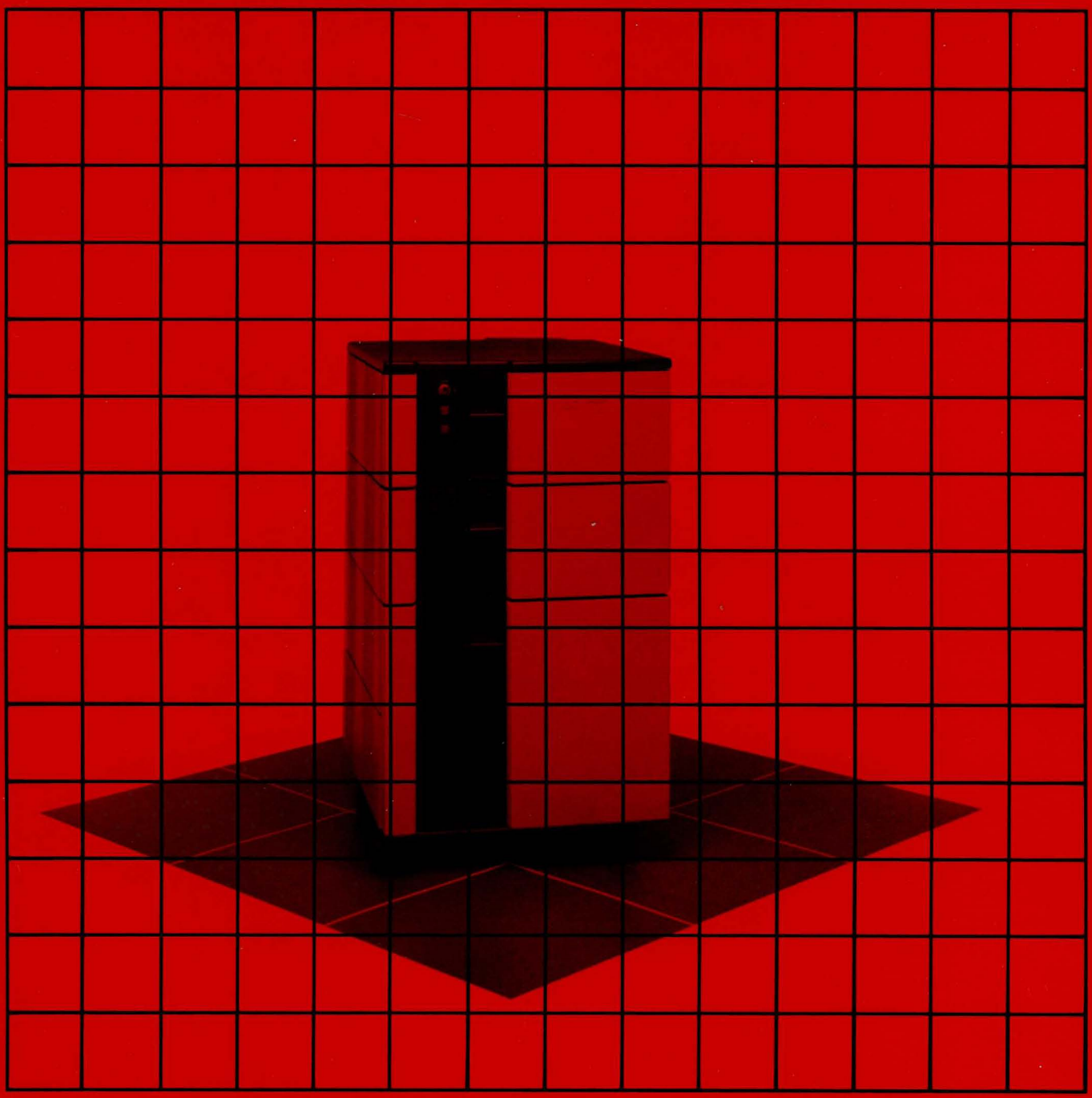
Zilog

EXXON COMPUTER SYSTEMS



System 8000

System 8000

Hardware Reference Manual

03-3198-01

March, 1982

NOTICE TO OWNER

FEDERAL COMMUNICATIONS COMMISSION RADIO FREQUENCY INTERFERENCE STATEMENT

Warning: This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. As temporarily permitted by regulation it has not been tested for compliance with the limits for Class A computing devices pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

SYSTEM 8000 HARDWARE REFERENCE MANUAL

PRELIMINARY VERSION

The information contained in this draft may undergo changes, both in content and organization, before arriving at its final form.

PREFACE

This manual contains the information that is necessary to install, operate, understand, and maintain Zilog's System 8000™ family of 16-bit microcomputers. The audience that this manual addresses consists of field engineers (FE), service technicians, and all others who require in-depth knowledge of the hardware aspects of the system.

This manual and the related manuals listed below provide the complete technical documentation of the System 8000.

Title	Zilog Number
System 8000 User Manual	03-3199
ZEUS System Administrator Manual	03-3197
ZEUS Utiltites Manual	03-3196
ZEUS Reference Manual	03-3195
CPU Hardware Reference Manual	03-3200
Secondary Serial Board Manual	03-3201
Memory Subsystem	03-3202
Winchester Disk Controller	03-3203
Cartridge Tape Controller	03-3204

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

LIST OF ILLUSTRATIONS

Figure

LIST OF TABLES

Table

LIST OF TABLES (continued)

Table

SECTION 1

OVERVIEW

1.1 System Description

The System 8000 is a free-standing, semiportable unit, based on Zilog's 16-bit Z8001 microprocessor and running on the ZEUS Operating System at 5.5 megahertz. Figure 1-1 shows the enclosure containing the modules that make up the system. The top module has a ten-slot card cage and, behind the card cage, two power supplies. The module beneath the top module has a 24-megabyte Winchester disk drive and a 17-megabyte cartridge tape drive. The remainder of the enclosure beneath the two modules can be used for storage or for expanding the capabilities of the system by adding additional disk or tape units or both.

The basic system contains five printed circuit boards in the ten-slot card cage. These boards (Figures 1-2 through 1-6) are the CPU, the Winchester Disk Controller, the Cartridge Tape Controller, the Memory Subsystem Controller, and the 256K Byte ECC Memory. The lower five slots of the card cage are dedicated to memory; with the addition of three more memory boards, the basic 256-kilobyte system can have a physical memory space of 1 megabyte, not including the small bootstrap memory on the CPU board.

1.2 Functional Relationships

The block diagram in Figure 1-7 shows the functional relationships of the major elements (boards) that make up the basic system. These elements communicate with one another over Zilog's Z-Bus Backplane Interconnect (ZBI), a high-speed, 32-bit, semisynchronous bus. The following paragraphs briefly describe the functions of each element on the bus.

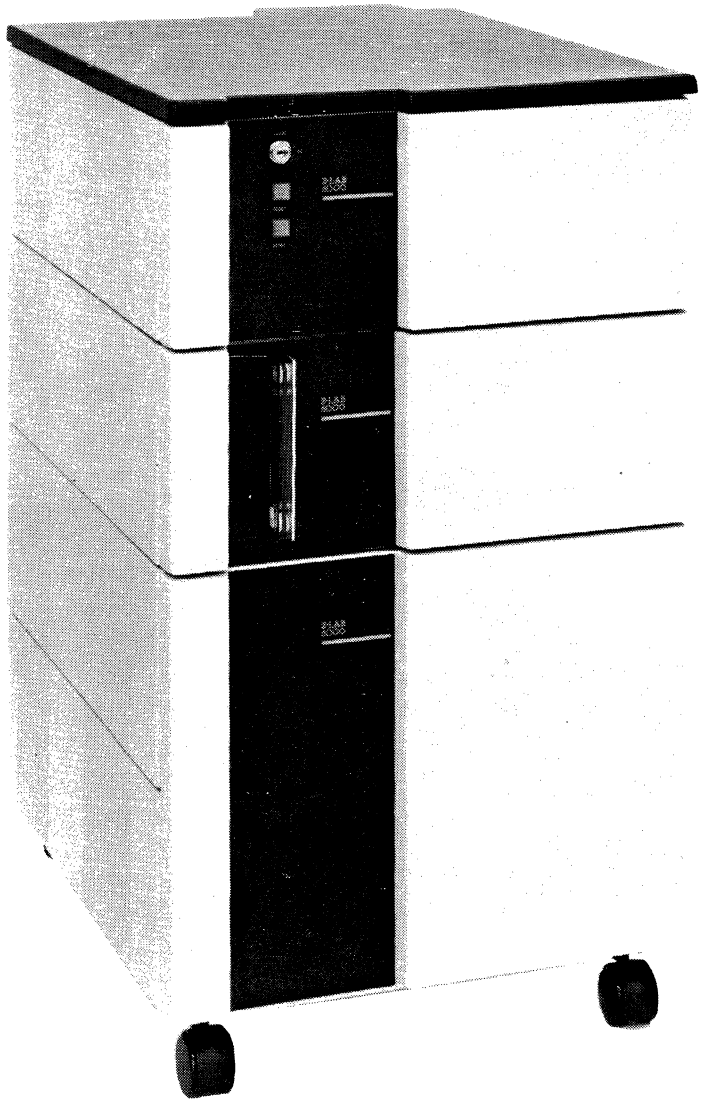


Figure 1-1. System 8000

1.2.1 CPU Board

The CPU is the host of the System 8000; it controls the ZBI and terminal communications into and out of the system. As shown in Figure 1-7, the CPU supports eight serial I/O ports and a parallel I/O port. The serial ports are compatible with both the RS232-C and the RS423 standards. The parallel port, with the appropriate jumpers inserted, is compatible with the line printer standards of either Centronics or Data Products.

1.2.2 Winchester Disk Controller

The disk controller enables the CPU to communicate with up to four Winchester disk drives. An intelligent bus module with an on-board Z80B microprocessor, the controller can be polled or interrupt-driven by the CPU. The appropriate jumper arrangement determines the controller's mode of operation. The disk is organized into 512-byte sectors. A single command can cause the transfer of up to 128 512-byte sectors. In addition, the disk controller uses a full-track buffer which permits the transfer of 24 512-byte sectors in one disk revolution.

1.2.3 Cartridge Tape Controller

The tape controller is the intelligent interface between the CPU and up to four cartridge tape drives. A Zilog Z80B microprocessor controls the operation of the controller. The controller uses direct-memory access (DMA) to transfer data between the cartridge tape drive and the CPU. When the CPU wants to initiate an operation, it sends a command to the controller. The controller completes the specified operation and then interrupts the CPU to notify it that the operation is complete.

1.2.4 Memory Subsystem Controller

The memory controller can control the operation of up to 16 megabytes of dynamic read/write memory. The data can be transferred as bytes (eight bits), words (16 bits), or long words (32 bits); the controller translates the width of the data and places the data in the proper locations. The controller transparently corrects all single-bit errors and detects all double-bit errors. The controller uses a soft-error logging system that counts soft errors in each 64K-byte block of memory.

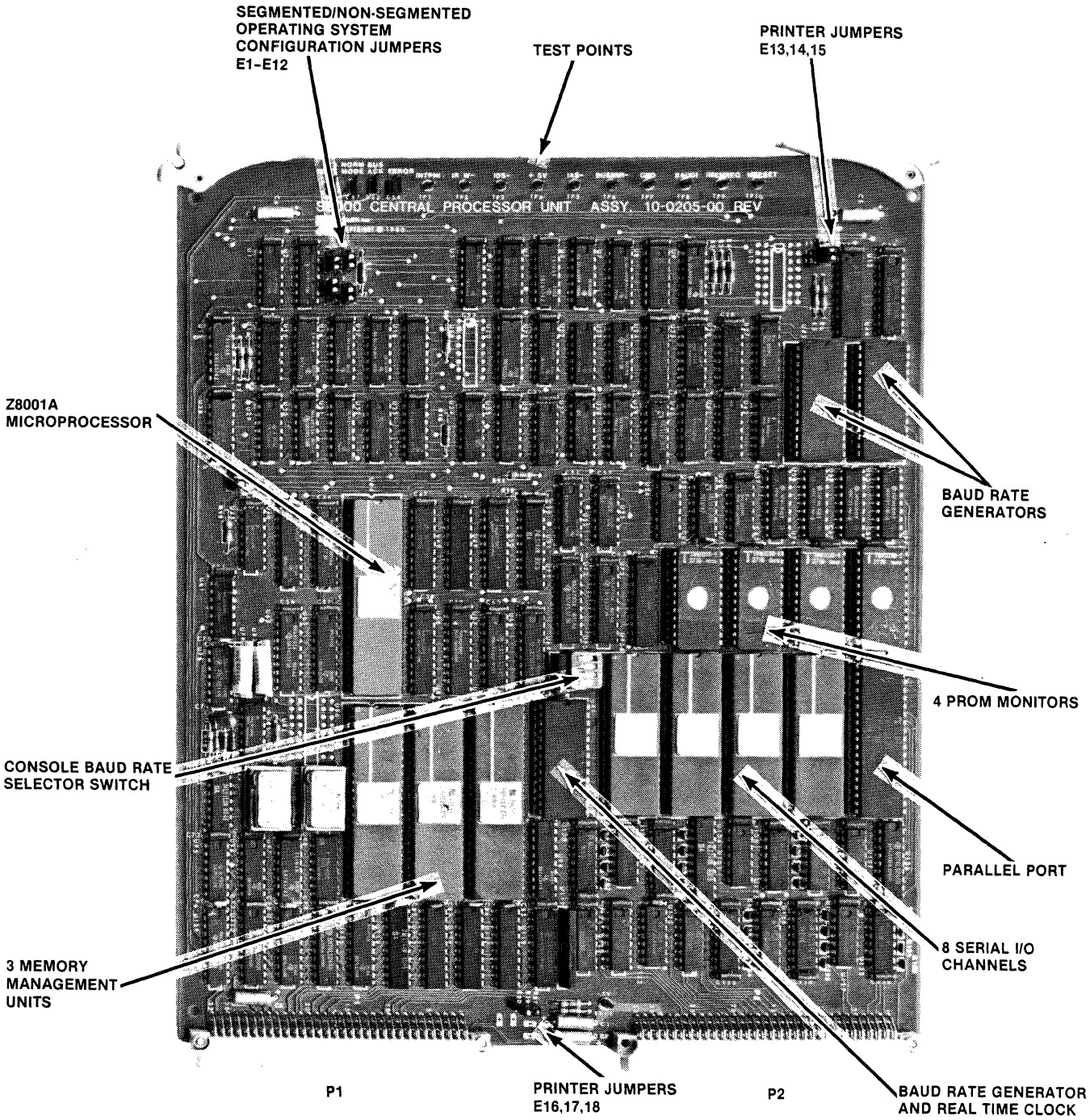


Figure 1-2. System 8000 CPU Board

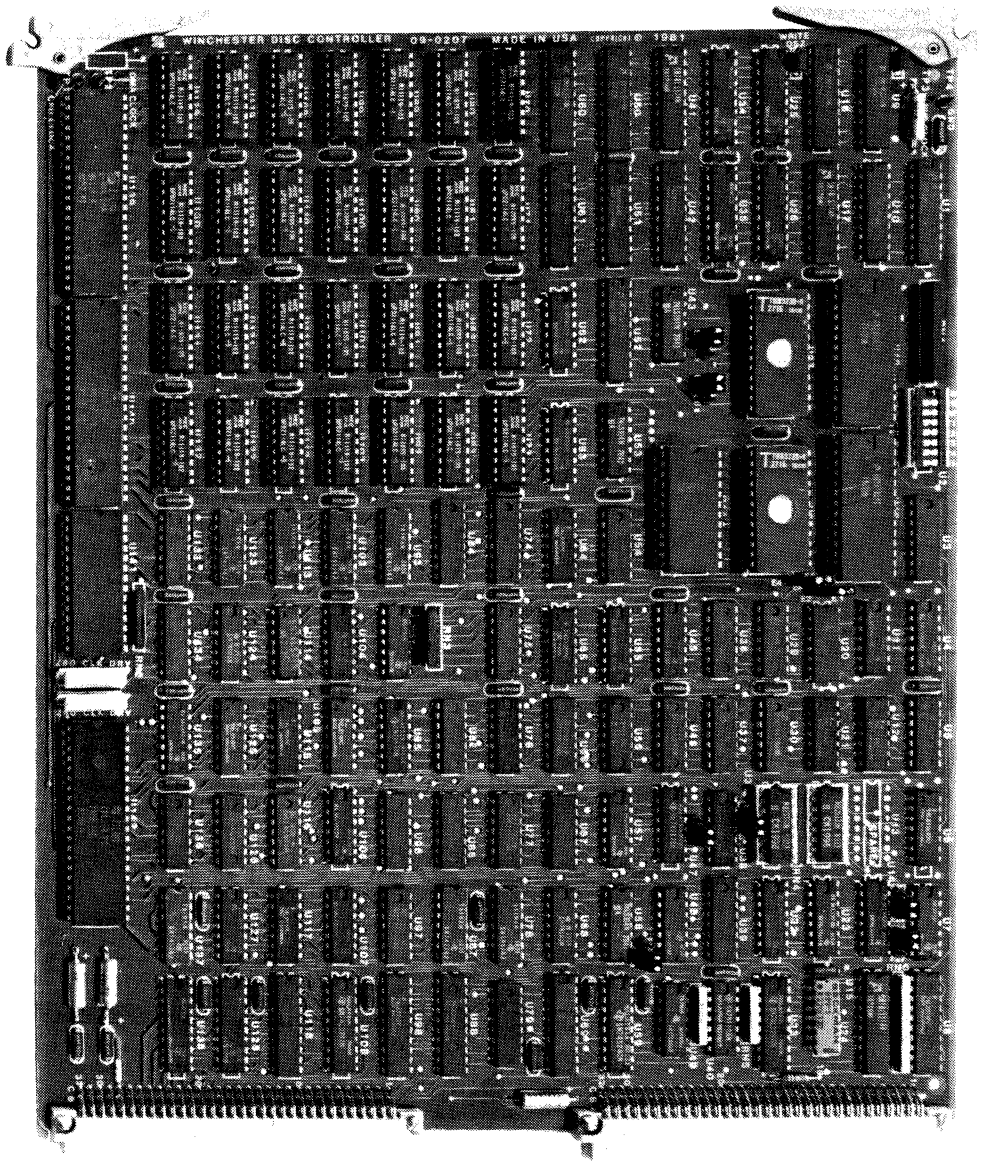


Figure 1-3. System 8000 Winchester Disk Controller

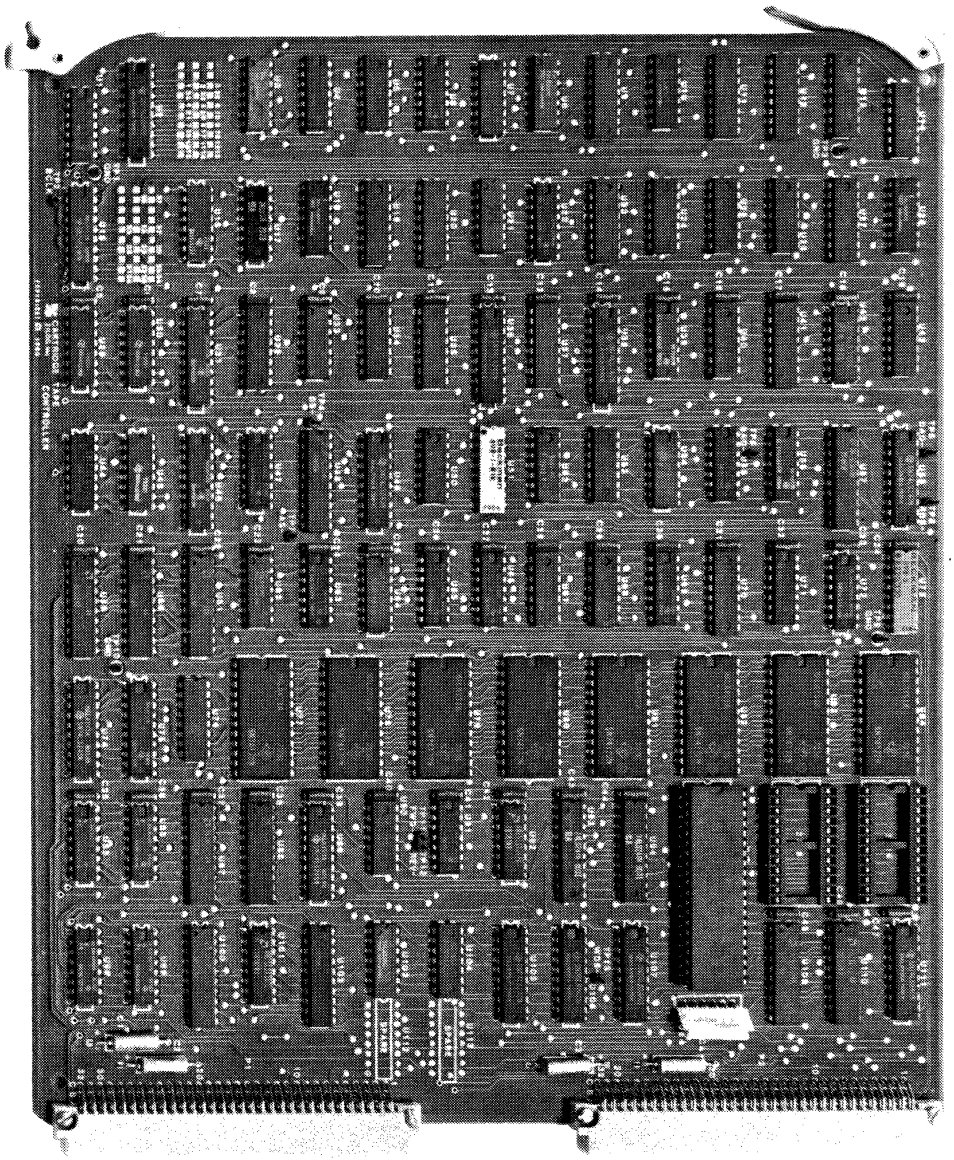


Figure 1-4. System 8000 Cartridge Tape Controller

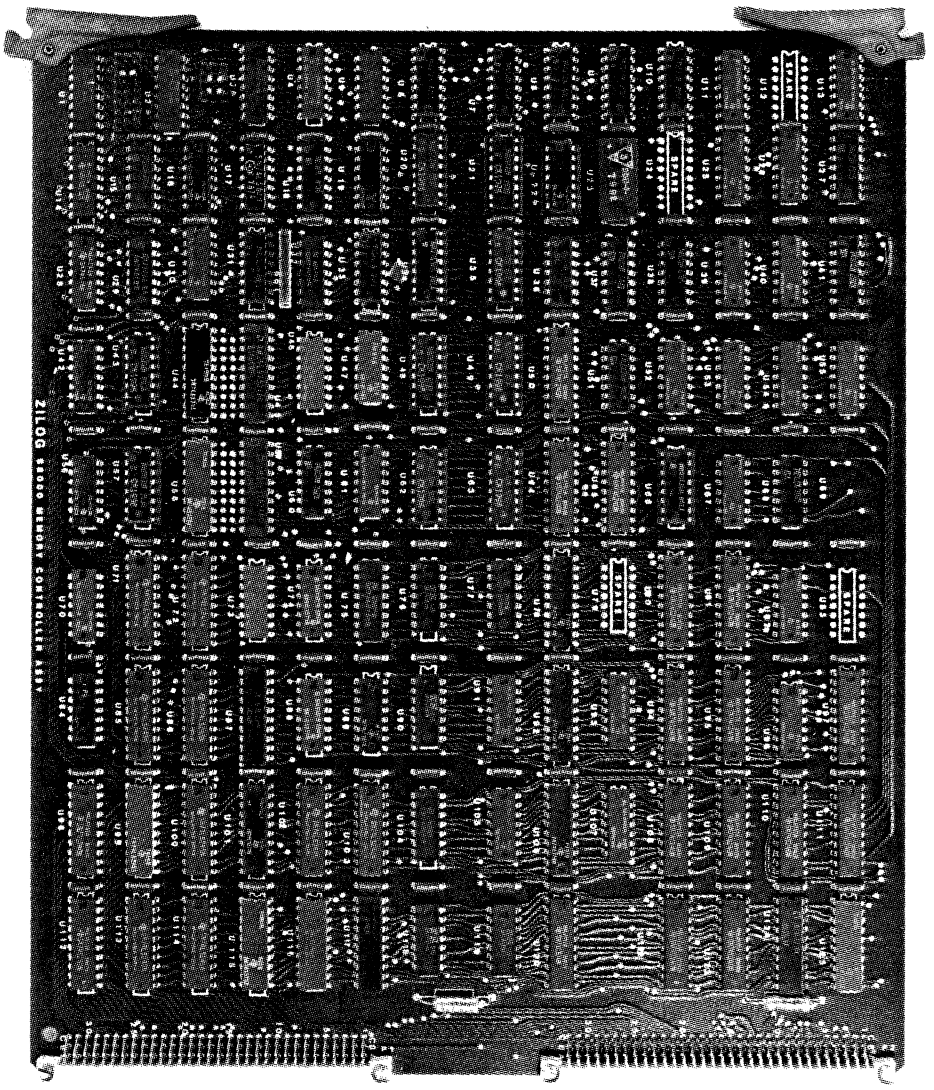


Figure 1-5. System 8000 Memory Subsystem Controller

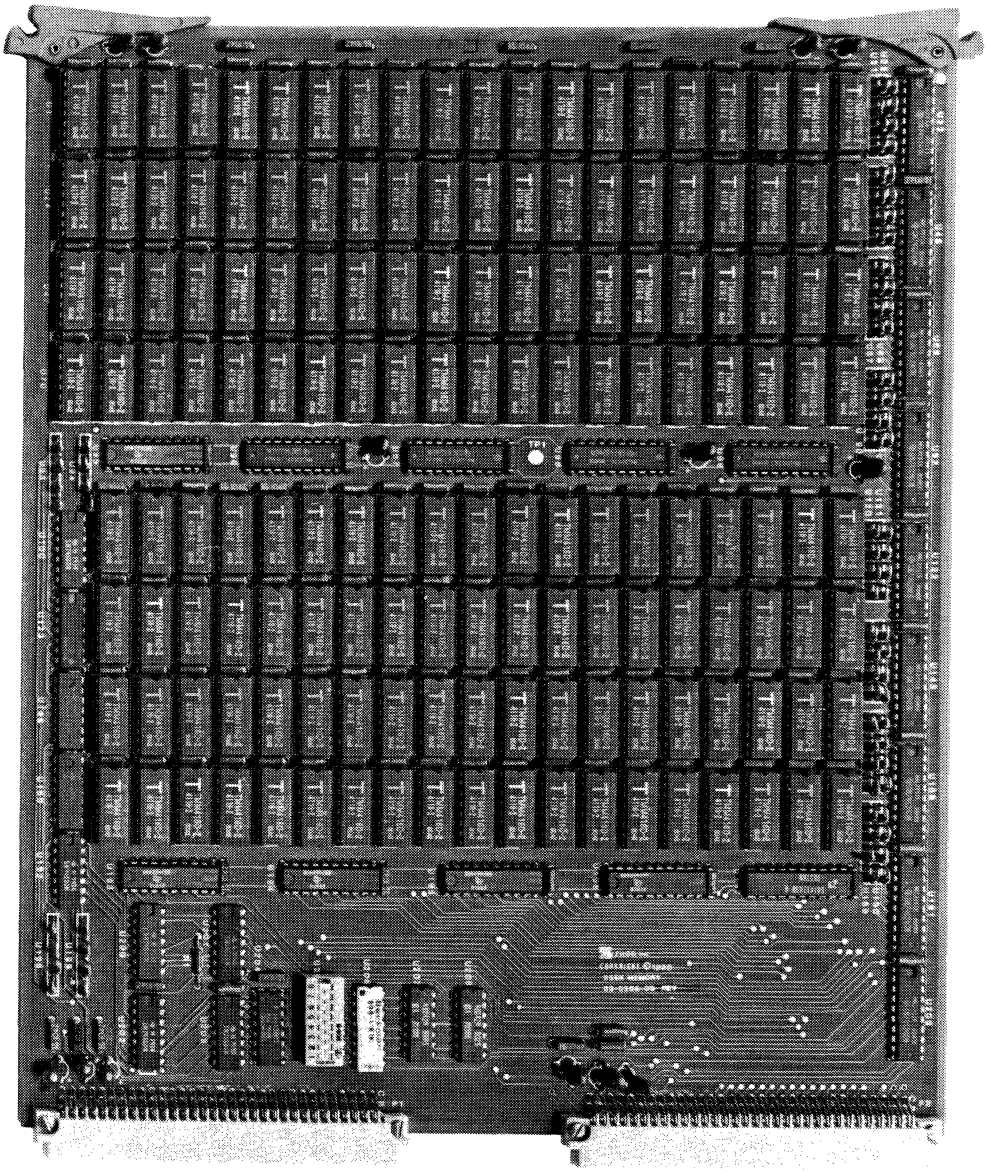


Figure 1-6. ECC Memory Array, 256 Kilobytes

1.2.5 Memory Array

The ECC (error checking and correcting) Memory Array contains storage space for 256 kilobytes of data; in addition, it has storage for the syndrome bits used by the error checking and correcting logic. All data during memory transactions passes through the memory controller; the memory has no direct connection to the system bus (ZBI) (Figure 1-7). Instead, a high-speed, 32-bit bus connects the memory to the memory controller.

1.3 System Expansion

The System 8000 can be expanded by adding up to three additional 24-megabyte or 40-megabyte disk drives, three additional cartridge tape drives, up to six megabytes of ECC RAM, and a Secondary Serial I/O Board that can support an additional eight serial I/O ports and one additional parallel printer port.

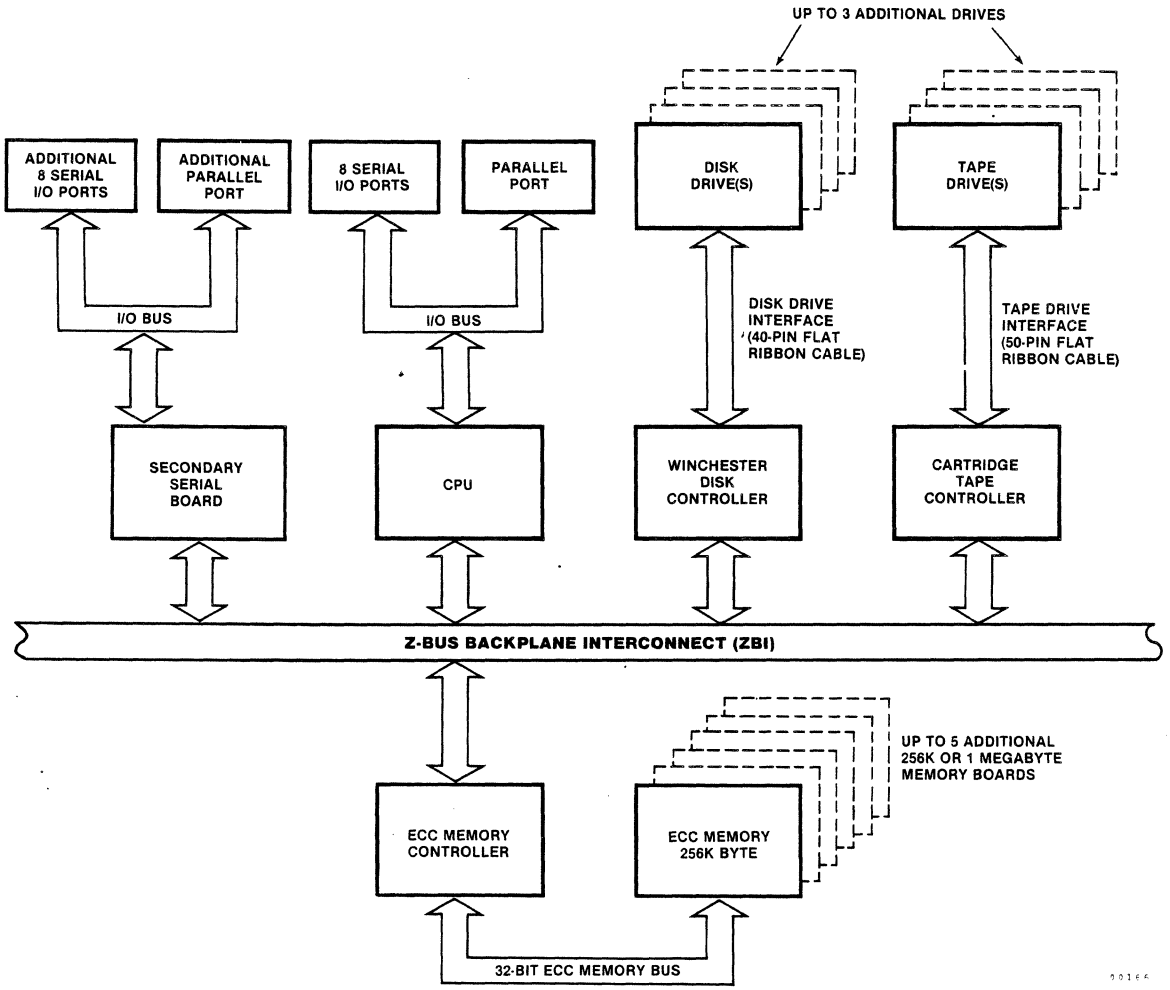


Figure 1-7. System 8000, Functional Relationships

SECTION 2

SYSTEM SPECIFICATIONS

2.1 Introduction

This section contains the electrical, performance, physical, and environmental specifications for the System 8000, Model 20.

2.2 Electrical Specifications

Table 2-1 lists the power requirements of systems that are installed in the U.S. (domestic). Table 2-2 lists the power requirements of systems that are installed outside of the U.S.; these requirements are the international requirements.

Table 2-1. Electrical Requirements, Domestic

ITEM	REQUIREMENT
Voltage	117 Vac nominal (range: 105 to 128 Vac)
Phase	Single
Frequency	60 Hertz
Current	Sustained: 5 amperes maximum Surge : 8 amperes maximum

Table 2-2. Electrical Requirements, International

ITEM	REQUIREMENT
Voltage	220 Vac (range: 198 to 242 Vac)
Phase	Single
Frequency	50 Hertz
Current	Sustained: 2.5 amperes maximum Surge: 4 amperes maximum

2.3 Performance Specifications

Tables 2-3 and 2-4 define the performance characteristics of the disk and tape drives.

Table 2-3. Winchester Disk Performance Characteristics

ITEM	CHARACTERISTIC
Rotational speed	3,600 RPM
Power-on to ready time	15 seconds
Average positioning time	42 milliseconds
Number of surfaces	3 or 5
Tracks per surface	600
Sectors per track	24
Bytes per sector	512
Rate of data transfer	801 kilobytes per second

Table 2-4. Cartridge Tape Drive Performance Characteristics

ITEM	CHARACTERISTIC
Read/Write speed	30 inches per second (ips)
Rewind/Search speed	90 inches per second (ips)
Tracks	4
Recording density	6400 BPI

2.4 Physical Specifications

This paragraph defines the specifications for the modules, boards, card cages, and connectors that make up the System 8000. Table 2-5 lists the basic specifications of an assembled system.

Table 2-5. Physical Specifications

ITEM	SPECIFICATION
Height	84 centimeters (33 inches)
Width	48 centimeters (19 inches)
Depth	61 centimeters (24 inches)
Weight	60 kilograms (132 pounds) approximate

2.4.1 Modules

The modules, with their side panels removed, can be mounted in standard 19-inch racks. In stand-alone configurations, the modules can be stacked. The dimensions of the module with its side panel removed are shown in Figure 2-1.

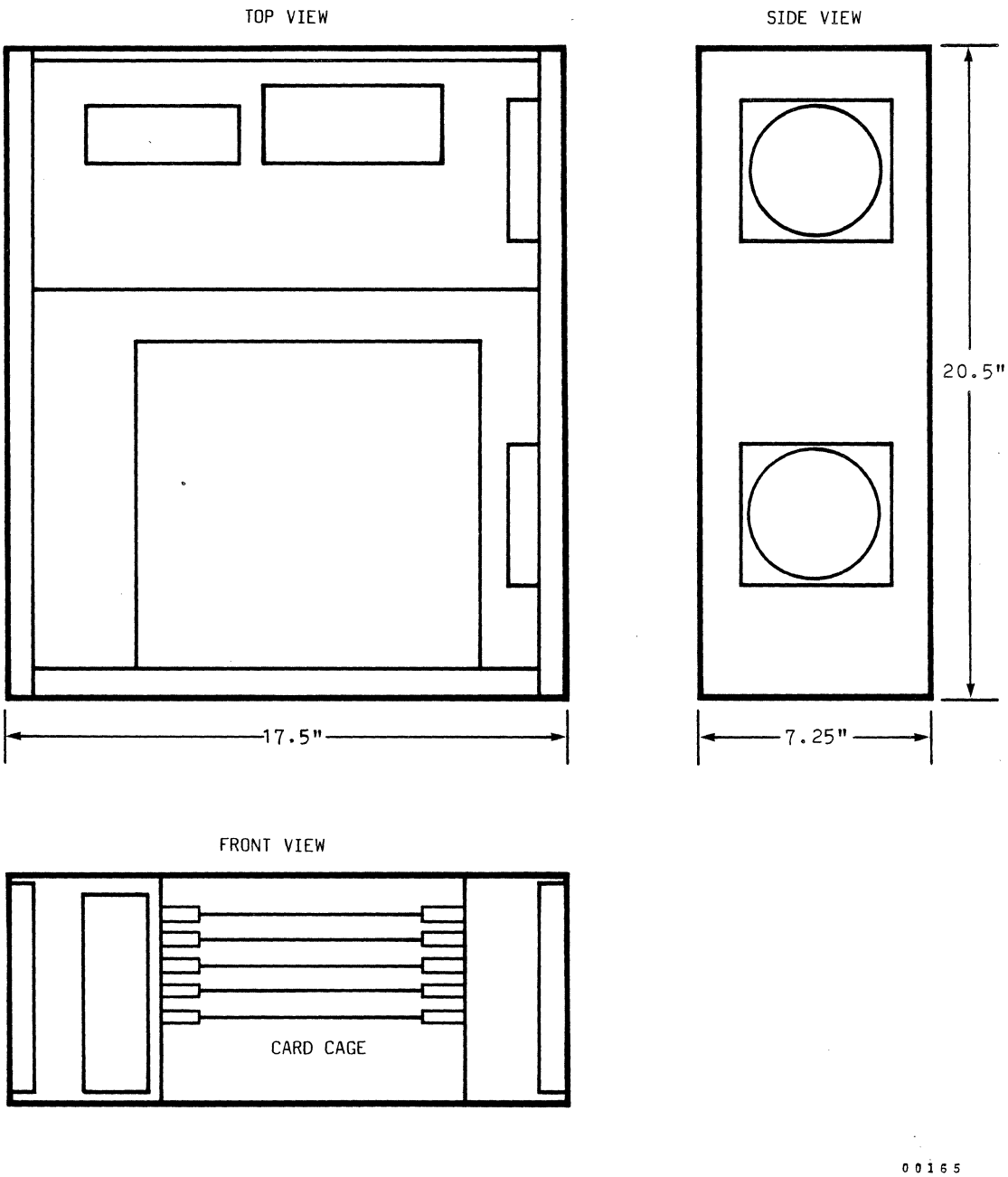


Figure 2-1. Modules without Side Panels, Dimensions

2.4.2 I/O Connectors

Figure 2-2 identifies the I/O connectors on the rear panel of the CPU Module. Table 2-6 lists the required mating connectors and sources; Table 2-7 lists the pin assignments of the serial (TTY) I/O connectors. Table 2-8 lists the pin assignments of the parallel (PRINTER) connectors.

NOTE

When serial I/O uses only the transmit and receive data lines, connect:

a) pin 4 (RTS) to pin 5 (CTS)
b) pin 6 (DSR) to pin 20 (DTR)

on the connector to avoid intermittent operation of serial lines.

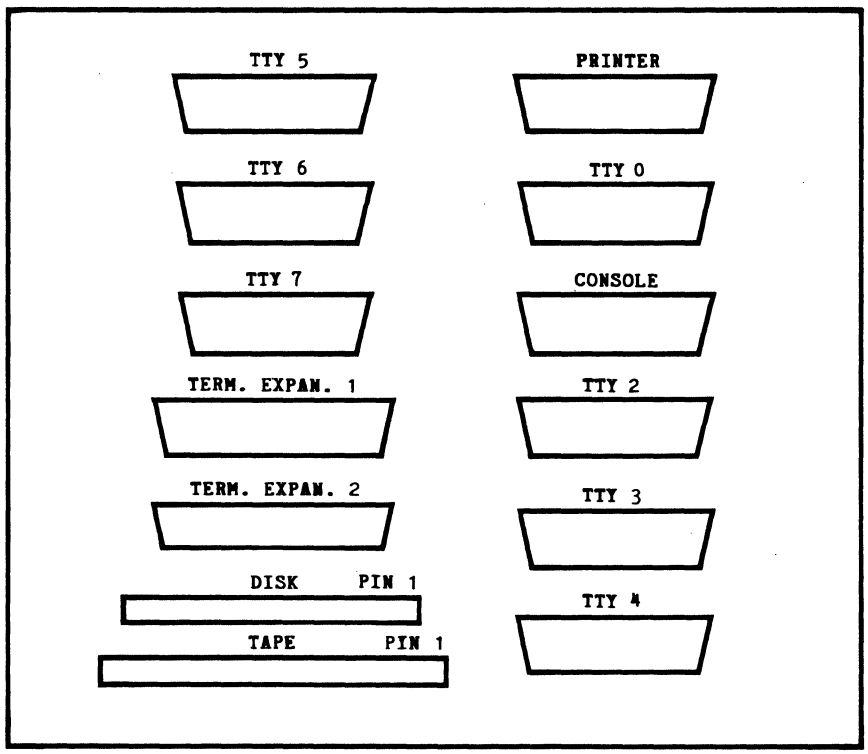


Figure 2-2. I/O Connector Panel, Back of CPU Module

Table 2-6. I/O Connectors

DESIGNATION	DESCRIPTION	VENDOR/PART NUMBER
Printer	25-pin D Connector	ITT Cannon DBUE25SBB
TTY0-TTY7		TRW Cinch DBUE25SBF
Terminal Expansion 1 and 2	37-pin D Connector	ITT Cannon DCUE37SBB TRW Cinch DCUE37SBF
Disk Drive	40-wire flat ribbon socket	3M 3417-6040
Tape Drive	50-wire flat ribbon socket	3M 3425-6050

Table 2-7. TTY Connector, Pin Assignments

SIGNAL NAME	PIN
Ground	7
DTR	20
RTS	4
TXD	2
DSR	6
CTS	5
RXD	3

Table 2-8. Printer Connector, Pin Assignments

Data Products Interface

Signal Name	P2 Backplane	Printer Port Connector Pins	Printer Connector Pins
DATA 0	P2-1C	1	B
DATA 1	P2-2C	2	F
DATA 2	P2-3C	3	L
DATA 3	P2-5C	4	R
DATA 4	P2-6C	5	V
DATA 5	P2-8C	6	Z
DATA 6	P2-9C	7	n
DATA 7	Not Used	Not Used	Not Used
DATA STROBE	P2-12C	9	j
INPUT PRIME	Not Used	Not Used	Not Used
DATA DEMAND	P2-16C	11	E
FAULT	P2-20C	12	C
READY	P2-17C	23	cc
ONLINE	P2-21C	24	y
SIGNAL GROUND	P2-19C	22	X

Centronics Interface

Signal Name	P2 Backplane	Printer Port Connector Pins	Printer Connector Pins
DATA 0	P2-1C	1	2
DATA 1	P2-2C	2	3
DATA 2	P2-3C	3	4
DATA 3	P2-5C	4	5
DATA 4	P2-6C	5	6
DATA 5	P2-8C	6	7
DATA 6	P2-9C	7	8
DATA 7	Not Used	Not Used	Not Used
DATA STROBE	P2-12C	9	1
INPUT PRIME	Not Used	Not Used	Not Used
ACKNOWLEDGE	P2-16C	11	10
FAULT	P2-20C	12	32
GROUND	P2-32C	18	24
GROUND	P2-32B	19	25
GROUND	P2-32A	20	26
BUSY	P2-17C	23	11
SELECT	P2-21C	24	13

2.5 Environmental Specifications

The System 8000 can be expected to perform reliably provided the environmental specifications listed in Table 2-9 are maintained.

Table 2-9. Environmental Specifications

Operating temperature:

10 degrees C (50 degrees F) minimum
40 degrees C (104 degrees F) maximum

Relative humidity:

80% noncondensing

Noise level:

48 dba

2.6 Backplane (ZBI) Pin Assignments

Figure 2-3 shows the backplane into which the circuit boards are plugged. The view is from the front of the card cage. Only the connectors designated J11 through J20 (on the right) connect to the system bus, known as the Z-Bus Backplane Interconnect (ZBI). The connectors designated J21 through J30 (on the left) are auxiliary connectors. The pin assignments of all ZBI connectors are the same; Table 2-10 lists these ZBI pin assignments.

Table 2-10. Backplane Connector, J11 through J20 (ZBI), Pin Assignments

PIN	Row A SIGNAL	Row B SIGNAL	Row C SIGNAL
1	RESET\	WAIT\	CAVAIL
2	CAI\	CAO\	CPUREQ\
3	BAI\	BAO\	BUSREQ\
4	MMAI\	MMAO\	GND
5	IEI3	IEO3	MMREQ\
6	IEI2	IEO2	-
7	IEI1	IEO1	GND
8	INT1\	INT2\	INT3\
9	R/W\	B/W\	W/LW\
10	S2	S3	S4
11	S0	S1	GND
12	ME\	AS\	DS\
13	-	STOP\	N/S\
14	-	-	-
15	AD31	-	GND
16	AD28	AD29	AD30
17	AD25	AD26	AD27
18	AD22	AD23	AD24
19	AD20	AD21	GND
20	AD17	AD18	AD19
21	AD14	AD15	AD16
22	AD11	AD12	AD13
23	AD9	AD10	GND
24	AD6	AD7	AD8
25	AD3	AD4	AD5
26	AD0	AD1	AD2
27	PWRBAD\	MCLK	BLCK
28	+5v	+5v	+5v
29	-5v	-5v	-5v
30	+12v	+12v	+12v
31	-12v*	-12v*	-12v*
32	GND	GND	GND

* -12v is allocated space on the ZBI backplane, but is not used nor generated by the S8000 system.

