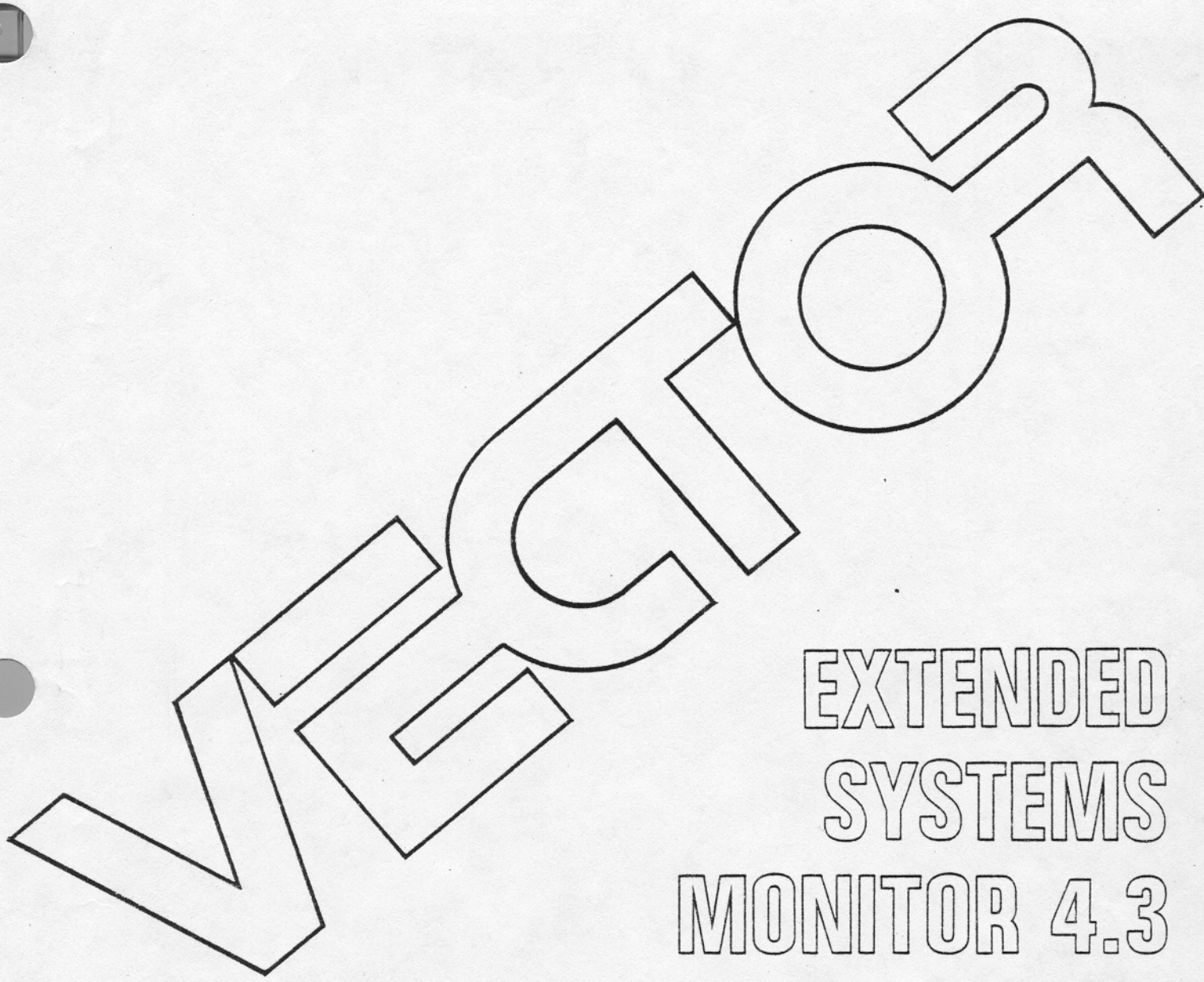

EXTENDED SYSTEMS MONITOR 4.3

User's Manual

EXTENDED SYSTEMS MONITOR

Version 4.3

USERS MANUAL

Revision A

July 10, 1981

P/N 7100-0245-00-00

Disclaimer

Vector Graphic makes no representations or warranties with respect to the contents of this manual itself, even if the product it describes is covered by a warranty or repair agreement. Further, Vector Graphic reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Vector Graphic to notify any person of such revision or changes, except when an agreement to the contrary exists.

