5.3 SYM-1 MONITOR COMMANDS

5.3.1 M (Display and/or Modify Memory)

Number of Associated Parameters			
0	1	2	3
Memory Examine and modify, begin at (OLD)	Memory Examine and modify, begin at (1)	Memory Search for for byte (1), in locations (OLD)-(2)	Memory Search for byte (1), in locations (2)-(3)

- The standard form for this command uses **one parameter** and is shown below.

M addr **CR**

SUPERMON will then display the address and the byte contained in the location "addr." The following options are then available:

1. Enter 2 Hex digits: bb is replaced and the next address and byte are displayed.
2. Enter single quote (from terminal) and any character: bb is replaced with the ASCII code for the entered character.
3. Enter →or←(>or< from terminal): bb is left unchanged and addr+1 or addr-1, with its contents, is displayed.
4. Enter + or - : bb is left unchanged and addr+8 or addr-8 with its contents, is displayed.
5. Enter **CR** : Return to monitor command mode; bb unchanged.

- Another form of the display memory command uses **no parameter** as shown below:

M CR

This will cause SYM-1 to resume memory examine and modify at (OLD).

- The same memory (M) key may be used to search for a particular byte in memory, using **three parameters** in this form:

M bb,addr1,addr2 **CR**

This instructs the system to search for byte bb from addr1 to addr2. When an occurrence of bb is found, the location and contents are displayed, and all of the standard **M** options described above become available. In addition, a "G" entered following any halt will continue the search.

- Similarly, the **two parameter** sequence:

M bb,addr CR

will resume memory search for byte bb from (OLD) to addr.

The following examples demonstrate the various uses of memory display/modify commands. Characters entered by the user are underlined.

One Parameter

```
.M ↵                    Display memory location (OLD); return to Monitor
0215 BB ↵
.
```

```
.M A656↵                Display memory location A656
A656 00 0A              Put some data there; return to Monitor
A657 4D ↵
.
```

```
.M 200↵                 Display memory location 200
0200 20 'A              Replace data with ASCII code for A
0201 86 'B              Next location displayed; replace data with ASCII B
0202 8B 'C              Next location displayed; replace data with ASCII C
0203 20 ↵               Return to Monitor
.
```

```
.M 0200↵                Display memory location 200
0200 41 >               Display next location; data unchanged
0201 42 >               Display next location; data unchanged
0202 43 _               Use space bar for same purpose as arrow
0203 20 _
0204 AF ↵               Return to Monitor
.
```

```
.M 0300 ↵               Display memory location 300
0300 B4 <               Display previous location; data unchanged
02FF BB <
02FE 44 <
02FD BB ↵               Return to Monitor
.
```

```
.M 0200 ↵               Display memory location 200
0200 41 +               Advance 8 bytes and display memory
0208 F0 _               Space used to advance one location; data unchanged
0209 06 _
020A 20 -               Reverse 8 bytes and display memory
0202 43 ↵               Return to Monitor
.
```

```
.M 0200↵                Display memory location 200
0200 41 ↵               Return to Monitor
.M↵                     Display (OLD) which is still 200
0200 41 ↵               Return to Monitor
.
```

```
.M 6C,8000,8400↓                    Search for 6C in range 8000-8400
801F,6C,-
8017,29,↓
.U↓
8017 29 10 F0 07 68 AA 68 28,02
 0202
.M 6C,8400↓                         Continue search
801F,6C,_
8020,F6,_
8021,FF,G                           Continue search
8026,6C,_
8027,F8,_
8028,FF,↓                           Halt search
.
```

5.3.2 R (Display and/or Modify User Registers)

Number of Associated Parameters			
0	1	2	3
Examine and modify user registers PC,S,F,A,X,Y			

- The only pre-defined form of this command is with **no parameters**, i.e.:

R CR

As soon as the command is entered, the contents of the PC are displayed as follows:

P 8B4A,

Using a forward arrow (→ or >), you may examine the next register. Registers are displayed in the order PC, S, F, A, X, Y, with wrap-around (i.e., PC is displayed after Y). Each register except PC carries a Register Number on the display or TTY printout; S is R1, F is R2, A is R3, X is R4, and Y is R5 (see example below).

To modify the displayed register, enter two or four digits (four only in the case of the PC). The register will be automatically modified and the next will be displayed. A **CR** will cause control to return to the monitor for another command.