The pin assignments of the connectors J21 to J30 differ from slot to slot; the bottom 5 connectors are the same. Figure 2-3 shows how the slots are dedicated; the following list indicates the tables that have the pin assignments for the different J2 connectors.

1) Table 2-11:
   connector J21, CPU board

2) Table 2-12:
   connector J22, Extended Serial I/O board (optional)

3) Table 2-13:
   connector J23, Winchester Disk Controller board

4) Table 2-14:
   connector J24, Cartridge Tape Controller board

5) Table 2-15:
   connector J26 through J30, memory bus:
   Memory Subsystem Controller and Memory boards

* EXTENDED SERIAL I/O

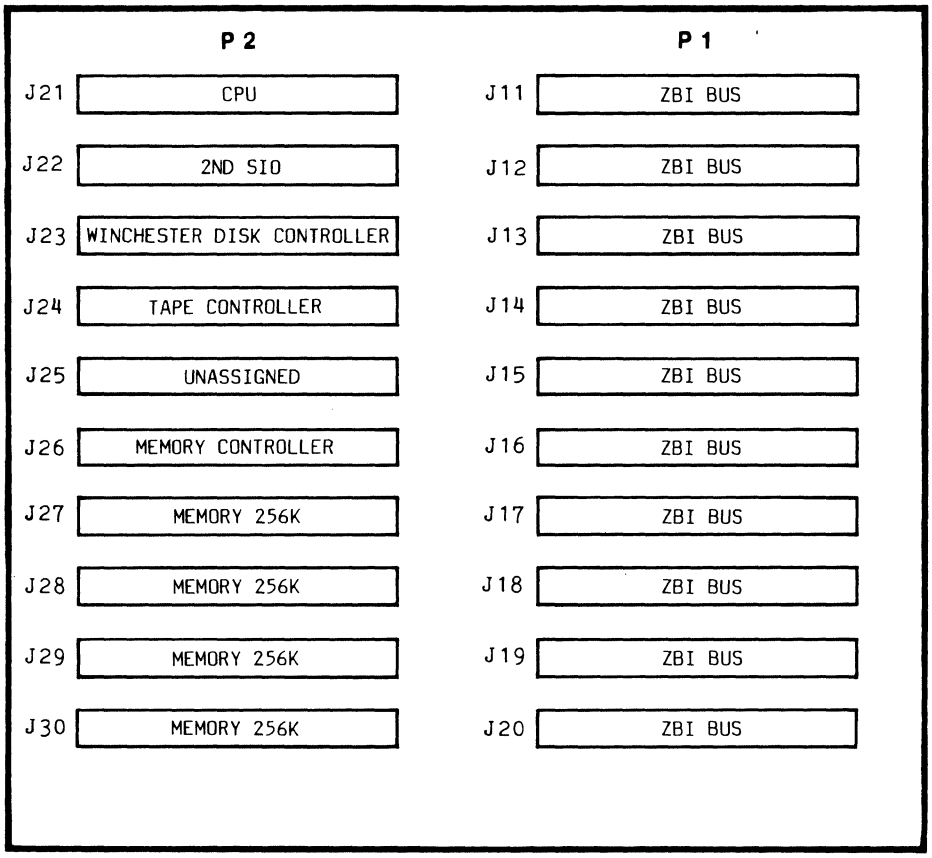


Figure 2-3. Backplane of Card Cage in CPU Module, Slot Assignment (Viewed from front of Card Cage)

Table 2-11. CPU Board, Connector P2/J21, Slot 1

PIN	Row A SIGNAL	Row B SIGNAL	Row C SIGNAL
1	TXRTN0	RXD0	DATA0
2	TXD0	CTS0	DATA1
3	RTS0	DTR0	DATA2
4	DSR0	RXD1	GND
5	TXD1	CTS1	DATA3
6	RTS1	DTR1	DATA4
7	DSR1	RXD2	GND
8	TXD3	CTS2	DATA5
9	RTS2	DTR2	DATA6
10	DSR2	RXD3	DATA7
11	TXD3	CTS3	GND
12	RTS3	DTR3	DATA STROBE/DATA STROBE\
13	DSR3	RXD4	N.U./INPUT PRIME
14	TXD4	CTS4	TXRTN1
15	RTS4	DTR4	GND
16	DSR4	RXD5	D.D./ACKNOWLEDGE\
17	TXD5	CTS5	BUSY\
18	RTS5	DTR5	TXRTN2
19	DSR5	RXD6	GND
20	TXD6	CTS6	IFVALID/FAULT\
21	RTS6	DTR6	ON-LINE/SELECT
22	DSR6	RXD7	F.P BUSACK INDICATOR
23	TXD7	CTS7	F.P. POWER-ON INDICATOR (GND)
24	RTS7	DTR7	F.P. NORMAL INDICATOR
25	DSR7	TXRTN5	NMI SWITCH (NORMALLY CLOSED)
26	TXRTN3	TXRTN6	NMI SWITCH (NORMALLY OPEN)
27	TXRTN4	TXRTN7	SW RESET
28	+5v	+5v	F.P. INDICATOR V+ (+5v)
29	-5v	-5v	-5v
30	+12v	+12v	+12v
31	-12v	-12v	-12v
32	GND	GND	GND

Table 2-12. Secondary Serial I/O, Connector P2/J22, Slot 2

PIN	Row A SIGNAL	Row B SIGNAL	Row C SIGNAL
1	TXRTN0	RXD0	DATA0
2	TXD0	CTS0	DATA1
3	RTS0	DTR0	DATA2
4	DSR0	RXD1	GND
5	TXD1	CTS1	DATA3
6	RTS1	DTR1	DATA4
7	DSR1	RXD2	GND
8	TXD3	CTS2	DATA5
9	RTS2	DTR2	DATA6
10	DSR2	RXD3	DATA7
11	TXD3	CTS3	GND
12	RTS3	DTR3	DATA STROBE/DATA STROBE\
13	DSR3	RXD4	N.U/INPUT PRIME
14	TXD4	CTS4	TXRTN1
15	RTS4	DTR4	GND
16	DSR4	RXD5	D.D./ACKNOWLEDGE\
17	TXD5	CTS5	BUSY\
18	RTS5	DTR5	TXRTN2
19	DSR5	RXD6	GND
20	TXD6	CTS6	IFVALID/FAULT\
21	RTS6	DTR6	ON-LINE/SELECT
22	DSR6	RXD7	
23	1XD7	CTS7	GND
24	RTS7	DTR7	N.V./LP. CONT
25	DSR7	TXRTN5	
26	TXRTN3	TXRTN6	
27	TXRTN4	TXRTN7	
28	+5v	+5v	+5v
29	-5v	-5v	-5v
30	+12v	+12v	+12v
31	-12v	-12v	-12v
32	GND	GND	GND

Table 2-13. Winchester Disk Controller, Connector P2/J23, Slot 3

PIN	Row A SIGNAL	Row B SIGNAL	Row C SIGNAL
1	CB0		
2	CB2	CB1	
3	CB3		
4	CB5	CB4	GND
5	CB6		
6	DIRECTION\	CB7	
7	CWD0\		GND
8		CWD1\	
9	CYACK\	ATTN\	
10	INDEX\		
11		SECTOR\	GND
12	WRITE CLK\	SEEKEND\	
13	DRIVEFAULT\	WRITE-CLK	
14	SYSTEMCLK	ATTACK	
15	UNITACK0\	SYSTEMCLK\	GND
16	BI-DATA\	UNITACK1\	
17	UNITACK2\	BI-DATA	
18	READ-ENABLE\	UNITACK3\	
19	MASTER RESET\	READ ENABLE	GND
20	WRITE ENABLE	WRITE ENABLE\	
21			
22			
23			GND
24			
25			
26			
27			
28	+5v	+5v	+5v
29	-5v	-5v	-5v
30	+12v	+12v	+12v
31	-12v	-12v	-12v
32	GND	GND	GND

Table 2-14. Cartridge Tape Controller, Connector P2/J24, Slot 4, Pin Assignments

PIN	Row A SIGNAL	Row B SIGNAL	Row C SIGNAL
1		SLD\	GND
2		RDY\	GND
3		WND\	GND
4		FLG\	GND
5		LPS\	GND
6		FUP\	GND
7		BSY\	GND
8		EWS\	GND
9		RWD\	GND
10		REV\	GND
11		FWD\	GND
12		HSP\	GND
13		WEN\	GND
14		SL1\	GND
15		SL2\	GND
16		SL4\	GND
17		SLG\	GND
18		RNZ\	GND
19		RDS\	GND
20		DAD\	GND
21		WDE\	GND
22		WNZ\	GND
23		TR2\	GND
24		WDS\	GND
25		TR1\	GND
26			
27			
28	+5v	+5v	+5v
29	-5v	-5v	-5v
30	+12v	+12v	+12v
31	-12v	-12v	-12v
32	GND	GND	GND

Table 2-15. System 8000 Memory Bus, Connector P2/J26-30, Slots 6 through 10

PIN	Row A SIGNAL	Row B SIGNAL	Row C SIGNAL
1	MD38\	MD37\	
2	MD36\	MD35\	
3	MD34\	MD33\	
4	MD32\	MD31\	GND
5	MD30\	MD29\	
6	MD28\	MD27\	
7	MD26\	MD25\	GND
8	MD24\	MD23\	
9	MD22\	MD21\	
10	MD20\	MD19\	
11	MD18\	MD17\	GND
12	MD16\	MD15\	
13	MD14\	MD13\	
14	MD12\	MD11\	
15	MD10\	MD09\	GND
16	MD08\	MD07\	
17	MD06\	MD05\	
18	MD04\	MD03\	
19	MD02\	MD01\	GND
20	MD00\	MA16	
21	MA17	MA18	
22	MA19	MA20	READ\
23	MA21	MA22	GND
24	MA23	RC0\	RAS\
25	RC1\	RC2\	REF\a
26	RC3\	RC4\	CAS\
27	RC5\	RC6\	WRITE\
28	+5v	+5v	+5v
29	-5v	-5v	-5v
30	+12v	+12v	+12v
31	-12v	-12v	-12v
32	GND	GND	GND

SECTION 3

INSTALLATION

3.1 Introduction

This section provides information to enable easy installation of the System 8000. It contains receiving inspection, unpacking, system locations, interconnection, and turn-on information.

3.2 Receiving Inspection

The system has been pre-tested by factory specialists prior to being packed and installed in the shipping containers. However, during shipment mishandling may have occurred that will affect the operational capability of the system.

Visually inspect the exterior of the shipping carton. If visual inspection reveals the presence of unusual shipping carton damage, fill out a Shipping Damage Report for the shipping agent.

3.3 Unpacking

The system has been carefully packed in individual shipping containers with styrofoam retaining supports that absorb shock and prevent movement within the packing carton during normal shipping and handling procedures. During unpacking procedures, carefully remove and keep all styrofoam retaining supports and other packing materials as required to allow access to the system enclosure. Save all styrofoam supports and other packing material until the performance verification procedures have been completed. Proceed with the unpacking instructions as follows:

Unpacking Procedure

1. Place the carton in an upright position.

2. Remove styrofoam and other packing materials from each carton as required, to allow for removal of the contents.

3. Remove the system components from the shipping carton.

3.4 System 8000 Console Inspection

Visually inspect the system console for dents or evidence of mishandling. Make notations of any mishandling evidence in the Shipping Damage Report.

3.5 Internal Inspection

Unusual shipping and handling damage that is not obvious from packing carton or external cabinet visual inspections may have caused the printed circuit cards in the peripheral module to become displaced from connector sockets. If this condition is found, replace the PC cards in their appropriate slots as shown in Figure 2-3. The 256K Memory Array boards are inserted starting at slot ten (bottom) of the card cage, i.e., if one 256K Memory is included in the system, it is inserted in slot 10, if two 256K memories are included, they are inserted in slots 9 and 10. After all components have been inspected and any defective conditions have been corrected, proceed with the System 8000 assembly instructions beginning at paragraph 3.6.

The Peripheral Module is packed with foam packing material above, below, and on both sides of the module. This packing material should be removed before proceeding further. The disk and tape drives should not be powered up with the packing material in place because it will interfere with proper cooling of the module. The BASF Winchester disk drive is packed with the lockscrew for the linear voice-coil motor in the locked (IN) position as shown in Figure 3-2. If the lockscrew is in the unlocked (OUT) position, shipping damage has probably occurred. Record on the Shipping Data Report. Repackage for return to Zilog in the same carton.

CAUTION

The lockscrew should never be disengaged prior to assembly, nor before applying power to the drive.

3.6 Assembly Instructions

The basic system assembly consists of stacking and attaching the Processor and Peripheral Modules on the Storage Compartment base. The separate modular units are stacked vertically to form the System 8000 (Figure 3-3). A double Storage Compartment is shown for the Model 20.

The Model 30 consists of an additional Peripheral Module and a single Storage Compartment.

3.6.1 Assembly and Interconnecting

Remove front panels of the Processor and Peripheral Modules, and the Storage compartment by pulling the panels forward.

1. Locate on the top front corners of the storage compartment the two aluminum guide posts. At the bottom front corners of each of the modules are two U-shaped slots. These slots will slide into the guides and align a rear key hole when stacking each module. Remove the front panels from the processor and peripheral modules.

2. With the Peripheral Module placed approximately one inch behind the Storage Compartment guide posts, slide the module forward until alignment guides are fully inserted in slots. Lift the module slightly from the rear making sure the key is engaged.

3. Loosen thumb screws and open cable ducts at rear of the Peripheral Module.

4. Loosen thumb screws of clamps stored at rear of module and attach at top of Storage Compartment.

5. Repeat the above procedures and attach the processor or next module to the top of the Peripheral Module. (Model 30 includes two Peripheral Modules.)

6. Locate card cage at front of processor and ensure that PC boards are firmly mounted. This may be done by lifting board levers and reinserting boards.

7. Ensure that all packing material has been removed from the disk and tape drive units.

8. Replace front covers on modules.

9. Connect wider ribbon cable jumper to the tape connection on the Processor and Peripheral Module. The red stripe on the edge of the cable must be oriented on the right side looking from rear.

10. Connect the smaller ribbon cable and jumper between the disk connections with the red stripe to the right.

11. Install AC output power cable at rear of Processor Module and jumper to Periphral Module. In the case of the Model 30, install a second AC power cable between the two Peripheral Modules.

12. Install AC power cord at AC input of Processor Module.

13. Turn AC power with the power switch at the rear of the Processor Module (wait approximately 30 seconds).

14. Check cartridge tape slot for illumination to verify that power is applied to the disk and tape drive units.

15. Turn off AC power.

16. Connect the CRT interface cable to the CONSOLE port on back of Processor Module. If a printer is included with the system, connect the printer to the PRINTER port.

17. Replace cable duct and tighten thumb screws.

18. Unlock the voice-coil actuator on the disk drive through an access hole shown in Figure 3-5. Refer to Section 3.10 for voice-coil unlocking procedure.

3.6.2 Disassembly

During disassembly, and before removal of the Peripheral Module, power must be removed from the module and the voice-coil actuator lockscrew must be locked on the BASF Winchester disk drive. Refer to Section 3.10 and Figures 3-2 and 3-5.

Procedure:

1. Open cable ducts by removing thumb screws.

2. Remove interconnecting cables between the Processor and Peripheral Modules.

3. Remove top cover.

4. Remove the clamp screws attaching the Processor Module to the Peripheral Module and store.

5. Remove front covers from modules.

6. Remove Processor Module by lifting and sliding back from guideposts.

7. Apply AC power to the Peripheral Module and wait until the disks (disk drive) are rotating at nominal speed (at least 20 seconds).

CAUTION

Before the Peripheral Module can be dismounted, the BASF Winchester disk drive voice-coil actuator lockscrew must be in the locked position (refer to Figure 3-2).

8. If the system is using a rev. 8 BASF Winchester disk drive, proceed as follows:

   a. Loosen thumb screws attaching the module to the storage compartment.

   b. Elevate the front of the module approximately six (6) inches. This will cause the head-carriage assembly to move toward track 00 (outer diameter of the disk).

   c. Gently engage the voice-coil actuator lockscrew in a clockwise direction using light pressure with a small blade screwdriver.

   d. Remove AC power from the module and remove it from the Storage Compartment.

9. If the system is using a rev. 15 BASF Winchester disk drive, proceed as follows:

   a. Loosen the thumb screws attaching the module to the Storage Compartment.

   b. Remove the module front panel.

   c. Switch the head retract toggle switch on the Servo PCA. This will cause the head-carriage to move toward track 00 (outer diameter at the disk).

   d. Gently engage the voice-coil actuator lockscrew in a clockwise direction using light pressure with a small blade screwdriver.

e. Return the toggle switch to its normal operating position and remove AC power from the module.

f. Remove the module from the Storage Compartment.

10. If another Peripheral Module is used (Model 30), follow the same removal procedure.

Side Panels on Modules

Side panels are removed by removing four screws on each panel.

Front Panels on Modules

Front panels are secured by spring fasteners, and pull away from the module.

Base Plate

The Storage Compartment is attached by four screws which secure it to the base plate.

3.7 System Location

The choice of a location for the System 8000 should conform to the guide-lines described in paragraphs 3.7.1 through 3.7.4.

3.7.1 System Enclosure Location

The System 8000 contains internal air circulating fans that exhaust through ports located on the sides of the enclosure. To allow normal system ventilation, the enclosure sides should be at least three inches from an adjacent wall or other object that could interfere with proper exhaust air circulation.

3.7.2 Operating Temperature

The System 8000 has been designed to operate within an ambient temperature range of 50 to 104 degrees Fahrenheit (10 to 40 degrees Celsius).

3.7.3 Relative Humidity

Acceptable humidity conditions are from 0 to 80 percent non-condensing.

3.7.4 Access

Access to internal components is required periodically for maintenance. To allow access to the enclosure for the purpose of removal or replacement of peripherals or components, place the system in a location that will allow it to be easily rolled away from nearby walls or other objects.

3.8 Electrical Power Requirements

System input power requirements are nominally 117 volts alternating current (Vac), 60 Hertz at 5 amps (sustained) maximum or 220 Vac, 50 Hertz at 2.5 amps (sustained) maximum, single phase. Ranges of 105 to 129 Vac and 198 to 242 Vac respectively, are acceptable input power voltage ranges.

3.9 System Interconnection

As shipped from the factory, a typical System 8000 could consist of the freestanding, modular unit, CRT terminals, and high-speed printers. Additional peripherals can be integrated into the system as dictated by application requirements. Peripheral I/O connectors, located at the rear of the Processor Module, provide the means for the addition of peripherals. Identification of the I/O connectors is shown in Figure 3-4.

CAUTION

Interconnection of the terminals and the printer must be made with the 117 Vac power turned OFF.

The basic system consists only of the system enclosure. The system may also include CRT terminals and a printer. These are connected by means of the peripheral equipment connectors at the rear of the Processor Module.

The serial system components are interconnected with the standard RS232C cable included in the shipping carton.

3.9.1 Interconnect Procedure

1. Place all components at the location where they will be used.

2. Position the system units for easy access to the RS232C cable connectors located at the rear of each unit.

3. Attach one end of the RS232C cable to the CRT Terminal Connector and the other end of the cable to the CONSOLE system connector Identification as shown in Figure 3-4.

4. Move the system units to the position in which they will be used.

5. Insert the system input power cable into an available 117 Vac 60 hertz wall receptacle.

6. Insert the CRT Terminal power cable into one of the TTY connectors at the rear of the system, as shown in Figure 3-4.

3.10 Voice-Coil Actuator Unlocking Procedure

The following paragraphs cover the unlocking of the voice-coil actuator for rev. 8 and rev. 15 BASF Winchester Disk Drives. The actuator must be unlocked before the disk drive can become operational. Read the instructions thoroughly before performing the procedure.

NOTE

Use this procedure to unlock the voice-coil actuator on System 8000 Model 20 and Model 30 systems. The second Disk/Tape Module configured on Model 30 systems, contains only a BASF Winchester Disk Drive.

To determine the revision level of the disk drive, refer to Figure 3-1, and proceed as follows:

1. Remove the front panel from the Disk/Tape Module.

2. From the front of the disk drive, identify the Servo PWA (Printed Wiring Assembly).

3. If the Servo PWA contains a head retract toggle switch, the disk drive is a rev. 15; otherwise, the drive is a rev. 8.

The following steps required to unlock the voice-coil actuator are typical for both disk drive revision levels, Steps 4 and 5 are only applicable to rev. 15 drives.

1. Turn system power ON, and wait at least 20 seconds for the disk to reach nominal rotating speed.

NOTE

The voice-coil actuator lockscrew (Figure 3-2) must be in the locked position; if not, damage to the disk surface may have occurred during shipment.

CAUTION

The voice-coil actuator lockscrew should only be unlocked with the disk rotating at full speed.

2. Unlock the voice-coil actuator lockscrew. Access to the lockscrew is by an access hole at the rear of the Disk/Tape Module (refer to Figure 3-5).

   Rotate the lockscrew counter-clockwise, using light pressure with a small blade screwdriver. The lockscrew will spring out after approximately 10 to 15 rotations.

3. Check that the actuator is free of the rubber stop be gently pushing on the lockscrew. This moves the actuator toward the body of the disk. Some resistance should be felt due to the voice-coil magnet and the spring on the lockscrew.

CAUTION

A hard push or sudden jolt can send the actuator across the disks and into the opposite stop with the possibility of some damage.

STEPS 4 AND 5 ARE ONLY FOR REV. 15 DISK DRIVES

4. If the actuator can be moved away from the stop, but then returns to the outer stop, the head retract toggle switch on the Servo PWA is in the retract position.

5. Place the toggle switch in the normal position (toward the front of the drive). This in conjunction with unlocking the voice-coil actuator lock-screw (step 2), makes the drive operational.

6. Replace the module front panel.

7. Cycle the system power OFF/ON. This completes the procedure for rev. 8 and rev. 15 disk drives.

3.11 System Self-Test

The System 8000 has a built-in system self-test feature. The System Power-up Diagnostics (SPUD) verify the integrity of the system.

No special procedures are required to initiate the self-test feature; SPUD is automatically performed at system turn-on when the RESET and START control switches are pressed during a normal system power-up sequence. The self-test feature can also be initiated while in the PROM Monitor by typing "T <CR>". If degraded system operational capability is suspected, the self-test feature can be manually initiated by typing "T <CR>". Refer to Section 5.10 for a description of SPUD.

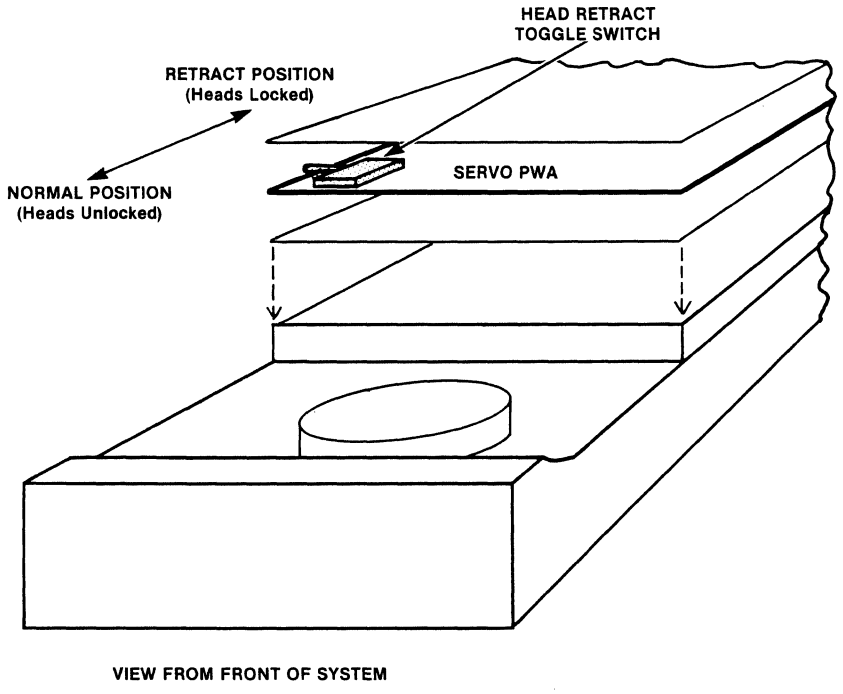


Figure 3-1. Location of Head Retract Toggle Switch

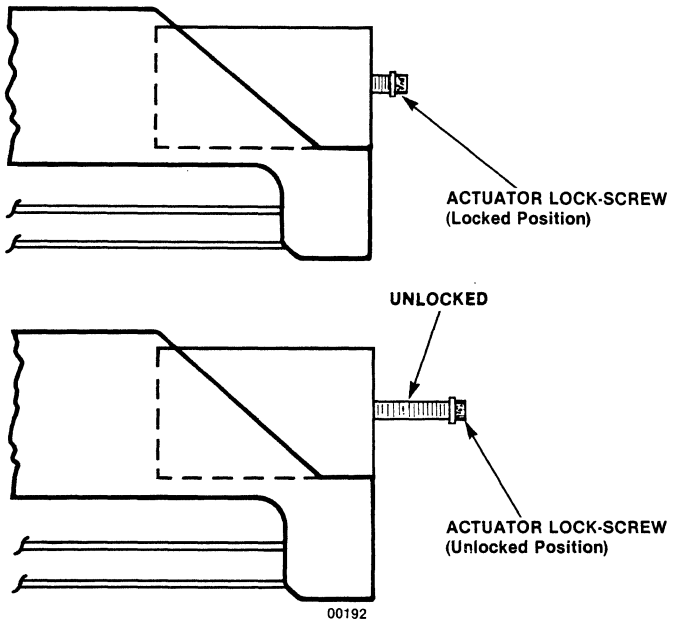


Figure 3-2. Linear Voice-Coil Actuator Lockscrew, BASF Drives

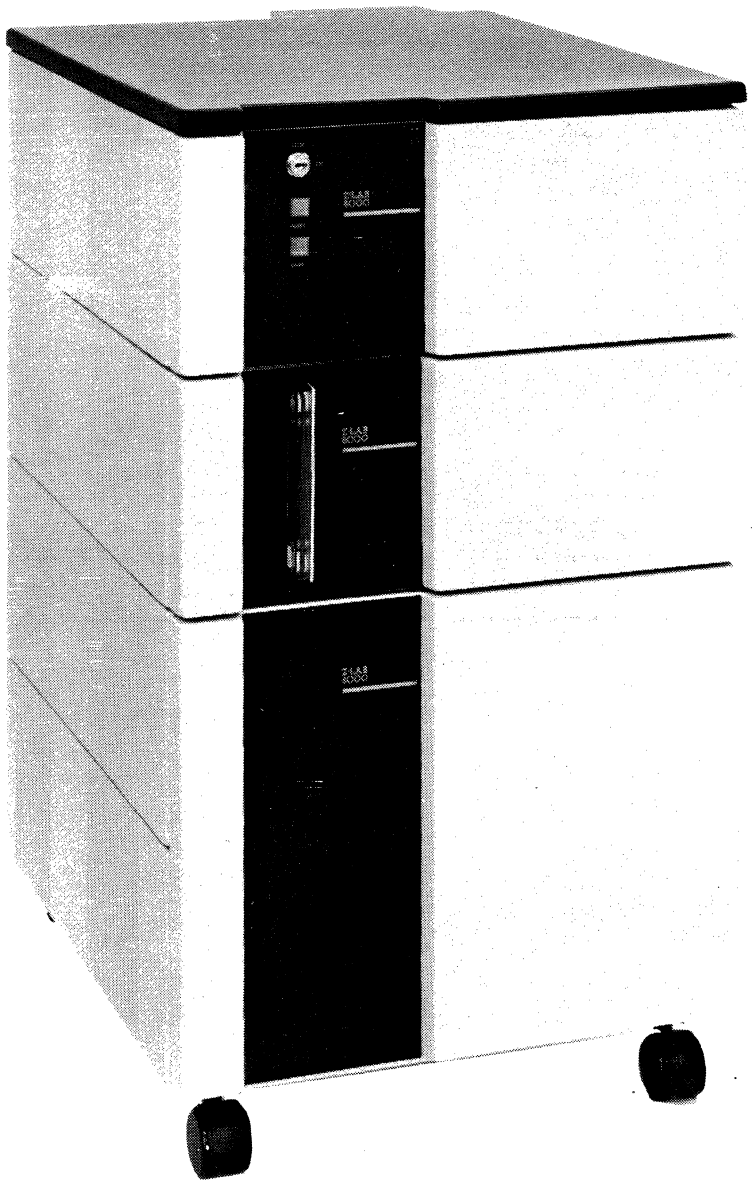


Figure 3-3. System 8000

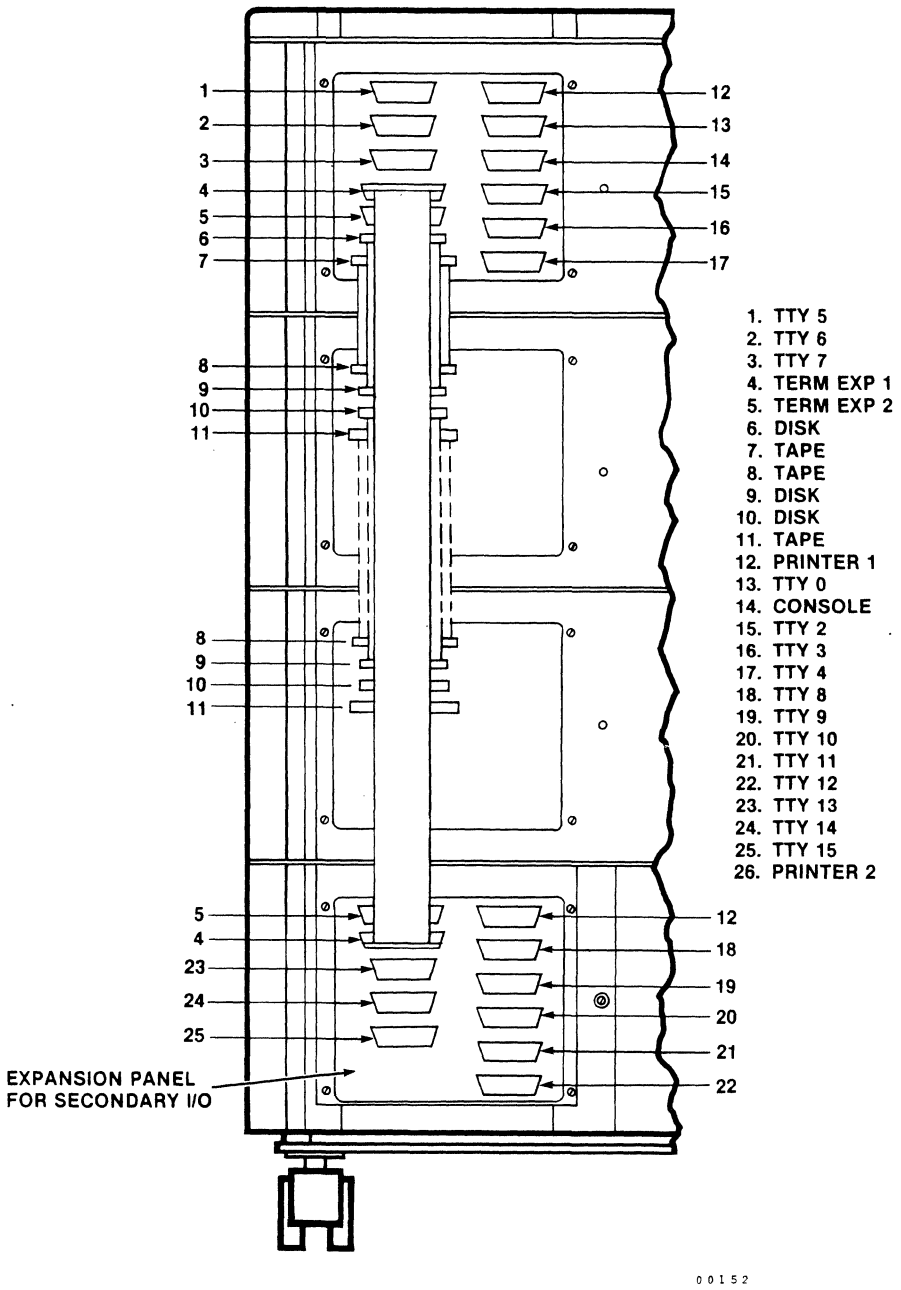


Figure 3-4. Connector Identification

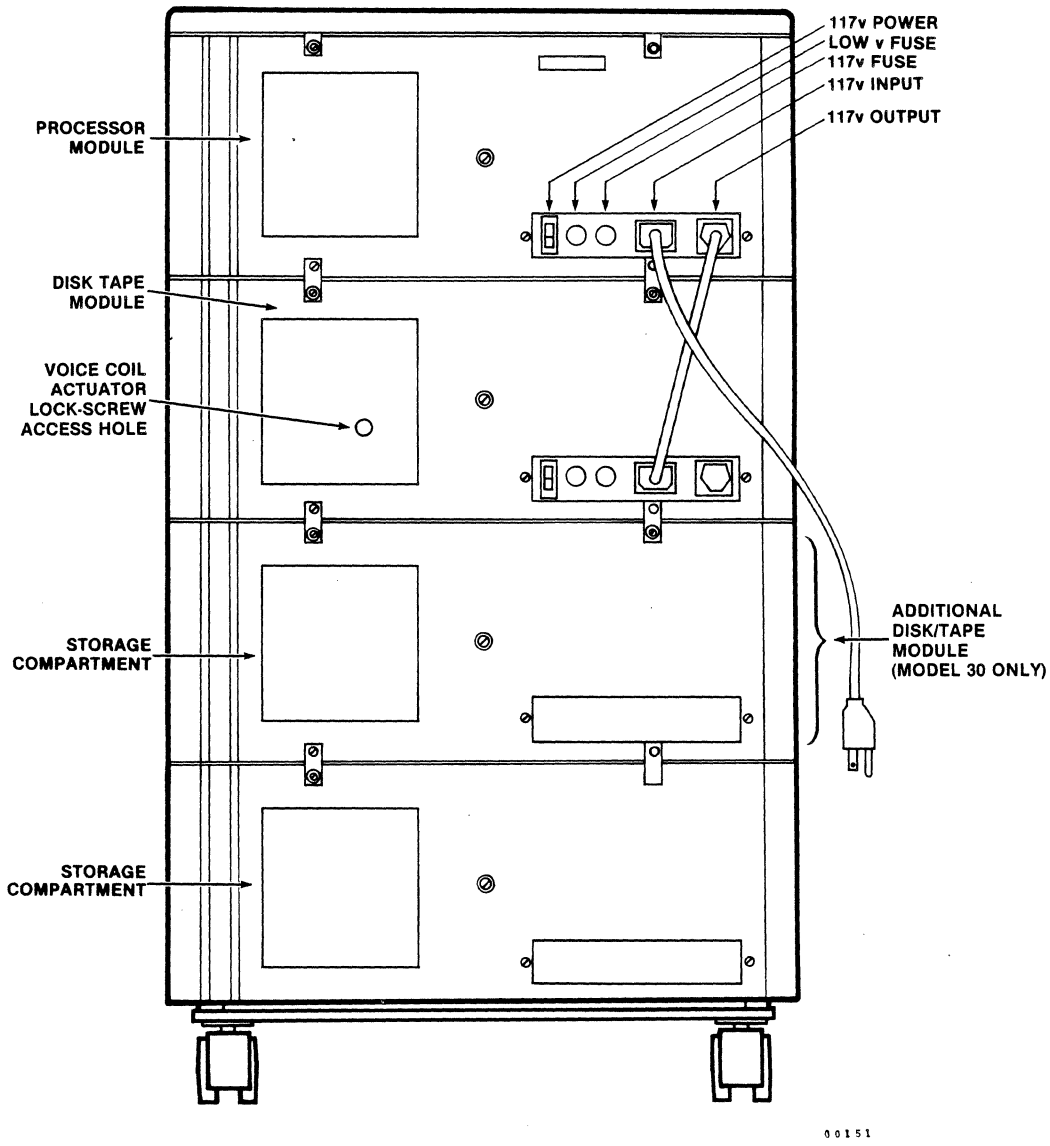


Figure 3-5. System 8000, Model 20, Rear View

SECTION 4

THEORY OF OPERATION

4.1 General

This section discusses the theory of operation of the System 8000 based on block diagrams of various levels of complexity. The discussion begins with the basic building blocks of the system, its bus, the modules on its bus, bus conventions, and, its I/O. The text goes more deeply into the operation of the system: interrupts, addressing, disk and tape operations, and others. For the most part, this discussion does not cover the internal operation of the system boards mentioned in the Preface of this manual.

4.2 System Bus

The block diagram in Figure 4-1 shows all the major elements of the System 8000 connected to the Z-Bus Backplane Interconnect (ZBI). The ZBI is the system bus over which all communication between the elements on the bus take place. On the backplane of card cage, the ZBI is connected to connectors J11 through J20. On the logic diagrams and assembly drawings for the printed circuit boards, the signal lines of the ZBI are connected to the P1 connectors.

The CPU is the bus controller, and any other element on the bus that needs to gain control of the bus must request control from the CPU.

4.3 Bus Conventions

Signals on the bus may be active in either a high or a low state. All signals that use the bus have names or mnemonics that identify them. These names also indicate the active state of the signal; for example, the signal AS\ (address strobe) is an active low signal because it has a back slash appended to it. If the signal appears as AS with no back-slash, it is active high. The back-slash is the same notation as the over bar that indicates the logical complement of a signal. Sometimes, active-low signals have a minus sign appended to them as in AS-. The over-bar, the back-slash, and the minus sign all mean the same thing. This text uses the back-slash; some of the drawings use either the over-bar or the minus sign (-).

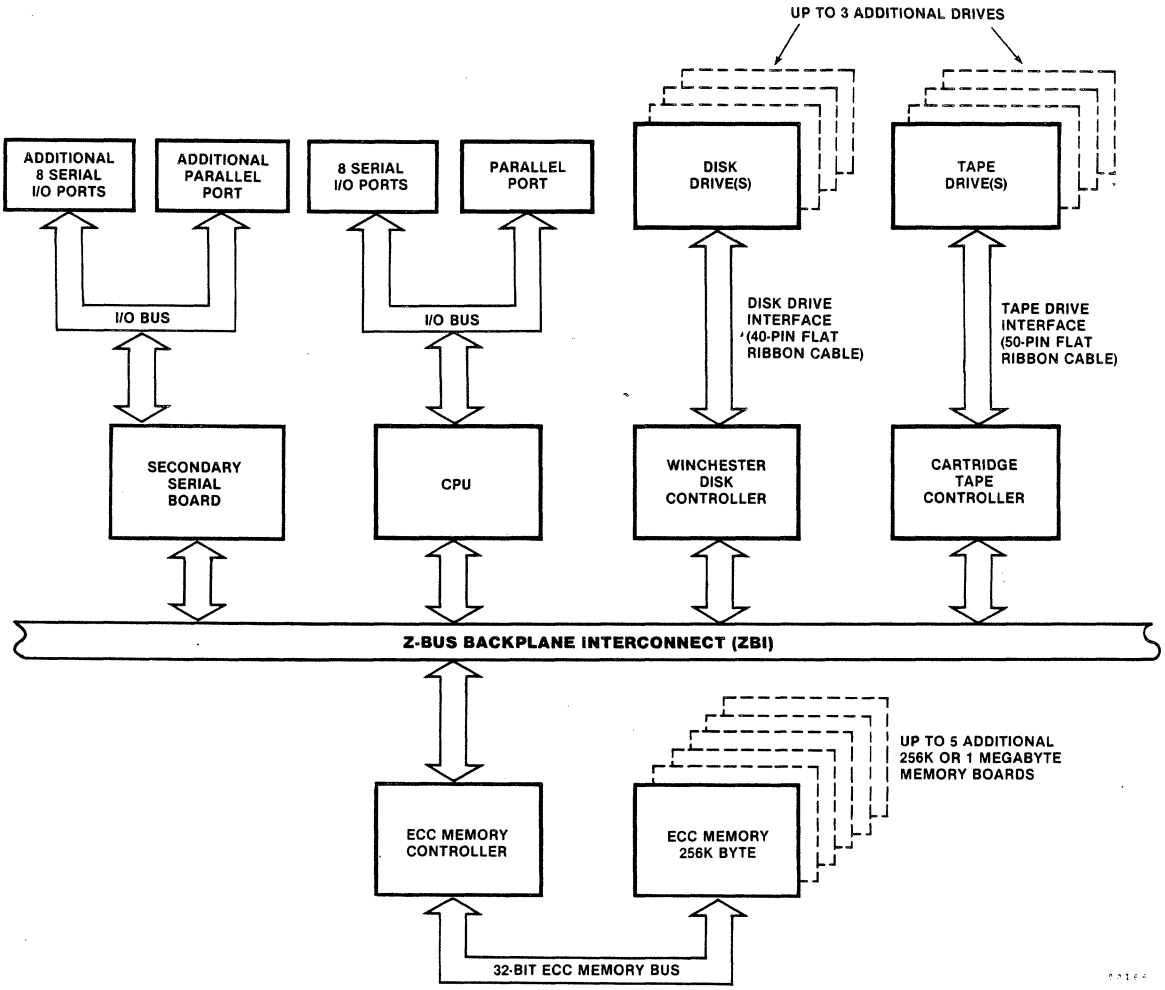


Figure 4-1. System 8000, Functional Relationships

4.4 Bus Signals

Table 4-1 lists all the ZBI signal lines and their definitions. One signal name can designate more than one signal line; in this case, the signal name is followed by numbers in angles brackets (<>) which indicate the quantity of lines and their designations. For example, the signal designated AD<31:0> is the name for the 32 address and data lines (AD0 through AD31) that are part of the bus. Within the angled brackets, the 31 indicates the most significant line.

The status lines, ST<4:0>, and the data width lines, B/W\ and W/LW\, form specific codes that cause a number of discrete operations to occur. Tables 4-2 and 4-3 list the various codes on the status and data width lines respectively, and the operations that the codes initiate.

Table 4-1. ZBI Bus Lines

SIGNAL NAME\ MNEMONIC	NUMBER OF LINES	FUNCTION
AD<31:0>	32	Multiplexed Address/Data lines: These lines are driven by the bus master. The address strobe (AS\) and data strobe (DS\) determine when the information on these lines is valid.
ME\	1	Memory Error: During a memory access, if the memory controller detects an uncorrectable error, the controller sends the ME\ signal to the bus master.
ST<4:0>	5	Status Lines: These active-high lines indicate the type of transaction currently occurring on the bus. (See Table 4-2 for the various codings and their associated transactions.)

Table 4-1. ZBI Bus Lines (continued)

SIGNAL NAME\ MNEMONIC	NUMBER OF LINES	FUNCTION
R/W\	1	Read-Write: This is a dual-purpose line. When this line is high, it indicates that the current operation is a read operation; when the line is low, the operation is a write operation.
N/S\	1	Normal-System: Indicates the mode of operation of the master that is currently controlling the bus.
B/W\	1	Byte-Word Select: This signal is used with signal W/LW\ (listed below) to define the data access width. (See Table 4-3 for coding.)
W/LW\	1	Word/Long-Word Select: This signal is used with signal B/W\ (listed above) to define the data access width. (See Table 4-3 for coding.)
AS\	1	Address Strobe: The bus master drives this line low to initiate a bus transaction. The rising trailing edge indicates that the current address and status are valid.
DS\	1	Data Strobe: The bus master uses this signal to time the movement of data to and from itself along the data bus.

Table 4-1. ZBI Bus Lines (continued)

SIGNAL NAME\ MNEMONIC	NUMBER OF LINES	FUNCTION
WAIT\	1	Wait: By forcing this line low, a bus slave causes the bus master to suspend operation while the slave completes its activity.
STOP\	1	Stop Line: This line, when driven low by an EPU device causes a Z8000 CPU to generate null transactions. During a null transaction, the data strobe (DS\) remains high.
BAI\	1	Bus Acknowledge In: This signal along with BAO\ forms the bus priority chain.
BAO\	1	Bus Acknowledge Out: This signal along with signal BAI\ forms the bus priority chain.
BUSREQ\	1	Bus Request: A module uses the BUSREQ\ signal to gain access to the bus. This signal is part of the priority scheme that is set up by the connection of signals BAI\ and BAO\.
CAI\	1	CPU Acknowledge In: Not currently implemented-- the CAI, CAO, CPUREQ, CAVAIL signals are designed to allow multiple CPUs to share a single bus. They may be used on a future S8000 product and should be considered reserved.
CAO\	1	CPU Acknowledge Out: Not currently implemented-- reserved

Table 4-1. ZBI Bus Lines (continued)

Signal Name\ Mnemonic	Number of Lines	Function
⌣PUREQ\	1	CPU Request: Not currently implemented--reserved
CAVAIL	1	CPU Available: Not currently implemented--reserved
INT1\	1	Level-1 Interrupt: This interrupt line has the highest priority in the system. This line when driven by a slave generates a non-maskable interrupt (NMI).
INT2\	1	Level-2 Interrupt: This interrupt line has the second to the highest priority in the system. This line, when driven by a slave, generates a vectored interrupt.
INT3\	1	Level-3 Interrupt: This interrupt line has the lowest priority in the system. This line, when driven by a slave, generates a non-vectored interrupt.
IEI1	1	Level-1 Interrupt Enable In: This signal works with Level-1 interrupt enable out to form the NMI acknowledge daisy chain.
IEO1	1	Level-1 Interrupt Enable Out:
IEI2	1	Level-2 Interrupt Enable In: This signal works with Level-2 interrupt enable out to form the VI acknowledge daisy chain.

