Learning

16-BIT MICROCOMPUTER TECHNOLOGY

Part 7: How to use the monitor to create software

By George Meyerle

LAST month we introduced the ROM-based monitor and its entry commands. Here we will show how to use the monitor to create software.

All monitor commands require one or two character entries plus their optional address and/or iteration fields. Command lines are terminated by operating the carriage return key which starts execution of the command.

Following is a list of conventions used:

1. The > character is the monitor video prompt. When it is displayed, the monitor is waiting for a command entry.
2. The < character indicates that a carriage return must be initiated. This character does not appear onscreen, and is used here only to indicate carriage return.
3. Any values within parenthesis are optional. The parenthesis are not entered but appeared for clarity only.
4. The value "FA" represents the Full Address i.e.: an address that may contain both a segment and an offset address value. The segment and offset are always separated by a colon as in the form CS:IP (CS = segment register; IP = instruction pointer). The segment value is optional and if it is not used, then the current code segment is assumed to be operative.
5. The value "AD" represents an *offset* address value. No segment address is allowed within a "AD" field.
6. Any register pair designation may be used in place of an address.
7. All address fields must be separated by one or more spaces.
8. All entries are in hexidecimal with the exception of the iteration count which is in decimal.
9. The iteration count is entered by typing in an asterisk followed by a numeric value. The default iteration count is 32.

The Command Set.

DISPLAY

The format is:

```
D (FA) (AD) <
D (FA) *nnnn <
```

Example:

```
D F000:E000 *5 <
```

In this example, F000 is the code segment start address (however, since we are dealing with a 16-bit processor, this is actually F0000), E000 is the offset address found in the instruction pointer), and 5 is the number of bytes required to be displayed.

The absolute hex address where the monitor will actually go to is found by adding the code segment and offset addresses as follows:

```
F000     code segment
 E000    offset address
FE000    absolute hex address
```

Thus, if the command D F000:E000*5 were executed, the first five bytes of whatever started at address FE000 (1,040,384 decimal) would be displayed. In our case, this happens to be FA B4 D5 9E 73, first five bytes of the Explorer-88 ROM-based monitor program as shown in Fig. 19.

The same absolute address can be obtained by using any combination of code segment and instruction pointer values that, when added, equal the absolute value FE000

Being able to set the code segment value at 64K boundaries has the advantage of breaking up the 1 million memory addresses into sixteen 64K blocks which is convenient. Keep in mind though, that the flexibility of almost endless combinations of segment and offset addresses can be very useful. Because of its complexity, it would require a small book on 8088 addressing modes to describe how this addressing scheme is employed. The reader should consult any of the currently available 8086/8088 books for details.

The "D" command, can also be used to display the following:

D SS:SP *2 < (SS is stack segment while SP is stack pointer) displays the first two bytes of the stack pointer.

D 0:0 FF < Displays the absolute locations

00000 to 000FF. Note that the first 32H bytes displayed are the interrupt jump vectors.

D DS:0 < Displays the first 32 bytes of the data segment (DS).

REGISTER

Format R <

Example R (RP)

This group of commands is perhaps one of the most important features of the monitor. To know what has happened in your program, you must be able to review what has happened to the registers inside the microprocessor.

The 8088 registers are shown in Fig. 20. The general registers are subdivided into two sets of four registers each. Each segment is separately addressable which means that one register can be used as one 16-bit, or two 8-bit registers. All other registers are 16 bits wide. In addition to the registers, nine status flags are available.

An R followed by a carriage return will cause a display of *all* the internal 8088 registers. An R followed by a register pair designator (RP) will cause the selected pair to be displayed and allow modification of that pair. If a change is required, enter the change followed by a < (carriage return). If no change is required, entering a < will terminate the review.

A typical register display might look like this:

```
DX  = 0000  SP    = 007E
CX  = 0000  BP    = 0000
BX  = 0000  SI    = 0000
AX  = 04FF  DI    = 0000
SS  = 0FFB  CS:IP = E8
ES  = 0000
DS  = 0FF0  FL    = 0000
CS  = F000  IP    = EDB7
```

```
E000   INIT    LABEL   NEAR
E000   FA      CLI                 ;INTERRUPTS MUST BE OFF
E001   B4 D5   MOV     AH,0D5H     ; SET FLAGS
E003   9E      SAHF
E004   73 4E   JNC     ERR01       ;JUMP IF CF NOT SET
E006   75 4C   JNZ     ERR01       ;JUMP IF ZF NOT SET
E008   7B 4A   JMP     ERR01       ;JUMP IF PF NOT SET
```

Fig. 19. First five bits of the monitor program.

BLOCK MOVE

The block move command is helpful to relocate a program/subroutine, or, if you want to fill a portion of memory with a constant. The formats are as follows:

B FA AD AD<

B FA AD *nnnn<

The first address is the beginning address of the block to be moved. The second address is the beginning address of where the block is to be moved to. The third address is the last address of the destination block which could alternatively be replaced by the iteration count designator which is simply the number of bytes to be moved.

