Zilog

Zilog

03-3199-01

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 USER MANUAL

PRELIMINARY VERSION

PREFACE

This manual provides an introduction and user information for the ZEUS™ Operating System used with the Zilog System 8000™ . Detailed description is given for system features, including the programming environment, the Monitor Program, and Monitor I/O procedures.

This manual is organized by sections, each section representing a major component that will familiarize the user with the system.

SECTION	
1	General Description -- Describes the System 8000, including system features and characteristics.
2	Programming Environment -- Provides hardware and software overviews of the system.
3	System 8000 Monitor Program -- Introduces the Monitor Program and explains the basic debugging commands, I/O controls, and upload/download software.
4	Monitor Program I/O Procedures -- Introduces the I/O procedures used with the Monitor Program.

APPENDIX	
A	Glossary -- Lists the most important terms and acronyms introduced in this manual.

For a better understanding of the system hardware components and operating system, the user is encouraged to read the following manuals:

Title	Part Number
ZEUS Reference Manual	03-3195
ZEUS Utilities Manual	03-3196
ZEUS System Administrator Manual	03-3197
System 8000 Hardware Reference Manual	03-3198

System 8000™ and ZEUS™ are registered trademarks of Zilog, Inc.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION 1

GENERAL DESCRIPTION

1.1 Introduction

The Zilog S8000 System (Figure 1-1) is a high performance microcomputer system based on the Z8001A 16-bit microprocessor. Program development and text processing are accomplished with the ZEUS Operating System. Supporting up to 16 users, the system develops code for all Zilog CPUs. This section is a general description of the S8000.

1.2 System Environment

The S8000 uses Winchester disk storage and supports a communication interface with other ZEUS-based systems, emulation devices, and development modules. The S8000 provides comprehensive software development and documentation tools to maximize programmer productivity and documentation quality. It includes the following features:

- A 6 MHz Z8001A 16-bit microprocessor
- 256K bytes of error-correcting memory
- A 32-bit ZBI(TM) backplane with an 8-megabyte/second bandwidth
- Intelligent Z80B-based controllers for disk and tape drives
- A 24-megabyte (unformatted) eight-inch Winchester disk drive
- A 17-megabyte (unformatted) cartridge tape drive
- ZEUS multi-user, multitasking operating system

The following hardware options are also available:

- Additional 256K-byte memory boards for up to 1.5 megabytes of error-correcting memory
- Up to four 24-megabyte Winchester drives
- Up to eight additional serial I/O ports

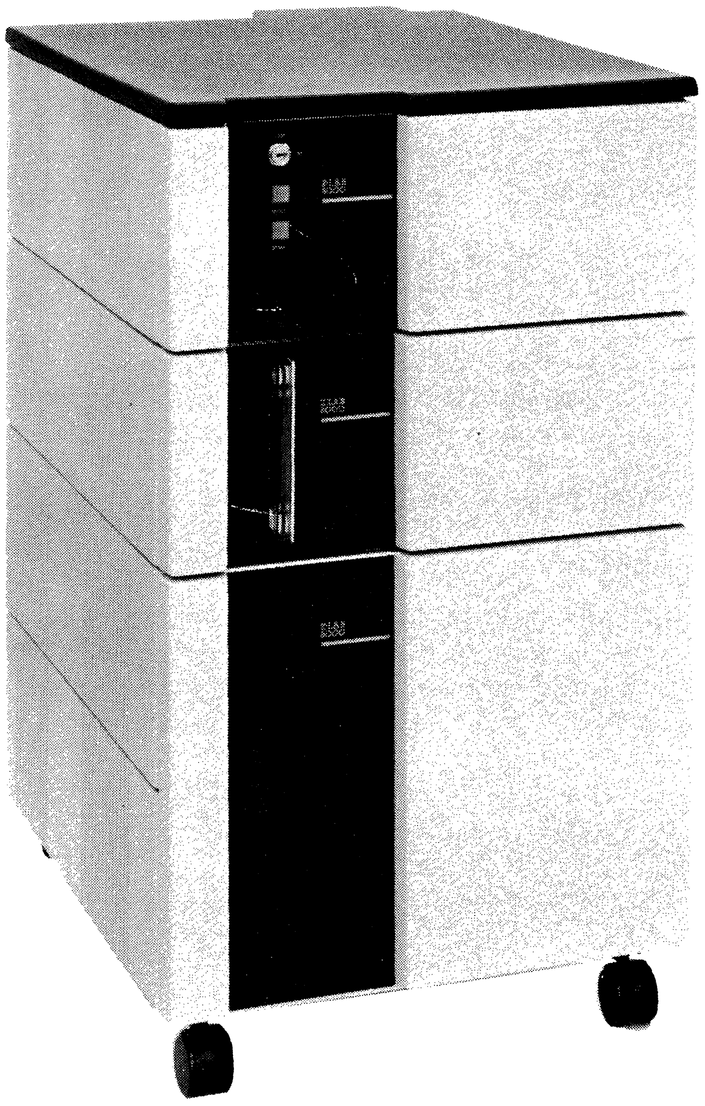

Figure 1-1. S8000 Basic System Configuration

- Up to four 17-megabyte cartridge tape drives
- Character and line printers

The number of controls and indicators have been minimized to facilitate system use. Only the keylock ON/OFF switch, the RESET and START switches, and the AC power switch are necessary to power up and maintain the S8000. Refer to Figure 1-2 for control and indicator locations. Controls for the optional Lear Siegler ADM-31 Data Display Terminal include a brightness/contrast control knob and an AC power ON/OFF switch.

The resources of the S8000 are controlled by the ZEUS Kernel. The Kernel or the operating system provide process management, file management, input/output (I/O) processing, and increased program functionality with compatible file, device, and interprocess I/O.

ZEUS is a multi-user, multitasking operating system consisting of a hierarchical file system for efficient file organization and a comprehensive command language. A communication program allows the S8000 to interface with other ZEUS or UNIX-based systems. Also, with ZEUS, it is possible to communicate with emulation devices and development modules.

ZEUS development tools include extensive language capabilities such as C, Pascal, PLZ/SYS, PLZ/ASM, a compiler-writing system, and a general purpose macroprocessor. Additional enhancements to the development system include a full CRT-oriented text editor, text processing, spelling error detection, and document formatters for the optional printers.

1.3 System Characteristics

Processor:	Segmented 48-pin Z8001A CPU
CPU Clock Frequency:	5.5 MHz
I/O:	Eight RS-232C serial I/O ports and one parallel printer port
Baud Rate:	From 110 to 19,200 baud (set by software)