Table 4-1. ZBI Bus Lines (continued)

SIGNAL NAME\ MNEMONIC	NUMBER OF LINES	FUNCTION
IEO2	1	Level-2 Interrupt Enable Out:
IEI3	1	Level-3 Interrupt Enable In: This signal works with Level-3 interrupt enable out to form the NVI acknowledge daisy chain.
IEO3	1	Level-3 Interrupt Enable Out:
MMREQ\	1	Multimicro Request: When this signal is active a module can request the use of a common resource. The MMREQ\ signal works with signals MMAI\ and MMAO\.
MMAI\	1	Multimicro Acknowledge In: This signal works with signal MMAO\ to form the resource-request daisy chain.
MMAO\	1	Multimicro Acknowledge Out: This signal works with signal MMAI\ to form the resource-request daisy chain.
PWRBAD\	1	Power Bad: The processor power supply generates this as an early warning to the system that the DC power will soon disappear.
MCLK	1	Master Clock: This signal is the system clock and is the foundation for all timing in the system. The frequency of the MCLK signal is four times (4X) that of the bus clock (BCLK).

Table 4-1. ZBI Bus Lines (continued)

SIGNAL NAME\ MNEMONIC	NUMBER OF LINES	FUNCTION
BCLK	1	Bus Clock: The system derives this clock from the master clock (MCLK). The BCLK is one fourth the frequency of the master clock and synchronizes the operation of the elements in the system that require synchronization. All bus transfers are synchronized to this clock. The system CPU board is the generator of this clock and MCLK above.
RESET\	1	Reset: This is the master reset signal for the entire system. This signal is generated by the front panel master reset switch or upon power-up by the power-up reset circuit. When it is forced low, it initializes the entire system.

Table 4-2. ZBI Status Lines, Transaction Coding

S4	S3	S2	S1	S0	TRANSACTION
0	0	0	0	0	Internal operation
0	0	0	0	1	Memory refresh
0	0	0	1	0	I/O reference
0	0	0	1	1	Special I/O reference
0	0	1	0	0	Segment trap acknowledge
0	0	1	0	1	INT 1 Interrupt acknowledge
0	0	1	1	0	INT 3 Interrupt acknowledge
0	0	1	1	1	INT 2 Interrupt acknowledge
0	1	0	0	0	Data memory request
0	1	0	0	1	Stack memory request
0	1	0	1	0	Transfer between data memory and an EPU
0	1	0	1	1	Transfer between stack memory and an EPU
0	1	1	0	0	Program reference, nth cycle
0	1	1	0	1	Program reference, 1st cycle
0	1	1	1	0	Transfer between CPU and EPU
0	1	1	1	1	Reserved
1	X	X	X	X	Reserved

Table 4-3. Data Width Codes: Byte, Word, and Long Word

B/W\	W/LW	DATA WIDTH
1	1	Sets the data width to byte width, 8 bits, data on lines AD<7:0>
0	1	Sets data width to word size, 16 bits, data on lines AD<15:0>
1	0	Sets data width to double-word size, 32 bits, data on lines AD<31:0>
0	0	Reserved

4.5 Bus Modules

The bus modules are the major blocks that communicate directly with the CPU over the bus. The one exception is the ECC Memory Array module whose communications path to the bus is through the Memory Subsystem Controller. The following paragraphs deal more closely with the individual modules.

4.5.1 CPU Module

The CPU module is the bus controller, sometimes called the host, which initiates and controls transactions on the bus. Also as shown in Figure 4-2, the CPU connects directly to and controls the I/O bus. All transactions with the outside world pass through either the parallel port or one of the eight serial I/O ports. The I/O lines from the CPU module pass through mating connectors P2 and J21, located on the CPU module and backplane, respectively. Table 4-4 lists the lines on the CPU I/O bus and their definitions.

Table 4-4. CPU I/O Bus, Signal Definitions

SIGNAL NAME	DEFINITION
TXD7 to TXD0	Transmit Data, 8 bits
RXD7 to RXD0	Receive Data, 8 bits
CTS7 to CTS0	Clear to Send
DTR7 to DTS0	Data Terminal Ready
RTS7 to RTS0	Request to Send
DCD7 to DCD0	Data Carrier Detect
RXRTN7 to RXRTN0	Receive Return
DATA7 to DATA0	PIO Data
DATA STROBE	Data Products Data Strobe
DATA STROBE\	Centronics Data Strobe, Active low
DATA DEMAND/ ACKNOWLEDGE\	Demand (Data Products) when high Acknowledge (Centronics) when low
BUSY\	Printer Busy
IFVALID	Interface valid (Data Products)
FAULT\	Paper empty indication (Centronics)
ON-LINE	Online (Data Products)
F.P. BUSACK INDICATOR (Front Panel)	DMA in process Disk or tape controller in control of bus
F.P. POWER-ON INDICATOR (Gnd) (Front Panel)	Ground for power-on indicator
F.P. POWER-ON INDICATOR V+ (Front Panel)	Indicates system is on
F.P. NORMAL INDICATOR (Front Panel)	CPU running user process

Table 4-4. CPU I/O Bus, Signal Definitions (continued)

SIGNAL NAME	DEFINITION
SWITCH N.C. START (Front Panel)	Auto boot
SWITCH N.O. START (Front Panel)	Auto boot
SWITCH RESET (Front Panel)	Resets system

Hardware jumpers on the CPU board can be set for either a segmented or nonsegmented mode of instruction execution by the Z8001 CPU. Other jumpers can be set for either a Centronics or Data Products printer interface. Refer to Table 4-5 for possible jumper configurations.

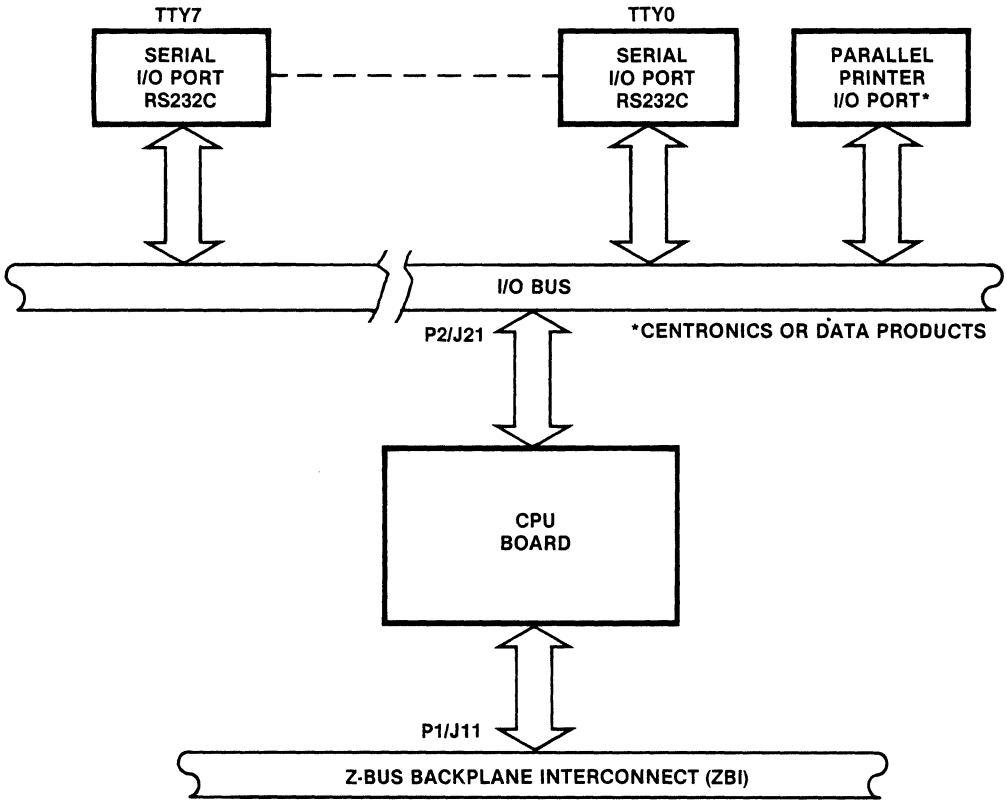


Figure 4-2. CPU Board, Functional Relationships

Table 4-5. CPU Board Jumper Selection

The System 8000 CPU board is currently configured for only non-segmented operating systems. A segmented mode will be available on future configurations. Jumpers E1 through E12 (refer to Figure 1-2) determine the operating mode. Jumper selections are as follows:

NON-SEGMENTED	SEGMENTED
E2 to E3	E1 to E2
E4 to E5	E6 to E5
E7 to E8	E8 to E9
E11 to E12	E10 to E12

NOTE

These jumpers are properly set by the manufacturer and must not be changed by the user. Segmented jumper selections are shown for reference only.

For either a Data Products or Centronics interface, connect Jumpers E13 through E18 as follows:

CENTRONICS INTERFACE	DATA PRODUCTS INTERFACE
E13 to E14	E14 to E15
E17 to E18	E16 to E17

4.5.1.1 I/O Bus. The CPU board I/O bus connects the System 8000 with the outside world. All devices that control both the serial and parallel I/O are on the CPU board; functionally, these devices form eight serial I/O channels and two parallel I/O channels. The serial channels support the RS-232C standard and the parallel ports can be configured for either the Data Products or Centronics standard. One parallel port, Port B of a Z80-PIO, is the data-out port; port A handles status and control information. Table 4-6 lists the control signals for the parallel printers. Table 4-7 lists the status signals for the printers, and Table 4-8 lists the data output of port B.

Table 4-6. Parallel Printer Output Control Signals, Port A

BIT NUMBER	CENTRONICS	DATA PRODUCTS
0	Data strobe\	Data strobe
1	not used	not used
2	not used	not used
3	not used	not used

Table 4-7. Parallel Printer Input Status Signals, Port A

BIT NUMBER	CENTRONICS	DATA PRODUCTS
4	Busy\	Busy\
5	Select	Online
6	Fault\	Interface Valid
7	Acknowledge\	Data Demand

Table 4-8. Parallel Printer Data, Port B

BIT NUMBER	CENTRONICS	DATA PRODUCTS
Bit 0	Data Bit 0	Data Bit 0
Bit 1	Data Bit 1	Data Bit 1
Bit 2	Data Bit 2	Data Bit 2
Bit 3	Data Bit 3	Data Bit 3
Bit 4	Data Bit 4	Data Bit 4
Bit 5	Data Bit 5	Data Bit 5
Bit 6	Data Bit 6	Data Bit 6
Bit 7	Data Bit 7	Data Bit 7

4.5.1.2 Serial I/O. The serial I/O comprises 4 Z80B-SIO/2 devices. Each device has two channels; Table 4-9 lists the devices and their assigned channels.

Table 4-9. Serial I/O Devices and Channel Assignments

DEVICE NUMBER	CHANNEL ASSIGNMENT
SIO 0	Channels 0 and 1
SIO 1	Channels 2 and 3
SIO 2	Channels 4 and 5
SIO 3	Channels 6 and 7

Each channel of the serial I/O connects to its own baud-rate generator. These generators are channels in Z80B-CTC devices. The SIO channels and their corresponding baud-rate generators are listed in Table 4-10. The baud rate clock comes from an independent baud-rate oscillator. The frequency of the baud-rate clock is 1.2288 megahertz.

Table 4-10. Serial Channels and Baud-rate Generators

SIO	SIO CHANNELS	BAUD NO.	CTC NO.	CTC CHANNEL
0	0	0	0	0
0	1	1	0	1
1	2	2	0	2
1	3	3	1	0
2	4	4	1	1
2	5	5	1	2
3	6	6	2	0
3	7	7	2	1

Serial channel 1 (console), with one exception, is like every other serial channel. The exception is that the on-board monitor on the CPU board uses channel 1 to communicate with the system operator when the system is turned on. The initial baud rate for channel 1 can be set to one of the four values listed in Table 4-11, using switch U70 on the CPU board. These settings permit the use of a variety of terminals as the system console. After the system has been booted, the console baud rate can also be changed under software control.

Table 4-11. Baud Rate Settings, Switch U70

SWITCH SELECTION (ON = 0)		BAUD RATE
1	4	
0	0	300
0	1	1200
1	0	9600
1	1	19200

System software can access the I/O channels using standard Z8000 I/O instructions. Table 4-12 lists the I/O addresses of the I/O channels.

Table 4-12. I/O Channels and Their Addresses

I/O ADDRESS	I/O DEVICE AND CHANNEL
FF81	SIO 0, channel 0, data
FF83	SIO 0, channel 1, data
FF85	SIO 0, channel 0, control
FF87	SIO 0, channel 1, control
FF89	SIO 1, channel 2, data
FF8B	SIO 1, channel 3, data
FF8D	SIO 1, channel 2, control
FF8F	SIO 1, channel 3, control
FF91	SIO 2, channel 4, data
FF93	SIO 2, channel 5, data
FF95	SIO 2, channel 4, control
FF97	SIO 2, channel 5, control
FF99	SIO 3, channel 6, data
FF9B	SIO 3, channel 7, data
FF9D	SIO 3, channel 6, control
FF9F	SIO 3, channel 7, control
FFA1	CTC 0, channel 0 (baud 0 for SIO 0, channel 0)
FFA3	CTC 0, channel 1 (baud 1 for SIO 1, channel 1)
FFA5	CTC 0, channel 2 (baud 2 for SIO 1, channel 2)
FFA7	CTC 0, channel 3
FFA9	CTC 1, channel 0 (baud 3 for SIO 1, channel 3)
FFAB	CTC 1, channel 1 (baud 4 for SIO 2, channel 4)
FFAD	CTC 1, channel 2 (baud 5 for SIO 2, channel 5)
FFAF	CTC 1, channel 3
FFB1	CTC 2, channel 0 (baud 6 for SIO 3, channel 6)
FFB3	CTC 2, channel 1 (baud 7 for SIO 3, channel 7)
FFB5	CTC 2, channel 2
FFB7	CTC 2, channel 3
FFB9	PIO 0, channel A, data
FFBD	PIO 0, channel A, control
FFBB	PIO 0, channel B, data
FFBF	PIO 0, channel B, control

4.5.2 Winchester Disk Controller

The Disk Controller controls the fully buffered transfer of data between the CPU (host) and a selected disk drive. The block diagram in Figure 4-3 shows the relationship between the controller and both the ZBI and the disk drives. All transactions between the controller and the host pass through connectors P1 and J1 and over the ZBI. Transactions between the controller and a selected disk drive pass through connectors P2 and J2, the drive bus. The signals on connector P1 are the standard ZBI signals; the signals on P2 are common to only slot three on the back plane. Table 4-13 lists the interface signals between the disk controller and the disk drives.

4.5.2.1 CPU Interface. The CPU communicates with the controller through 16 8-bit command registers and an 8-bit command-status (C/S) register (Figure 4-4). Each register has a specific command assignment and a specific address. The CPU writes commands into command registers xx00 through xx0F; the controller reads these registers and performs the specified commands. The controller places the results or status of the specified command in the C/S register which the CPU reads. Table 4-14 lists the command and C/S registers.

The command and C/S registers reside in the CPU's I/O space on any 256-word boundary. Within this 256-word block, the command registers are at relative addresses xx00 to xx0F hexadecimal and the C/S register at address xx10. The two most significant hex values of the address, xx, can be set by jumpers on the controller board. Table 4-15 lists the jumpers and the bits (15 through 8) that the jumpers control. This scheme allows more than one controller within the same I/O space.

Figure 4-5 shows a segment of I/O space, containing three 256-word blocks. The two most significant hexadecimal nibbles (AA, BB, and CC) of the addresses can be set by using the jumpers listed in Table 4-15. For example, in the address AA00, if AA is to equal FF hex, then no jumper is connected in any of the jumper groups, and all the lines are high. When a jumper is connected in any jumper group, the jumper shorts its associated line to ground, a low level.

The controller can accommodate either 2716 or 2732 EPROMs. Jumpers on the controller board permit the selection of both the type of EPROM and any necessary wait states. Table 4-16 lists the jumpers for memory selections.

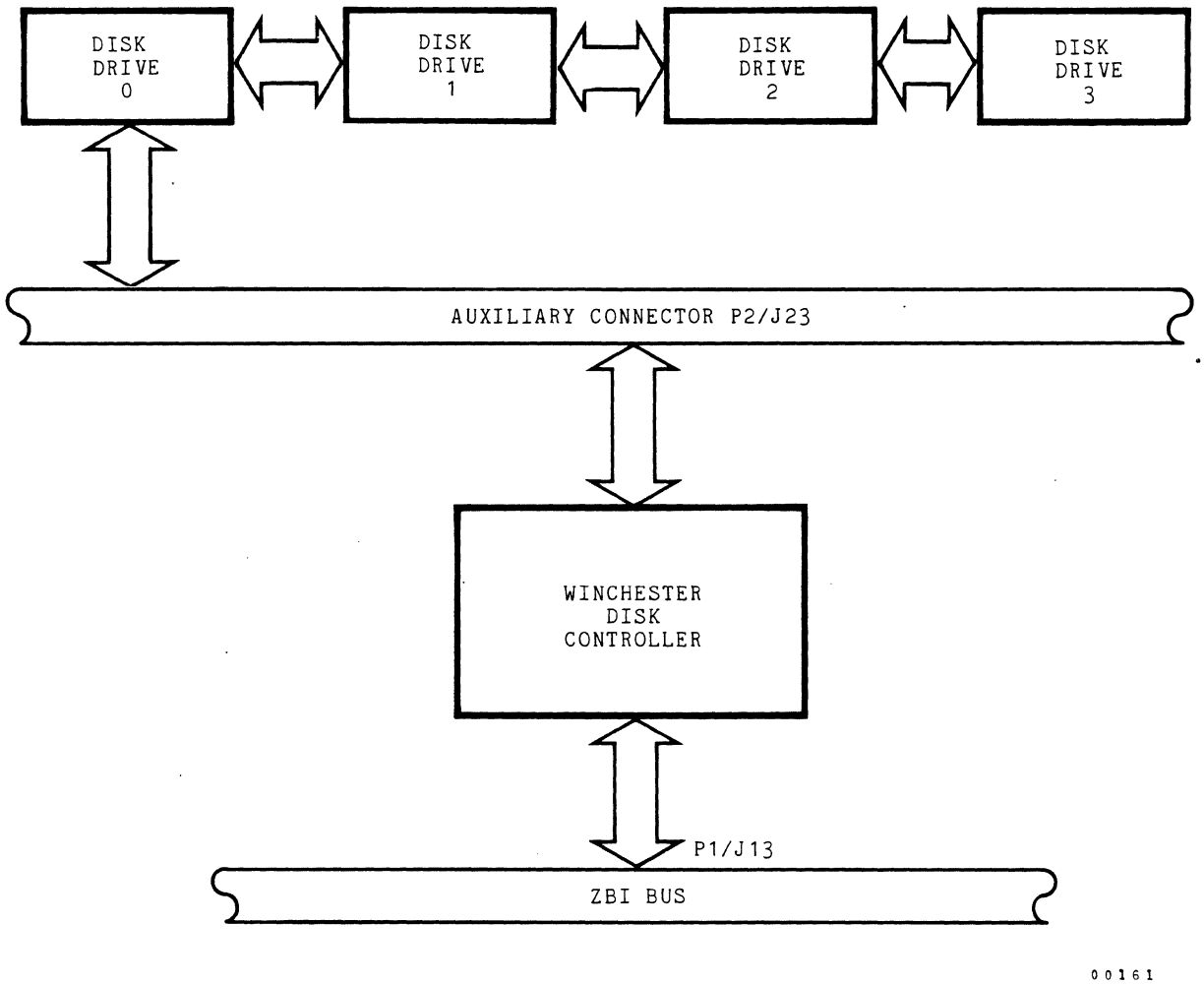


Figure 4-3. Winchester Disk Controller, Functional Relationships

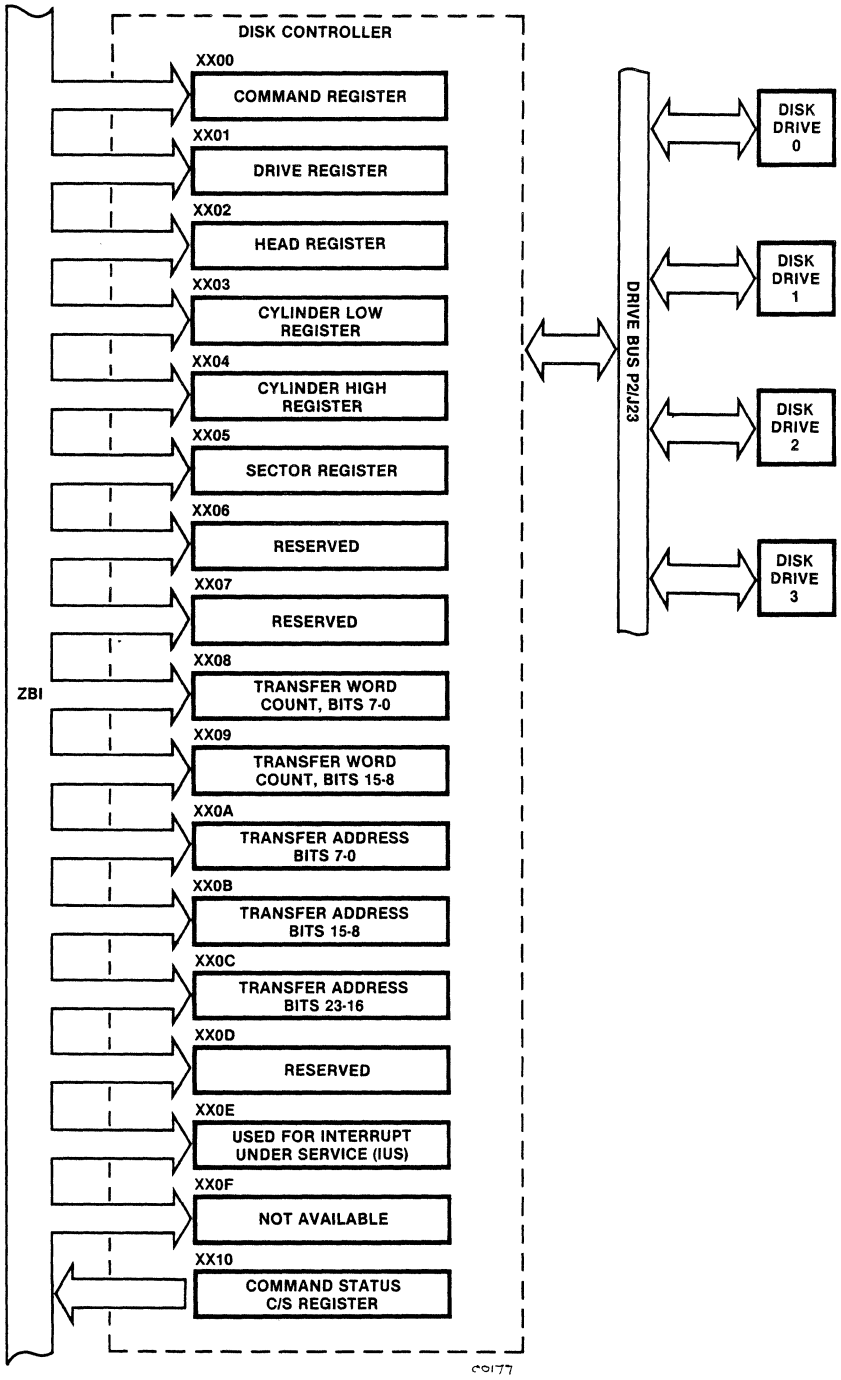


Figure 4-4. Disk Controller Command and Command Status (C/S) Registers

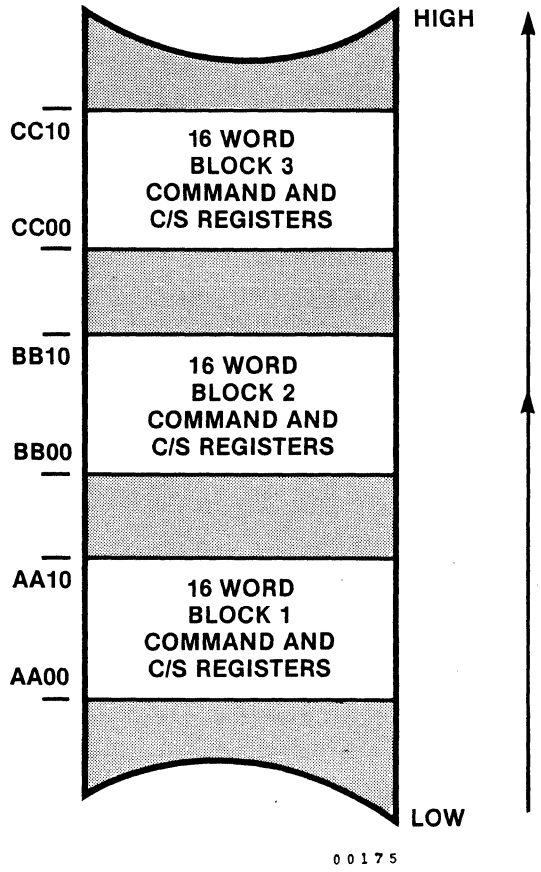


Figure 4-5. Disk Controller, I/O Space

4.5.2.2 Controller/Drive Interface. Communications between the disk controller and a selected disk drive travel over the P2/J23 mating connectors. Table 4-13 lists and defines the signals on these lines. The disk controller can control either BASF 6171,6172, or 6173 drives or IMI 7710, 7720, or 7740 drives. However, all drives attached to any controller must be of one type. Table 4-17 lists the jumpers that must be set to select a particular type of drive. Table 4-18 lists the codes for the command and status words on the control bus between the disk controller and the disk drive.

Table 4-13. Disk Controller and Disk Drive, Interface Signals, Connector P2/J23, Slot 3

SIGNAL NAME	FUNCTION
Control Bus CB7 to CB0	These eight signal lines are bidirectional. Control signals from the Controller to the selected drive are transmitted on these lines. The DIRECTION\ signal controls the direction transfer.
DIRECTION\	This signal determines the direction of transfer over the control bus (CB7 through CB0). When DIRECTION\ is high, the controller reads status information from one of the four status registers. When this line is low the controller is sending commands to the selected drive.
Control Words 0 and 1, CWD0 and CWD1	These signals identify one of four bytes that can be on the control bus. The DIRECTION\ signal identifies the current byte as a command or status byte.
Attention, ATTN\	The controller sends this signal to all drives to initiate handshaking.
Cycle Acknowledge CYAK\	Each drive generates the CYAK\ signal in response to the ATTACK signal.
Attention Acknowledge, ATTACK	The selected drive generates this active-high signal in response to the ATTN\ signal nal from the controller. ATTACK then causes the selected drive to generate the CYACK signal.
INDEX\	The selected drive generates an index pulse for each revolution of the disk. Each cylinder produces a pulse which is 2.5 microseconds long and which occurs every 16.67 milliseconds. Sector 00 immediately follows the pulse.
SECTOR\	The selected drive generates this 2.5 microseconds signal.
SEEK END\	The selected drive generates this signal at the end of a seek operation.

Table 4-13. Disk Controller and Disk Drive, Interface Signals, Connector P2/J23, Slot 3 (continued)

SIGNAL NAME	FUNCTION
DRIVE FAULT\	The selected drive generates this signal if there was a drive fault.
Unit Acknowledge 0 to 3 UNITACK0\ to UNITACK3\	The selected drive places its binary address on these lines.
READ ENABLE\	The controller generates this signal to read the selected drive at the current cylinder. Before activating this signal the controller must send a control word 2 and then wait until the selected drive returns the CYACK signal.
WRITE ENABLE\	The controller generates this signal to write to the selected drive at the current cylinder. If the heads are write-protected, a fault results. The controller must send a control word 2 and then wait until the selected drive retunrs the CYACK signal.
MASTER RESET\	A high-to-low transistion on this line resets all internal latches and output ports.
BI-DATA\	These lines form a bidirectional, differential pair that transmits NRZI read data from the currently selected drive to the controller whenever READ ENABLE\ is active.
SYSTEM CLK\	A pre-recorded pattern on the surface of one of the disks on the drive generates this signal. The selected drive sends this signal to the controller: the controller derives the write clock (WRITE CLK\) from system CLLX. The controller sends the WRITE CLK\ to the selected drive.
WRITE CLK\	This is a differential signal which the controller derives from the SYSTEM CLK\.

Table 4-14. Command and Command-Status Registers

REGISTER ADDRESS	NAME	DEFINITION
xx00	Command Register	The CPU sends specific commands to the controller through this register. This must be the last register the CPU writes to, for it clears the status byte to enable handshaking between the CPU and the controller and begins the operation.
xx01	Unit Register	The CPU uses this register to specify which unit (drive) it wants to participate in I/O activity. One of four drives (0, 1, 2, or 3) can be specified.
xx02	Head Register	The CPU uses this register to specify the read-write head it wants for the current operation. The 8 megabyte BASF drive 6171 has 1 head:0. The 24 megabyte BASF drive 6172 has 3 heads: 0, 1, 2. The 40 megabyte BASF drive 6173 has 5 heads: 0, 1, 2, 3, 4.
xx03	Cylinder Address Low Register	The CPU stores the low-order 8 bits (7 to 0) of the 16-bit cylinder address in this register.
xx04	Cylinder Address High Register	The CPU stores the high-order 8 bits (15 to 8) of the cylinder address in this register. With the current disk drives, only bits 8 and 9 are used.
xx05	Sector Register	The CPU stores in this register the number of the sector it wants to read from or write to. With with the BASF drive, the sector numbers range from 0 to 23.

Table 4-14. Command and Command-Status Registers (continued)

REGISTER ADDRESS	NAME	DEFINITION
xx06	Reserved	For expansion
xx07	Reserved	For expansion
xx08	Transfer Word Count Bits 7 to 0	This register contains the loworder byte of the transfer word count.
xx09	Transfer Word Count Bits 15 to 8	This register contains the highorder byte of the transfer word count.
xx0A	Transfer Address Bits 7 to 0	This register contains the loworder byte of the 3-byte transfer address. The transfer address is the location of the first word of a block of memory allocated for the transfer. Data can be leaving memory or coming to it. The command from the CPU determines the direction of transfer. The Read Sector command moves data from disk to main memory. The Write Sector command moves data from main memory to disk.
xx0B	Transfer Address, Bits 15 to 8	The CPU stores the intermediate byte of the transfer address in this register.
xx0C	Transfer Address	The CPU stores the high-order byte of the transfer address in this register.
xx0D	Reserved	For transfer address bits 32 to 24.
xx0E	Reserved	
xx0F	Reserved	
xx10	Command-Status (C/S) Register	

Table 4-15. Jumper Settings for Address of Command and C/S Registers

JUMPERS	PURPOSE	CONNECTION AND RESULT
E10,E18	Causes bit 15 of address to be either a high or low level	E10 to E18: bit 15 low Open: bit 15 high
E11,E19	Causes bit 14 of address to be either a high or low level	E11 to E19: bit 14 low Open: bit 14 high
E12,E20	Causes bit 13 of address to be either a high or low level	E12 to E20: bit 13 low Open: bit 13 high
E13,E21	Causes bit 12 of address to be either a high or low level	E13 to E21: bit 12 low Open: bit 12 high
E14,E22	Causes bit 11 of address to be either a high or low level	E14 to E22: bit 11 low Open: bit 11 high
E15,E23	Causes bit 10 of address to be either a high or low level	E15 to E23: bit 10 low Open: bit 10 high
E16,E24	Causes bit 09 of address to be either a high or low level	E16 to E24: bit 09 low Open: bit 09 high
E17,E25	Causes bit 08 of address to be either a high or low level	E17 to E25: bit 08 low Open: bit 08 high

Table 4-16. Disk Controller Jumper Settings for Memory

JUMPER GROUP (E)	PURPOSE	CONNECTION AND RESULT
E1,E2,E3 E41,E42,E43	Permit inserting one wait state whenever any on-board memory is accessed or whenever only on-board EPROMs are accessed.	E2 to E3 and E42 to E43: Inserts one wait state during access of any on-board memory. E1 to E3 and E42 to E43: Inserts one wait state during access of EPROM only. E41 to E43: No wait states for controller on-board RAM or EPROM.
E4,E5,E6 E7,E8,E9	Permit selecting either 2716 or 2732 EPROMs.	E4 to E6 and E7 to E9: Use 2716 EPROMs. E5 to E6 and E8 to E9: Use 2732 EPROMs.

Table 4-17. Disk Controller Jumper Settings for Selection of BASF or IMI disk Drive

JUMPER GROUP	PURPOSE	CONNECTION AND RESULT
	NOTE Jumper groups E26 through E40 permit the use of either BASF or IMI disk drives.	
E26,E27,E28		E26 to E28: Select BASF drive E27 to E28: Select IMI drive
E29,E30,E31		E29 to E31: Select BASF drive E30 to E31: Select IMI drive
E32,E33,E34		E33 to E34: Select BASF drive E32 to E34: Select IMI drive
E35,E36,E37		E36 to E37: Select BASF drive E35 to E37: Select IMI drive
E38,E39,E40		E38 to E40: Select BASF drive E39 to E40: Select IMI drive

4.5.2.3. Command and Status Words. The disk drive control bus contains a command word when the direction line is activated, otherwise it contains a status word. The number of the command or status words is determined by the coding of CWD0 and CWD1.

Command Word 0--Command Word 0 is used to select one of fifteen drives by a four bit unit address (uA).

Command Word 1--Command Word 1 is used, together with the two low-order bits of Command Word 0, to establish the binary address of the desired cylinder specified by the Cylinder Address Register (CAR).

Command Word 2--Command Word 2 has three basic purposes: (1) select forward or reverse offset, (2) select early or late data strobe, and (3) select one of three possible head positions by the HAR0 and HAR1 bit positions.

Command Word 3--Command Word 3 enters a Diagnostic mode, Customer Engineering (CE) mode, Rezero (Return the heads to cylinder zero), clears a fault, or establishes which heads are to be write protected.

Status Words--The control bus contains a status word when the direction line is deactivated for four possible standard status words. Status Word 0--Status Word 0 communicates eight specific error conditions to the Controller-Formatter when a fault condition exists.

Status Word 1--Status Word 1 signals seven different error/exception conditions to the Controller-Formatter.

Status Words 2 and 3--Together contain the contents of the Position Address Register (PAR) for current locations of the heads.

Table 4-18. Disk Command and Status Words

		DIR	CWD1	CWD0	(MSB) Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	(LSB) Bit 0
COMMAND	WORD 0	1	0	0	UNIT ADDR3	UNIT ADDR2	UNIT ADDR1	UNIT ADDR0	SPARE 0	SPARE 0	CAR9	CAR8
	WORD 1	1	0	1	CAR7	CAR6	CAR5	CAR4	CAR3	CAR2	CAR1	CAR0
	WORD 2	1	1	0	SERVO OFFSET REV	SERVO OFFSET FWD	STROBE LATE	STROBE EARLY	SPARE 0	SPARE 0	HAR1	HAR0
	WORD 3	1	1	1	DIAG MODE	CE MODE	REZERO	FAULT CLEAR	SPARE	SPARE	EXT PROT1	EXT PROT0
STATUS	WORD 0	0	0	0	NOT READY	SERVO ERROR	R/W FAULT	SPEED ERROR	PWR LOSS	WRITE PROTD	SEEKING RZRNG	NOT ON CYL
	WORD 1	0	0	1	GUARD BAND	PLO ERROR	UNSAFE	INVAL CMND	TIME OUT	POR/ MR	SPARE	ILL ADDR
	WORD 2	0	1	0	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	PAR9	PAR8
	WORD 3	0	1	1	PAR7	PAR6	PAR5	PAR4	PAR3	PAR2	PAR1	PAR0

4.5.3 Cartridge Tape Controller

The tape controller is the intelligent interface between the System 8000 CPU and the tape drives (also called decks). The controller derives its intelligence from its on-board Z80B microprocessor. Figure 4-6 shows the basic relationship between the controller and both the System 8000 bus (ZBI) and the tape decks. Information flows between the controller and the CPU over the ZBI (mating connectors P1/J14). The flow of information between the controller and a selected tape drive is through mating connectors P2/J24 of slot 4 only of the system backplane. Figure 4-7 shows the allocation of the I/O space of the controller.

4.5.3.1 ZBI Interface. The controller and the host communicate through eight 16-bit (word) read/write registers. These registers appear in the controller's I/O space at addresses 40H through 4EH. (The H stands for hexadecimal.) Table 4-19 lists these registers and their assignments. On-board jumpers provide a means of changing the ZBI address of the controller board. These jumpers are listed in Table 4-20. The bit assignments of the upper byte of the interrupt vector are listed in Table 4-21. The commands that the host sends to the controller are listed in Table 4-22, and Table 4-23 defines the bits in the status register. Table 4-24 lists the bits in the Master Interrupt Control (MIC) register. All of the possible error conditions are listed in Appendix F.

4.5.3.2 Drive Interface. The tape controller sends commands to the tape drive to control its operation. These commands set the drive address, track address, and motion controls. Table 4-25 lists the commands that the controller sends to the drive. The drive responds to the controller by sending information back to the controller. Table 4-26 lists the information that the drive sends to the controller.

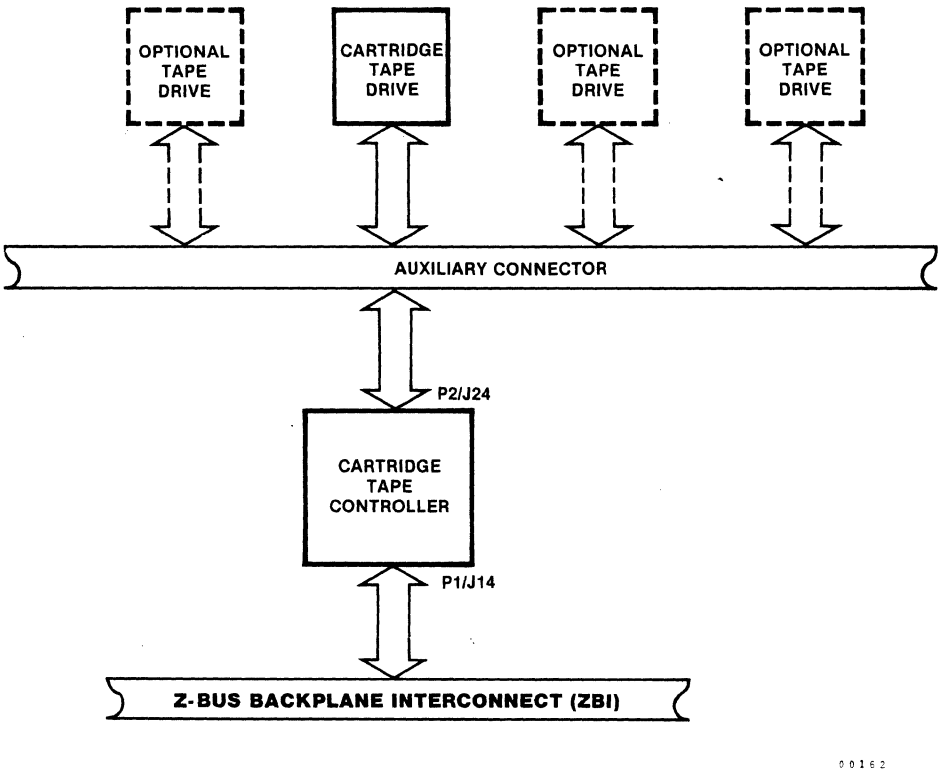


Figure 4-6. Cartridge Tape Controller Functional Relationships

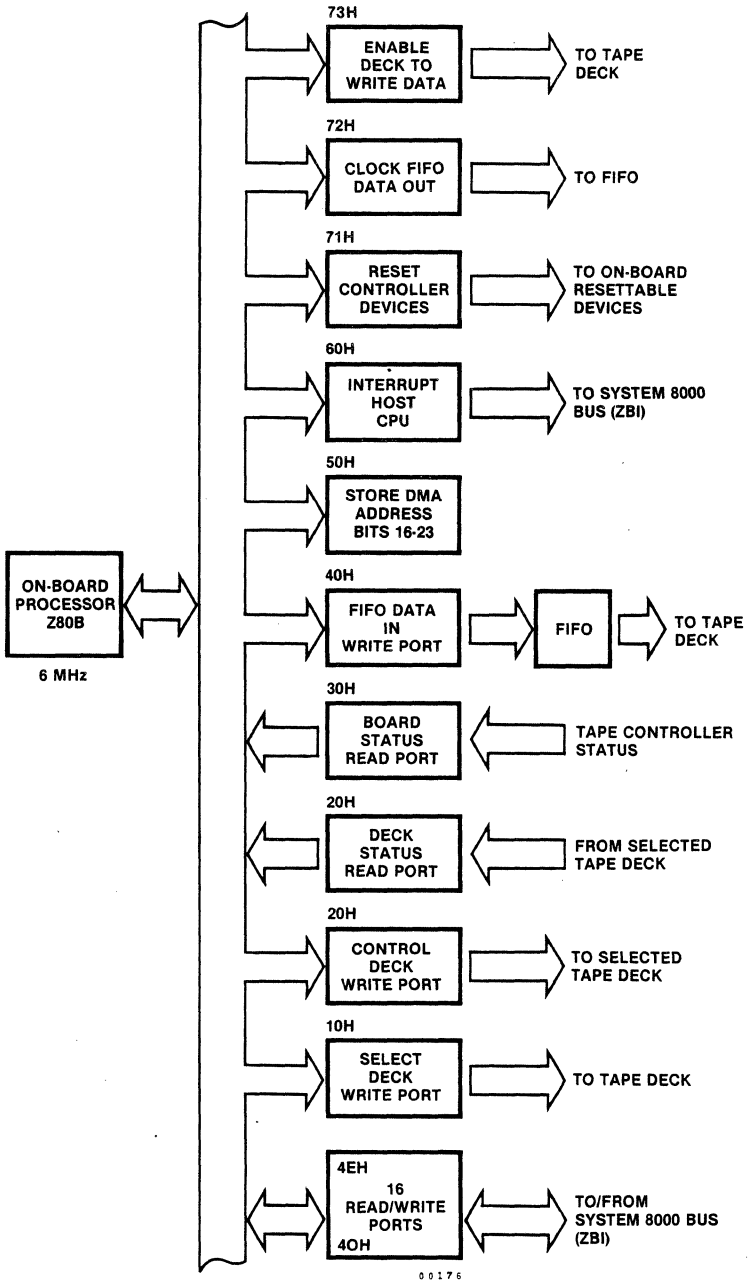


Figure 4-7. Cartridge Tape Controller I/O Address Space

Table 4-19. ZBI Tape Controller Interface Registers

ADDRESS	REGISTER	DESCRIPTION
40H	Interrupt Vector	The low-order byte contains the interrupt vector that host CPU writes to the controller. The high-order byte contains status information that the controller sends to the host.
42H	Command	The host sends commands to this register. The controller accepts only valid commands.
44H	Low DMA Start Address	The host sends the low word of the DMA starting address in this register. Bit 0 of this byte must be a 0 so that the address starts on a word boundary.
46H	High DMA Start Address	This register contains the high-order byte (bits 16 to 23) of the DMA start address.
48H	DMA Length	This register contains the length of the DMA transfer. This value must be less than 32 kilobytes (1k=1024).
4AH	Status	The controller stores information about the tape drive and controller in this register. The host reads this information. Another table defines the bits.

Table 4-19. ZBI Tape Controller Interface Registers (continued)

ADDRESS	REGISTER	DESCRIPTION
4CH	Status 1	Bits 0 to 3 define the number of retries for a read or write command. Bits 8 to 15 define the number of blocks or files that have been skipped during a skip command.
4EH	Interrupt Control	This is the master interrupt control register. Another table lists the bit definitions.

Table 4-20. Tape Controller Jumper Selection for Base Address

JUMPER GROUP (E)	PURPOSE	CONNECTION AND RESULT
E1, E2, E3	Set to expect either a low or high bit SAD15	E1 to E2 (normal): bit SAD15 low E2 to E3: bit SAD15 high
E4, E5, E6	Set to expect either a low or high bit SAD14	E4 to E5 (normal): bit SAD14 low E5 to E6: bit SAD14 high
E7, E8, E9	Set to expect either a low or high bit SAD13	E7 to E8 (normal): bit SAD13 low E8 to E9: bit SAD13 high
E10,E11,E12	Set to expect either a low or high bit SAD12	E10 to E11 (normal): bit SAD12 low E11 to E12: bit SAD12 high
E13,E14,E15	Set to expect either a low or high bit SAD11	E13 to E14 (normal): bit SAD11 low E14 to E15: bit SAD11 high

Table 4-20. Tape Controller Jumper Selection for Base Address (continued)

JUMPER GROUP (E)	PURPOSE	CONNECTION AND RESULT
E16,E17,E18	Set to expect either a low or high bit SAD10	E16 to E17 (normal): bit SAD10 low E17 to E18: bit SAD10 high
E19,E20,E21	Set to expect either a low or high bit SAD09	E19 to E20 (normal): bit 09 low E20 to E21: bit 09 high
E22,E23,E24	Set to expect either a low or high bit SAD08	E22 to E23 (normal): bit 08 low E23 to E24: bit 08 high
E25,E26,E27	Set to expect either a low or high bit SAD07	E25 to E26 (normal): bit 07 low E26 to E27: bit 07 high
E28,E29,E30	Set to expect either a low or high bit SAD06	E28 to E29: bit 06 low E29 to E30 (normal): bit 06 high
E31,E32,E33	Set to expect either a low or high bit SAD05	E31 to E32 (normal): bit 05 low E32 to E33: bit 05 high
E34,E35,E36	Set to expect either a low or high bit SAD04	E34 to E35 (normal): bit SAD04 low E35 to E36: bit SAD04 high
E37,E38,E39	Set to enable or disable the Controller board	E37 to E38 (normal): enables board to receive address from ZBI bus E38 to E39: disenable board

NOTE

The normal connection is wired on the board. Changing the normal connection to a new base address requires the cutting of traces and adding jumpers.

