SYSTEM MONITOR BOARD

User's Manual

TECHNICAL DESIGN LABS, INC.

RESEARCH PARK, BUILDING H

1101 STATE ROAD

PRINCETON, NEW JERSEY 08540

26 August 1977

ERRATA SHEET FOR THE SMB

1. On page 5 under TTY 20ma CURRENT LOOP: Insert the following note:

   Current loop operation above 600 buad is not recommended. Above 600 baud, RS232 should be used. However, over 600 baud the value of C13 and/or R22 must be changed to reflect a shorter time constant. Decreasing the value of C13 will accomplish this. (e.g. .01mfd or .001mfd).

2. On page 7 under VIDEO 20ma CURRENT LOOP:

   REFER TO THE NOTE ABOVE.

3. On page 9 under 2.2 BAUD RATE JUMPERS:

   2400 should be added to list of baud rates.

4. On page 30:

   Change V Ø F7FF FØØØ (return)

   to

   V Ø Ø7FF FØØØ (return)

Technical Design Labs

TABLE OF CONTENTS

System Monitor Board User's Guide

INTRODUCTION:

The System Monitor Board (SMB) is designed to be used with the Technical Design Labs Z-80 CPU board, the ZPU. As such it is meant to operate in an S-100 bus computer mainframe such as TDL's Xitan series. The SMB is extremely versatile in that it will perform the functions of several boards all on one. The following is a list of it's principal functions:

1. 2K MONITOR operating system (the ZAPPLE(tm) Monitor) in ROM (read only memory).
2. 2K RAM, read/write storage for user programs, auxiliary monitor routines, and/or stack area.
3. 2 SERIAL I/O PORTS, 110-9600 baud, RS232 or 20ma current loop for each port, and software-initialized via the monitor. Can be used for TTY, CRT terminal, or other serial device.
4. PARALLEL I/O PORT, bi-directional, can be configured as an input or output port or dynamically changed under program control.
5. SENSE SWITCH INPUT PORT, allows the user to specify the I/O configuration which the system will initialize to.
6. 1200 BAUD CASSETTE INTERFACE- Allows rapid loading and dumping of programs/data in either a checksummed hex file or a binary image format.
7. JUMP TO MONITOR- Contains circuitry for automatically jumping to the monitor at F000H prompted by a power on clear (POC) and/or reset.

SECTION 1

CONNECTION

How to hook up various I/O devices to the SMB.

1.1 TTY

TTY stands for Teletype (a registered trademark for the Teletype Corp.) which is the brand name for the most popular teleprinter. You may hook up your TTY to either of the two serial I/O ports. That is, the one labelled "TTY" or the one labelled "VIDEO". The difference between the two ports lies in the fact that the monitor will handle paper tape input and output via the "TTY" and not via the "VIDEO" port.

TTY RS232:

Using the standard EIA 25 pin connector which should be wired to the proper places on the TTY, the following connections should be made:

TTY (DB25)	SMB (J1)*
1 Frame Ground	16 Ground
2 Transmit	13 TTY Input RS232
3 Receive	11 TTY Output RS232
4 Request to Send	10 TTY 20ma IN **
5 Clear to Send	14 minus (-) 12 volts **
6 Data Set Ready	8 TTY 20ma OUT
7 Signal Ground	* J1 on the SMB is the blue
8 Rec'd Line Detect	Ansley ribbon cable connector
20 Data Terminal	at the top right corner of the
Ready	board.
	** Lift pin 8 of U32 (1489) out
	of the socket if this jumper
	is not installed.

TTY RS232 Checklist:

Make the following connections on the 25 pin RS232 connector (DB25) going to your TTY.

( ) Connect a jumper between pin 4, Request to Send, and pin 5, Clear to Send.

( ) Connect a jumper between pin 6, Data Set Ready, pin 8, Received Line Signal Detect, and pin 20, Data Terminal Ready.

( ) Connect a jumper between pin 1, Frame Ground, and pin 7, Signal Ground.

Make the following connections on the SMB's J1-- the blue Ansley ribbon cable connector at the top right of the board.

( ) Connect a jumper between pin 10, TTY 20ma IN, and pin 14, Minus (-) 12 Volts. Note: This is to disable the 20ma loop circuit. The RS232 circuit will not work if this is not done. The same disabling can also be accomplished by lifting pin 8 of IC U32 (1489) out of its socket.

Make the following connections between the DB25 connector and J1 on the SMB.

( ) Connect pin 1 of the DB25 to pin 16 of J1.
( ) Connect pin 2, Transmit, of the DB25 to pin 13, TTY RS232 IN, of J1.
( ) Connect pin 3, Receive, of the DB25 to pin 11, TTY RS232 OUT, of J1.

TTY 20ma CURRENT LOOP:

Connection to the TTY's current loop can be made at either the Terminal Strip (TS) or J2. Note: In this reference, J1 refers to the jack on the SMB and J2 refers to the jack on the TTY. Follow the following procedure:

( ) Connect pin 8 of J1 to either pin 7 of TS or pin 8 of J2.
( ) Connect pin 10 of J1 to either pin 4 of TS or pin 6 of J2.
( ) Connect pin 14 of J1 to either pins 3 and 6 of TS or pins 5 and 7 of J2.
( ) Make sure pin 8 of IC U32 (1489) has not been removed from its socket.

CONVERTING A TELETYPE TERMINAL FROM HALF- TO FULL-DUPLEX OPERATION

To convert a Teletype terminal connected for half-duplex operation to full-duplex operation, the following modifications should be made.

1. Locate the black terminal strip in the back of the data terminal. See Fig. 5 on Page 6.

2. Move the brown/yellow and white/blue wires from pins 3 and 4 to pin 5.

CONVERTING A TELETYPE TERMINAL FROM 60-ma to 20-ma OPERATION

To convert a Teletype terminal connected for 60-ma operation to 20-ma operation, the following modifications should be made.

