DEMON

For TRS-80 Model I and Model III

Distributed by:

Mumford Micro Systems
Post Office Box 400
Summerland, California 93067
(805) 969-4557

Molymerx

DISTRIBUTED IN NEW ZEALAND
by

MOLYMERX Ltd.

P.O.Box 60152 Titirangi.
Tel.817-4372

DISTRIBUTED IN AUSTRALIA
by
MICRO-80 PTY LTD
P.O. BOX 213 GOODWOOD 5034
433 MORPHETT ST ADELAIDE 5000

```
*****************************
*    THE DEMON DEBUGGER     *
*                           *
*    Table Of Contents      *
*****************************
```

INTRODUCTION

The DEMON (for DEbugging MONitor) is a relocating machine language monitor and stepper for the TRS-80 Model I and Model III. It supports input and output commands for both tape and disk systems, and will also output to a line printer. It will step through your program or run in slow motion at several speeds, displaying disassembled instructions and register and flag contents on the bottom two lines of the CRT screen, without interfering with your program's use of those lines.

Programs may be rapidly debugged and/or their operation traced with CALLS and RST's executed at full speed as needed. High speed stepping can stop at each branching instruction, or at a preset address, or if a number in a given range is referenced, or by counting steps. Breakpoints set in RAM are removed when they are used, but may be reset automatically. DEMON will run in systems with as little as 16K of RAM. The following features also merit mention:

1) Numbers may be entered in decimal or hexadecimal.
2) Full screen memory display allows rapid editing in ASCII or hexadecimal.
3) You can disassemble at an independent address while single stepping.
4) Disk input and output is allowed down to 5200H, or even lower if your system will allow it.
5) When saving memory on disk or tape, up to 8 separate blocks can be combined in a single recording.
6) A labelling disassembler is included with output in EDTASM source format.
7) DEMON will relocate both itself and other programs.

GROUND RULES

To begin with, let's define some of the general conditions and conventions that will be observed in the following instructions and in DEMON itself. There are two major modes in DEMON: the MONITOR mode and the STEP mode. Each of these primary modes has a variety of commands available to it (see COMMAND SUMMARY). For the purposes of these instructions, the expression "command level" will refer to the state DEMON is in when it can accept new commands in either of these two modes. The "command level" for the MONITOR MODE will be indicated by the prompt "CMD?" appearing in the upper left corner of the screen. The "command level" of the STEP MODE will be indicated by the prompt "?" or "!" in the lower left corner of the screen.

There are a couple of other expressions that will be used frequently. One of these is "target program". What we will mean by this is the program being debugged or examined by DEMON. Another word that will be used a lot is "step". For our purposes, a "step" will be considered a single machine language instruction in the target program, or the execution of this instruction.

Abbreviations

There will also be some abbreviations used in these instructions. Many times, you will be told to type a key on your computer. The key you are to press will be bracketed by "greater-than" and "less-than" characters. In these cases, you should only press the key that is between these two characters. Some typical examples are:

<Y>	-Press the "Y" key if the answer is "Yes".
<N>	-Press the "N" key if the answer is "No".
<ENTER>	-Press the "ENTER" key.
<SHIFT UP ARROW>	-Hold the SHIFT key, and press the UP ARROW key.

Number Entry

When entering numbers, such as addresses, you may enter either decimal or hexadecimal values. To enter a hexadecimal number, type from 1 to 4 hexadecimal digits (0-9, A-F), and press <ENTER>. To enter a decimal number, type from 1 to 5 decimal digits (0-9), and press the decimal point (or period) instead of <ENTER>.

Illegal entries will erase the complete number and you will have to start over. Some illegal entries are addresses greater than 65535 decimal or FFFF hexadecimal, bytes greater than 255 decimal or FF hexadecimal, and decimal numbers with hexadecimal digits (A-F).

Before pressing <ENTER> or <.>, <LEFT ARROW> will backspace over any characters you might want to change. If <ENTER> or <.> is pressed when DEMON expects a numeric entry, but you haven't entered any digits, a default value of 0 will be entered (the "0" doesn't display on the screen). One exception to this is the default value for the RUN address, which is FFFF (see STEPPER command R).

When DEMON displays a number, most hexadecimal values will display 2 or 4 digits. Decimal numbers will end in "." except for the step count at bottom right corner of the stepper display, which will not have the decimal point.

The <BREAK> Key

The <BREAK> key is recognized at almost every point in DEMON as an escape to the command level of the mode you are in. <SHIFT BREAK> will return to the command level of the MONITOR MODE from the STEP MODE. Most commands may be aborted by hitting <BREAK>. Disk and tape operations, as well as line printing, will require that you hold the <BREAK> key down until the operation terminates. When writing to disk, <BREAK> will terminate writing but not KILL the file. In other words, you will end up with a partially written file on your disk.

COMMAND SUMMARY

DEMON has two main parts, the MONITOR, and the STEPPER. Each has its own set of commands and distinctive command prompts. The following list is a summary of these commands. A thorough description of each one will follow.

MONITOR COMMANDS
(Prompt = "CMD?")

A	ARITHMETIC
B	BLOCK MOVE
C	CONTINUE STEPPING
D	DISASSEMBLE TO SCREEN
<SHIFT D>:	D WITH LINE PRINT
E	DISASSEMBLE TO EDTASM SOURCE
F	FIND BYTES
<SHIFT F>:	F WITH LINE PRINT
-	--
H	NUMBER BASE CONVERSION
-	--
J	JUMP TO MEMORY
-	--
L	LOAD FROM DISK OR TAPE
M	MEMORY EDIT
-	--
-	--
P	PROGRAM RELOCATION
<SHIFT P>:	LINE PRINT CONTROL
Q	RETURN TO MEMORY EDIT
R	RELOCATE DEMON
S	STEPPER MODE
-	--
-	--
-	--
W	WRITE TO DISK OR TAPE
Z	ZERO BLOCK OF MEMORY

<BREAK> RETURNS TO COMMAND LEVEL

STEPPER COMMANDS
(Prompt = "?" or "!")