Table 4-21. Tape Interrupt Vector, Bit Definitions

BIT	NAME	MEANING
Bit 8	INTV	The current operation requires invention.
Bit 9	BUSY	The controller is busy executing the last command.
Bit 10	CMDREJ	The controller rejects the current command.
Bit 11	DATERR	An uncorrectable data error has occurred.
Bit 12	SKNDNE	The current skip operation has not been completed.
Bit 13	OVERFL	A buffer overflow has occurred.
Bit 14	FFERR	A FIFO error has occurred.
Bit 15	Not Used	

Table 4-22. Host-Tape Controller Commands

CODE (HEX)	NAME	DEFINITION
0000	NOP	The controller loops while waiting for a command from the host.
0001	READ	The controller reads one block. If necessary, controller backspaces and retries.
0002	WRITE	The controller writes one block. If necessary the controller back-spaces, erases three inches of tape and retries.
0003	REWIND	Controller rewinds tape to its logical beginning (6 inches past the load point).
nn04	SKBF	Controller skips nn blocks forward (nn is any value from 0 to 255).
nn05	SKBR	Controller skips nn blocks in reverse (nn=0 to 255).
nn06	SKFF	Controller skips nn files forward (nn= 0 to 255). A file is a group of blocks followed by a file mark.
nn07	SKFR	Controller skips nn files in reverse (nn=0 to 255).
0008	WFM	Controller writes file mark on tape.
0009	LOAD	Controller moves the tape from the phyical load point to logical load point: beginning of tape, track 0 selected.
000A	UNLD	Controller moves the tape to the physical load point.

Table 4-22. Host-Tape Controller Commands (continued)

CODE (HEX)	NAME	DEFINITION
0n0B	SEL	Controller selects a new drive: address is n, a value from 0 to 3.
0n0C	MRTRY	This sets the maximum number of retries the controller is allowed for reads and writes. At power on, the default is 10 retries: n=0 to 15.
0n0E	STRK	Controller rewinds the tape and selects new track: n=0 to 3.
0n0F	MODE	Controller changes to mode n (n=0,1). In mode 1, tape is divided into four separate tracks. The logical beginning of tape is at the beginning of each track. Logical end of tape is at the end of each track. REWIND moves the tape to the start of each track; skips do not carry from track to track. In mode 0, tape is one long track. Logical beginning of tape is at the start of track 0 and the logical end of tape is at the end of track 3. REWIND moves the tape to the start of track 0; skips carry from track to track. At power on, the default is mode 0.
0010	DIAG	The controller executes a diagnostic test, checking the ROM, the FIFO, and the host interface ports.

Table 4-23. Status Register, Bit Definitions

BIT	NAME	DEFINITION
Bit 0	NOTAP	No tape cartridge in drive
Bit 1	FMDET	File mark detected during read or or skip blocks
Bit 2	HWERR	Hardware error
Bit 3	INVAL	Invalid command
Bit 4	INAP	Inappropriate command
Bit 5	(Not Used)	
Bit 6	BPARM	Bad DMA parameters
Bit 7	BLKTAP	Blank tape
Bit 8	PROT	Tape cartridge write protected
Bit 9	LBOT	Tape at logical beginning of tape
Bit 10	LEOT	Tape at logical end of tape
Bit 11	RTRYAT	One or more retries attempted
Bit 12	UNIT0	Tape drive address bit 0
Bit 13	UNIT1	Tape drive address bit 1
Bit 14	TRK0	Track address bit 0
Bit 15	TRK1	Track address bit 1

Table 4-24. Master Interrupt Control Register, Bit Definitions

BIT	NAME	DEFINITION
Bit 0	MIE	Master interrupt enabled
Bit 1	IE	Interrupt enabled
Bit 2	DLC	Disable lower chain
Bit 3		Not defined
Bit 4		Not defined
Bit 5		Not defined
Bit 6	IUS	Interrupt under service
Bit 7	IP	Interrupt pending

Table 4-25. Tape Controller to Drive Interface Signals

SIGNAL	DEFINITION
RWD\	Rewinds the tape
REV\	Moves the tape backwards
FWD\	Moves the tape forward
WEN\	Enables writing and erasing on the tape
WDE\	Enables sending of write-data strobes and the writing of data on the tape
WNZ\	Serial data to be written to the tape drive
TR1\,TR2\	Select tracks during read, write, and erase track operation, according to following code:

Track Number	TR2\	TR1\
0	High	Low
1	Low	High
2	Low	Low
3	High	High

SLG\	Allows selection of tape drive designated by unit select codes: SL4\, SL2\, SL1\.
SL4\,SL2\,SL1\	These form the unit (drive) select code listed below:

Drive Selected	SL4\	SL2\	SL1\
0	H	H	L
1	H	L	H
2	H	L	L
3	L	H	H

4.5.4 Tape Controller Operation

To start a tape operation, the host CPU reads the high-order byte of the controller's interrupt vector to see if BUSY is set. If the busy bit is set, the controller is still executing the last command. If the controller is not busy, the host initializes the interface registers and then writes a non-zero command in the controller's command register. The flowchart in Figure 4-8 shows the steps taken by the host.

The controller normally loops while it waits for a non-zero command from the host. When the controller receives a command, it resets the interrupt-pending bit (IP) in the master interrupt control register. The controller sets the busy bit in the upper byte of the interrupt vector register to inform the host that it is busy with the current command. However, before the controller processes the command, it checks the validity of the command.

After processing the command, the controller resets the command register and sets the IP bit. Next, the controller sends an interrupt to the host and waits for an acknowledgement, the controller sends its interrupt vector in response to the acknowledgement. An upper byte of zero means that no errors occurred. The controller also sets the interrupt-under-service bit (IUS).

Table 4-26. Tape Drive to Controller Interface Signals

SIGNAL	DEFINITION
SLD\	Selected drive informs controller that the drive has received its unit address.
RDY\	Tape cartridge is installed.
WND\	Selected drive has received a write enable signal.
FLG\	Rewind completed.
LPS\	Load point sensed.
FUP\	Installed tape cartridge is unprotected.
BSY\	The drive is doing one of the following: 1) Automatic rewind after cartridge is installed. 2) Executing rewind, forward, or reverse command.
EWS\	The upper early warning hole in forward direction has been reached.
WDS\	The drive is examining the state of WNZ\ signal.

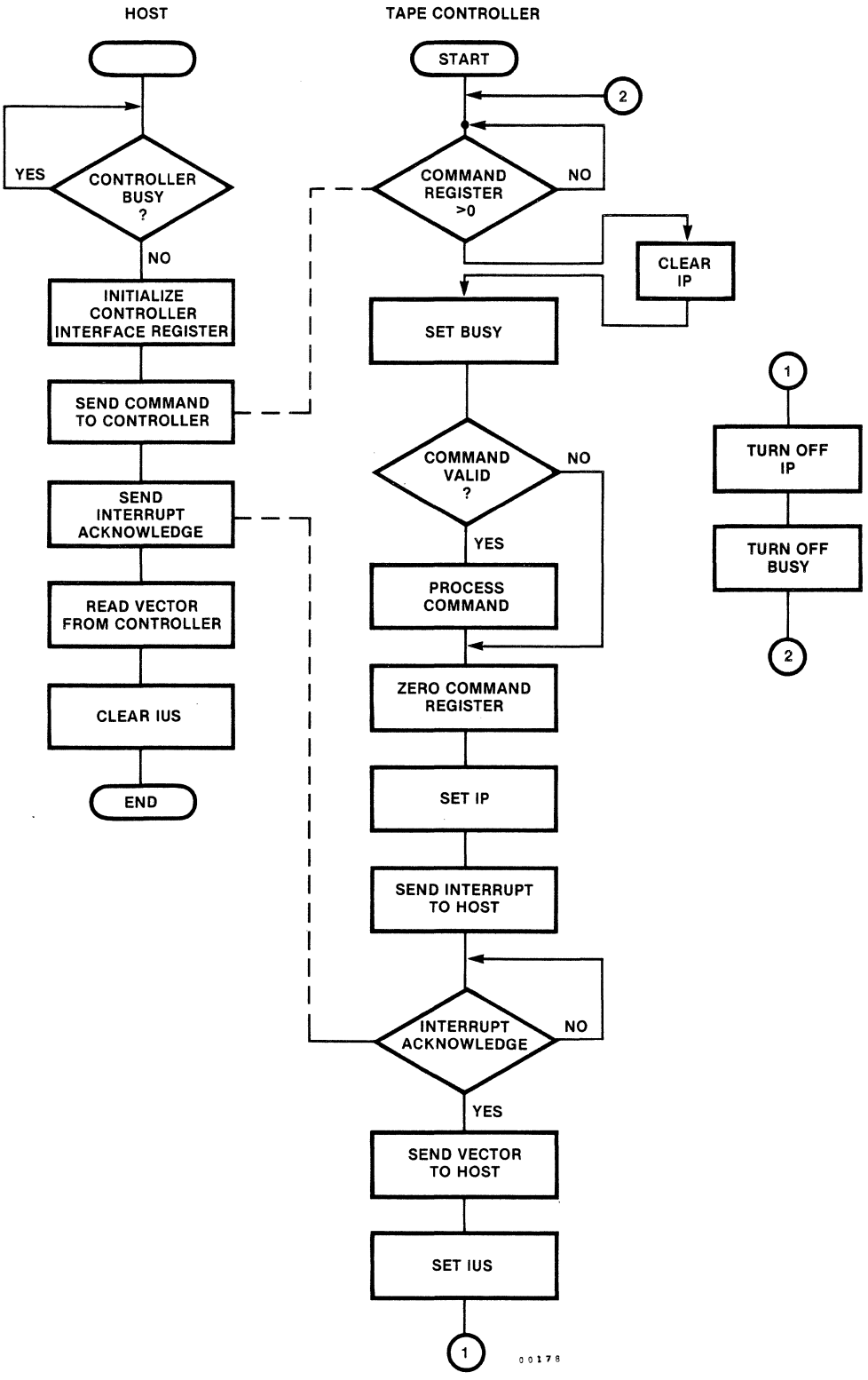


Figure 4-8. Cartridge Tape Controller, Command Processing

The host reads the vector from the controller and clears the controller's IUS bit. This action ends the interrupt subroutine for the host. The controller clears both the IP and the BUSY bits and loops while waiting for a new command.

4.5.5 Memory Subsystem Controller

The Memory Subsystem controller controls up to 16 megabytes of dynamic read-write memory. The controller can perform read-write operations with byte (8-bit), word (16-bit), and long-word (32-bit) quantities. Figure 4-9 shows the functional relationship between the controller and the ZBI and the memory modules that it controls.

The controller stores data as 32-bit long words and adds to this, seven bits of information for use by the error-detection and correction circuits. Figure 4-10 shows the overall organization of memory. During memory transactions, the controller accepts a 24-bit address and the B/W\ and W/LW\ control signals over the ZBI. The two least-significant address bits and the B/W and W/LW\ signals select one of the four bytes at the location to be read or modified.

4.5.5.1 Byte Translation. During transactions involving bytes, the controller receives a data byte from ZBI lines AD0 through AD7. Figure 4-11 shows the flow from a register, through the controller, and to memory. The controller places byte A, the first byte, in location 0 (bits 31 to 24). For this transaction, the bus controller sets both the B/W\ and W/LW\ control lines high to identify the current transfer as a byte transfer. The bus controller also places low levels (0) on ZBI address lines AD00 and AD01 to tell the memory controller to place byte A in bit positions 31 to 24.

Next, the memory controller places byte B, C, and D in the succeeding memory locations to fill up the current double word of memory. The memory controller places the next, the fifth, byte in the first location of the next double word of memory, bits 31 through 24 of byte 4 (not shown).

4.5.5.2 Word Translation. For word (16-bit) translations, the bus controller sets line B/W\ low and W/LW\ high. This code tells the memory controller that the current transfer is a 16-bit transfer. The bus controller places the 24-bit address on the ZBI to point to the double-word location in memory where the memory controller is to place the current transfer. Only the 22 most-significant bits of the address

point to the memory location. The memory controller ignores the least-significant address bit (AD00); the state of address bit AD01 tells the memory controller to store the current word (16-bits) in either the upper half or lower half of the double-word space in memory. Figure 4-12 shows the path of two 16-bit words, E and F. The controller stores word E in the word 0 location (bits 31 to 16) and stores word F in the word 1 location (bits 15 to 0)). All word-size transfers must occur only on a word boundary.

4.5.5.3 Long-Word Translation. Long words (Figure 4-13 contain four bytes (32 bits) and occupy the entire width of the ZBI and memory controller, and end up in a location in memory. The 22 most-significant address bits point to the location; the controller ignores the two least-significant bits, AD00 and AD01.

4.6 System Reset

A system reset can be generated from a power-up circuit on the CPU board or by the RESET button on the front panel of the System 8000. The power-up reset circuit makes certain that all functions in the system start in an orderly manner. The RESET button on the front panel is disabled by the ON/LOCK keylock switch when in the LOCK position.

When the system is reset, the following actions occur:

1) All eight serial I/O channels are disabled. All SIO control registers must be initialized.

2) All CTC channels stop counting and all interrupt-enable bits are cleared. CTC control registers must be initialized.

3) Parallel data is inhibited.

4.7 Non-Maskable Interrupts (NMI)

Non-maskable interrupts are typically reserved for external events that require immediate attention. They cannot be disabled (masked) by software. The System 8000 provides three sources of NMI: manual NMI, power-fail NMI, and a double-bit non-correctable ECC memory error NMI.

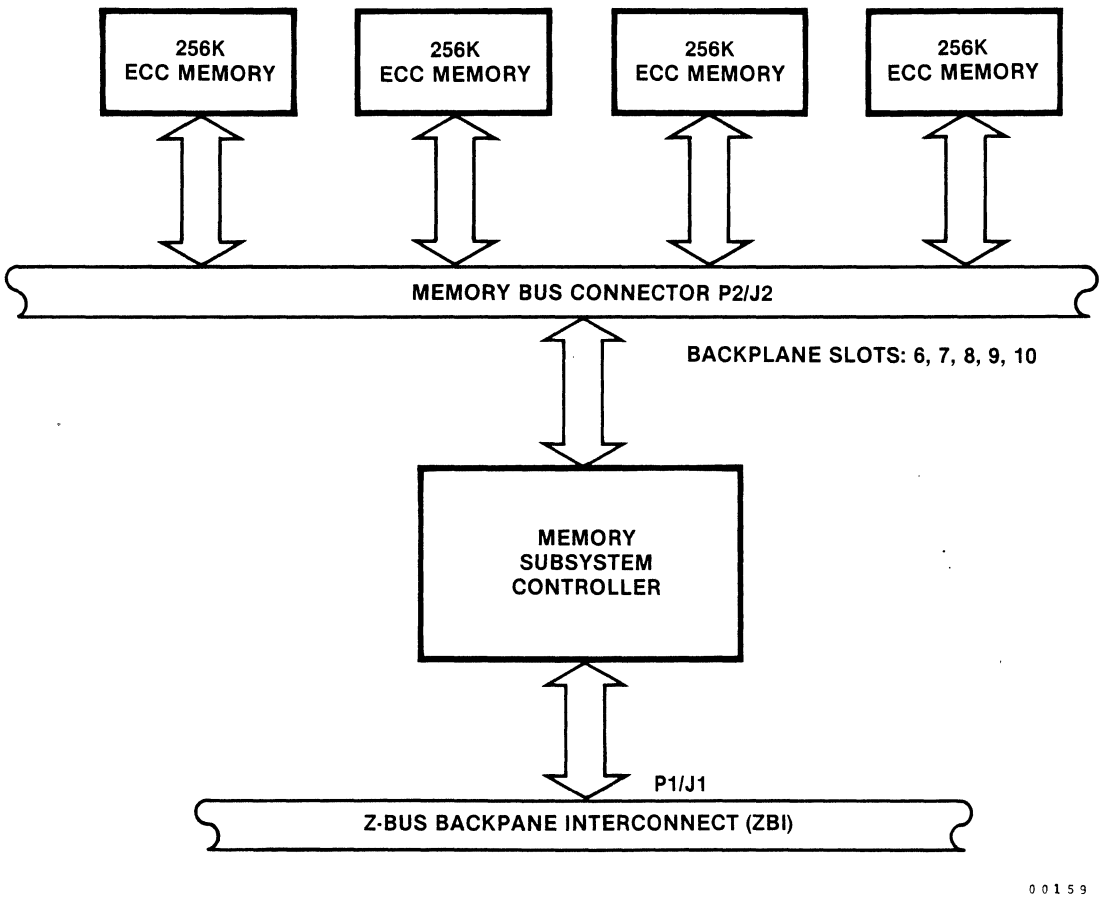


Figure 4-9. Memory Subsystem Controller, Functional Relationships

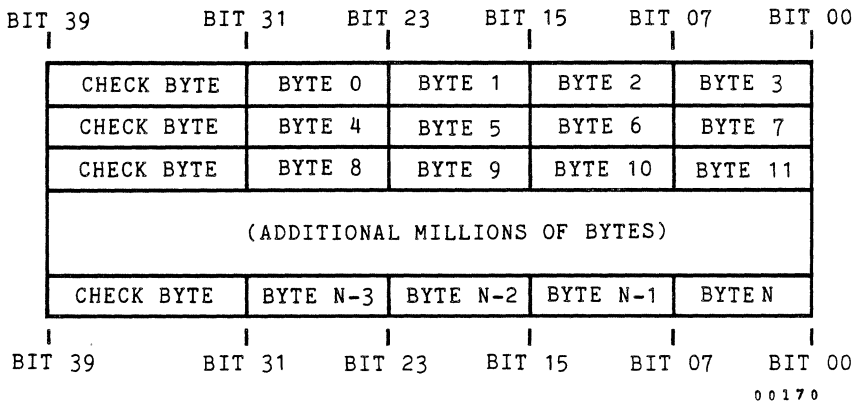


Figure 4-10. Memory Organization

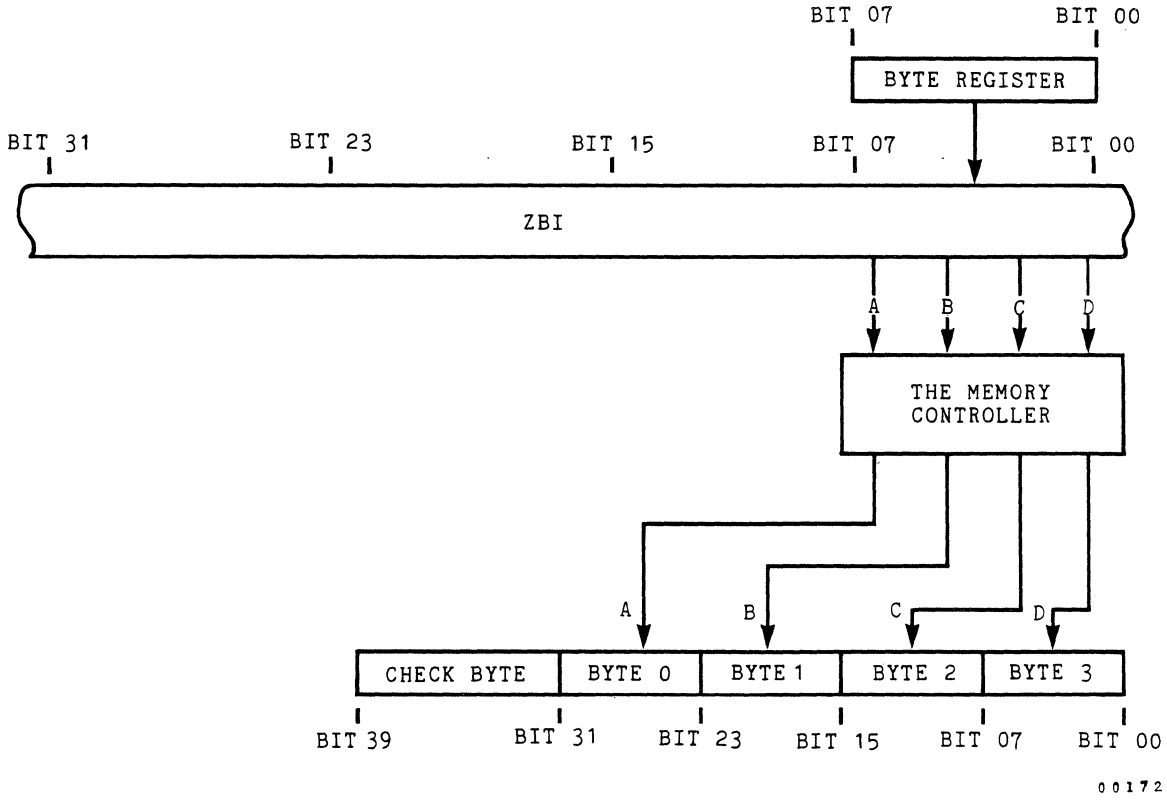


Figure 4-11. Byte Translation

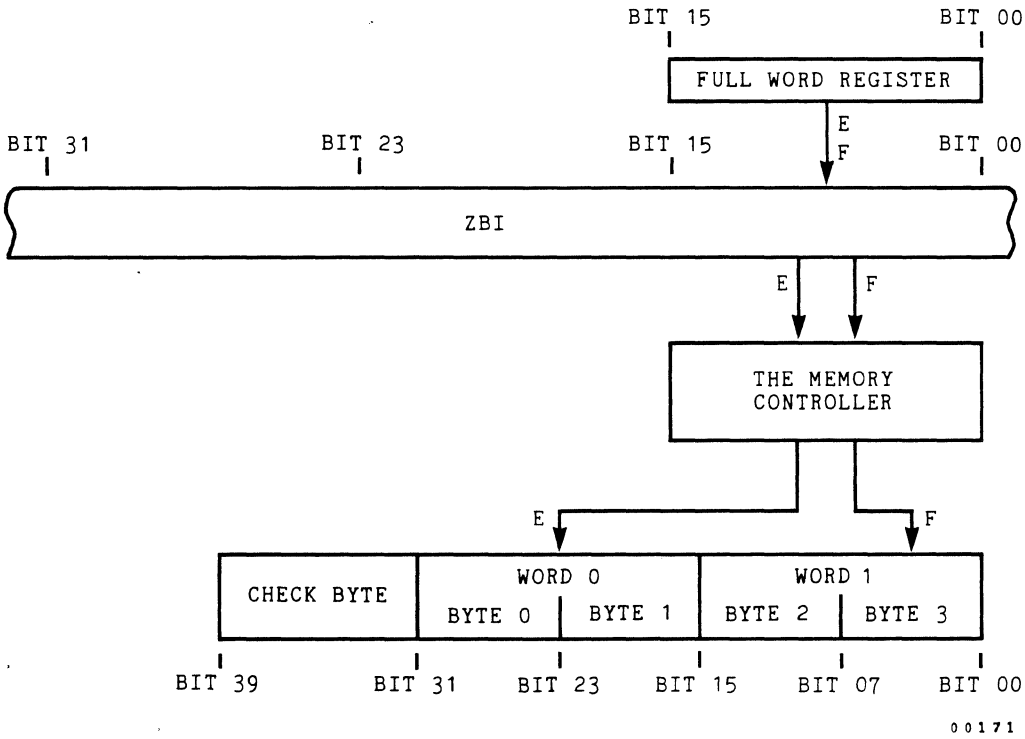


Figure 4-12. Word Translation

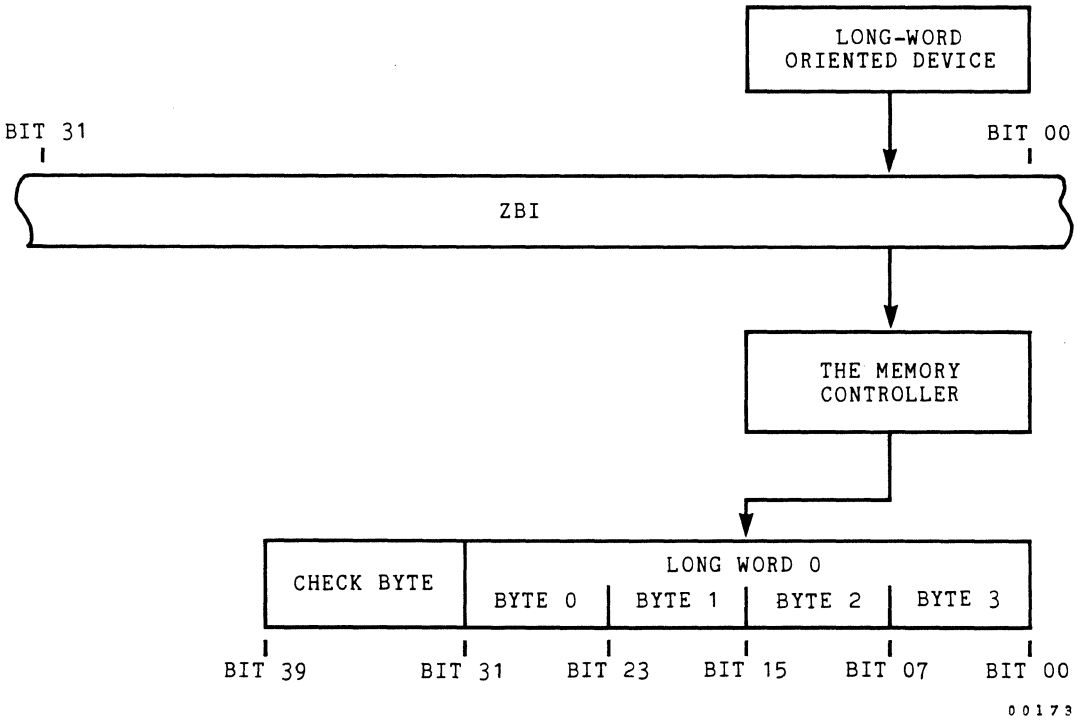


Figure 4-13. Long-Word Translation

4.7.1 Manual NMI

A manual NMI can be generated in the System 8000 from the START pushbutton on the front panel. The NMI can be disabled by placing the ON/LOCK keylock switch in the LOCK position. When the START pushbutton is pressed immediately after a manual or power-up reset, the system power-up diagnostics (SPUD) firmware is invoked. At the conclusion of the diagnostic, if no errors have been recorded, the message "POWER UP DIAGNOSTICS COMPLETE" appears on the console screen and the ZEUS Operating System is automatically booted.

4.7.2 Power-Fail NMI

A power-fail NMI will be sent to the Z8001 CPU when the system power supply detects a decrease in line voltage signifying a potential power failure. After receiving a power-fail NMI, the system has approximately 2 msec to power-down. When a power-fail NMI identifier is read by the operating system, the operating system generates a software reset which in turn, becomes a hardware system reset. Therefore, the Winchester disk drives are protected from crashing during a power failure.

4.7.3 ECC Memory Error NMI

A non-maskabe interrupt will be sent to the CPU when a double-bit non-correctable ECC memory error is flagged by the ECC Controller, and when the CLEAR ECC ERROR-bit of the System Configuration Register (SCR) is set to enable an ECC NMI. The error bit is initially cleared at the SCR during a system reset or power-up.

4.7.4 NMI Identifier

When a non-maskable interrupt is detected by the CPU, the subsequent initial instruction fetch cycle is initiated, but aborted. The program counter (PC) is not updated, but the system stack pointer is decremented. The next CPU machine cycle is the interrupt acknowledge cycle. Theis cycle acknowledges the interrupt and reads a 16-bit IDENTIFIER word (all 16 bits can represent peripheral device status and ID information) from the device that generated the interrupt (in this case, an NMI source). This identifier word, along with the program status information, is stored on the system stack and new status information is loaded into the PC and FCW (flag and control word register).

When the CPU generates an NMI acknowledge status code (0101), the source of the NMI will be either a manual, power-fail, or ECC memory error NMI. Dedicated logic on the CPU board enables a 4-bit error buffer to place a 4-bit NMI IDENTIFIER on mulitplexed address/data lines AD0 to AD3, as follows:

AD3	AD2	AD1	AD0	SOURCE
0	0	0	1	Manual NMI
0	0	1	0	Power-fail NMI
0	1	0	0	ECC Memory Error NMI

Bits AD4 to AD15 are "don't care" bits in an NMI acknowledge identifier word if the NMI source is any one of the three listed sources of NMI in the System 8000. If the NMI source is external to the CPU and not one of the three listed sources of the NMI, the NMI buffer will remain off and the 16-bit identifier word will be read from the system bus.

4.8 Vectored Interrupts

When the CPU acknowledges an interrupt from a peripheral device, it reads a 16-bit identifer word to identify the source of the interrupt. In vectored interrupts, the identifer is also used by the CPU as a pointer to select a particular interrupt service routine associated with the peripheral that was the source of the interrupt.

The System 8000 CPU can configure all PIO (Parallel I/O Controller), CTC (Counter/Timer Circuit), and SIO (Serial I/O Controller) peripheral devices for vectored interrupt operation. The interrupt vectors associated with these peripherals can be loaded at any time since these devices are initialized with their interrupts disabled (masked).

NOTE

It is recommended that vectored interrupts be disabled by the CPU until all peripherals on the System 8000 CPU have been properly initialized because of the vectored interrupt daisy-chain.

Since there are several Z80B peripheral devices on the CPU board, an interrupt daisy-chain is used to prioritize the devices and to accelerate their interrupt request time. Each Z80B device contains two lines that function as links in the daisy-chain: IEI, Interrupt Enable In (INPUT, active high) and IEO, Interrupt Enable Out (OUTPUT, active high).

The Z80B peripherals on the CPU board are prioritized in a daisy-chain as indicated in Table 4-27. Figure 4-14 illustrates the daisy-chain configuration.

Table 4-27. Device Priority Scheme

PRIORITY	PERIPHERAL DEVICE	FUNCTION
1	CTC 0	Single Step, also generates BAUD0, BAUD1, BAUD2
2	CTC 1	Generates BAUD3, BAUD4, BAUD5
3	CTC 2	Generates BAUD6, BAUD7, and the Real Time Clock
4	SIO 0	Serial Channels 0, 1
5	SIO 1	Serial Channels 2, 3
6	SIO 2	Serial Channels 4, 5
7	SIO 3	Serial Channels 6, 7
8	PIO 0	Line Printer Interface
9	(OFF CPU BOARD I/O)	(Secondary Serial Board)
10	Winchester Disk Controller	
11	Tape Controller	

Vectored interrupts from any of the peripherals, 1 to 8 in Table 4-27, automatically disables interrupts from lower priority peripheral devices in the chain. Off-board devices have the lowest priority in the chain.

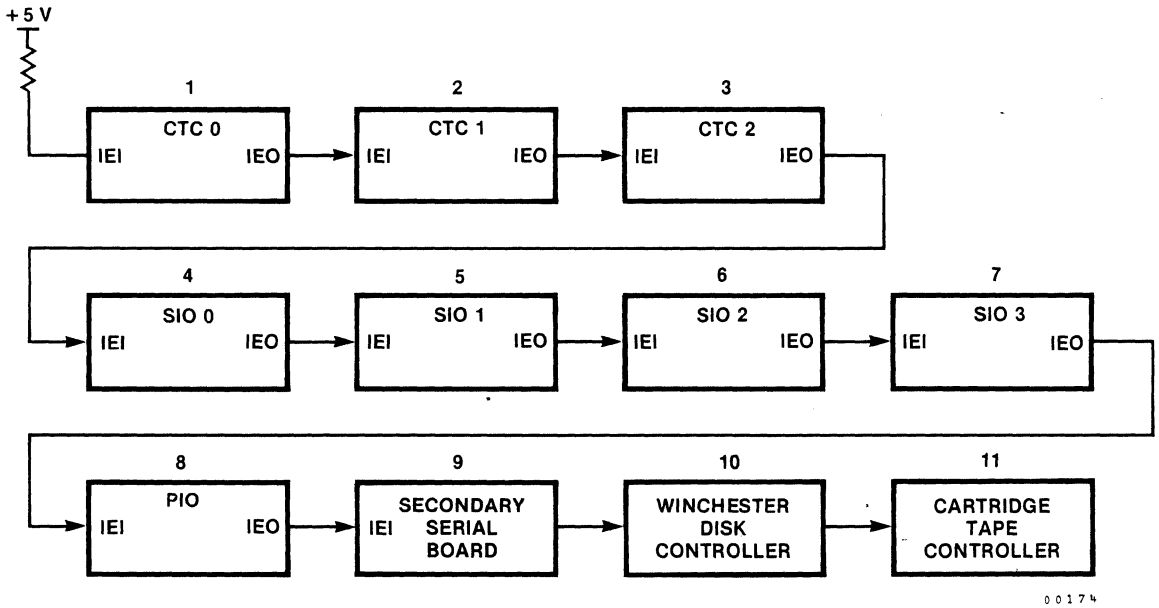


Figure 4-14. Interrupt Priority Connections

4.9 Memory Management Unit (MMU)

The Z8010 Memory Management Unit manages the 16M byte main memory address space of the System 8000 CPU. The MMU also provides the following features:

1. Flexible and efficient allocation of main memory resources during the execution of both operating system and user tasks.

2. Support multiple, independent tasks that share access to common resources.

3. Protection from unauthorized or unintentional access to data or other memory resources.

4. Detection of incorrect use of memory by an executing task.

5. Partitioning of main memory resources to separate user functions from system functions.

4.9.1 MMU Operation (Non-Segmented)

A non-segmented operating system runs in segment 0, using Segment Descriptor Register (SDR) number 0 of memory management units M1, M2 and M3 for code, data, and stack areas, respectively. MMU M1 is used for translating program memory references while M2 and M3 translate all other memory references. The selection between M2 and M3 is based on a comparison between the logical address and the contents of the System Break Register (SBR), a program addressable hardware register. Logical addresses with values lower than the SBR are treated as data addresses and are directed to MMU M2 (data). Logical addresses that are equal to or greater than the SBR, are treated as stack addresses and are directed to MMU M3 (stack).

A non-segmented user program runs in segment 63; however, the hardware will allow any segment between 1 and 63 to be used. As in the non-segmented operating system, Segment Descriptor Register number 63 in M1, M2 and M3 is used to provide separate code, data, and stack areas, respectively. The only difference is that the Normal Break Register (NBR), also a program addressable hardware register, is used to distinguish between code, data, and stack references instead of the System Break Register.

4.9.2 MMU Operation (Segmented)

A segmented operating system uses MMU M1 (code) to provide an address space consisting of up to 63 segments, e.g., segments 0 to 62. Segment 63 is used to run non-segmented user programs. Since the attribute flags in the segment descriptor registers of MMU M3 (stack) are used to configure different segments, no separation between code, data or stack references is required.

A segmented user program uses M2 and M3 to provide an address space consisting of 126 or 128 segments, without separating code, data, and stack areas. If the operating system is non-segmented, then segment numbers 0 and 64 are reserved for the operating system since it requires SDR number 0 of M2 and M3. In a segmented operating system, all 128 segments are usable.

4.9.3 MMU Configurations

The MMU configuration is set by hardware jumpers on the CPU board and by the operating system software. The jumpers are used to configure the MMU select logic for either a segmented or nonsegmented operating system. The operating system software configures the System Configuration Register (SCR) for running segmented or nonsegmented user processes (programs). Refer to Table 4-5 for possible jumper configurations.

4.9.4 Break Registers

Two 8-bit hardware registers, the System Break Register (SBR) and the Normal Break Register (NBR), are accessable as I/O ports on the CPU board. During any memory reference, the 16-bit logical address offset generated by the CPU is compared to the break value given by the contents of either the SBR or the NBR. The SBR is referenced for the break value if the segment number is zero, and the NBR if the segment number is nonzero. If the MMU configuration specifies separation of code, data, and stack areas, and the CPU status indicates a nonprogram reference (status 10xx), then the result of this comparison selects between data and stack references. If the logical address offset is less than the break value, the current reference is for data (MMU M2 is enabled); otherwise, it is a stack reference (MMU M3 is enabled). The following paragraphs describe the possible configurations of operating systems (OS) and user programs (USER).

Nonsegmented OS, Nonsegmented USER

This configuration is intended to run operating systems in memory segment 0 and user programs in any segment, 1 to 63. (Segment 63 is recommended for running user programs.)

For operating systems executing in this configuration, MMU MI (code) is enabled for program references indicated by a CPU status code 11xx, an instruction space access. For memory references other than program references, the logical address offset generated by the CPU is compared against the contents of the SBR, if the segment number is zero; however, if the segment number is nonzero, the comparison is made against the ocntents of the NBR. If the result of the comparison is less than zero, the select logic enables MMU M2 (data); otherwise, MMU M3 (stack) is enabled. Additionally, logic on the CPU board detects memory references made to segment 0 while the CPU is in normal operation mode. This

logic generates a segment trap violation to the Z8001 CPU, disables the three MMUs, and asserts a suppress signal that prohibits main memory references.

If the operating system, or any part thereof, executes in segment mode, the separation of code, data, and stack spaces still applies.

NOTE

> The separation between data and stack spaces is based on the contents of the SBR for segment 0, and the NBR for references to all other segments.

A segmented user program can run in this configuration, although the Nonsegmented OS, Segmented USER configuration is intended for that purpose. Such a user program has a potential address space of 63 code and data segments.

Nonsegmented OS, Segmented USER

This configuration is intended to run exactly as the previous configuration, provided that the CPU is in system mode and the operating system is running in memory segment 0. In addition, code, data, and stack references are directed to M1, M2, and M3 respectively, and the contents of the SBR are used to select between data and stack references. However, if the CPU is in normal mode, MMU M2 is enabled for segment numbers 1 to 63, and MMU M3 is enabled for segment numbers 65 to 127. If a memory reference is made, a segment trap violation is generated and the three MMUs are disabled. Also, the suppress signal is generated by the CPU to protect the system data and stack areas from being accessed by the user program. In system mode, if the segment number of a user segment is generated (segments 1 to 63, or 65 to 127), the address translation is the same as in normal mode. Separation of code, data, and stack spaces are deactivated; MMU M2 is enabled for segments 1 to 63; and MMU M3 is enabled for segments 65 to 127. This allows the operating system to directly access any user segment.

4.9.5 System Access to User Space

To access a user segment, the operating system can use a free segment slot and set its Segment Descriptor Register to point at the same memory area as the target user segment's SDR.

A nonsegmented operating system running a nonsegmented user program can directly access the user data and stack areas by switching to system mode and using the user segment number. To access the user code segment, one of the unused segment slots is set to point at the code segment; for example, number 62. The SDRs for this segment slot in M2 and M3 are both set to point at the code segment, negating the contents of the NBR.

A nonsegmented operating system running a segmented user program can directly access any protion of the user space by switching directly into segmented mode.

4.9.6 System Segments and System Segment Protection

Logic on the CPU board partitions segments into system segments (logical segments 0, 1, 64 and 65), and user segments (logical segments 2 to 63 and 66 to 127). Any reference to a system segment always enables the System Break Register for comparison with the logical address offset, while any reference to a user segment always enables the Normal Break Register. These comparisons are independent of whether the Z8001 CPU is executing in system or normal mode. The function of the system segment detection logic is to prohibit normal mode programs from accessing system mode segments. Normal mode references to system segments are not valid and cause no MMU to be selected, and a segment violation forced upon the CPU. This violation is maintained until cleared by the segment trap acknowledge status of the CPU.

SECTION 5

MAINTENANCE

5.1 Introduction

This section contains preventive maintenance, corrective maintenance, and component removal and replacement procedures.

5.2 Preventive Maintenance

Preventive maintenance consists of routine cleaning procedures and adjustments performed in compliance with schedules provided in Sections 5.3 through 5.4.

5.3 Tape Drive Periodic Maintenance

Components of the Cartridge Tape Drive requiring cleaning are shown in Figure 5-1. The Magnetic Tape Mechanism cleaning procedures described in paragraph 5.3.1 through 5.3.4 should be performed in accordance with the schedule in Table 5-1.

Table 5-1. Cleaning Schedule

ITEM		HOURS OF USE
Magnetic Head	#	8
Tape Cleaner	#	8
Motor Capstan	#	8

5.3.1 Magnetic Head Cleaning

The magnetic head should be cleaned daily if the tape drive is in regular use. Dirty heads cause loss of data during read and write operations. Use a nonresidue, noncorrosive cleaning agent, such as DuPont Freon TF or isopropyl alcohol, and a soft cotton swab to clean the head assembly. Be sure to wipe off any excess cleaning agent and allow the heads to dry prior to operating the drive.

CAUTION

Spray head cleaners are not recommended because overspray will contaminate the motor bearings. Also, never clean the head with hard objects. This results in permanent head damage.

5.3.2 Tape Cleaner Cleaning

The tape cleaner removes loose tape oxide and other foreign material from the tape before it contacts the head. Refer to Figure 5-1. This foreign material accumulates in and around the tape cleaner and must be removed to ensure that the tape cleaner continues to work effectively. The tape cleaner should be cleaned on the same schedule as the head.

To clean, insert a folded sheet of paper into the bottom of the cleaning slot of the cleaner. Slide the paper up, lifting the foreign material from the cleaner. Compressed air or a soft brush can be used to remove the foreign material from the area around the tape cleaner and head assembly. Alternatively, the tape cleaner can be cleaned using the same materials used to clean the magnetic head.

CAUTION

Do not use hard objects to clean the tape cleaner. If the tape cleaner becomes chipped, it will scratch the tape surface, resulting in lost data and/or permanent tape damage.

5.3.3 Motor Capstan Cleaning

The drive capstan is composed of hard polyurethane and must be cleaned after foreign material has built up. Clean, using isoprophyl alcohol and a soft cotton swab. The cleaning schedule is the same as for the head.

CAUTION

Do not allow cleaning solvent to contaminate the drive motor bearings.

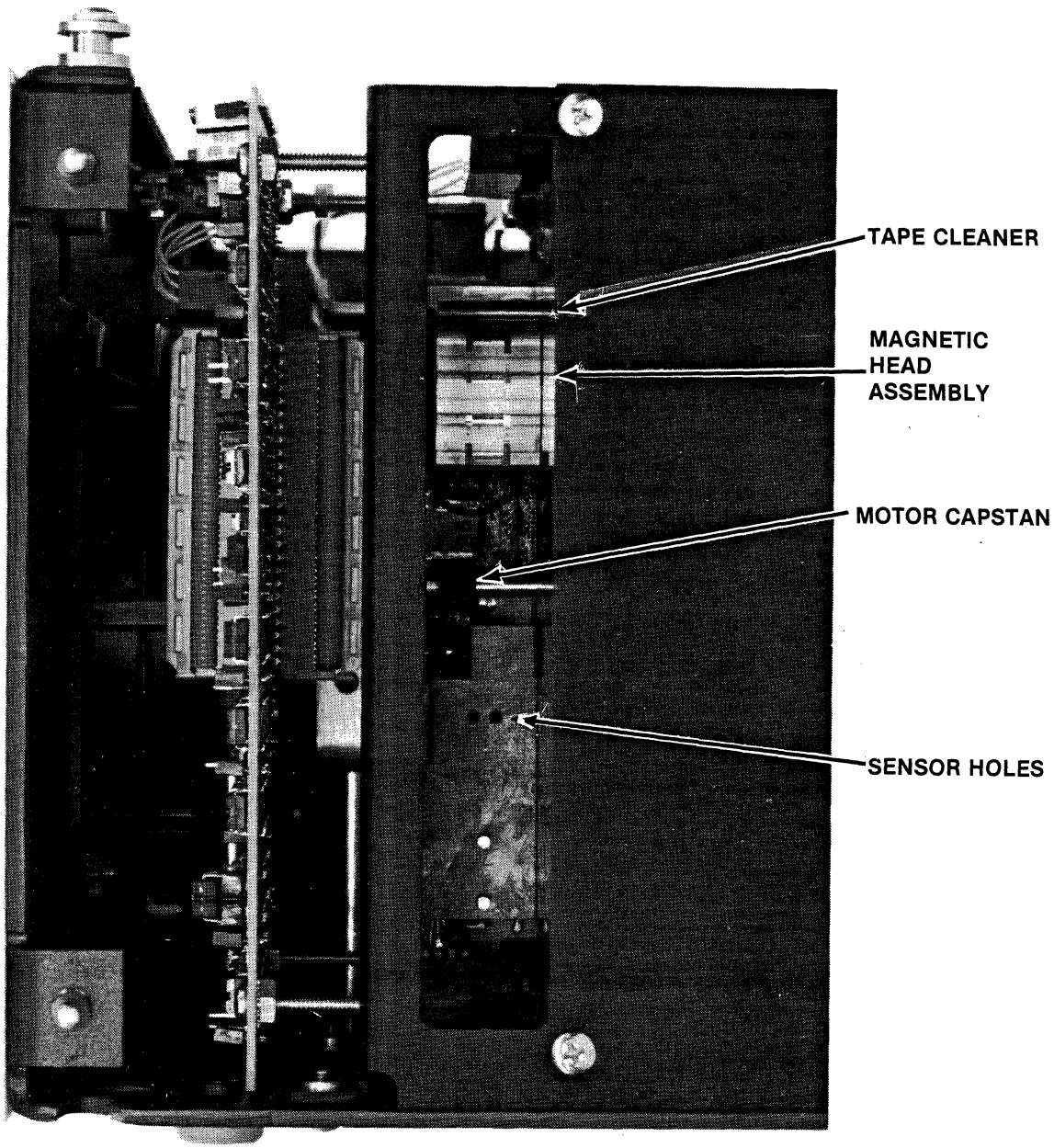


Figure 5-1. Location of Parts Requiring Periodic Cleaning

5.3.4 Heat Sink, Circuit Board, and Sensor Hole Cleaning

To prevent possible overheating, dust and dirt must be removed from the heat sink and drive assembly components. The time period between cleanings varies widely, depending on the operating environment. Use a soft brush and/or compressed air for cleaning. The sensor holes are cleaned in the same manner.

5.4 Disk Drive Assembly Cleaning

The disk assembly is a sealed unit and therefore does not require preventive maintenance procedures.