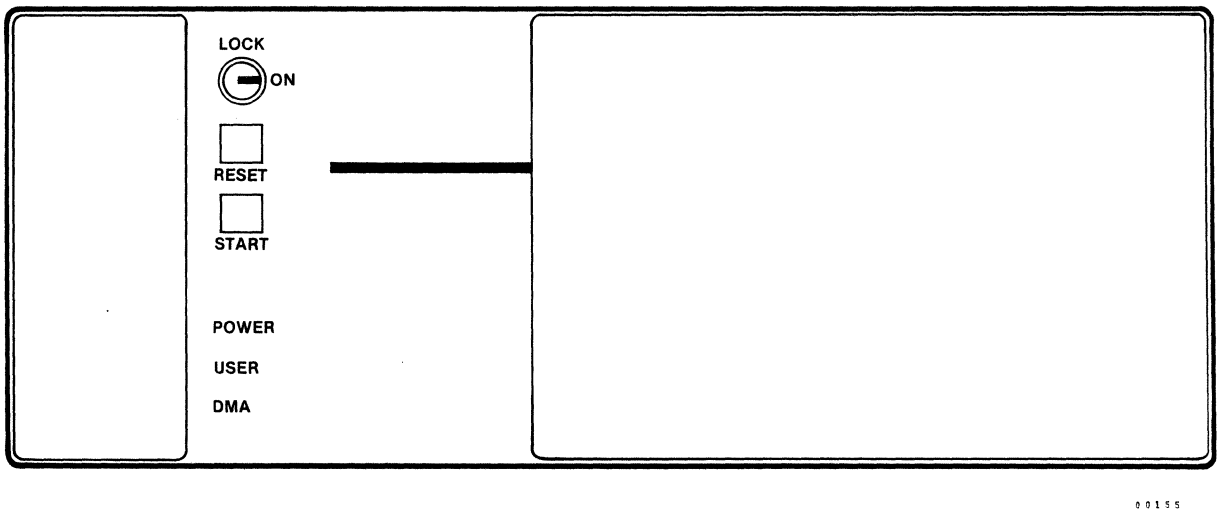

Figure 1-2. Processor Module Controls and Indicators

Front Panel:	Cutouts for keylock ON/OFF switch, RESET switch, and START switch. Translucent plastic for three indicator lamps: POWER (+5V DC), USER (CPU is in normal state), and DMA (CPU is giving up the bus for Direct Memory Access devices)
Rear Panel:	Eight RS-232C serial I/O ports, a parallel I/O port for a printer, a 50-pin connector for the DEI cartridge tape unit interface, a 40-pin connector for the Winchester disk drive interface, and two spare 37-pin connectors for the terminal expansion option
Domestic Power:	117Vac +10% -20%, single phase, 60 Hz. Current: 10A max. (sustained), 15A max. (surge)
International Power:	220Vac +10 -20%, single phase, 50 Hz. Current: 5A max. (sustained), 8A max. (surge)
Environmental:	Operating temperature: 50 degrees F (10 C) minimum 104 degrees F (40 C) maximum Relative humidity: 80% noncondensing
Cabinet Size:	Height: 33 inches (84 cm) Width: 19 inches (48 cm) Depth: 24 inches (61 cm)
Cabinet Weight:	Approximately 132 pounds (60 kg)

1.4 Winchester Disk Performance

Rotation speed:	3600 RPM
Power On to ready time:	15 seconds
Average random positioning time:	42 MS
Number of surfaces:	3
Tracks per surface:	600
Sectors per track:	24

Bytes per sector:	512
Data transfer rate:	801K bytes/second

1.5 Cartridge Tape Drive Performance

Cartridge:	ANSI X3.55 - 1977 300 ft. or 450 ft. Tape length
Speed Read/Write (rewind):	30 ips (90 ips)
Tracks:	4
Recording density:	6400 BPI

1.6 ADM-31 Data Display Terminal Performance

DISPLAY

Refresh Rate:	60 Hz or 50 Hz, depending on line frequency
Character Set:	128 ASCII characters (uppercase, lowercase, and control characters)

KEYBOARD FUNCTIONS

Keyboard:	26-letter alphabet with uppercase and lowercase, numeric 0 through 9
Cursor Control:	Individual cursor control keys
Edit Keys:	Character insert, character delete, line insert, line delete, line erase, page erase, and clear
Function Command Keys:	ESCape, BREAK, PRINT, SEND LINE, SEND PAGE, TAB/BACK TAB, NEW LINE, and FUNCTION
Special Purpose Keys:	RETURN, CTRL (control), and RUB

TRANSMISSION MODES

Interface:	RS-232C point-to-point or 20mA current loop; RS-232C EXTENSION port
Data Rate:	Variable

Parity:	None
POWER	
Standard:	115Vac, 60 Hz
Optional:	230/240Vac, 50 Hz
Heat Dissipation:	222 BTU/HR
Environmental:	Operating temperature: 41 to 122 degrees F (5 to 50 C) Relative humidity: 5% to 95% without condensation

SECTION 2

PROGRAMMING ENVIRONMENT

2.1 Introduction

The S8000 System uses a Z8000 microprocessor-based operating system to perform software development tasks. This section provides the basis for all later discussions of Monitor Program applications and I/O procedures.

The S8000 Monitor sets software breakpoints for program debugging, and includes I/O control, interface software for use with a serial interface to a remote computer system, and the primary bootstrapper used to bring the system up.

2.2 Hardware Configuration

The following paragraphs briefly describe the major characteristics of the S8000 hardware. Detailed general installation and maintenance information is contained in the S8000 Hardware Reference Manual. Figure 2-1 illustrates the functional relationship of the S8000 hardware components.

2.2.1 Microprocessor

The architectural resources of the Z8000 CPU include sixteen 16-bit general-purpose registers, seven data types ranging from 8-bit to 32-bit long words and byte strings, eight user-selectable addressing modes, and 110 distinct instruction types. The CPU can address up to 16 megabytes in 128K byte segments (64K bytes of data and 64K bytes of instruction). Moreover, more than 90% of the instructions can use any of five main addressing modes, with 8-bit byte, 16-bit word, and 32-bit long word data types.

The CPU has two operating modes, system and normal (user), that keep operating system and applications programming separate, as in computer systems. This separation of CPU resources promotes the integrity of the system, since programs operating in normal mode cannot access those aspects of the CPU that deal with time-dependent or system interface events.

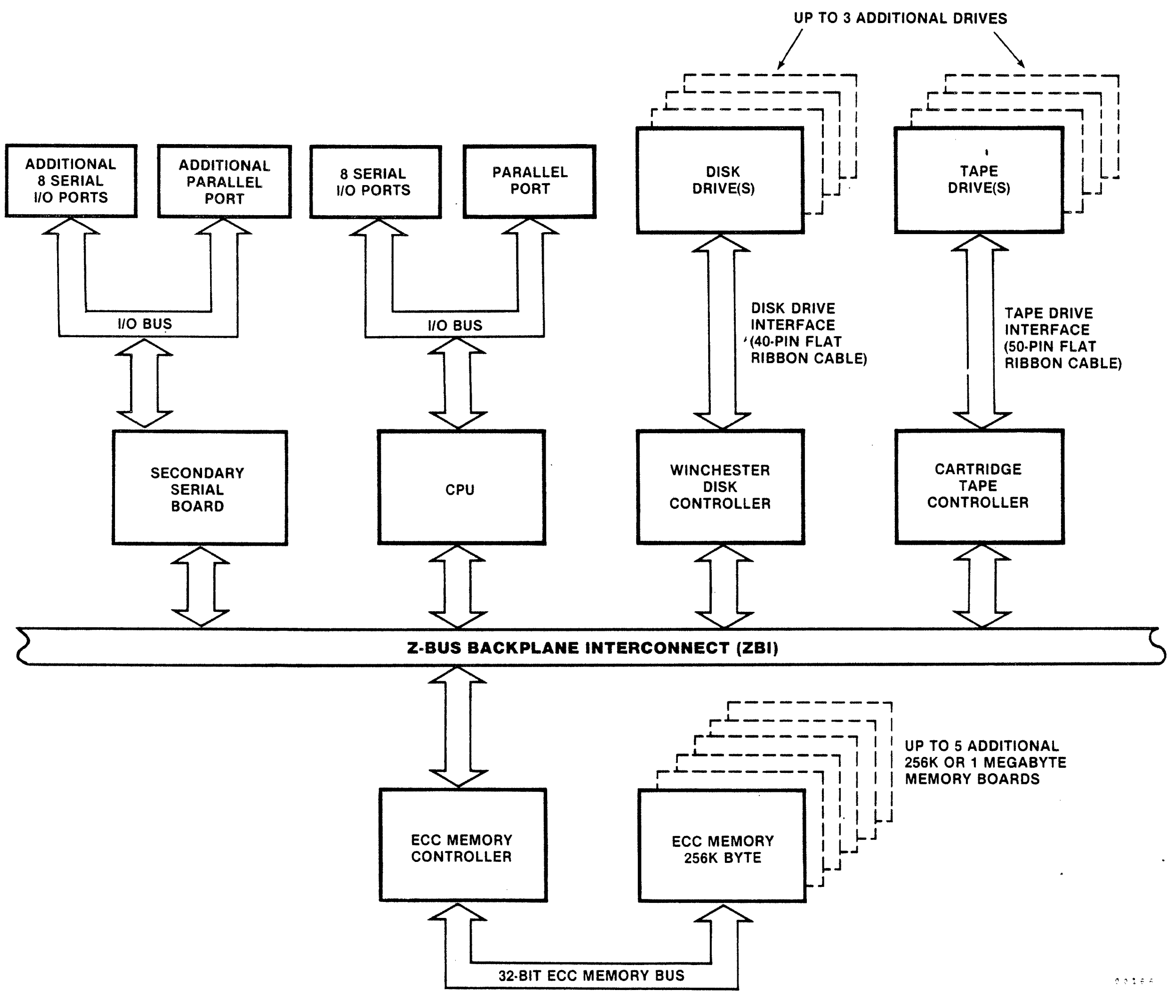
UP TO 3 ADDITIONAL DRIVES
ADDITIONAL 8 SERIAL I/O PORTS
ADDITIONAL PARALLEL PORT
8 SERIAL I/O PORTS
PARALLEL PORT
DISK DRIVE(S)
TAPE DRIVE(S)
I/O BUS
I/O BUS
DISK DRIVE INTERFACE (40-PIN FLAT RIBBON CABLE)
TAPE DRIVE INTERFACE (50-PIN FLAT RIBBON CABLE)
SECONDARY SERIAL BOARD
CPU
WINCHESTER DISK CONTROLLER
CARTRIDGE TAPE CONTROLLER
Z-BUS BACKPLANE INTERCONNECT (ZBI)
UP TO 5 ADDITIONAL 256K OR 1 MEGABYTE MEMORY BOARDS
ECC MEMORY CONTROLLER
ECC MEMORY 256K BYTE
32-BIT ECC MEMORY BUS