In the following example, we have modified the contents of the PC register to become 0200, and the A register to be set to 16. The other registers are not modified and at the conclusion of the complete register cycle and redisplay of PC, a **CR** is used to return to monitor command mode.

```
.R↓
P 8B4A,__                Display registers
R1  FF,__                PC; space is used to advance
R2  00,__                S
R3  00,__                F
R4  00,__                A
R5  00,__                X
P 8B4A,↓                 Y
.                        PC re-displayed; return to Monitor

.R↓
P 8B4A,0200
R1  FF,>
R2  00,_
R3  00,16                Alter PC = 200, A = 16
R4  00,↓
.
```

5.3.3 G (GO)

Number of Associated Parameters			
0	1	2	3
Restore all user registers and resume execution at PC	Restore user registers except PC = (1) S = FD; monitor return address is pushed onto stack		

- The GO command may be used with **no parameters** to restore all user registers and begin execution at PC:

G CR

- With **one parameter**, the command will restore user registers except that PC is set to addr, S is set to FD and SUPERMON's return address is pushed onto the stack. Thus, if a subroutine return is executed, it will result in a return to monitor command mode (with the user's stack not saved). Its format is as follows:

G addr **CR**

5.3.4 V (VERIFY)

Number of Associated Parameters			
0	1	2	3
Display 8 bytes with checksum beginning at (OLD)	Display 8 bytes with checksum beginning at (1)	Display (1)-(2), 8 bytes per line, with addresses and cumulative checksums	

- With **one parameter**, this command will result in the display of 8 bytes beginning at addr, with checksum. The format is as follows:

V addr CR

In this example, bytes stored in locations 200-207 are displayed, along with their checksum:

```
.V 200↓
0200 41 42 43 20 AF 88 C9 0D,F3
 02F3
.
```

Note that on the on-board display, only the two-byte checksum will be visible.

The checksum is a 16-bit arithmetic sum of all of the data bytes displayed. The low byte is displayed on the data line, and the full checksum on the next. The address is not included in the checksum.

- With **no parameters**, the command will display 8 bytes beginning at (OLD).

V CR

```
.V↓
0200 41 42 43 20 AF 88 C9 0D,F3
 02F3
.
```

- With **two parameters**, the "V" command will display memory from addr1 through addr2. Eight bytes per line are displayed, with cumulative checksums. A single byte checksum is included on each data line, and a final two-byte checksum is printed on a new line.

V addr1,addr2 CR

```
.V 8000,8015↓
8000 4C 7C 8B 20 FF 80 20 4A,5C
8008 81 20 71 81 4C 03 80 08,C6
8010 48 8A 48 BA BD 04,5B
 085B
.
```

5.3.5 D (Deposit)

Number of Associated Parameters			
0	1	2	3
Deposit to memory, beginning at (OLD), CRLF/address after 8 bytes, auto spacing	Deposit to memory, beginning at (1)		

- This command is used for entering data to memory from a terminal. With **one parameter**, this command instructs the system to output a **CR** and line feed and print addr. As each two-digit byte is entered, a space is output. If you enter a space (instead of a two-digit byte), you will cause two more spaces to be output, and that memory location will remain unchanged.

D addr CR

```
.D 200↓
0200 A9 3A 85 46 20 13 08 20
0208 EE 08 85 44 84 45 C6 46
0210 D0 F2 60 ↓
.
```

- As with other commands, the "D" with **no parameters** will deposit beginning at (OLD).

D CR

Notice that V and D line up, so that a line displayed with V may be altered with D, as shown below:

```
.V 200↓                                  Verify contents of 0200-0207
0200 A9 3A 85 46 20 13 08 20-09
 0209                                    Checksum
.D↓
0200 _ 0D _ 45 _ 80 03 ↓                 Deposit memory from 0200; space to advance
.V 200↓                                  Re-verify contents of 0200-0207
0200 A9 0D 85 45 20 80 03 20-43
 0243                                    New checksum
.
```

5.3.6 C (Calculate)

Number of Associated Parameters			
0	1	2	3
	Calculate 0-(1), the two's complement of (1)	Calculate (1)-(2) or displacement	Calculate (1)+(2)-(3) or displacement with offset

This command is used to do Hexadecimal arithmetic. It is very useful in programming to compute branch operands required for SY6502 instructions.