5.5 Disk Drive Configuration Options

The Disk Drive Unit contains a variety of options that the user selects. These options must be setup prior to the installation of a disk unit in the System 8000.

1. UNIT ADDRESS SELECTION - There are four jumper locations; they are UN0 through UN3 and are located on the jumper block in PCB location 1C. PCB location is as shown in the lower right corner of Figure 5-2.

2. MODEL I.D. - The Model I.D. is established by the presence or absence of a jumper across pins 5 and 16 of the jumper block in PCB location 1C, which is labeled "Model I.D." on the PCB (Figure 5-3).

   Set the Model I.D. jumper to the desired model number as shown in Figure 5-3.

3. SECTOR SIZE SELECTION - Figure 5-4 shows the jumper locations that result in specific sector-size selections. The sector-size selections in turn affect the number of bytes per sector and hence, the number of bytes per track.

   Set the Sector-Size Selection jumpers to the desired sector size.

Unit Address
Jumper Configuration

Unit Address	UN 3	UN 2	UN 1	UN 0
Unit 0				
Unit 1				●
Unit 2			●	
Unit 3			●	●
Unit 4		●		
Unit 5		●		
Unit 6		●	●	
Unit 7		●	●	●
Unit 8	●			
Unit 9	●			●
Unit 10	●		●	
Unit 11	●		●	●
Unit 12	●	●		
Unit 13	●	●		●
Unit 14	●	●	●	

● Indicates Jumper Installed

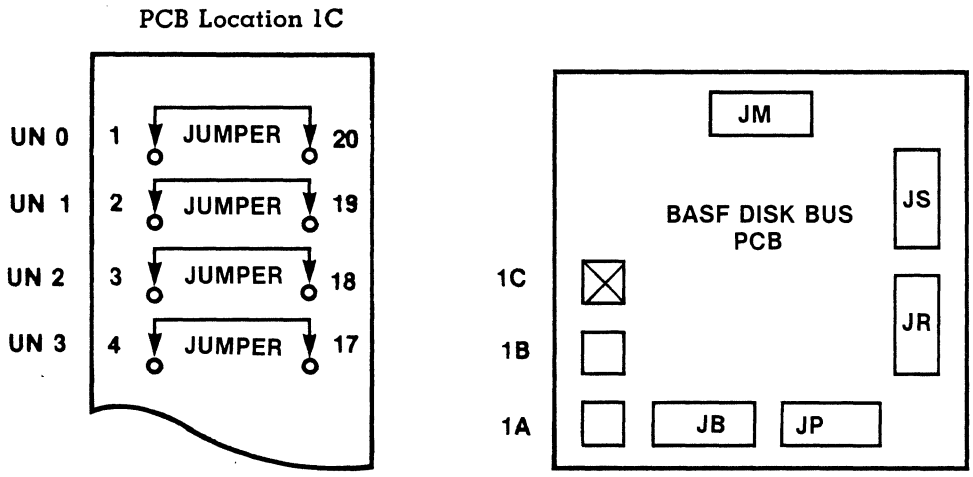


Figure 5-2. Unit Address Selection

Model/Drive Capacity	Model ID Jumper Locations		
	ID 0	ID 1	ID 2
6171/8MB		●	●
6172/24MB	●		●
6173/40MB	●	●	
	1C-5/16	1C-6/15	1C-7/14

● = Jumper Installed

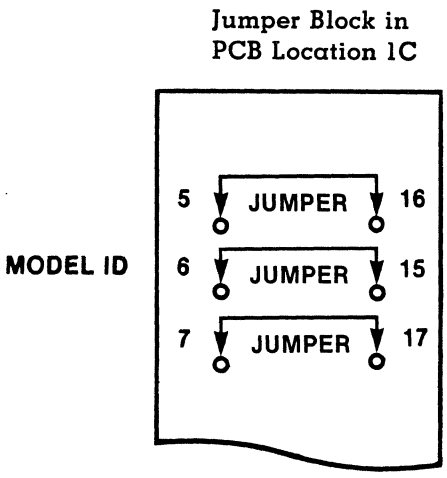
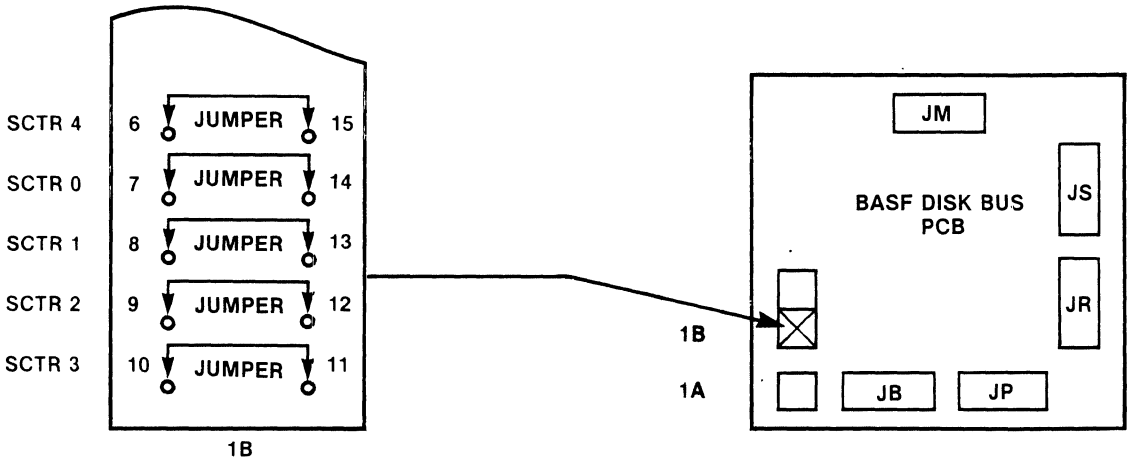


Figure 5-3. Model ID Selection



Sector Format				Jumper Locations				
Overhead Bytes	Data Bytes	# Sectors/Track		SCTR 4	SCTR 3	SCTR 2	SCTR 1	SCTR 0
		Decimal	Hex					
28	128	84	54					
28	256	46	2E					●
28	512	24	18				●	
28	1024	12	0C				●	●
28	2048	6	06			●		
28	4096	3	03			●		●
36	128	80	50			●	●	
36	256	45	2D			●	●	●
36	512	24	18		●			
44	128	76	4C		●			●
44	256	44	2C		●		●	
44	512	23	17		●		●	●
44	1024	12	0C		●	●		
44	2048	6	06		●	●		●
44	4096	3	03		●	●	●	
28	128	85	55	●				
36	128	81	51	●		●	●	
44	128	77	4D	●	●			●
*	*	Reserved		●	●	●	●	●

*Reserved for future use ● Indicates Jumper is Installed

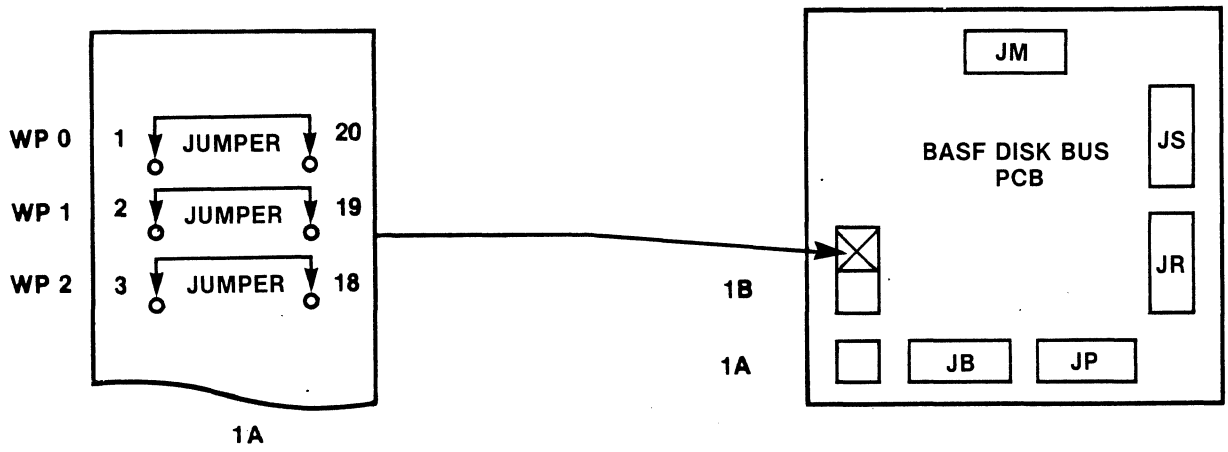
Figure 5-4. Sector Size Selection

4. HEAD WRITE PROTECTION - There are two methods of effecting write protection using the Disk Bus Interface PCA. One method uses jumper plugs; the other method uses programmable selection. The jumper plugs define the minimum head write protection configuration. Additional heads may be write protected under program control.

   Set the Head Write Protection jumpers to achieve the desired level of write protection.

   a. JUMPERED WRITE PROTECTION - Figure 5-5 shows the two jumpers available for write protection on Model 6171 and 6172 drives. Two jumper locations on the jumper block in PCB location 1B are used. They are between pins 1 and 20 (WP0) as well as between pins 2 and 19 (WP1). If a jumper is installed at WP0, head 0 is write protected; if a jumper is installed at WP1, both heads 0 and 1 are write protected; if jumpers are installed at WP0 and WP1, 1 and 2 are protected. Head 3 (the servo head) is always write protected.

   b. PROGRAMMABLE WRITE PROTECTION - Programmable write protection can be effected on heads which are not already "write protected" by the use of jumpers. A special command, "EXT PROT 0 and 1" is executed at command-word-3 time. This command must be sent from the host computer to the drive via the Controller-Formatter. The two least significant bits of the command are used for head write protection. These bits remain in position as they were set until a subsequent command-word-3 command is executed. Figure 5-6 shows the formatting of the command word and the resultant logical functions on write protect.



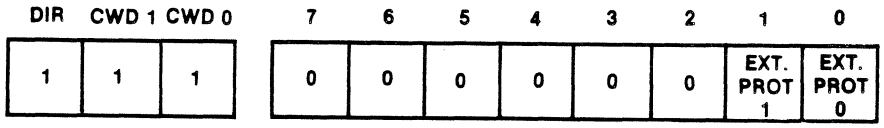
Write Protect Jumper Locations

Protected	WP 2*	WP 1	WP 0
None			
Head 0			●
HD 0 & 1		●	
HD 0, 1, & 2		●	●

● Indicates Jumper Installed
* Reserved for Future Use

Note: Head No. 3 (Servo Head) is always Write Protection.

Figure 5-5. Jumpered Head Write Protection



Ext. 1	Prot. 0	Function
0	0	No Head Protected
0	1	Head 0 Protected
1	0	Head 0 and 1 Protected
1	1	Head 0, 1, and 2 Protected

Figure 5-6. Programmed Head Write Protect

5.6 Disk Drive Mounting

The Disk Drive Unit should be permanently mounted in place before the linear motor lock-screw is unlocked. Further, the lock-screw should not be unlocked until after power is turned ON and the drive has been brought up to speed.

5.7 Cabling and Connections

The two cables that connect the BASF 6170 with Disk Bus Interface to other BASF drives and to the host system are listed in Table 5-2.

CAUTION

The interface connector for the Disk Drive Bus is located very close to W1 (the jumper location for connecting DC ground to frame ground). Use extreme care when inserting or removing the interface cable so the locking tabs for the connector do not break W1.

Table 5-2. Interface Cables

CABLE NAME	DESCRIPTION	VENDOR/PART NUMBER
Signal Cable	40-conductor flat ribbon cable -- maximum of 30 ft. (9.1 cm)	Spectra-Strip 455-248-40
	Connector (Open strain relief)	3M 3417-6040
Power Cable	10-conductor, 18 gauge wire-- maximum of 4 ft. (1.2 m)	Zilog Part Number 59-0137-00
	Connector	AMP 1-640426-D

The location of the above two cables is as shown in Figure 5-7. Marker bands on the cable are keyed to pin No. 1 of the receptacle on the PCB.

The signal cable can be connected to a maximum of 15 daisy-chained disk drives. Terminators must be installed in the controller as well as the last drive of the daisy-chain string.

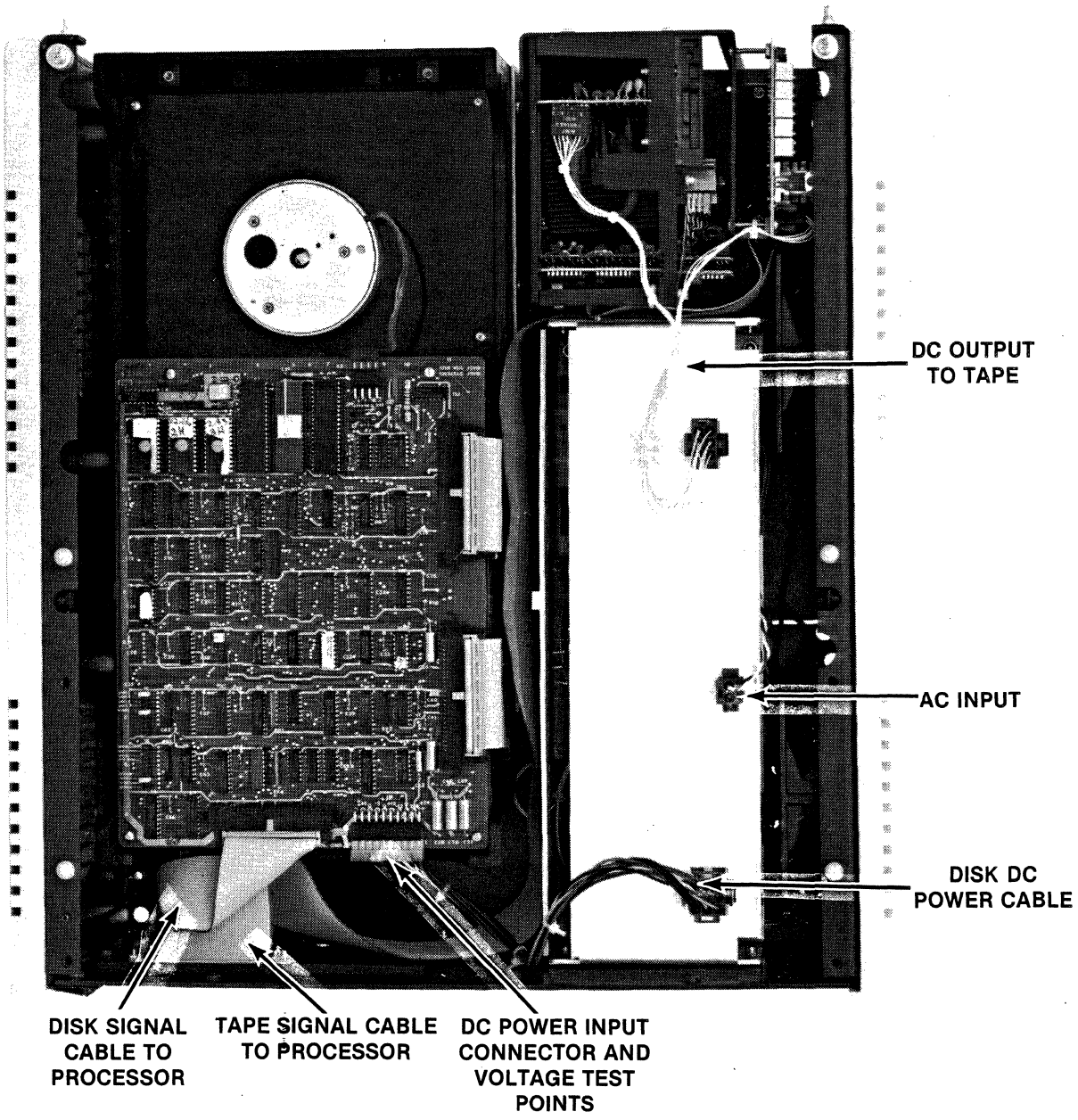


Figure 5-7. Disk Drive Cable Locations

5.8 Power ON Procedure and Precautions

This subsection contains the procedure for turning power ON for the first time. The disk is a nonvolatile storage media; therefore, once data is recorded it is not lost when power is turned OFF.

The following checks must be performed prior to turning power ON:

1. Inspect the physical mounting to ensure that the drive is secured at all six mounting points and that all ribbon cables are securely connected.

CAUTION

Improper orientation of the power connector can result in serious damage to the Disk Drive Unit. See Figure 5-8 for proper installation of the power connector.

If this is the first time that the user is connecting the drive to the host system, continuity/resistance checks through the ribbon cable and connectors are advised.

2. Turn the System 8000 power ON.

NOTE

The voice-coil actuator lock-screw must be in the locked position. (Refer to Section 3.10.)

3. Unlock the voice-coil actuator lock-screw by rotating the screw in a counter-clockwise direction using fingerpressure or a small blade screwdriver.

CAUTION

This should only be done after the disk is rotating at full speed. The lock-screw springs out after several rotations. When a signal is applied to the

head actuator and it does not respond as commanded, there is a possibility that it is adhering to the rubber stop. It can be freed by applying very light pressure to the lock-screw, pushing it gently against the actuator and thus forcing it away from the stop. A hard push or sudden jolt can send the actuator across the disks and into the opposite stop with the possibility of some damage.

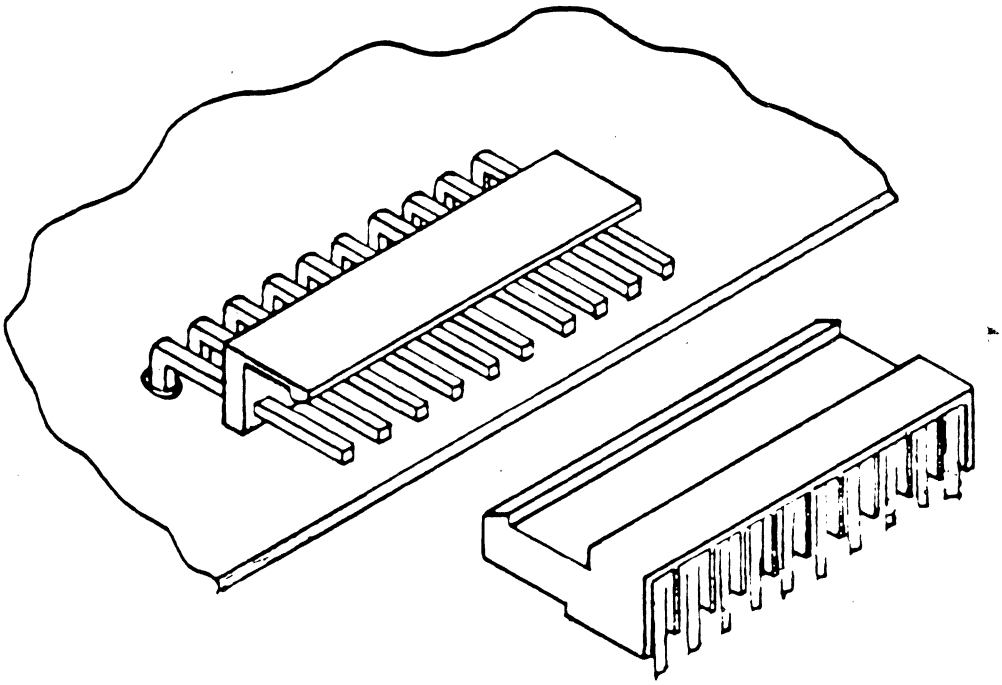


Figure 5-8. Disk Drive Power Connector

5.9 Problem Analysis

If the System 8000 does not respond in accordance to operating instructions, or if it appears that the system is not functioning normally, the SPUD and/or SADIE diagnostic procedures, described in the following sections, should be performed.

5.10 System Power-Up Diagnostics (SPUD)

SPUD exists in a Read Only Memory (ROM) integrated circuit unit located in the Central Processor Unit (CPU) of the system. This diagnostic routine is automatically performed at the time of system turn-on in response to pressing the RESET and START controls on the Processor Module. It can also be initiated from the CPU Monitor by pressing RESET and T <CR>.

SPUD tests the primary functions of the CPU and peripherals. It verifies the system's ability to execute a limited number of instructions and communicate with system peripherals. Functions performed by SPUD are:

1. System 8000 Instruction Test -- Specified System 8000 instructions are tested.

2. MMU -- All accessible internal registers and the segment trap functions are tested.

3. Memory Test -- All locations of memory are tested for read and write functions.

4. ECC Test -- The error detection capabilities of the system are tested.

5. Peripheral Equipment Test -- Does a cursory check of the Winchester Disk and Cartridge Tape Controllers.

If the SPUD diagnostics detect the presence of a problem in the System 8000, error messages are displayed on the system console. Table 5-3 describes these error messages. If error messages are displayed, remove all recording disks, turn off system power, and verify that all cable connectors and PC component boards are properly inserted in their plugs. If error messages are encountered after all cable connectors and PC boards have been checked, proceed to the SADIE diagnostic procedures (Section 5.11).

Table 5-3. SPUD Diagnostics Error List

ERROR #	P1	P2	P3	P4	CHRS * PRINTED	DESCRIPTION
0000	--	--	--	--	P	No External Memory **
0001	SEG #	ADDR	RD	--	O	Seg. Addr Fault **
0100	SEG #	ADDR	TD	RD	W	Mem. Addr Fault
0101	SEG #	ADDR	TD	RD		Data Line Fault
0102	SEG #	ADDR	TD	RD		'As' Data Fault
0103	SEG #	ADDR	TD	RD		'5s' Data Fault
0104	--	--	--	--		No Good Segments ** Above Zero
0100	SEG #	ADDR	TD	RD	E	Segment Zero Memory Test (Descriptions As Above)
0101	SEG #	ADDR	TD	RD		
0102	SEG #	ADDR	TD	RD		
0103	SEG #	ADDR	TD	RD		
0200					R (sp)	ECC Check Byte generator Failure
0201					U	ECC Check Byte RAM Failure
0202					P (sp)	ECC Uncorrectable Error Trap Failure
0300	MMU PORT #	SDR FIELD #	TD	RD	D	MMU's Not Individually Addressable
0301	MMU PORT #	SDR FIELD #	TD	RD		SAR Or DSCR Indexing Fault
0302	MMU PORT #	SDR FIELD #	TD	RD		SDR 'As' Or '5s' Data Fault
0303	MMU CMD #	TD	RD	--		MMU Control Register 'As' Or '5s' Fault
0304	REG #	TD	RD	--		System/Normal Break Register 'As' Or '5s' Fault
0305	MMU ID #	SDR #	VDAT	--	I	Stack MMU Did Not Trap On Limit Test
0305					A	Unexpected Trap
0305					G	Unexpected Trap
0305					N	Data MMU Did Not Trap On Limit Test
0305					O	Stack MMU Did Not Trap On Read-Only Test
0305					S	Data MMU Did Not Trap On Read-Only Test
0306	MMU PORT #	SDR #	TD	RD	T	Translation Fault On Data MMU
0307	MMU PORT #	SDR #	VDAT	--		Unexpected Trap
0308	MMU PORT #	SDR #	TD	RD	I	Translation Fault On Stack MMU
0309	MMU PORT #	SDR #	VDAT	--		Unexpected Trap
0310	MMU PORT #	SDR #	TD	RD	C	Translation Fault On Code MMU

Table 5-3. SPUD Diagnostics Error List (continued)

0311	MMU PORT #	SDR #	VDAT	--		Unexpected Trap
0312	MMU PORT #	SDR #	--	--	S (sp)	No Trap On Code MMU Limit Test
...COMPL...						Interim Characters of SPUD Message
1000	--	--	--	--	E	No WDC Board In System
1001	DS1	DS2	DS3	DS4		Self-Test Error
2000	--	--	--	--	T	No TCU Board In System
2001	--	--	--	--		Busy Bit Always Set ***
2002	REG #	TD	RD	--		'5s' Data Fault
2003	REG #	TD	RD	--		'As' Data Fault
2004	IV REG	STAT0 REG	MIC REG	--		TCU Self-Test Error ***
2005	IV REG	STAT0 REG	MIC REG	--		TCU Hardware Error ***
...E						Last Character of SPUD Message

* -> Characters of SPUD message printed before entering test

** -> Fatal error preventing further memory-related tests from being run

*** -> The TCU test may take up to two minutes if the drive is busy or if the 'busy' status bit is stuck. The last two TCU error messages dump out the contents of the status registers for troubleshooting.

Pn -> Test parameters of error printed (in hexadecimal):

```
SEG #        -> segment number
ADDR         -> address offset
TD           -> test data
RD           -> returned data
MMU PORT #   -> full word port number of MMU under test
MMU CMD #    -> MMU port number with command 'ored' in
SDR FIELD #  -> indicates a particular SDR in the range 0-255
MMU ID #     -> ID of MMU(s) returned from a segment trap
             ->   1 = code MMU
             ->   2 = data MMU
             ->   4 = stack MMU
SDR #        -> logical segment number or set of SDR's (0-63)
VDAT         -> violation data from a single MMU trapping
        (HB) -> bus cycle status register data
        (LB) -> violation type register data
DS1          -> WDC detailed status - disk ready register
DS2          ->                     - disk status register
DS3          ->                     - operation error status
DS4          ->                     - self-test error status
REG #        -> register port number of unit under test
--           -> no parameter printed
```

When the diagnostics are complete, the maximum available segment number will be displayed as follows (xx in hexadecimal):

```
POWER UP DIAGNOSTICS COMPLETE
MAXSEG=<xx>
```

5.11 Stand-alone Diagnostic Interactive Executive (SADIE)

SADIE is a stand-alone diagnostic executive and diagnostic library. It provides thorough testing of all mainframe components. SADIE is based on the construction, modification, storage, and execution of test lists. The commands to operate on these test lists are displayed in a menu-oriented format on the CRT to minimize the documentation support required and to allow a first-time user to execute SADIE without constant reference to an instruction manual. The SADIE program code and associated data sets reside on the SADIE diagnostic tape cartridge, which is inserted in the tape slot and run whenever degraded system operational capability is suspected. The object code for SADIE resides in the lower half of a memory segment; the diagnostic test resides in the upper half of the segment.

5.11.1 Purpose of SADIE

The basic purpose of SADIE is to test the system hardware. To do this and to provide necessary support, even single tests are considered to be test lists. A test list consists of lines executed sequentially. A line contains either a test or a control statement, up to four parameters necessary for execution of the test or control statement, and a repetition count for all noninteractive tests. Normal sequential execution of a test list can be altered by a GOTO control statement or interrupted by a PAUSE control statement in the list. Every test list includes a termination line consisting of an EOL (end-of-line) in the test name field of the display.

5.11.2 Organization and Principles of Operation

SADIE is organized as a tree of menus to provide its diagnostic functions. Menu items are either submenu names (lower branches of the tree) or functions (the end of the branch) of SADIE. The tree is traversed by entering the item required to perform a function or to move to another menu on the path to a function. Choices are entered on the console. In addition, the START and RESET buttons can be used to interact with SADIE.

The characters identifying menu choices are letters, integers, <CR> (carriage return), and ^ (↑ on some terminals). Entry of ^ causes a return to the menu from which the present menu was selected or to the master (COMMAND LEVEL) menu. A <CR> is used when there is only one choice or when there is a default choice, as in verifying that a

prior choice is correct. Integers are used to identify tests, test lists, or lines in a test list. Letters are used for all other choices. When a letter is used, it is usually the initial letter of a capitalized keyword in the item description. Either upper or lowercase letters can be used.

When a menu is too large to fit into the CRT screen, entry of '+' will get the next screen, and '-' will get the previous screen.

SADIE maintains in memory both the current single test and current test list. Initially the current single test and the current test list only contain an EOL.

5.12 SADIE Tape Organization

The SADIE tape contains the following data sets:

1. The bootstrap loader functions as a tape loading supervisor that loads the SADIE machine code.

2. The SADIE machine code provides I/O support for tests. When invoked, it loads the library and test list catalogs, and initializes the CPU. After completing these initial functions, the executive function is available for interactive use.

3. The diagnostic tape library contains object code blocks.

   An object code block contains the object code for a stand-alone test. A maximum of 29K bytes of memory is allocated to each object code block. This limit is necessary to allow the SADIE machine code to be coresident and to provide room for the stack that grows downward from the end of the segment.

4. The diagnostic library catalog contains all information necessary to identify the test, prepare for its execution, and load it. Included are the test name (1 to 8 characters), a short description of the test (1 to 50 characters), declarations of base (decimal, hexadecimal, or octal) and default values for up to four integer parameters, a functional classification of the test, and its file and track location on the tape.

5. The test list library contains test lists that have been stored. A total of 19 test lists can be stored on the diagnostic tape.

6. The test list catalog contains a one to 50-character description of each test list.

5.13 SADIE Program Initialization

To initialize SADIE:

1. Insert the tape cartridge into the cartridge tape drive.

2. Press RESET.

3. Enter Z T<CR>.

This command executes the monitor-resident primary tape bootstrap routine that loads SADIE's bootstrap routine and transfers control to SADIE. After the bootstrap process is complete, a command level menu is displayed on the CRT that includes the version of SADIE that is present.

5.14 SADIE Diagnostic Functions

SADIE's diagnostic functions fall into five classes accessible from the COMMAND LEVEL menu. This is the first menu displayed after product invocation. Additionally, the user can interact with SADIE using the system RESET and START buttons. The first class is execution of a test list reached by the LIST command, L, or the CHOOSE command, C. The second class is execution of a single test reached by the TEST command, T, or the REPEAT command, R. The third class is display of a SADIE-maintained error log which summarizes the results of running a test or a test list. (This is the DISPLAY command, D.) The fourth class is the EDIT command, E, which provides editing functions for test lists such as creation, modification, storage, and retrieval. The fifth class is the QUIT command, Q, which performs an orderly return to the PROM Monitor.

5.14.1 Console Interactions

RUNNING A TEST LIST

Since single tests are executed as if they were one-line test lists, the running of test lists is treated first. The user can run the test list currently in memory with the LIST

command, L, or choose any of the lists stored on tape with the CHOOSE command, C. In the second case, the catalog of test lists is displayed, and the one selected by the user is transferred from tape to memory. In either case, the subsequent actions are identical.

The user can specify the line where list execution begins; S begins execution at line 1, <CR> begins execution at the current line. Initially, the current line is line 1; however, by entering another number, the user can change the current line to that number.

After a request for execution of the list, a check of the list is performed by SADIE. A line may contain a reference to a test not on the tape due to alteration or corruption of the tape. Any such line will be skipped in execution. The user is notified of any such line or of any line containing a test that would overwrite the disk or tape. If any line overwrites the disk, the user must verify that this is acceptable before execution of the list begins. During execution, if any test would overwrite the tape, the test is loaded, but execution halts until the SADIE tape has been replaced with a scratch tape. A request to reload the tape is not generated until SADIE next attempts to reposition the tape.

NOTE

When the SADIE tape is replaced, wait until it is FULLY REWOUND before responding with 'Y'.

During execution, SADIE updates an error log whenever an error is detected or a test lap is completed. When execution is complete, enter a <CR> to return to the COMMAND LEVEL.

RUNNING A SINGLE TEST

Any test can be selected and run with the TEST command, T. The catalog of tests and control statements is presented. The selected one is displayed with default parameters set by SADIE. The default repetition count, 1, or the parameters can be modified. When satisfied, enter a <CR> to change the test into a special one-line test list. The list check and verification of any attempt to overwrite the disk or tape are then performed just as in running a regular test list. When the test is complete, enter a <CR> to return to COMMAND LEVEL.

To rerun a test previously chosen with the TEST command use the REPEAT command, R. It is now possible to modify the existing repetition count or parameters.

DISPLAYING THE LIST ERROR LOG

The DISPLAY command, D, displays an error log generated by execution of a test or test list. This log contains a line for each line of the test list. Each error log line contains the test name, the number of times it was run, the number of errors it detected, and an indication of whether or not the line was run to completion and whether supplied parameters were found inappropriate and replaced.

EDITING AND EDIT LEVEL

One test list resides in memory and can be altered using the EDIT LEVEL subcommands. This list can be a null list containing only a termination (EOL) line. Up to seven lines of the current list are displayed by the editor. One line of the display is the current line, marked by a *, upon which line-oriented commands operate. Enter an integer to change the current line of the list.

Previously created and stored test lists can be moved to memory from the tape with the FETCH command, F. The memory resident list can be moved to tape with the STORE command, S. In either case, the test list catalog containing list descriptions is displayed, and the source or destination for the test list can be selected. When storing lists, the list description in the test list catalog can be changed.

The memory resident list can be altered in five ways. The CLEAR command, C, erases the current list by making it a null list. The DELETE command, D, removes the current line from the list. The MODIFY command, M, resets the repetition count or parameters of the current line.

The REPLACE command, R, can be used to replace an entire line. The catalog of tests and control statements is displayed and the user makes a choice. Default parameters and repetition count are supplied and the line is displayed. The repetition count or parameters can now be reset. When the line is correct, enter a <CR> to substitute the new line for the original line of the list and to return to the EDIT LEVEL.

The INSERT command, I, presents the catalog of tests and test lists and then offers an optional reset of default repetition count or parameters just as the R command does.

Entering a <CR> when the line is correct, inserts the line in the test list and displays the catalog of tests for another insertion. Entering an E, (for EXIT) when the line is correct, inserts the line in the test list and returns to the EDIT LEVEL menu.

QUITTING

The QUIT command, Q, rewinds and unloads the SADIE tape and returns to the PROM Monitor program.

5.14.2 START and RESET Interactions

RESETTING

Pressing the RESET button forces a hardware reset. It performs the same function as the QUIT command. It is the only sure way to abort a malfunctioning diagnostic.

FORCING A PAUSE WITH THE START BUTTON

When the START button is pressed, a nonmaskable interrupt is generated, and SADIE responds by displaying a PAUSE menu with five choices. This is typically done when running a test or test list. The LIST command, L, displays the current test list. The DISPLAY command, D, displays the error log previously discussed. The ERROR command, E, displays a detailed error log for the current test if it maintains one. The SKIP command, S, sets a flag which, when read by the test, causes it to cease execution. SADIE then runs the next line of the test list. Entering a ^ sets the same flag, but SADIE terminates execution of the test list when the test returns control to it. Entering a <CR> resumes the interrupted process.

NOTE

SADIE cannot force an immediate abort of a test in progress.

5.15 Command Level Test Functions

Stand-alone diagnotics are provided for testing of all mainframe components. These diagnostics are supplied on the SADIE tape. Inserting this tape into tape drive 0 and, after completing the bootstrap procedure (refer to paragraph

5-17), the COMMAND LEVEL menu will appear on the display screen as follows:

```
*****  COMMAND LEVEL *****

T      choose and run single TEST
R      REPEAT current single test
L      run current test LIST
C      CHOOSE and run a test list
E      EDIT test lists
D      DISPLAY error log
M      perform tape MAINTENANCE functions
Q      QUIT

Enter your choice ]=>
```

After this menu appears, the desired diagnostic test may be selected by simply keying in its letter code in response to the prompt "Enter your choice]=>" and pressing the RETURN key.

The following test descriptions are presented in the same sequence as they appear in the COMMAND LEVEL menu. If a test function has subcommands, they are detailed before proceeding to the next test description. This allows for all the information of one test function to appear together for convenient reference.

5.15.1 Command Level T: Choose and run a single TEST

General Description: A test is chosen from the first CRT display. A second CRT display allows for parameters and repetition count to be changed.

CRT Display Contents: ** CHOOSE A TEST OR CONTROL LINE ** Followed by a list of test numbers and corresponding titles

Optional Commands: Optional available commands for this display are:

Command	Description
+	CRT displays next page of available tests.
-	CRT displays preceding page of available tests.

#	Select the test number of your choice.
^	Return to COMMAND LEVEL.

CRT Display Contents: ** RESET TEST LINE (OPTIONAL) **

Optional Commands: Optional available keyword commands for this display are:

Command	Description
R	Reset REPETITION count
P	Reset all PARAMETERS
#	Reset parameter #
I	Test name is INCORRECT--choose a different test
E	Test line is correct--EXIT insert mode
<CR>	Test line is correct--continue insert mode
^	Return to CHOOSE A TEST

The P and # commands appear only when the test has parameters to be set. E appears only in insert mode. I appears only if this menu was reached immediately after the user chose a test.

T SUBCOMMAND LEVEL: REPETITION

General Description: This function is used in conjunction with other commands to specify the number of times a particular test function is to be run.

CRT Display Contents: ***** SET REPETITION COUNT *****

Optional Commands: Optional available commmands in this mode are:

Command	Description
0	Runs until START button is pressed
#	A number # of iterations expressed in decimal notation between 1 and 9999
^	Return to CHOOSE A TEST or CONTROL LINE

T SUBCOMMAND LEVEL: PARAMETERS

General Description: Parameters can be required to completely specify the conditions under which a test is performed.

CRT Display Contents: ***** SET PARAMETERS *****

. .
. .

Optional Commands: Optional available commands in this mode are:

Command	Description
#	A number # representing the value of the para-parameter.
^	Return to CHOOSE A TEST or CONTROL LINE.

5.15.2 Command Level R: REPEAT previously loaded single test

General Description: A previously chosen single test is run. Parameters, repetition counts, and options can be reset.

CRT Display Contents: ** RESET TEST LINE (OPTIONAL) **

Optional Commands: Optional available keyboard commands in this mode are the same as T (choose and run a single TEST).

5.15.3 Command Level L: Run current test LIST

General Description: Current test list (previously chosen or selected at edit level) is run.

CRT Display Contents: ***** RUN CURRENT TEST LIST *****

Optional Commands: Optional available keyboard commands in this mode are:

Command	Description
<CR>	Execute list beginning at current line.
S	START execution at line 1
#	Make line # the CURRENT line (>1)
^	Return to COMMAND LEVEL

5.15.4 Command Level C: CHOOSE and run a test list

General Description: The description of the test lists are displayed and one is selected. The list is examined and, if it contains tests that overwrite the disk, a warning appears. It must be verified that this overwriting is permissible before execution of the list begins. If a test on the list does not appear in the catalog, this information is displayed and execution continues. If a test writes to tape, execution of the list pauses after the test has been loaded until the user signals a scratch tape has been installed and the proper command is entered. Upon completion, the process is reversed to get the SADIE tape installed.

CRT Display Contents: ***** CHOOSE AND RUN LIST ***** Followed by a list of test numbers, and short descriptions

Optional Commands: Optional available keyboard commands in this mode are:

Command	Description
<CR>	EXECUTE list beginning at line 1
S	START execution at line 1
#	Make line # (>1) the current line
^	Return to COMMAND LEVEL

5.15.5 Command Level E: EDIT test list

General Description: This level performs storage and list modification function options.

CRT Display Contents: ***** EDIT LEVEL *****
Followed by a menu containing a test number column, a repetitions column, and a parameter column

Optional Commands: Optional available keyboard commands in this mode are:

Command	Description
C	CLEAR current test list
F	FETCH list from tape to current test list
S	STORE current test list
D	DELETE line x
R	REPLACE line x
M	MODIFY repetitions on parameters on line x
I	INSERT line(s) before line x

#	Make line # (<x+1) current line
^	Return to COMMAND LEVEL

E SUBCOMMAND LEVEL: C - CLEAR current test list

General Description: The current test list residing in RAM is deleted before creating a new test list.

CRT Display Contents: Not applicable

Optional Commands: Not applicable

E SUBCOMMAND LEVEL: F - FETCH list from tape to current test list

General Description: The CRT displays a catalog of test lists stored on tape and a test list is selected and loaded into memory in preparation for desired modification.

CRT Display Contents: ***** FETCH LIST *****
Followed by a menu containing a list of the 19 test lists.

Optional Commands: Not applicable

E SUBCOMMAND LEVEL: S - STORE current test list

General Description: The current catalog of test lists stored on the tape are displayed. The user chooses a list number where the current test list is to be stored. The current test list description is displayed and the user has the opportunity to enter a new description. Then the current test list overwrites the original test list on the tape and the test list catalog is updated.

CRT Display Contents: ***** STORE LIST *****
Followed by a menu containing a list of the 19 test lists.

Optional Commands: Not applicable

E SUBCOMMAND LEVEL: D - DELETE line x

General Description: Line is removed from test list.

CRT Display Contents: Not applicable

Optional Commands: Not applicable

E SUBCOMMAND LEVEL: R - REPLACE line x

General Description: A test or control statement, including parameters and repetition count, are chosen and installed in the test list that replaces the selected line contents.

CRT Display Contents: ** CHOOSE A TEST OR CONTROL LINE **

Optional Commands: Not applicable

E SUBCOMMAND LEVEL: M - MODIFY repetitions or parameters on line x

General Description: The number of repetitions or value of any parameter can be changed.

CRT Display: ** RESET TEST LINE (OPTIONAL) **

Optional Commands: Commands in this mode are the same as T (choose and run a single TEST).

E SUBCOMMAND LEVEL: I - INSERT line(s) before line x

General Description: A line is inserted before the selected line. The list is displayed again, centered above the originally selected line. This procedure continues as long as new lines are entered.

CRT Display: ** CHOOSE A TEST OR CONTROL LINE **

Optional Commands: Not applicable

E SUBCOMMAND LEVEL: # - Make line # the current line

General Description: Pointer is moved to line number #.

CRT Display: Not applicable

Optional Commands: Not applicable

5.15.6 Command Level D: DISPLAY error log

General Description: The error log maintained in SADIE is displayed, including the test name, the number of times a test was executed, the number of errors reported by the test (organized by test list line) and the completion status of each test line. The error log is cleared prior to the start of execution of a test list or single test.

CRT Display Contents: TEST #REPS #ERRORS

Optional Commands: Not applicable

5.15.7 Command Level Q: QUIT

General Description: The QUIT command terminates the SADIE program functions. The SADIE tape is rewound to the physical load point and system control is returned to the PROM Monitor.

CRT Display Contents: When this command is complete, the Monitor message appears.

Optional Commands: Not applicable

5.16 SADIE Test Line and Control Statements

5.16.1 SADIE Test List

Any test can be selected and run with the TEST command, T. When selected on the COMMAND LEVEL menu, the following catalog of tests and control statements are displayed:

```
1  PAUSE     CONTROL LINE - wait for keyboard input
2  GOTO      CONTROL LINE - jump to chosen line specified times
3  WDCCRC    Verifies CRC of data portion for each sector
```

4	WDCFMT	Formats disk-DESTRUCTIVE of disk data
5	CYLSPARE	Formats disk; shows bad cylinders-DESTRUCTIVE
6	WDCMEDIA	Formats, reads, writes, and reads-DESTRUCTIVE
7	WDCTST3	Queue test for Winchester disk-DESTRUCTIVE
8	WDCTST7	Multi-sector disk read-write-compare-DESTRUCTIVE
9	WDCMON	Exercise monitor for the disk controller
10	TCUMON	Exercise monitor for the tape controller
11	TCOM	TCU Tape Command Exerciser
12	TEX	Tape media verification program
13	NEWMEM1	Random data memory test
14	NEWMEM2	Quick memory test
15	NEWMEM3	Very thorough memory test-HSLOW
16	MMUTST5	Thorough testing of MMU registers and functions
17	CENT.PRT	Verifies Centronics printer interface
18	DP.PRT	Verifies Data Products printer interface
19	S16SIO	Interactive SIO and CTC test

Appendix E contains detailed information of each test on the SADIE diagnostic tape. Control Statements are defined in the following paragraph.

5.16.2 Control Statements

Control statements GOTO and PAUSE can be inserted in test lists to modify list execution.

GOTO

The GOTO statement is useful for setting up test loops. It causes the execution of the test list to jump to a specified line. A loop count of 0 to 9999 is specified. When the specified loop count is reached, control will fall to the next statement in the test list. When special value 0 is specified, GOTO will always be executed.

PAUSE

This statement causes the PAUSE menu to be displayed. SADIE programs are not disturbed; they are only temporarily halted to allow for optional action. The optional available commands are listed in the following PAUSE menu:

```
***** PAUSE *****

L          Display current test LIST

D          Display error log

E          Display detailed error log
```

S	SKIP to next line in test list
<CR>	RESUME current test
^	Return to COMMAND LEVEL

All the commands operate the same way, except the SKIP command, S, which causes the line of the test list following the PAUSE line to be skipped, rather than the PAUSE line itself.

5.17 Using SADIE

The purpose of this example is to familiarize the user with the CRT displays and interaction with the console during SADIE execution.

Inserting the SADIE tape into drive 0 and pressing the RESET button causes the following CRT display to appear:

```
S8000 Monitor 1.0 - Press START to Load System
```

To load the primary bootstrapper, enter the command:

```
ZT <CR>
```

The primary bootstrapper displays:

```
BOOTING FROM TAPE
```

When booting is complete, information pertaining to the current version of the diagnostic tape will appear on the CRT display. For example:

```
SADIE   (Stand Alone Diagnostic Interactive Executive)
Customer Release:   1.6      Released:  October 13, 1981

This version of SADIE works with Rev.8 Winchester Disk
Drives only
```

This display will appear momentarily followed by the COMMAND LEVEL menu:

```
*****  COMMAND LEVEL  *****

T    choose and run single TEST
R    REPEAT current single test
L    run current test LIST
C    CHOOSE and run a test list
E    EDIT test lists
D    DISPLAY error log
```

```
M     perform tape MAINTENANCE functions
Q     QUIT

Enter your choice ]=>
```

After this menu appears, the desired diagnostic function may be selected by simply entering its letter code in response to the prompt "Enter your choice]=>". For example, selecting T, followed by <CR>, produces the following submenu:

```
*****  CHOOSE A TEST OR CONTROL LINE  *****

1  PAUSE      CONTROL LINE - wait for keyboard input
2  GOTO       CONTROL LINE - jump to chosen line specified times
3  FMTVER     Verifies that all sectors are formatted
4  WDCFMT1    Formats all sectors - DESTRUCTIVE of disk data
5  WDCFMT2    Formats and reads all sectors - DESTRUCTIVE of data
6  WDCFMT3    Formats, reads, writes, and reads: DESTRUCTIVE
7  WDCTST3    Queue test for Winchester disk - DESTRUCTIVE
8  WDCTST7    Multi-sector disk read-write-compare - DESTRUCTIVE
9  WDCMON     Exercise monitor for the disk controller
10 TCUMON     Exercise monitor for the tape controller
11 TCOM       TCU Tape Command Exerciser
12 TEX        Tape media verification program
13 NEWMEM1    Random data memory test
14 NEWMEM2    Quick memory test
15 NEWMEM3    Very thorough memory test-HSLOW
16 MMUTST5    Thorough testing of MMU registers and functions
+  Show the next page of tests
-  Show the preceding page
^  To return to COMMAND LEVEL

Enter your choice ]=>
```

The CHOOSE A TEST OR CONTROL LINE submenu presents the catalog of tests and control statements available to the user. Testing of the MMU registers and functions can be accomplished by selecting test option 16 (MMUTST5).