Figure 2-1. S8000 Functional Block Diagram

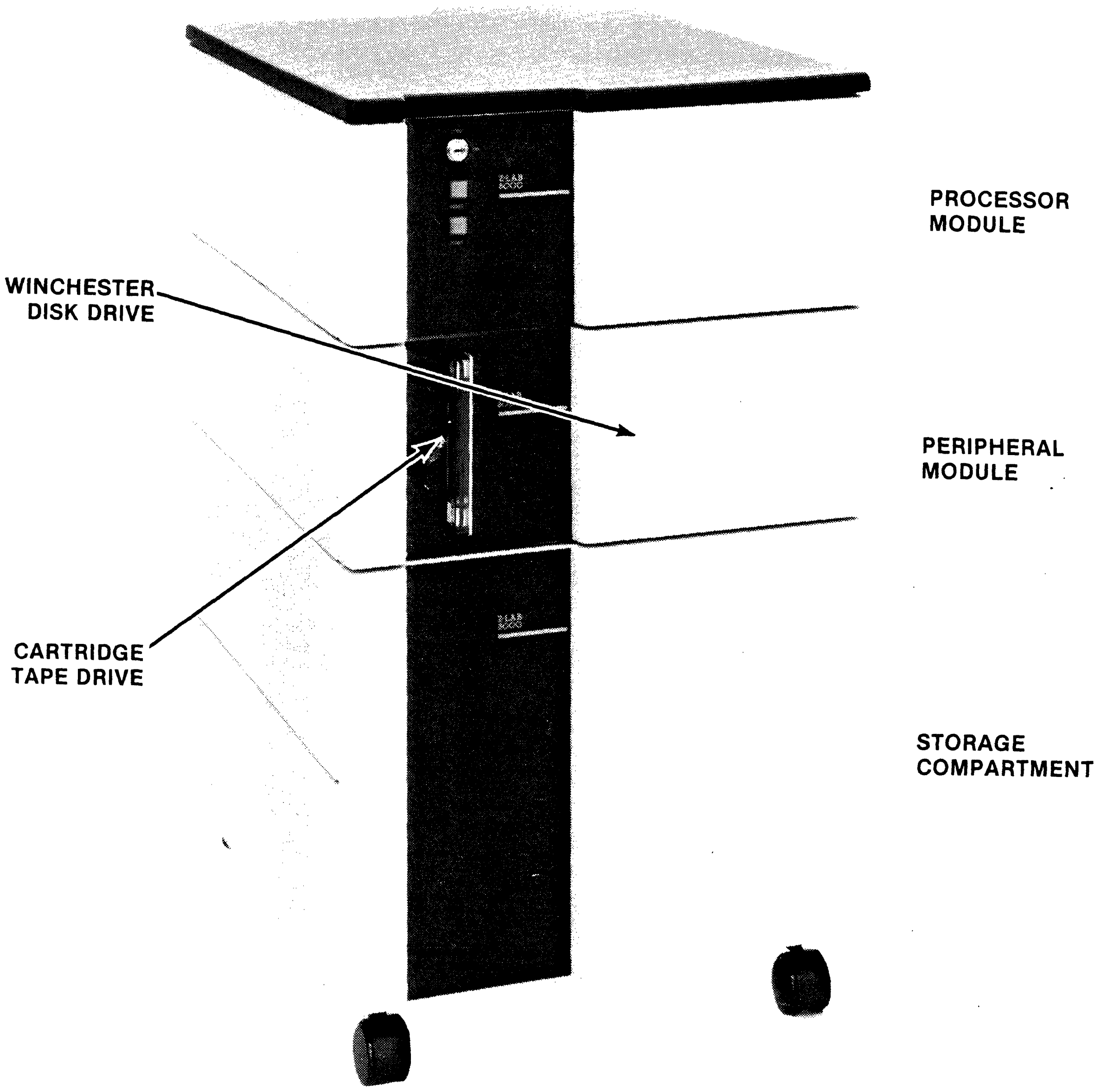

Figure 2-2. Peripheral Module Hardware Components

2.2.2 Winchester Disk Drive

The hard disk subsystem consists of a 24-megabyte eight-inch Winchester disk drive that interfaces with an intelligent Z80B-based disk controller. A formatted disk is capable of retaining up to 22 megabytes of user process data.

The Winchester disk drive, which is housed in the peripheral module (Figure 2-2), provides rapid access to the ZEUS file system which is used for program development.

2.2.3 Cartridge Tape Drive

The cartridge tape drive can be used for loading the ZEUS Operating System, selective file storage, high speed program and data file back-up to the Winchester disk drive, and for executing standalone system diagnostics. The control of the tape drive is provided by a Z80B-based cartridge tape controller, located in the processor module. Up to 17.2 megabytes of unformatted data or 14 megabytes of formatted data can be stored on a 450-foot tape.

2.2.4 ADM-31 Data Display Terminal

The optional data display terminal is the primary bidirectional data interface between the user and the system. The display screen is a 12-inch diagonal CRT with a graphics matrix of 80 characters per line by 24 lines. All 128 printable ASCII characters can be displayed on the screen.

The terminal keyboard (Figure 2-3) is similar to that of a standard typewriter with the addition of special-purpose keys (Figure 2-4) to facilitate command execution. The following paragraphs describe the function of the special-purpose keys:

<u>Return Key</u>. Commands are read by the ZEUS Operating System character-by-character as they are entered. The command is executed only after the return key, labeled RETURN, is pressed.

<u>Control Key</u>. The control key, labeled CTRL, generates and sends control instructions to the terminal during command execution. Any character that is typed with the CTRL key pressed is transparent to the user, but is recognized by the CPU. For example, to slow down or temporarily interrupt data transmission to the terminal without permanently halting execution of a command, enter <u>control-s</u> by typing <u>s</u>

while holding down CTRL. This freezes the screen. To restart transmission, enter control-q. This sequence is used to display data a few lines at a time.