Revision Numbers

The date and revision of each page herein appears at the bottom of each page. The revision letter such as A or B changes if the manual has been improved but the product itself has not been significantly modified. The date and revision on the Title Page corresponds to that of the page most recently revised. When the product itself is modified significantly, the product will get a new revision number, as shown on the manual's title page, and the manual will revert to revision A, as if it were treating a brand new product. THIS MANUAL SHOULD ONLY BE USED WITH THE PRODUCT(S) IDENTIFIED ON THE TITLE PAGE.

TABLE OF CONTENTS

GENERAL DESCRIPTION

The Version 4.3 Monitor is a complete systems Monitor, able to support the Flashwriter II (80 X 24) board, and the Vector Graphic Keyboard. Thus it is recommended for use with the Mindless Terminal. All keyboard and video I/O can be done through the Monitor's I/O routines, freeing higher level software from carrying a variety of versions for different hardware configurations. Version 4.3 was designed to be used with the Flashwriter II board. Use Version 4.0C for serial terminals.

Version 4.3 differs from 4.2 in that the serial port initialization routine has been slowed down to accomodate Vector systems using 6 MHz. ZCB boards. 4 MHz. ZCB boards are also appropriate with this Monitor program.

In addition to I/O, the Monitor includes an extensive command executive, a compactly written program designed to facilitate manipulation and display of memory data. The "prompt" which indicates that the Monitor Executive is waiting for operator entry is "Mon>".

There are 26 commands which are entered as a single letter followed by up to four hexadecimal data fields. After each field is entered, a space is automatically output as a prompt. Either upper or lower case alpha characters may be used, but lower case characters will be converted to upper case, and any non-hex characters will be ignored. Allowable hex characters are 0-9, A-F. Address fields are four digits long; other fields are two digits long. The executive is useful in debugging hardware and software, particularly assembly language software, because it is resident in the system.

If a space is typed at any time during field entry, a default value of zero is assumed for all leading zeroes. This applies to an entire field as well as one that has been partially entered, and the cursor will advance to the next field if required. For example, typing (SP) will have the same effect as typing 0000; typing 100(SP) will have the same effect as 0100.

Any command that generates a display can be temporarily halted with a space and continued with another space. The ESCape key will abort a display or command entry.

The 4.3 Monitor is located at address E000H - E7FFH in Vector Graphic systems.

The hexadecimal number system may seem confusing if you are not familiar with it, but it has become the standard of the microcomputer field and is clearly the best system with 16 bit addresses and 8 bit data. It is usually not necessary to convert between number systems, as this is usually done by software (i.e. assemblers). Remembering a few values in hex should make things easy:

HEX NUMBER	DECIMAL VALUE	JARGON	BINARY BITS
A	10		4
B	11		4
C	12		4
D	13		4
E	14		4
F	15		4
10	16		5
FF	255		8
100	256	1 PAGE	9
3FF	1,023		10
400	1,024	1K	11
FFF	4,095		12
1000	4,096	4K	13
4000	16,384	16K	15
8000	32,768	32K	16
FFFF	65,535	64K-1	16

The familiar rules of arithmetic work just the same in hex as in decimal:

```
    10          Hex (trivial)
40) 400
```

COMMAND FORMAT

Mon>A <ADR1> <ADR2> - ASCII DUMP

Memory contents from ADR1 through ADR2 will be displayed as ASCII characters, or graphic symbols for values less than 20 hex. If the most significant bit is high, reverse video is displayed. This command is useful for examining files such as those created by SCOPE, BASIC or MEMORITE. ASCII strings embedded in object code are easy to recognize.

Mon>B - BOOT FLOPPY

Typing this command causes a jump to location E80CH which is located on the disk boot PROM. This will cause the disk operating system to be loaded into memory and transfer control to CP/M. This is designed to be used with a Vector system using the DualMode controller board. If a Micropolis Disk Controller board is present in the system, it may be accessed by typing G F800 in response to the "Mon>" prompt.