After 16 is entered in response to the prompt "Enter your choice]=>", followed by <CR>. The following display appears:

```
*****  RESET TEST LINE  (OPTIONAL)  *****

The test line is currently set as follows:

NAME      #REPS    PARAMETER 1    PARAMETER 2    PARAMETER 3  PARAMETER 4
--------  -----    -----------    -----------    -----------  -----------
MMUTST5       1    No parameters to set
R     to reset REPETITION count
I     test name is INCORRECT -- choose different test
<CR>  if test line is correct
^     to return to COMMAND LEVEL

Enter your choice ]=>
```

The MMUTST5 test is displayed with default parameters set by SADIE. The default repetition count, 1, or the parameters can be modified. To reset the repetition count, enter R in response to the prompt "Enter your choice]=>", followed by <CR>. The following display appears:

```
*****  SET REPETITION COUNT  *****

Current repetition count = 1

0      to run test until NMI pressed
#      (positive # <= 9999) to run test this many times
^      to return to RESET TEST LINE
<CR>   to leave repetition count the same

Enter your choice ]=>
```

To modify the default or existing repetition count, following the prompt "Enter your choice]=>", enter the number of times the test is to run. In this case, the default repetition count, 1, is changed to 5. When <CR> is entered, the following display appears:

```
*****  RESET TEST LINE  (OPTIONAL)  *****
```

The test line is currently set as follows:

```
NAME      #REPS    PARAMETER 1    PARAMETER 2    PARAMETER 3  PARAMETER 4
--------  -----    -----------    -----------    -----------  -----------
MMUTST5       5    No parameters to set
```

```
R      to reset REPETITION count
I      test name is INCORRECT -- choose different test
<CR>   if test line is correct
^      to return to COMMAND LEVEL

Enter your choice ]=>
```

When satisfied with the repetition count, enter <CR> in response to the prompt "Enter your choice]=>". This changes the test into a special one-line test list. The list check and verification of any attempt to overwrite the tape is then performed just as in running a regular test list. During the list check and verification process, the message

***** CHECKING TEST LIST... *****

is displayed, followed by the message

***** CHECK COMPLETE *****

After the CHECK COMPLETE message, the following display appears:

```
************************************************************ ************
                The following test is now running
NAME       #REPS    PARAMETER 1    PARAMETER 2    PARAMETER 3 PARAMETER 4
--------   -----    -----------    -----------    ----------- -----------
MMUTST5        5    No parameters to set
************************************************************ ************

This is MMUTST5 - version 1.01
        Now doing a block random data test on all MMUs.
        Now testing SDRs with random data and random accesses
        Now testing control registers with random data.
        Now testing READ ONLY flags in the data and stack MMUs
        Now testing the LIMIT flags of the stack and data MMUS
        Now testing the DIRW (direction) flag of the stack and data MMUs
        Now testing TRANSLATION on the DATA MMU seg(1-63)
        Now testing TRANSLATION on the STACK MMU seg(1-63)
        Now testing TRANSLATION on the CODE MMU seg(1-63)
```

As the first repetition of MMUTST5 progresses, the test line for each MMU register or function is displayed, in sequence, until the test is complete. If an error occurred during the test, error messages will be displayed immediately following the last test line.

When the test repetition is complete, the following lap count summary appears:

```
This is MMUTST5 - version 1.01
   LAPCNT=1 ERROR COUNT=0

CODE MMU ERRORS
SDR: BLOCK 0       CONTROL: SAR  0       FLAGS: RD  0       TRANS: 0
     RNDM  0                DSC  0              LIM 0
                            MODE 0              DIR 0
```

The MMUTST5 test running and lap count summary screens appear the number of times set in the SET REPETITION COUNT display. When the last repetition is complete, the message

Hit <CR> to return to COMMAND LEVEL]=>

appears immediately following the last lap count summary.

When the COMMAND LEVEL menu appears, any of the diagnostic functions may be selected. To facilitate this example, entering the QUIT command, Q, in response to the prompt "Enter your choice]=>" terminates the SADIE diagnostic functions. The tape is rewound to the physical load point and system control is returned to the PROM Monitor.

5.18 Power Supply Voltages

The SPUD or SADIE programs do not diagnose problems caused by system power supplies. If the system does not respond normally when the SPUD or SADIE diagnostics are performed, degraded system power supplies could be the source of the problem.

Power Supply Voltage Checks

The Processor Module main power supply should be tested for DC voltage output if a problem exists that cannot be found by SPUD or SADIE diagnostic procedures. A separate power supply is contained in the Peripheral Module. Voltage testing of the Processor Module power supply requires that the top cover be removed from the module; access is from the right rear corner of the module.

TEST EQUIPMENT Required: HP3466A Multimeter or equvalent

Processor DC Voltage Test

1. Remove top cover from processor module by unscrewing the thumb screws at the back of the top cover. Pull the top cover toward the back of the unit and gently lift at the same time.

2. Remove the sheet metal under cover over the front 3/4 of the unit by lifting upward and once the cover guides are cleared pulling forward very carefully. Use caution when doing this so as not to deform the backplane pins.

CAUTION

This operation should be done with the unit turned off.

3. Test points on backplane are accessed from top of module at right rear of card cage (Figure 5-9).

4. Check DC voltage outputs for the following:

   - At TP2 check for +5Vdc ±0.1
   - AT TP3 check for +12Vdc ±0.1
   - AT TP5 check for -5 Vdc ±0.1

Adjustment for +5Vdc is made by removing the left side panel and accessing the adjustment screw on the rear of the power supply, through the rear panel opening. It is recommended that only Zilog trained field service personnel perform power supply adjustments on this unit.

Peripheral DC Voltage Test

The Peripheral Module power supply output voltages are checked at the Winchester disk power connector at the rear of the Winchester disk (Figure 5-7). The processor module must be unstacked from the Peripheral Module in order to access the power connecter. The DC voltages produced by the Peripheral Module power supply are:

+24 Vdc ±0.2
±12 Vdc ±0.1
±5 Vdc ±0.1

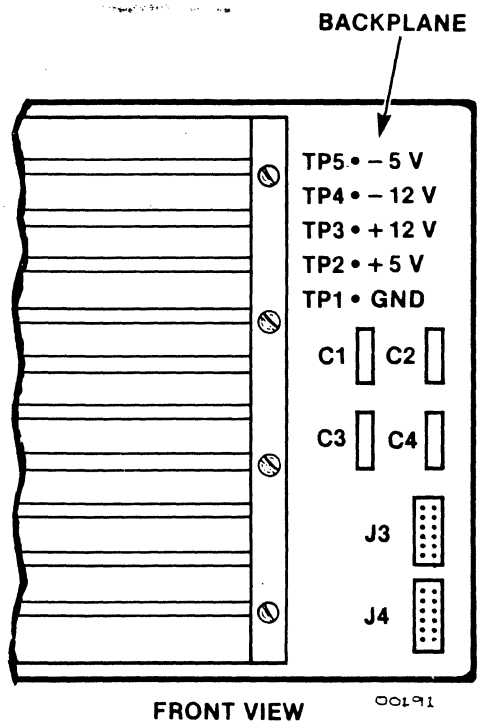


Figure 5-9. Power Supply Voltage Test Points

APPENDIX A

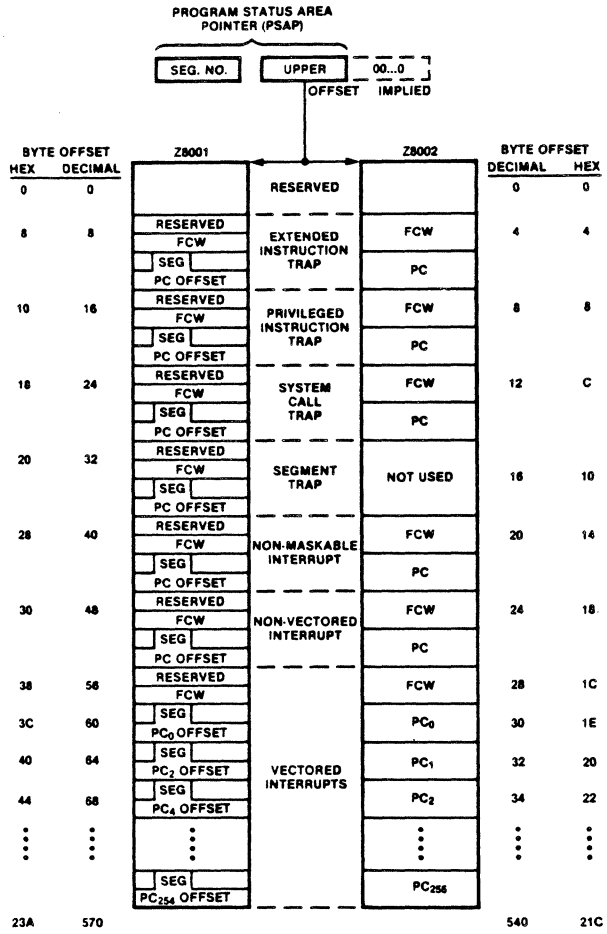
SUMMARY OF Z8001 INSTRUCTION SET

A.1 General

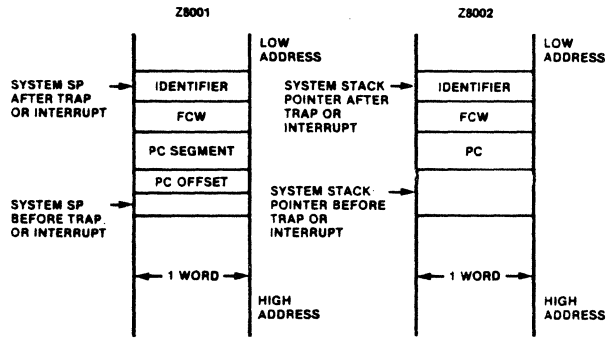
Appendix A contains a summary of the instruction set of the Z8001 microprocessor. This summary does not present the details of the instruction set; rather, it is an aid to those who are familiar with the instruction set. For detailed information regarding the Z8001 instruction set, refer to the following manual:

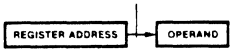
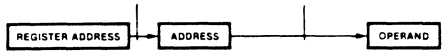
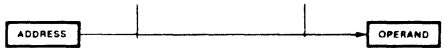
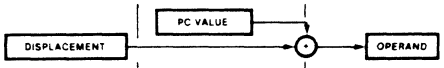
Z8000 CPU Technical Manual Part Number 00-2010-C

Program Status Area



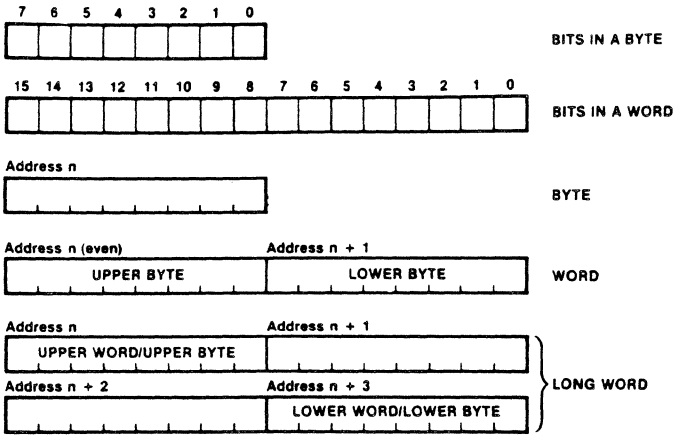
Format of Saved Program in the System Stack



Addressing Mode	Operand Addressing			Operand Value
	In the Instruction	In a Register	In Memory	
R Register				The content of the register
IM Immediate				In the instruction
*__IR__ Indirect Register				The content of the location whose address is in the register
DA Direct Address				The content of the location whose address is in the instruction
*__X__ Index				The content of the location whose address is the address in the instruction plus the content of the working register.
RA Relative Address				The content of the location whose address is the content of the program counter, offset by the displacement in the instruction
*__BA__ Base Address				The content of the location whose address is the address in the register, offset by the displacement in the instruction
*__BX__ Base Index				The content of the location whose address is the address in a register plus the index value in another register.

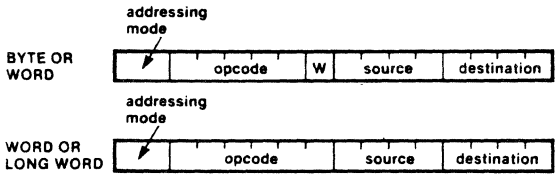
*Do not use R0 or RR0 as indirect, index or base registers.

Addressable Data Elements



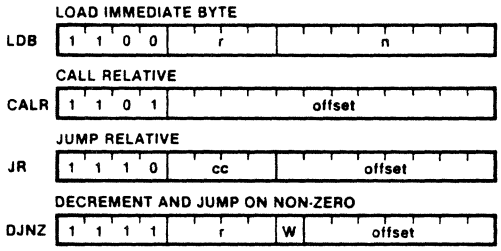
Instruction Formats

GENERAL INSTRUCTION FORMAT (FIRST WORD)

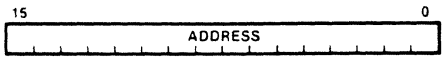


Note: W indicates Word (1) or Byte (0)

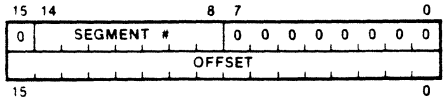
COMPACT INSTRUCTION FORMAT



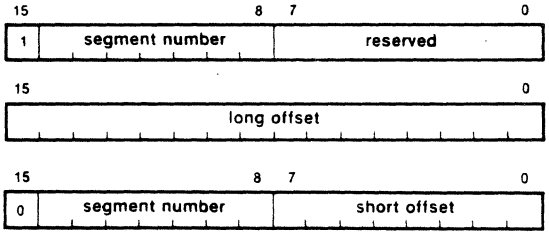
Non-Segmented Memory Address (Z8002 Only)



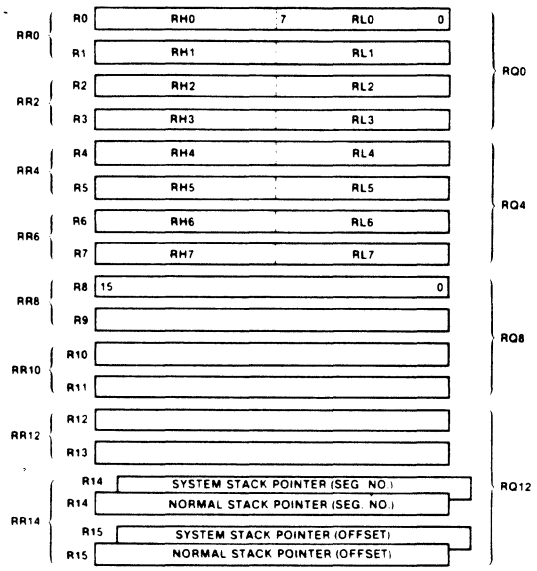
Segmented Memory Address (Z8001 Only)



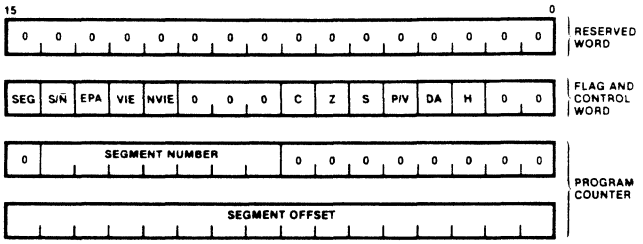
Segmented Memory Address Within Instruction



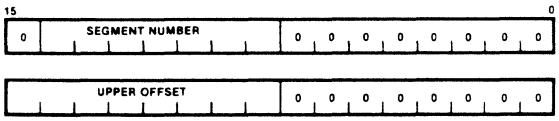
Z8001 General-Purpose Registers



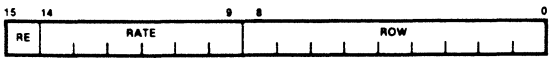
Z8001 Special Registers



Z8001 Program Status Registers

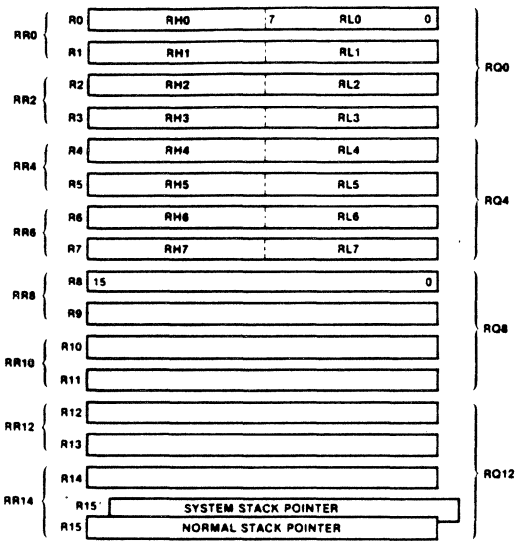


Z8001 Program Status Area Pointer

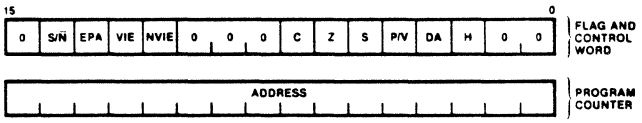


Z8001 Refresh Counter

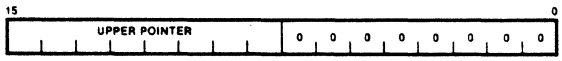
Z8002 General-Purpose Registers



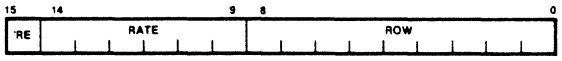
Z8002 Special Registers



Z8002 Program Status Registers



Z8002 Program Status Area Pointer



Z8002 Refresh Counter

Condition Codes

Code	Meaning	Flag Setting	CC Field Code Binary	Hex
	Always True	-	1000	8
Z	Zero	Z = 1	0110	6
NZ	Not Zero	Z = 0	1110	E
C	Carry	C = 1	0111	7
NC	No Carry	C = 0	1111	F
PL	Plus	S = 0	1101	D
MI	Minus	S = 1	0101	5
NE	Not Equal	Z = 0	1110	E
EQ	Equal	Z = 1	0110	6
OV	Overflow	P/V = 1	0100	4
NOV	No Overflow	P/V = 0	1100	C
PE	Parity Even	P/V = 1	0100	4
PO	Parity Odd	P/V = 0	1100	C
GE	Greater than or Equal (signed)	(S XOR P/V) = 0	1001	9
LT	Less Than (signed)	(S XOR P/V) = 1	0001	1
GT	Greater Than (signed)	[Z OR (S XOR P/V)] = 0	1010	A
LE	Less than or Equal (signed)	[Z OR (S XOR P/V)] = 1	0010	2
UGE	Unsigned Greater than or Equal	C = 0	1111	F
ULT	Unsigned Less Than	C = 1	0111	7
UGT	Unsigned Greater Than	[(C = 0) AND (Z = 0)] = 1	1011	B
ULE	Unsigned Less than or Equal	(C OR Z) = 1	0011	3

Note that some condition codes have identical flag settings and binary fields in the instruction:
Z = EQ, NZ = NE, C = ULT, NC = UGE, OV = PE, NOV = PO.

Status Line Codes

ST_3-ST_0	Definition	ST_3-ST_0	Definition
0 0 0 0	Internal operation	1 0 0 0	Data memory request
0 0 0 1	Memory refresh	1 0 0 1	Stack memory request
0 0 1 0	I/O reference	1 0 1 0	EP data memory request
0 0 1 1	Special I/O reference (e.g., to an MMU)	1 0 1 1	EPU stack memory request
0 1 0 0	Segment trap acknowledge	1 1 0 0	Program reference, nth word
0 1 0 1	Nonmaskable interrupt acknowledge	1 1 0 1	Instruction fetch, first word
0 1 1 0	Nonvectored interrupt acknowledge	1 1 1 0	EPU-CPU transfers
0 1 1 1	Vectored interrupt acknowledge	1 1 1 1	Reserved

Binary Encoding for Register Fields

Register Quad	Register Pair	Register	Byte	Binary	Hex	Register Quad	Register Pair	Register	Byte	Binary	Hex
RQ0	RR0	R0	RH0	0000	0	RQ8	RR8	R8	RL0	1000	8
		R1	RH1	0001	1			R9	RL1	1001	9
	RR2	R2	RH2	0010	2		RR10	R10	RL2	1010	A
		R3	RH3	0011	3			R11	RL3	1011	B
RQ4	RR4	R4	RH4	0100	4	RQ12	RR12	R12	RL4	1100	C
		R5	RH5	0101	5			R13	RL5	1101	D
	RR6	R6	RH6	0110	6		RR14	R14	RL6	1110	E
		R7	RH7	0111	7			R15	RL7	1111	F

The subsequent tables display the following information:

Instruction. The instruction mnemonic, with a /B and/or /L specifies word, byte, or long word formats. All usable addressing modes are shown for each instruction.

Example: *ADD/B/L* means the ADD instruction also has ADDB and ADDL forms, signifying byte and long word forms.

Hexadecimal Code. These columns show the hex codes used in each 4-bit field. When coding differs for word, byte, and/or long word extensions, hex numerals are shown in the instruction mnemonic: 6/5/5. When a field contains addresses, data, flags, or register or condition codes, the field is marked appropriately.

Example: *ADD/B/L Rd,#n 0/0/1* means that the first hex digit in coding ADD Rd,#n or ADDB Rd,#n is 0. The first digit in ADDL Rd,#n is 1.

Flags. These symbols indicate flag conditions:

0 = flag is reset by instruction execution.
1 = flag is set by instruction execution.
‡ = flag is affected according to result of instruction execution.
★ = flag is unaffected by instruction execution.
X = flag condition is undefined following instruction execution.

When flag conditions differ between word, byte, and/or long word extensions, flag descriptions are marked by W, B, or L, indicating word, byte, or long word.

Cycle Times. Z8000 instruction execution time varies due to addressing modes used, whether operating in segmented mode or not, and whether the long word extension is used or not. Cycle times in each addressing mode are aligned with the left instruction column. The six vertical columns indicate cycle times for word or byte instructions in the first three columns and the times for long word instructions in the second three columns. In each set of columns, the first signifies nonsegmented operation; the second, short segmented operation; and the third, long segmented operation.

Arithmetic

Instruction Mnemonic	Hexadecimal Code							Flags C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
ADC/B Rd,Rs	B	5/4	Rs	Rd			W	‡	‡	‡	‡	*	*	5	5	5			
							B	‡	‡	‡	‡	0	‡						
ADD/B/L Rd,#data	0/0/1	1/0/6	0	Rd	data	data (L only)	W	‡	‡	‡	‡	*	*	4	4	4	8	8	8
ADD/B/L Rd,Rs	8/8/9	1/0/6	Rs	Rd			B	‡	‡	‡	‡	0	‡	7	7	7	14	14	14
ADD/B/L Rd,@Rs	0/0/1	1/0/6	Rs ≠ 0	Rd			L	‡	‡	‡	‡	*	*	7	7	7	14	14	14
ADD/B/L Rd,addr	4/4/5	1/0/6	0	Rd	addr									9	10	12	15	16	18
ADD/B/L Rd,addr(Rs)	4/4/5	1/0/6	Rs ≠ 0	Rd	addr									10	10	13	16	16	19
DAB Rbd	B	0	Rd	0				‡	‡	‡	*	*	*	5	5	5			
DEC/B Rd,#n	A	B/A	Rd	n				*	‡	‡	‡	*	*	4	4	4			
DEC/B @Rd,#n	2	B/A	Rd ≠ 0	n										11	11	11			
DEC/B addr,#n	6	B/A	0	n	addr									13	14	16			
DEC/B addr(Rd),#n	6	B/A	Rd ≠ 0	n	addr									14	14	17			
DIV/L Rd,Rs	9	B/A	Rs	Rd				‡	‡	‡	‡	*	*	107	107	107	744	744	744
DIV/L Rd,#data	1	B/A	0	Rd	data	data (L only)								107	107	107	744	744	744
DIV/L Rd,@Rs	1	B/A	Rs ≠ 0	Rd										107	107	107	744	744	744
DIV/L Rd,addr	5	B/A	0	Rd	addr									108	109	111	745	746	748
DIV/L Rd,addr(Rs)	5	B/A	Rs ≠ 0	Rd	addr									109	109	112	746	746	749
EXTS/B/L Rd	B	1	Rd	A/0/7				*	*	*	*	*	*	11	11	11	11	11	11
INC/B Rd,#n	A	9/8	Rd	n				*	‡	‡	‡	*	*	4	4	4			
INC/B @Rd,#n	2	9/8	Rd ≠ 0	n										11	11	11			
INC/B addr,#n	6	9/8	0	n	addr									13	14	16			
INC/B addr(Rd),#n	6	9/8	Rd ≠ 0	n	addr									14	14	17			
MULT/L Rd,Rs	9	9/8	Rs	Rd				‡	‡	‡	‡	*	*	70	70	70	282	282	282
MULT/L Rd,#data	1	9/8	0	Rd	data	data (L only)								70	70	70	282	282	282
MULT/L Rd,@Rs	1	9/8	Rs ≠ 0	Rd										70	70	70	282	282	282
MULT/L Rd,addr	5	9/8	0	Rd	addr									71	72	74	283	284	286
MULT/L Rd,addr(Rs)	5	9/8	Rs ≠ 0	Rd	addr									72	72	75	284	284	287
														(plus 7 cycles for each 1 in the absolute value of the low-order 16 bits of the multiplicand)					
NEG/B Rd	8	D/C	Rd	2				‡	‡	‡	‡	*	*	7	7	7			
NEG/B @Rd	0	D/C	Rd ≠ 0	2										12	12	12			
NEG/B addr	4	D/C	0	2	addr									15	16	18			
NEG/B addr(Rd)	4	D/C	Rd ≠ 0	2	addr									16	16	19			
SBC/B Rd,Rs	B	7/6	Rs	Rd			W	‡	‡	‡	‡	*	*	5	5	5			
							B	‡	‡	‡	‡	1	‡						
SUB/B/L Rd,Rs	8/8/9	3/2/2	Rs	Rd			W	‡	‡	‡	‡	*	*	4	4	4	8	8	8
SUB/B/L Rd,#data	0/0/1	3/2/2	0	Rd	data	(L only)	B	‡	‡	‡	‡	1	‡	7	7	7	14	14	14
SUB/B/L Rd,@Rs	0/0/1	3/2/2	Rs ≠ 0	Rd			L	‡	‡	‡	‡	*	*	7	7	7	14	14	14
SUB/B/L Rd,addr	4/4/5	3/2/2	0	Rd	addr									9	10	12	15	16	18
SUB/B/L Rd,addr(Rs)	4/4/5	3/2/2	Rs ≠ 0	Rd	addr									10	10	13	16	16	19

Bit Manipulation

Instruction Mnemonic	Hexadecimal Code						Flags C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
BIT/B Rd,b	A	7/6	Rd	b			*	‡	*	*	*	*	4	4	4			
BIT/B @Rd,b	2	7/6	Rd ≠ 0	b									8	8	8			
BIT/B addr,b	6	7/6	0	b	addr								10	11	13			
BIT/B addr(Rd),b	6	7/6	Rd ≠ 0	b	addr								11	11	14			
BIT/B Rd,Rs	2	7/6	0	Rs	0 Rd	0 0							10	10	10			
RES/B Rd,#b	A	3/2	Rd	b			*	*	*	*	*	*	4	4	4			
RES/B @Rd,#b	2	3/2	Rd ≠ 0	b									11	11	11			
RES/B addr,#b	6	3/2	0	b	addr								13	14	16			
RES/B addr(Rd),#b	6	3/2	Rd ≠ 0	b	addr								14	14	17			
RES/B Rd,Rs	2	3/2	0	Rs	0 Rd	0 0							10	10	10			
SET/B Rd,#b	A	5/4	Rd	b			*	*	*	*	*	*	4	4	4			
SET/B @Rd,#b	2	5/4	Rd ≠ 0	b									11	11	11			
SET/B addr,#b	6	5/4	0	b	addr								13	14	16			
SET/B addr(Rd),#b	6	5/4	Rd ≠ 0	b	addr								14	14	17			
SET/B Rd,Rs	2	5/4	0	Rs	0 Rd	0 0							10	10	10			
TSET/B Rd	8	D/C	Rd	6			*	*	1	*	*	*	7	7	7			
TSET/B @Rd	0	D/C	Rd ≠ 0	6									11	11	11			
TSET/B addr	4	D/C	0	6	addr								14	15	17			
TSET/B addr(Rd)	4	D/C	Rd ≠ 0	6	addr								15	15	18			

Comparison

Instruction Mnemonic	Hexadecimal Code							C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
CP B/L Rd,Rs	8/8/9	B/A/0	Rs	Rd				‡	‡	‡	‡	‡	‡	4	4	4	8	8	8
CP B/L Rd,#data	0/0/1	B/A/0	0	Rd	data	data (L only)								7	7	7	14	14	14
CP B/L Rd,@Rs	0/0/1	B/A/0	Rs ≠ 0	Rd										7	7	7	14	14	14
CP B/L Rd,addr	4/4/5	B/A/0	0	Rd	addr									9	10	12	15	16	18
CP B/L Rd,addr(Rs)	4/4/5	B/A/0	Rs ≠ 0	Rd	addr									10	10	13	16	16	19
CP B @Rd,#data	0	D/C	Rd ≠ 0	1	data									11	11	11			
CP B addr,#data	4	D/C	0	1	addr	data								14	15	17	14	15	17
CP B addr(Rd),#data	4	D/C	Rd ≠ 0	1	addr	data								15	15	18	15	15	18
CPD B Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	8	0	r	Rd cc	X	‡	X	‡	*	*	20	20	20			
CPDR B Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	C	0	r	Rd cc	X	‡	X	‡	*	*	(11 + 9n cycles)					
CPI Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	0	0	r	Rd cc	X	‡	X	‡	*	*	20	20	20			
CPIR B Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	4	0	r	Rd cc	X	‡	X	‡	*	*	(11 + 9n cycles)					
CPSD B @Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	A	0	r	Rd ≠ 0 cc	‡	‡	‡	‡	*	*	25	25	25			
CPSDR B @Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	E	0	r	Rd ≠ 0 cc	‡	‡	‡	‡	*	*	(11 + 14n cycles)					
CPSI B @Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	2	0	r	Rd ≠ 0 cc	‡	‡	‡	‡	*	*	25	25	25			
CPSIR B @Rd,@Rs,r,cc	B	B/A	Rs ≠ 0	6	0	r	Rd ≠ 0 cc	‡	‡	‡	‡	*	*	(11 + 14n cycles)					

CPU Control

Instruction Mnemonic	Hexadecimal Code						C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
COMFLG flags	8	D	flags	5			‡	‡	‡	‡	*	X	7	7	7			
DI VI/NVI	7	C	0	2/1			*	*	*	*	*	*	7	7	7			
EI VI/NVI	7	C	0	6/5			*	*	*	*	*	*	7	7	7			
HALT	7	A	0	0			*	*	*	*	*	*	(8 + 3n cycles)					
LDCTLB FLAGS,Rbs	8	C	Rs	9			‡	‡	‡	‡	‡	‡	7	7	7			
LDCTLB Rbd,FLAGS	8	C	Rd	1			*	*	*	*	*	*	7	7	7			
LDPS @Rs	3	9	Rs ≠ 0	0			‡	‡	‡	‡	‡	‡	12	12	12			
LDPS addr	7	9	0	0	addr								16	20	22			
LDPS addr(Rs)	7	9	Rs ≠ 0	0	addr								17	20	23			
MBIT	7	B	0	A			*	X	‡	*	*	*	7	7	7			
MREQ Rd	7	B	Rd	D			*	‡	‡	*	*	*	(12 + 7n cycles)					
MRES	7	B	0	9			*	*	*	*	*	*	5	5	5			
MSET	7	B	0	8			*	*	*	*	*	*	7	7	7			
NOP	8	D	0	7			*	*	*	*	*	*	7	7	7			
RESFLG flags	8	D	flags	3			‡	‡	‡	‡	*	*	7	7	7			
SETFLG flags	8	D	flags	1			‡	‡	‡	‡	*	*	7	7	7			

Data Movement

Instruction Mnemonic	Hexadecimal Code						C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
CLR/B Rd	8	D/C	Rd	8			*	*	*	*	*	*	7	7	7			
CLR/B @Rd	0	D/C	Rd ≠ 0	8									8	8	8			
CLR/B addr	4	D/C	0	8	addr								11	12	14			
CLR/B addr(Rd)	4	D/C	Rd ≠ 0	8	addr								12	12	15			
EX/B Rd,Rs	A	D/C	Rs	Rd			*	*	*	*	*	*	6	6	6			
EX/B Rd,@Rs	2	D/C	Rs ≠ 0	Rd									12	12	12			
EX/B Rd,addr	6	D/C	0	Rd	addr								15	16	18			
EX/B Rd,addr(Rs)	6	D/C	Rs ≠ 0	Rd	addr								16	16	19			
LD/B/L Rd,Rs	A/A'9	1/0/4	Rs	Rd			*	*	*	*	*	*	3	3	3	5	5	5
LD/B/L Rd,#data	2/2'1	1/0/4	0	Rd	data								7	7	7	11	11	11
LD/B/L Rd,@Rs	2/2'1	1/0/4	Rs ≠ 0	Rd									7	7	7	11	11	11
LD/B/L Rd,addr	6/6'5	1/0/4	0	Rd	addr								9	10	12	12	13	15
LD/B/L Rd,addr(Rs)	6/6'5	1/0/4	Rs ≠ 0	Rd	addr								10	10	13	13	13	16
LD/B/L Rd,Rs(disp)	3	1/0/5	Rs ≠ 0	Rd	disp								14	14	14	17	17	17
LD/B/L Rd,Rs(Rx)	7	1/0/5	Rs ≠ 0	Rd	0 Rx ≠ 0	0 0							14	14	14	17	17	17
LD/B/L @Rd,Rs	2/2'1	F/E/D	Rd ≠ 0	Rs			*	*	*	*	*	*	8	8	8	11	11	11
LD/B/L addr,Rs	6/6'5	F/E/D	0	Rs	addr								11	12	14	14	15	17
LD/B/L addr(Rd),Rs	6/6'5	F/E/D	Rd ≠ 0	Rs	addr								12	12	15	15	15	18
LD/B/L Rd(#disp),Rs	3	3/2'7	Rd ≠ 0	Rs	disp								14	14	14	17	17	17
LD/B/L Rd(Rx),Rs	7	3/2'7	Rd ≠ 0	Rs	0 Rx ≠ 0	0 0							14	14	14	17	17	17
LD Rd,#data	2	1	0	Rd	data		*	*	*	*	*	*	7	7	7			
LDB Rd,#data	C	Rd	data	data									5					
LDL Rd,#data	1	4	0	Rd	data	data							7	7	7			
LD/B @Rd,#data	0	D/C	Rd ≠ 0	5	data		*	*	*	*	*	*	11	11	11			
LD/B addr,#data	4	D/C	0	5	addr	data							14	15	17			
LD/B addr(Rd),#data	4	D/C	Rd ≠ 0	5	addr	data							15	15	18			
LDA Rd,addr	7	6	0	Rd	addr		*	*	*	*	*	*	12	13	15			
LDA Rd,addr(Rs)	7	6	Rs ≠ 0	Rd	addr								13	13	16			
LDA Rd,Rs(#disp)	3	4	Rs ≠ 0	Rd	disp								15	15	15			
LDA Rd,Rs(Rx)	7	4	Rs ≠ 0	Rd	0 Rx ≠ 0	0 0							15	15	15			
LDAR Rd,addr	3	4	0	Rd	disp		*	*	*	*	*	*	15	15	15			
LDD/B @Rs,@Rd,r	B	B/A	Rs ≠ 0	9	0 r	Rd ≠ 0 8	*	X	*	‡	*	*	20	20	20			
LDDR/B @Rs,Rd,r	B	B/A	Rs ≠ 0	9	0 r	Rd ≠ 0 0	*	X	*	1	*	*	(11 + 9n cycles)					
LDI/B @Rd,@Rs,r	B	B/A	Rs ≠ 0	1	0 r	Rd ≠ 0 8	*	X	*	‡	*	*	20	20	20			
LDIR/B @Rd,@Rs,r	B	B/A	Rs ≠ 0	1	0 r	Rd ≠ 0 0	*	X	*	1	*	*	(11 + 9n cycles)					
LDK Rd,#data	B	D	Rd	data			*	*	*	*	*	*	5	5	5			
LDM Rd,@Rs,#n	1	C	Rs ≠ 0	1	0 Rd	0 num	*	*	*	*	*	*	11	11	11			
LDM Rd,addr,#n	5	C	0	1	0 Rd	0 num addr							14	15	17	(plus 3n		
LDM Rd,addr(Rs),#n	5	C	Rs ≠ 0	1	0 Rd	0 num addr							15	15	18	cycles)		
LDM @Rd,Rs,#n	1	C	Rd ≠ 0	9	0 Rs	0 num							11	11	11			
LDM addr,Rs,#n	5	C	0	9	0 Rs	0 num addr							14	15	17	(plus 3n		
LDM addr(Rd),Rs,#n	5	C	Rd ≠ 0	9	0 Rs	0 num addr							15	15	18	cycles		
LDR/B/L Rd,addr	3	1/0/5	0	Rd ≠ 0	disp		*	*	*	*	*	*	14	14	14	17	17	17
LDR/B/L addr,Rs	3	3/2'7	0	Rs ≠ 0	disp								14	14	14	17	17	17
LDR/B addr(Rd),#data	4	D/C	Rd ≠ 0	5	addr	data							15	15	18			
POP/L Rd,@Rs	9	7/5	Rs ≠ 0	Rd			*	*	*	*	*	*	8	8	8	12	12	12
POP/L @Rd,@Rs	1	7/5	Rs ≠ 0	Rd ≠ 0									12	12	12	19	19	19
POP/L addr,@Rs	5	7/5	Rs ≠ 0	0	addr								16	16	19	22	23	26
POP/L addr(Rd),@Rs	5	7/5	Rs ≠ 0	Rd ≠ 0	addr								16	16	19	23	23	26
PUSH/L @Rd,Rs	9	3/1	Rd ≠ 0	Rs			*	*	*	*	*	*	9	9	9	12	12	12
PUSH/L @Rd,#data	0	D	Rd ≠ 0	9	data								12	12	12	19	19	19
PUSH/L @Rd,@Rs	1	3/1	Rd ≠ 0	Rs ≠ 0									13	13	13	20	20	20
PUSH/L @Rd,addr	5	3/1	Rd ≠ 0	0	addr								14	14	17	21	21	24
PUSH/L @Rd,addr(Rs)	5	3/1	Rd ≠ 0	Rs ≠ 0	addr								14	14	17	21	21	24

Input/Output

Instruction Mnemonic	Hexadecimal Code								Flags C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
INI/B Rd,@Rs	3	D/C	Rs ≠ 0	Rd					*	*	*	*	*	*	10	10	10			
INI/B Rd,port	3	B/A	Rd	4	port															
SINI/B Rd,port	3	B/A	Rd	5	port															
IND/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	8	0	r	Rd ≠ 0	8	*	*	*	‡	*	*	21	21	21			
SIND/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	9	0	r	Rd ≠ 0	8												
INDR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	8	0	r	Rd ≠ 0	0	*	X	*	1	*	*	(11 + 10n cycles)					
SINDR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	9	0	r	Rd ≠ 0	0												
INI/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	0	0	r	Rd ≠ 0	8	*	X	*	1	*	*	21	21	21			
SINI/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	1	0	r	Rd ≠ 0	8												
INIR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	0	0	r	Rd ≠ 0	0	*	*	*	1	*	*	(11 + 10n cycles)					
SINIR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	1	0	r	Rd ≠ 0	0												
OUT/B @Rd,Rs	3	F/E	Rd ≠ 0	Rs					*	*	*	*	*	*	10	10	10			
OUT/B port,Rs	3	B/A	Rs	6	port										12	12	12			
SOUT/B port,@Rs	3	B/A	Rs ≠ 0	7	port															
OUTD/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	A	0	r	Rd ≠ 0	8	*	X	*	‡	*	*	21	21	21			
SOUTD/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	B	0	r	Rd ≠ 0	8												
OTDR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	A	0	r	Rd ≠ 0	0	*	X	*	1	*	*	(11 + 10 cycles)					
SOTDR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	B	0	r	Rd ≠ 0	0												
OUTI/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	2	0	r	Rd ≠ 0	8	*	X	*	‡	*	*	21	21	21			
SOUTI/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	3	0	r	Rd ≠ 0	8												
OTIR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	2	0	r	Rd ≠ 0	0	*	X	*	1	*	*	(11 + 10 cycles)					
SOTIR/B @Rd,@Rs,r	3	B/A	Rs ≠ 0	3	0	r	Rd ≠ 0	0												

Logical

Instruction Mnemonic	Hexadecimal Code						Flags C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
AND/B Rd,Rs	8	7/6	Rs	Rd			W	*	‡	‡	*	*	*	4	4	4			
AND/B Rd,#data	0	7/6	0	Rd	data		B	*	‡	‡	‡	*	*	7	7	7			
AND/B Rd,@Rs	0	7/6	Rs ≠ 0	Rd										7	7	7			
AND/B Rd,addr	4	7/6	0	Rd	addr									9	10	12			
AND/B Rd,addr(Rs)	4	7/6	Rs ≠ 0	Rd	addr									10	10	13			
COM/B Rd	8	D/C	Rd	0			W	*	‡	‡	*	*	*	7	7	7			
COM/B @Rd	0	D/C	Rd ≠ 0	0			B	*	‡	‡	‡	*	*	12	12	12			
COM/B addr	4	D/C	0	0	addr									15	16	18			
COM/B addr(Rd)	4	D/C	Rd ≠ 0	0	addr									16	16	19			
OR/B Rd,Rs	8	5/4	Rs	Rd			W	*	‡	‡	*	*	*	4	4	4			
OR/B Rd,#data	0	5/4	0	Rd	data		B	*	‡	‡	‡	*	*	7	7	7			
OR/B Rd,@Rs	0	5/4	Rs ≠ 0	Rd										7	7	7			
OR/B Rd,addr	4	5/4	0	Rd	addr									9	10	12			
OR/B Rd,addr(Rs)	4	5/4	Rs ≠ 0	Rd	addr									10	10	13			
TEST/B/L Rd	8/8/9	D/C/C	Rd	4/4/8			W	*	‡	‡	*	*	*	7	7	7	13	13	13
TEST/B/L @Rd	0/0/1	D/C/C	Rd ≠ 0	4/4/8			B	*	‡	‡	‡	*	*	8	8	8	13	13	13
TEST/B/L addr	4/4/5	D/C/C	0	4/4/8	addr		L	*	‡	‡	X	*	*	11	12	14	16	17	19
TEST/B/L addr(Rd)	4/4/5	D/C/C	Rd ≠ 0	4/4/8	addr									12	12	15	17	17	20
TCC/B cc,Rd	A	F/E	Rd	cc				*	*	*	*	*	*	5	5	5			
XOR/B Rd,Rs	8	9/8	Rs	Rd			W	*	‡	‡	*	*	*	4	4	4			
XOR/B Rd,#data	0	9/8	0	Rd	data		B	*	‡	‡	‡	*	*	7	7	7			
XOR/B Rd,@Rs	0	9/8	Rs ≠ 0	Rd										7	7	7			
XOR/B Rd,addr	4	9/8	0	Rd	addr									9	10	12			
XOR/B Rd,addr(Rs)	4	9/8	Rs ≠ 0	Rd	addr									10	10	13			

Program Control

Instruction Mnemonic	Hexadecimal Code					C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
CALL @Rd	1	F	Rd ≠ 0	0		*	*	*	*	*	*	10	15	15			
CALL addr	5	F	0	0	addr							12	16	21			
CALL addr(Rd)	5	F	Rd ≠ 0	0	addr							13	18	21			
CALR addr	D	displacement				*	*	*	*	*	*	10	10	15			
DJNZ/DBJNZ r,disp	F	r	1/0	displacement		*	*	*	*	*	*	11	11	11			
IRET	7	B	0	0		‡	‡	‡	‡	‡	‡	13	13	16			
JP cc,@Rd	1	E	Rd ≠ 0	cc		*	*	*	*	*	*	(10 if taken, 7 if not					
JP cc,addr	5	E	0	cc	addr							7	8	10			
JP cc,addr(Rd)	5	E	Rd ≠ 0	cc	addr							8	8	11			
JR cc,addr	E	cc	displacement			*	*	*	*	*	*	6	6	6			
RET cc	9	E	0	cc		*	*	*	*	*	*	10	10	13 (Return taken			
SC #src	7	F	src			*	*	*	*	*	*	33	33	39			

Rotate and Shift

Instruction Mnemonic	Hexadecimal Code					C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
RL/B Rd,#n	B	3/2	Rd	0 = 1 bit 2 = 2 bits		‡	‡	‡	‡	*	*	(6 for n = 1, 7 for n = 2					
RLC/B Rd,#n	B	3/2	Rd	8 = 1 bit A = 2 bits		‡	‡	‡	‡	*	*	(6 for n = 1, 7 for n = 2					
RLDB Rbl,Rbs	B	E	Rbs	Rbl		*	‡	X	*	*	*	9	9	9			
RR/B Rd,#n	B	3/2	Rd	4 = 1 bit 6 = 2 bits		‡	‡	‡	‡	*	*	(6 for n = 1, 7 for n = 2					
RRC/B	B	3/2	Rd	D = 1 bit E = 2 bits		‡	‡	‡	‡	*	*	(6 for n = 1, 7 for n = 2					
RRDB Rbl,Rbs	B	C	Rbs	Rbl		*	‡	X	*	*	*	9	9	9			
SDA/B/L Rd,Rs	B	3/2/3	Rd	B/B/F	0 Rs 0 0	‡	‡	‡	‡	*	*	(15 + 3n cycles					
SDL/B/L Rd,Rs	B	3/2/3	Rd	3/3/7	0 Rs 0 0	‡	‡	‡	X	*	*	(15 + 3n cycles					
SLA/B/L Rd,#b	B	3/2/3	Rd	9/9/D	b	‡	‡	‡	‡	*	*	(13 + 3n cycles					
SLL/B/L Rd,#b	B	3/2/3	Rd	1/1/5	b	‡	‡	‡	X	*	*	(13 + 3n cycles					
SRA/B/L Rd,#b	B	3/2/3	Rd	9/9/D	-b	‡	‡	‡	0	*	*	(13 + 3n cycles					
SRL/B/L Rd,#b	B	3/2/3	Rd	1/1/5	-b	‡	‡	‡	X	*	*	(13 + 3n cycles					

Translation

Instruction Mnemonic	Hexadecimal Code					C	Z	S	P/V	D	H	Cycle Times Word, Byte NS	SS	SL	Long Word NS	SS	SL
TRDB @Rd,@Rs,r	B	8	Rd ≠ 0	8	0 r Rs 0	*	X	*	‡	*	*	25	25	25			
TRDRB @Rbd,@Rbs,r	B	8	Rd ≠ 0	C	0 r Rs 0	*	X	*	1	*	*	(11 + 14n cycles					
TRIB @Rd,@Rs,r	B	8	Rd ≠ 0	0	0 r Rs 0	*	X	*	‡	*	*	25	25	25			
TRIRB @Rd,@Rs,r	B	8	Rd ≠ 0	4	0 r Rs 0	*	X	*	1	*	*	(11 + 14n cycles					
TRTDB @Rs1,@Rs2,r	B	8	Rs1 ≠ 0	A	0 r Rs2 ≠ 0 0	*	‡	*	‡	*	*	25	25	25			
TRTDRB @Rs1,@Rs2,r	B	8	Rs1 ≠ 0	E	0 r Rs2 ≠ 0 E	*	‡	*	‡	*	*	(11 + 14n cycles					
TRTIB @Rs1,@Rs2,r	B	8	Rs1 ≠ 0	2	0 r Rs2 ≠ 0 0	*	‡	*	‡	*	*	25	25	25			
TRTIRB @Rs1,@Rs2,r	B	C	Rs1 ≠ 0	6	0 r Rs2 ≠ 0 E	*	‡	*	‡	*	*	(11 + 14n cycles					

Upper Nibble of Upper Instruction Byte	Lower Nibble of Upper Instruction Byte							
	0	1	2	3	4	5	6	7
0	ADDB	ADD	SUBB	SUB	ORB	OR	ANDB	AND
1	CPL	PUSHL	SUBL	PUSH	LDL	POPL	ADDL	POP
2	LDB	LD	RESB	RES	SETB	SET	BITB	BIT
3	LDB LDRB	LD LDR	LDB LDRB	LD LDR	LDA LDAR	LDL LDRL	RSVD	LDL LDRL
4	ADDB	ADD	SUBB	SUB	ORB	OR	ANDB	AND
5	CPL	PUSHL	SUBL	PUSH	LDL	POPL	ADDL	POP
6	LDB	LD	RESB	RES	SETB	SET	BITB	BIT
7	LDB	LD	LDB	LD	LDA	LDL	LDA	LDL
8	ADDB	ADD	SUBB	SUB	ORB	OR	ANDB	AND
9	CPL	PUSHL	SUBL	PUSH	LDL	POPL	ADDL	POP
A	LDB	LD	RESB	RES	SETB	SET	BITB	BIT
B	DAB	EXTS EXTSB EXTSL	T.4*	T.4*	ADCB	ADC	SBCB	SBC
C	LDB ----------							LDB
D	CALR ----------							CALR
E	JR ----------							JR
F	DJNZ DBJNZ ----------							DJNZ DBJNZ

*See appropriate table following.