1. Locate the black terminal strip in the back of the data terminal. See Fig. 5.

2. Move the violet wire from pin 8 to pin 9.

3. Move the blue wire connected to the current source resistor (a flat green resistor having four tabs located to the right of the keyboard) from the 750-ohm tab to the 1450-ohm tab.

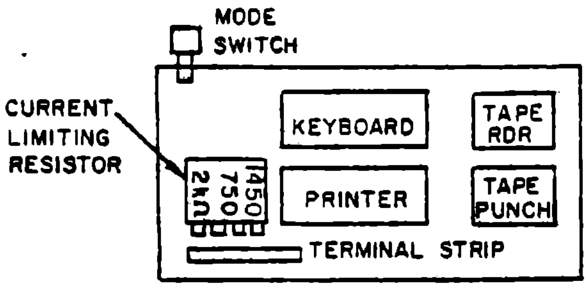

FIG. 5a

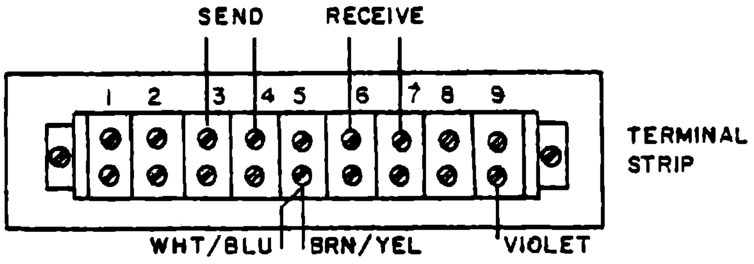

FIG. 5b

1.2 VIDEO

You may connect either a TTY or a CRT terminal to the "Video" port, however, a CRT terminal is usually connected. The connections are very similar to the TTY connections. Refer to the TTY section for any abbreviations used here without explanation. The explanations are not repeated.

( ) Make the same 3 jumper connections on the DB 25 as described for the TTY (i.e.- 4 to 5, 6 to 8 and 20, and 1 to 7).
( ) Connect a jumper between pins 12 and 14 of J1. Note: This disables the video port's 20 ma. loop circuit. This may also be done by removing pin 6 of IC U32 (1489) from its socket.
( ) Connect pin 1 of DB 25 to pin 16 of J1.
( ) Connect pin 2, Transmit, of DB 25 to pin 15, VIDEO RS232 IN, of J1.
( ) Connect pin 3, Receive, of DB 25 to pin 9, VIDEO RS232 OUT, of J1.

VIDEO 20 MA CURRENT LOOP:

There should be two sets of 2 wires. One set for the keyboard and one set for the screen. Proceed as follows:

( ) Take negative wire from each of the two sets of wires and connect them together. Then connect that junction to the minus (-) 12 volts on J1 (pin 14).
Note: Some terminals provide their own -12 volts and must be connected to it to work in the current loop mode. Check the terminal's manual.
( ) Connect the one remaining wire from the terminal's keyboard circuit to the VIDEO 20MA IN (pin 12) of J1.
( ) Connect the one remaining wire from the terminal's screen circuit to the VIDEO 20MA OUT (pin 6) of J1.

1.3 CASSETTE:

At the top of the SMB, to the right of U33, there are three connection points. The far left of these is labelled with an "R" and an arrow pointing up. The center one is labelled "GND" for ground. And the far right one is labelled with a "P" and an arrow pointing downward. The following is the connection procedure:

( ) Connect the shields of two shielded audio cables together and then connect them to the center ("GND") terminal.
( ) Connect the center wire of the one going to the recorder's input to the left terminal ("R").
( ) Connect the center wire of the cable coming from the recorder's output to the right terminal ("P").

1.4 PARALLEL PORT:

See the schematic for the pin designations on J1 which pertain to the parallel port. They are labelled "PB" for the port's data bits and "CB" for the port's control bits.

SECTION 2

SETTINGS

How to configure jumpers and dip switches.

2.1 POC/PRESET JUMPER

At the bottom of the SMB and below IC U9 there is a jumper arrangement. There are three terminals comprising the jumper arrangement. One is not labelled and the other two are labelled POC and PRESET respectively. Install a jumper between the unlabelled terminal and one of the others. If you connect the jumper to the POC, a jump to the monitor will occur when the power is turned on. If TDL's ZPU board is used, the jump to the monitor will also occur during a reset since the ZPU issues a POC during reset. Should this be undesirable in your system it can be easily disabled by severing the connection between the POC and reset line on the ZPU. Consult the ZPU schematic to find this connection. If you connect the jumper to the PRESET, a jump to the monitor will occur only when the reset switch is activated.

2.2 BAUD RATE JUMPERS:

Immediately above IC U24 (14411) in the upper left of the board, there are the "baud rate jumpers". There are two main terminals labelled "TTY" and "VIDEO" respectively. To their left is a row of Augat pins which may not have a silk-screened label indicating their function. These are the individual baud rate terminals. From left to right they are: 9600,4800,1200,600,300, and 110 baud. To set the SMB for the correct baud rate for the "TTY" port, merely place the jumper wire from the "TTY" terminal into the baud rate terminal which corresponds to that of the "TTY" port's terminal (usually 110). If a terminal is connected to the "VIDEO" port, place the VIDEO jumper into the proper baud rate pin (usually 9600). If the baud rate of both the "TTY" and the "VIDEO" ports are the same, there is an extra pin on each so that either one may be plugged into the other first and the remaining one plugged into the correct baud rate pin.