Mon>C <ADR1> <ADR2> <ADR3> - COMPARE BLOCKS

A byte-by-byte comparison will be made between the block of memory data starting at ADR1 and ending at ADR2 and a block of identical length starting at ADR3. The differences will be printed out with the address, the byte in the first block and the byte in the second block. This command is useful to compare two versions of a program or to verify that proms have been programmed correctly.

Mon>D <ADR1> <ADR2> - DUMP IN HEX

Memory contents from ADR1 through ADR2 will be displayed as pairs of hexadecimal characters. The left character in each pair represents the four most significant bits of the memory location. The display may be halted and interrupted as described above. The ASCII representation is displayed in a column on the right.

Mon>E - EXTERNAL COMMUNICATIONS

The monitor will output anything typed on the keyboard through port 4 on the ZCB single board computer, the Bitstreamer II I/O board or an appropriately addressed Bitstreamer I board. Anything received on this port will be displayed on the screen. Normally a 300 baud modem would be connected to the serial RS 232 output from the I/O board, and this feature allows the system to be used as a simple terminal to communicate with a host in a full duplex mode. Operation at speeds above 300 baud requires the host to send null characters after linefeeds, so that characters are not lost when the screen scrolls up.

Mon>F <ADR1> <ADR2> <BYTE1> <BYTE2> - FIND TWO BYTES

This memory range from ADR1 through ADR2 will be searched for the particular code combination BYTE 1 BYTE 2. This is useful for locating particular commands or jump addresses. For example, if you wish to change a control character (say control D) in a program you may try FE 04, which is CPI 04 since this is a common way of testing input characters. If you wish to find all locations that call or jump to a particular address, say C700H, then search for 00C7. There is no guarantee that each location displayed is valid object code - it may be part of a data table, ASCII string, or second and third bytes of a three byte instruction.

Mon>G <ADR1> - GO TO AND EXECUTE

This command will cause a jump to ADR1 to execute a program or user subroutine. As with all Monitor jump commands, the address contained on the stack is "START" (E04CH) and if the user routine at ADR1 ends in "RET", program execution will return to the Monitor. Approximately 96 levels of stack space is available, but of course, pushing more registers on the stack than are popped will defeat the return feature with undesirable effects.

Mon>H - JUMP TO HI RAM

This command jumps to FC00H which is the start of the 1K scratchpad RAM. This is a useful area for small machine language programs.

Mon>I <PORT> - INPUT FROM A PORT

Execution of this command will cause the CPU to execute an "IN PORT" instruction and the accumulator contents immediately following this to be displayed. This command is useful in checking out peripheral equipment. Only those ports used by the terminal, cassette interface, etc., will contain interesting values. All others will read FF since the data bus will be floating when the "IN" command is executed.

Mon>J - JUMP TO LOADED DOS

This command permits easy return to the MDOS disk operating system at 04E7H, or if not present, jump will be 0000H, which is the CP/M warm start location.

Mon>K - SET BREAKPOINTS

This command expects a 4 digit address, and will place a RESTART 7 (FF) at that location in RAM. When that instruction is executed, which is a call to location 0038H, the CPU will jump to the monitor routine that dumps the register contents. The instruction replaced with FF will also be restored. If a program is loaded over 0038H, the breakpoint instruction will be defeated unless RESET is depressed. Entry of the monitor at E000H will clear the breakpoint, as will pressing the RESET switch.

Mon>L - JUMP TO LOW RAM AT 0000H

This command jumps to memory location 0000H which is the beginning of program memory. This is the CP/M warm start location.

Mon>M <ADR1> <ADR2> <ADR3> - MOVE MEMORY BLOCK

The data contained in memory starting at ADR1 and ending at ADR2 is moved to memory locations starting at ADR3. This command is useful for moving a program from a temporary storage location to its correct address. If there is an overlap of the two memory areas, interesting results are obtained. For example, M 6000 7BFF 6400 will cause the block of data from 6000H through 63FFH to be repeated 8 times from 6000H through 7FFFH, since by the time location 6400H is read, it has been overwritten with data from 6000H. This is useful for bank programming of proms, or for creating repeating instruction sequences for test purposes.

Mon>N - NON-DESTRUCTIVE MEMORY TEST

Memory locations starting at 0000H are read and the data temporarily stored. The memory location is then tested to see if 00 and FF can be written and read correctly. This continues after rewriting the original data until the first error is detected, whereupon the address is displayed followed by the data written into memory and what was read from it. This command is most useful for checking how much memory a system contains. For example, if the system contains 16K of memory, 4000 00 FF should be printed, indicating that there is no memory at address 4000H. Since the test is non-destructive to data in memory, it can be used at any time.

