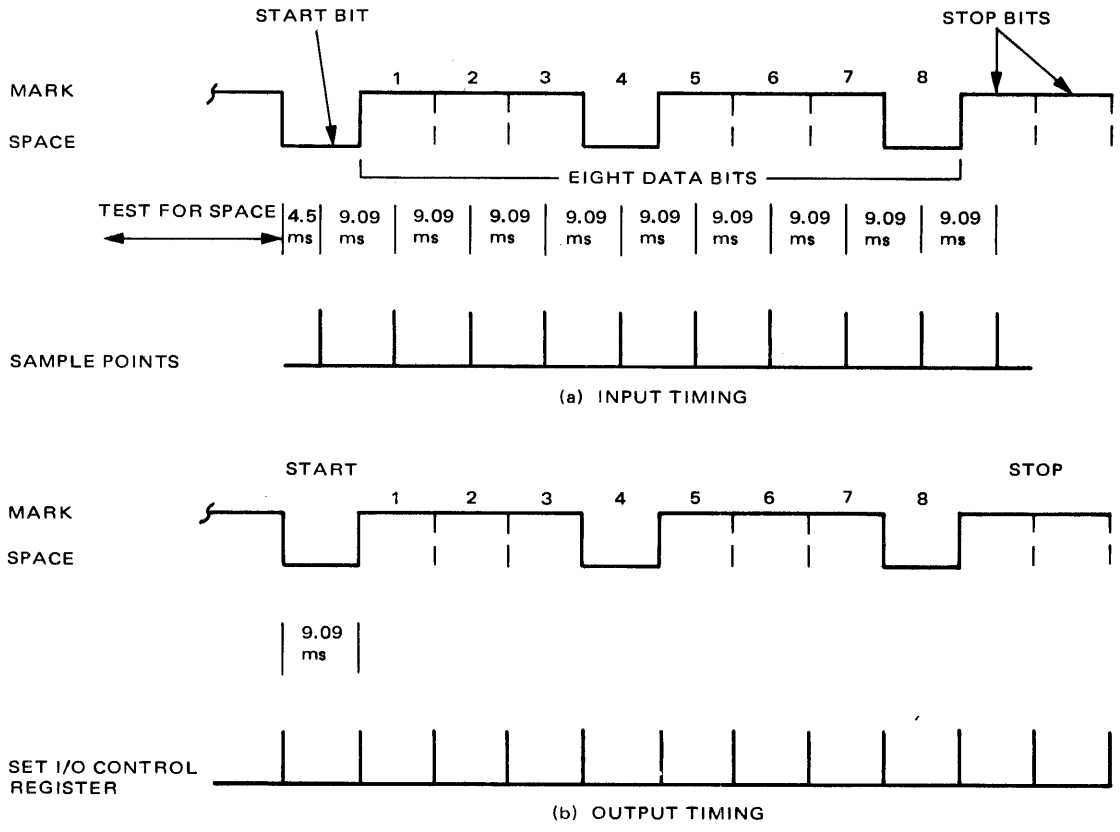


Figure 4-2. Serial I/O Timing

During an input operation the firmware program searches for the leading edge of the start bit by continuously testing bit 6 of File Register 0. Once a space level is detected, the firmware program delays 4.5 milliseconds and samples the input every 9.09 milliseconds, shifting each bit into the least significant byte of the A Register (File Register 4). The initial delay of 4.5 milliseconds, after detecting the leading edge of the start bit, causes sampling to occur in the middle of each bit. The firmware routine exits after eight bits have been assembled.

During an output operation, the firmware program sets the I/O control register to the appropriate mark or space condition every 9.09 milliseconds according to the start and stop bits and the data to be serially transmitted. Before the first information bit is transferred, the I/O control register is set to mode 3 to transmit the start bit. The firmware program for transmitting a teletype character remains active for 11 intervals (100 milliseconds) to assure the proper stop interval before the next character is transmitted.

4.3.3 SERIAL I/O INSTRUCTIONS

Two macro instructions affect the operation of the serial I/O interface: Input Byte Serially (IBS), and Output Byte Serially (OBS).

The Input Byte Serially instruction transfers an eight-bit character from the Teletype into the eight low-order bits of the A Register. The execution of this instruction terminates when a complete teletype character has been received. For proper operation, execution of the instruction must be started before the start of the teletype character. Once the instruction is started, the computer becomes tied up until a teletype character is received. The execution time of the instruction extends approximately 84 milliseconds after the leading edge of the teletype character start bit. When the program echoes input characters back to the Teletype, the effective input rate cannot exceed five characters per second, (no input can be handled during the 100 milliseconds required for output).

The Output Byte Serially macro instruction disassembles the eight low-order bits of the A Register and transfers them serially, as a teletype character, through the serial I/O interface. During the execution of this instruction, the eight low-order bits of the A Register are set to ones, the eight high-order bits remain unchanged.

4.3.4 TELETYPE INTERFACE CONNECTION

The standard Teletype model ASR-33TY with 20 ma loop interface is directly compatible with all Microdata TTY controllers. Procedures for modifying other standard ASR-33 or ASR-35 Teletypes for use with Micro 1600 series computers are given in Appendix C.

SECTION 5

CONFIGURATION AND INSTALLATION

5.1 INTRODUCTION

The Micro 1600 series computers are designed for packaging in a variety of configurations to meet a wide range of system mounting requirements. Entire computer systems, including several types of peripheral devices, can be mounted in standard 19-inch computer cabinets, while smaller systems may be completely contained in a compact desk-top enclosure. Each Micro 1600 packaging scheme has been designed around expandability to ensure minimum costs as systems grow to fill new requirements. Cabinet-mounted and desk-top systems are shown in Figure 5-1. This section describes the various packaging configurations available and the cabling used to interconnect Micro 1600 systems.

5.2 CIRCUIT BOARDS

All CPU logic, core and control memory, computer options, and Microdata I/O controllers are constructed on 8.575-inch by 12.50-inch printed circuit (PC) boards which plug side by side into card slots in the computer mainframe chassis and/or expansion chassis. Each PC board, except the alterable control memory (ACM) board, requires one card slot. The ACM, because of its high power dissipation and added cooling requirements, must have a vacant card slot on each side.

Note

To ensure propagation of prioirty lines PROT/ and SELO/ along the chassis backplanes to all I/O controllers, the card slot between the CPU data board and any I/O controller cannot be vacant. A short PC board, PN A20001202, is available for use in a vacant card slot to propagate the prioirty signals down the backplane.

In addition to the Micro 1600 printed circuit boards, two types of general purpose wire-wrap boards are available on which the user can construct I/O controllers of his own design. One of these boards utilizes IC sockets with pins extending through to the back of the board. The user installs the IC modules in the sockets and connects the socket pins using conventional wire-wrap techniques. This type of board has a capacity of 135 14- or 16-pin sockets, and 10 24-pin or 4 40-pin dual in-line sockets. When this board is installed in a chassis back-plane, one card slot must remain vacant on the non-component side because of the length of the wire-wrap pins. (Refer to the above note.)

CABINET-MOUNT

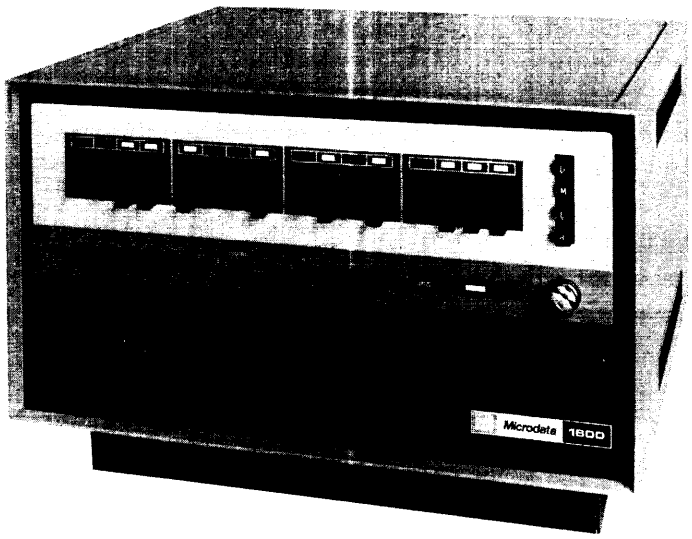


TABLE-TOP

Figure 5-1. Cabinet-Mount and Table-Top Micro 1600 Computers

On the second type of wire-wrap board, IC sockets are soldered to the front side between the two rows of wire-wrap pins (also on the front of the board). This board has a capacity of 64 16-pin sockets and 6 24-pin sockets. Because wire-wrap pins are located on the front of the board with the IC sockets, only one backplane card slot is required.

Figure 5-2 shows typical printed circuit and wire-wrap boards used in the Micro 1600 series computers. The main circuit boards used in Micro 1600 systems can be classified into five categories according to their functions:

• Central Processor Unit (CPU)	2 PC boards
• Core Memory (up to eight 8K modules)	1 PC board per module
• Control Memory (firmware)	1 PC board
• Interface Board (expansion chassis receivers/ drivers and integral teletype controller)	1 PC board
• I/O controllers	1 or 2 PC/wire-wrap boards per controller

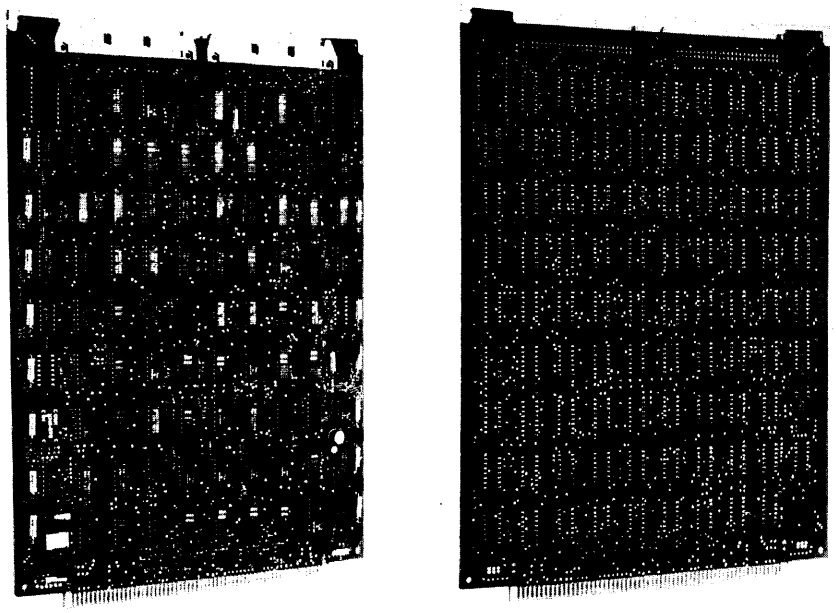
5.3 MAINFRAME CHASSIS (SINGLE-CPU SYSTEMS)

Two types of mainframe card cage chassis are available to house the Micro 1600 CPU boards, interface board, core and control memory boards, and a number of I/O controllers and options. Selection of mainframe chassis for single-CPU Micro 1600 systems is based primarily on the number of circuit board slots required. The mainframe chassis used in Micro 1600D dual-CPU systems is discussed in Paragraph 5.5.

5.3.1 BASIC CARD CAGE CHASSIS

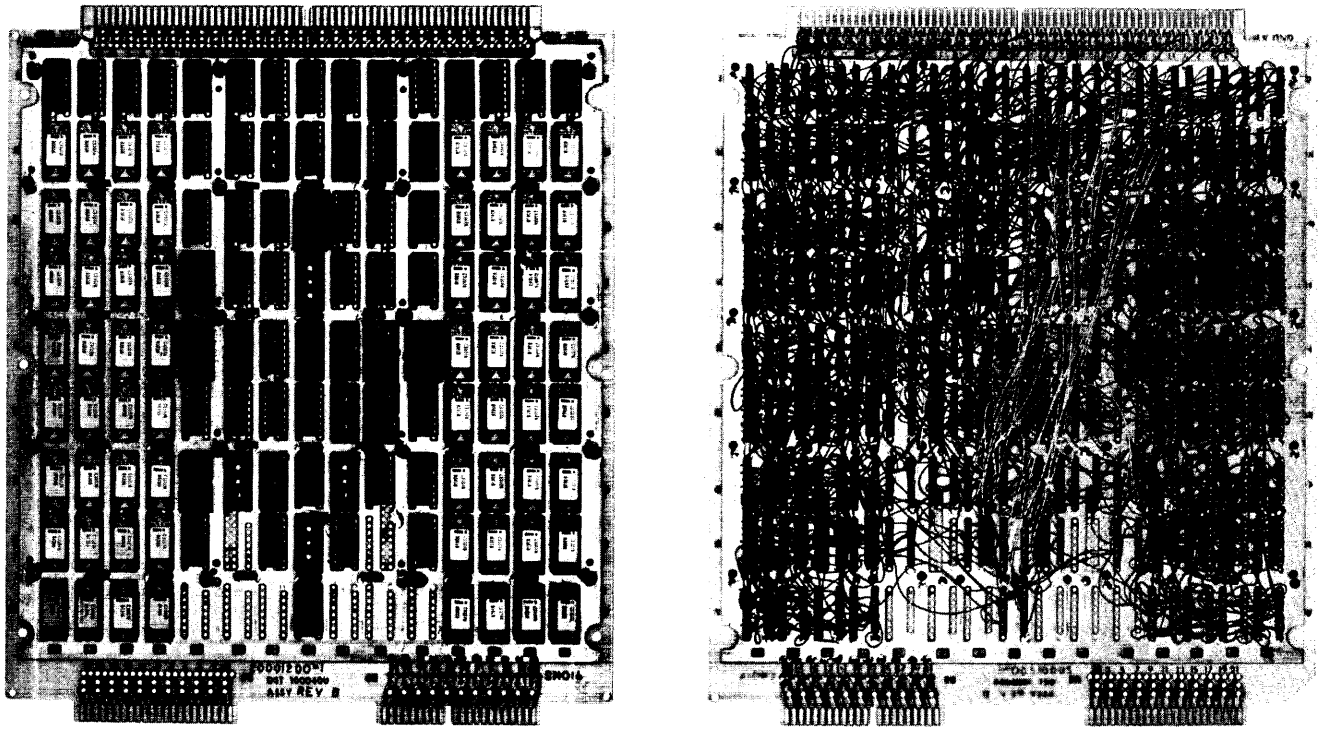
The basic card cage chassis, illustrated in Figure 5-3, is used where not more than 13 mainframe card slots are required. It is an open card cage chassis which contains a printed circuit backplane with 14 card slots, an integral power supply, and a front panel containing operator controls. A circuit card attached to the power supply attaches to card slot 14 to route dc power to the backplane, leaving 13 slots available for the various mainframe circuit boards.

Slot 1 of the mainframe always contains the interface board. If the system contains an expansion chassis, drivers and receivers extending the I/O bus to the expansion chassis are located on this board (and on a corresponding expansion interface board in J1 of the expansion chassis). The CPU control and data boards mount in slots 2 and 3, respectively. Starting at slot 4 and filling successive card slots are the core memory boards. The system may contain from one to eight core memory modules, each module containing either 8K or 4K bytes, providing from 4K to 65K bytes of core storage. Each system can contain only one 4K module which must occupy the uppermost slot used for core memory. The core memory module in slot 4 contains memory addresses starting at 0. Memory addresses are sequential through each successive memory module in the backplane.



FRONT REAR

PRINTED CIRCUIT BOARD



FRONT REAR

WIRE-WRAP BOARD

Figure 5-2. Typical Micro 1600 Circuit Boards

The ramining card slots are available for control memory (including ACM), I/O options and controllers, and a DMA interface. Any of these boards may be plugged into any slot; however, the order in the chassis of the I/O controllers/options determines their order on the external interrupt and concurrent I/O priority chain. I/O boards closest to the CPU boards have highest priority, with mainframe chassis boards having higher priority than expansion chassis boards. Figure 5-3 shows typical board placement in the basic card cage chassis.

The integral power supply in the basic card cage chassis provides dc power to all mainframe circuit cards. This supply is described in detail in Paragraph 5.7.

5.3.2 EXTENDED BACKPLANE CARD CAGE CHASSIS

Systems requiring more than the 13 card slots in the basic card cage may utilize an extended backplane chassis, illustrated in Figure 5-4. This chassis has an 18-card-slot backplane extending the entire width of the chassis. The dc power requirements for the chassis are provided by a remote power supply which mounts at the rear of the equipment cabinet. Power is cabled to a power distribution board which plugs into card slot 18. This leaves 17 slots available for system circuit boards.

The CPU boards, interface board, core and control memory boards, DMA interface, I/O controllers and options are installed in the same order as in the basic card cage, leaving four additional card slots for I/O controllers. Typical board placement in the extended backplane chassis is shown in Figure 5-4.

5.4 EXPANSION CHASSIS

When the system exceeds the capacity of the basic or extended-backplane mainframe chassis, the expansion chassis (Figure 5-5) is used to house I/O controllers/options. Dimensions of the expansion chassis are identical to the mainframe chassis, and construction is similar. The primary difference is the number of card slots in the backplane. Backplane connector slots, extending the entire width of the chassis, are provided for 21 cards. The dc operating voltages are provided by a remotely mounted power supply and are routed to the chassis by a power cable which is connected to lugs on the backplane.

The I/O bus is routed to the expansion chassis by a flat cable from the interface board in the mainframe. An expansion interface board at the end of the cable plugs into card slot 1 to distribute the bus onto the expansion chassis backplane.

Circuit boards in the chassis start in slot 2 and occupy successive slots across the backplane. The boards must occupy successive slots to ensure propagation of priority lines PROT/ and SELO/. The external interrupt and concurrent I/O priority of expansion chassis controllers is determined by their order on the backplane. Card slot 2 has highest priority, with slot 21 having the lowest. This same expansion chassis is also used in Micro 1600D dual-processor systems. Special dual-processor system considerations are discussed in Paragraph 5.5.