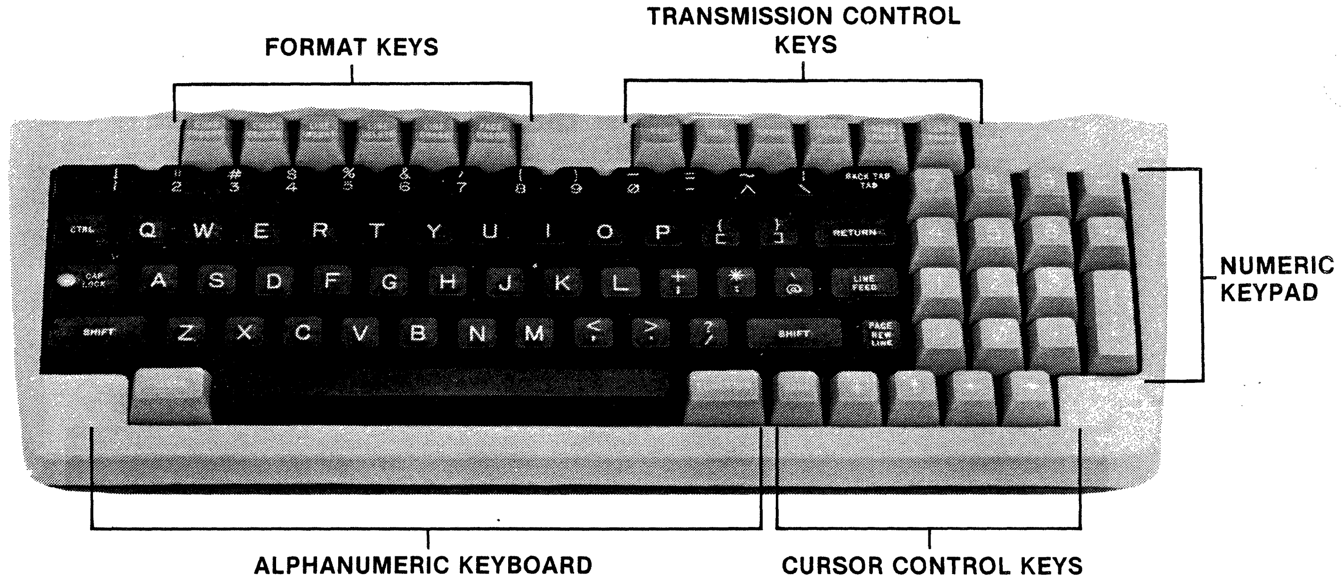

Figure 2-3. ADM-31 Data Display Terminal Keyboard

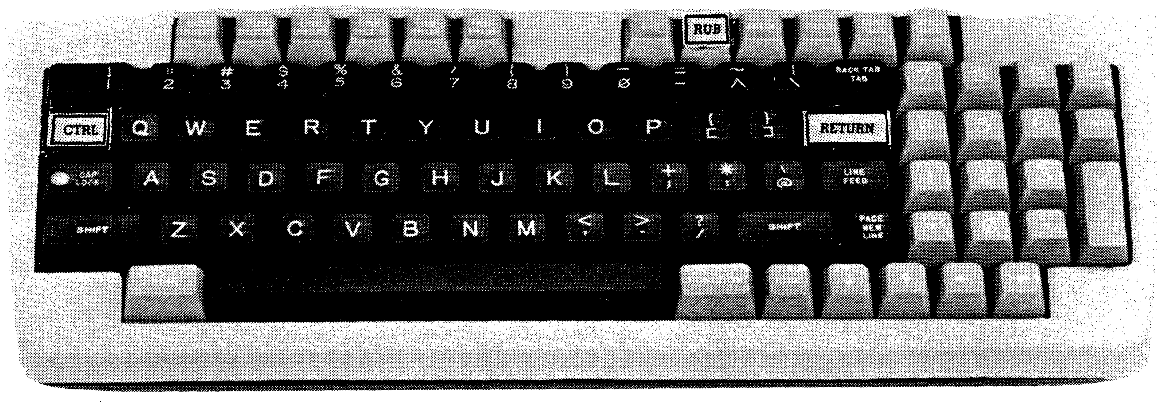

Figure 2-4. ADM-31 Keyboard Special-Purpose Keys

<u>RUB</u> <u>Key</u>. This special-purpose key, labeled RUB, stops the execution of a command before it reaches completion. After successfully stopping the command execution, the ZEUS Operating System responds with a new prompt (%).

2.2.5 Communication Ports

The S8000 communicates with peripheral devices that are compatible with an RS-232C interface. The physical interface to the eight serial I/O ports, the parallel printer port, and the two terminal expansion ports is with the connectors located on the rear of the processor module. Figure 2-5 shows the location of the communication ports.

The Winchester disk drive and the cartridge tape drive communicate with their respective controllers by using connectors located on the rear of the processor and peripheral modules.

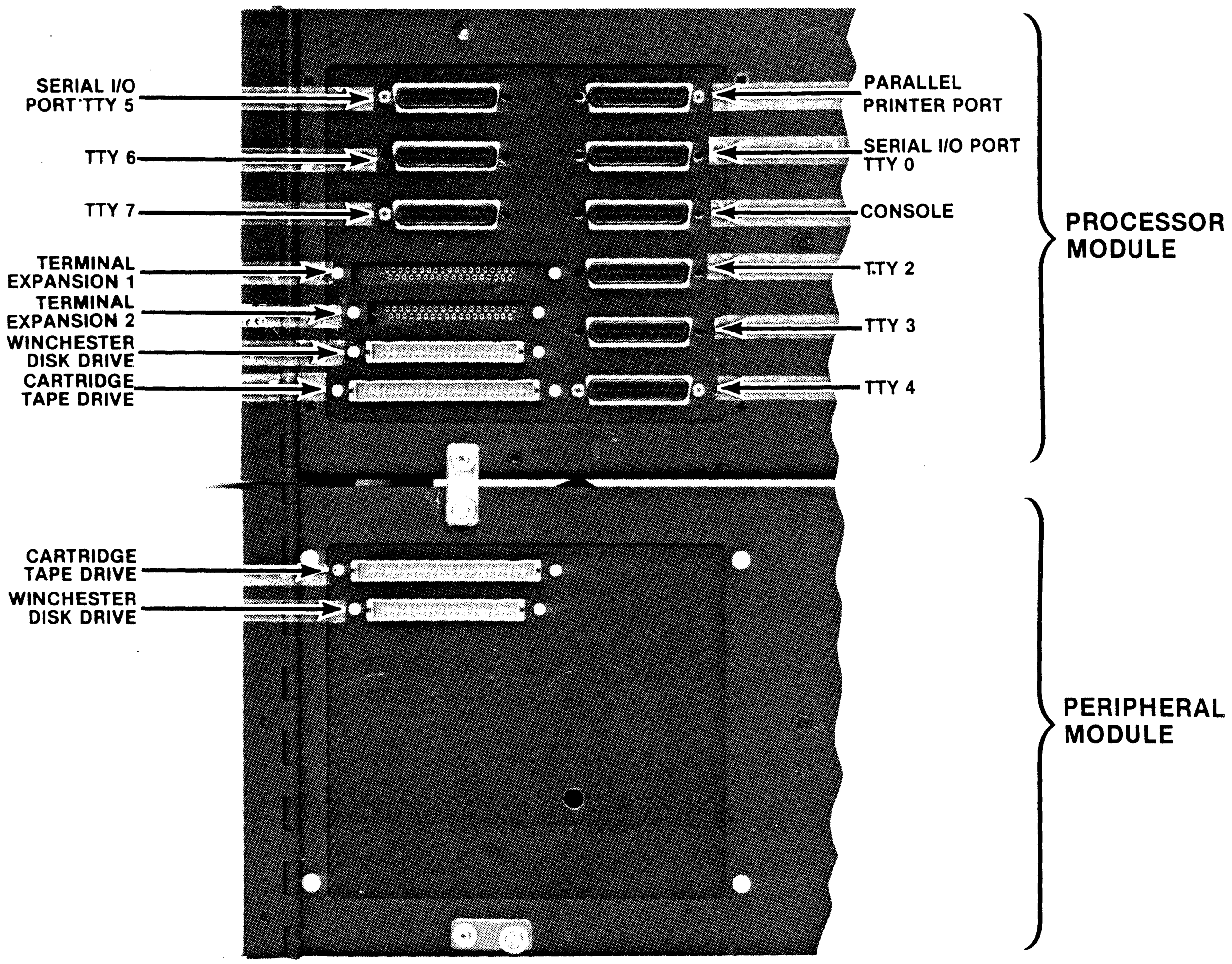

Figure 2-5. S8000 Communication Ports

USER

18

SECTION 3

S8000 MONITOR PROGRAM

3.1 Introduction

The S8000 Monitor Program includes basic debugging commands, I/O control, and interface software for use with a serial interface to a remote computer system. Detailed interfacing procedures are found in the S8000 Hardware Reference Manual (03-3198).

3.2 Monitor Program Debug Environment

The Monitor Program sets software breakpoints for program debugging. A breakpoint is a command that interrupts or stops program execution at a specified address in the program. The address specified in the breakpoint is the address of the instruction. When encountered during program execution, the breakpoint suspends execution of the user's program and saves all registers, program counters (PC), and the flag control word (FCW) in the memory area provided. It then displays a message reporting the break and the address where it occurred.

Any number of breakpoints can be set manually by setting the desired breakpoint address to %7F00 (% indicates the address is in hex notation). This interrupts the executing program and jumps (traps) to the breakpoint procedure. The breakpoint must be located at an even address. When the breakpoint is no longer required, the original instruction must be manually restored.

The BREAK command saves the address where the breakpoint is being set and the instruction that it is replacing. When the breakpoint is cleared, the instruction is automatically restored. The BREAK command also stores a repetition counter, n. Execution is not suspended until the nth time this breakpoint is encountered unless another breakpoint is encountered first.