Mon>O <PORT> <DATA> - OUTPUT TO PORT

The two hex digits "DATA" are loaded into the accumulator and the instruction "OUT PORT" is executed. This command is useful for checking out peripheral equipment. For example, if a printer is connected to I/O port 6, O 06 41 will cause an "A" to be printed since 41 is the hex ASCII code for "A".

Mon>P <ADR1> - PROGRAM MEMORY

The contents of 16 bytes of memory containing ADR1 are displayed in both hex and ASCII, allowing preceding and following instructions to be viewed. Advancing to the next instruction is accomplished by typing space or cursor right (right arrow). Backspace or cursor left (left arrow) goes backwards. The cursor up and down keys move to an adjacent 16 byte block. Any hex characters typed will replace the existing contents of RAM. After every keypress, the screen display is refreshed by reading from memory, so the display reflects the exact memory contents. To terminate, depress ESCAPE.

Mon>Q <ADR1> <ADR2> - COMPUTE CHECKSUM

The MOD 256 checksum of memory contents in the address range specified is computed and displayed. This command is useful for checking proms or files to see if anything has changed. Any source file or program written in pure code (it does not write on itself) will have the same checksum as when it was loaded. While debugging assembly language programs, it is useful to be able to verify that a program being debugged has not written garbage in the source file or assembler.

Mon>R - REGISTER DUMP

This command will print a header identifying the Z-80 registers, and immediately below it the contents of all the registers. The flags are displayed with the letters Z C M E H for the zero, carry, minus, parity even, and auxiliary or half carry flags respectively. The presence of the letter indicates the flag is true. The contents of the memory locations pointed to by the B, D, and H register pairs are also displayed as is the return address on the stack.

Mon>S <ADR1> <ADR2> <BYTE> - SEARCH FOR SINGLE BYTE

This is similar to the "F" command, except that only one byte is searched for instead of two. An example of the use of this command is to display all locations in a program where an output to a port occurs (D3). The address of each location will be displayed followed by "D3" and the next byte (the port number).

Mon>T <ADR1> <ADR2> - TEST MEMORY

This is an extremely useful command, especially when first setting up a system. This command permits thorough testing of the system memory. A portion of a 64K byte pseudorandom number sequence is written into memory from ADR1 through ADR2, and the exact same sequence is regenerated from the initial point and compared with what is read from memory. If all locations compare, another portion of the sequence is used to repeat the test which continues until it is interrupted. Any memory errors are displayed with the address, what was written into memory and what was read from memory, respectively. This information is all that is needed to pinpoint a malfunctioning memory chip. This test is quite exhaustive if used for at least 10 cycles and is far superior to incrementing or complementing tests which may not reveal addressing problems. The only area of system memory that cannot be tested with this routine is the few bytes required for the stack and video flags in the vicinity of FFD0H on the ZCB board.

Mon>U - JUMP TO 0100H

This command permits easy return to programs in the transient program area of CP/M.

Mon>V - 8" DRIVE BOOT

Typing this command will cause a jump to E800H (contained on all current Disk Boot PROMs) which is the location of the 8" drive bootstrap loader. The boot program will cause the CP/M operating system to be loaded into memory and control to be transferred to CP/M.

Mon>W - WINCHESTER DRIVE BOOT

Typing this command will cause a jump to E802H (contained on all current Disk Boot PROMs) which is the location of the Winchester drive bootstrap loader. The boot program will cause the CP/M operating system to be loaded into memory and control to be transferred to CP/M.

Mon>X <ADR1> <ADR2> <ADR3> - EXCHANGE MEMORY BLOCKS

A block of memory from ADR1 through ADR2 is exchanged with an equal length block starting at ADR3. This command is useful in comparing the operation of two versions of a program, or for rapid switching of portions of a program without destroying the original. A loaded BASIC program can be exchanged with another if care is used to include the stack area (usually below the top of allowed memory).

Mon>Y - KEYBOARD ECHO

This command causes keyboard input to be echoed directly to the video driver and can be used for demonstration purposes. An ESCape returns to the Monitor.