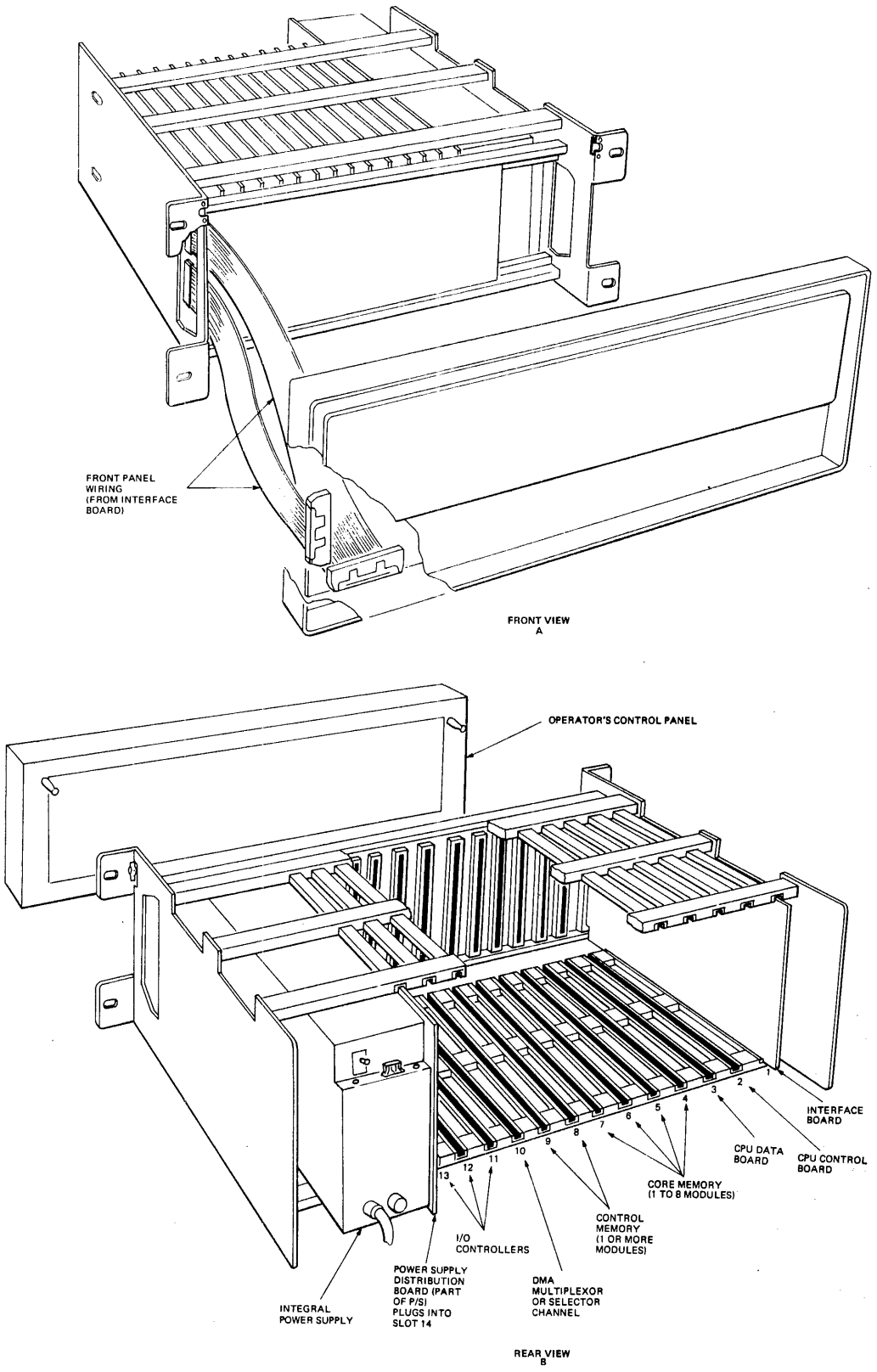


Figure 5-3. Basic Card Cage Mainframe Chassis and Integral Power Supply

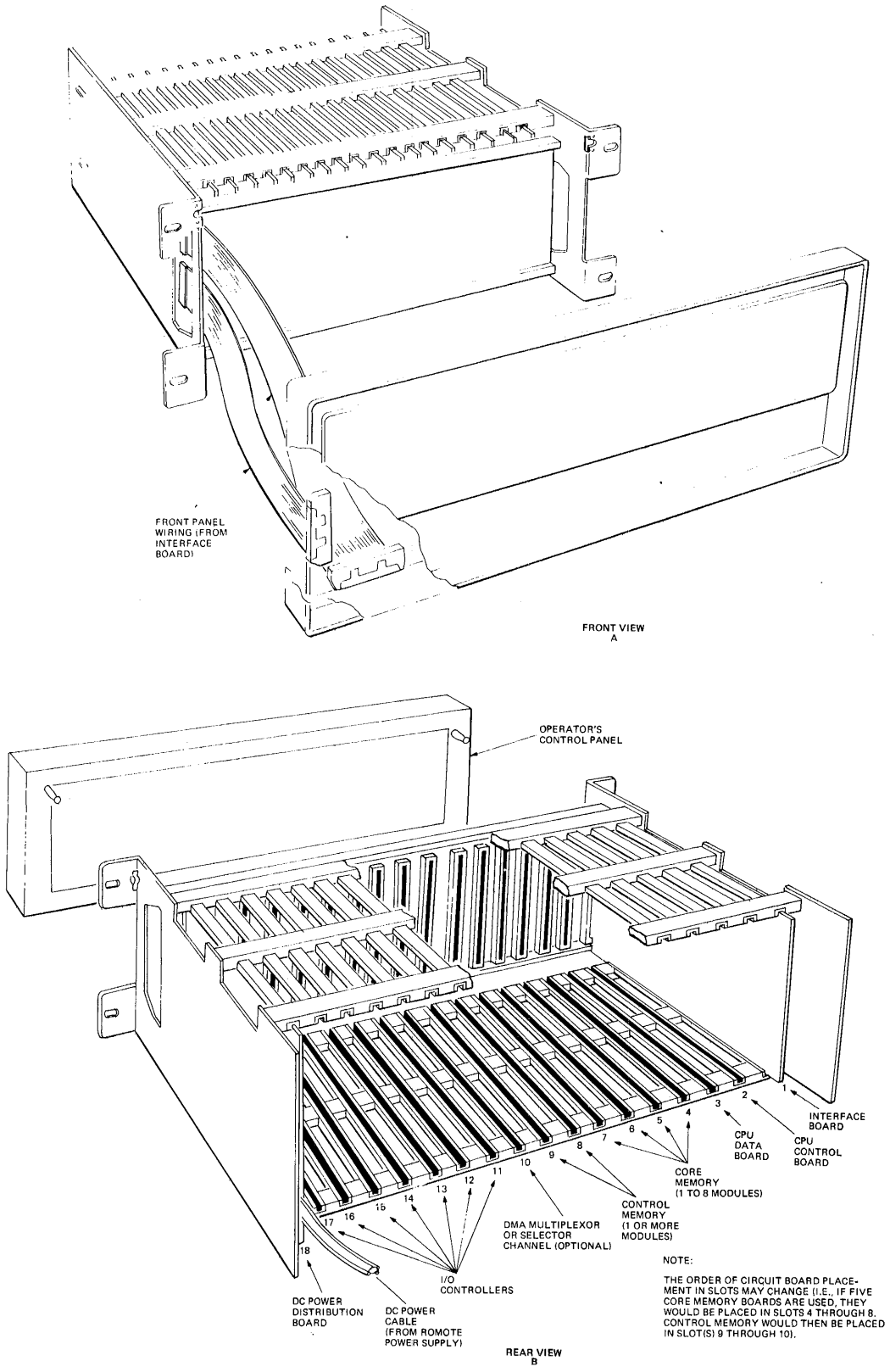


Figure 5-4. CPU Extended Backplane Card Cage Chassis

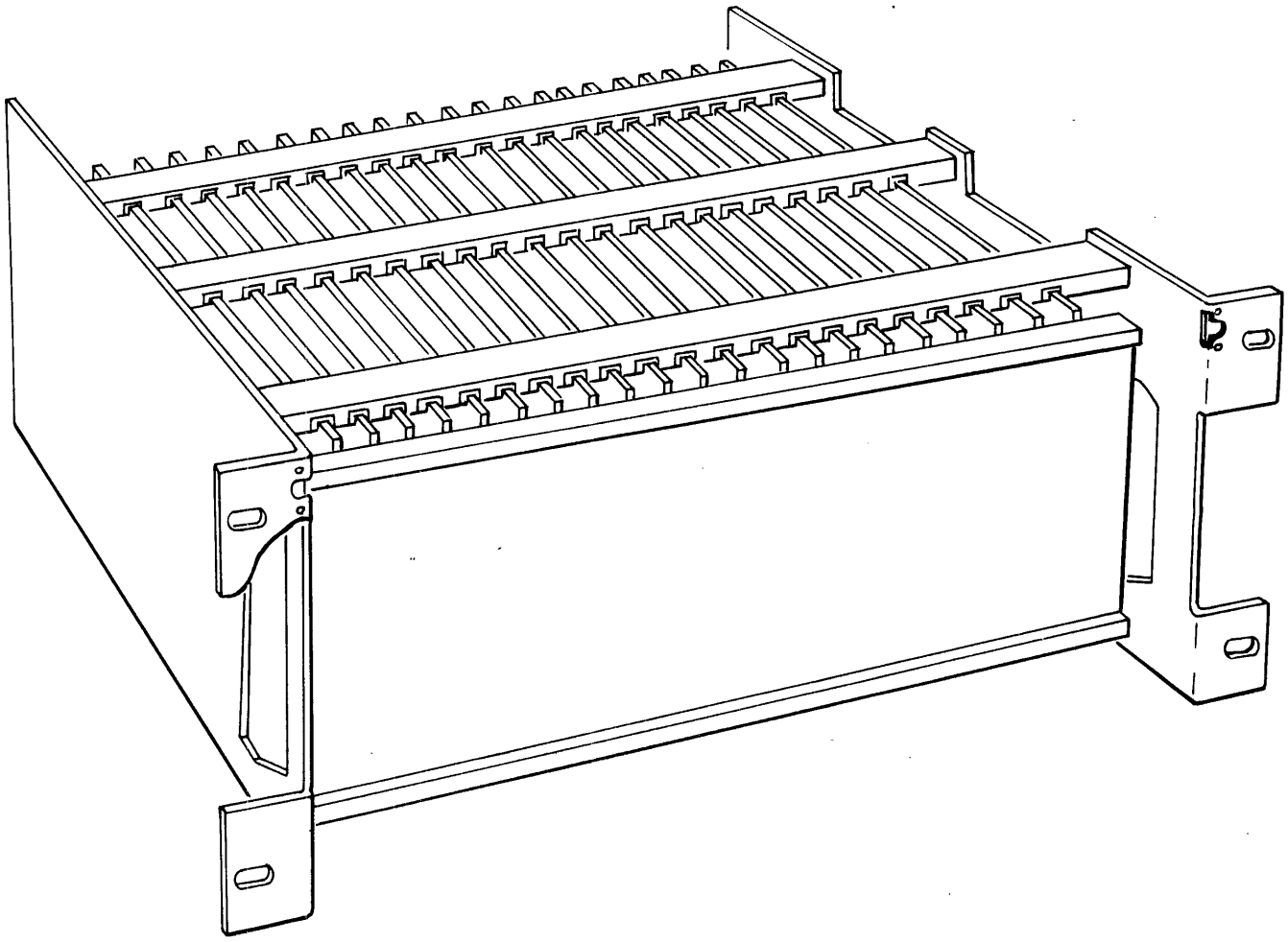


Figure 5-5. Expansion Chassis

5.5 MICRO 1600D DUAL-PROCESSOR SYSTEM

5.5.1 MAINFRAME CHASSIS

The Micro 1600D dual-processor system consists of two complete processor systems sharing a common core memory. The mainframe chassis (which houses the two CPUs, interface boards, core and control memories, and several I/O controllers and options) contains 21 backplane card slots, as does the optional expansion chassis.

As shown in Figure 5-6 and 5-7, the interface board and the CPU control and data boards for CPU A mount in the order prescribed for single CPU systems (starting in slot 1 of the mainframe). The CPU B interface board mounts at the opposite end of the chassis in slot 21, with the CPU B control and data boards occupying slots 20 and 19 respectively. The common core memory mounts on the CPU A side of the chassis, starting in slot 4 and occupying from one to eight successive card slots.

Core memory modules must be mounted in specific card slot locations. The designated memory slots which do not contain memory boards can be used for control memory, I/O controllers and/or a DMA controller (CPU A only). The assigned memory locations are:

Card Slot	Memory Addresses
4	0 - 8191
5	8192 - 16,383
6	16,384 - 24,575
7	24,576 - 32,767
8	32,768 - 40,959
9	40,960 - 49,151
10	49,152 - 57,343
11	57,344 - 65,535

Following the core memory on the CPU A side and the CPU boards on the CPU B side of the chassis are the A and B control memories, optional DMA interface (CPU A only), and I/O controllers/options. These boards may be located in any order in the remaining slots, as long as a CPU A/CPU B dividing line is observed. As in the single-CPU mainframe chassis, the order of I/O controllers and options on the backplane determines their interrupt and concurrent I/O priority.

Since the dual-CPU system maintains a separate control memory and I/O facility for each CPU, these buses on the backplane are separated at a point determined for each individual system. The core memory bus and system power bus, however, are common to both sides of the chassis.

Figure 5-6. Dual CPU System

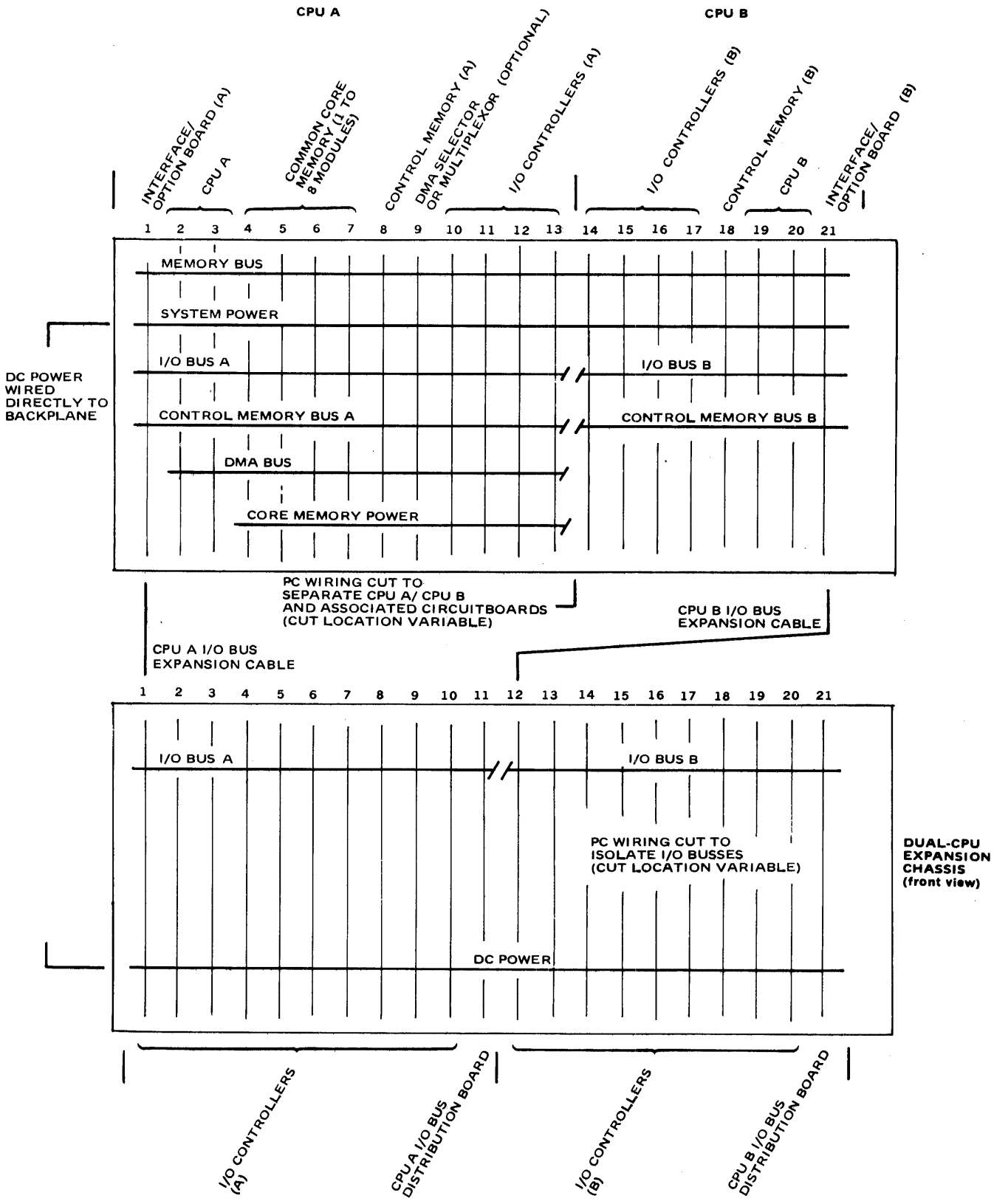


Figure 5-7. Dual-CPU Backplane Wiring

5.5.2 EXPANSION CHASSIS

Micro 1600D systems containing more I/O controllers/options than can be accommodated by the mainframe, utilize an optional expansion chassis. This is the same 21-slot expansion chassis used in single-CPU Micro 1600 systems.

In dual-CPU systems, one side of the expansion chassis is used for CPU A I/O controllers and the other side is used for CPU B controllers. The backplane printed wiring for the I/O bus is separated at some point (determined for each individual system) to isolate the A side from the B side. The system power bus, however, is common to the entire backplane.

The CPU A and CPU B I/O buses are routed to the expansion chassis by cables from the interface boards in the mainframe. An expansion interface board on the expansion chassis end of each cable plugs into the backplane. The expansion interface board from CPU A plugs into card slot J1 followed by A controller boards in successive card slots. The expansion interface board from CPU B mounts in the card slot immediately following the I/O bus separation point. Successive slots are filled with CPU B I/O controllers.

External interrupt and concurrent I/O priority is determined by the order of the controllers in the expansion chassis. The expansion chassis controllers always have lower priority than controllers in the mainframe. Controllers nearest each I/O bus distribution board have highest expansion chassis priority for each respective CPU.

5.5.3 DUAL-CPU SYSTEM POWER

The dc operating voltages for the mainframe and expansion chassis are provided by remotely mounted power supplies (one for each chassis). The dc power cables from these supplies are wired directly to the chassis backplanes.

5.6 CHASSIS ENCLOSURES

Two types of wrap-around enclosures are available for all Micro 1600 series card cage chassis for desk-top and cabinet mounted installations. Figures 5-8 and 5-9 show the two types of enclosures installed on card cage chassis.