- With **one parameter**, it calculates 0 minus addr (i.e., the two's complement).

C addr **CR**

- With **two parameters**, the "C" command will calculate addr1 minus addr2 (i.e., displacement).

C addr1, addr2 **CR**

- With **three parameters**, the "C" command will calculate addr1 plus addr2 minus addr3 (i.e., displacement with offset).

C addr1,addr2,addr3 **CR**

5.3.7 B (Block Move in Memory)

Number of Associated Parameters			
0	1	2	3
			Move all of (2) thru (3) to (1) thru (1)+(3)-(2)

- This command is only defined for **three parameters** and is demonstrated by the following examples:

```
.B 200,300,320↓
.
```

Move 300 thru 320 to 200 thru 220.

```
.B 200,220,250↓
.
```

Move 220 thru 250 to 200 thru 230. No data is lost, even though the regions overlap.

```
.B 220,200,230↓
.
```

Move 230 thru 200 to 250 thru 220. (Note that this move occurs in the opposite direction. No data is lost.)

5.3.8 J (JUMP)

Number of Associated Parameters			
0	1	2	3
	Restore user registers except PC=entry (1) of JUMP TABLE, S=FD, monitor return pushed on stack		

- This command is only defined for **one parameter.**

J n CR

The parameter, n, must be in the range 0-7. All user registers are restored, except PC is taken from the JUMP TABLE in System RAM, and S=FD. The monitor return address is pushed onto the stack.

(Because the monitor return is on the stack, a JUMP to a subroutine is allowable.)

Note also that certain useful default addresses are inserted in the JUMP TABLE at Reset. (See Memory Map.)

5.3.9 SD (Store Double Byte)

Number of Associated Parameters			
0	1	2	3
		Store high byte of (1) in (2)+1 then low byte of (1) in (2). Good for changing vectors	

- This command is defined only for **two parameters** and is most useful for changing vectors.

SD addr1,addr2 CR

The example below was used to enter the address of the Hex keyboard input routine into INVEC, in correct order (low byte-high byte). Note that this vector could not have been altered with M, because after one byte had been altered, the vector would have pointed to an invalid address.

```
.SD 89BE,A661↓
.
```

5.3.10 F (Fill)

Number of Associated Parameters			
0	1	2	3
			Fill all of (2)-(3) with data byte (1)

- Defined only for **three parameters**, this command will fill the defined region of memory (addr1-addr2) with a specified byte (bb).

F bb,addr1,addr2 **CR**

For example:

```
.F EA,200,300↓
.
```

Fill the region 200 thru 300 with the byte EA, which is a NOP instruction.

5.3.11 W (Write Protect)

Number of Associated Parameters			
0	1	2	3
	Write protect user RAM according to 3 digits of (1)		

- This command is defined for only **one parameter.** To unprotect all of user RAM, the command is:

W 0 CR

Its general form is:

W $d_1d_2d_3$ CR

Where each of d_1, d_2, d_3 are the digits 0 (unprotect) or 1 (protect).

d_1	= 400-7FF	1K above first K of RAM
d_2	= 800-BFF	2K above first K of RAM
d_3	= C00-FFF	3K above first K of RAM

For example

```
*W 101↓
*
```

1 protect 400-7FF
0 unprotect 800-BFF
1 protect C00-FFF

Note that write protect applies to extended user RAM on-board, and also that it requires a jumper insertion (see Chapter 4).

5.3.12 E (Execute)

Number of Associated Parameters			
0	1	2	3
	Get monitor input from RAM, starting at (1)	Get monitor input from RAM, starting at (2) and store (1) for later use at A64C	Get monitor input from RAM, starting at (3) and store (1) and (2) for later user at A64E and A64C

- The standard form of the execute command uses **one parameter**.

E addr CR

SUPERMON adjusts its INPUT vectors to receive its input from RAM, beginning at addr. It is assumed that the user has entered a string of ASCII codes into RAM locations beginning at addr, terminated by a byte containing 00. When 00 is encountered, input vectors will be restored. The easiest way to enter these codes is to use the M command with the single-quote option (Section 5.3.1).