A	ARITHMETIC
B	BRANCH STEP
C	CLEAR BREAKS
D	DISASSEMBLE TO SCREEN
-	--
E	EXTERNAL BREAK
F	FLAG REGISTER EDIT
-	--
G	BREAK ON NUMBER OF STEPS
H	NUMBER BASE CONVERSION
I	INTERNAL BREAK
J	JUMP TO MEMORY
K	RESET E BREAK AND JUMP
L	LIST BREAKPOINTS
M	MEMORY EDIT
N	BREAK ON NUMBERS IN A RANGE
O	RESET STEP COUNTER TO ONE
P	ENABLE LINE PRINTER
<SHIFT P>:	LINE PRINT CONTROL
Q	RETURN TO MEMORY EDIT
R	RUN BY CONTINUOUS STEPPING
S	RESTORE STEPPER DISPLAY
T	RESTORE TARGET DISPLAY
U	JUMP UNDER CALL OR RST
V	VARY CONTINUOUS STEP SPEED
-	--
Z	DISPLAY OR ALTER Z80 REGISTERS

<BREAK> RETURNS TO COMMAND LEVEL
<SHIFT BREAK> RETURNS TO MONITOR MODE
<SHIFT RIGHT ARROW> = LINE PRINT
<ENTER> TAKES ONE STEP

PART 1: LOADING AND RUNNING DEMON

1.1 LOADING AND RUNNING DEMON FROM TAPE

DEMON is a machine language program supplied on a cassette recorded at 500 baud, so MODEL III TRS-80 users should begin by hitting RESET (hold down <BREAK> if you have disks) and responding to "Cass?" by pressing <L>. MODEL I users may just hit RESET. Both will find "MEMORY SIZE?" or "MEM SIZE?" displayed. Press <ENTER> to get to the BASIC prompt "READY". Next, type SYSTEM, press <ENTER>, and type the file name DEMON. Press PLAY on the tape recorder, and press <ENTER> on the computer. Asterisks should flash in the upper right corner of the screen while the tape is loading, and then the SYSTEM prompt "*?" will return. If the asterisks do not appear, or a C (for checksum error) or D (for data error) appear, check your connections to the tape recorder and the volume level and start over. Once you have gotten a successful load, press </> and <ENTER> to begin running DEMON.

1.2 LOADING AND RUNNING DEMON FROM DISK

If you received DEMON on disk, you will find that the disk is formatted for the current version of TRSDOS for your machine. If you have two drives, you may put this disk in drive 1 and your TRSDOS compatible operating system in drive 0. To load and execute DEMON from "DOS READY", just type DEMON and hit <ENTER>.

If you have only one drive, you will need to use a different procedure. The DEMON disk has a special structure that will allow you to copy the program on it to a TRSDOS system disk of your own. To do this, use the following step by step procedure:

1) Put a current TRSDOS system disk in drive 0 and hit RESET.
2) When the DOS READY prompt is displayed, remove the TRSDOS disk, insert the DEMON disk, and hit RESET again.
3) The disk should "boot up" with our sign on message. This message will tell you which version of TRSDOS it is designed to work with. If the system disk you are using is not the same type, put the correct system disk in drive 0 and go back to step 1.
4) The program name DEMON/CMD will also be displayed and you will be asked to put your system disk back in drive 0.
5) Put your system disk in drive 0 and enter W to write the program onto your own disk.
6) When the program has been safely stored on your own disk, hit RESET to re-boot the system. You may then run DEMON from DOS READY by just typing the file name DEMON and hitting <ENTER>.

When the DEMON begins execution, the screen will clear and display the title page with "DEMON" in large letters. Press any key. The screen will clear again and display the monitor command level prompt "CMD?".

The re-entry point to DEMON is always the first address it uses. If you relocate DEMON, this address will change. The re-entry point to the program as distributed is 24320 decimal or 5F00 hex (see Section 2.4, <R>).

PART 2: THE MONITOR COMMANDS

As stated earlier, there are two primary modes in DEMON. These are the MONITOR mode and the STEPPER mode. When you first enter DEMON, it "comes up" in the monitor mode. In this mode, the screen is clear except for the monitor mode prompt "CMD?", which will be at the upper left corner of the display. DEMON is now ready to respond to one of its commands. Most of the commands in DEMON require a single key, though some also require the shift key. When you give DEMON a command by pressing one of these keys, nothing will happen until you release the key. A prompt will then indicate what to do next.

2.1 MONITOR UTILITY COMMANDS <A>, <H>, <M>, <Q>, <D>, <SHIFT D>

<A> - Hexadecimal arithmetic (X+Y, X-Y, Y-X)

This command is used to perform simple arithmetic on two numbers. The numbers may be decimal or hexadecimal, but the answers are always given in hexadecimal (monitor command <H> can be used to get the decimal equivalents of the hex answers). After pressing <A>, the message "X = " will appear. You may now enter any decimal or hexadecimal value for X, following the conventions defined under GROUND RULES. When X is entered, the message "Y = " will be displayed and you can likewise enter a decimal or hex value for Y. After X and Y are both entered, the results of the arithmetic will be immediately displayed. You may then hit <SHIFT RIGHT ARROW> to print this line of information on the line printer. To return to the command level without line printing, just hit <BREAK>.

These are two byte calculations, so if the sum is greater than FFFF, the overflow or carry to a non-existent third byte is discarded. For example, 8888 hex plus 7778 hex equals 0000.

In the following examples, the text that is not underlined is supplied by DEMON. The underlined characters are your responses.

EXAMPLE 1: From the command level, press <A> and enter "234" and "345D". Note that these are both hex numbers because they are followed by the <ENTER> key instead of a decimal point. The resulting display will be:

```
X = 234  Y = 345D  X + Y = 3691  X - Y = CDD7  Y - X = 3229
```

You can now return to the command level by pressing <BREAK>, or send this line to your line printer by pressing <SHIFT RIGHT ARROW>.

EXAMPLE 2: Press <A> and try X = "2345." and Y = "45678.". Note that these are both decimal entries because they are entered with a decimal point instead of the <ENTER> key. The complete display will be:

```
X = 2345. Y = 45678. X + Y = BB97  X - Y = 56BB  Y - X = A945
```

Note that the sum and differences are always hexadecimal, regardless of the number types that were entered for X and Y.

<u><H> - Hexadecimal to decimal and decimal to hexadecimal conversion</u>

This command is used to convert numbers from base 10 to base 16 or vice-versa. After typing the command <H>, you will see the following display: "ENTER #: ". You may now enter any number according to the conventions defined under GROUND RULES. If you enter a decimal number (a number ending with a decimal point), you will be shown the hexadecimal value of this number. If the number you enter is hexadecimal (a number ending with the <ENTER> key) you will be given the decimal value of this number. After the conversion has taken place, you may use <SHIFT RIGHT ARROW> to have the results printed on your line printer. Otherwise, press <BREAK> to return to the command level.