The desk-top and cabinet-mounted enclosures for the basic card cage chassis contain a cooling fan at the back to ensure proper circulation through the chassis. (The integral power supply contains a second fan.) Enclosures for chassis without integral power supplies contain two cooling fans at the rear of the enclosure. Adequate cooling of the Micro 1600 is extremely important, and these fans should be used in both cabinet-mounted and desk-top installations. These fans can be snapped out of the back panel of the enclosure and hung from the top rear of the enclosure while servicing PC boards on extenders.

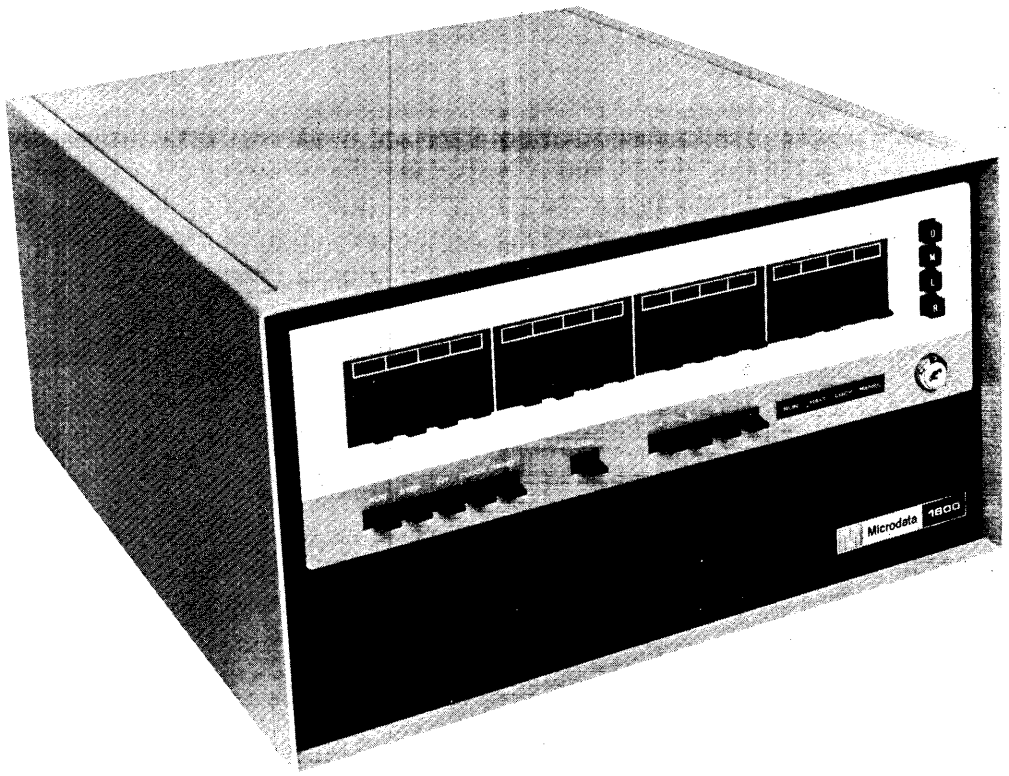


Figure 5-8. Desk-Top Wrap-Around Enclosure

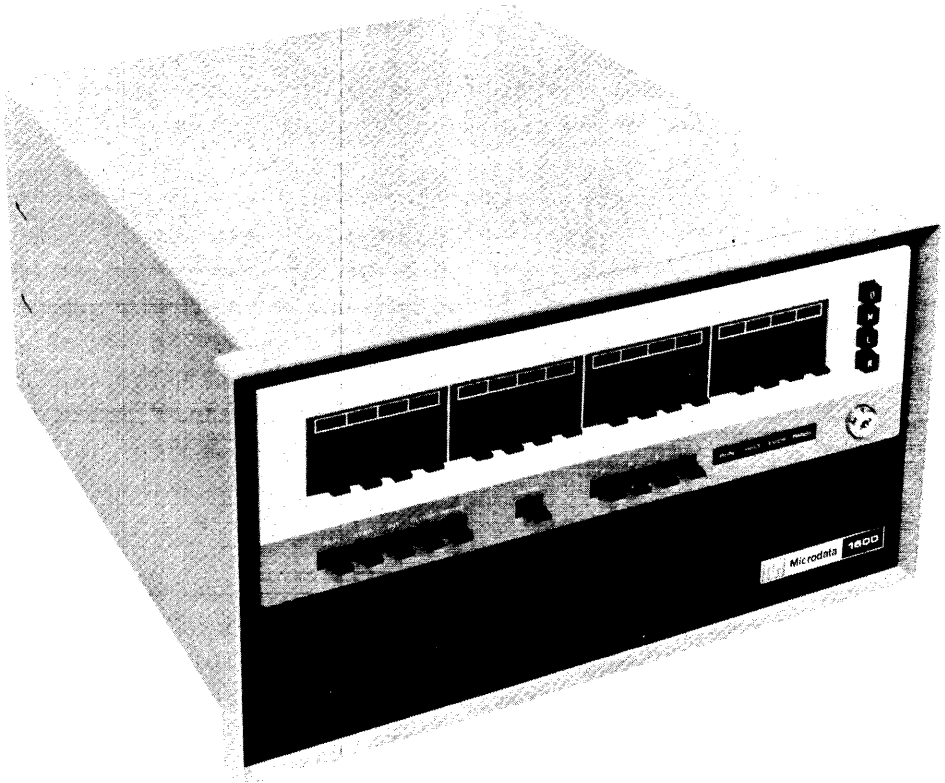


Figure 5-9. Rack-Mount Wrap-Around Enclosure

5.7 POWER SUPPLIES

Two basic types of power supplies are available for use in Micro 1600 systems; the integral power supply used in the basic mainframe chassis, and the external (remote) power supply used to power the extended-backplane mainframe and the expansion chassis. The integral power supply is shown in Figure 5-3. The remote power supply shown in Figure 5-10 is mounted on the rear mounting rails of a Microdata equipment cabinet.

The voltages and current capabilities of the two basic power supplies are listed below:

+5 Vdc @ 20 amperes
+12 Vdc @ 3 amperes
-16.75 Vdc @ 3.5 amperes

The mainframe chassis for dual-CPU systems is powered by a supply identical in dimensions and mounting, but which provides 40 amperes at +5 volts. (The +12 volts and -16.75 volts current capabilities are the same as the basic supplies.)

5.8 SPECIAL ACM CONSIDERATIONS

As previously explained, the alterable control memory (ACM) circuit board requires three card slots in the computer mainframe chassis to provide additional air circulation around the board. In addition, an ACM cooling unit must be located immediately below the mainframe chassis. The cooling unit contains fans which aid the upward circulation of cooling air through the mainframe and around the ACM board.

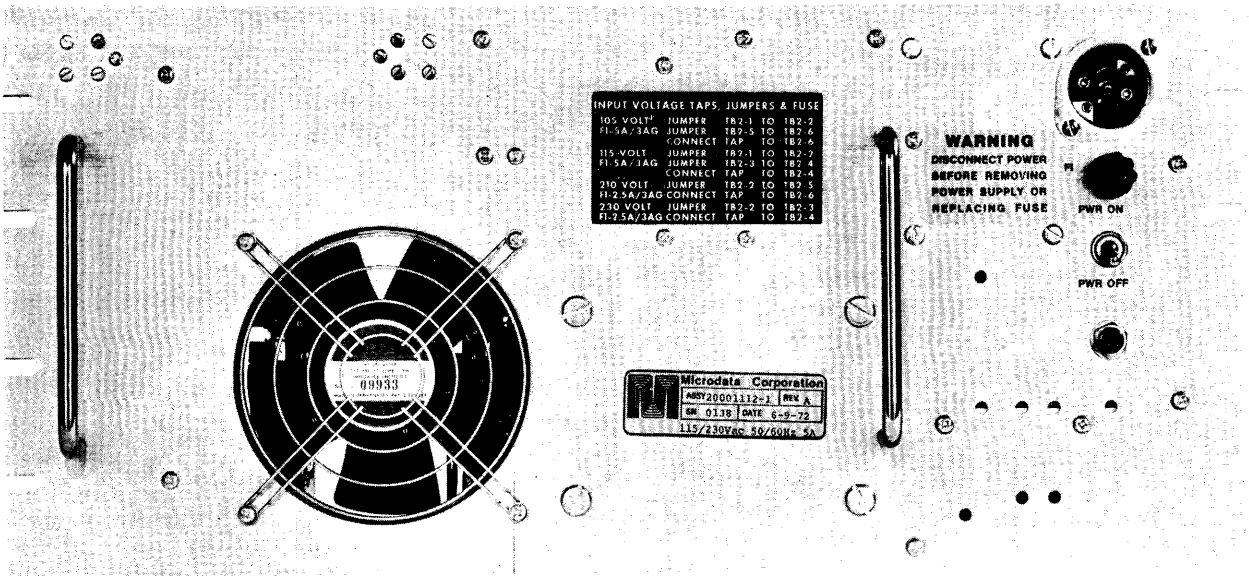
Note

The mainframe chassis must be used without the wrap-around endlosure in order for the cooling unit to be effective.

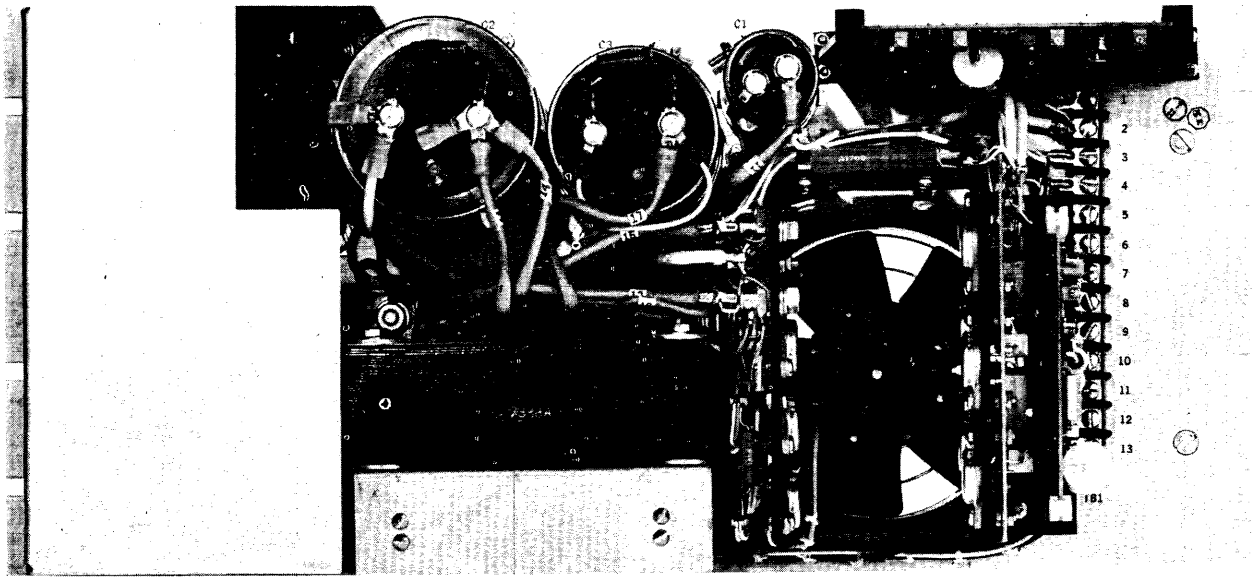
An ACM control panel, which gives the operator the necessary control of the ACM and fixed ROM, mounts on the front of the 3.5-inch high ACM cooling unit. Because of the high current requirement of the ACM board, its operating voltages are provided by an ACM power supply, physically identical to the standard remote power supply. The dc power from the ACM power supply routes to a terminal board on the rear of the ACM cooling unit. This terminal board routes the operating voltages to the ACM board and signals from the ACM control panel to the ACM and the fixed ROM boards. Mounting and interconnection of the ACM board and ACM cooling unit is shown in Figure 5-11.

5.9 CABINET MOUNTING

All Microdata chassis and rack-mount wrap-around enclosures are designed for mounting in Microdata computer cabinets (or other cabinets with standard 19-inch-wide-mounting rails). With the installation of cabinet mounting hardware (Figure 5-12), the enclosures mount so they slide out to the front for easy servicing. The ac power cords plug into grounded sockets on strips within the cabinet.



FRONT VIEW



REAR VIEW

Figure 5-10. Remote Power Supply

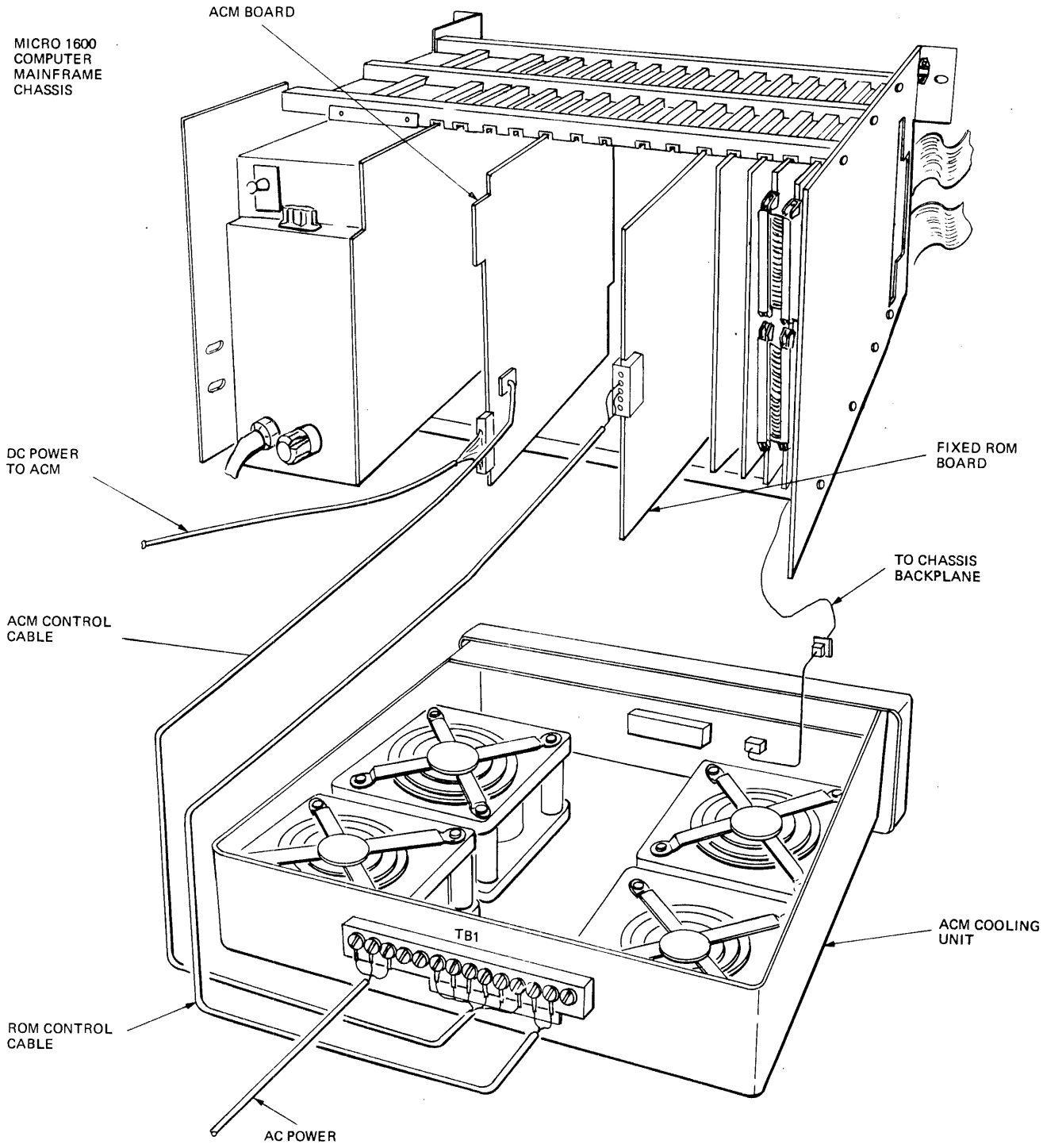


Figure 5-11. Alterable Control Memory (ACM) Installation

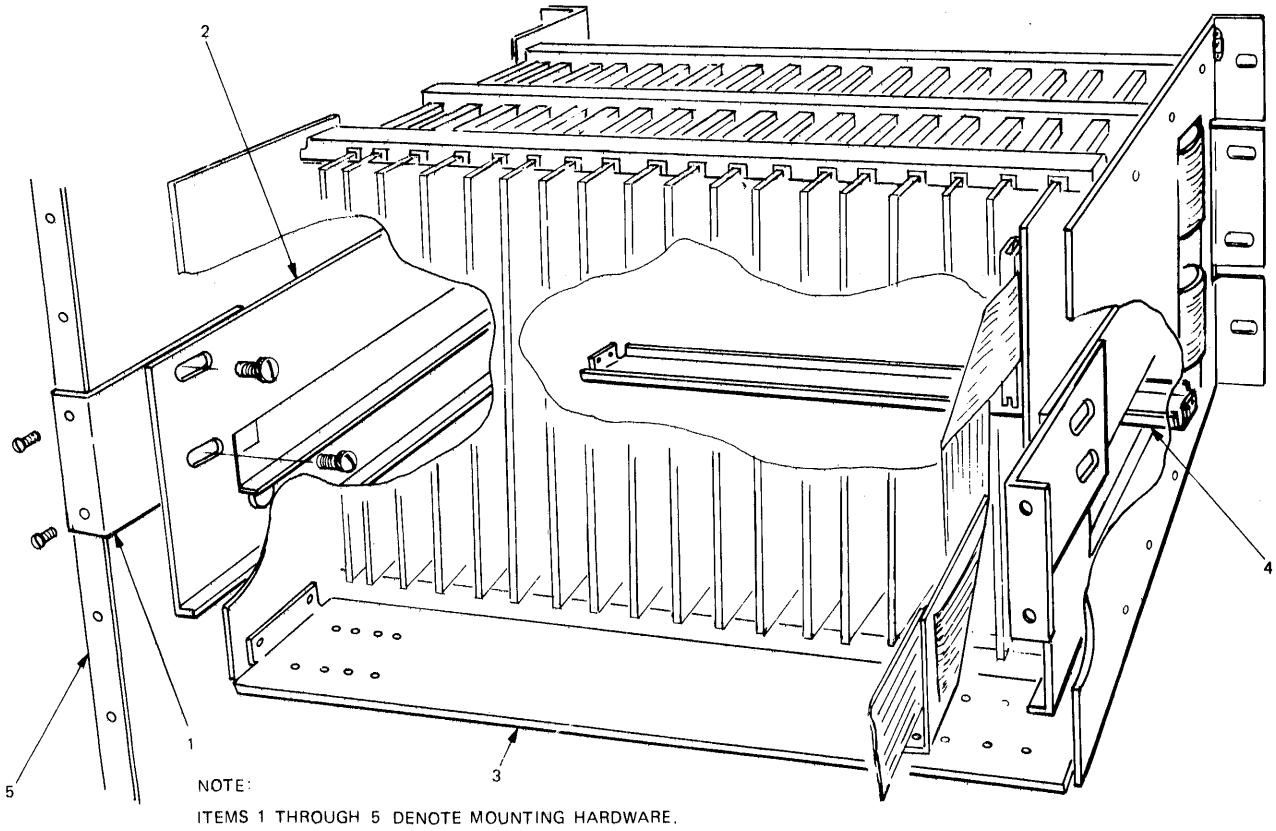


Figure 5-12. Cabinet Mounting Hardware

The Microdata computer cabinets are available in both single-bay (Figure 5-13) and double-bay (Figure 5-14) versions that are 24 inches, 26 inches, 28 inches, or 30 inches deep. (The figures show 24-inch versions.) All cabinets feature a convenient operator work surface at normal desk height. Fans located in the bottom on the cabinet provide upward air circulation for cooling of the units mounted within the cabinet.