Mon>Z <ADR1> <ADR2> <DATA> - ZERO OR FILL MEMORY

The memory block from ADR1 through ADR2 is filled with the byte "DATA". This is useful for setting memory to Zero. The end of a file or assembled program will stand out more clearly if memory is first zeroed. For test purposes, single instructions can be executed continuously so that bus waveforms are more easily interpreted. This is done by filling a block of memory with a repeated instruction sequence with a jump to the start of the block so that the program loops continuously.

ENTRY POINTS

A jump table at the beginning of the Monitor can be used to access several routines:

E000 - The normal cold entry point to the Monitor Executive, this is a jump to the initialization routine which clears the screen and initializes 8251 USARTS through I/O ports 3, 5, and 7. This is compatible with the Bitstreamer I addressed starting at port 4 ,the Bitstreamer II addressed starting at port 2 or all ZCB's with standard port addressing. The USARTS are set for an X16 baud rate factor and other parameters as would be used with a serial printer or extra terminal.

E003 - This is a jump to the routine which should be used for console keyboard status test. Return with the zero flag set indicates no keyboard input.

E006 - This is a jump to the keyboard data input which returns with the character in the "A" register. The keyboard code conversions described below are carried out. There is no checking for ESC key depression.

E009 - This is a jump to the video driver which displays the character in "A" on the screen.

E00C - This is a jump to the "ESCAPE" routine which returns zero if no input, or with the character in the "A" register if there is. Keyboard code conversions are carried out. If the ESC key was pressed, the system returns to the Monitor Executive.

VIDEO DRIVER

Version 4.x of the Monitor contains a more elaborate video driver than previous versions. The purpose of the video driver is to accept a stream of ASCII codes, and to write them into the screen memory in the proper place, interpreting certain non printing control codes in a special way. There are several entry points to the video driver. E009H is recommended. The character code to be printed must be in the A register. A CALL E009 will cause the character to be printed on the screen at the cursor position. All registers will be preserved.

Control codes are generated by the keyboard by holding the contrgd (CTRL) key down while a letter key is pressed. Control codes have values between 0 and 31, and are 64 less than the codes for the corresponding upper case letters. To demonstrate the features of the video driver, type Y after the Monitor prompt, and any keyboard generated code will be echoed to the video driver. The following control codes are interpreted as special functions, while all others are ignored:

Decimal Value	Hex Value	Control Code	Description
2	2	(^B)	HOME THE CURSOR
4	4	(^D)	CLEAR THE SCREEN AND HOME CURSOR
5	5	(^E)	DISPLAY THE CODE IN B REGISTER
8	8	(^H)	DESTRUCTIVE BACKSPACE (also BACKSPACE key)
9	9	(^I)	TAB OVER TO THE NEXT 8 MULTIPLE (also TAB)
10	A	(^J)	LINEFEED (also LF Key)
13	D	(^M)	CARRIAGE RETURN (also RETURN key)
14	E	(^N)	TOGGLE CURSOR
16	10	(^P)	CLEAR TO END OF SCREEN
17	11	(^Q)	CLEAR TO END OF LINE
18	12	(^R)	CURSOR DOWN
20	14	(^T)	TOGGLE REVERSE VIDEO
21	15	(^U)	CURSOR UP
23	17	(^W)	CURSOR LEFT
24	18	(^X)	CLEAR TO START OF LINE
26	1A	(^Z)	CURSOR RIGHT
27	1B	ESC	CURSOR XY POSITION LEAD-IN

Experiment with the keys. There are special keys on the keyboard to generate some of the codes such as RETURN, TAB and linefeed (LF). If you are using the Vector Graphic Keyboard or Mindless Terminal, there are also keys for the cursor control and BACKSPACE. A few of the functions are not self explanatory. A Control D sets the reverse video flag to normal in addition to clearing the screen and homing the cursor. A Control T will then toggle the reverse video flag from normal to reverse and back without printing on the screen.