In the following examples, the text that is not underlined is supplied by DEMON. The underlined characters are your responses.

```
ENTER #: AE00  =  44544. (A hex number is entered, a decimal value is returned)
ENTER #: 44544. =  AE00  (A decimal number is entered, a hex value is returned)
```

<u><M> - Full screen memory display and edit</u>

The memory edit function in DEMON is very fast and flexible. It will display 256 bytes at a time (16 bytes by 16 lines) in ASCII, hexadecimal, and graphics modes. After giving the <M> command, you will see the following prompt: "M ADDRESS: ". You may specify any valid address according to the conventions described under GROUND RULES. After entering the address you will see one page (256 bytes) of memory displayed in either hexadecimal or ASCII format, depending on which mode was last used. At the left edge of the screen will be the addresses of the first byte of each row of values. There will be a blinking cursor over the first character of the first byte in the top row. Finally, there will be a hexadecimal number in parenthesis at the top right of the screen. The blinking cursor indicates the position at which editing may be done, and the number in parenthesis is the exact address of this location. While you are in the memory mode, there are several special commands available:

THE ARROW KEYS - These keys move the blinking cursor up, down, right, and left. They will repeat automatically if held down. If the use of these keys causes the cursor to reach the top or bottom of the screen, the display will scroll to accomodate movement in the desired direction. The cursor's exact address will always be displayed at the extreme right of the screen, so there is no need to count columns. As stated earlier, the position of this blinking cursor is the location at which changes in memory can be made.

<SHIFT UP ARROW> and <SHIFT DOWN ARROW> - These commands cause the area of memory being displayed to move backward or forward by one page (256 bytes).

<SHIFT RIGHT ARROW> - This command is used to send the memory display to your line printer. After typing <SHIFT RIGHT ARROW>, the screen will clear and you will be asked for a "LAST ADDRESS: ". After entering this address, the memory display starting at the last cursor position and ending with your "LAST ADDRESS" will be sent to the line printer. The memory display on the printer will contain hexadecimal bytes and an ASCII representation of those bytes on each line. The hex bytes will be grouped in pairs and the ASCII characters will be bracketed in parentheses. Because the screen is cleared before you are asked for a LAST ADDRESS, be sure to make a mental note of this address before typing <SHIFT RIGHT ARROW>. <BREAK> may be used to abort printing early.

<CLEAR> - This command is used to switch between hexadecimal display and ASCII display. The <CLEAR> key will "toggle" the mode: if you are in hex mode, it will change the display to ASCII mode; if you are in ASCII mode, it will change to hex mode. In the ASCII mode, bytes with values greater than 7F hex will be displayed as periods. When line printing in this mode, values less than 20 hex are also changed to print as periods. (This change is also available as an option for screen display, see TECHNICAL INFORMATION.)

<SHIFT LEFT ARROW> - This command is used to switch back and forth between regular ASCII mode and the GRAPHICS/ASCII mode. The only difference between these two modes is the upper limit where bytes are changed to print as periods. In the regular ASCII mode the limit is 80 hex. In the GRAPHICS/ASCII mode it becomes C0 hex. This will allow screen graphics characters to be displayed. When line printing with <SHIFT RIGHT ARROW> in the GRAPHICS/ASCII mode, you will automatically be switched back to regular ASCII mode.

While you are in the memory mode, you may at any time change a value in RAM that is displayed on the screen. To edit in the hexadecimal mode, type two hexadecimal digits for each byte. The blinking cursor will advance one position after each character. You will notice that two digits are required for each byte, and after you have entered one of them, you must enter another valid digit before you will be able to use the arrow keys again.

To edit in the ASCII mode, just type characters. In this mode only one character is required for each byte. While it may appear as though the <ENTER> key leaves a letter "M" in memory, it is actually entering a 0D hex which is the value of a carriage return. The appearance of the letter M is a peculiarity of the TRS-80. Some TRS-80s, however, will show some other character in place of an "M".

Note that the display shows the actual result in memory. If you try editing at ROM addresses, there will be no change either in memory or on the screen. The <M> command will, however, change any byte within DEMON itself but unless you know exactly what you're doing, don't do it (see TECHNICAL INFORMATION). To return to the command level from this function, hit <BREAK>.

<Q> - Quick return to memory edit

This command may be used as a quick return to the last memory edit display. After typing <Q>, you will re-enter the memory edit mode. The page of memory displayed will be the same as was last displayed in the <M> mode, and the cursor will be in the upper left corner.

<D> - Disassemble

This command will display disassembled Z-80 instructions on the video screen, starting at a specified address. After pressing <D>, the message "D ADDRESS: " will appear. You may now enter any valid address in hex or decimal, and 16 lines of disassembled instructions will be immediately displayed on the screen. Now that you are in the disassembly mode, there are three special commands available. <UP ARROW> will cause continuous disassembly, with the screen scrolling as necessary. <ENTER> will advance the display by 16 lines (one screen full). <SPACE BAR> will advance the disassembly by one line only.

Each line displays the hexadecimal address, the object codes that form the instruction, and the disassembled instruction in Zilog mnemonics. Undocumented or illegal machine language code combinations (such as the two hexadecimal bytes "DD DD") lead to the display of the address, from 2 to 4 object codes (starting at the specified address and ending at the first bad code), and the message "BAD CODE". The address is then incremented by 1 and disassembly continues with the next byte.

EXAMPLE OF A BAD CODE: With DEMON in its original location, place the codes "DD DD 21 00 00" at memory address 9500 (hex) by using the monitor command <M>. Return to the command level and attempt to disassemble these codes by pressing <D> and entering the address 9500. The display just before pressing <ENTER> will be:

```
D ADDRESS: 9500
```

The display after pressing <ENTER> will be:

```
9500 DDDD                          BAD CODE
9501 DD210000    LD    IX,0000
```
(And 14 more disassembled instructions below this.)

EXAMPLE WITH NORMAL CODES (The first few codes of DEMON):
From the command level, press <D> and type 5F00 <ENTER>. The top lines of the display will be:

```
5F00 C3A670      JP    70AA
5F03 EDB8        LDDR
5F05 FDE9        JP    (IY)
5F07 1A          LD    A,(DE)
5F08 13          INC   DE
```

<SHIFT D> - Disassemble to line printer

This is the monitor command for disassembly to the line printer. DEMON uses the standard "device control block" for printer output, so if you have a non-standard printer and a special driver for it, it will be automatically supported (see TECHNICAL INFORMATION). After entering <SHIFT D> the message "FIRST ADDRESS: " will be displayed. You may enter any valid hex or decimal address. You will then be prompted for a "LAST ADDRESS: ". To abort this function, press <BREAK> instead of entering an address. Otherwise, as soon as you enter the address, disassembly will begin and continue until the "LAST ADDRESS" is reached or exceeded.