5.10 GROUNDING CONSIDERATIONS

The Micro 1600 computer logic is isolated from earth ground (floating) to give the user the option of grounding the system at the point of his choosing. Grounding must be accomplished either at some point on the peripheral equipment or by strapping the computer power supply. Failure to provide a return to earth ground at some point will cause problems inherent in a floating system (such as increased RFI sensitivity and cross-talk).

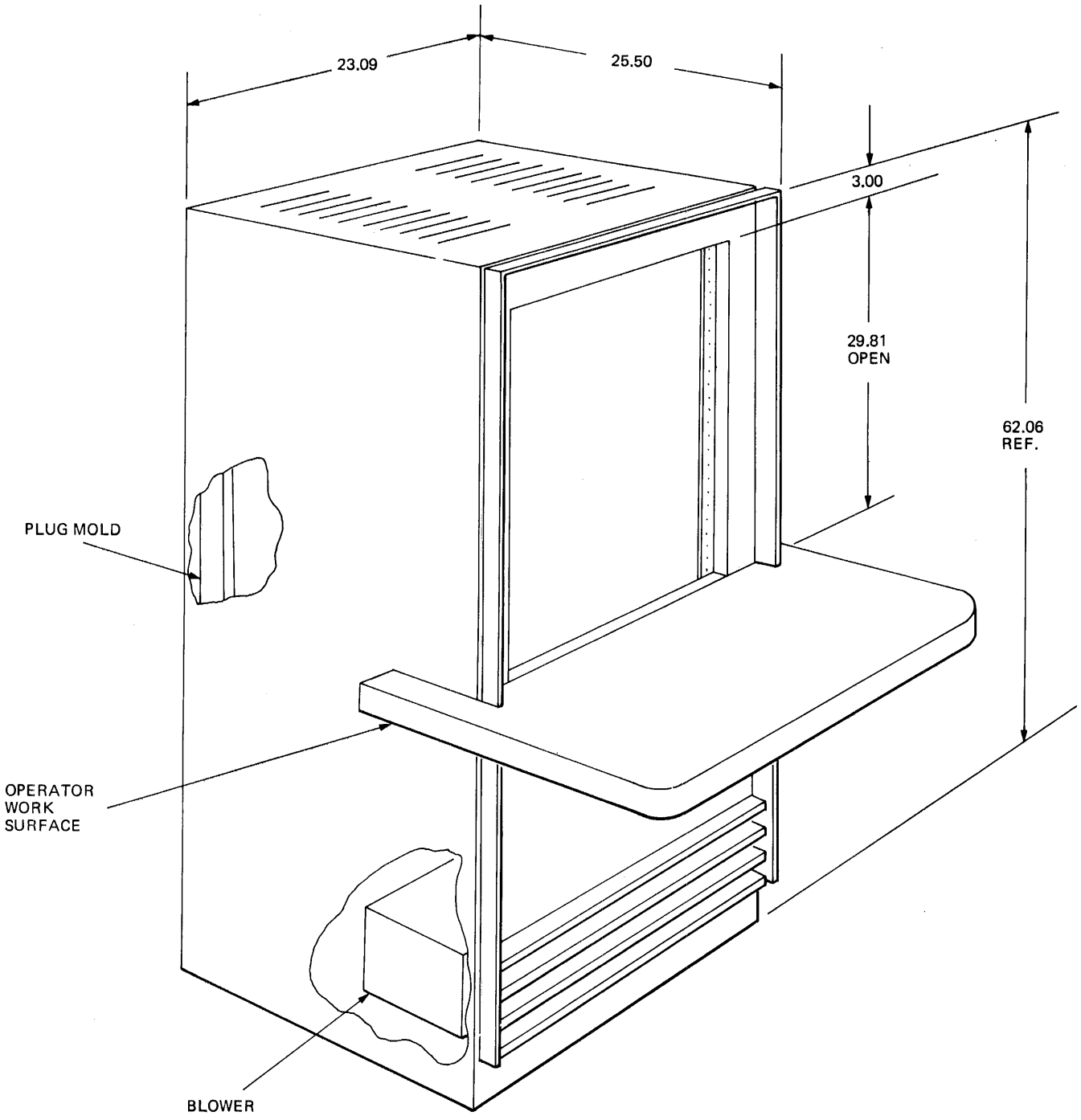


Figure 5-13. Single Bay Mounting Cabinet

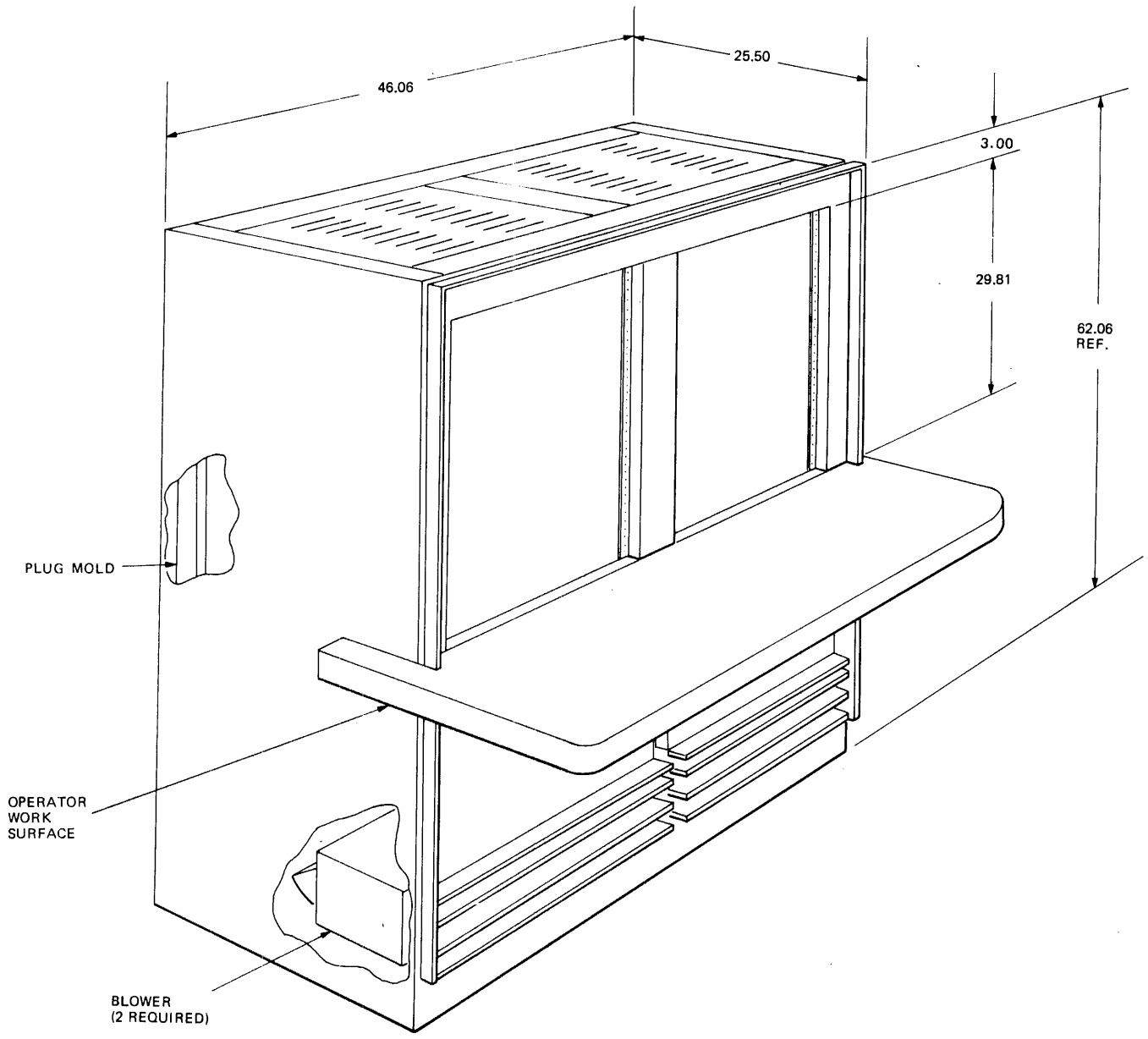


Figure 5-14. Double Bay Mounting Cabinet

APPENDIX A

BACKPLANE CONNECTOR SIGNAL LIST

This appendix contains a signal list of the Micro 1600 backplane connectors which can be used for I/O controllers and the DMA interface. Table A-1 lists the signals on mainframe backplane connectors J4 through J13 (J4 through J17 in the case of the extended backplane). Those pins which carry dc power and the signals used in the I/O bus and the DMA port have the same signals in the dual-CPU Micro 1600D mainframe as those listed in Table A-1. The signal/pin designations in the expansion chassis are identified by a 1 in the SEE NOTE column of Table A-1.

Notes

1. These are the I/O bus signals and the dc power distribution lines, present in both the computer mainframe and the expansion chassis. (In the expansion chassis these are the only pins receiving signals from the mainframe. All other pins are available for interwiring of user-designed circuit boards.)

2. These pins, although not bused together in the expansion chassis backplane, must not be used by the customer at any expansion chassis card slot into which a standard Microdata controller is to be plugged. This is because signals on these lines would be erroneously interpreted by the controllers.

3. These pins are used to assign memory address banks to each of the eight card slots which may be used for core memory (slots J4 through J11). As shown below, the mainframe backplane grounds (GRD) or floats (-) these pins at each of these slots to provide three binary memory select lines.

Memory Select Lines

Card Slot	MS3	MS2	MS1	Selected Memory Bank
J4	--	--	--	0 - 8K
J5	--	--	GRD	8K - 16K
J6	--	GRD	--	16K - 24K
J7	--	GRD	GRD	24K - 32K
J8	GRD	--	--	32K - 40K
J9	GRD	--	GRD	40K - 48K
J10	GRD	GRD	--	48K - 56K
J11	GRD	GRD	GRD	56K - 65K

Table A-1. Backplane Connector Signal List

PIN	SIGNAL	SEE NOTE	PIN	SIGNAL	SEE NOTE
A1	GRD	1	B1	+5vdc	1
A2	GRD	1	B2	+5vdc	1
A3	COXX/	2	B3	-16.75vdc	1
A4	-16.75vdc	1	B4	IACK/	2
A5	+12vdc	1	B5	DIXX/	2
A6	CHP1	1	B6	CACK/	2
A7	COXX/	2	B7	M04A/	
A8	-16.75vdc	1	B8	MPAR/	
A9	N07A/		B9	M06A/	
A10	OD05/	1	B10	OD01/	1
A11	M01A/		B11	A04L/	
A12	M02A/		B12	MS1	3
A13	A00L/		B13	M03A/	
A14	A01L/		B14	M00A/	
A15	NO6A/		B15	RS00	
A16	L00X		B16	MS2	3
A17	L11X		B17	RS04	
A18	L04X		B18	M05A/	
A19	RS01		B19	L10X	
A20	L01X		B20	RTXX/	
A21	WTXX/		B21	RS05	
A22	L05X		B22	CPH2/	1
A23	RS02		B23	READ	
A24	L02X		B24	CGL0/	
A25	L06X		B25	RS06	
A26	OD02/	1	B26	OD06/	1
A27	RS03		B27	MD07	
A28	L03X		B28	MD03	
A29	A02L/		B29	RS07	
A30	L07X		B30	MD05	
A31	IO2X/	1	B31	IO1X/	1
A32	ID04/	1	B32	ID00/	1
A33	CPEN/		B33	L08X	
A34	A03L/		B34	MD00	
A35	MD01		B35	L09X	
A36	MD04		B36	RS08	
A37	RS09		B37	OD04/	1
A38	EINT/	1	B38	A05L/	
A39	RINH/		B39	OD00/	1
A40	CG1B/		B40	MD06	

Table A-1. Backplane Connector Signal List (continued)

PIN	SIGNAL	SEE NOTE	PIN	SIGNAL	SEE NOTE
A41	CSTP/		B41	A06L/	
A42	RS10		B42	EC10/	1
A43	RS11		B43	MD02	
A44	DMAR/		B44	MRST/	1
A45	DMAS/		B45	MS3	3
A46	SP2	1	B46	SP7	1
A47	RS13		B47	M07A/	
A48	RS14		B48	A07L/	
A49	CONT		B49	SPARE/	
A50	N03A/		B50	IRPY/	
A51	RS12		B51	N00A/	
A52	SELO/	1	B52	SELI/	1
A53	N04A/		B53	N01A/	
A54	N05A/		B54	PRIN/	1
A55	PROT/	1	B55	N02A/	
A56	DMAW/		B56	MBDY/	
A57	MBSY		B57	MPR0/	
A58	OD07/	1	B58	OD03/	1
A59	RS15		B59	ID05/	1
A60	ID01/	1	B60	ID07/	1
A61	ID06/	1	B61	IO3X/	1
A62	ID03/	1	B62	ID02/	1
A63	-16.75vdc	1	B63	-16.75vdc	1
A64	GRD	1	B64	+5vdc	1
A65	GRD	1	B65	+5vdc	1

APPENDIX B

I/O INTERFACE SIGNAL GLOSSARY

This appendix is a glossary of the signals used to interface byte I/O controllers and DMA Port interfaces to the Micro 1600 computer. The list is arranged alphabetically by signal mnemonic. The origin for each signal (computer or controller), its availability (mainframe and/or expansion backplanes), the connector pin number, and its function are provided in Table B-1.

Note

A slash (/) at the end of a signal or line mnemonic denotes that the line is low when the function specified by the mnemonic is occurring.

Table B-1. I/O Interface Signal Glossary

SIGNAL MNEMONIC	AVAILABILITY MAIN CHASSIS	AVAILABILITY EXP. CHASSIS	PIN NO.	FUNCTION	ORIGIN CPU	ORIGIN CONT
CPH1	X	X	A6	Processor Clock. 5.0 MHz square wave	X	
CPH2/	X	X	B22	Processor Clock. Inverted version of CPH1, delayed 33ns.	X	
DMAR/	X		A44	DMA Request. DMAR/ initiates a memory cycle for DMA.		X
DMAS/	X		A45	DMA Select. DMAS/ selects memory for a DMA operation by inhibiting CPU access to memory.		X
DMAW/	X		A56	DMA Write. Causes the data byte on memory data lines MD00 through MD07 to be written into memory.		X
ECIO/	X	X	B42	Concurrent I/O Request. Low signal from I/O device requesting a concurrent I/O transfer. ECIO/ appears in CPU as bit 3 of File Register 0 where it acts as an interrupt to the macroprogram and initiates a firmware routine for handling a concurrent transfer.		X

Table B-1. I/O Interface Signal Glossary (continued)

SIGNAL MNEMONIC	AVAILABILITY MAIN CHASSIS	AVAILABILITY EXP. CHASSIS	PIN NO.	FUNCTION	ORIGIN CPU	ORIGIN CONT
EINT/	X	X	A38	External Interrupt. Low signal from I/O device requesting interruption of the macro-program. EINT/ appears in CPU as bit 7 of File Register 0 where it initiates a firmware routine for transferring control to a macro-program interrupt handling routine.		X
ID00/	X	X	B32	Data Input Bit 0. Connects to CPU via B Bus gating.		X
ID01/	X	X	A60	Data Input Bit 1. Connects to CPU via B Bus gating.		X
ID02/	X	X	B62	Data Input Bit 2. Connects to CPU via B Bus gating.		X
ID03/	X	X	A62	Data Input Bit 3. Connects to CPU via B Bus gating.		X
ID04/	X	X	A32	Data Input Bit 4. Connects to CPU via B Bus gating.		X
ID05/	X	X	B59	Data Input Bit 5. Connects to CPU via B Bus gating.		X
ID06/	X	X	A61	Data Input Bit 6. Connects to CPU via B Bus gating.		X
ID07/	X	X	B60	Data Input Bit 7. Connects to CPU via B Bus gating.		X
I01X/	X	X	B31	Bit 1 of I/O Control Register	X	
I02X/	X	X	A31	Bit 2 of I/O Control Register	X	
I03X/	X	X	B61	Bit 3 of I/O Control Register	X	

Table B-1. I/O Interface Signal Glossary (continued)

SIGNAL MNEMONIC	AVAILABILITY MAIN CHASSIS	AVAILABILITY EXP. CHASSIS	PIN NO.	FUNCTION	ORIGIN CPU	ORIGIN CONT
MBSY	X		A57	Memory Busy. MBSY is a status signal from memory indicating that memory is busy.	X	
MD00	X		B34	Bidirectional data line (bit 0) to memory for DMA operation	X	X
MD01	X		A35	Bidirectional data line (bit 1) to memory for DMA operation	X	X
MD02	X		B43	Bidirectional data line (bit 2) to memory for DMA operation	X	X
MD03	X		B28	Bidirectional data line (bit 3) to memory for DMA operation	X	X
MD04	X		A36	Bidirectional data line (bit 4) to memory for DMA operation	X	X
MD05	X		B30	Bidirectional data line (bit 5) to memory for DMA operation	X	X
MD06	X		B40	Bidirectional data line (bit 6) to memory for DMA operation	X	X
MD07	X		B27	Bidirectional data line (bit 7) to memory for DMA operation	X	X
M00A/	X		B14	Bit 0 of upper half of memory address (used by DMA)		X
M01A/	X		A11	Bit 1 of upper half of memory address (used by DMA)		X
M02A/	X		A12	Bit 2 of upper half of memory address (used by DMA)		X
M03A/	X		B13	Bit 3 of upper half of memory address (used by DMA)		X