The following restrictions on the user program are necessary to set breakpoints:

1. The program must be able to execute with interrupts enabled after encountering the breakpoint.

2. The program should not be timing-dependent because there will be some timing distortion each time the breakpoint is encountered.

3. The user program must not use Channel 3 of the Z80A Counter Timer Circuit (CTC), because it is used to implement the multiple execution feature.

4. The breakpoint cannot be within an interrupt procedure entered by an interrupt from Channels 0 through 2 of the Z80A CTC.

The BREAK and the NEXT commands use instruction modification and the interrupt system. Therefore, the program being debugged cannot be in the PROM area and cannot involve modifications of the interrupt status.

Any set breakpoints must be cleared before a new program is loaded from the S8000; otherwise, previously set breakpoints continue to operate on the new program during debugging.

The user stack is used whenever a JUMP or GO command is executed. The command must be set to some address within writable memory. If the JUMP or GO address has a system breakpoint set, the execution of the instruction immediately following the JUMP or GO does not cause suspension of execution. Subsequent executions suspend the breakpoint to permit breaking and continuing execution without resetting the breakpoint.

3.3 Monitor Program

The following conventions are used in command descriptions:

< >	Angle brackets enclose descriptive names for the quantities to be entered.
[ ]	Square brackets denote optional quantities.
|	A bar denotes an OR condition. For example, W|B means either W or B can be used.
---	Underscore indicates user input.
(CR)	Return and line feed.

Apply the following when entering commands and options:

1. All commands and options must be entered in uppercase.

2. Commands can be abbreviated to their first letter.

3. Numbers are represented in hex notation and must begin with a numeric digit.

4. The first character typed on a new line identifies which command is being invoked. If an invalid character is entered, a "?" is displayed, prompting a new command.

5. Addresses are specified by an optional segment number in angle brackets, followed by a hex address. For example, <00>4000 <00>0 <01>F800.

3.3.1 Monitor Mode Commands

Summary of Commands in Monitor Mode

NAME	PARAMETERS
DISPLAY	<address> [<# of long words/words/bytes>] [L|W|B] Display and alter memory
REGISTER	[<register name>] Display and alter registers
BREAK	<address> [<n>] Set and clear breakpoint
NEXT	[<n>] Step instruction
GO	Branch to last PC
JUMP	<address> Branch to address
FILL	<address1> <address2> <data> Fill memory
IOPORT	<port address> [W|B] I/O port read/write
MOVE	<address1> <address2> Move memory block
COMPARE	<address1> <address2> <n> Compare memory block
QUIT	Enter Transparent Mode

SIOPORT	<port address> [W|B] SIO port read/write
TEST	Enter Test Mode
ZBOOT	[D|T] Read a 512-byte program from disk or tape and execute

NOTE

All outputs in Monitor Mode can be suspended with XOFF (%13) control-s and resumed with XON "(%11) control-q.

COMMAND DESCRIPTIONS

DISPLAY

Syntax
DISPLAY <address> <# of long words/words/bytes> [L|W|B]

Description
This command displays at the terminal the contents of specified memory locations starting at the given address, for the given number of bytes.

If the L|W|B parameter is specified, the contents of the memory locations are displayed in hex notation and as ASCII characters.

If the L|W|B parameter is not specified, the memory locations are displayed one at a time, with an opportunity to change the contents of each location. For each location, the address is displayed, followed by the contents of L|W|B and a space. To change the contents at a given location, enter the new contents in the form long word|word|byte. If RETURN is pressed, either alone or after the new contents, the next sequential location is displayed. Entering a "Q" (for QUIT), followed by a RETURN terminates the command.

Example
Display memory starting at %5200 for ten words.
<D 5200 10 (CR)>

```
<00> 5200 1808 FE2B 2004 D923 7ED9 CD35 2238 0AED
  *...+..#...5"8..*
<00> 5208 6F23 ED6F 2B1E 0118 EDD9 2218 14D9 5778
  *o#.o+...H...Wx*
```

Example
Display memory starting at %5200 for 10 bytes.
<D 5200 10 B (CR)>

```
<00> 5200 18 08 FE 2B 20 04 D9 23 7E D9 CD 35 22
  38 0A ED *...+..#...5"8..*
```

Example
Display memory location %5200 and alter its contents.
<D 5200 (CR)>

```
<00> 5200 1808 <1922 (CR)

<00> 5201 FE2B <(CR)>
<00> 5202 2004 <Q(CR)>
```

REGISTER

Syntax
REGISTER [<register name>]

Description
The REGISTER command is used to examine or modify a specified register.

The following register names can be used in the command:

1. Any of the sixteen 16-bit registers named R0, R1, R2 ... R15.

2. Any of the sixteen 8-bit registers named RH0, RL0, RH1, RL1 ... RH7, RL7.

3. Any of the eight 32-bit registers named RR0, RR2, RR4 ... RR14.

4. Program counter register named RPC.

NOTE

The new contents of the program counter must be given in even hex numbers.

5. Flag and control word named RFC. If no register name is given, all registers R0, R1, R2 ... R15 PC and FCW are displayed. If a register name is given, the specified register name is displayed, followed by a space. To change the contents of that register, enter the new contents followed by a

RETURN, either alone or after the new contents. This displays the next register. A "Q" followed by a RETURN terminates the command.

Example
Display all registers.
<R (CR)

```
R0    R1    R2    R3    R4    R5    R6
0000  0000  0000  0000  0000  0000  0000

R6    R7    R8    R9    R10   R11   R12
0000  0000  0000  0000  0000  0000  0000

R13   R14   R15   RPC   RFC
0000  0000  0000  0000  0000
```

Example
Display 32-bit word register RR4 and alter its contents.
<R RR4 (CR)

```
RR4 00000000 <A257FFFF (CR)

RR6 00000000 <Q (CR))
```

BREAK

Syntax
BREAK <address> [<n>]

Description
The BREAK command sets a breakpoint at a given even address after clearing any previously set breakpoint. If <n> is given, program execution is not interrupted until the nth time the breakpoint instruction is encountered (<n> is in the range %1-%FFFF). If <n> is not given, 1 is assumed. If the BREAK command is issued with no parameters, any previously set breakpoint is cleared. When program execution is suspended by the BREAK command, the Monitor Program displays a message reporting the break and the address where it occurred.

Example
Message: BREAK AT 6A5E

NEXT

Syntax
NEXT [<n>]

Description
The NEXT command causes the execution of the next n machine instructions, starting at the current PC, and displays all registers after executing each instruction. (<n> is in the range %1-%FFFF.) If <n> is not given, 1 is assumed.

GO

Syntax
GO

Description
This command causes a branch to the current PC, continuing program execution from the location where it was last interrupted. All registers and the FCW are restored before branching.

JUMP

Syntax
JUMP <address>

Description
The JUMP command branches unconditionally to the given even address. All registers and the FCW are restored before branching.

Example
Execute user program starting at %5000.
<JUMP 5000 (CR)

FILL

Syntax
FILL <address1> <address2> <word data>

Description
The FILL command stores the given data word in a memory location, from address1 to address2. The command address must be an even hex number.

Example
Store data FFFF in memory from %5400 to %5410.
<F 5400 5410 FFFF (CR)

I/O PORT

Syntax
IOPORT <port address> [W|B]

Description
This command reads data in either byte or word form from the given port address and displays the value. Enter a hex value to be output to the specified port or enter only a carriage return if no output is to be made. If the W|B parameter is not given, byte data is read from the I/O port.

Example
Output data FF to port address %FF29.

```
<I FF29 (CR)

<00> FF29 00 <FF (CR)
```

MOVE

Syntax
MOVE <address1> <address2> <n>

Description
This command moves the contents of a block of memory from the source address specified by <address1> to the destination address specified by <address2>. <n> is the number of bytes to be moved.

Example
Move memory from address %5080 to %5090 for 100 bytes.

```
<M 5080 5090 100 (CR)
```

COMPARE

Syntax
COMPARE <address1> <address2> <n>

Description
This command compares the contents of two blocks of memory. <address1> and <address2> specify the starting addresses of the two blocks, and <n> specifies the number of bytes to be compared. If any locations of the two blocks differ, the addresses and contents of those locations are displayed.

Example
Compare two blocks of memory with starting addresses %4000 and %5000 for 20 bytes.

```
<C 4000 5000 20 (CR)
```

QUIT

Syntax
QUIT

Description
The QUIT command is used to enter Transparent Mode from Monitor Mode. In Transparent Mode, all keyboard inputs and console outputs are passed between the remote computer system and the S8000. The console controls the remote computer system's operating system. Channels A and B of the SIO2 must be set to the same baud rates when operating in Transparent Mode.