Each disassembled instruction will be displayed briefly on the bottom line of the screen as it goes to the printer. To stop printing at any time, press and hold <BREAK> until the printer stops. Note that if your printer has a large buffer in it, it may take some time for it to stop even though DEMON has stopped sending it code.

The message "BAD LIMITS" will appear and there will be no printing if the range of addresses specified overlaps the area of memory used by DEMON itself. See the stepper command <D> for another way to disassemble to the printer.

2.2 MONITOR BLOCK COMMANDS <B>, <F>, <SHIFT F>, <Z>

<B> - Block move

This command will move the data in a block of memory from an existing location to any new location in RAM. This command will not function on blocks that overlap DEMON either before or after the indicated move. It will, however, function on blocks that overlap each other in either direction. DEMON is smart enough to prevent the move itself from writing over its own data.

After entering the <B> command, you will be asked for a "B FIRST ADDRESS:". This is the starting location of the data you want to move. After entering either a decimal or hex address, you will be asked for a "LAST ADDRESS:". This is the ending location of the data you wish to move. After entering this address, you will be asked for a "NEW FIRST ADDRESS:". This is the start of the area in RAM you want to move the data to. After entering this address, the move will be made (if allowed) and you will return to the command level.

The examples which follow assume DEMON is in its original location. Start from the command level each time, and press <B>. The text which is not underlined is supplied by DEMON. The characters that are underlined are your responses.

EXAMPLE 1:

```
   B FIRST ADDRESS: 4000   LAST ADDRESS: 4200
   NEW FIRST ADDRESS: 6000
   BAD LIMITS
```

In this case, the data would overlap DEMON after the proposed move, so no move is made and the message "BAD LIMITS" is displayed. Press <BREAK> to return to the command level.

EXAMPLE 2:

```
   B FIRST ADDRESS: 9000   LAST ADDRESS:  900F
   NEW FIRST ADDRESS: 9003
```

This is the display just before hitting <ENTER> after "9003" has been typed. The move will be made as soon as <ENTER> is pressed, and you will return to the command level. As shown here, the new block may overlap the old block in either direction. Assume the contents of memory at address 9000 hex before this move had been:

```
   01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 00 00 00
```

After the proposed move, the contents would be:

```
   01 02 03 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
```

<F> - Find a set of bytes

This command finds all occurances of a set of consecutive bytes in a specified range. After typing <F> from the command level, you will be asked for an "F FIRST ADDRESS:". This is the start of the block of memory you want to search through. You will then be asked for a "LAST ADDRESS:". This is the end of the block of memory you want to search through. Finally, you will be given the prompt "FIND:", which asks for the bytes you want to search for. You may enter from 1 to 10 bytes, though 3 or 4 is usually enough. To enter the first byte, type either a decimal or hex value as defined under GROUND RULES. After ending this number with either a period (for decimal numbers) or <ENTER> (for hex numbers), a question mark will be displayed. DEMON is now waiting for the next byte you want to search for.

If you enter another number, the process will be repeated. If you type <LEFT ARROW>, all numbers entered so far will be erased. If you type <RIGHT ARROW>, the search will begin and the locations of all occurances of the defined string of bytes in the defined range will be displayed. These addresses will be displayed 10 at a time, after which the display will stop and wait for you to hit <RIGHT ARROW> again to get the next 10 locations. When the search is complete, the prompt "FIND:" will appear again so a new set of bytes may be entered for a search between the same addresses. To change limits or terminate the search, press <BREAK>. If you specify a range that overlaps the area of memory used by DEMON, the message "BAD LIMITS" will be displayed. If nothing is found, the ">" sign will be displayed on a blank line, and you will be given the "FIND:" prompt again.

The following example shows the result of a search of the top of video RAM (the top line of the video screen, in fact), first for three consecutive spaces, and then for a space and "A". The text which is not underlined is supplied by DEMON, the underlined characters are your responses:

```
F FIRST ADDRESS: 3C00   LAST ADDRESS: 3C40
FIND: 20  20  20 ?    <RIGHT ARROW>
3C15 3C2A 3C2B 3C2C 3C2D 3C2E 3C2F 3C30 3C31 3C32  <RIGHT ARROW>
3C33 3C34 3C35 3C36 3C37 3C38 3C39 3C3A 3C3B 3C3C  <RIGHT ARROW>
3C3D
FIND: 20  41 ?    <RIGHT ARROW>
3C07 3C1C
FIND:                <BREAK>
```

<SHIFT F> - Search and print results on the line printer

This command functions exactly like the <F> command, except that all entries and the results of the search are sent to the line printer instead of the video screen, and there is no pause after every 10 matches.

<Z> - Zero a block of memory, or set to a byte

This command will load the contents of every memory location between the defined limits with a specific value. After typing <Z> from the command level, you will be asked for a "Z FIRST ADDRESS:". Enter the decimal or hex address of the beginning of the block of memory you want to modify. You will then be asked for a "LAST ADDRESS:". Enter the ending address of the block of memory you want to modify. (If you enter a range that overlaps the area of memory used by DEMON, the error message "BAD LIMITS" will be given and you will have to hit <BREAK> to return to the command level.) Finally, you will be asked for a "BYTE:". This is the value you want to give every memory location within the defined range. If you hit <ENTER> without specifying a value, zero will be used as the default. After the block of memory has been filled with the defined value, you will return to the command level.

In the following example, the text that is not underlined is supplied by DEMON. The characters that are underlined are your responses:

```
Z FIRST ADDRESS: 9500   LAST ADDRESS: 9700   BYTE: 88
```

In this example, every byte between and including 9500 hex and 9700 hex will be loaded with the value 88 hex, and you will immediately return to the command level.

2.3 MONITOR INPUT/OUTPUT COMMANDS <L>, <W>, <E>

All of the following input and output functions will require a name to specify which file you will be reading or writing. Certain conventions must be observed in the use of these names. All tape operations will require a "FILE NAME". This FILE NAME must start with a letter, and have from 0 to 5 alphanumeric characters following the letter. Examples of valid FILE NAMEs for tape operations are "TEST", "G100", and "ABCDEF". Punctuation is not allowed.

Disk operations will require a "FILESPEC" (short for "file specification"). The FILESPEC for disk I/O (input/output) must meet the requirements given in your Disk Operating System Owner's Manual. Typically, this will mean a name of up to 8 characters, followed by an optional slash and three more characters. A FILESPEC may also include a drive specification by typing a colon followed by a drive number. Examples of valid FILESPECs for disk operations are "TEST", "G100", "DATA/CMD", and "ABCDEFGH/CIM:2".