Table B-1. I/O Interface Signal Glossary (continued)

SIGNAL MNEMONIC	AVAILABILITY MAIN CHASSIS	AVAILABILITY EXP. CHASSIS	PIN NO.	FUNCTION	ORIGIN CPU	ORIGIN CONT
M04A/	X		B7	Bit 4 of upper half of memory address (used by DMA)		X
M05A/	X		B18	Bit 5 of upper half of memory address (used by DMA)		X
M06A/	X		B9	Bit 6 of upper half of memory address (used by DMA)		X
M07A/	X		B47	Bit 7 of upper half of memory address (used by DMA)		X
N00A	X		B51	Bit 0 of lower half of memory address (used by DMA)		X
N01A/	X		B53	Bit 1 of lower half of memory address (used by DMA)		X
N02A/	X		B55	Bit 2 of lower half of memory address (used by DMA)		X
N03A/	X		A50	Bit 3 of lower half of memory address (used by DMA)		X
N04A/	X		A53	Bit 4 of lower half of memory address (used by DMA)		X
N05A/	X		A54	Bit 5 of lower half of memory address (used by DMA)		X
N06A/	X		A15	Bit 6 of lower half of memory address (used by DMA)		X
N07A/	X		A9	Bit 7 of lower half of memory address (used by DMA)		X

Table B-1. I/O Interface Signal Glossary (continued)

MNEMONIC	AVAILABILITY MAIN CHASSIS	AVAILABILITY EXP. CHASSIS	PIN NO.	FUNCTION	ORIGIN CPU	ORIGIN CONT
MRST/	X	X	B44	Master Reset. Signal used to clear all control flip-flops in controllers.	X	
OD00/	X	X	B39	Output Data Bit 0	X	
OD01/	X	X	B10	Output Data Bit 1	X	
OD02/	X	X	A26	Output Data Bit 2	X	
OD03/	X	X	B58	Output Data Bit 3	X	
OD04/	X	X	B37	Output Data Bit 4	X	
OD05/	X	X	A10	Output Data Bit 5	X	
OD06/	X	X	B26	Output Data Bit 6	X	
OD07/	X	X	A58	Output Data Bit 7	X	
PRIN/	X	X	B54	Priority In. Low signal from preceding controller indicating I/O controller has priority to request interrupt operation. PRIN/ is passed serially from CPU to controller to controller, etc.	X	X
PROT/	X	X	A55	Priority Out. Low signal originating in CPU and passed from controller to controller carrying interrupt request priority. PROT/ becomes PRIN/ on input to each controller.	X	X

Table B-1. I/O Interface Signal Glossary (continued)

SIGNAL MNEMONIC	AVAILABILITY MAIN CHASSIS	AVAILABILITY EXP. CHASSIS	PIN NO.	FUNCTION	ORIGIN CPU	ORIGIN CONT
SELI/	X	X	B52	Select In. A low signal from preceding controller which occurs after an interrupt or concurrent I/O request during acknowledge time. SELI/ indicates controller has priority to place its address on data lines for an interrupt or concurrent I/O operation. SELI/ is passed serially from CPU to controller to controller, etc.	X	X
SELO/	X	X	A52	Select Out. Low signal originating in CPU and passed from controller to controller which passes select priority from controller to controller. SELO/ becomes SELI/ at input to each controller.	X	X
SPARE/	X		B49	Spare. Available for use by DMA controller or other customer-designed option as an internal CPU interrupt. Enters CPU at internal status register bit 1 where it is OR'ed with other internal status bits, and ultimately sets bit 4 of file register 0.		X

APPENDIX C

TELETYPE MODIFICATION PROCEDURES

A standard ASR 33 TY Teletype can be modified for compatibility with Micro 1600 series computers using the following procedure (changes from 60-ma to 20-ma current loop):

Note

Teletype Corporation wiring diagrams 6353 WD (sheet 1) or 7833 WD (sheet 1) are helpful, but not mandatory, for the following changes.

a. Remove cover of teletype

b. Move purple wire from terminal 8 to terminal 9 of terminal strip 151411 located at lower left of Teletype (veiwed from rear).

c. Move white-blue wire from terminal 4 to terminal 5 of the same terminal strip.

d. Move brown-yellow wire from terminal 3 to terminal 5 of the same terminal strip.

e. Move blue wire from terminal 3 to terminal 4 of power resistor 181816. This resistor is located at the center of the right side of the teletype (viewed from the front with the cover removed). When the cover is in place, the resistor can be seen centered under the removable plate on the right side.

APPENDIX D

BYTE I/O CONTROLLER DESIGN NOTES

D.1 INTRODUCTION

This appendix presents an example design of a general I/O controller for Microdata 1600 computers utilizing the byte, concurrent, and interrupt I/O facilities. This example is intended to illustrate design conventions and procedures, and is not intended to imply that other approaches cannot be used. For example; the circuitry shown in portions of Section 2 follows another approach.

The example controller illustrates a design for basic data transfers, concurrent input/output, and interrupts and includes the decoding required for channel control logic for operation on the byte I/O channel.

D.2 FUNCTIONAL DESCRIPTION

The following paragraphs functionally describe the example controller. Accompanying logic and control sequence diagrams are given in Figures D-1 through D-13. Basic I/O timing diagrams for status input, byte output, and concurrent input/output data transfers are included for reference in Figures D-14 through D-16. Table D-2 at the end of the appendix contains a glossary of signals for the byte I/O controller.

D.2.1 OUTPUT DATA BUS RECEIVERS (FIGURE D-1)

The output data bus (OD00/ through OD07/) is received at only one point in the controller. This limits the loading on the output data bus to one unit load for each device. Information transferred via this bus (at unique times) consists of data, functions, device addresses and device orders.

D.2.2 FREE-RUNNING CLOCK GENERATOR (FIGURE D-1)

Lines CPH1 and CPH2/ on the backplane are used to provide synchronization and timing signals within the controller. CPH1 and CPH2/ are gated together with COXX/ to yield a 33-nanosecond pulse occurring at 200-nanosecond intervals. (COXX/ is decoded from IO1X through IO3X/.)

D.2.3 MASTER RESET (FIGURE D-1)

The general reset signal (MRST/) is generated by the RESET switch on the processor control panel or when power is applied or removed from the CPU. When this signal is enabled, peripheral devices should clear all control elements.

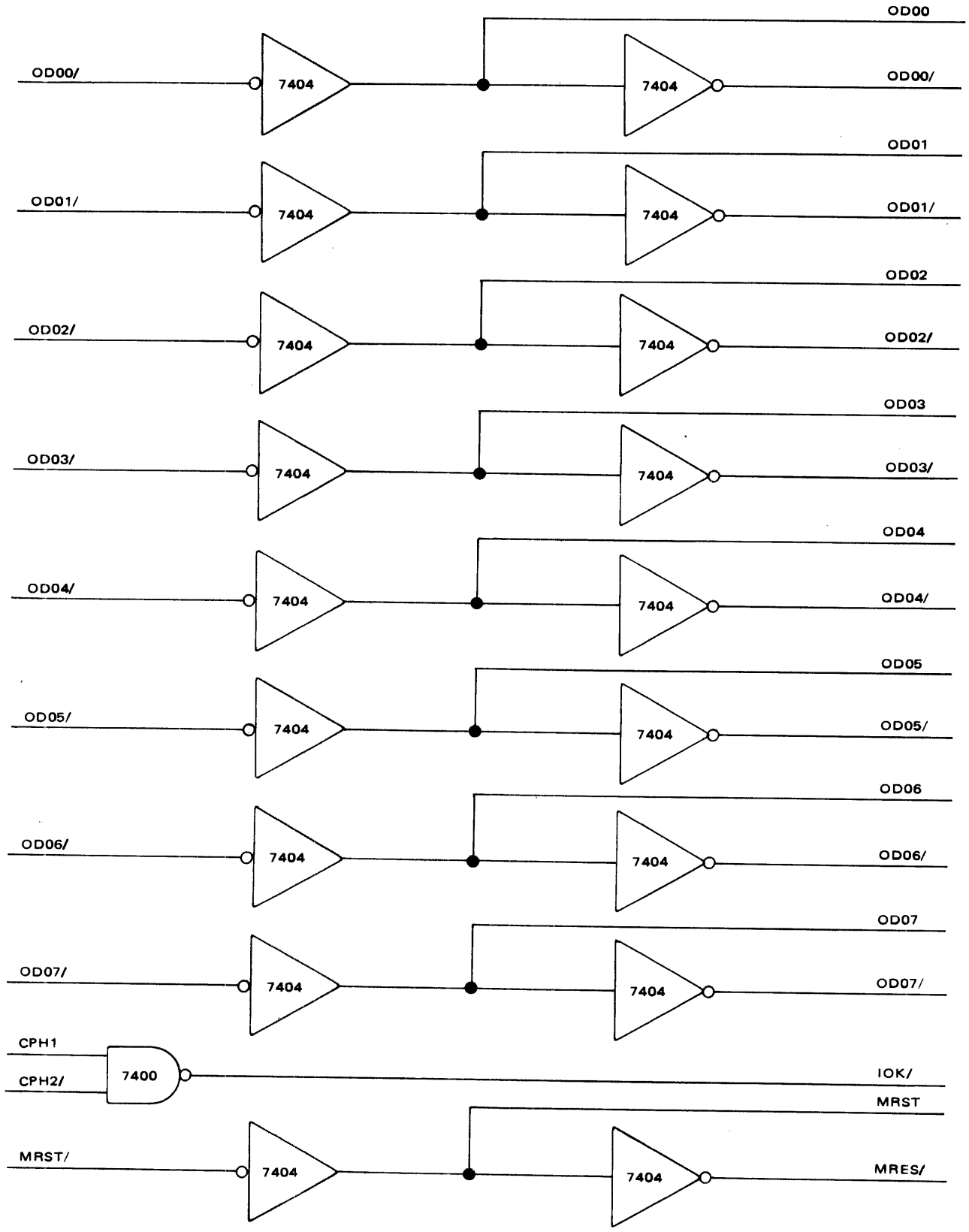


Figure D-1. Output Bus Receivers, Free-Running Clock Generator, and Master Reset Logic

D.2.4 CONTROL DECODER (FIGURE D-2)

Outputs of the I/O Control Register (IO1X/, IO2X/, and IO3X/) in the CPU are received at only one point in the interface. These signals are then reinverted and used as the three least significant bit inputs to a one-of-ten decoder. Switching the most significant bit input to ground when any of these three lines are enabled allows this element to be mechanized as an octal decoder with an enabling input. The most significant bit is enabled through a latch. IO1X/ through IO3X/ occur at CPH1 time and are applied as inputs to the decoder. In addition, the OR of IO1X/, IO2X/, and IO3X/ is AND'ed with CPH2/ and used as the set term to the latch providing a 33-nanosecond enable delay. The latch is cleared on the trailing edge of IO1X/, IO2X/, or IO3X/.

D.2.5 DEVICE ORDER DECODER (FIGURE D-3)

The three most significant bits of the second byte of an input or output byte instruction are used for controller control functions (device orders). Bits 5, 6, and 7 are decoded in an octal decoder. The enabling signal for this decoder is an AND function of the device address (least significant five bits of this byte) and control signal COXX/. The standard device order codes are listed in Table 3-3 in Section 3 of this manual. Although the coding shown in Table 3-3 is applicable to most peripheral devices, certain interface designs may require encoding unique to that unit.

D.2.6 GATED CLOCK GENERATOR (FIGURE D-3)

For simplicity of operation, most of the stored control information in the I/O controller unit can be clocked with the decoded control signals. This allows the control flip-flops to switch at the trailing edge of the control pulses. This clock (KIXX) is generated by OR'ing COXX/, DOXX/, DIXX/, and ANXX/.

D.2.7 CONTROL STORAGE (FIGURE D-4)

When a function requires use of the second byte transferred by an I/O instruction, or specifies an operation that will continue until disconnected by a later event, control information decoded from the device order at COXX/ time from the second byte of the input/output command must be stored. The functions that require storage (from the standard function code table) are:

Flip-Flop	Description
DATA	Data transfer control flip-flop. Set by a data transfer device order code or at the time that a concurrent request has been honored. The flip-flop is reset by KIXX at the conclusion of the following DIXX/ or DOXX/ time.