2.3 MEMORY PROTECT SWITCHES- S1 & S2:

In the upper left of the board there is a 4 bit dip switch. The two top switches are S1 and S2 respectively. Each protects a 1K segment of the SMB's 2K RAM memory. S1 protects the RAM from F800 to FBFF and S2 protects the RAM from FC00 to FFFF. Depressing the right side of the switch protects the RAM while depressing the left side causes it to be unprotected (may be written into as well as read from).

2.4 CASSETTE SWITCHES- S3 & S4:

Located directly below and on the same 4 bit dip switch as the RAM protect switches are two more switches, S3 and S4 respectively. S3 is used to choose either a microphone (MIC) of auxiliary (AUX) input for the cassette interface. This setting must match with the type of input your recorder has. Some have both. Depressing the right side of the switch selects the AUX input and depressing the left side selects the MIC input. S4 is the bottom switch. This switch causes the data coming into the interface to be inverted. If a particular cassette recorder has an odd number of inverting audio stages in its input circuit, the data recorded on the tape will be inverted. If the number of inverting audio stages in its output is odd, the data will appear inverted from what was recorded on the tape. Thus two conditions exist. One in which the data is either inverted or not on RECORD. And the other in which the data is either inverted or not on PLAYBACK. When recording a tape and playing it back on the same unit, it is a simple matter to determine the position of the invert switch. It will be the same for all such recordings. However, when playing back tapes recorded on another unit, the switch should be tried in the opposite position if the recording unit did not have the same inversion as yours. Trial and error will find the correct setting quickly.

2.5 SENSE SWITCHES

This port consists of an 8-bit dip switch which is located in the upper right corner of the System Monitor Board and is examined by the operating system everytime initialization occurs (i.e.- whenever the system is reset). It will determine which of the user's I/O devices is to be assigned to each one of the four (4) LOGICAL DEVICES. The four LOGICAL DEVICES are as follows:

a. CONSOLE - The console is defined as the device which enables the user to communicate to the computer and observe it's response. Teletypes (tm) and CRT terminals are the most commonly used console devices.
b. TAPE READER - The tape reader is defined as the device which allows the computer to input from paper or magnetic tape. TTY paper tape readers, high speed paper tape readers, and cassette decks are typical tape reader devices.
c. TAPE PUNCH - The tape punch is defined as the device which allows the computer to output to paper or magnetic tape. TTY paper tape punches, high speed paper tape punches, and cassette decks are typical tape punch devices.
d. LIST DEVICE - The list device is defined as the device used for hard copy output. It is typically a line printer.

The dip switch is physically arranged upside down on the board. This is done so that while looking at the top row of switch paddles, a logical 1 is represented as a pushed in paddle and a logical 0 as a protruding paddle (accomplished by depressing the BOTTOM switch paddle). If you take a look at the switch you will notice that each of the eight switches is numbered starting with 1 on the right and ending with 8 at the left.

Switches 1 & 2 specify the CONSOLE device.
Switches 3 & 4 specify the TAPE READER device.
Switches 5 & 6 specify the TAPE PUNCH device.
Switches 7 & 8 specify tne LIST DEVICE.

The following is a table showing how the switches are configured to enable the various I/O devices to be assigned as one or more of the LOGICAL DEVICES:

LOGICAL DEVICE > > >	LIST		PUNCH		READER		CONSOLE	
Switch > > >	8	7	6	5	4	3	2	1
Teletype > >	0	0	0	0	0	0	0	0
Video <CRT>	0	1	x	x	x	x	0	1
High Speed * Paper Tape	x	x	0	1	0	1	x	x
Line Printer *	1	0	x	x	x	x	x	x
Cassette > >	x	x	1	0	1	0	x	x
Batch Mode(**)	x	x	x	x	x	x	1	0
User Defined(*)	1	1	1	1	1	1	1	1

NOTES: "x" means "not applicable"

(*) The user must supply his own driver software to interface these devices to the monitor's operating system.

(**) In the Batch Mode the assigned Reader becomes the Console input and the List Device becomes the Console output. Note that the computer will read from the assigned Reader as if it were a keyboard and consequently the bits on the tape are read and interpreted as an ASCII file rather than a binary dump or hex file as it normally would.

SOFTWARE CONTROL

Please note that the assignment of the I/O device to one of the LOGICAL DEVICES as is done by the 8-bit dip switch can also be accomplished under software control by using the ASSIGN command of the monitor's operating system. For more details on this consult the section dealing with the ZAPPLE MONITOR.

SECTION 3

OPERATION

This part of the manual covers the operation of the various parts of the System Monitor Board.

3.1 JUMP ON RESET

The System Monitor Board employs a special circuit that allows the user to exercise control over the computer by gaining access to the monitor system whenever the reset button is activated.

If the board is being used in a computer which has a front panel, the computer should be in the "RUN" mode. If the reset is then activated, the monitor will then sign on. The same is accomplished with just activation of the reset switch on those computers without front panels.

3.2 THE ZAPPLE MONITOR

The Zapple Monitor is a universal operating system with comprehensive DEBUG and I/O handling capabilities. It contains all the needed tools to fully debug both hardware and software as well as support the I/O used by the system. It has been instrumental in establishing the I/O independency of TDL software and thereby bringing it "big system" features not found elsewhere. All of our resident software contains NO I/O routines whatsoever! It handles all I/O through vectors at the beginning of the program. As long as the I/O vectors are honored, then a Basic, Fortran, Text Editor, etc. program does not have to be concerned about whether the system is running a keyboard with video monitor display, model 33, CRT terminal or whatever. Another feature of the Zapple Monitor is its expandability. This feature is of tremendous use as it allows the user to attach his own additional monitor routines at the end of the monitor. Such routines often include I/O drivers. Typical additions might be a VDM driver routine or Tarbell cassette driver routine. Specific routines submitted to TDL will be made available to other users via TDL's newsletter and user's library. The monitor also includes many useful subroutines that may be used by user written programs. For details on these, see the assembly listing elsewhere in this manual.