<L> - Load memory from disk or tape

This command is used to load memory with a "program" which has previously been stored on tape or disk. This "program" does not have to be a real program. It can be any data that is stored on tape or disk, as long as it is stored in the proper format. If it is a tape file, it must be a "SYSTEM" recording of the type that the BASIC "SYSTEM" command can load. If it is a disk file, it must be in the format of a command file that can be loaded with the DOS command "LOAD" or executed as a program by simply typing its name from DOS READY. The DEMON command <W> will create files in this format, so anything written by DEMON can later be loaded by DEMON.

It should be mentioned that there is not a tape version of DEMON that is different from the disk version of DEMON. There is but one DEMON. If you don't have disk drives, therefore, you will not be able to use the disk commands. If in fact you try to use a disk command in a tape-based computer, you will probably crash. For this reason tape users should avoid disk commands. If you accidentally enter a disk command, you may abort it with the <BREAK> key before any damage is done. While this may be bothersome to some users, it saves you the trouble and expense of having to buy another version of DEMON when and if you finally get a disk drive. We sincerely hope this causes no inconvenience.

Loading from disk

When you first type <L> from the command level, DEMON will respond with "LOAD FROM DISK OR TAPE (D/T)?" Press <D> for disk. You will then be asked for a "FILESPEC?". You may enter any valid FILESPEC (as defined above or in your DOS manual) and hit <ENTER>. You will then be prompted to "READY DISK", which means to be sure the right disk is in the right drive. When this condition is met, hit <ENTER> again. The drives will start up and your operating system will search for the file in question. When it finds it, it will be loaded into memory and you will see an asterisk blinking in the upper right corner of the screen. When the load is complete you will be given a message that tells you the FIRST ADDRESS, the LAST ADDRESS, and the ENTRY ADDRESS of the file just loaded. Make a note of these hexadecimal numbers if needed, and hit <BREAK> to return to the command level.

If a disk error occurs during loading, it will be reported on the video screen. Disk error messages are supplied by your operating system. If you try to load a program that occupies the same memory area DEMON does, the error message "BAD ADDRESS" will be displayed along with the address of the conflict. If you try to load a file that uses memory below the limit allowed by DEMON, the same error message will be given. This limit, however, may be changed (see TECHNICAL INFORMATION). See also ERROR MESSAGES.

The following example demonstrates the use of this command. It will load DEMON itself. Because you can't load DEMON on top of itself, the version of DEMON used to load DEMON from disk must have previously been relocated to upper memory (see <R> command). In the following example, the text that is not underlined is supplied by DEMON. The underlined characters are your responses. From the command level, type the letter <L>.

```
LOAD FROM DISK OR TAPE (D/T)?  D
FILESPEC?  DEMON/CMD <ENTER>

READY DISK   <ENTER>
```

The file "DEMON/CMD" will then be loaded into memory while an asterisk flashes in the top right corner of the screen. When the load ends, the following will be displayed:

```
FIRST ADDRESS = 5F00  LAST ADDRESS = 7F68  ENTRY ADDRESS = 7ECF
```

You may then press <BREAK> to return to the command level.

Loading from tape

When you first type <L> from the command level, DEMON will respond with "LOAD FROM DISK OR TAPE (D/T)?". Type <T> for tape. If you have a MODEL III, you will then be asked "Cass?". This prompt is asking for the baud rate of the cassette tape you are using. Enter <H> for 1500 baud or <L> for 500 baud (DEMON will operate at both speeds). You will then be told to "READY CASS" which simply means to get your cassette player ready by putting in the proper tape and hitting the PLAY button. When these conditions have been met, hit <ENTER>.

As soon as the load begins, the file name will be displayed and an asterisk will begin blinking in the upper right corner of the screen. When the load ends, the FIRST, LAST and ENTRY addresses will be displayed on the screen. If an error occurs during loading, an error message will be displayed (see ERROR MESSAGES). Press <BREAK> to return to the command level.

The following example demonstrates the use of this command. It will load DEMON itself. Because you can't load DEMON on top of itself, the version of DEMON used to load DEMON from tape must have previously been relocated to upper memory (see <R> command). In the following example, the text that is not underlined is supplied by DEMON. The underlined characters are your responses. From the command level, type the letter <L>.

```
LOAD FROM DISK OR TAPE (D/T)? T
Cass? L

READY CASS   <ENTER>
```

The FILE NAME "DEMON" will appear on the screen as soon as the data begins loading, and the program will be loaded into memory while an asterisk flashes in the top right corner of the screen. When the load ends, the following will be displayed:

```
FIRST ADDRESS = 5F00  LAST ADDRESS = 7F68  ENTRY ADDRESS = 7ECF
```

You may then press <BREAK> to return to the command level.

<W> - Write from memory to disk or tape

This command is used to make recordings in the formats required by the DEMON monitor command <L>. Tape recordings will also be compatible with the BASIC "SYSTEM" command, and disk recordings will be in standard command file format which can be loaded with the DOS command LOAD or run as a program from DOS READY by simply typing the name of the file.

Up to eight separate blocks of memory may be recorded at one time with this command. The procedure is a little complex, however, so the following step-by-step instructions should be followed:

1) From the command level, press <W>. DEMON will respond with the message "WRITE TO DISK OR TAPE (D/T)?". Press <D> for disk or <T> for tape.
2) If you are writing to tape, and if you have a MODEL III, you will be asked the question "Cass?". Enter <H> for 1500 baud or <L> for 500 baud.
3) You will then be asked for a "FIRST ADDRESS:". Enter the address at which you want the recording to begin, using the conventions defined under GROUND RULES. You will then be asked for a "LAST ADDRESS:". Enter the address at which you want the recording to end. The FIRST ADDRESS and LAST ADDRESS are included in the recording, and this range is considered a "block".
4) The next message given by DEMON will be the question "MORE?". This means "Are there more blocks?", or "Do you wish to include another area of memory in this same recording?". Press <Y> to begin entry of another pair of addresses and return to step 2. Press <N> if there are no more blocks to be recorded. If more than one pair of addresses has been entered and you want to change the last one, <UP ARROW> will erase the last pair of addresses. <BREAK> will abort and return you to the command level. Because the <L> command displays the first address used by the first block and the last address used by the last block, it is helpful to enter multiple blocks in order or memory location.
5) When <N> is the response to "MORE?", you will be asked for an "ENTRY ADDRESS:". Enter the address at which you want execution of the recording to begin. If the recording is not really a program and the ENTRY ADDRESS has no meaning, you may hit <ENTER> without specifying an address, and 0 will be used as a default.
6) If you are writing to disk you will then be asked for a "FILESPEC?". If you are writing to tape you will be asked for a "FILE NAME?". Enter this name in the usual format required for a disk FILESPEC or tape FILE NAME as defined at the beginning of this section.