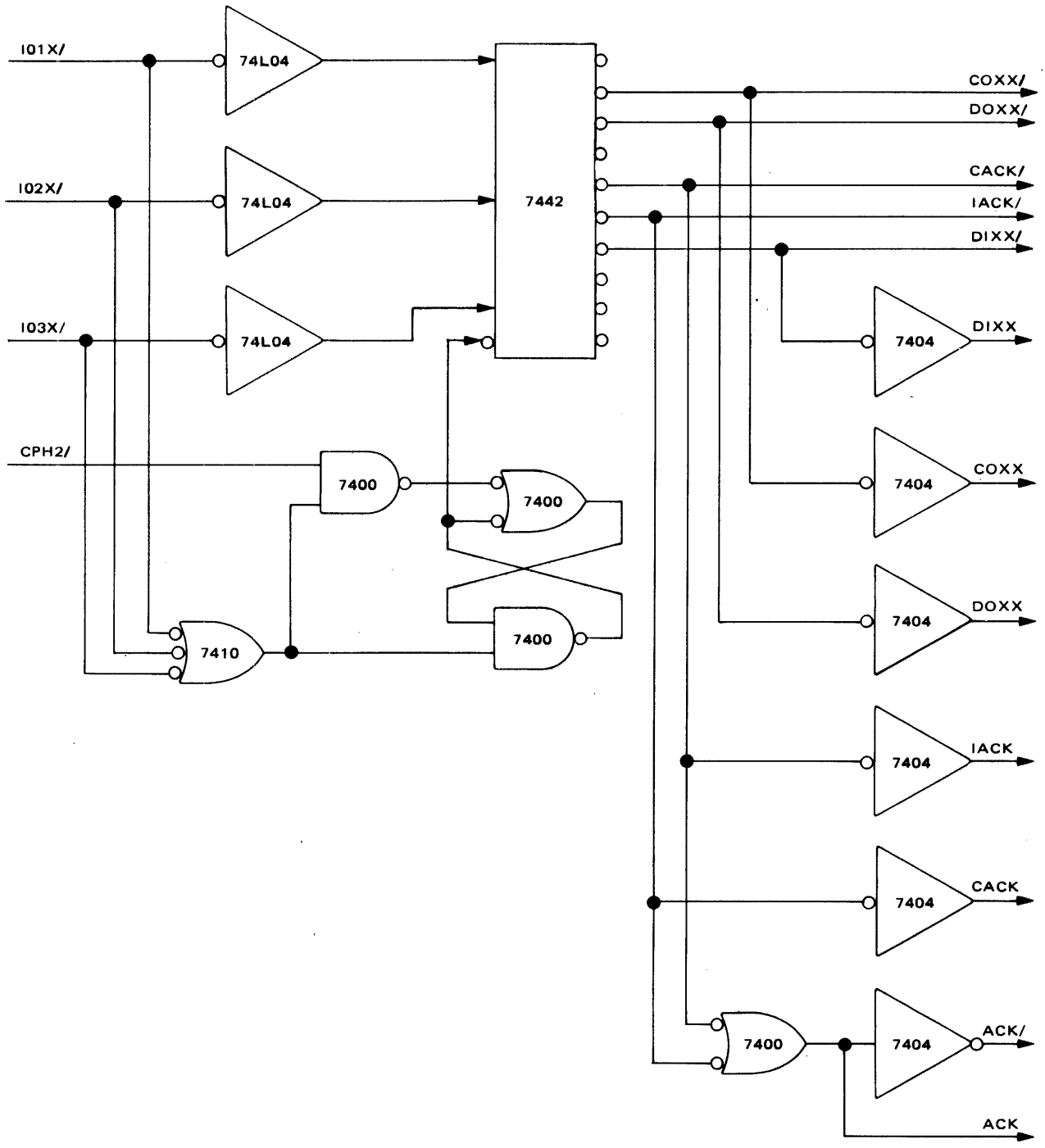


Figure D-2. Control Decoder Logic

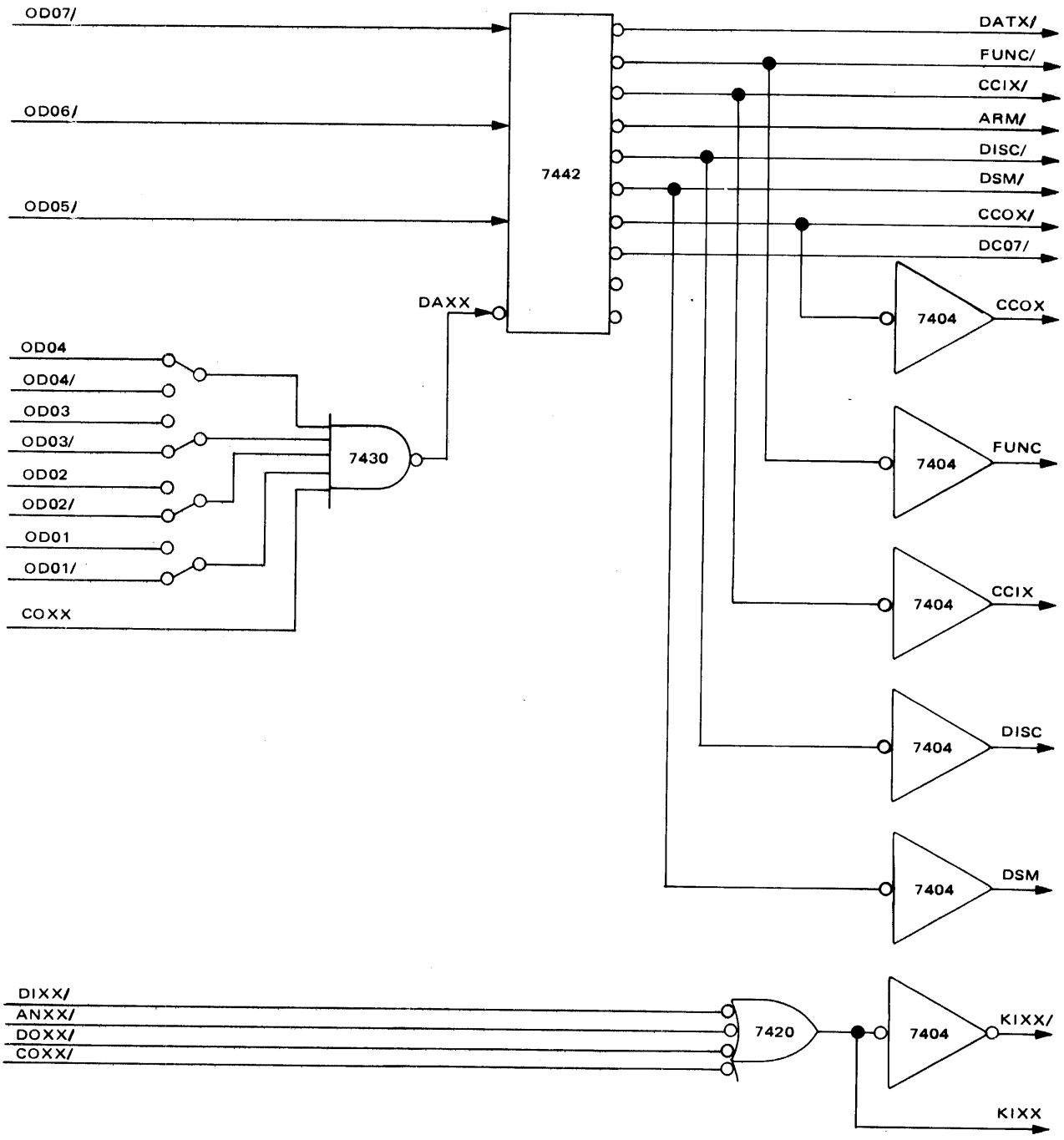


Figure D-3. Function Decoder and Clock Generator Logic

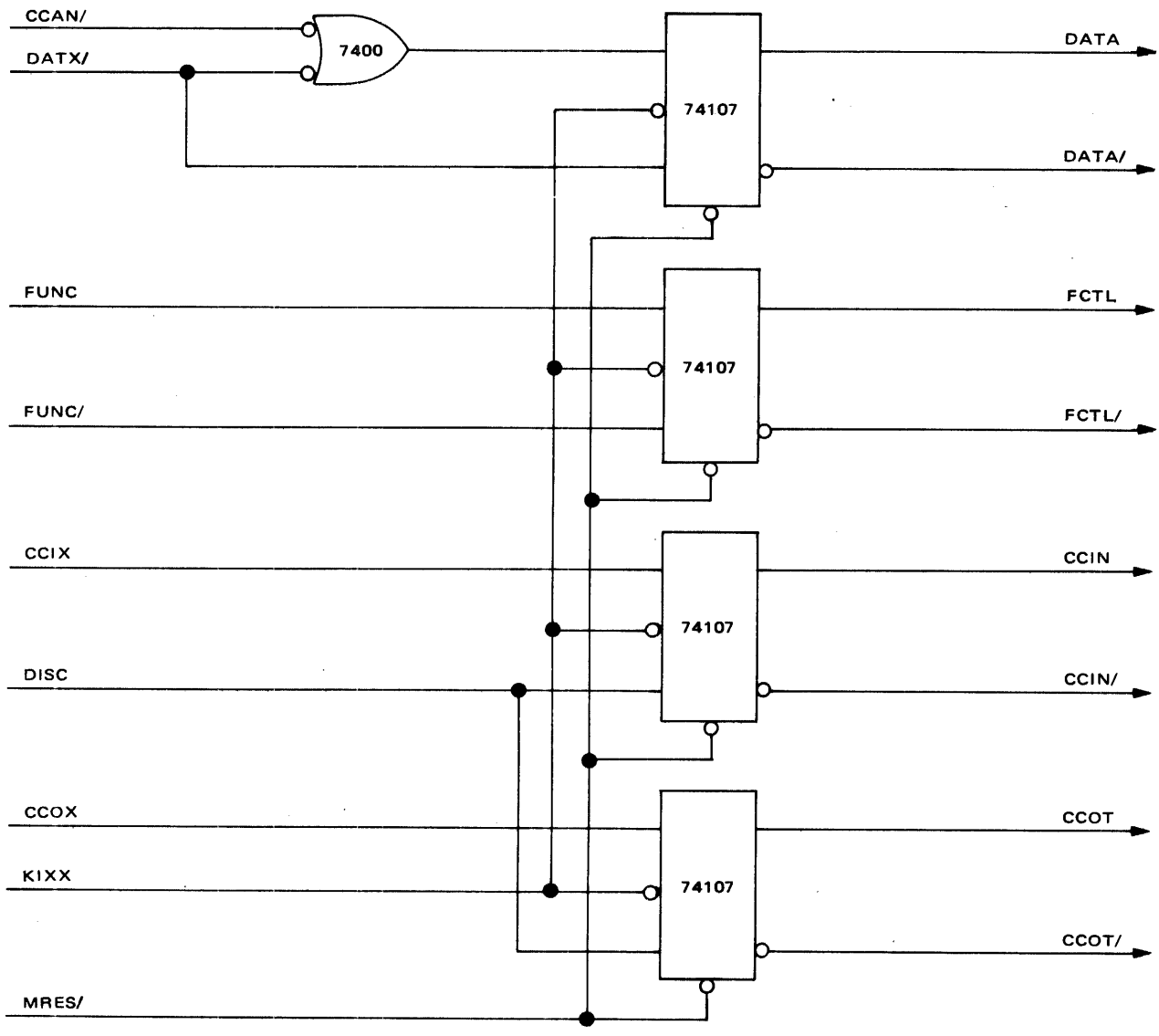


Figure D-4. Function Storage Logic

Flip-Flop	Description
FCTL	Function/status control flip-flops. Set by a function/status device order code when a function output or status input operation is required. The flip-flop is reset by KIXX at the conclusion of the following DIXX/ or DOXX/ time (See Figure D-14).
CCIN, CCOT	Mode storage for a concurrent input or concurrent output operation. Flip-flops are set by a concurrent input or output device order code and reset by a disconnect device order code. The disconnect device order code will be transmitted automatically by the concurrent firmware at the conclusion of the transfer of the last byte in the specified block or by an output instruction with that device order code.

D.2.8 INPUT DATA BUS (FIGURE D-5)

Because the input data bus may be used by several controllers (not simultaneously), gated open-collector drivers must be used to drive the bus. In this example, enabling signal IDEN is made up as follows: When controller function/status flip-flop FCTL or DATA flip-flop is true, and the control strobe DIXX is present, or when address enable control signal ANXX is true, the bus drivers are enabled.

D.2.9 INPUT DATA SWITCH (FIGURE D-6)

In this example, information to be placed on the input data bus comes from one of three sources; data from the peripheral device, device status, or the address for concurrent or interrupt identification. Data is switched by dual four-input multiplexers. Address assignment for switching is as follows:

Address	Assignment
0	Data
1	Status
2	Address
3	Not Used

Note

In place of the multiplexers, the user can wire-OR type 7438 IC chips to perform the data switching function. In some cases, this may prove to be a lower cost technique.

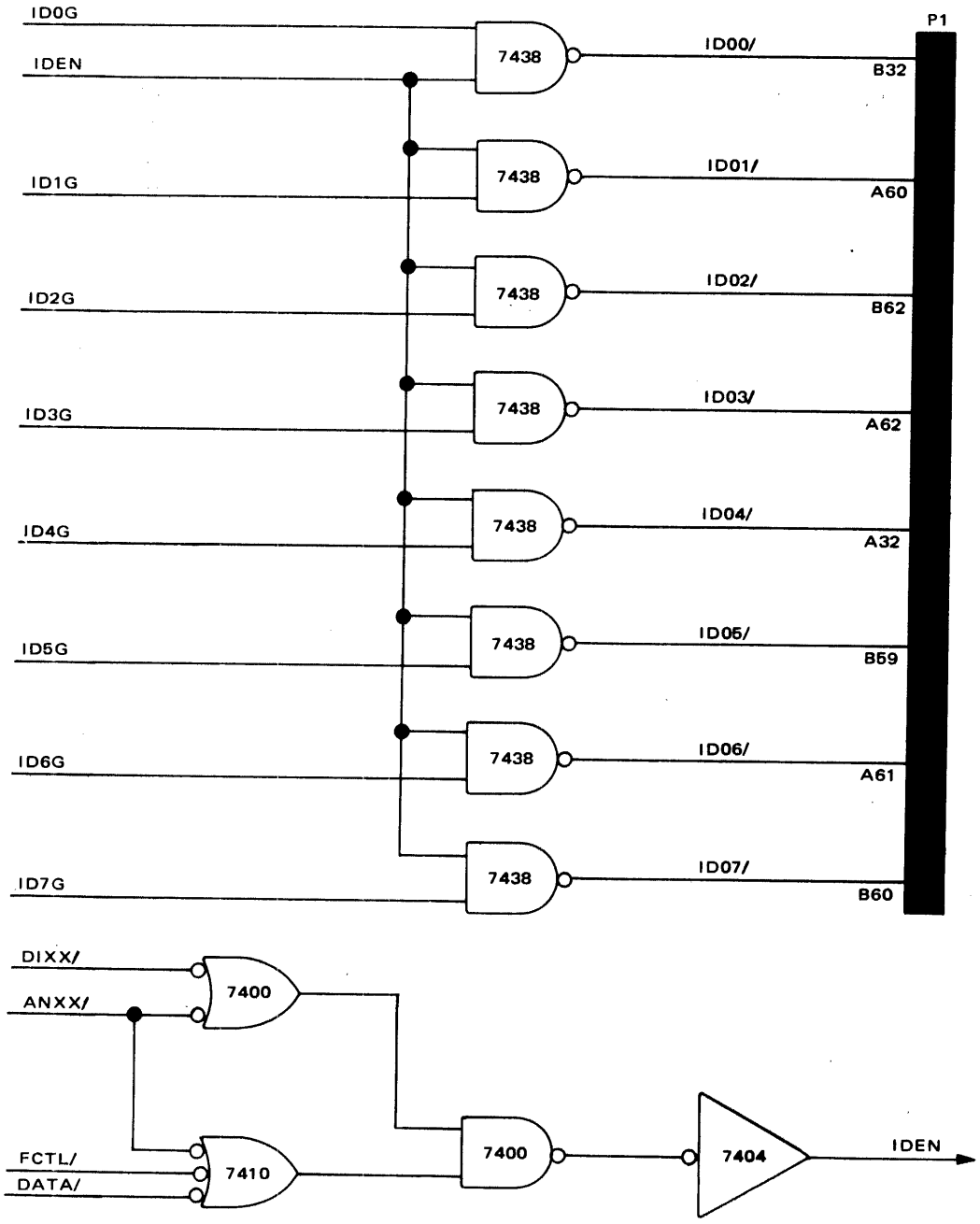


Figure D-5 Input Data Bus Logic

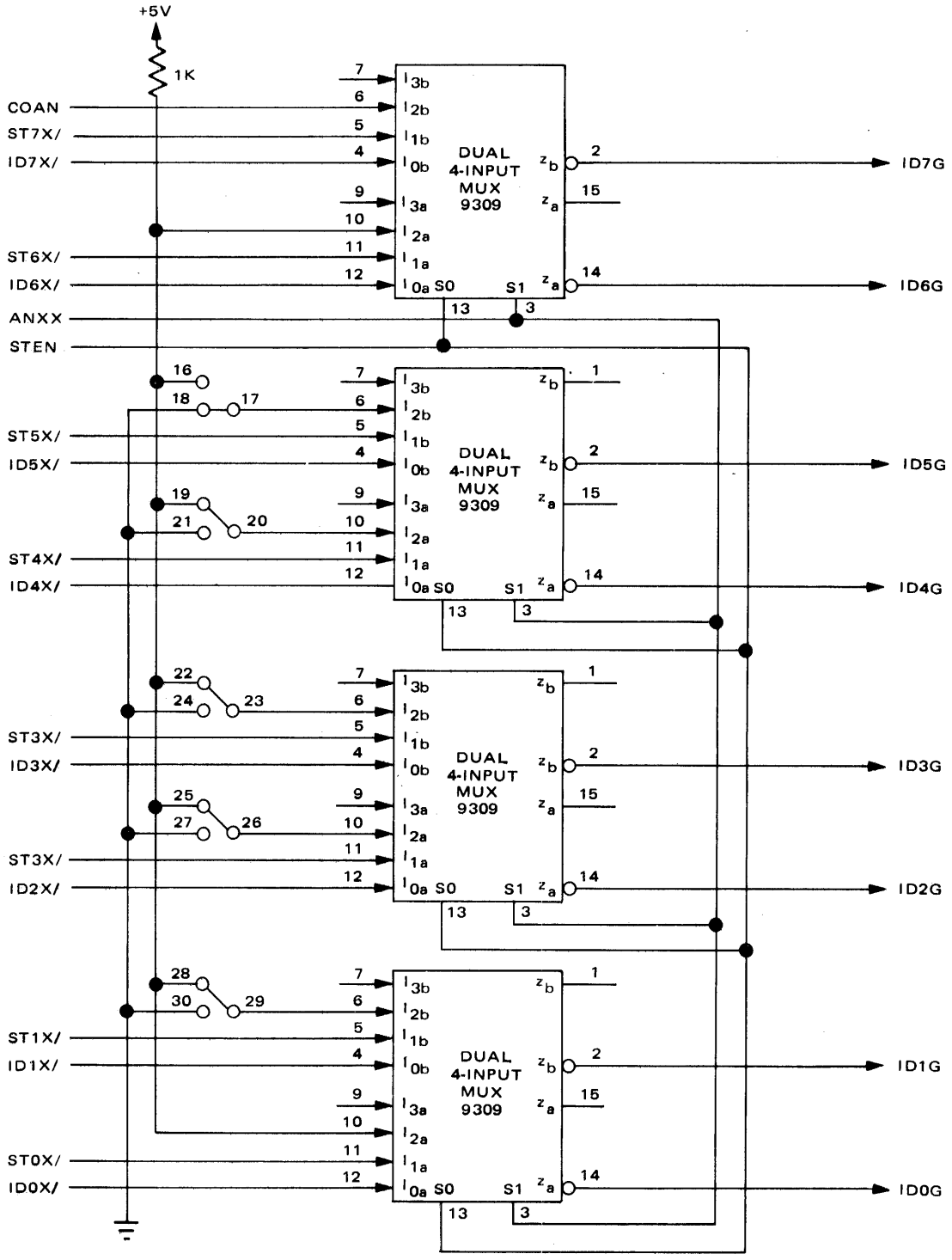


Figure D-6. Input Data Switch Logic

D.2.10 CONTROLLER CONTROL (FIGURE D-7)

Select Latch - The processor responds to an external interrupt (EINT) or concurrent I/O request (ECIO) by generating an interrupt acknowledge (IACK) or concurrent I/O acknowledge (CACK). These signals are generated by setting the I/O control register to state 5 or 4 respectively. The interrupting controller with the highest priority responds during the acknowledge period by placing its address on the input data bus (ANXX). Priority for this operation is controlled by select line SELI. Peripheral controllers respond to the acknowledge and select signals as follows: The controller located in the highest priority slot receives the concurrent or interrupt acknowledge and, if not making the appropriate request at the time select in goes true, enables select out (SELO/) and latches to that condition. This condition is then propagated to the next controller in priority. This latch remains on until acknowledge disappears, at which time all devices unlatch simultaneously. When select in is received by a controller and is making the appropriate request, the select out line will not be enabled. Instead, answer (ANXX) is turned on during the acknowledge period. At this time the device presents its address to the processor on the input data bus.

Because the controller must be capable of propagating select out for interrupt and concurrent acknowledge pulses, systems with only an interrupt system should implement their latch as shown in Figure D-8.

Concurrent Address Enable - Bits 1 through 5 are used to transmit the address placed on the input data bus during the acknowledge period. However, bit 7 (most significant bit) is used as a direction tag for concurrent addressing. Bit 7 is a one if a concurrent output is requested and a zero if an input is requested. Concurrent address enable (CCAN/) is generated to enable this bit only during the period the concurrent address is on the bus.

Concurrent Request - Any controller, regardless of its priority in the system, can make a concurrent I/O request (ECIO) whenever it requires a data transfer. No priority qualifications are needed for the request because the processor completes the concurrent operation in firmware without testing for an external interrupt or concurrent request.

Interrupt Priority Line - The interrupt priority line is used to control which device can make an external interrupt request at any given time. Propagation of the priority line occurs only if priority has been received and the interrupt system is not requesting or in the active state (see Figure D-9).

External Interrupt Request - A device can make an external interrupt request (EINT) only if it has priority and is in the wait state (see Figure D-9).