THIS MONITOR WILL COME TO BE THE MOST IMPORTANT PIECE OF SOFTWARE IN YOUR SYSTEM.

COMMANDS

The following is a list of commands for the Zapple Monitor. Precise definitions and usage notes are covered in the next section.

A - ASSIGN reader, punch, console or list device options from the console.
B - BYE (system shut down).
C - COMPARE the contents of memory with the reader input and display any differences.
D - DISPLAY the contents of any defined memory area in Hex.
E - END OF FILE statement generator.
F - FILL any defined area of memory with a constant.
G - GOTO an address and execute. With breakpointing.
H - HEX MATH. Gives the sum and difference of two Hex numbers.
I * USER DEFINED.
J - JUSTIFY MEMORY - a non-destructive test for hard memory failures.
K * USER DEFINED.
L - LOAD a binary file.
M - MOVE a defined memory area to another starting address.
N - NULLS to the punch device.
O * USER DEFINED.
P - PUT ASCII characters into memory from the keyboard.
Q - QUERY I/O ports - may output or input any value to or from any I/O port.
R - READ a Hex file. Performs checksum, relocating, offsetting, etc.
S - SUBSTITUTE and/or examine any value at any address (in hex).
T - TYPEs the contents of a defined memory block in their ASCII equivalent.
U - UNLOAD a binary tape to the punch device.
V - VERIFY the contents of a defined memory block against that of another block and display the differences.
W - WRITE a checksummed hex file to the punch device.
X - EXAMINE and/or modify any or all registers including the special Z-80 registers.
X'- EXAMINE and/or modify any or all of the Z80's prime registers.
Y - "Yis there". Search memory for defined byte strings and display all the addresses where they are found.
Z - "Z end". Locate and display the highest address in memory.

COMMAND SET USAGE

The following section lists the commands, and describes their format and their use. It should be noted that the Zapple Monitor recognizes both upper and lower case letters for its commands, and that in general, a command which is printing can be stopped with a CONTROL C, which is checked during a carriage return - line feed sequence. The following EXAMPLES show a comma [,] as a delimiter between parameters, however a space may also be used. If an error is made while inputting a command from the keyboard, it may be terminated by a rubout and the command re-typed. An asterisk is displayed indicating an ABORT of some kind.

COMMAND	DESCRIPTION
A	ASSIGNMENT OF I/O DEVICES: The monitor system is capable of supporting up to 4 logical devices, these being: The CONSOLE, The READER, the PUNCH, and the LIST DEVICE. To these may be connected 4 different actual I/O devices, for a total of 16 direct combinations of I/O device and function. The specific permutations are:

LOGICAL DEVICE	ASSIGNED DEVICES
CONSOLE	TTY VIDEO (CRT terminal) BATCH USER (user defined)
READER	TTY CASSETTE PAPER (HIGH SPEED READER user written) USER (user defined)
PUNCH	TTY CASSETTE PAPER (HIGH SPEED PUNCH user written) USER (user defined)
LIST DEVICE	TTY VIDEO (CRT terminal) LINE PRINTER (user written) USER (user defined)

The default mode for each logical device is always the teleprinter. Assignments are made using the following format:

EXAMPLE: AC=V(cr)

assigns the console equal to the Crt (video terminal) device. similarly:

EXAMPLE: AR=T(cr)

assigns the reader device to be the teleprinter.

While performing a command which requires a reader input (C,L,R), if the assigned reader is the Teleprinter, the software will look for a character from the TTY input. If a character is not received within a few seconds, it will ABORT, printing an asterisk [*], and return to the command mode. Similarly, if the assigned reader is the Cassette device, and you WISH to abort for some reason, changing the position of any of the SENSE switches will force an ABORT. On the external reader routines, returning with the carry set indicates an abort (or OUT OF DATA) condition.

When assigning a device, only the first letter initial of its name is required.

The Monitor itself is set-up to support the TTY, VIDEO (CRT terminal) and Cassette routines. The other assignments require the addition of user's routines. These are addressed via the commands, which vector to starting addresses.

EXAMPLE: AL=L(cr)

assigns the list device to be the line printer. It vectors to (start address) +812H, or 12H above the end of the monitor. That would be the address for the line printer routine. For details of these arrangements, see the Source Documentation.

Within the above, the assign console equals batch "AC=B(cr)" deserves further mention. In BATCH mode, the READER is made the Keyboard input, and the LIST DEVICE is made the console output. This allows the running of a job directly from the reader input, with the result being output to the LIST DEVICE.

A typical use of this assignment would be the reconstruction of a lengthy text editing job where the text and your editing commands have all been saved on paper tape. With the BATCH MODE, you may assign the reader equals the TTY, the List device equals the TTY, and Console equals BATCH. Running the tape through the reader is the same as you redoing the entire text editing by hand, and the output will go to the TTY and be printed. On a very lengthy job, you could even start the process, and go away until it's done. Its usefulness is limited only by your imagination.

B BYE. This command completely shuts down the system. It is useful where children might have access to the system, where a telephone communications link is established under remote control, or anytime when the operator wishes to make the system inaccessible to unauthorized use.

EXAMPLE: B

completely kills the keyboard. Recovery from the shut-down is accomplished simply by inputting a CONTROL-SHIFT N from the keyboard. (ASCII equivalent is a Record Separator - "RS"; HEX character is a 1EH.) The monitor will sign on and print a greater-than sign (>), however the register storage area will not be cleared.

C COMPARE the reader input with memory. This command is useful for verifying correct loads, verifying that a dumped tape matches with its source etc.