Upper Nibble of Upper Instruction Byte	Lower Nibble of Upper Instruction Byte							
	8	9	A	B	C	D	E	F
0	XORB	XOR	CPB	CP	T.1*	T.1*	EPI	EPI
1	MULTL	MULT	DIVL	DIV	T.2*	LDL	JP	CALL
2	INCB	INC	DECB	DEC	EXB	EX	LDB	LD
3	RSVD	LDPS	T.3*	T.3*	INB	IN	OUTB	OUT
4	XORB	XOR	CPB	CP	T.1*	T.1*	EPI	EPI
5	MULTL	MULT	DIVL	DIV	T.2*	LDL	JP	CALL
6	INCB	INC	DECB	DEC	EXB	EX	LDB	LD
7	RSVD	LDPS	HALT	T.7*	EI DI	T.7*	RSVD	SC
8	XORB	XOR	CPB	CP	T.1*	T.1*	EPI	EPI
9	MULTL	MULT	DIVL	DIV	T.2*	RSVD	RET	RSVD
A	INCB	INC	DECB	DEC	EXB	EX	TCCB	TCC
B	T.5*	RSVD	T.6*	T.6*	RRDB	LDK	RLDB	RSVD
C	LDB ----------							LDB
D	CALR ----------							CALR
E	JR ----------							JR
F	DJNZ ----------							DJNZ DBJNZ

*See appropriate table following.

NOTE: EPI means Extended Processor Instruction.

Table 1
Upper Instruction Byte

Lower Nibble of Lower Instruction Byte	0C	0D	4C	4D	8C	8D
0	COMB	COM	COMB	COM	COMB	COM
1	CPB	CP	CPB	CP	LDCTLB	SETFLG
2	NEGB	NEG	NEGB	NEG	NEGB	NEG
3	RSVD	RSVD	RSVD	RSVD	RSVD	RESFLG
4	TESTB	TEST	TESTB	TEST	TESTB	TEST
5	LDB	LD	LDB	LD	RSVD	COMFLG
6	TSETB	TSET	TSETB	TSET	TSETB	TSET
7	RSVD	RSVD	RSVD	RSVD	RSVD	NOP
8	CLRB	CLR	CLRB	CLR	CLRB	CLR
9	RSVD	PUSH	RSVD	RSVD	LDCTLB	RSVD

Table 2
Upper Instruction Byte

Lower Nibble of Upper Instruction Byte	1C	5C	9C
1	LDM	LDM	RSVD
8	TESTL	TESTL	TESTL
9	LDM	LDM	RSVD

Upper Instruction Byte

Upper Nibble of Upper Instruction Byte	Table 3		Table 4		Table 5	Table 6		Table 7	
	3A	3B	B2	B3	B8	BA	BB	7B	7D
0	INIB INIRB	INI INIR	RLB	RL	TRIB	CPIB	CPI	IRET	RSVD
1	SINIB SINIRB	SINI SINIR	SLLB SRLB	SLL SRL	RSVD	LDIB LDIRB	LDI LDIR	RSVD	RSVD
2	OUTIB OTIRB	OUTI OUTIR	RLB	RL	TRTIB	CPSIB	CPSI	RSVD	LDCTL
3	SOUTIB SOTIRB	SOUTI SOTIR	SDLB	SDL	RSVD	RSVD	RSVD	RSVD	LDCTL
4	INB	IN	RRB	RR	TRIRB	CPIRB	CPIR	RSVD	LDCTL
5	SINB	SIN	RSVD	SLLL SRLL	RSVD	RSVD	RSVD	RSVD	LDCTL
6	OUTB	OUT	RRB	RR	TRTIRB	CPSIRB	CPSIR	RSVD	LDCTL
7	SOUTB	SOUT	RSVD	SDLL	RSVD	RSVD	RSVD	RSVD	LDCTL
8	INDB INDRB	IND INDR	RLCB	RLC	TRDB	CPDB	CPD	MSET	RSVD
9	SINDB SINDRB	SIND SINDR	SLAB SRAB	SLA SRA	RSVD	LDDB LDDRB	LDD LDDR	MRES	RSVD
A	OUTDB OTDRB	OUTD OTDR	RLCB	RLC	TRTDB	CPSDB	CPSD	MBIT	LDCTL
B	SOUTDB SOTDRB	SOUTD SOTDR	SDAB	SDA	RSVD	RSVD	RSVD	RSVD	LDCTL
C	RSVD	RSVD	RRCB	RRC	TRDRB	CPDRB	CPDR	RSVD	LDCTL
D	RSVD	RSVD	RSVD	SLAL	RSVD	RSVD	RSVD	MREQ	LDCTL
E	RSVD	RSVD	RRCB	RRC	TRTDRB	CPSDRB	CPSDR	RSVD	LDCTL
F	RSVD	RSVD	RSVD	SDAL	RSVD	RSVD	RSVD	RSVD	LDCTL

APPENDIX B

SUMMARY OF PROGRAMMABLE DEVICES

B.1 General

Appendix B summarizes the programming aspects of the major programmable devices that the System 8000 employs. These devices, which are listed below, perform many important functions throughout the system. For more detailed information about not only the listed devices, read the associated manuals whose part numbers are listed below with the devices.

Z80 Counter Timer Circuit (CTC)	03-0036
Z80 Parallel I/O (PIO)	03-0008
Z80 Serial I/O (SIO)	03-3033
Z80 Direct Memory Access Device (DMA)	00-2013

Another source of information about not only these Zilog devices, but all Zilog devices is the current Zilog Data Book, part number: 00-2034.

B.2 Programming The Z80 CTC

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

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

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

A control byte is identified by a 1 in bit 0. A 0 in bit 2 means that a time constant byte follows. Interrupt vectors are always addressed to Channel 0, and identified by a 0 in bit 0.

Addressing. During programming, channels are addressed with channel select pins CS1 and CS2. A two-bit binary code selects the appropriate channel as shown in the following table:

Channel	CS1	CS2
0	0	0
1	0	0
2	1	1
3	1	1

RESET. The CTC has both hardware and software resets. The hardware reset terminates all down-counts and disables all CTC interrupts by resetting the interrupt bits in the control registers. In addition, the ZC/TO and interrupt outputs go inactive, IEO reflects IEI, and D0-D7 go to the high-impedance state. All channels must be completely reprogrammed after a hardware reset.

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

If the channel control byte has both bits D1 and D2 set to 1, the addressed channel stops operating, pending a new time constant byte. The channel is ready to resume operation after the new constant is programmed. In timer mode, if D3=0, operation is triggered automatically when the time constant byte is loaded.

CHANNEL CONTROL BYTE PROGRAMMING. The channel control byte is shown in Figure B-1. It sets the modes and parameters described in this Appendix.

INTERRUPT ENABLE. D7 enables the interrupt to generate an interrupt output (INT) at zero count. Interrupts can be prorammed in either mode and can be enabled or disabled at any time.

OPERATING MODE. D6 selects either timer or counter mode.

PRESCALER FACTOR. (Timer Mode Only). D5 selects the factor--either 16 or 256.

TRIGGER SLOPE. D4 selects the active edge or slope of the CLK/TRG input pulses. Reprogramming the CLK/TRG slope during operation is equivalent to issuing an active edge. If the trigger slope is changed by a control word update while a channel is pending operation in timer mode, the result is the same as CLK/TRG pulse and the timer starts. Similarily, if the channel is in counter mode, the counter decrements.

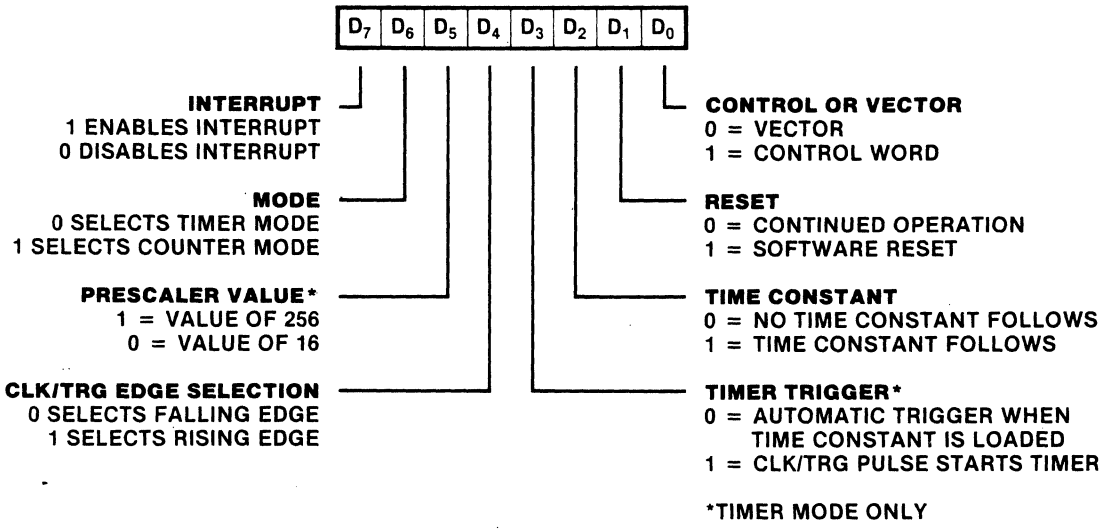


Figure B-1. Channel Control Word

TRIGGER MODE (TIMER MODE ONLY). D3 selects the trigger mode for timer operation. When D3 is reset to 0, the timer is triggered automatically. The time constant byte is programmed during an I/O write operation, that takes one manchine cycle. At the end of the write operation there is a setup delay of one clock period. The timer starts automatically (decrements) on the rising edge of the second clock pulse (T2) of the machine cycle following the write operation. Once started, the timer runs continuously. At zero count the timer is loaded automatically and continues counting without interruption or delay, until stopped by a reset.

When D3 is set to 1, the timer is triggered externally through the CLK/TRG input. The time constant word is programmed during an I/O write operation, one machine cycle. The timer is ready for operation on the rising edge of the second clock pulse (T2) of the following machine cycle. The first timer decrement follows the active edge of the CLK/TRG pulse by a delay time of one clock cycle if a minimum setup time to the rising edge of clock is met. If this minimum is not met, the delay is extended by another clock period. Consequently, for immediate triggering, the CLK/TRG input must precede T2 by one clock cycle plus its minimum setup time. If the minimumm time is not met, the timer starts on the third clock cycle (T3).

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

TIME CONSTANT TO FOLLOW. A one in D2 indicates that the next byte addressed to the selected channel is a time constant data byte for the time constant register. The time constant byte can be written at any time.

A 0 in D2 indicates no time constant byte is to follow. This is ordinarily used when the channel is already in operation and the new channel control byte is an update. A channel does not operate without a time constant value. The only way to write a time constant value is to write a control byte with D2 set.

SOFTWARE RESET. Setting D1 to one causes a software reset, which is described in the Reset section.

CONTROL BYTE. Setting D0 to one identifies the byte as a control byte.

TIME CONSTANT PROGRAMMING. Before a channel can start counting it must receive a time constant byte from the CPU. During programming or reprogramming, a channel control word in which bit 2 is set must precede the time constant byte to indicate that the next byte is a time constant. The time constant byte can be any from 1 to 256. Note that 0016 is interpreted as 256.

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

- Φ The system clock period (0)
- Φ The prescaler factor (P) that multiplies the interval by either 16 or 256

- The time constant (T) that is programmed into the time constant register

Consequently, the time interval is the product of 0xPxT. The minimum timer resolution is 16x0 (4 us with a 4MHz clock). The maximum timer interval is 256x x256(16.4 ms with a 4MHz clock). For longer intervals, timers can be cascaded. (Refer to Figure B-2.)

INTERRUPT VECTOR PROGRAMMING. If the Z80 CTC has one or more interrupts enabled, it can supply interrupt vectors to the Z80 CPU. To do so, the Z80 CTC must be preprogrammed with the most-significant five bits of the interrupt vector. Programming consists of writing a vector word to the I/O port corresponding to the Z80 CTC Channel 0. Note tht D0 of the vector word is always zero, to distinguish the vector from a channel control word. Dl and D2 are not used in programming the vector word. These bits are supplied by the interrupt service with a unique interrupt vector (Figure B-3). Channel 0 has the highest priority.

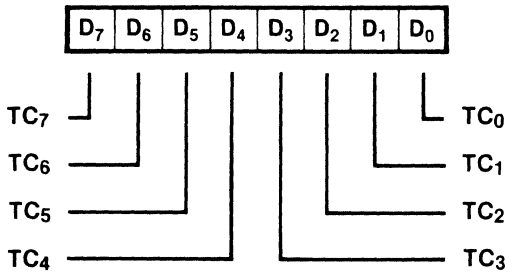


Figure B-2. Time Constant Word

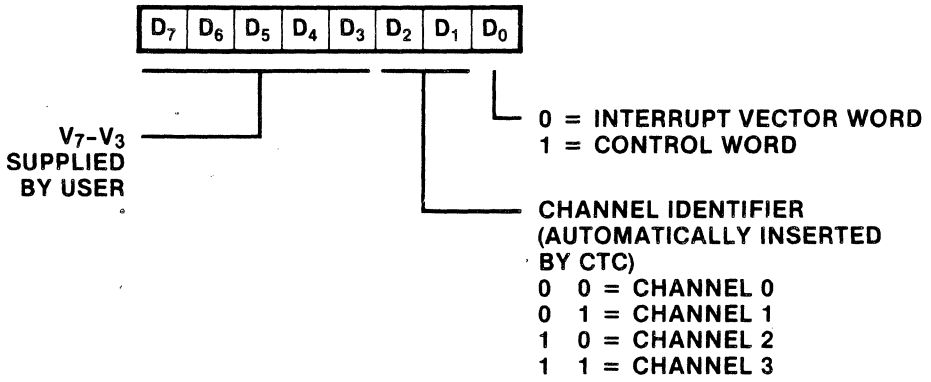


Figure B-3. Interrupt Vector Word

B.3 Programming the Z80 PIO

Mode 0, 1, or 2. IByte Input, Output, or Bidirectional. Programming a port for Mode 0, 1, or 2 requires two words per port. These words are:

A Mode Control Word. Selects the port operating mode (Figure B-4). This word can be written any time.

An Interrupt Vector. The Z-80 PIO is designed for use with the Z-80 CPU in interrupt Mode 2 (Figure B-5). When interrupts are enabled, the PIO must provide an interrupt vector.

Mode 3. Bit Input/Output. Programming a port for Mode 3 operation requires a control word, a vector (if interrupts are enabled), and three additional words, described as follows:

I/O REGISTER CONTROL. When Mode 3 is selected, the mode control word must be followed by another control word that sets the I/O control register. This in turn defines what port lines are inputs and which are outputs (Figure B-6).

INTERRUPT CONTROL WORD. In Mode 3, handshake is not used. Interrupts are generated as a logic function of the input signal levels. The interrupt control word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available: AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits to the active level, an interrupt is triggered). Bit D6 sets the logic function, (Figure B-7). The active level of the input bits can be set either high or low. The active level is controlled by Bit D5.

MASK CONTROL WORD. This word sets the mask control register, allowing any unused bits to be masked off. If any bits are to be masked, D4 must be set. When D4 is set, the next word written to the port must be a mask control word (Figure B-8).

INTERRUPT DISABLE. There is one other control word that can be used to enable or disable a port interrupt, without changing the rest of the interrupt control word (Figure B-9).

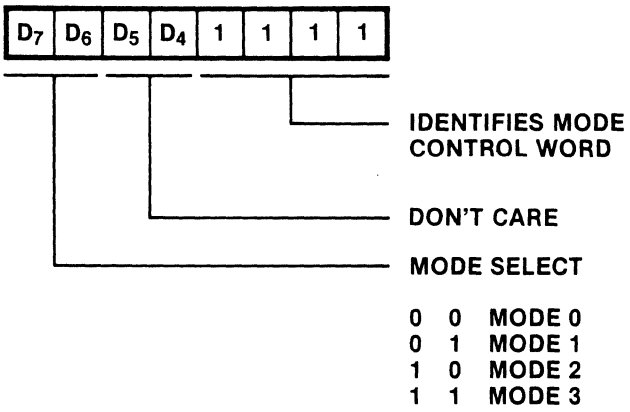


Figure B-4. PIO Mode Control Word

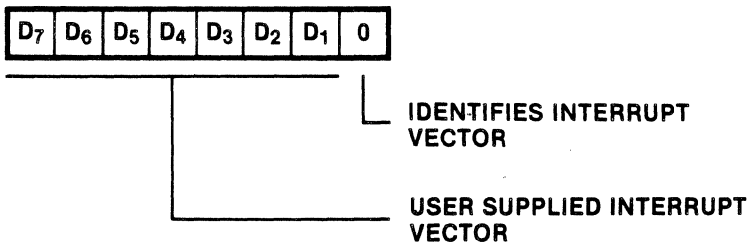


Figure B-5. PIO Interrupt Vector Word

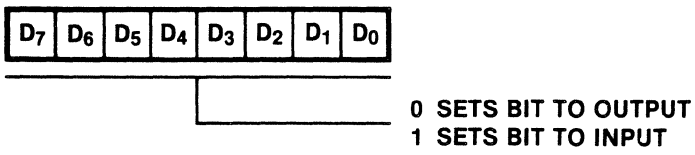


Figure B-6. PIO I/O Register Control Word

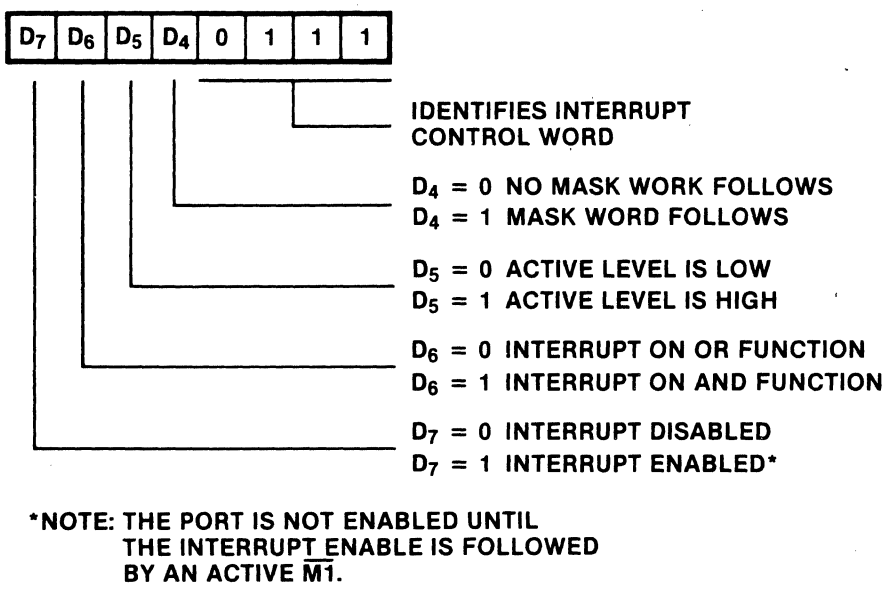


Figure B-7. PIO Interrupt Control Word

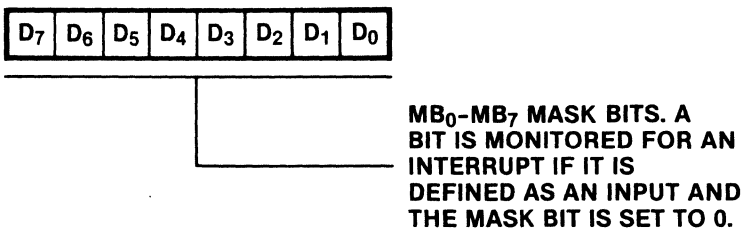


Figure B-8. PIO Mask Control Word

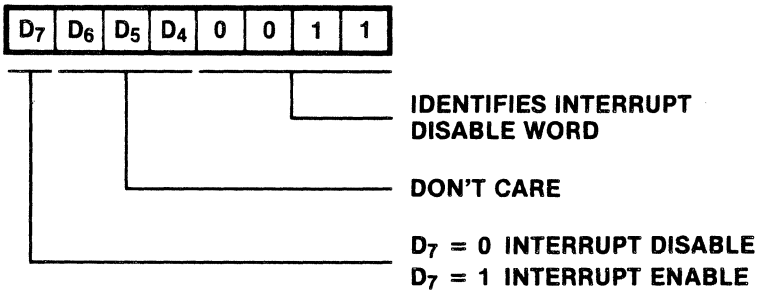


Figure B-9. PIO Interrupt Disable Word

B.4 Programming the Z80 SIO

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

Both channels contain registers that must be programmed via the system program prior to operation. The channel-select input (B/A) and the control/data input (C/D) are the command-structure addresssing controls, and are normally controlled by the CPU address bus.

READ REGISTERS. The SIO contains three read registers for Channel B and two read registers for Channel A (RR0-RR2 in Figure B-10). This register can be read to obtain the status information; RR2 contains the internally-modifiable interrupt vector and is only the Channel B register set. The status information includes error conditions, interrupt vector and standard communications-interface signals.

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

addressed read register can be read by the CPU.

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

WRITE REGISTER. The SIO contains eight write registers for Channel B and seven write registers for Channel A (WR0-WR7 in Figure B-11) that are programmed separately to configure te functional personality of the channels; WR2 contains the interrupt vector for both channels and is only the Channel B register set. When the exception of WR0, programming the write registers requires two bytes. The first byte is to WR0 and contains three bits (D0-D2) that point to the selected register; the second byte is the actual control word that is written into the register to configure the SIO.

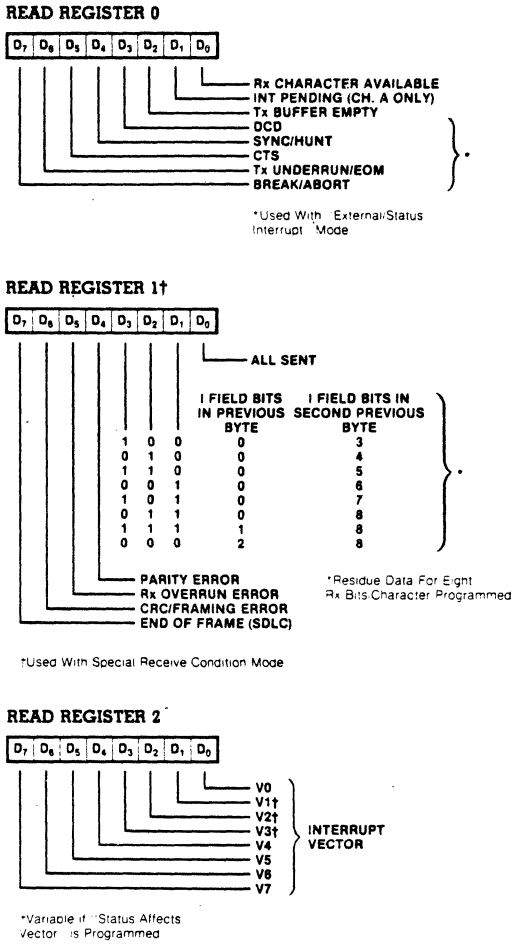


Figure B-10. SIO Read Register Bit Functions

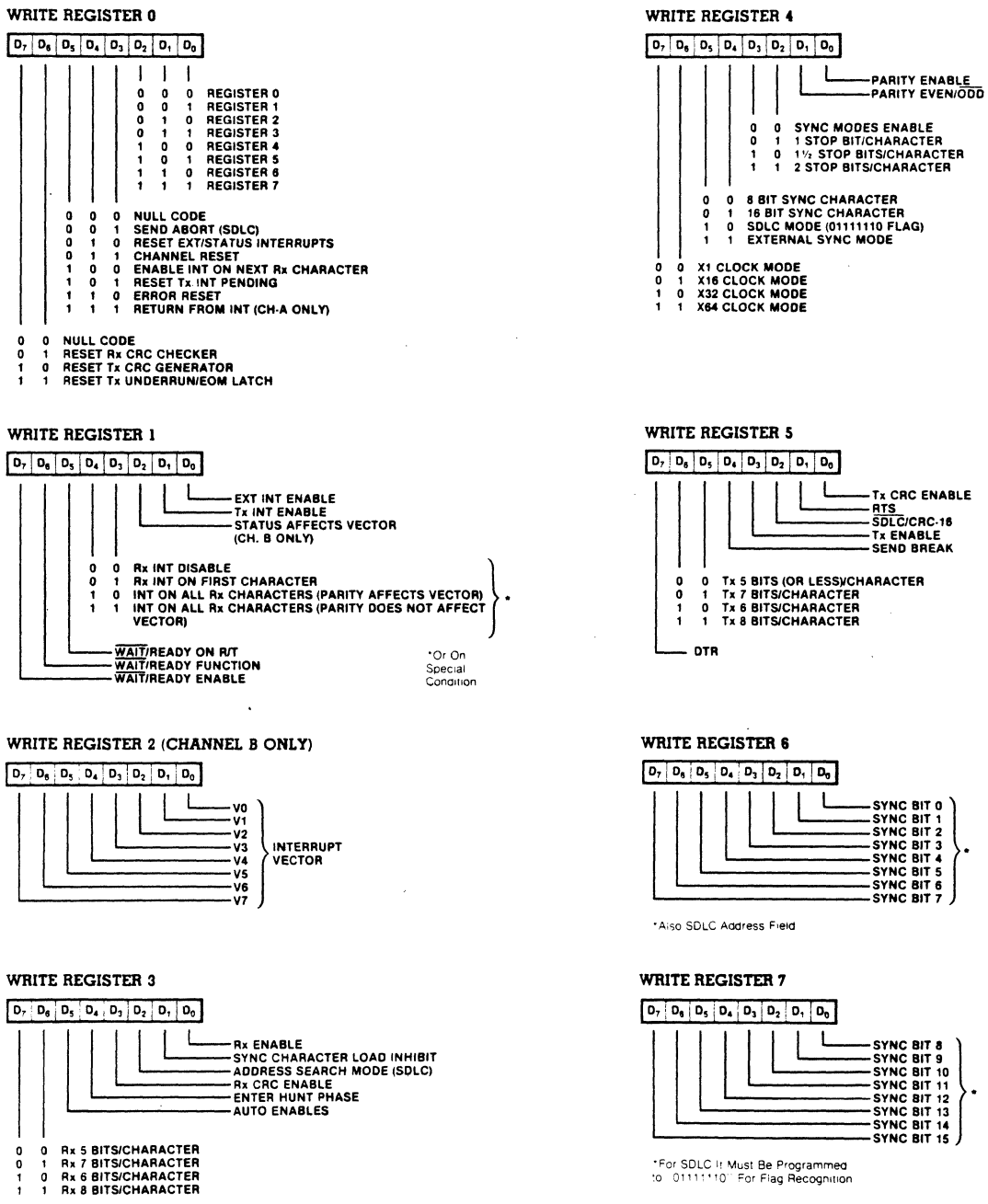


Figure B-11. SIO Write Registers Bit Functions

WR0 is a special case because all of the basic commands can be written to it with a single byte. Reset (internal or external) initializes the pointer bits D0-D2 to point to WR0. This implies that a channel reset must not be combined with the pointing to any register.

APPENDIX C

WINCHESTER DISK CONTROLLER COMMANDS

C.1 General

Appendix C describes all the commands that the host CPU sends to the controller through the command registers. Each description indentifies the registers that the host CPU uses. Bits 0 through 4 of the command byte form the command field. Refer to Section 4 for additional information. The term CP (Command Port) means command register as used in Section 4. For example, CP0 is command register xx00 and CP1 is command register xx01.

C.2 Format / UNIT /

The controller formats an entire disk drive with a single controller command. The CPU sends the FORMAT command to the controller by writing the unit number of the drive to be formatted as CP1 and by issuing the Format command to the command port, CP0. Format writes the data field of each sector with a pattern of alternating ones and zeros (0AA hexadecimal) and rewrites each sector's header. After formatting the target unit, the entire format is read back and the header field of each sector is validated. If an invalid header is found, the Unrecoverable Error bit in the operation ending status byte is set, together with the Verify Failure bit in the Operation Error Status Byte.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

C.3 Read Sector / UNIT / HEAD / CYLINDER / SECTOR / / WORD COUNT / ADDRESS

The Read Sector command requires both a disk address [unit, head, cylinder, sector] and a host system buffer address [word count, address]. Less than one sector of information is transferred if the word count is less than a sector (512 bytes) in length. The controller command for multiple sector reads and single sector reads is equivalent with the word count determining the number of sectors for the transfer.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

/ HEAD / Head is written to the controller through CP2.

/ CYLINDER / The target cylinder number is written to the controller through command ports CP3 and CP4. The low order eight bits of the ten-bit cylinder address are issued to the controller through CP3 and the high order two bits are issued through CP4 bits zero and one.

/ SECTOR / The sector number of the sector to be read is passed to the controller through the sector register, CP5.

/ WORD COUNT / The number of words to read from the disk is sent to the controller through the word count control register pair, CP8 and CP9. A word count greater than a single sector length is interpreted as a request for a multiple sector read.

/ ADDRESS / The 24-bit address of the host data buffer is passed to the controller through a register triplet; CP10, CP11, and CP12.

C.4 Write Sector / UNIT / HEAD / CYLINDER / SECTOR / / WORD COUNT / ADDRESS/

The Write Sector command requires both a disk address [unit, head, cylinder, sector] and a host system buffer address [word count, address]. Less than one sector of data is read from host memory if the word count is less than a sector in length. The controller command for multiple sector writes and single sector writes is equivalent to the word count determining the number of sectors for the transfer.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

/ HEAD / Head is written to the controller through CP2.

/ CYLINDER / The target cylinder number is written to the controller through command ports CP3 and CP4. The low order eight bits of the ten-bit cylinder address are issued to the controller through CP3 and the high order two bits are issued through CP4 bits zero and one.

/ SECTOR / The sector number of the sector to be read is sent to the controller through the sector register, CP5.

/ WORD COUNT / The number of words to be written to the disk is sent to the controller through the word count control register pair, CP8 and CP9. A word count greater than a single sector length is interpreted as a request for a multiple sector write.

/ ADDRESS / The 24-bit address of the host data buffer is sent to the controller through a register triplet; CP10, CP11, and CP12.

C.5 Read Detailed Status / UNIT / ADDRESS /

The Read Detailed Status command transfers three words of the detailed disk status for unit number UNIT to the host system memory beginning at location ADDRESS. The act of reading detailed status clears the operation error status byte unless the read status command itself times out. In this case the time-out error will be set. See Table 11 for the error status byte definition.

/ UNIT / Unit is written to the controller through CP1. Any value between 0 and 3 is valid.

/ ADDRESS / The 24-bit address of the host data buffer is sent to the controller through a register triplet; CP10, CP11, and CP12.

C.6 Restore / UNIT /

The Restore command recalibrates the drive seek circuitry by positioning the heads at track zero, and clears the fault status bit in the drive. The restore operation executes at a slower rate than a seek to track zero. Recalibrate should only be used in response to a drive fault. The controller automatically provides a restore and command retry when a drive fault occurs unless that option has been reselected.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

C.7 Null

The null command presents an operation ending status code "operation complete" and performs no disk or controller operation.

C.8 Seek / UNIT / CYLINDER /

The controller generates an automatic seek before disk read and write operations. This feature can be deselected under host software control. The host system can opt to defeat the automatic seek before read/write and issue its own seek commands. Deselection of the automatic seek facility has

the side affect of deselecting overlapped seeks in the controller.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

/ CYLINDER / The target cylinder number is written to the controller through command ports CP3 and CP4. The low order eight bits of the ten-bit cylinder address are issued to the controller through CP3 and the high order two bits are issued through CP4 bits zero and one.

C.9 Set Strobe/Offset / UNIT / SO /

The Set Strobe/Offset command is used to select a value for either or both the data strobe timing or the head positioning during data read operations. Both the data strobe and head offset are reset during disk write operations as required by the disk drive circuitry. This command reduces the sector read time dramatically on media that require a strobe/offset value to recover the data. The set strobe/offset command is useful with removable media since it can correct small differences between head alignment and/or data strobe times between drives. It is of questionable merit for use with fixed media. The Strobe/Offset value is input to the controller through the low byte of CP1, the HEAD select register.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

/ SO / The Strobe/Offset value is issued to the controller by command register CP2.

C.10 Set Write Protect / UNIT / SURFACES /

The Set Write Protect command allows each unit to be selectively write-protected at the controller. Write commands to a protected surface are inhibited at the controller.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

C.11 Format Read / UNIT / HEAD / CYLINDER / SECTOR / / ADDRESS /

When the Format Read command is issued, the host diagnostics can read an entire sector, both sector header information and data.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

/ HEAD / Head is written to the controller through CP2.

/ CYLINDER / The target cylinder number is written to the controller through command ports CP3 and CP4. The low order eight bits of the ten bit cylinder address are issued to the controller through CP3 and the high order two bits are issued through CP4 bits zero and one.

/ SECTOR / The sector number of the sector to be read is sent to the controller through the sector register, CP5.

/ ADDRESS / The 24-bit address of the host data buffer is passed to the controller through a register triplet; CP10, CP11, and CP12.

C.12 Set Interrupt Address / SECTOR /

The Set Interrupt Address allows the host system to specify the low order eight bits of the address for interrupt transfer. That address and an eight-bit Operation Ending Status byte are sent to the address bus lines when the controller issues an interrupt. Typically, the Set Interrupt Address command is issued once shortly after the controller is powered up.

/ SECTOR / The sector number of the sector to be read is sent to the controller through the sector register, CP5.

C.13 Self Test

This command forces the controller to execute its self diagnostics. Both the command reject and the command accepted bits are set in the command status byte if a self test error is detected. It is possible that the self test error will preclude the setting of the bits in the command status byte.

C.14 Format Verify / UNIT / HEAD / CYLINDER /

This is a diagnostic command that verifies the integrity of the disk format on a track. The controller reads each individual sector of the track and ensures that its header is good. If a format error is detected, the Verification Error bit is set in the Operation Error Status Byte and the Unrecoverable Error on Operation bit in the Operation Ending Status byte is set.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

/ HEAD / Head is written to the controller through CP2.

/ CYLINDER / The target cylinder number is written to the controller through command ports CP3 and CP4. The low order eight bits of the ten-bit cylinder address are issued to the controller through CP3 and the high order two bits are issued through CP4 bits zero and one.

C.15 Unit Format Verify / UNIT /

This is a diagnostic command. It is used to verify the format of an entire unit. If a format error is detected, the Verification Error bit is set in the Operation Error Status Byte, and the Unrecoverable Error on Operation bit in the Operation Ending Status Byte is also set. The cylinder where the failure occurred is read back by the Read Detailed Status command.

/ UNIT / Unit is written to the controller through CP1. Any value between zero and three is valid.

C.16 Command Notes

An explicit format write command is not provided in the controller's command repertoire. All disk write operations write the sector address field as well as the data. Therefore, a write sector command performs a format write operation.

APPENDIX D

CARTRIDGE TAPE ERROR CONDITIONS

D.1 General

Table D-1 lists the error conditions that can result from the commands that the tape controller receives from the host CPU. The table lists the commands, the resulting conditions, and the status bits that the conditions set.

Table D-1. Cartridge Tape Error Conditions

COMMAND RECEIVED	CONDITION	STATUS BITS SET
Initialization	Tape (if present) rewinds for more than 88 sec	INTV and HWERR
DIAG	ROM checksum error	INTV and HWERR
DIAG	Fifo test error	INTV and HWERR
DIAG	Handshake test failed	INTV and HWERR
Any command	Host wrote to ports in byte or double word mode	CMDREJ
Any undefined command		CMDREJ and INVAL
Any command except DIAG	Drive busy, or drive not selected	CMDREJ, INTV, INAP, and HWERR
Any command except DIAG, MRTRY, SEL, MODE	No tape	CMDREJ, INTV, and NOTAP
Any command except DIAG, MRTRY, SEL, LOAD, MODE	Tape not logically loaded	CMDREJ and INAP
WRITE, WFM EGP	Tape write-protected	CMDREJ, INAP, and PROT
LOAD	Tape alread logically loaded	CMDREJ and INAP

Table D-1. Cartridge Tape Error Conditions (continued)

COMMAND RECEIVED	CONDITION	STATUS BITS SET
READ, WRITE, SKBF, SKFF, EGP	Tape at logical end of tape	CMDREJ, INAP, and LEOT
SKBR, SKFR, REWIND	Tape at logical beginning of tape	CMDREJ, INAP, and LBOT
READ, WRITE	DMA buffer length greater than or equal to 32K bytes (k=1024 bytes)	CMDREJ and BPARM
READ, WRITE	DMA start address and DMA length greater than 24 bits	CMDREJ and BPARM
READ	DMA buffer length not even (bit 0=0)	CMDREJ and BPARM
READ	Blank tape (more than 48 inches) encountered	DATERR and BLKTAP
READ	Attempted buffer overflow during DMA	OVERFL
READ, WRITE, WFM	Bad read (read after write for WRITE and WFM) as indicated by a bad CRCC after retrying the operation the maximum permissible number of times	DATERR
READ	File mark encountered	CMDREJ and FMDET
READ, WRITE	Fifo error (overflow or underrun) after retrying the operation the maximum permissable number of times	FFERR
WRITE	Deck stopped taking data during write	INTV and HWERR
READ, WRITE, WFM	One or more retry attempts made	RTRYAT and number of retries in low byte of status 1 register

Table D-1. Cartridge Tape Error Conditions (continued)

COMMAND RECEIVED	CONDITION	STATUS BITS SET
Any command except READ, WRITE, WFM		Number of retries in low byte of status 1 register = 0
WRITE, WFM	Encountered end of tape before WRITE began or retry after error pushes the beginning of the block past the end of tape.	CMDREJ, INAP, and LEOT (and RTRYAT and number of retries in low byte of status 1 register if retries attempted)
SKBF, SKFF	Encounterd logical end of tape	SKNDE and LEOT and number of blocks/ files skipped in high byte of status 1.
SKBR, SKFR	Encountered logical beginning of tape	SKNDNE and LBOT and number of blocks/ files skipped in high byte of status 1.
SKBF, SKFF	Blank tape (more than 48 inches) encountered	SKNDNE and BLKTAP and number of blocks/ files skipped in high byte of status 1 register.
SKBF, SKBR	File mark encountered	SKNDNE and FMDET and and number of blocks skipped in high byte of status 1 register.
SKBF, SKBR, SKFF, SKFR		Number of blocks/ files skipped in high byte of status 1 register.
Any command except SKBF, SKBR, SKFF, SKFR		High byte of status 1 register = 0.

Table D-1. Cartridge Tape Error Conditions (continued)

COMMAND RECEIVED	CONDITION	STATUS BITS SET
READ, WRITE, SKBF, SKFF, UNLD, REWIND, STRK	Rewind takes more 88 than seconds. Rewind occurs whenever track boundaries are crossed.	INTV and HWERR
SKFR	During forward motion (after FM detected) blank tape or end of tape encountered	INTV and HWERR
SKBR, SKFR, READ WRITE, WFM	During forward motion (after track boundary) tape failed to move off BOT for greater than 166 ms	INTV and HWERR
Any command	Tape write-protected	PROT
Any command	Tape at logical load point	LLP
Any command	Tape at end of tape	LEOT
Any command		UNIT0, 1 and TRK0, 1 set to indicate unit and track selected. TRK0, 1 = 0 if drive not logically loaded
READ, WRITE, WFM, SKBF, SKBR, SKFF, SKFR	Data detected for > 40" (32K bytes)	INTV and HWERR
REWIND, STRK, READ, WRITE, WFM, EGP, SKBF, SKBR, SKFF, SKFR	Tape moves forward > 2.1 sec and fails to move from Bot to LPS	INTV and HWERR
READ, WRITE, WFM, SKBR, SEL, MODE	No block found where one is know to exist	INTV and HWERR
READ, WRITE, WFM	During retry blank tape found while moving in reverse	INTV and HWERR

APPENDIX E

SADIE TEST DESCRIPTIONS

This appendix gives detailed information for the diagnostic tests contained on the System 8000 Diagnostic Tape, SADIE (Part Number 14-0009). The following test descriptions are arranged in the same order as they appear in the CHOOSE A TEST OR CONTROL LINE menu:

WDCCRC	TEX
WDCFMT	NEWMEM1
CYLSPARE	NEWMEM2
WDCMEDIA	NEWMEM3
WDCTST3	MMUTST5
WDCTST7	CENT.PRT
WDCMON	DP.PRT
TCUMON	S16SIO
TCOM	SI0MODEM

TEST NAME PAGE

TEST NAME

WDCCRC - a non-destructive verification of the Winchester Disk Data, Cyclic Redundancy Checks (CRCs)

PARAMETERS

Disk drive to be tested (default=0)

DESCRIPTION

WDCCRC reads all tracks of the selected drive. If an error is detected, each sector of the track is read. WDCCRC repeats n times, where n is the #REPS in the test line.

ERROR MESSAGES

UNCORRECTABLE ERROR

The disk controller returned an uncorrectable error (hard error). The cylinder, head, and sector number of the bad sector are given.

CRC ERROR

The data field of the sector number displayed contains a CRC error.

CONTROLLER SEEK ERROR

The header field of the sector prior to the sector displayed, contains a CRC error or the wrong information.

FULL TRACK READ

The track, beginning at the cylinder head of the sector displayed, had an error when the whole track was read; however, no errors were found when each sector was read individually.

LAP SUMMARY

A lap summary is displayed when each repetition of WDCDRC is completed. The lap summary includes:

- The lap number.
- The number of hard errors.
- The number of invalid commands issued to the disk controller.