When E is used with **two** or **three parameters**, the additional parameters will be stored in system RAM at A64C and A64E. It is the user's responsibility to interpret them. (Note that the E command is vectored; see Chapter 9.)

```
.M 300↓
0300 'C 'F 'F 'F 'E ', 00 ↓
.
```

The sequence at 300 is part of a commonly used Calculate routine.

```
.E 300↓
.C FFFE,200,280↓
 FF7E
.
```

Notice that part of this C command came from RAM, and part was entered at the terminal.

5.4 CASSETTE AND PAPER TAPE COMMANDS

The SYM-1 handles cassette I/O in two formats, KIM-compatible format (8 bytes/sec), and SYM high-speed format (185 bytes/sec).

The S1 and L1 commands refer to KIM format, while the S2 and L2 commands refer to SYM high-speed format.

With each Save command you specify a two-digit ID, as well as starting and ending addresses. The ID, the addresses, and the contents of all memory locations from starting to ending address, inclusive, will be written to tape. Each Save command will create one **RECORD.**

You should be careful to assign unique ID's to different records on the same tape, and to label the tape with the ID's and addresses of all the records it contains.

While SYM is searching for a record or trying to synchronize to the tape, an "S" will be lit in the left-most digit of the display on the on-board keyboard. If the "S" does not turn off, SYM is unable to locate or to read the requested record.

5.4.1 S1, S2 (Save Cassette Tape)

Number of Associated Parameters			
0	1	2	3
(S1)			Save cassette tape, locations (2) - (3) with ID = (1) in KIM format
(S2)			Save cassette tape, locations (2) - (3) with ID = (1) in High Speed format

- These commands are discussed together, as their syntax is identical. Recall that S1 refers to KIM format while S2 refers to SYM high-speed format.

Both are defined only for **three parameters.**

S2 bb,addr$_s$,addr$_e$ **CR**

The first parameter is a 2-digit ID, which may be any value other than 00 or FF. It is followed by the starting address and the ending address. In the example below, all memory locations from 0200 thru 0280, inclusive are written to tape, and given the ID 05.

```
.S1 5,200,280
.
```

5.4.2 L2 (Load High-Speed Format Record)

Number of Associated Parameters			
0	1	2	3
Load first Hi Speed record found into locations from which it was saved	Load Hi Speed record with ID = (1)		(1) must = FF. Load first Hi Speed record found into (2) - (3)

- The standard form of this command uses **one parameter**, as follows:

L2 bb CR

The parameter bb is the ID of the record to be loaded. When found, the record will be loaded into memory, using the addresses saved in the record itself.

If the record bb is not the first high-speed record on the tape, the "S" light will go out as VIM reads through, but ignores, the preceding records. After each unselected record is read, the "S" will re-display.

- With **no parameters** (or a single parameter of zero), the instruction will load the first high-speed format record found, without regard to its ID, using the addresses saved in the record itself.

L2 CR

or

L2 0 CR

- The L2 command exists in a third form, using **three parameters**, as follows:

L2 FF,addr1,addr2, CR

This usage will load a record into a **different** area of memory from where it was saved. The first parameter **must** be FF, followed by the requested starting and ending address. It is your responsibility to supply addr1 and addr2 such that their difference is the same as the difference of the addresses used to save the record.

5.4.3 L1 (Load KIM Format Record From Tape)

Number of Associated Parameters			
0	1	2	3
Load first KIM format record found into locations from which it was saved	Load Kim record with ID = (1) into locations from which it was saved	(1) must = FF. Load first KIM record found, but start at location (2)	

- The L1 command, used with **zero or** with **one parameter**, is identical in syntax to the L2 command (see Section 5.4.2, above).
- With **two parameters**, the L1 command is used to load into a **different region of the memory than that with which the record was saved.**

L1 FF,addr **CR**

The first parameter **must** be FF, followed by the requested starting address. No ending address is necessary, as the load operation will halt when the end of the record is found.

5.4.4 SP (Save Paper Tape)

Number of Associated Parameters			
0	1	2	3
		Save data from locations (1) - (2) in paper tape format. To create end of file record, unlock punch, switch to local mode, lock punch, type ;00 **CR**	

- Defined only for **two parameters**, this command will output data from RAM in paper tape format (see Appendix D).