EXAMPLE: C1000,2000(cr,start reader)

compares the memory block 1000H to 2000H with the input from the reader device.

For those with automatic readers, the operation is very simple. Assign the Reader equal to the device you wish to enter the data against, type C(starting address),(ending address)(cr), and the reader will start. The first character read by the reader will be the one matched with the starting address. If any discrepencies are encountered, the reader will stop, and the address (in hex) of the error will be printed on the display. The reader will restart, and continue in this fashion until the entire tape is compared.

If your reader cannot operate automatically, start the reader manually. If an error is encountered, however, while the incorrect address is being printed, the reader will continue, and get "out of sync" with the compare action. Therefore, it is necessary to manually stop the reader if an error is encountered, and manually reposition the tape to the byte following the error. (An excellent article on how to convert ASR33 type readers to automatic operation was recently presented in INTERFACE AGE magazine.)

D DISPLAY memory contents. This command displays the contents of memory in Hex. Memory is displayed 16 bytes per line, with the starting address of the line given as the first piece of data on the line.

EXAMPLE: D100,1FF(cr)

will display in hex the values contained in the memory block 100H to 1FFH.

E END OF FILE. This command generates the end of file pattern for the checksum loader. It is used after punching a block of memory to the punch device using the "W" command. An address parameter for the end of file may be given if so desired.

EXAMPLE: E(cr)

will generate an "end of file marker".

EXAMPLE: E100(cr)

generates the EOF marker with the address parameter "100H". When loading such a file, upon completion, the address contained in the End of File will be placed in the "P" register. Execution of the program may then be initiated by typing "G(cr)".

F FILL command. This command fills a block of memory with a specific value. It is quite handy for initializing a block to a specific value (such as for tests, zeroing memory when starting up, etc.) *NOTE: Avoid doing this over the monitor's stack area. This area may be determined as being between the value you get when typing the Z command, and the value in the S register upon sign-on. It is approximately 60H bytes below the "Top of memory" (Z).

The format for the command is:

EXAMPLE: F100,1FF,FF

fills memory block 100H to 1FFH with the value FFH.

G GOTO command. This command allows the user to cause the processor to GOTO an address and execute the program from that address. In the actual performing of the G command, a program, which has been placed in the stack area during the sign-on of the monitor, is executed. This program will first take all of the values in the register storage area (displayed with the X command), and stuff them in their correct registers in the CPU, and finally JMP to the program address being requested by the operator. If this short program up in the stack has been destroyed (as a result of a "blow-up", or the F or M commands, etc.) the monitor will not be able to GO anywhere, and a manual restart of the monitor will be required. Whenever the monitor is restarted at the initialization point (first address I.E. 0F000H), the contents of the registers are set to ZERO with the exception of the S (stack), which contains a valid stack address. This actual value depends on the amount of memory in the system, etc. In its simplest form, the letter "G" accompanied by a parameter causes the processor to go to that address and start execution.

```
EXAMPLE:      G1000
```

would cause the processor to go to address 1000(H) and execute from that address.

Additionally, one or two breakpoints may be set.

```
EXAMPLE:      G1000,1005,1010
```

would cause the program to start execution at address 1000H, and IN THE EVENT that the program gets to address 1005, OR 1010, the program will stop execution, and return to the monitor, printing an "at" sign, and the address of the breakpoint that was executed. (I.E. @1010) It then prints the ">" prompt, awaiting further instructions. This action also cancels any breakpoints previously set.

Breakpoints must be set at locations containing an instruction byte. This is a SOFTWARE breakpoint system, and requires either RAM at RST 7 (restart 7, addr. 0038H), or if using ROM, a permanent JMP to the monitor TRAP address (0F01EH) at 0038H. Remember, this is a SOFTWARE breakpoint system, and the program being debugged must be in non-protected Read/Write memory.

```
EXAMPLE:      *C2   JNZ   1234H
               34
               12
              *3E   MVI   A,CR
               0D
              *21   LXI   H,1000H
               00
               10
              *77   MOV   M,A
              *23   INX   H
```

```
*CD   CALL  5678H
 78
 56
```

The asterisks (*) mark the bytes that may be used as breakpoints.

H HEX MATH. This command allows the execution of hexidecimal arithmetic directly from the console. it will give the sum and difference of any two hex numbers entered.

```
EXAMPLE:     H1000,1010(cr)
             2010 FFF0
             >
```

2010H being the sum, and FFF0 being the difference of the two hex values.

J The J command is a non-destructive memory test. The command reads any given byte, complements it, writes into the location the complement, compares the complement with the accumulator, and rewrites the original byte into the location. The command is used with two parameters, delineating the block of memory to be checked.

```
EXAMPLE:     J1000,1FFF
```

would perform the above test on the block 1000H to 1FFFH.

If errors are detected, the address at which the error is found and the error are displayed on the console before the test is continued.

```
EXAMPLE:     J1000,1FFF(cr)
             1F00  00001000
             >
```

would indicate that the 4th bit (D3) at location 1F00H did not correctly complement itself.

This test is useful for the discovery of hard memory failures, and also serves as a quick check for accidentally protected memory. A fully protected memory block would print out as entirely "1s". (11111111)

L LOAD BINARY FILE. This command loads a binary file from either a cassette or paper tape.

```
EXAMPLE:     L1000(cr)
```

would load the tape at address 1000H. This would require that the program be an absolute program, designed for address 1000H. The start-of-file mark (automatically generated by the "U" command) is a series of 8 0FFh's (rubouts). When this is detected at the start of file, the bell will ring on the TTY to indicate the start of the load process. When the end-of-file is detected (again, a series of 8 rubouts) the load is terminated, and the address of the NEXT location that would have been loaded is printed on the console. There are two constraints on this type of file system. The middle of the program cannot contain more than 6 0FF's (11111111) in a row (an unusual occurrence), and if 0FFH is the LAST data byte in the file, it will be ignored. This too is unusual, and only a minor inconvenience.
Binary programs loaded at other than their design address will not run. The "L" command does not perform checksum functions, and cannot handle relocatable files. This is a pure and simple byte-for-byte binary loader (see "U" command).