7) Respond to "READY DISK" or "READY CASS" by pressing <ENTER> when your disk or tape is ready to receive the recording. The appropriate device will then start up and the defined area of memory will be written to it. As the data is being written, an asterisk will blink in the top right corner of the screen.
8) If you are writing to disk, and if the FILESPEC you specified already exists, you will be given the message "REWRITE?". This is to warn you that you are about to write over an existing file. Press <Y> to continue and write over the file, or press <N> or <BREAK> to abort and return to the command level.
9) Unless an error occurs, the message "AGAIN?" will be displayed when the recording is complete. To make an identical recording on a second disk or on another tape, press <Y>, and you will be back at step 7 above.

While this procedure is difficult to describe and sounds complicated, in practice it is quite easy. The following example demonstrates the recording of a two-block file to disk. The text that is not underlined is supplied by DEMON. The underlined characters are your responses. To begin, enter the WRITE function by typing <W> from the command level.

```
WRITE TO DISK OR TAPE (D/T)?  D
FIRST ADDRESS: 5200    LAST ADDRESS: 5300
MORE? Y
FIRST ADDRESS: 6800    LAST ADDRESS: 8794
MORE? N
ENTRY ADDRESS: 6800
FILESPEC? TEST2/CMD <ENTER>

READY DISK   <ENTER>
```

<E> - Labelling disassembler

This command is used to generate a source file from a program that is already in memory. The source file may be written to tape or disk, and will be in a format that is compatible with Radio Shack's original Editor/Assembler. There are now many versions of this assembler available, and most of them will accept the same format of source code. It is, however, impossible to support every version of assembler that has been written. More details about the operation of this complex function are given in the NOTES at the end of this section.

When you select <E> from the command level, you will be given the prompt "WRITE SOURCE TO DISK OR TAPE (D/T)?". You should select <D> for disk or <T> for tape. If you are writing to tape, and if you have a MODEL III, you will then be asked "Cass?" for the baud rate. Select <H> for 1500 baud or <L> for 500 baud. You will then be asked for a "FIRST ADDRESS:". Enter the address at which you want the disassembly to begin. You will then be asked for a "LAST ADDRESS:". Enter the address at which you want disassembly to terminate. If you enter a range that overlaps DEMON, the error message "BAD LIMITS" will be given. This message will also appear if you enter a range that does not allow room for the symbol table (see NOTE 5 below). When these numbers have been successfully entered, there will be a rapid disassembly of many instructions visible on the 15th line of the screen while the symbol table is being constructed.

If you are writing to disk, you will be asked for a "FILESPEC?". If you are writing to tape, you will be asked for a "FILE NAME?". Enter this item and you will be given the message "READY DISK" or "READY CASS". You may then press <ENTER> to record the disassembled instructions. Each instruction appears briefly on the 15th line of the screen while it is being sent to the disk drive or cassette recorder.

The following example is typical. The text that is not underlined is supplied by DEMON. The underlined characters are your responses. Begin by typing <E> from the command level.

```
WRITE SOURCE TO DISK OR TAPE? (D/T)?  T
FIRST ADDRESS: 0000   LAST ADDRESS:  0100
```

(The symbol table will now be created and many instructions will appear in rapid succession on the 15th line of the screen.)

```
FILE NAME? TEST2  <ENTER>
READY CASS        <ENTER>
```

When the recording begins, each instruction will appear briefly on the 15th line of the screen. When the recording is finished you will return to the command level. As can be seen from this example, it is possible to generate source code from the TRS-80 ROMS. This is considered reverse engineering, however, and is frowned upon by Radio Shack. Programs written that use code generated in this way may be a violation of copyright law if they are offered for sale.

NOTES

1) The recording made by this command is complete, but an important step remains. All of the line numbers in the source file have been set to 00000. This is easily remedied once the file has been loaded into your assembler by simply renumbering. Your assembler manual will give you the format for this command (very likely something like "N100,10").
2) To facilitate relocation of the source, all references to addresses that are within the range being disassembled have been changed to labels. These labels are created by adding an "X" to the left of each number. In addition, each line referenced by one of these labels begins with its own address, converted to a label the same way.
3) Labels that refer to addresses that are in the middle of an instruction will lead to the error message "UNDEFINED SYMBOL" when you try to assemble the file. These will have to be corrected by editing in your assembler. As an example, let's assume that memory address 7506 is within the range being disassembled, and that it is referenced by some instruction such as "LD A,(7506H)". DEMON will convert this address to a label, and the instruction will become "LD A,(X7506H)". Let's also assume, however, that DEMON finds a 3 byte instruction at address 7505. This will place 7506 in the middle of that instruction and DEMON will be unable to place a label at address 7506. "X7506H" will be undefined because it is between instructions in the source. In this case, you would place the label "X7505H" before the instruction at address 7505 and change all occurances of label "X7506H" to "X7505H + 1". The fact that this situation occurred at all, however, may be an indication that the disassembler is out of sync and something is wrong in that area.

4) The ability to generate source code from existing object code can be very useful in making changes to existing machine language programs. You must realize, however, that there is no way a program can do this flawlessly. All it can do is try to disassemble memory and make intelligent sense of it. If the "target program" was written in a way that does not disassemble correctly (which is not only possible but likely), the resultant source file will be inaccurate. DEMON will try to make sense of the defined area of memory and will generate a source file for it. While the source file may not be perfectly accurate, DEMON was written to accomodate irregularities in a way that will produce a source file that will reassemble correctly. Once you make changes to the source file, however, you introduce the possibility of errors.

   One obvious situation of this type is the case in which a program stores variables or messages in between instructions. DEMON will try to disassemble these variables and messages into instructions, and will come up with very peculiar codes indeed. The source file will, however, reassemble correctly. But if you make changes to this source file and shift the position of these variables and messages around, problems can arise. It is, therefore, helpful to survey the area of memory you want to disassemble first to try and determine which areas are instructions and which areas are variables or messages. Be forewarned that some judgement will be required in the use of this function.