SP addr1,addr2 **CR**

For example:

```
.SP 200.215↓
;10020034AB743B44BB44BB44BB44BB44BB44BB079A
;060210AC1BF49BD4BB03FD
.
```

5.4.5 LP (Load Paper Tape)

Number of Associated Parameters			
0	1	2	3
Load data in paper tape in format. To signal end of file for tape without EOF record, type ;00 **CR** in on-line mode			

- This command is defined for **no parameters** only. It will load memory with data in paper tape format (see Appendix D).

LP CR

5.5 USER-DEFINED FUNCTIONS

You may, as we have previously pointed out, write programs to be called from the on-board keyboard. You may do this by using any combination of command and number of parameters which is not already defined (e.g., B MOV with only two parameters) or by using any or all of the eight keys along the bottom two rows of the on-board keyboard (those labeled "USR 0" through "USR 7"). The exact means of implementing these special functions is discussed in detail in Chapter 9.

5.6 ERROR CODES

The SYM-1 microcomputer system handles error codes in an interactive way, with codes being designed to be determined by the context in which the error occurs. No table of error conditions and their meanings is therefore provided with this manual, since these are context dependent.

However, you should be aware of the general method by which errors are handled by your SYM-1 system.

When your SUPERMON encounters an error of some type, it displays a 2-digit representation of the byte which was being processed when the error was detected. For example, if you attempt to carry out a CALC command with no parameters (and you haven't defined such a routine yourself as explained in Chapter 9), the system will display a "43." which is the ASCII representation for the "C" which represents the CALC function.

Similarly, if you attempt to use an ID of 00 or FF with either SAV1 or SAV2, the system will display the ID used in error.

After the "er" message is printed, a new prompt (decimal point) is displayed, and SUPERMON waits for a new command. Note that you do not need to RESET when an error condition occurs, since that results in System RAM being cleared and necessitates a re-start of your routine. It is also worth noting that when you carry out an EXEC command at the on-board keyboard the system does not halt when an error occurs; rather, it continues in the same fashion as if new commands were coming directly from the keyboard. The error condition therefore flashes too rapidly on the LED display for you to see it. Command sequences to be executed by EXEC should be pretested prior to such use.

Some fixed error codes do exist in the monitor. Four such codes are used in audio cassette operations and are defined in Table 5-3. Additionally, if in carrying out LD P, FILL or B MOV commands you either attempt to store data in a non-existent or WRITE-protected memory location or if during execution of one of these commands a memory error occurs, the LED display will show the number of locations read incorrectly. This number will always stop at "FF" if it exceeds that number, so that the display will have some intelligible meaning.

Table 5-3. ERROR CODES IN AUDIO CASSETTE OPERATIONS

Code Displayed	Meaning
2F	Last-character error. The last character in a tape record should be a 2F. If that is not the case, the system displays the error code shown.
CC	Checksum error. Usually indicates data transfer problems. Re-position the tape and try again.
FF	In KIM-1 format loading, this error code means a non-Hexadecimal character has been encountered. This almost always means a synchronization error. Restart the procedure.
	In High-Speed format loading, a framing (i.e., synchronization) error is the cause. Restart the procedure.

The following examples provide some representative errors to enable you to become familiar with how they are reported on SYM-1 using a TTY or CRT.

```
.W 111↵
.F EA,300,400↵
ER 01
.
```

Memory location 400 write protected, therefore it could not be modified. One byte only in error.

```
.S2 200,280↵
ER 1E
.
```

S2 is not defined for two parameters. The hash code for S2 is 1E.

```
.C A,230,500,
ER 2C
.
```

Three parameters only permitted.

```
.C 200,2X
ER 58
```

X is not a valid Hex digit.

```
.S2 FF,200,280↵
ER FF
.
```

ID may not be FF or 00.

```
.L2 AA,200,280↵
ER AA
.
```

ID must be FF.

```
.M 6000↵
6000,60,F5?
6001,60,↵
.
```

No RAM at 6000, therefore it cannot be modified.

```
.D 8000↵
8000 AA? DD? ↵
.
```

ROM at 8000, therefore it cannot be modified

```
.D 200,280↵
ER 44
.
```

Deposit not defined for 2 parameters. D = ASCII 44.

```
.F EA,5000,6000↵
ER FF
.
```

No RAM at locations 5000-6000, therefore no modification was possible. The number of bytes which were not correctly changed is greater than or equal to 255 (decimal).