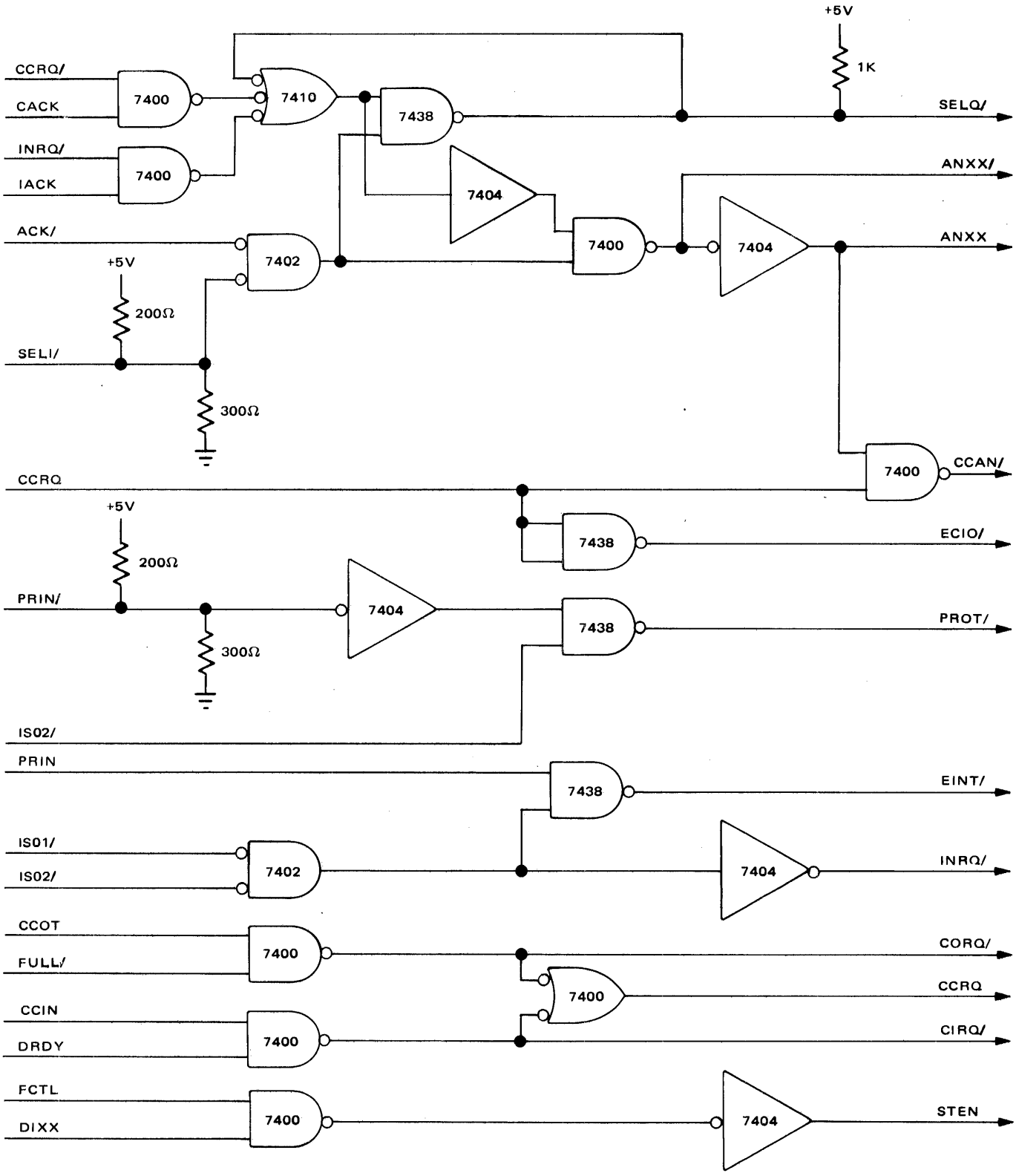


Figure D-7. Controller Control Logic

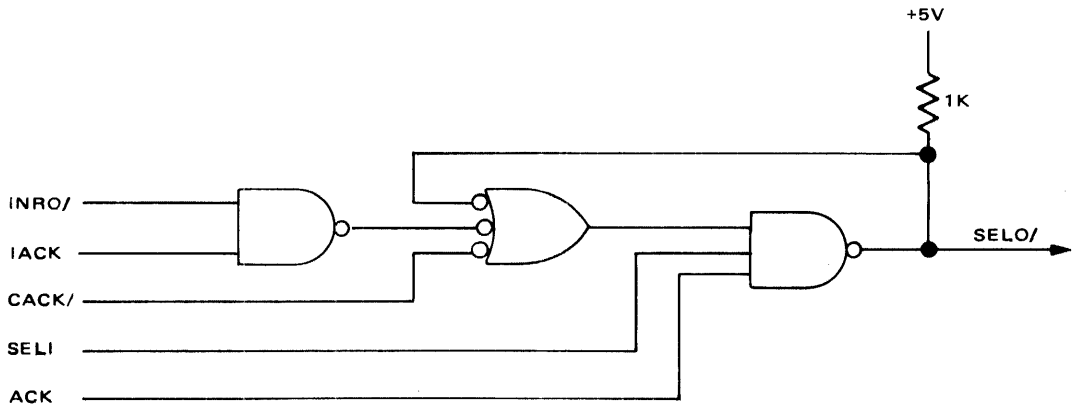


Figure D-8. Select Latch For Interrupt System Only

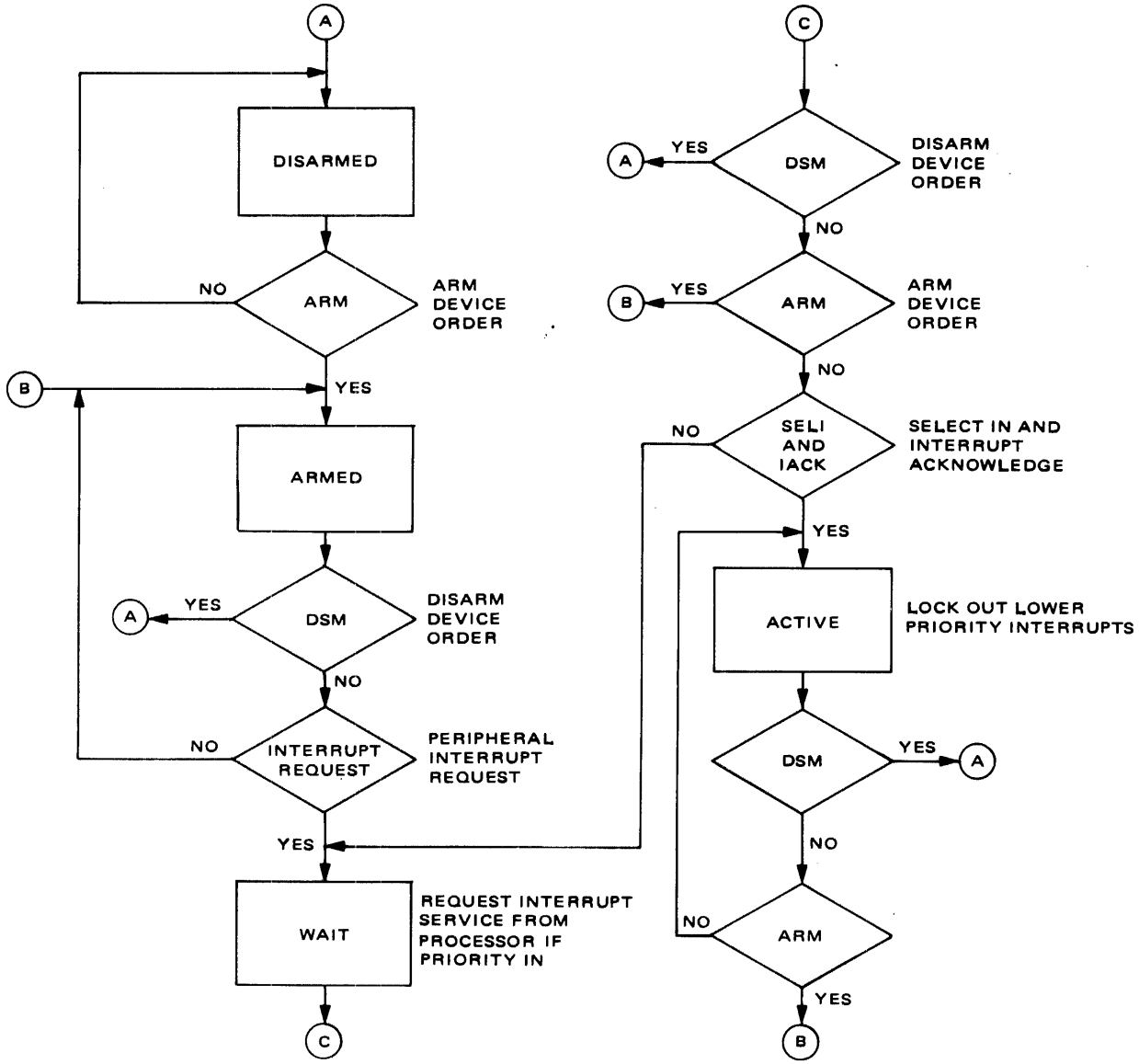


Figure D-9. Interrupt Sequence Flow Diagram

D.2.11 INTERRUPT SEQUENCER (FIGURE D-10)

The interrupt sequencer in the controller contains two JK flip-flops (and associated circuits) which generate the interrupt request (EINT/) and control the priority line (PROT/) to the next controller. The four states of the flip-flops determine the priority interrupt status of the controller. Figure D-9 shows the operating sequence of the interrupt sequencer. The four states are illustrated in Figure D-11 and described in Table D-1.

When the controller is initialized, the sequencer is set to state 0 (disarmed). When disarmed, the controller cannot generate an interrupt request and always passes PROT/ to the next controller.

In order to allow an interrupt request, the program must execute an instruction to arm the controller interrupt, setting the sequencer to state 1 (armed). In this state the controller can generate EINT/ providing that it has priority from the preceding controller on the priority chain.

When the controller is ready to generate an interrupt request to the CPU, the sequencer advances to state 2 (wait), the priority line (PROT/) is removed from the next controllers, and interrupt request EINT/ is generated. When the firmware responds to the interrupt request with acknowledge (IACK/), the controller (if it has SELI/) places its address on the input data lines and the sequencer moves to state 3 (active) and removes EINT/.

While the sequencer is in the active state, PROT/ is not passed to the next controller. This prevents a lower priority controller from making an interrupt request while the interrupt handling subroutine is in process. One of the functions of the interrupt subroutine is to execute an instruction to rearm the controller is another interrupt request is expected, or to disarm the controller if no further interrupts are desired. The rearming or disarming normally takes place near the end of the subroutine and restores priority (PROT/) to the lower priority controllers.

Concurrent I/O operations are normally terminated with an end-of-operation interrupt to inform the CPU that the block of data has been transferred. The concurrent I/O request automatically arms the controller.

D.2.12 INTERFACE CONTROL STORAGE (FIGURE D-12)

One means of setting up mode or sequence control of the peripheral controller is shown in this example. The eight interface control flip-flops (IC01 through IC07), are set or reset by the data pattern in the second byte of an output function command.

Note

The flip-flops are clocked by control signal KIXX so setting and resetting occurs on the trailing edge of DIXX/, DOXX/, and ANXX/.

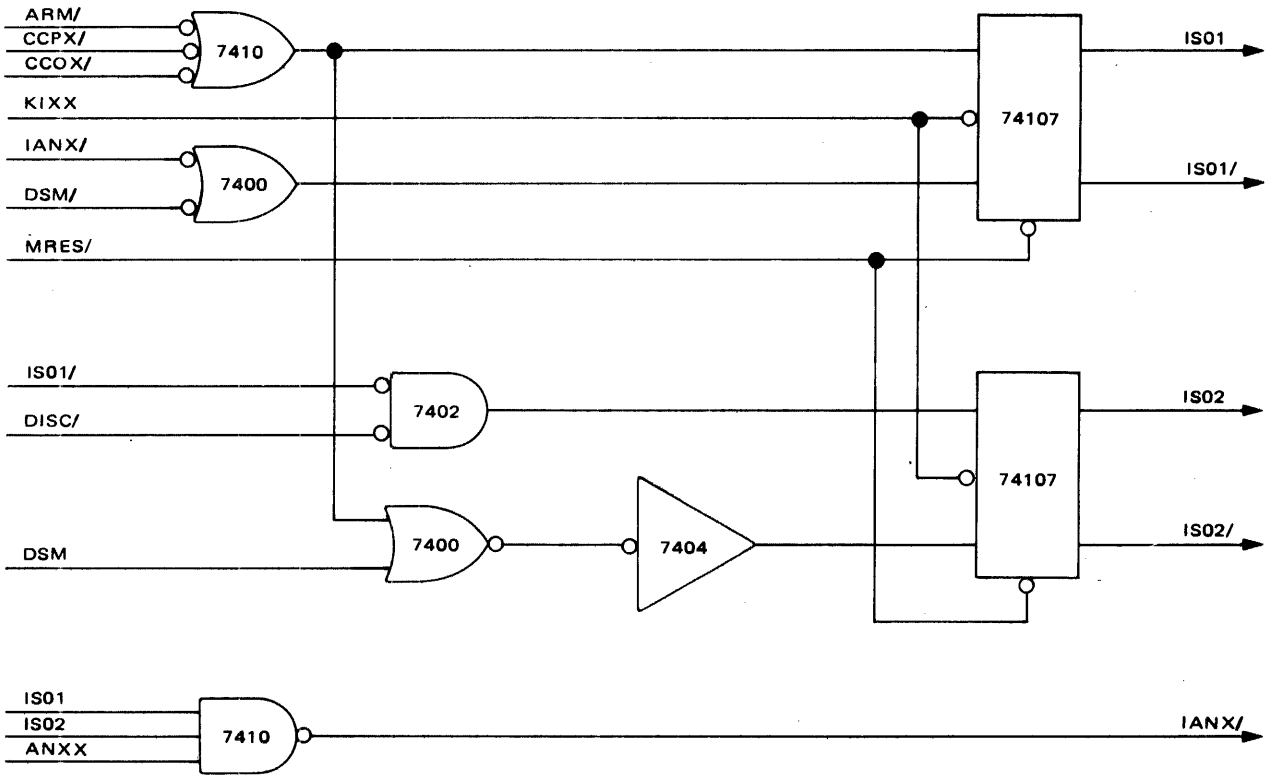


Figure D-10. Interrupt Sequencer Logic

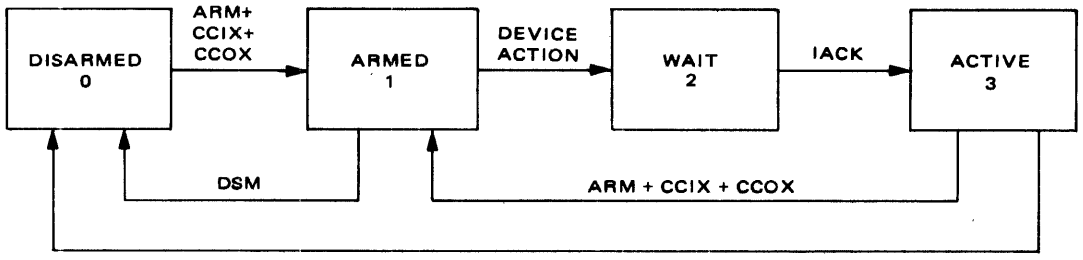


Figure D-11. Interrupt Seqeuncer States

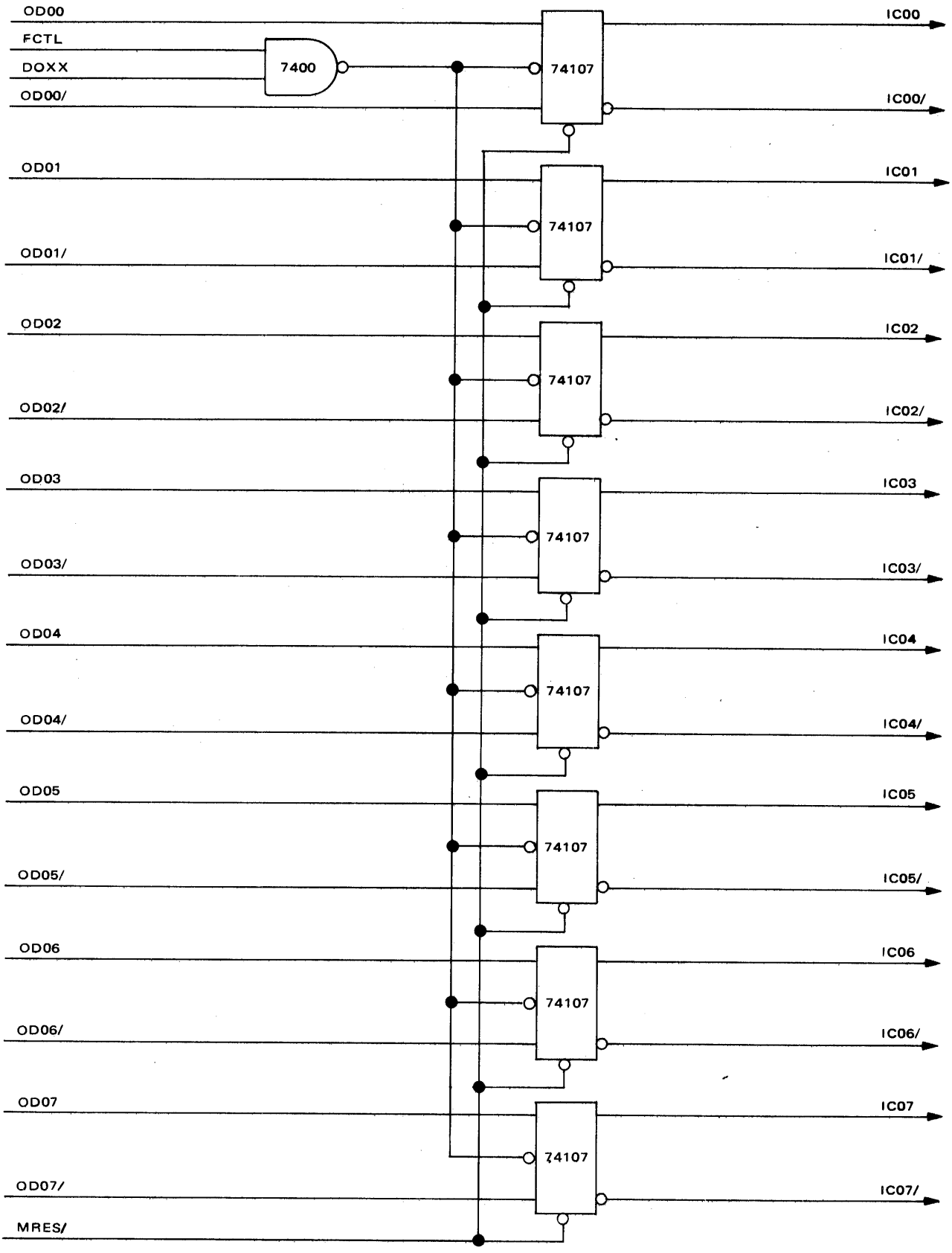


Figure D-12. Interface Control Storage Logic

Table D-1. Interrupt Sequencer States

STATE	FLIP-FLOP STATES		DESCRIPTION
	IS02	IS01	
0	0	0	Disarmed state. Disregards any received interrupt and does not move to requesting state.
1	0	1	Armed state. Allows system to move to requesting state when controller conditions are met.
2	1	1	Wait state. Generates an external interrupt to the processor when priority in is received.
3	1	0	Active state. Inhibits propagation of priority to lower level priority controllers.

D.2.13 INTERRUPT AND CONCURRENT I/O ACKNOWLEDGE SEQUENCE (FIGURE D-13)

The general flow diagram to handle the acknowledge sequence for both interrupts and concurrent I/O is shown in Figure D-13. The control signals generated from this sequence are used to perform the indicated operations. Logic implementation for this sequencer is not shown.