5) This function uses a two pass disassembler. On the first pass, the defined area of memory is disassembled to build a symbol table which will be used on the second pass. This table will occupy memory above the area used by DEMON. The amount of memory needed for this table depends on the amount of memory you are disassembling. In fact, it occupies exactly 1/8 the amount of memory that you are disassembling. This information may be used to determine how large the symbol table will be. If DEMON is at its original location of 5F00 to 7FFF, it will use a disk buffer from 8000 to 80FF, and the symbol table will begin at 8100. To find the last address needed by the symbol table, divide the number of bytes you are disassembling by 8 and add this number to 8100. Similar calculations can be made if DEMON has been relocated to other addresses. DEMON is smart enough, however, to know if the symbol table will crash into your source area, and will give you a "BAD LIMITS" error if the first address you enter is not high enough to allow room for the symbol table.

2.4 MONITOR RELOCATION COMMANDS <R>, <P>

<R> - Relocate DEMON

This command will allow you to move DEMON nearly anywhere in memory. DEMON may be relocated by any multiple of 100H in either direction, as often as you like, except that it won't relocate below 4500H. This limit can be removed, however (see TECHNICAL INFORMATION). Some common sense is required when using this command. For example, don't relocate into the area of memory used by the disk operating system (below 5200H for most systems) and then try to use disk I/O commands. You should also avoid trying to relocate above the end of memory in your machine (7FFF for 16K, BFFF for 32K, and FFFF for 48K).

With DEMON in its original location, the command <R> will clear the screen and display:

```
DEMON NOW:  5F00-7FFF-80FF
OK?
```

This means that DEMON now occupies memory from 5F00 to 7FFF, and it uses memory between 7FFF and 80FF for disk I/O commands. The re-entry point to DEMON at this location is 5F00. To relocate, respond to "OK?" by pressing <N> (for "NO"). You will then be given the message "ENTER BYTE:". The byte it is asking for is the "high order" byte of the new address for DEMON, or the first two digits of this new address (the last two digits are always "00"). If you want DEMON to begin at address D000, for example, type D0 and press <ENTER>. The display will then show:

```
DEMON NOW:  5F00-7FFF-80FF

ENTER BYTE: D0
RESULT:     D000-F0FF-F1FF
OK?
```

At this point, nothing has been changed. DEMON is asking you, however, if the new addresses are acceptable. If you press <BREAK>, you will return to the command level and DEMON will stay at 5F00. If you press <Y> (for "YES"), DEMON will relocate itself to D000-F0FF-F1FF and you will return to the command level (if you don't have 48K of memory, however, DEMON would crash at this address). If you press <N> (for "NO"), you will be asked to "ENTER BYTE:" again for a new address to relocate to. <R> destroys DEMON at the original location, so don't relocate DEMON and then try to jump to the original entry address.

If you intend to use DEMON for stepping through a program using E BREAKS or the <U> command, it is a good idea to make a note of the re-entry address of the version of DEMON you are working with. Press <R>, and the first address displayed is the re-entry address. You may also want to create a very small program which can be saved on disk or tape and will transfer control back to DEMON if you inadvertently lose control. This program only needs to be a jump to the entry point of DEMON. The three bytes needed to do this may be stored in a safe area of memory with the <M> command and then saved on disk or tape with the <W> command. The "program" to re-enter DEMON at its original location is "C3 00 5F" (JP 5F00).

<u><P> - User program relocation</u>

This command may be used to relocate machine language programs that are in memory. You should be warned, however, that there is no fool-proof way to do this successfully. The same problems and limitations hold true for this function as did for the <E> command (see NOTE 4 under that command). User programs are relocated by disassembling them, adjusting CALL and JUMP addresses, and then moving the code. If DEMON encounters messages or variables while it is disassembling, it will not know they are messages and variables. It will try to disassemble them, and if they disassemble to CALLs or JUMPs, they will be incorrectly modified. There are other conditions that can exist where addresses are referenced and need to be adjusted, but DEMON will not know it. For these reasons program relocation is a tricky business that may or may not be successful. As with the <E> command, some judgement in determining what areas need to be relocated and what areas do not will make this function more successful. Forewarned is forearmed.

When you select <P> from the command level, you will be asked for a "PROGRAM FIRST ADDRESS:". You should enter the starting address of the block of memory you want to relocate. You will then be asked for a "LAST ADDRESS:". Enter the last address of the block of memory you want to be relocated. You will then be asked for a "NEW FIRST ADDRESS:". Enter the address you want to move your program to. Finally, you will be asked for a "LIMIT ADDRESS:". The LIMIT ADDRESS will often be the same as the LAST ADDRESS. All references to addresses in the range FIRST ADDRESS to LIMIT ADDRESS will be adjusted for the new location. If the LIMIT ADDRESS is less than the LAST ADDRESS, the memory in between these addresses will be moved to the new location, but it will not be disassembled and no adjustments will be made for CALLs or JUMPs. This allows you to move tables or messages with the <P> command without changing their values.

The following example will demonstrate the use of the <P> command. The text that is not underlined is supplied by DEMON. The underlined characters are your responses. Let's assume you have a program that resides between 9000 hex and 9400 hex. Let's also assume that you have examined this block of memory with the <M> command, and you can see that there are a lot of messages at the end of this program, and the messages start at address 9300 hex. Finally, you want to move this program from address 9000 hex to address A000 hex. Type <P> from the command level:

PROGRAM FIRST ADDRESS: <u>9000</u> LAST ADDRESS: <u>9400</u>
NEW FIRST ADDRESS: <u>A000</u>
LIMIT ADDRESS: <u>9300</u>

After entering the last number, there will be a slight pause while the relocation takes place. You will then return to the command level. Because this command functions by adjusting the program in memory and then moving it, it will not work on ROM routines. You can accomplish the same thing, however, by using the <E> command to generate source code from ROM routines and then just changing the origin before assembly.

2.5 MONITOR "JUMP TO STEPPER" COMMANDS <S>, <C>

<S> - Transfer control to stepper

This command will transfer you to the other primary mode in DEMON, which is the STEP mode. When you press <S> from the command level, you will be given the following prompt: "S ADDRESS:". Enter the address at which you want to begin stepping. You will then be transferred to the STEP MODE. See Part 3 for Stepper Commands.

<C> - Continue stepping

If you have been in the STEP mode and end up back in the MONITOR mode for some reason, this command will resume stepping at the last instruction displayed while you were in the STEP mode. One way you might find yourself back in the MONITOR mode is by hitting <SHIFT BREAK> while in the STEP mode. This could also occur if the target program failed to return to DEMON by way of an E BREAK. In the latter case you would have to exit the target program (with RESET if necessary) and then re-enter DEMON.

When you select <C> from the command level, the screen will clear except for the letter "C" at the top left corner of the screen and the STEP display on the bottom two lines. You will then be back in the STEP mode. All breaks will be cleared.