M MOVE COMMAND. This command is used to move a block of memory from one location to another. The original block is NOT affected by the move, remaining intact so long as the block moved into does not overlap with the block currently occupied. This command, like the "F" command should be used with some caution as moving a block into an area occupied by the stack, or the program or the monitor will cause unpredictable results.

EXAMPLE: M1000,1FFF,2000(cr)

moves the contents of memory contained in the block 1000H to 1FFFH to a starting address of 2000H. The new block has the limits 2000H to 2FFFH.

This command is very useful for working on programs without destroying the original, verifying blocks of memory loaded with existing memory, etc.

N NULL. This command punches nulls to the punch device. 72 nulls are punched whenever the command is used. It may be used repetitively for any desired leader length.

EXAMPLE: (N)
*Note: The "N" or "n" will NOT echo, so as to not spoil the paper tape.

It will punch 72 nulls to the punch device.

P PUT ASCII characters into memory. This command allows ASCII characters to be written directly into memory. It is useful for placing labels in files etc.

EXAMPLE: P1000(cr)

activates the command, and any further inputs via the keyboard would be placed into memory in their ASCII equivalent. The command is terminated by a CONTROL D character, with the address of the location following the last entry printed on the console (the Control-D is NOT stored). Recovery of the input data is affected by use of the "T" or "U" command.

Q QUERY INPUT/OUTPUT PORTS. This command allows any value to be output to any I/O port, and allows the value in binary on any I/O port to be read on the console.

EXAMPLE: QO1,7(cr)

would output an ASCII "7" to I/O PORT 1. (ASCII seven is a "bell" so on a TTY, the bell would ring.)

EXAMPLE: QI1(cr) 00001101

inputs the value at port 1, in the illustration above, we see that bits 0,2 and 3 are high, the others low. This is useful for observing the condition of status bits and other diagnostic activities.

R READ A CHECKSUMMED HEX FILE. This command reads checksummed hex files in the INTEL format, as well as being capable of loading the relocatable TDL files at any selected address and bias offset. When reading an ABSOLUTE file (INTEL format), there may be only a BIAS added. These files cannot be relocated. The format is: R[bias],[relocation](cr).

If a checksum error or a failure to write the data to memory occurs, the loading process is stopped, an asterisk is printed (indicating some error condition), and the address that was attempting to be written will be displayed on the console device. This is to assist in determining the failure.

EXAMPLE: R(cr, start reader)

will load a hex file at its absolute address.

EXAMPLE: R,1000(cr,start reader)

will load a TDL relocatable hex file at address 1000H and modify the program to run at address

1000H.

EXAMPLE: R1000,100(cr,start reader)

loads the file set up to run at 100H, but with a positive BIAS of 1000H added to it. Thus, the file, set up to run at 100H will be loaded at 1100H.

EXAMPLE: R1000(cr)

will load the file, set up to run at address 0000H, at address 1000. In other words, using the TDL relocating format, you may load any program, to execute anywhere in memory, anywhere in memory. (Think about it.....)

S SUBSTITUTE and examine. This command allows any address in memory to be examined directly, and allows substitution of one value for another at that address if desired.

```
EXAMPLE:      SF810(sp)00-(sp)1A-(sp)C3-(sp)(cr)
              >
```

In this case the "S" command examines address F810H. The hitting of the space bar (sp) displays the value at that address. (assuming value 00H at that address.) Hitting the space bar again displays the NEXT location in memory (F811H), and so forth. Simply typing S(sp) starts display from address 0000H. By repetitive typing of (sp), all of memory could be displayed one address at a time.

EXAMPLE: SF810(sp)00-(kb)FF(cr)

This command examines address F810H, showing the value 00H at that address. Immediately typing in FFH from the keyboard SUBSTITUTES FFH for 00H at that address. Repeating the example above would show:

EXAMPLE: SF810(sp)FF-

When an address is being examined, the address being examined may be moved BACKWARD by entering a backarrow (ba) or SHIFT-O, or underline, depending on the terminal used.

EXAMPLE: SF810(sp)00-(ba)AA-

shows that at address F80FH, the value AA exists. Typing a space bar will examine F810H again.

T TYPE ASCII characters from memory. This command allows the contents of memory to be displayed in their ASCII equivalents. All non-printing characters will be displayed as periods [.]. It may be used to display the results of the "P" command which allows keyboard entry of ASCII characters directly into memory. Also useful for finding text strings and messages in software. The initial address is first displayed, then the first 64 characters, the next address, etc. until the upper limit has been reached.

```
EXAMPLE:     T1000,2000(cr)
```

displays the ASCII equivalents of memory locations 1000H to 2000H. If the "P" command had been used to place a "message" into memory somewhere in that memory block, it would soon be apparent on the console display.

U UNLOAD BINARY. This command simply dumps core to the punch device. It may be used with a cassette system as well, with no start-up problems. It does not generate a checksum. The format which is generated will be a leader, eight 0FFHs, binary data, eight 0FFHs, and a trailer. The 0FFHs are "rubouts" and are called file ques. These are detected and counted to determine the start and the end of files.

```
EXAMPLE:     U00,FF(cr,start reader)
```

will generate a binary tape, formatted as described above, of the values contained in memory locations 00H to FFH.