The START switch on the S8000 is used to return to Monitor Mode.

SIO PORT

Syntax
PORT <port addressw> [W|B]

Description
The PORT command is similar to the IOPORT command; however, it is used to read data from a Z8010A MMU.

TEST

Syntax
TEST

Description
The TEST command executes the S8000 standalone diagnostic tests.

Example
T (CR)

ZBOOT

Syntax
ZBOOT [D|T]

Description
This command is commonly used to manually bootstrap the ZEUS Operating System. The ZBOOT command reads a 512-byte program from block 0 of the disk or the cartridge tape drive. Generally, there is no return to the Monitor.

<u>Example</u>
<u>Z</u> <u>T</u> (CR)

3.3.2 Upload/Download Mode Commands

Summary of Commands in Upload/Download Mode

<u>NAME</u>	<u>PARAMETER</u>
LOAD	<filename>
	Load S/W from S8000 system

NOTE

Filenames can be specified in either upper or lowercase. They can be full path names. LOAD only loads data into segment 0.

Upload/Download Mode transfers data between the S8000 and a remote computer system. Channels A and B of the SIO2 must be set to the same baud rates when operating in Upload/Download Mode. The LOAD program is required on the remote system to perform upload/download functions through console I/O.

The Upload/Download Mode uses the Tektronix record format, which uses only ASCII characters. Each record contains two checksum values, a starting address, and a maximum of 30 bytes of data. The format of the record is:

RECORDS 1 to n

```
/ <address(4)> <count(2)> <checksuml(2)> <data(2)>...
  <data(2)> <checksum2(2)> <carriage return>
```

where:

<address(4)>:	is the address of the first byte of data in the record (address is represented in four ASCII characters)
<count(2)>:	is the number of <data> in current record (two ASCII characters)
<checksuml(2)>:	is the checksum for the address and count field (two ASCII characters)

<data(2)>:	is the value of byte data (represented in two ASCII characters)
<checksum2(2)>:	is the checksum for the data portion of the record (two ASCII characters)
<carriage return>:	indicates the end of the record

No segment information is transferred. All downloaded data is loaded into segment 0 with the LOAD command. Data for segments other than 0 must be transferred by the MOVE command.

LAST RECORD

/ <entry address(4)> 00 <checksum(4)> <carraige return>

where:

<entry address>:	is the starting execution address for the program
<checksum>:	is the checksum for the entry address

NOTE

A record with 00 in the count field indicates the end of load data.

RECORD WITH ERROR MESSAGE

If either the local or remote system has to abort the load process, it sends a record of the form:

// <error messages in ASCII text> <carriage return>

ACKNOWLEDGE

During the loading process, after each record is received from the remote system, an acknowledge (ASCII 0) is sent when the checksum values are verified. If a nonacknowledge (ASCII 7) is received, the remote system attempts to load the same data record up to ten times. After the tenth try, the Monitor Program returns to Monitor Mode for the next command. An abort-acknowledge (ASCII 9) is sent to the remote system if the escape (ESC) key is pressed, aborting the loading process. The Monitor Program then returns to Monitor Mode for the next command. The address used in the data record during the loading process is provided by the file description record; it must be greater than %4000 (%2000 through %4000 are used by the Monitor Program).

COMMAND DESCRIPTION

LOAD

Syntax:
LOAD <filename>

Description:
This command downloads a Z8000 program named <filename> that resides in the remote system.

The Monitor Program transmits the exact command line to the remote system. The command causes a remote procedure file (LOAD) to be executed, to open the file specified by <filename>. The binary data in the file is converted to Tektronix record format and transmitted to the S8000. The Monitor Program verifies the two checksum values in the receiving record and stores the data in RAM memory as specified by the address indicated in the record. An acknowledgment from the S8000 causes the next record to be downloaded from the remote system. A nonacknowledgment from the S8000 causes the current record to be retransmitted up to ten times, after which a record with an error message is sent, and the Monitor Program returns to Monitor Mode. The LOAD program in the remote system is also aborted. When the loading process is completed, the entry point received on the first record is displayed. Pressing ESC aborts the LOAD command. Any breakpoints set from a previous program must be cleared before a new program is loaded from the remote system.

Possible error messages:

/ABORT
/UNABLE TO OPEN FILE (XX), where (XX) is the ZEUS error code from the remote system
/FILENAME ERROR
/NOT PROCEDURE FILE
/ERROR IN READING FILE (XX), where (XX) is the ZEUS error code from the remote system
/RECORD CHECKSUM ERROR
/INCORRECT LOAD ADDRESS

Example:
Transfer file named MYFILE from the remote system to the S8000 RAM memory.

```
<LOAD MYFILE (CR)
```

NOTE

The address of RAM memory and the entry address used in the download process are provided by the information in the descriptor record of the file specified by <filename> in the LOAD command.

3.4 System Parameters

The following system parameters are accessible to the user:

NAME	PARAMETER
NULLCT:	Null Count (%23F6) This address stores the number of null characters that are inserted after a line feed. Modifying the null count adapts the S8000 to the return delays of various terminals. NULLCT is initialized to 0.
LINDEL:	Line Delete (%23F3) This address stores the character intercepted by the input line procedure as a line delete. When it is read from the terminal, this procedure purges the buffer and continues reading the input stream. LINDEL is initialized to %7F (RUB).

CHRDEL:	Character Delete (%23F2) This address stores the character intercepted by the input line procedure as a character delete. When it is read from the terminal, the last character entered is purged from the input buffer. Multiple character deletes can be used to delete the last n characters entered. CHRDEL is initialized to %08 (control-h).
XOFCHR:	XOFF Character (%23F5) The character stored at this address is interpreted by the input interrupt procedure as a character that stops outputting data to the terminal. When it is read from the terminal, all output is suspended until an XONCHR is received. XOFCHR is initialized to %13 (control-s).
XONCHR:	XON Character (%23F4) The character stored at this address is interpreted by the input interrupt procedure as a character that resumes output after XOFCHR is entered. When it is read from the terminal, all output is resumed. XONCHR is initialized to %11 (control-q).
STACK:	Stack Pointer (%20A0) This address is the base of the user stack set by the Monitor Program at reset. The top of the stack is %4000.
PSAREA:	Program Status Area (%2400) The Program Status Area for entering various interrupts and trap handling procedures starts at this address. This area includes the program status blocks (FCW and PC) for different types of interrupts and traps. The S8000 Monitor Program sets up these program status blocks as shown in Table 3-1.

Table 3-1. Program Status Area

WORD	VALUE	COMMENT
0-1	unused	
2-3	unused	RESERVED
4-5	unused	Unimplemented instruction
6-7	unused	
8-9	unused	PRIVILEGED INSTRUCTION
A-B	unused	
C-D	%4000	SYSTEM CALL entered in Segmented Mode
E-F	#BREAK	Address of BREAK interrupt procedure
10-11	unused	SEGMENT TRAP
12-13	unused	
14-15	%4000	FCW for NONMASKABLE interrupt procedure
16-17	#NMINT	Address of NONMASKABLE interrupt procedure
18-19	unused	
1A-1B	unused	NONVECTORED INTERRUPT
1C-1D	%4000	FCW for all VECTORED INTERRUPTS
1E-1F	unused	VECTOR 0
20-21	unused	VECTOR 2
22-23	unused	VECTOR 4
24-25	unused	VECTOR 6
26-27	unused	VECTOR 8
28-29	unused	VECTOR A
2A-2B	unused	VECTOR C
2C-2D	unused	VECTOR E
2E-2F	unused	VECTOR 10
30-31	unused	VECTOR 12
32-33	#PTYINT	VECTOR 14 (SIO Channel B input interrupt procedure address)
34-35	#CHASRC	VECTOR 16 (SIO Channel B special receive condition procedure address)
36-37	unused	VECTOR 18
38-39	unused	VECTOR 1A
3A-3B	#MCZINT	VECTOR 1C (SIO Channel A input interrupt procedure address)
3C-3D	#CHASRC	VECTOR 1E (SIO Channel A special receive condition procedure address)
3E-3F	unused	VECTOR 20
40-41	unused	VECTOR 22
42-43	unused	VECTOR 24
44-45	unused	VECTOR 26
46-47	unused	VECTOR 28
48-49	unused	VECTOR 2A
4A-4B	unused	VECTOR 2C
4C-4D	unused	VECTOR 2E
4E-4F	unused	VECTOR 30

The port addresses shown in Table 3-2 are used in the Monitor Program.