The command: B 0:0 500 5FF < will move the data in locations 00000-000FF to 00500-005FF. The data in the source field is not changed.

The command B 0:0 500 *1000< will fill 1000 locations starting from 00500 with the data value stored at location 00000.

MEMORY

The M command opens memory locations for display or modifications, and is used to enter programs. The format is:

M (FA)<

After inputting the M command, the designated memory location is displayed followed by a dash. If a space is entered the next sequential location will be displayed.

If a change is desired, enter the new data followed by a space. A carriage return will end the command. If a memory location cannot be changed (as would be the case if ROM or bad or nonexistant RAM were addressed) a bell or beep will be outputted.

INPUT

The I command inputs one byte from the specified I/O address to the Accumulator and displays the value on the terminal.

Format: I AD<

Entering I 61< inputs and displays the state of the dip switch S1 on the mother board of the Explorer-88.

OUTPUT

The O command outputs one byte to the output device specified.

Format: 0 AD xx<

Where xx is the value of data outputted to port address AD

CASSETTE

The cassette read and write commands will not be discussed in any detail since the format used is the same as for the IPM PC.

TRACE

The T commands are perhaps the most important in the monitor. They allow you to single-step through a program tracing the execution of each instruction and observing the registers at every step. After execution, control is returned to the monitor and the new value of CS:IP along with the next instruction to be executed is displayed. Optionally, all registers may be displayed if the T command is followed by an R (TR command). This command also allows tracing more than one instruction at a time by entering an asterisk followed by the desired iteration count value. The formats are:

T FA<

T FA *nnnn<

TR FA<

TR FA *nnnn<

As an example of how to use the trace command, lets trace the P(ROM) test portion of the hardware test. Before we start the trace, we have to make sure that code, data, and stack segments are set to match the definitions established when the monitor program was written.

Using the register commands, modify the following registers.

CS= F000 DS=0FF0 SS=0FF8

Now enter

TR F000:ECDO *5<

This command will begin tracing the first 5 steps of the ROM testing program which starts at the absolute address FECDO. Note that the next address to be executed is EDB7. To complete the trace to the end of the test will require in excess of 16,000 steps because all 8K locations of the ROM are added together and checked for the proper result. Step through the first part checking the registers against the results you might expect.

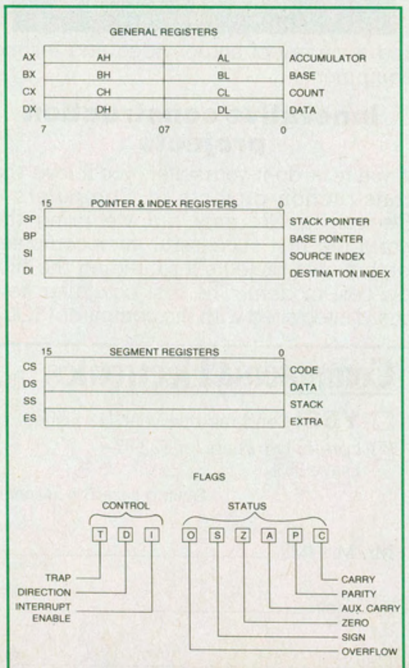

Fig. 20. 8088 registers.

GO

The G command starts program execution at the Full Address specified. If no address is specified, then execution starts at the current CS:IP location. There is an optional break-point address that can be entered to cause control to be passed back to the monitor if the break-point is entered, the monitor looses control of the system. A dash or hyphen entry must precede the break-point address. All registers may be displayed after the break-point is reached if the "GR" command is used. Format:

G (FA) (-AD)<

GR (FA) (-AD)<

Try the Go command by going to the beginning of the monitor hardware test which begins at F000:EDB7. Make sure that the CS, DS, &SS registers are set as described in the trace test. Enter

G F000:EDB7<

The hardware tests should now be executed just as if they were called directly by the monitor. To test the breakpoint enter GR F000:EDB7-EE05. This will cause the program to break at the beginning of the cassette test.

HARDWARE TEST

The format is: HT<

The final test described is a complete hardware test to perform diagnostics.

1. Read only memory
2. Random Access memory
3. The cassette interface
4. The interrupt controller
5. The timer circuitry
6. The DMA controller
7. The RS 232 port (not tested in the IBM compatible version)

Enter:

HT<

Conclusion. Keep in mind that the 8086 and 8088 are almost "clones," although the 8088 has an 8-bit external data path to memory and I/O and the 8086 can transfer 16 bits at a time, the two processors are otherwise identical. Software written for one CPU can run on the other, without alteration.

What is more important is to experiment with the commands within the Explorer-88 monitor. There is no way you can cause physical harm to the system by issuing wrong or faulty commands. The worst thing that can happen is for the system to "hang up." In this case, simply reset and start over. ◇