V VERIFY. This command allows the user to verify the contents of one memory block against the contents of another memory block. This is very useful for functions such as verifying that a file generated from a program is a duplicate of the actual program, etc.

```
EXAMPLE:     V1000,2000,3000
```

will compare the contents of the memory block 1000H to 2000H against the contents of the memory block commencing at 3000H and extending to 4000H. Any differences will be displayed.

```
EXAMPLE:     V1000,2000,3000
             100F  00 FF
```

indicates that the contents of address 100FH is a 00 while that at 300FH is an FF.

W WRITE Hex file. This command dumps memory to the punch device in the standard "Intel-style" hex file format. Both start and end of file parameters are required. The proper "end of file" (EOF) is generated by the "E" command.

```
EXAMPLE:      W00,FF(cr,start punch)
              (after punching)
              E(cr)
```

will generate a checksummed hex file of the values in the memory block 00H to FFH. If the assigned punch and console are the same, the program will pause and wait for the operator to turn on the punch (ASR33, etc.). Use of the "N" command at either the beginning and/or end of the file is optional, but recommended.

X EXAMINE REGISTERS. The "X" command allows the user to examine and/or modify all of the Z80 registers.

A=Accumulator
B,C,D,E,H,L=CPU REGISTERS
M=Memory (pointed to by H&L)
P=Program Counter (PC)
S=Stack Pointer (SP)
I=Interrupt Register
X=Index (IX)
Y=Index (IY)
R=Refresh Register

```
EXAMPLE:      X(cr)
```

displays the contents of MAIN registers A,B,C,D,E,F,H,L,M,P,S and I, in hex.

```
EXAMPLE:      X'(cr)
```

displays the contents of PRIME registers A,B,C,D,E,F,H,L,M,X,Y and R.

Typing the letter "X" (or X'), followed by a specific register letter will display the contents of that register. Entering a new value via the keyboard (kb) will substitute the new value in the specific register. Hitting the space bar will display the next register in which you may then perform substitutions, etc. In the unique case of the "M" register, you may modify the 16 bit pointer (H&L) to that memory location.

```
EXAMPLE:      XA 00-(kb)FF(cr)
              XA FF-(sp)00-(kb)FF(cr)
              XA FF-(sp)FF-(cr)
              >
```

first examines the contents of register "A" (00H), then substitutes an FF. In the next line, the FF is displayed, a space character displays the next register (again a 00H), and substitutes an FF for this value. The last line displays both registers as containing FFHs.

Y SEARCH. This command allows unique byte strings, from one up to 255 bytes to be searched for in memory, and the addresses where they are found to be displayed. It is advisable to search for unique patterns rather than single bytes. The search operation may be stopped with a control-C.

EXAMPLE:
```
YC3,21,F3,01(cr)
0081
00B2
0F08
>
```

indicates that the byte string (in hex) C3, 21, F3, 01, is found in memory at locations 0081H, 00B2H and 0F08H. This routine will search all 65-K of memory for a unique sequence of bytes in less than one second.

Z Z TOP OF MEMORY. This command locates and gives the highest address of available memory in your system.

EXAMPLE:
```
Z
7FFF
>
```

indicates that the highest available memory is at address 7FFFH. Note that NO carriage return is required. Also, If only one 1K board were in the system, and it was addressed to have its top byte at address 7FFFH, the Z command would so indicate regardless of the absence of lower memory.

ZAPPLE SOURCE DOCUMENTATION

If you are familiar with the 8080 INTEL mnemonics, you have a head start. We at TDL have tried to make the cross-over from the 8080 to the Z-80 as painless as possible, and have used all of the previous OP-CODE mnemonics which were compatible between the 8080 & Z-80. In addition, any obvious extensions were used to simplify learning of the new Z-80 op-codes. For example, just as in the 8080 you have a "LHLD" for "Load H&L Direct", in the Z-80 there is also "LBCD" for "Load B&C Direct", and "LDED" for "Load D&E Direct", etc.

USER WRITTEN COMMAND ROUTINES.

There are 3 command letters left open for your use. They are "I","K", & "O". Both "I" & "O" are naturals for implementing custom I/O routines.
"K" is left for your own imagination. The locations in the command table NOW contain the vector for the ERROR routine. However, in the listing, vectors to the 0F800H block are given, and should be patched to those vectors as the commands are implemented. Then, JMPs to the ACTUAL routines should be placed in the 0F800H portion. At the conclusion of the CUSTOM COMMAND, a RET instruction will return to the normal monitor command loop, printing the ">" prompt.

USER WRITTEN I/O ROUTINES.

There are occasions when some device needs a specialized piece of software in order to make it work. Line printers, parallel keyboards, punches, optical readers, etc. These will have to be handled on an individual basis. The general idea is to NOT MODIFY any registers other than those mentioned above, and to NOT upset the stack pointer. Things may be pushed during the routine in order to avoid modifying the other registers, as long as the POP's match the PUSH's. All routines that are vectored out of the monitor should end with a RET instruction. Remember to clear the carry before returning from a USER defined "RI" routine, unless you are intending to indicate an OUT-OF-DATA condition. In that case, you SHOULD set the carry flag before returning (STC).

Using MEMORY as a Reader/Punch device can also be very useful. Here is an example of how this might be accomplished:

```
MEMRD:      PUSH  H     ;FIRST SAVE H&L
            LHLD  01EH  ;PICK UP A POINTER
            MOV   A,M   ;GET MEMORY BYTE
            INX   H
            SHLD  01EH  ;REPLACE POINTER
            POP   H     ;RESTORE H&L
            ORA   A     ;INSURE CARRY CLEAR
            RET         ;ALL DONE


MEMWR:      PUSH  H     ;SAVE H&L
            LHLD  01CH  ;OUTPUT POINTER
            MOV   M,C   ;STORE OUTPUT BYTE
            INX   H
            SHLD  01CH  ;REPLACE POINTER
            POP   H     ;RESTORE H&L
            MOV   A,C   ;FOLLOW THE RULES
            RET         ;ALL DONE
```

3.3 RAM STORAGE

The fast access static RAM memory is comprised of EMM SEMI 4804, 1K x 4 bit devices. These are located from F800H to FFFFH. The "top" .25K bytes of this memory will be used by the operating system as stack area if no other RAM is present in the system. The "lower" 1.75K bytes, beginning at F800H, would then be available for operating system extensions, display drivers or other short programs. If additional RAM is available on the bus, the entire 2K block, beginning at F800H would be available for user extensions.