Table 3-2. System Hardware I/O Port Addresses

PORT	ADDRESS
CTC CHANNEL 0	FFA1
CTC CHANNEL 1	FFA3
CTC CHANNEL 2	FFA5
CTC CHANNEL 3	FFA7
SIO DATA CHANNEL A	FF81
SIO DATA CHANNEL B	FF83
SIO CONTROL CHANNEL A	FF85
SIO CONTROL CHANNEL B	FF87
RETI PORT	FFE1
SWITCH BANK (SPEED)	FFC1

SECTION 4

MONITOR I/O PROCEDURES

4.1 Introduction

The I/O procedures most frequently used in the Monitor Program are given in this section. These procedures are accessed by system calls in user programs to perform console I/O functions.

4.2 I/O Procedures

TYIN

Description
Gets a character from the keyboard buffer. If the buffer is empty, this procedure waits for a character to appear. The character is stored in register RL0, and the contents of register RH0 are lost.

Example
```
CONSTANT
    TYIN := %04
          .
          .
          .
    SC    #TYIN
    (character in RL0)
```

TYWR

Description
Displays the character in RL0. The character is not displayed if the XOFF character has been received before this procedure is executed. In this case, the procedure waits until an XON character is received from the console before displaying the character in RL0. If the character to be displayed is a carriage return, the zero flag is set, and RH0 is lost.

<u>Example</u>

```
CONSTANT
    TYWR := %06
          .
          .
          .
    SC    #TYWR
    (character in RL0)
```

PUTMSG

<u>Description</u>
Sends a character string to the terminal. Register R2 contains the address of the character string buffer, and the first byte in the buffer contains the number of characters to be displayed. If there is no return in the string, the entire specified string is displayed. Otherwise, the string is displayed up to and including the first return. Register contents R0, R1, and R2 are lost.

<u>Example</u>

```
CONSTANT
    PUTMSG:= %0C
        .
        .
        .
    (string address in R2)
    SC  #PUTMSG
```

TTY

<u>Description</u>
Receives and echoes at the terminal a character string up to the first return. The character string is stored in a buffer pointed to by register R2. Register R1 contains the size of the buffer. If the size of the character string exceeds the size of the buffer, the zero flag is set. All lowercase alpha characters are converted to uppercase characters before they are stored in the buffer. R1 returns the actual number of characters received from the terminal. The contents of registers R0 and R2 are lost.

<u>Example</u>

```
CONSTANT
    TTY  := %08
          .
          .
          .
    SC    #TTY
    (string address in R2, size in R1)
```

CRLF

Description
Outputs a return followed by a line feed to the terminal. The contents of register R0 are lost.

Example
```
CONSTANT
    CRLF := %0A
          .
          .
          .
    SC     #CRLF
```

APPENDIX A

GLOSSARY

The most important terms and acronyms introduced in this manual are listed in this Appendix.

address	A number that specifies one particular element in a set of similar elements. May be either a memory address or an I/O address. (See also segmented address, logical address, physical address.)
address space	A set of addresses. The Z8000 can access eight separate address spaces: normal-mode program memory space, system-mode program memory space, normal-mode data memory space, system-mode data memory space, normal-mode stack memory space, system-mode memory space, standard I/O space, and special I/O space.
addressing mode	The way in which the address of an operand is specified. There are eight addressing modes: Register, Immediate, Indirect Register, Direct Address, Index, Base Address, Relative Address, and Base Index.
autodecrement	The contents of a register are decremented and then used, as specified, by the instruction.
autoincrement	The contents of a register are used, as specified, by the instruction and then incremented.
Base Address (BA) addressing mode	A based address consists of a register that contains the base and a 16-bit displacement. The displacement is added to the base and the resulting address indicates the effective address. In nonsegmented mode, the base address is held in a word register and the displacement is in the instruction. In segmented mode,

	the segmented base address is held in a register pair and the displacement is in the instruction.
Base Index (BX) addressing mode	
	Based Indexed addressing is similar to Based addressing except that the displacement (index), as well as the base, is held in a register. In nonsegmented mode, the base address is held in a word register and the index is held in a word register. In segmented mode, the segmented base address is held in a register pair and the index is held in a word register.
BCD digit	A Binary Coded Decimal digit is encoded of the ten decimal digits into a 4-bit code that is simply the first ten binary numbers in the binary number system (starting with 0). This code is used to represent and process numbers in the base-10 (decimal) format.
break	The break is a built-in command used to exit from loops within the control structure of the shell. (See shell.)
breakpoint	A command that interrupts or stops program execution at a specified address in the program. The address specified in the breakpoint is the address of the instruction.
byte	A byte is eight contiguous bits; a byte in memory starts on an addressable byte boundary.
byte register	An 8-bit register. The Z8000 CPU contains 16 general-purpose byte registers, designated RLn and RHn (n = 0-7).
code	The characters of an originating or source language, each correlated with its equivalent expression in an intermediate or target language, for example, alphanumeric characters correlated with their equivalent 6-bit expressions in a binary machine language.

command	A function performed by the system, either by the shell or by a program residing in a file in the ZEUS system.
context switching	Interrupting the activity in progress and switching to another activity. A context switch involves saving for later restoration the contents of the general-purpose registers, the Program Counter and the Flag and Status Word.
CONTROL	The CONTROL key, labeled CTRL, generates and sends control instructions to the terminal during command execution. Any character typed with the CTRL key pressed is transparent to the user, but is recognized by the CPU.
CPU (central processing unit)	The unit of a computing system that includes the circuits controlling the interpretation of instructions and their execution.
data structure	A logical organization of primitive elements (e.g byte or word) whose format and access conventions are well defined. Examples of data structures are tables, lists, and arrays.
data type	The way in which bits are grouped and interpreted. For an instruction, the data type of an operand determines its size and the significance of its bits. Operand data types include byte, word, long word, byte string, word string, and BCD digit.
debugging	Debugging is the process of correcting mistakes in programs and shell scripts. The shell has several options and variables that can be used to aid in shell debugging.

diagnostic	An error message produced by a program is often referred to as a *diagnostic*. Most error messages are not written to the standard output, since that is often directed away from the terminal. Instead, error messages are written to the *diagnostic output*, which usually appears on the terminal.
Direct Address (DA) addressing mode	In this mode, the operand address is contained within the instruction.
directory	A structure that contains files is called a *directory*. The directory in which the user first logs in is the *home* directory.
disk	A flat circular plate with a magnetic surface on which data can be stored by selective polarization of portions of the flat surface.
displacement	A number contained in the instruction for use in calculating the *effective address* of an operand. The displacement is added to the contents of a register during the calculation.
DMA	Direct Memory Address is a method for transferring data to or from main memory at high speed by avoiding the CPU registers.
effective address	The address obtained after indirect or indexing modification. In non-segmented mode, the effective address is a 16-bit number. In segmented mode, the effective address consists of a 7-bit segment number and 16-bit offset. In systems with memory management, the effective address is the logical address which must be translated to obtain the physical memory address.

EOF	An *end-of-file* is generated whenever a command reads to the end of a file that it has been given as input. It can also be generated at the terminal with a *control-d*. Commands receiving input from a *pipe* receive an end-of-file when the command sending them input completes. Most commands terminate when they receive an end-of-file. The shell has an option to ignore end-of-file from a terminal input, which makes it possible to avoid logging out accidentally by typing too many control-d's.
file name	Each file in ZEUS has a name consisting of up to 14 characters, not including the slash character (/), which is used in path name building. Most file names do not begin with the period character. They contain only letters and digits, with perhaps a period separating the root portion of the file name from an extension.
home directory	Each user has a home directory, which is given in the password file /*etc*/*passwd*. The user is placed in the home directory when first logging in. The *cd* or *chdir* command with no arguments returns the user to this directory. The name of this directory is recorded in the shell variable *home*.
Immediate (I) addressing mode	In this mode, the operand is contained within the instruction.
Index (X) addressing mode	In this mode, the operand address is obtained by adding the contents of an index register to a base address contained in the instruction.
index register	A word register used to contain a displacement for use in effective address calculation.