In some cases it is desirable to print the symbol for a control code on the screen. This can be done in assembly language programs by putting the code for the symbol in the B register and calling the video driver with Control E (05) in A. Enter the following machine code at FC00H and execute it to demonstrate this feature:

```
at FC00  06 01 3E 05 04 CD 09 E0 CD 0C E0 C3 02 FC
```

CURSOR X Y POSITIONING

Many programs utilize random X Y positioning of the cursor. This is done by outputting a three byte sequence to the video driver. The first code is ESC (1BH) followed by the desired X position and Y position in hex. The top left corner of the screen is 0, 0. The assembly language sequence 1B 40 08 would cause the cursor to move to line 8, character position 64 on the screen. To send the same sequence to the Monitor via Microsoft Basic, the following statement would be used: "PRINT CHR$(27);CHR$(X+128);CHR$(Y+128);" where X would equal 64 (40H) and Y would equal 08 (08H). Adding the value of 128 to X and Y in this example sets the eighth bit high. This is done to avoid Microsoft Basic from confusing the values as control codes. This may not be demonstrated using the keyboard since ESC causes a return to the monitor.

The video driver provides an extensive range of special controls, however, they must be incorporated into the software generating the video stream to be meaningful. For instance a piece of software that merely echoes all characters as they go into its input buffer will allow cursor motion on the screen, but this will probably be meaningless to the software.

KEYBOARD CODE CONVERSION - VECTOR GRAPHIC KEYBOARDS

Due to limitations in the keyboard encoder chip, the [] key on Vector Graphic keyboards is not encoded properly. The correct code is generated by a conversion routine in the Monitor's CONVERT routine. The codes for backslash and tilde are also produced by the control and control shift mode of this key.

[] KEY CONVERSION:

MODE	KEYCODE	CONVERTED CODE	ASCII SYMBOL
unshifted	F1	5B	[
shifted	E1	5D	]
control	B1	5C	\
control shift	A1	7E	~

The cursor up key is also converted from 60H to 15H which is interpreted correctly by the video driver. Room is provided in the routine for up to 15 keycode conversions. Foreign languages require additional conversions, and versions are available for French, German, Swedish and Spanish. It i

essential that software utilize the monitor conversion routine for this reason.

USING THE I/O ROUTINES

The I/O routines in the Monitor are used as the Main System I/O in Vector Graphic Systems. This makes software I/O independent and easily interchangeable between systems. An example of how this is done is shown below:

```
INPUT ROUTINE:      INPT    CALL E00CH
                            JZ   INPT
                            RET  (RETURNS WITH CHAR INPUT IN A)

OUTPUT ROUTINE:     OUTPT   JMP  E009H (CHARACTER IN A)

BREAK TEST:         CONTL   CALL E00CH
                            RET  (RETURNS WITH ZERO FLAG SET IF NO
                                 INPUT, OR CHARACTER IN A.  JUMPS
                                 TO MONITOR EXECUTIVE IF ESCAPE
                                 INPUT.)
```

Note that the ESC key will break to the Monitor, which provides a convenient way of transferring control from any executive such as the DOS or BASIC to the Monitor, but necessitates the use of another character (Control C is standard) for a single level break. The routines above are merely given to illustrate how simple it is to use the Monitor I/O routines. Many programs require additional instructions to move the character to be output into the accumulator, or may require different flag conditions or accumulator contents on return from the input and Break Test routine, but the variations are easily implemented.

OTHER USEFUL MONITOR ROUTINES

The Monitor contains a number of routines that can be called by user programs, and which will save considerable programming effort. In addition to the keyboard input and video output described elsewhere, we have:

AHEX inputs four hex digits from the keyboard and returns the binary value in D,E registers. A space is automatically output at the end. All registers, except B, are used. Entry at AHE0 with a value of 1-3 in C will convert that many digits. Non hex values will be ignored.

CRLF will output a carriage return and line feed to the screen. The A register is used.

SPCE will output a space to the screen. The A register is used.

RNDM returns a new random number in B,C based on the seed in B,C as it is called. B,C should not contain 0000. The pseudorandom number sequence generated is $2^{16}-1$ entries long and is based on a software simulation of a shift register with maximum length feedback. PSW is used.

PTAD first outputs a CRLF, then outputs the binary value in H,L as four hex digits followed by a space. PSW used.

PT2 outputs (A) as two hex digits.

TAHEX calls AHEX twice, inputting two address fields of four hex digits. The first value is returned in H,L; the second in D,E.

The addresses of these routines and others may be found by consulting the listing which follows.