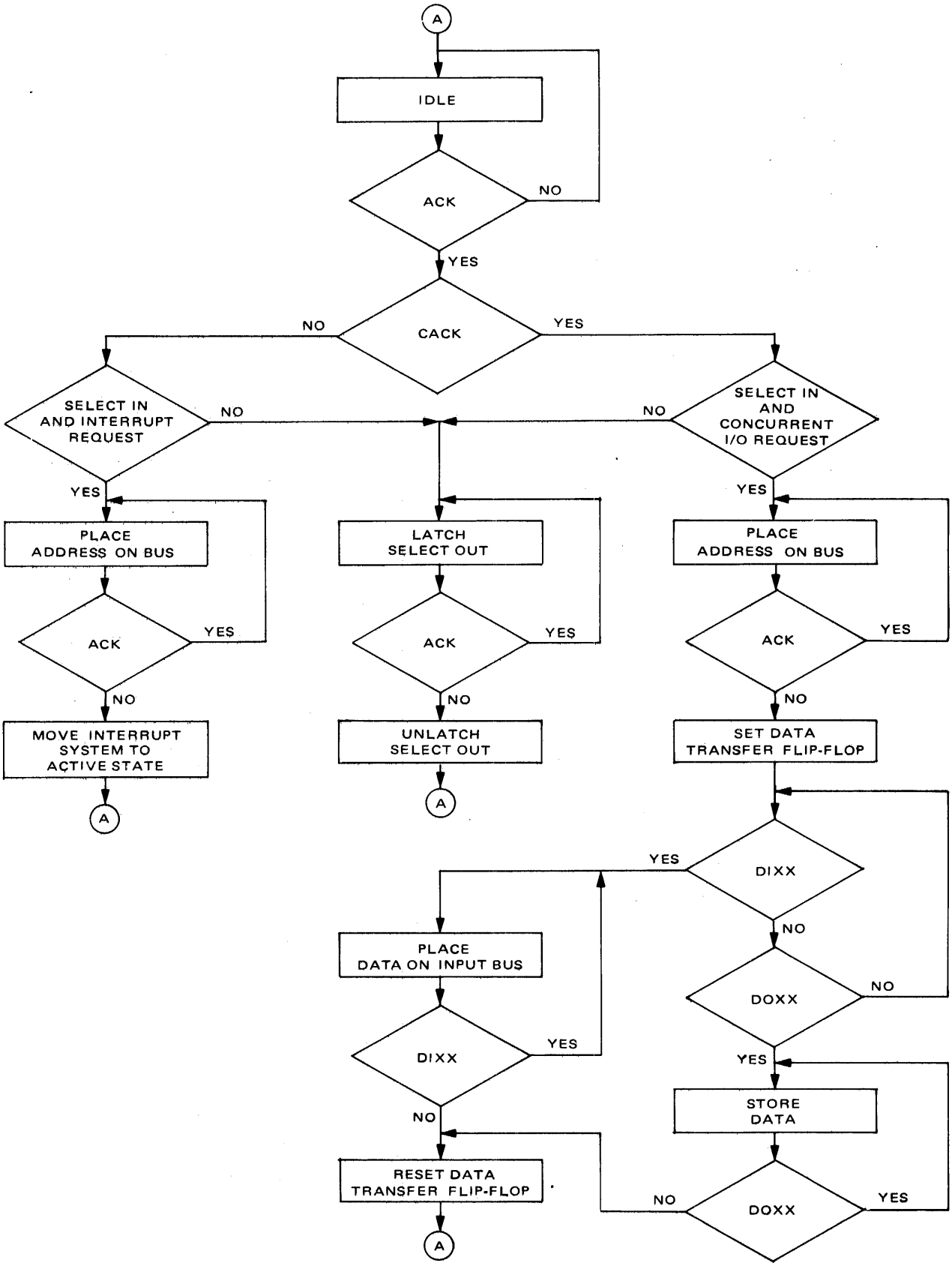


Figure D-13. Interrupt and Concurrent I/O Acknowledge Sequence

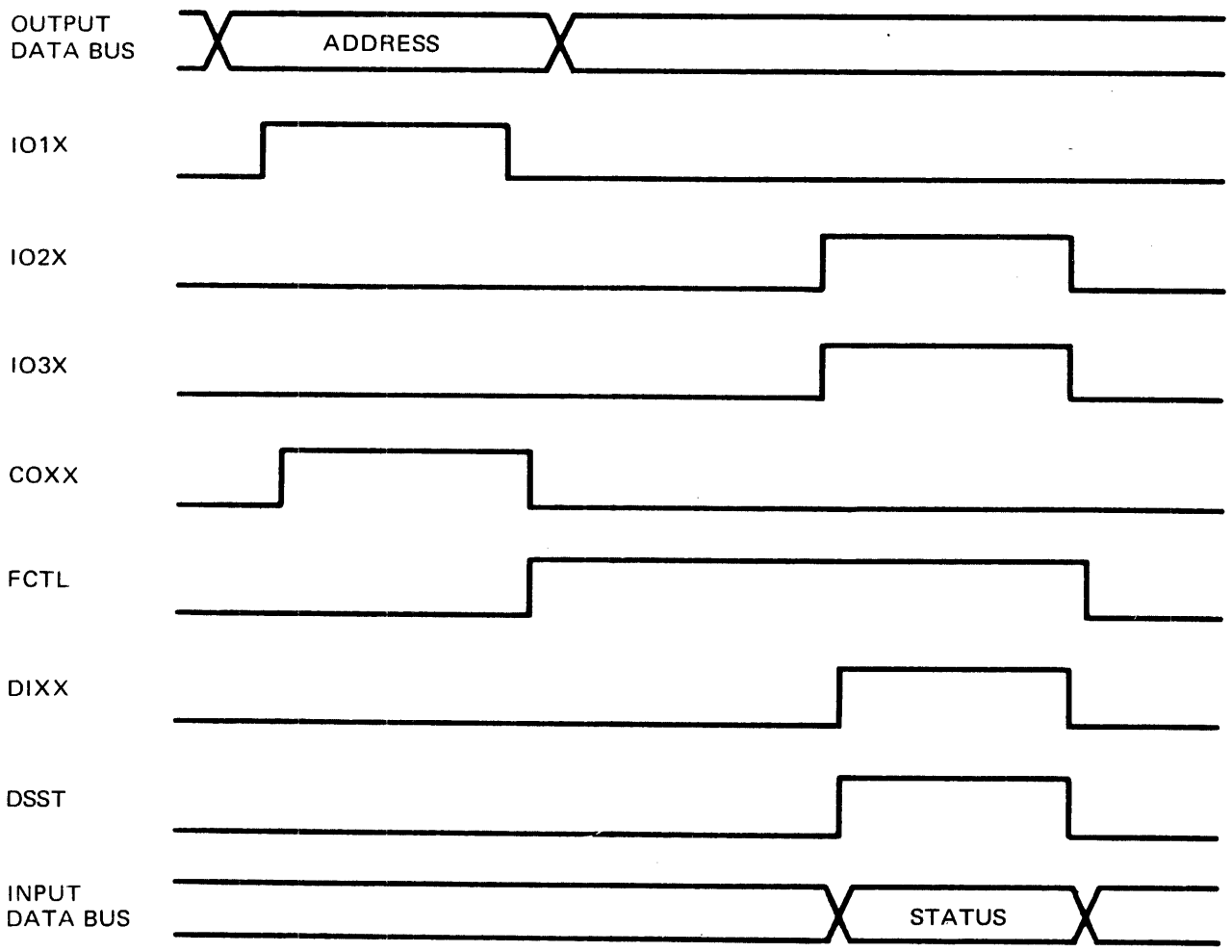


Figure D-14. Input Status Timing Diagram

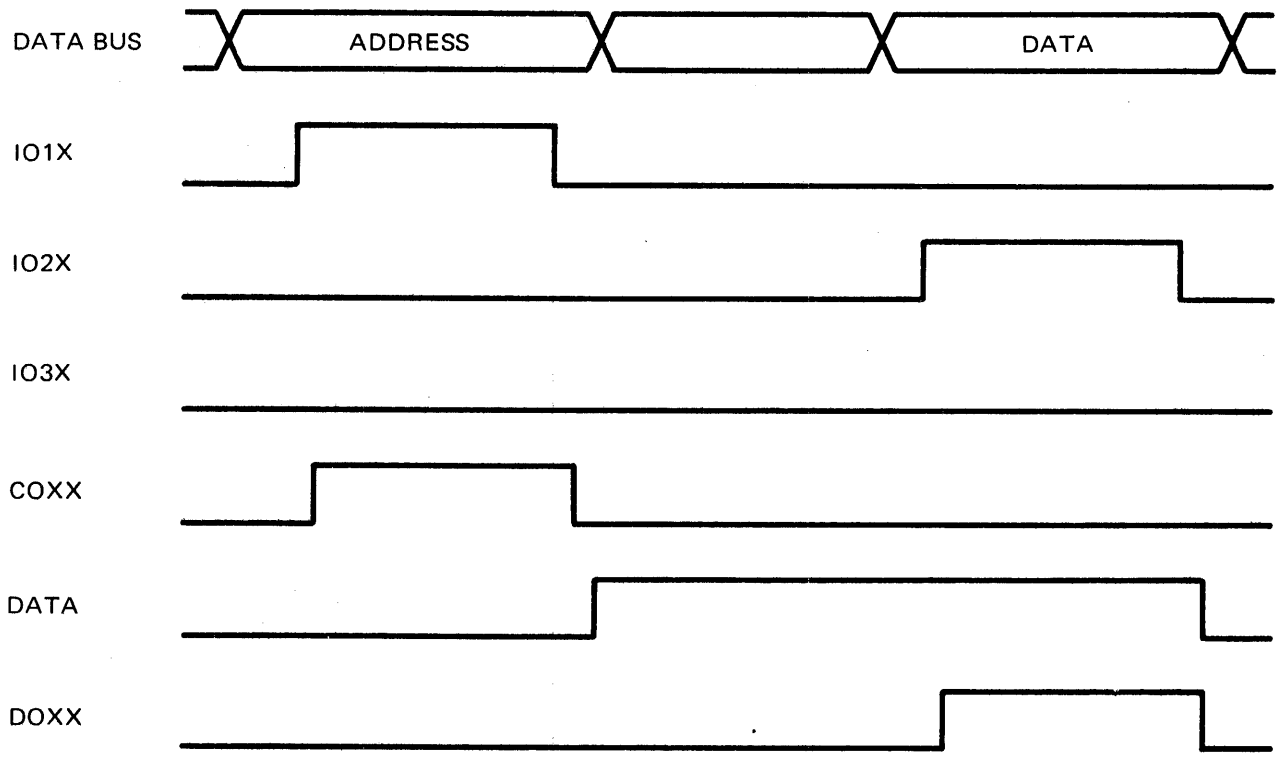


Figure D-15. Output Byte Timing Diagram

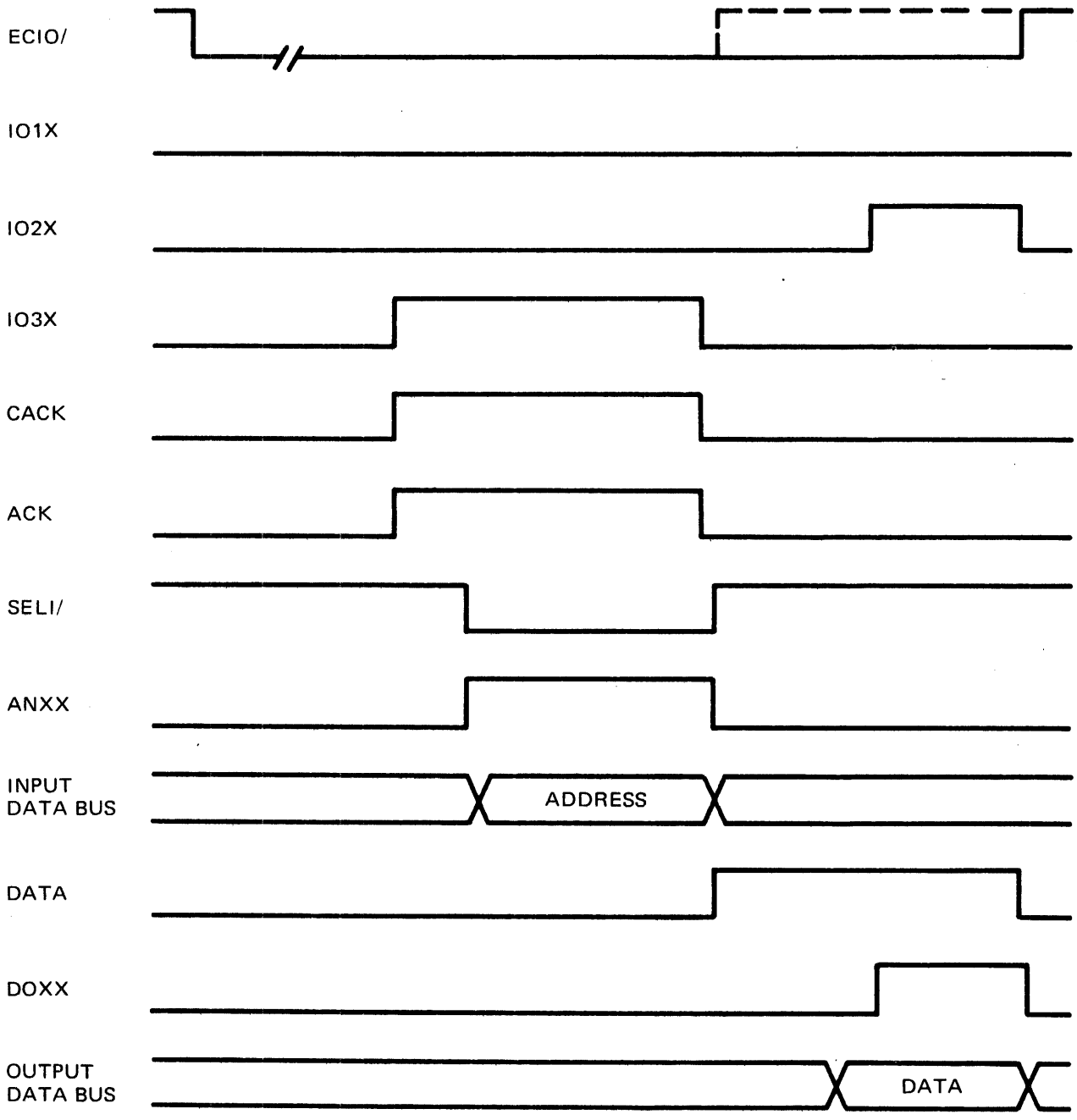


Figure D-16. Concurrent Out Data Transfer Timing Diagram

Table D-2. Byte I/O Controller Glossary

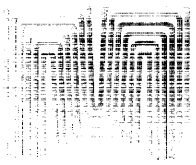
FIGURE	SIGNAL	SOURCE	NAME
D-1	OD00 thru OD07	Generated	Output data bus
	IOK/	Generated	Free-running clock
	MRES/	Generated	Master reset
	CPH1, CPH2/	I/O Bus	Processor Clock
	MRST/	I/O Bus	Processor master reset
D-2	DIXX	Generated	Decoded data input control signal
	IACK	Generated	Decoded interrupt acknowledge
	CACK	Generated	Decoded concurrent I/O acknowledge
	DOXX	Generated	Decoded data output control signal
	COXX	Generated	Decoded control output
	ACK	Generated	IACK or CACK
	IO1X/ IO2X/ IO3X/	I/O Bus	I/O Control Register outputs; contains encoded control information
D-3	DATX/	Generated	Decoded device order code signal for I/O data transfer
	FUNC/	Generated	Decoded device order code signal for function output or status input transfer
	ARM/	Generated	Decoded device order code signal to arm interrupt sequencer
	DSM/	Generated	Decoded device order code signal to disarm interrupt sequencer
	DISC/	Generated	Decoded device order code signal to disconnect from concurrent I/O mode
	CCIX/	Generated	Decoded device order code signal to enable concurrent input mode
	CCOX/	Generated	Decoded device order code signal to enable concurrent output mode

Table D-2. Byte I/O Controller Glossary (continued)

FIGURE	SIGNAL	SOURCE	NAME
D-3	DC07/	Generated	Spare decoded device order code signal
	KIXX/	Generated	Control clock made up of DIXX/, DOXX/, COXX/, or ANXX
D-4	DATA	Generated	Device order code store for data transfer. Data is transferred in or out during the second byte
	FCTL	Generated	Device order code store for function or status transfer. Function out or status information is transferred during the second byte
	CCIN	Generated	Device order code store for concurrent input operation. This mode remains active until disconnected
	CCOT	Generated	Device order code store for concurrent output operation. This mode remains active until disconnected
D-5	ID00/ thru ID07/	Generated	Input data bus drivers. Data to be driven on the bus comes from the data switch. Controlling the data presentation to the input bus is enabled by IDEN
	IDEN	Generated	Input data bus enable. This enable term is logically expressed as follows: (FCTL + Data) DIXX + ANXX
D-6	ANXX	Generated	Answer acknowledge. Allows controller to present its address during acknowledge period
	ID0G thru ID7G	Generated	Input Data Gate. Outpus of multiplexers which select status, data, or interrupt address to be gated on to computer I/O input data lines

Table D-2. Byte I/O Controller Glossary (continued)

FIGURE	SIGNAL	SOURCE	NAME
D-6	ID0X/ thru ID7X/	Generated	Input Data Multiplexer. Data lines internal to controller; input to multiplexers which select signals for I/O bus data lines
	STEN	Generated	Status enable
	ST7X/	Generated	Input Status, Multiplexer. Status lines internal to computer; input to multiplexers which select signals for I/O bus data lines.
D-7	SELO/	Generated	Select out. Priority line propagated to next controller in priority. Line is activated during acknowledge period
	CCAN/	Generated	Concurrent address enable. Occurs from concurrent I/O request only
	ECIO/	Generated	Concurrent I/O request
	PROT/	Generated	Interrupt priority out line. Propagated to next device in priority
	EINT/	Generated	External interrupt request
	CORQ/	Generated	Concurrent output request
	CCRQ	Generated	Concurrent I/O request. OR of concurrent input and output request
	CIRQ/	Generated	Concurrent input request
	INRQ/	Generated	Interrupt request
	STEN	Generated	Status enable
	SELI/	I/O Bus	Select in
	PRIN/	I/O Bus	Priority in external (interrupt system only)
D-10	IS01, IS02	Generated	Interrupt sequencer flip-flops
	IANX/	Generated	Interrupt address enable
D-12	IC01, IC02	Generated	Interface control storage register



™ **Microdata**

Microdata Corporation
17481 Red Hill Avenue
Irvine, California 92714
(714) 540-6730 TWX: 910-595-1764