- The number of warnings issued by the disk controller.
- The number of times each disk controller command was issued during the test.
- A tally of the number of times each of the following status bits were returned by the disk controller when a hard error occurred:

```
TOT  -- total number of errors
DAT  -- CRC errors
POS  -- sector not found
FOR  -- format error (sector header
        field error)
NR   -- unit not ready
SVE  -- servo error
RWF  -- read or write fault
SPE  -- speed error
PL   -- power loss
WPT  -- write protected
DSE  -- seek error
NCL  -- not on cylinder
GB   -- guard band error
PLE  -- PLO error
UNS  -- unsafe
DCE  -- invaid command
DTO  -- timeout error
P/M  -- POR/MR
ADE  -- address error
DF   -- drive fault
NOL  -- drive not on-line
CTO  -- controller operation timed out
CWP  -- write protect error
VF   -- verify failure
BD   -- bad disk (713 bad cylinders)
CRDT -- can't read defect table
ME   -- map error
```

- The last eight error messages of the lap.

For detailed information the disk controller commands and status registers, refer to the Winchester Disk Controller Hardware Reference Manual (03-3203).

NOTES

WDCCRC does not recognize if the drive selected for testing is on-line, or if the disk medium is present in the drive. WDCCRC will display an error message for each sector it attempts to read on the nonexistent drive. The same results occur when the drive and disk

medium are present but the disk is locked. To recover, press the system START button, then respond with `^' to the PAUSE menu.

TEST NAME

WDCFMT - data destructive formatting of the entire disk

PARAMETERS

Disk drive to be tested (default=0)

DESCRIPTION

WDCFMT formats every sector of the disk. This is accomplished by issuing the format command, FMT, to the Winchester Disk Controller. WDCFMT reformats the disk n times, where n = #REPS in the test line.

ERROR MESSAGES

Error Indicated During Format

The message DONE=XXXX, appears when an uncorrectable error bit is returned in the disk controller's status register, at the completion of the format command. The 'XXXX' is replaced by the operation ending status byte value.

CRC Error

The data field of the sector number shown had a CRC error.

Controller Seek Error

The header field of the sector just before the sector shown had a CRC error, or the sector shown had wrong header information.

Not CSG or CRC

An error other than CRC ERROR or CONTROLLER SEEK ERROR occurred during the FMT command.

LAP SUMMARY

The lap summary displayed on completion of each test repetition shows the same type of information as the WDCCRC test.

NOTES

If the START button is pressed while the FMT command is being executed, requests by the user to skip to the

next test line, or return to the COMMAND LEVEL, will not be honored until the FMT command completes execution. The FMT command runs for approximately 90 minutes. To stop this test, press the RESET button.

If the drive selected for testing does not exist, the FMT command will still execute, and display the error message:

"NOT CSE OR CRC"

TEST NAME

WDCMEDIA - data-destructive, write-read test of the Winchester disk, using several data patterns

PARAMETERS

disk drive to be tested (default=0)

DESCRIPTION

WDCMEDIA exercises the disk medium by writing four separate data patterns to each sector of the medium. After each data pattern is written, the disk is read for verification. All errors incurred while writing or reading a data pattern are displayed. The four data patterns are:

1. 'AAAA'
2. '5555'
3. 'FFFF'
4. The 'worst case' pattern 'B6DB6DB6DB6D'

ERROR MESSAGES

Uncorrectable Error

The disk controller returned an uncorrectable (hard) error code. The cylinder, head, and sector numbers are displayed.

CRC Error

The data field of the sector number displayed had a CRC error.

Controller Seek Error

The header field of the sector, just before the sector number displayed, had a CRC error. Also, the sector number displayed, had the wrong information in the header field.

Not CSE or CRC

An error occurred other than a CRC error, or a Controller Seek Error.

Full Track Read
The track beginning at the disk address displayed, had an error when the entire track was read; however no errors were encountered when each sector on the track was read individually.

LAP SUMMARY
At the end of each test repetition, a lap summary is desplayed showing cumulative statistics for all repetitions. The lap summary includes:

- The lap number.
- The error count.
- The number of invalid commands issued to the disk controller.
- The number of warnings issued by the disk controller.
- The number of times each disk controller command was issued during the test.
- A tally of the number of times each status bit, in each of the following disk controller registers, was set by a soft error. The registers are:

  the disk ready register
  the disk status register
  the operation error status register
  the self test status register

- A tally of the number of times each status bit, in each of the disk controller registers, was set by a hard error.

For detailed information the disk contoller commands and status registers, refer to the Winchester Disk Controller Hardware Reference Manual (03-3203).

NOTES
None

TEST NAME

WDCTST3 - a random queue test of the Winchester disk

PARAMETERS

disk drive to be tested (default=0)

DESCRIPTION

WDCTST3 creates a 128 element queue. Each queue member has the following three addresses:

1) a source buffer address
2) a destination buffer address
3) a disk address

Each source and destination buffer is 512 bytes long. The source buffers are in memory segment 1, and the destination buffers are in memory segment 2.

Initially, the test randomizes the source buffer and disk addresses of the 128 queue elements. For each element, the source buffer contents are written to the disk sector.

Next the test reads the sector at the disk address of each element, into the destination buffer. The source and destination buffers are then compared.

Finally, the disk addresses are again randomized to prevent subsequent test repetitions from testing the same disk sectors.

WDCTST3 repeats n times, where n=#REPS in the test line.

ERROR MESSAGES

Compare Error

The contents of the source and destination buffers did not match. The address and data of the source and destination buffers is displayed with the disk address.

Hard and Soft Errors Indicated Concurrently

Both hard and soft errors were reported concurrently during a disk transfer. The error is logged as a hard error in the lap summary table.

Error During Read Detailed Status Command

A hard or soft error occurred; however, the error statistics could not be logged in the lap summary table due to an error reading detailed status.

LAP SUMMARY

WDCTST3 tallies a number of statistics whenever a hard or soft error occurs. At the end of each test repetition, these statistics are displayed in tabular form. The lap summary includes:

- The lap number.
- The error count.
- The number of invalid commands issued to the disk controller.
- The number of warnings issued by the disk controller.
- The number of times each disk controller command was issued.
- A tally of the number of times each of the following status bits were returned by the disk controller when a hard error occurred;

  TOT -- total number of errors
  DAT -- CRC errors
  POS -- sector not found
  FOR -- format error (sector header field error)
  NR -- unit not ready
  SVE -- servo error
  RWF -- read or write fault
  SPE -- speed error
  PL -- power loss
  WPT -- write protected
  DSE -- seek error
  NCL -- not on cylinder
  GB -- guard band error
  PLE -- PLO error
  UNS -- unsafe
  DCE -- invaid command
  DTO -- timeout error
  P/M -- POR/MR
  ADE -- address error
  DF -- drive fault
  NOL -- drive not on-line
  CTO -- controller operation timed out

CWP -- write protect error
VF -- verify failure
BD -- bad disk (713 bad cylinders)
CRDT -- can't read defect table
ME -- map error of hard errors,

- A tally of hard errors, identical to the one just described.

For detailed informatioon on the disk controller commands, and status registers, refer to the Winchester Disk Controller Hardware Reference Manaul (03-3203).

NOTES
None

TEST NAME
WDCTST7 - a comprehensive, multisector, write-read-compare test of the Winchester disk

PARAMETERS
disk drive to be tested (default=0)

number of test loops equals the number of iterations per lap (default=0)

DESCRIPTION
WDCTST7 exercises the Winchester Disk Controller by doing large, variable sized writes and reads to random disk addresses. The test repeats n times, where n=#REPS in the test line. Each test contains 'loop' iterations of the following:

1. Fill segment 1-7 with 'AAAA's.
2. Fill a ramdomly chosen source buffer with random data.
3. Write the source buffer into a random disk address.
4. Read from the disk into a destination file.
5. Compare the source and destination buffers.
6. Fill the buffers with 'AAAA's.
7. Check segments 1-7 for all 'AAAA's to verify that the disk transfers did not corrupt locations outside the buffers.

ERROR MESSAGES

Compare Error
The source address and data, the destination address and data, and the disk address are shown.

Both Hard and Soft Errors Appear Concurrently
The error is tallied as a hard error in the lap summary table.

An Error Occurred During a Read Detailed Status Command
A hard or soft error occured; however, the error statistics could not be logged in the lap summary table because of an error reading detailed status.

LAP SUMMARY
WDCTST7 tallies statistics whenever a hard or soft error occurs. At the end of each test repetition, the statistics are displayed in tablular form. Refer to diagnostic test WDCTST3 for a description of the statistical table.

NOTES
WDCTST7 requires eight segments of memory.

TEST NAME
WDCMON

PARAMETERS
None

DESCRIPTION
WDCMON is an interactive monitor for the Winchester Disk Controller. The user interacts with the monitor by issuing commands and parameters in response to the prompt: CMD? In order to run this test, the user must know the following:

1. Commands are entered in upper case letters.

2. Numeric parameters are interpreted by the monitor as decimal, unless the digits are followed by an H (which indicates a hexadecimal number).

   Example: 100H = 256.

3. Missing parameters take on their previous values.

   ```
   Example: CMD?
            READ 1 1000H 100H 0 1 0
            CMD?
            READ
   ```

   The second READ and the first READ are identical commands.

4. A number in front of a command is a repeat factor.

5. Commands can be nested within command lines by using parenthesis. Each command on a command line must be parenthesized (unless there is only one command).

   ```
   Example: CMD?
            READ 1 1000H 100H 0 1 0
            CMD?
            10 ((READ) (ISEC 1))
   ```

   The first command reads from cyl:0, head:1, sector:0, 100H words into segment 1, offset 1000H.

The second command does the following:

a. Reads 1000H words from the disk address into the memory buffer.

b. Increments the sector number by 1 (carrying into head and cylinder numbers).

c. Decrements the loop counter (initially 10), and returns to step "a" if it is still positive.

6. If the user forgets the available commands, the command HELP will display the entire list of commands and parameters.

NOTES

WDCMON does not protect the user against commands that will destroy the memory resident SADIE code, the WDCMON code, the SADIE test catalog, and the test list catalog. These reside in segment 0.

COMMAND DESCRIPTIONS

The following list of commands are accepted by WDCMON. Command names are indicated in uppercase letters; parameters are indicated in lowercase letters.

COMMAND: BADCYL

Description

BADCYL reads the defect table residing at physical cylinder 0, and displays a list of bad physical cylinders.

Example

```
BADCYL
```

COMMAND: CEC/REC

Description

CEC clears all error counters; REC displays all error and instruction counters.

Example

```
REC
CEC
```

COMMAND: CEMODE, EXITCE

Description
CEMODE enters Customer Engineer mode. This disables defect mapping, and subsequent disk accesses will be to physical, not logical, cylinders. EXITCE exits Customer Engineer mode, enabling defect mapping.

Example
```
CEMODE
EXITCE
```

COMMAND: CLC/ILC/RLC

Description
CLC clears the lap counter; ILC increments the lap counter by 1; and RLC displays the lap counter.

Example
```
CLC
ILC
RLC
```

COMMAND: CMP srcseg srcoff desseg desoff count

Description
This command compares two buffers. Count is a word count.

Example
```
CMP 1 0 2 0 8000H
```

COMMAND: DISP dseg doff dlnth

Description
DISP displays dlnth words beginning at segment dseg and offset doff. Addresses appear at the left margin of the display. Words are displayed in hexadecimal.

Example
```
DISP 1 0 100H
```

COMMAND: DISPRT/ENPRT

Description
DISPRT disables the printing of operational messages from the monitor. ENPRT enables the printing of operational messages from the monitor.

Example
DISPRT

COMMAND: FMT unit

Description
This command will home the selected drive to cylinder 0, and clear any drive fault.

Example
FMT 0

COMMAND: FRD fseg roff cyl head sec unit

Description
This command is the same as a read, except, all disk format header and crc information is also transferred. This reads the exact image of a disk sector into the host memory. This command transfers only one sector.

Example
FRD 2 0 1 2 14H 0

COMMAND: HALT/NOHALT

Description
HALT enables all subsequent CMP errors to cause a halt until a <cr> is entered. NOHALT enhibits CMP to halt on an error.

Example
HALT
NOHALT

COMMAND: HELP

Description
HELP displays a list of the WDCMON commands and parameters.

Example
HELP

COMMAND: HOME unit

Description
This command will home the selected drive to cylinder 0, and clear any drive fault.

Example
HOME 0

COMMAND: INHIS/ENAIS

Description
INHIS sets the 'inhibit implicit seek' bit in each command to the WDC. ENAIS clears the 'inhibit implicit seek' bit.

Example
INHIS
ENAIS

COMMAND: INHRTY/ENTRY

Description
INHRTY sets the 'inhibit retry flag' in each command to the WDC. ENTRY clears the 'inhibit retry flag'. WDC.

Example
INHRTY
ENTRY

COMMAND: ININT/ENINT

Description
ENINT sets the `interrupt enable flag' in each command to the WDC. ININT clears the interrupt enable flag.

Example
```
ENINT
ININT
```

COMMAND: INV N

Description
This command forces any value of N as a command to the WDC. N must be less than 100H.

Example
```
INV 23H
```

COMMAND: IROFF, IWOFF, ILNTH, ICYL, IHEAD, ISEC value

Description
These commands are issued in the following format:

Ixxxx value

The selected variable is incremented by `value'. Incrementing sec beyond 23 causes a carry over to the head number. Incrementing HEAD beyond 2 causes a carry over into the CYLinder number. Incrementing ROFF or WOFF beyond FFFH causes a carry over into RSEG or WSEG.

Example
If CYL=0 HEAD=2 SEC=3, after entering ISEC 1, CYL=0 HEAD=2 SEC=4. If CYL=0 HEAD=2 SEC=23, after entering ISEC1, CYL=1 HEAD=0 SEC=0.

COMMAND: NULL

Description
NULL sends a NOP command to the WDC.

Example
NULL

COMMAND: Q, QUIT

Description
QUIT exits WDCMON

Example
QUIT
Q

COMMAND: RAND rndseg rndoff rlnth

Description
This fills rlnth WORDs of memory, starting at <rndseg>rndoff, with random data.

Example
RAND 1 0 8000H

COMMAND: RCYL, RHEAD, RSEC, RALL

Description
RCYL sets CYL=random value from 0 to 599, inclusive.
RHEAD sets HEAD=random value from 0 to 2, inclusive.
RSEC sets SEC=random value from 0 to 23, inclusive.
RALL does RCYL, RHEAD and RSEC all in one command.

Example
RCYL
RHEAD
RSEC
RALL

COMMAND: RDDT, RBDT

Description
RDDT issues a `read defect table' command to the WDC. RBDT issues a `rebuild defect table' command to the WDC.

Example
RDDT
RBDT

COMMAND: READ rseg roff lnth cyl head sec unit

Description
This command reads (lnth) words of data from unit#(unit) into memory <rseg>roff. The first disk address accessed is cyl-head-sec.

Example
READ 1 1000H 100H 23 2 20 0

COMMAND: SEEK cyl unit

Description
This command does an explicit seek, for the cylinder specified, on the selected unit.

Example
SEEK 240 0

COMMAND: SETOFF offset

Description
This command sets the strobe offset to the given value.

Example
SETOFF 1

COMMAND: SROFF, SWOFF, SLNTH, SCYL, SHEAD, SSEC, SUNIT value

Description

These commands are entered in the following format:
Sxxxx value

The selected variable is initialized to the given `value'.

Example

```
SROFF 1000H
SCYL 500
```

COMMAND: SRSEG, SWSEG segnum

Description

SRSEG sets the read segment number to the given segment number (segnum). SWSEG, sets the write segment number to the given `segnum'.

Example

```
SRSEG 1
SWSEG 3
```

COMMAND: STAT unit

Description

STAT returns the detailed disk controller status registers.

Example

```
STAT 0
```

COMMAND: WP unit, UNPROT unit

Description

WP does a software write protect on the selected unit. UNPROT disables the WP command.

Example

```
WP 0
UNPROT 1
```

COMMAND: WRITE wseg woff lnth cyl head sec unit

Description

This command writes (lnth) data words to unit#(unit) from memory <rseg>roff. The first disk address accessed is cyl-head-sec.

Example

WRITE 2 1024 100H 2 0 20 0

TEST NAME
TCUMON

PARAMETERS
None

DESCRIPTION
This program is an exercise monitor for the Tape Controller Unit.

All commands must be entered in uppercase letters. Some commands require no parameters; but those that do, interpret the values entered as a decimal number, unless followed by an H (hex number).

For some commands to execute properly, other commands must be issued first. These dependencies are described in the command descriptions.

The command line may contain multiple commands. Repetition counts can be specified for the commands, and parentheses are used to force command groupings. Some parameters are maintained from command to command.

NOTES
TCUMON allows almost complete control of the Tape Controller Unit. It is possible to transfer data into segment 0, and crash the current invocation of SADIE. Before using this monitor, it is advisable to be familiar with the Cartridge Tape Controller Hardware Reference Manual (03-3204).

EXAMPLES

1. LOAD

   This command loads the tape from the physical beginning-of-tape to the logical beginning-of-tape.

2. STRK 1

   This command performs a rewind and select track 1 (tracks are numbered 0-3).

3. WRITE 1 A000H 1000H

   If this command is executed after examples 1 and 2, it writes 1000 hex bytes of data, starting from location A000 of segment 1, onto track 1 of the tape.

4. 4((1WOFF 1000H) (WRITE)) (WFM)

If this command is executed after examples 1, 2, and 3, locations B000 through EFFF of segment 1 are written as the second through fifth blocks of track 1. Each block is 1000 hex bytes long, and there is a file mark terminating this file. This illustrates the use of parentheses, implied parameters, and repetition counts.

COMMAND DESCRIPTIONS

The following list of commands are used in TCUMON. All parameters, when applicable, are indicated as lowercase variable names. All variables are assumed to be hexadecimal numbers. If the conditions described under dependencies are not met, the command will be rejected.

COMMAND: CEC

Description
CEC clears all command, lap, and error counters.

Dependencies
None

COMMAND: CCC

Description
CCC clears all command counters.

Dependencies
None

COMMAND: CLC

Description
CLC clears the lap counter, and displays the command and error summary.

Dependencies
None

Example
```
CLC
```

COMMAND: CMP srcseg srcoff desseg desoff count

Description
This command compares two buffers. The memory locations are addressed by segments (srcseg) and (desseg), and offsets within segments (srcoff) and (desoff). The number of words compared is (count). The addresses and contents of the first nonmatching locations, if any, are displayed.

Dependencies
None

Example
```
CMP 1 0 1 100H 100H
```

COMMAND: DIAG1

Description
DIAG1 performs a quick check of the I/O ports that interface with the TCU.

Dependencies
None

Example
```
DIAG1
```

COMMAND: DISPRT

Description
DISPRT disables the echoing of commands, and prevents the displaying of the command and error summaries.

Dependencies
None

Example
DISPRT

COMMAND: EGP

Description
EGP erases a three inch gap on the tape. The purpose is to get past a defect in the media.

Dependencies
The tape must be at, or beyond, the logical beginning of tape.

Example
EGP

COMMAND: ENPRT

Description
ENPRT enables the echoing of commands, and prevents the displaying of the command and error summaries.

Dependencies
None

Example
ENPRT

COMMAND: F fseg foff flen pat

Description
This command fills memory from segment (fseg), address (foff), for a length of (flen) words, with data pattern (pat).

Dependencies
None

Example
F 1 8 50 1234H

COMMAND: HELP

Description
HELP displays the available commands with their parameters, and gives terse explanations of their use.

Dependencies
None

Example
HELP

COMMAND: ILC

Description
ILC increments the lap counter, and displays the command and error summaries.

Dependencies
None

Example
ILC

COMMAND: ILNTH addval

Description
This command increments the length counter for read or write operations by (addval) words. The length must be less than 0x8000.

Dependencies
None

Example
ILNTH 1000H

COMMAND: INV invcom

Description
This command forces any 2 digit hex number into the command register. This verifies that invalid commands are properly rejected.

Dependencies
None

Example
INV 2A

COMMAND: IROFF addval

Description
This command increments the destination offset for read operations by (addval) words. It also increments accross segment values. The segment value must be 3 or less.

Dependencies
None

Example
IROFF 80H

COMMAND: ITRK

Description
ITRK increments the track on which succeeding commands will operate, and sets the track. If on track 3, it sets track to 0.

Dependencies
Tape must be at, or beyond, the logical load point.

Example
ITRK

COMMAND: IVNT

Description
IVNT increments the unit on which succeeding commands will operate. If on unit 3, unit 0 is selected.

Dependencies
None

Example
IVNT

COMMAND: IWOFF addval

Description
This command increments the source offset address for write operations by (addval) words. It also increments across memory segment values. The segment value must be 3 or less.

Dependencies
None

Example
IWOFF 64

COMMAND: LOAD

Description
LOAD moves the tape from the physical beginning-of-tape to the logical beginning-of-tape.

Dependencies
Tape must be at the physical beginning-of-tape.

Example
LOAD

COMMAND: MODE m

Description
This command changes the mode of the tape controller to 0 or 1.

Dependencies
None

Example
MODE 1

COMMAND: MRTRY rtrycnt

Description
This command sets the maximum number of retries allowed for reads and writes. Default at power-up is 10 (rtrycnt = 0-15).

Dependencies
None

Example
MRTRY 8

COMMAND: Q

Description
This command quits TCUMON and returns to SADIE.

Dependencies
None

Example
Q

COMMAND: RAND rndseg rndoff rlnth

Description
This command fills memory with random data from segment (rndseg), address (rndoff), for a length of (rlen) words.

Dependencies
None

Example
RAND 3 0 1000H

COMMAND: READ rhad rlad rlen

Description
This command reads a block of data from tape and transfers it to segment (rhad), address (rlad). The (rlen) bytes are transferred to memory.

Dependencies
Tape must be at, or beyond, the logical beginning-of-tape.

Example
READ 0 B800H 2000H

COMMAND: REC

Description
REC displays all error counters, lap counters, and command counters.

Dependencies
None

Example
REC

COMMAND: REWIND

Description
This command rewinds the tape to the logical beginning of tape.

Dependencies
The tape must be at, or beyond, the logical beginning of tape.

Example
REWIND

COMMAND: SEL selcnt

Description
Controller selects a new drive (address selcnt, selcnt=0-3).

Dependencies
None

Example
SEL 1

COMMAND: SKBF skpcnt

Description
This command skips (skpcnt) blocks forward on the tape, or until either a file mark or the end of tape is detected.

Dependencies
Tape must be at, or beyond, the logical beginning-of-tape.

Example
SKBF 12

COMMAND: SKBR skpcnt

Description
This command skips (skpcnt) blocks backward on the tape, or until either a file mark or the logical beginning-of-tape is detected.

Dependencies
Tape must be beyond the logical beginning-of-tape.

Example
SKBR 14

COMMAND: SKFF skpcnt

Description
This command skips (skpcnt) files forward on the tape, or until the end of tape is detected.

Dependencies
Tape must be at, or beyond, the logical beginning-of-tape.

Example
SKFF 4

COMMAND: SKFR skpcnt

Description
Skips (skpcnt) files backward on the tape, or until the end of tape is detected.

Dependencies
Tape must be beyond the logical beginning of tape.

Example
SKFR 5

COMMAND: STAT

Description
STAT displays all tape controller interface registers.

Dependencies
None

Example
STAT

COMMAND: STRK trkcnt

Description
Controller rewinds the tape and selects a new track. (trkcnt = 0-3.)

Dependencies
Tape must be at, or beyond, the logical beginning-of-tape.

Example
```
STRK 3
```

COMMAND: UNLOAD

Description
UNLOAD moves the tape to the physical beginning-of-tape.

Dependencies
Tape must be at, or beyond, the logical beginning-of-tape.

Example
UNLOAD

COMMAND: WFM

Description
WFM writes a file mark on the tape.

Dependencies
The tape must be at, or beyond, the logical beginning-of-tape.

Example
WFM

COMMAND: WRITE whad wlad wlen

Description
This command writes one block of (wlen) bytes to the tape, from segment (whad), address (wlad).

Dependencies
Tape must be at, or beyond, the logical beginning-of-tape.

Example
WRITE 3 4000H 1A00H

COMMAND: WUP

Description
WUP moves tape to the end-of-tape, and back to the beginning, to establish tape tension.

Dependencies
None

Example
WUP

TEST NAME
TCOM- Tape Command Exerciser

PARAMETERS
Start=The first module of tape commands to be executed (default=0).
End =The last module of tape commands to be executed (default=0).

NOTE

The default values cause all modules between Start and End to be executed.

DESCRIPTION
The following TCOM modules exercise the tape controller call commands:

Module 1: LOAD and UNLOAD commands
Module 2: MODE and REWIND commands
Module 3: READ, WRITE, and STRK commands
Module 4: SKBF and SKBR commands
Module 5: SKFF, SDFR, and WFM commands
Module 6: SEL, MRTRY, and EGP commands

The DIAG1 command is executed before entering the Start Module.

ERROR MESSAGES
TCOM issues warnings for correctable errors (e.g., no tape in drive, tape write-protected, etc.), and self-explanatory error messages if a tape operation fails. In addition to error messages, the TCU registers are also displayed.

LAP SUMMARY
Each module includes an introductory statement, a running commentary on the test progress, and a message signalling successful completion. There is no lap summary for the entire TCOM test.

NOTES
TCOM halts execution when a tape operation fails.

TEST NAME
TEX - tests magnetic tape cartridges

PARAMETERS
datpat: Entered in hexadecimal. The data pattern written on the tape (default =0x5555).

pr_wr_rr: Entered in hexadecimal. The first two digits, pr, are the number of times the pattern written to tape is read and verified (default=0x10). The second two digits, wr, are the number of retries used in writing the data (default=0). The third two digits, rr, are the number of retries permitted during a read operation (default=0xA).

numblk: Entered in hexadecimal. The number of 0x10 byte blocks which are written, read, and verified as a group.

u_st_et_w: Entered in hexadecimal. The first digit selects which unit will test the tape. The second two digits specify the track where testing begins. The third two digits are the track where testing ends. The fourth digit, if non-zero, causes a tape warmup. A tape warmup moves the tape to the logical end-of-tape, and then rewinds. Default values are 0, 0, 3, and 1, respectively.

DESCRIPTION
This test performs the following:

1. Retries are set two wr.
2. Write numblk blocks of 1000H bytes (starting at track st) with a data pattern of datpat.
3. Retries are set to rr. Reads the blocks just written pr times, and compares to ensure valid data.
4. If EOT is encountered, proceed to step 5; if not, return to step 1.
5. Rewind and move to next track; if at track et+1, stop test; if not, return to step 1.

Counters and error totals are displayed at the end of each read and write. A detailed log of the last 17 errors recorded, is available by pressing START and entering an appropriate response to the menu prompt.

ERROR MESSAGES

The user is warned if the unit selected does not exist, is not loaded with tape, or if the tape is write protected. If a tape operation fails, the TCU registers are displayed. Normal error messages from the test are displayed if a verification yields a compare error. The track, block, and data are displayed.

LAP SUMMARY
At the end of each read and write of `numblk' blocks, a table is displayed containing lap and parameter information, number of retries attempted, and errors classified as read or write, or hard or soft. This information is cumulative.

NOTES
This test is based on a tape distributors, tape screening program.

TEST NAME
NEWMEM1

PARAMETERS
maxseg=maximum segment number to be tested (default=3)
minseg=minimum segment number to be tested (default=0)

DESCRIPTION
NEWMEM1 does a random data test on the memory segments given in the segment. Each memory segment is 64K bytes. The test gets repeated n times, where n=#REPS in the test line.

NEWMEM1 must relocate the test code if segment 0 is tested. If minseg=maxseg=0, the code is relocated to segment 1 and the test runs from segment 1. If minseg=0 and maxseg=0, the code is moved to the next higher segment for each segment tested. The code rotates through the segments to be tested. The code segment is moved each repetition.

The test fills each segment with random data, then reads it back for verification.

ERROR MESSAGES
If the data found in a memory location is not correct, a RANDOM TEST ERROR message appears. It gives the address where the error occurred, what the data should be (DATA=), and what was found in memory (WAS=).

LAP SUMMARY
After each repetition of NEWMEM1, the last 19 errors are displayed in tabular form. The errors are accumulated from one repetition to the next.

NOTES
NEWMEM1 turns the MMU's ON during the test. If NEWMEM1 indicates errors, SADIE diagnostic MMUTST5 or the SYSTEM POWER-UP DIAGNOSTICS (SPUD) should be run to ensure the MMU's integrity.

TEST NAME
NEWMEM2 - a quick memory test of memory segments

PARAMETERS
maxseg=maximum segment to be tested (default=3)
minseg=minimum segment to be tested (default=0)

DESCRIPTION
NEWMEM2 does three (3) write-read-compare tests on each segment tested. The three tests repeat n times, where n=#REPS in the test line. The tests are:

"Test 1:" A simple data line test.
"Test 2:" A simple address line test.
"Test 3:" A block test where the segment is filled with x'5555's; read and verified; filled with 'AAAA's; and read and verified.

NEWMEM2 relocates the code segment the same as NEWMEM1.

ERROR MESSAGES
The tests described above, display these messages if an error occurs:

DATA LINE TEST ERROR (test 1)
ADR LINE TEST ERROR (test 2)
BLOCK DATA FAULT (test 3)

Each message gives the address and data which is written, and the incorrect data which is read back.

LAP SUMMARY
After each repetition of NEWMEM2, the last 19 errors are displayed in tabular form. The errors accumulate from one repetition to the next.

NOTES
NEWMEM2 turns the MMU's ON during the test. If NEWMEM2 indicates errors, SADIE diagnostic MMUTST5 or the SYSTEM POWER-UP DIAGNOSTICS (SPUD) should be run to ensure the MMU's integrity.

TEST NAME
NEWMEM3 - a thorough n-cell-coupling test of memory segments

PARAMETERS
maxseg=maximum segment number to be tested (default=3)
minseg=minimum segment number to be tested (default=0)

DESCRIPTION
NEWMEM3 is a slow, but thorough, n-cell-coupling test. It flags problems the other memory tests may not reveal. The test takes approximately 20 minutes per segment tested. NEWMEM3 repeats n times, where n=#REPs in the test line.

NEWMEM3 relocates the code segment the same as NEWMEM1.

ERROR MESSAGES
NEWMEM3 displays a NEWMEM3 FAULT message for each error. The error message gives the address where the error occurred, the value written, and the incorrect value read back.

LAP SUMMARY
At the conclusion of each test repetition, the last 19 errors are displayed in tablular form. The errors accumulate from one repetition to another.

NOTES
NEWMEM3 turns the MMU's ON during the test. If NEWMEM3 indicates errors, SADIE diagnostic MMUTST5 or the SYSTEM POWER-UP DIAGNOSTICS (SPUD) should be run to ensure the MMU's integrity.

TEST NAME
MMUTST5 - a series of tests of the MMU's

PARAMETERS
None

DESCRIPTION
MMUTST5 performs a series of tests on the MMU's. For each test, MMUTST5 displays a message informing the user that it is about to begin the test. The tests are performed on each of the three MMU's (CODE, DATA and STACK) unless otherwise specified.

1. A block random data test, where 256 bytes of random data are written to memory, read back, and compared.

2. A SDR test, where random data is written to each SDR, then read and verified.

3. A CONTROL register test, where random data is written to each control register, then read and verified.

4. A test of the read-only flag, in the DATA and STACK MMU SDR's.

5. A test of the LIMIT register, in the DATA and STACK MMU SDR's.

6. A test of the DIRW flag, in the DATA and STACK MMU SDR's.

7. A test of address translation using each SDR.

ERROR MESSAGES
MMUTST5 displays messages whenever an access violation causes a segment trap. The messages indicate where the MMU error occurred.

LAP SUMMARY
On completion of each repetition of the series of tests, MMUTST5 displays the lap number, error count, and for each MMU:

1. The number of block data errors

2. The number of random SDR errors

3. The number of control register errors (SNR, DSC and MODE registers)

4. The number of access violations for the following types:

   a. read only

   b. limit

   c. direction

   d. translation

For information about the MMU's, refer to:

Z8010
Z80000 Z-MMU Memory Management Unit Product Specification, March, 1981
(Product Number 00-2046-A)

NOTES
None

TEST NAME
CENT.PRT (CENTRONICS printer interface test)
DP.PRT (DATA PRODUCTS printer interface test)

PARAMETERS
None

DESCRIPTION
CENT.PRT and DP.PRT are interactive tests of the Centronics and Data Products printer interface, respectively. The printer port tests, prompt the user to verify that the printer is online. If the printer is online, the tests send the printable character set to the PIO Channel B, n times, where n is the number of repetitions in the test line.

During these tests, PIO interrupts are disabled; the PIO is polled by the test.

ERROR MESSAGES
A message is displayed if the proper connection does not exist between the System 8000 and the printer, or if the printer port is busy too long.

Other error messges are self explanatory.

LAP SUMMARY
None

NOTES
None

TEST NAME
S16SIO

PARAMETERS
None

DESCRIPTION
S16SIO is an interactive, menu-driven test of the SIO's and CTC's not used by the console. (SADIE uses SIO #0, Channel B, to communicate with the console. The test assumes that SIO #0, Channel B, is functioning.)

A choice on the test menu is to exit the test. This is the only way the test should be exited; DO NOT PRESS `START' TO EXIT.

This test requires an auxiliary terminal, referred to by the test as AUX. The user selects the SIO to be tested, and the test prompts the user to plug AUX into a specified port on the system rear panel. Entering <CR> on the console signals the test to proceed.

The terminal must not be connected to TTY0 until SADIE transfers control to S16SIO.

The test proceeds by displaying the entire set of ASCII printable characters continuously, until any key is pressed on AUX. Then the test is in 'echo mode', and any character pressed on AUX is echoed back to AUX by the test. Pressing any key on the CONSOLE terminates `echo mode', and returns to the S16SIO menu.

ERROR MESSAGES
None

LAP SUMMARY
Not applicable.

NOTES
S16SIO turns OFF interrupts from the console, when the user signals for the test to proceed. Therefore, DO NOT PRESS `START' TO EXIT this test, as interrupts will continue to be disabled when the PAUSE menu is displayed.

If Sl6SIO does not respond after the user signals for the test to proceed, check that AUX is connected to the correct port. To recover, hit several keys in succession on the console.

The test of SIO #0, Channel A, is slightly different from the other SIO tests. The first key pressed, or AUX, returns control to the menu. There is no 'echo mode' test on SIO #0, Channel A.

TEST NAME

SIOMODEM - a test of the SIOs, including modem signals

PARAMETERS

SIOMODEM is an interactive test; it receives no parameters from SADIE.

DESCRIPTION

SIOMODEM tests the following SIO functions:

- o Character transmission, polled mode.
- o Character transmission, interrupt mode ("status affects vectors" false).
- o Character transmission, interrupt mode ("status affects vectors" true).
- o SIO modem signals (RTS,DCD,DTR,CTS), polled mode.
- o SIO modem signals, interrupt mode.
- o Transmit interrupts.
- o External status interrupts.
- o Character transmissions, mismatched baud rates.
- o Character transmission at all standard baud rates.

The tests are arranged in three different test sequences, each of which is performed for each pair of tty ports. The tests are:

1. Polled mode character/modem test. Performs a test of character transmission and SIO modem signals.
2. Interrupt mode character test. Performs a test of character transmission in interrupt mode where "status affects vectors" false; tests character transmission of mismatched baud rates; and tests all standard baud rates.
3. Modem/character interrupt test. Performs a test of character transmission in interrupt mode where "status affects vectors "true; tests SIO modem signals; transmit interrupts; and external status interrupts.

Each test continues indefinitly, until the user terminates it by pressing a key on the console. (Do not "type ahead" during the test, or unexpected/unintended choices may accidentally be made.)

All errors result in an audible "beep", and a message on the console screen explaining the error. Beeps also occur when the program expects input from the console.

All SIO ports, including the console port, remote line port, and CPU expansion ports, can be tested. SIOMODEM (also known as SIO Test #3) performs all tests that are in the earlier version (SIO Test #2, or S16SIO). Therefore, it is not necessary to run S16SIO if executing SIOMODEM.

SIOMODEM requires a special interconnecting cable to carry the modem signals between SIO ports (for example, port TTY2 might be looped back to TTY6). The test is interactive, and gives instructions to the user concerning how the SIOs are to be interconnected. It also explains what input is expected and at what time. The interconnecting cable consists of two standard RS232, 25-pin male connectors, wired as follows:

	A		B	
xmitted data	2	-----	3	received data
rcvd data	3	-----	2	transmitted data
RTS	4	-----	6	DCD
CTS	5	-----	20	DTR
DCD	6	-----	4	RTS
signal ground	7	-----	7	signal ground
DTR	20	-----	5	CTS

The other signals should not be wired. The length of the interconnecting cable should be about 8 feet. If the cable is not available, modem tests cannot be performed, and diagnostic test S16SIO should be run.

SIOMODEM offers the capability of moving the console port, so the standard console channel (TTY1, labelled "CONSOLE") can be tested the same as all other SIO channels. The console may be moved to any SIO channel which has already been tested. The remote line (TTY0) can also be tested when the console is moved. Testing of any pair of SIO ports may be skipped if desired. The test continues in a circular fashion, testing pairs of SIO channels, until the user responds to a prompt with "Q", to end all tests.

Console interrupts are disabled during the test, and hence NMI is not to be utilized to terminate the test. The NMI function does not normally operate while the test is executing.

ERROR MESSAGES

All error messages are preceded by an audible "beep" on the console.

ERROR: NO CHARACTER RECEIVED, POLLED MODE, TTY# FAILED ON CHARACTER XX

This is the first test performed. No character was transmitted between the ports in a polled-mode transmission. All possible ASCII characters from 00 to FF are transmitted, in both directions, and the error text reveals which character was attempting to be transmitted. If the failed character was 00, check the cable. This message occurs if one of the connectors is loose, a wire is broken, a pin has come loose, or if the cable is connected to the wrong port.

POLLED TTY# ###CHARS, ## ERRS, POLLED TTY# ###CHARS, ## ERRS

This message is displayed on the completion of each cycle of the polled mode test (after 256 characters have been sucessfully transmitted in each direction). The numbers represent cumulative totals since the test for the current pair of SIO ports was initiated. If the error count is zero in both directions, the test is successful. The error count represents the number of times that the character received was different from the character transmitted. The interrupt mode test continues until a character is input on the console.

TTY# RECEIVE MODEM ERROR STATUS1 = ##, DCD AND/OR CTS SET

An error occurred in polled-mode testing of the transmit modem signals. RTS and DTR were reset in the receive modem, but DCD and/or CTS failed to be reset in the transmit modem. Check the interconnecting cable for broken wires or loose pins.

TTY# TRANSMIT MODEM ERROR STATUS1 = ##, DCD AND/OR CTS SET

An error has occurred in polled-mode testing of the transmit modem signals. RTS and DTR were reset in the receive modem, but DCD and/or CTS failed to be reset in the transmit modem. Check the interconnecting cable.

TTY# RECEIVE MODEM ERROR STATUS2 = ##, DCD NOT TURNED BACK ON

Error in polled-mode testing of medem signals. RTS and DTR were set high in the transmit modem, but the expected status in the receive modem, CTS low and DCD high, did not occur.

TTY# TRANSMIT MODEM ERROR STATUS2 = ##, CTS NOT TURNED BACK ON

Error in polled-mode testing of modem signals. DTR was set in the receive modem, but the expected status in the transmit modem, CTS high and DCD low, did not occur.

SPEED= #### BAUD, TEST CYCLE ##

Displayed at the beginning of each interaction of the interrupt mode test. The interrupt mode test is similar to the polled mode test, except that SIO character received interrupts are used to signal the receipt of each character. All characters from 00 to FF are transmitted in each direction for each of the standard line speeds: 19,200, 9600, 4800, 1200, 300, and 110 baud. The interrupt mode test continues until a character is input on the console. When a character is input, the test terminates on completion of the current cycle. The LAST CYCLE display appears after the test cycle number, indicating that no new cycle will be started.

ERROR: INTERRUPT NOT RECEIVED, TTY#, FAILED ON CHARACTER ##

Character received interrupt, failed to occur in interrupt mode testing ("status affects vectors" = false, interrupt vector =0x20). The TTY number of the receive port is given.

INTERRUPT TEST ON TTY#, ###CHARS, ##ERRS, ON TTY#, ##CHARS, ##ERRS

This message is displayed at the end of each iteration of each cycle of the interrupt mode test, and is analagous to the similar display of the polled mode test. Character and error counts are cumulative for all test cycles. Zero error counts in both directions indicate a successful test. The error count is the number of times that the character transmitted, failed to match the character received.

ERROR: CHARACTERS MATCH WITH MIS-MATCHED BAUD RATES!!

Indicates that 256 characters were successfully transmitted, despite differing baud rates in the transmit and receive ports. The baud rate clocks are not currectly set.

ERROR: DCD INTERRUPT NOT RECEIVED, TTY#, STATUS= ##

An error occurred in interrupt-mode testing of modem signals. RTS and DTR were reset in the transmit modem, which should have generated an interrupt in the receive modem when DCD and CTS are reset. If both polled mode and interrupt mode modem tests fail, the interconnecting cable should be carefully checked; however, if only one modem test fails, the SIO is probably at fault.

ERROR: DCD AND/OR CTS NOT CLEARED AFTER RTS/DTR CLEARED, TTY#, STATUS= ##

The interrupt described in the previous error occurred in the receive SIO modem; however, the expected status of zero for both DCD and CTS did not occur.

ERROR: DCD INTERRUPT NOT RECEIVED, TTY#, STATUS= ##

RTS was set on the transmit modem, which should have caused an interrupt on the receive modem when DCD is set.

ERROR: TTY# DCD NOT SET AND CTS CLEARED AFTER RTS SET, STATUS = ##

The interrupt described in the previous error occurred, however the expected status, DCD set and CTS reset, did not occur.

ERROR: CTS INTERRUPT NOT RECEIVED, TTY#, STATUS= ##

DTR was set on the receive port, which should have caused an interrupt on the transmit port when CTS was set true.

ERROR: TTY# CTS NOT SET AND DCD RESET AFTER DTR SET, STATUS= XX

The interrupt described in the previous error occurred, but the expected status, CTS true and DCD false, did not occur.

ERROR: NO CHAR RECEIVED, MODEM INTERRUPT TEST ON TTY#, RECEIVE STATUS= ## FAILED ON CHARACTER= ##, TRANSMIT STATUS= ##

A character-received interrupt, failed to occur on the receive SIO during interrupt-mode modem testing. the interrupt-mode character test is repeated, except that "status affects vectors" is true, and interrupt vectors 0x24 and 0x2C are used for character interrupts, instead of 0x20. The status of the transmit and receive modems, and the character attempting to be transmitted, are given.

ERROR: MODEM LINES HAVE DROPPED BETWEEN TTY# AND TTY#, RECEIVE STATUS= ##, TRANSMIT STATUS= ##

An external/status interrupt, indicating a change in one of the states of the modem signals, occurred during the transmission of characters. The status of the receive and transmit modems is given. It may not be possible to recover from this error, causing the test to be restarted.

ERROR: NO TRANSMIT INTERRUPTS OCCURRED, MODEM INTERRUPT TEST, TTY#, STATUS= ##

No transmit buffer empty interrupts, have occurred on the transmit modem during the transmission of 256 characters.

MODEM INTERRUPTS ON TTY#, ##CHRS, ##ERRS, ON TTY#, ##CHRS, ##ERRS

Displayed on completion of each iteration of the modem interrupt test (after 256 characters are successfully transmitted in each direction). Analagous to the messages during the polled mode, and interrupt mode character testing. The character and error counts are cumulative from the beginning of the test. The error count is the number of times the character transmitted did not match the character received. The test is successful when the error count is zero in both directions.

ERROR: SPECIAL RECEIVE CONDITION INTERRUPT

No test currently implemented, should cause this interrupt to occur. The receive SIO modem believes it is detecting a serious error condition, such as a parity or framing error.

LAP SUMMARY

This is an interactive test and does not use a lap count or summary. The test continues until the user responds to a prompt with "Q", too quit (or terminate) all tests.

NOTES

A special SIO interconnecting cable, described in the Description of this test, is required.

If this test is successfully executed, it is not necessary to run the SI6SIO test.

Console interrupts are disabled during this test. START (NMI) is not to be used to terminate the test, and should not be pressed. The test will terminate upon responding to a prompt with a "Q".

Errors which occur during polled-mode character, or modem signal testing, are usually caused by broken wires or loose pins in the cable, or by a connector which is either loose or connected to the wrong I/O port.

The most all-inclusive test is the Interrrupt-Mode Modem Control test. This test is recommended for extended testing, as it tests for the greatest number of possible errors.

Systems Publications

Reader's Comments

Your feedback about this document helps us ascertain your needs and fulfill them in the future. Please take the time to fill out this questionaire and return it to us: this information will be helpful to us and, in time, to future users of Zilog products.

Your Name: _____

Company Name: _____

Address: _____

Title of this document: _____

Briefly describe application: _____

_____

_____

Does this publication meet your needs? ☐ Yes ☐ No If not, why not? _____

_____

_____

How are you using this publication?

☐ As an introduction to the subject?

☐ As a reference manual?

☐ As an instructor or student?

How do you find the material?

	Excellent	Good	Poor
Technicality	☐	☐	☐
Organization	☐	☐	☐
Completeness	☐	☐	☐

What would have improved the material? _____

_____

_____

Other comments and suggestions: _____

_____

_____

If you found any mistakes in this document, please let us know what and where they are: _____

_____

_____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 35 CAMPBELL, CA.

POSTAGE WILL BE PAID BY ADDRESSEE

Zilog

Systems Publications
1315 Dell Avenue
Campbell, California 95008
Attn: Publications Manager

Zilog, Inc. 10460 Bubb Road, Cupertino, California 95014 Telephone (408)446-4666 TWX 910-338-7621

Printed in USA