Indirect Register (IR) addressing mode	In this mode, the operand address is contained within a register.
input	Information taken from the terminal or from files is called *input*. Commands normally read input from their *standard input* which is, by default, the terminal. The metacharacter followed by a file name can be used to cause input to be read from a file. Many commands also read from a file specified as argument. Commands placed in pipelines are read from the output of the previous command in the pipeline. The leftmost command in a pipeline reads from the terminal if its input is not redirected and if a file name is not given to use as standard input. (See pipeline.)
interrupt	An *interrupt* is a signal that causes most programs to stop execution. It is generated by pressing the RUB key. Certain programs such as C shell and the editors handle an interrupt in special ways, usually by stopping what they are doing and prompting for another command.
interrupt request	An event other than a trap or jump or call instruction that changes the normal flow of instruction execution. (See nonmaskable interrupts.)
interrupt service routine	The routine executed in response to an interrupt.
interrupt/trap acknowledge transaction	The transaction initiated by the CPU in response to an interrupt or trap. Obtains an identifier word from the interrupting device or memory management hardware.

I/O address	The address of an I/O port, always 16 bits long. Word ports may have even or odd addresses, Special I/O byte ports are even, and Standard I/O byte ports are odd.
I/O transaction	A transaction that transfers data to or from a peripheral device or memory management hardware.
Kernel	The system software task that manages task scheduling and intercommunication for the Zilog S8000 system. The Kernel provides process management, file management, and input/output (I/O) processing.
logical address	The address manipulated by the programmer; used by instructions and output by the Z8001.
.login	The file .*login* in the user's *home* directory is read by the C shell each time the user logs in to ZEUS; the commands there are executed.
logout	The *logout* command causes a login shell to exit. Normally, a login shell exits when control-d is pressed, generating an end-of-file (EOF).
.logout	When a user logs off of ZEUS, the shell prints .*logout* and executes commands from the file .*logout* in the user's *home* directory.
long word	A long word is 32 contiguous bits; a long word in memory starts on an even addressable byte boundary.
memory address	An address specifying a location in memory. Word and long-word addresses must be even, byte addresses may be even or odd.

memory management	The process of translating logical addresses into physical address plus certain protection functions.
memory transactions	A transaction that transfers data to or from main memory.
Monitor	The S8000 Monitor sets software breakpoints for program debugging; includes I/O control, interface software for use with a serial interface to a remote computer system, and the primary bootstrapper used to bring the system up.
normal mode	A running-state mode where the S/N flag in the FCW is 0 and the N/S line is High. In this mode, the CPU may not execute privileged instructions.
non-maskable interrupts	Interrupts which cannot be disabled.
nonsegmented mode	A Running-state mode of the Z8001 CPU. In this mode, all addresses are generated with the same segment number.
non-vectored interrupts	Interrupts which do no use the identifier word as a vector to an interrupt service routine.
operand	An item of data operated on by an instruction.
output	Many commands in ZEUS produce data that is called output. This output is usually placed on what is known as the standard output, which is normally connected to the user's terminal. The shell has a syntax using the metacharacter > for redirecting the standard output of a command to a file. Using the pipe mechanism and the metacharacter |,

it is also possible for the standard output of one command to become the standard input of another command. Some commands do not direct their output to the standard output, the line printer command (lpr), for example, diverts its output to the line printer. The write command places its output on another user's terminal. Commands also have a diagnostic output where they write their error messages. Normally, these go to the terminal even if the standard output has been sent to a file or another command, but it is possible to direct error diagnostics along with standard output using a special mentanotation.

path name	A list of names, separated by a slash (/) characters forms a path name. Each component between successive / characters names a directory in which the next component file resides. Path names that begin with the character / are interpreted relative to the root directory in the file system. Other path names are interpreted relative to the current directory as reported by pwd. The last component of a path name can name a directory; however, it usually names a file.
physical address	The address required for accessing the memory, obtained from the logical address generated by the Z8001 by memory management hardware, for example, the Z8010 Memory Management Unit.
pipeline	A group of commands that are connected together with the standard output of each connected to the standard input of the next is called a pipeline. The pipe mechanism used to connect these commands is indicated by the vertical bar (|) metacharacter.

privileged instruction	An instruction intended for use primarily by an operating system, which can be executed only in system mode. In general, instructions that change the processor state or perform I/O are privileged.
program	A *program* (usually synonymous with *command*) is a binary file that performs a useful function.
Program Counter (PC)	One of the two *Program Status registers*. Contains the address of the current instruction word.
Program Status Area	The area in memory reserved for the starting program status of the interrupt and trap service routines.
Program Status Area Pointer	The register that contains the starting address of the Program Status Area.
Program Status registers	The two registers (PC and FCW) that contain the program status.
prompt	Many programs print a *prompt* on the terminal when they expect input. For example, the shell prompts for input with a percent sign (%).
pwd	The *pwd* command prints the full path name of the current working directory.
register	A storage location in hardware logic other than the memory. Bits within a register are numbered from 0, with the least significant being the rightmost. (See also byte register, word register, register pair, and register quad.)

Register (R) addressing mode	In this mode, the operand is in a general-purpose register.
register pair	One of eight pairs of general-purpose word registers, designated RRn (n = 0, 2, 4, ..., 12, 14).
register quad	One of four groups of four word registers, designated RQn (n = 0, 4, 8, 12).
Relative Address (RA) addressing mode	In this mode, the operand address is calculated by adding a displacement found in the instruction to the current PC value.
reset	An internal CPU operation that initializes the Program Status registers. It is activated by the RESET line.
RETURN	The RETURN key on the terminal is used to execute commands as they are entered.
RUBOUT	The RUBOUT key generates an interrupt signal that is used to stop programs or to cause them to return and prompt for more input.
Running state	One of the three CPU states. In this state, the CPU is fetching and executing instructions or handling interrupts.
segment	In a Z8001, a set of adjacent memory addresses (up to 64K) with the same segment number on lines SN0-SN6.
segment number	A number specifying a memory segment. Placed on the SN0-SN6 lines during memory transactions in the Z8001 system. Part of a segmented address.

segmented address	In Z8001 CPU's, a 23-bit value consisting of a 7-bit <u>segment number</u> and a 16-bit <u>offset</u>.
segmented mode	One of the Running-state modes of the Z8001 CPU. In this mode, the CPU generates addresses that can have different segment numbers.
shell	A shell is a command language interpreter. It is possible for users to write and run their own shells, as shells are no different from any other program in terms of system response.
stack	A data structure used for temporary storage or for procedure and interrupt service routine linkages. A stack uses the last-in, first-out concept. As items are added to, or pushed onto, the stack, the <u>stack pointer</u> decrements; as items are removed from, or popped off, the stack, the <u>stack pointer</u> increments.
stack pointer	A general-purpose register indicating the top (lowest address) of a stack.
status	A command normally returns a <u>status</u> when it finishes. By convention, a <u>status</u> of zero indicates that the command succeeded. Commands can return non-zero status to indicate that some abnormal event has occurred.
status flags	Status flags are set according to the outcome of certain instructions to direct the subsequent flow of the program as necessary. There are six status flags: Carry, Zero, Sign, Parity/Overflow, Decimal Adjust and Half Carry. The first four are grouped together to determine the condition code, the last two are used in programs manipulating <u>BCD digits</u>.

system mode	A Running-state mode in which the S/N flag in the FCW is 1 and the N/S line is Low. In this mode, the CPU may exercise <u>privileged instructions</u>.
termination	When a command being executed finishes, it is said to <u>terminate</u>. Commands normally terminate when they read an end-of-file from their standard input.
trap	A condition that occurs at the end of an instruction that caused an illegal operation. The Z8000 traps are internal traps arising from system call, unimplemented instruction and privileged instructions executed in normal mode, and an external trap, the segmentation trap, arising from memory access violations in systems with memory management. A trap is similar to an interrupt in that it causes the executing program Status registers to be saved on the system stack. Traps cannot be disabled.
vectored interrupts	Interrupts which use the identifier word as a vector to the <u>interrupt service routine</u>. May be disabled.
word	Two contiguous bytes (16 bits) starting on an even addressable byte boundary. Bits are numbered from the right, 0 through 15. A word is identified by the address of the byte containing the most significant bit, bit 15.
word register	A 16-bit register.
working directory	Any directory a user is currently working in is called a <u>working directory</u>.
ZEUS	<u>ZEUS</u> is the operating system for the S8000 system.

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