MEMORY PROTECT- This memory can be write-protected by means of the dip switch in the upper left-hand corner of the board. Switches one and two can be toggled on to protect the 1K blocks individually. Block F800H to FBFFH is protected by S1. Block FC00H to FFFFH is protected by S2. When the switches are closed, the appropriate block is protected. The switch is closed by pressing in the side of the 4-pole mini-dip switch which is labelled "+" and "on."

3.4 CASSETTE INTERFACE

A good quality cassette recorder should be used with this interface. Most tape recorders selling for $ 69.95 and above should be excellent choices. Due to the wide variety of recorders available and variation from recorder to recorder it is difficult to recommend a particular brand. However, the above price range should be of some help. Although less expensive recorders such as the General Electric (No. 3-5105A) at $ 39.95 and others have been used successfully at TDL, it is observed that they are harder to adjust and operate and some of them are extremely difficult, if not impossible, to get running reliably.

The volume control is the most critical adjustment to make. If the unit has a tone control, it should be set at the extreme "treble" setting. It is best to begin by making a recording of the Zapple Monitor contained in ROM and then playing it back and verifying it against the ROM. Begin by plugging the audio cable coming from "R" on the SMB to the "RECORD" jack of the recorder and the cable from "P" to the "SPEAKER" or "EARPHONE" jack of the recorder. The "RECORD" jack may be labelled either "MIC" or "AUX". Some recorders have both. See the section (2.4) on setting of S3, the MIC/AUX switch, and make sure it is in the correct position.

To copy the monitor, the following command should be used:

```
WF000 F7FF (return)
```

The tape recorder should be started well in advance of hitting return and a few nulls (N <return>) may be inserted before actually recording. When the monitor comes back with the ">" prompt, enter the following:

```
E (return)
```

This latter step is very important as it signals the end of the file. What has just been recorded is a checksummed hex file of the Zapple Monitor.

Rewind the tape to the starting position and note the volume control setting. The following command should be used on playback:

```
R1000 (return)
```

Start the recorder before hitting return so the speed has a chance to become stable. When the playback is finished, the monitor will return with the ">" prompt. At this time, if the volume setting was adequate, there will be a copy of the monitor at 0000H to 07FFH. Make sure there is RAM at that location. Test for accuracy of the copy by using the verify command:

```
V 0 F7FF F000 (return)
```

If everything is OK, the ">" prompt will return. If not, the addresses that do not match will be printed followed by the hex representation of the copy and then the hex of the monitor's byte.

Before changing the volume setting and re-recording, switch S4 to its other state (invert). If that isn't successful, re-record at different volume settings until the proper one is found. Don't forget to try playback in both the inverted and non-inverted states of S4.

Some units, especially the less expensive ones are very poorly isolated internally and will pick up the continuous tone put out by the interface's record circuitry and feed it back on playback. If trouble is experienced, try unpluging the jack to the recorder while playing back.

3.5 PARALLEL PORT

U26 on the System Monitor Board is a Motorola or equivalent 6820 PIA that contains two parallel I/O ports. One of these is used by the operating system to specify the I/O device currently being used.

The other parallel port is available to the user and can be configured as either an input port or an output port. As an input port, a keyboard, high speed paper tape reader, or other device can be used. As an output port, a high speed paper tape punch, line printer, etc. can be used. In order to use it, however, it must be set up with the proper software.

PORT ASSIGNMENTS

The devices are assigned to ports on the System Monitor Card in the following manner:

DEVICE	STATUS/CONTROL	DATA
Teletype (serial)	70	71
Video (serial)	72	73
Cassette	74	75
	76*	77*
Parallel Port	79	78
Sense Switch	7A	7B
Unused	7C,7D,7E,7F	

* Note: Ports 76 & 77 are used internally for operation of the monitor.

The Parallel Port is implemented by means of a Motorola MC6820 Peripheral Interface Adapter (PIA) chip. The user who intends to realize all of the capabilities of this device will find it helpful to study the Motorola data sheets.

3.6 LED

The LED is mounted by IC-U29 (1488). It should be mounted with the flat side toward the top of the board. This will cause the LED to be normally lit. It may be controlled by resetting (or setting) bit 6 on I/O port 74. The LED may be reversed (flat side down) to cause it to be normally off.

The LED is essentially a one-bit programmed output port. For example, it may be used in conjunction with the Cassette Port to indicate reception of sync characters. What follows is a short program to implement this feature:

```
TEST:      CALL    0F006H    ;GET READER CHAR
           PUSH    PSW       ;SAVE FLAGS & ACC
           CPI     06EH      ;TEST FOR UNIQUE CHAR
           MVI     A,10H     ;SET TO TURN OFF
           INZ     NOT
           MVI     A,50H     ;SET TO TURN ON
```

```
NOT:        OUT    074H       ;DO IT
            POP    PSW        ;RESTORE ACC
            RET  (OR JMP TEST)  ;DONE
```

Note: "TEST" would be called by a user program, or could just sit in a loop. This would allow the testing or adjusting of the Cassette Recorder. I.E. Record a section of tape with all "06EH" characters, and then play it back, making sure the LED does not flicker, etc.