2.6 MONITOR JUMP COMMAND <J>

<J> - Jump to an address

This command is used to jump to any address in memory. From the command level, type <J>. You will then be asked for a "J ADDRESS:". You may enter any address in ROM or RAM. You should note, however, that address 1 is reserved as a shorthand jump back to DOS in disk systems. In other words, if you specify 1 as the address to jump to, the jump will actually be to address 402DH which is a return to DOS. If you hit <ENTER> without specifying an address, the default value will be 0000. As soon as the number is entered and <ENTER> or <.> is pressed and released (see NUMBER ENTRY under GROUND RULES) the jump will be made and control will transfer to the address specified or implied. You may return to the command level at any time before this number is fully entered by hitting <BREAK>.

2.7 MONITOR LINE PRINT COMMAND <SHIFT P>

<SHIFT P> - Line printer control

This command is used to set the logical top of form or to skip a line on your line printer. When DEMON is line printing, it uses stored values (see TECHNICAL INFORMATION) to know how many lines fit on each page and how many lines to skip between each page. If you advance your printer to a new page and want to tell DEMON you have done so, this command will reset the line counter. It will not, however, advance the paper to the top of the next page automatically.

To use this command, type <SHIFT P> from the command level. You will see the letter "P" on the screen, and DEMON will only accept three commands. One of these is the letter "T". This tells DEMON to reset its counter for top of form and return to the command level. The second command DEMON will accept in this mode is the <SPACE BAR>, which will send a line feed to the line printer and advance the line counter. You may press the <SPACE BAR> as many times as you like. Finally, you may also hit <BREAK> to return to the command level.

PART 3: STEPPER COMMANDS

3.1 INTRODUCTION

The STEPPER is the second major mode of DEMON. It is entered by way of the monitor command <S> (see Section 2.5). When the step address is first entered, the top 14 lines of the screen are cleared. While DEMON is in the STEP mode, it will not use this part of the screen so you can see what your "target program" is doing. In fact, DEMON doesn't interfere with your target program's use of the bottom two lines of the screen either. When DEMON is actually stepping, it restores the full target program screen. After the step is taken, the bottom two lines are saved in memory and the stepper display returns. This happens so quickly that all you will see is a brief flicker. The stepper command <T> (see Section 3.6) can be used for a more leisurely examination of what the target program is doing on these lines.

Stepper Display Format

The STEP mode display at the bottom of your screen contains much useful information about the status of the target program and the Z-80 registers. A sample of this display is shown below. The middle two lines are what you might see on your screen. The numbers in parentheses above and below these two lines are references to the various items in the STEP mode display which will be explained below. These numbers will not appear on your screen.

```
(1)   (2)                (3)          (4)       (5)       (6)       (7)
9509  2A803C      LD     HL,(3C80)    AF=0044   BC=4444   DE=3333   HL=2222
?          - Z V -       (3C80)=2041  IX=0000   IY=0000   SP=7FFC        35
(8)         (9)            (10)        (11)      (12)      (13)         (14)
```

(1) This is the current value of the Program Counter: the address of the current instruction in the program you are stepping through. In this example, it is address 9509 hex. This number (and all numbers in the STEP mode display except the step count) is in hexadecimal notation.
(2) These are the object codes starting at this address. These three bytes form the current Z-80 instruction. To be very specific, address 9509 contains a 2A, address 950A contains an 80, and address 950B contains a 3C. Taken together, these three bytes form the instruction "LD HL,(3C80)".
(3) This is the disassembled instruction (in Zilog mnemonics) which is about to be executed.
(4) This shows the contents of the Z-80 register AF before the current instruction is executed.
(5) Same as item (4), for the BC register.
(6) Same as item (4), for the DE register.
(7) Same as item (4), for the HL register.
(8) This question mark is the command level prompt in the STEP mode. It means that DEMON is ready to accept a single character STEP mode command. If the line printer is enabled (see stepper commands <P> and <SHIFT RIGHT ARROW>) this prompt becomes an exclamation point instead of a question mark to remind you that the printer is enabled.

(9) This is the flag register display. The first hyphen holds a place for the sign flag, which is not set in the example. If the sign flag were set, an S would appear here. The Z means that the zero flag is set. If the zero flag were not set, the Z would be replaced with a hyphen. The V means the parity/overflow flag is set. If this flag were not set, the V would be replaced with a hyphen. The last hyphen holds a place for the carry flag. If the carry flag were set, the hyphen would be replaced with the letter C. It is beyond the scope of these instructions to explain the function of the Z-80 flag register. Refer to a book on assembly language programming if you are unfamiliar with the flags or any other part of the Z-80 architecture.

(10) This is what we will call the "bracket operand" display. What it means is that the number 2041 hex is stored at address 3C80. To be specific, the contents of memory location 3C80 is 41, and the contents of location 3C81 is 20. This may seem backward, but the Z-80 always stores its two-byte values backwards. The bracket operand display will be explained more fully in a moment. The reason it is given in this example is because the contents of address 3C80 are about to be loaded into the HL register pair with the current instruction.

(11) This shows the contents of the Z-80 register IX before the current instruction is executed.

(12) Same as (11), for the IY register.

(13) Same as (11), for the SP register.

(14) This is the current step count, which is one more than the number of steps that have been taken since first entering the STEP mode or resetting the step count (see command <O>). This number is the only one in the STEP mode display that is in decimal instead of hexadecimal format.

The "BRACKET OPERAND" Display

Whenever one of the operands of the current instruction is enclosed in brackets (parentheses), the numerical value of that operand is displayed below the operands (also in brackets), along with the value stored at that memory location. The following examples represent the possible configurations of this display and their meanings:

EXAMPLE 1: If the current instruction is "LD A,(4000)" and the byte at 4000H is 23H, the bracket operand display will be "(4000)=23". A single byte operation is about to be performed and a single byte value at address 4000 is shown.

EXAMPLE 2: In the case of "LD (4000),HL", a two byte transfer is indicated. The bracket operand display will show the two byte value at location 4000. The display might be "(4000)=2345". The value 2345 is stored at locations 4000 and 4001 BEFORE the instruction is executed.

EXAMPLE 3: In the case of "LD (IY+23),C", if IY = 5000 and the value stored at location 5023 is 45, the bracket operand display will be "(5023)=45". Notice that DEMON calculates and displays the offset address of "IY+23", as well as the contents of this location before the instruction is carried out. The value stored in the C register is also displayed (item (5) in our sample display), so you can see what the contents of location 5023 will be after this operation.

Step Mode Commands

There are even more commands available in the STEP mode than there are in the MONITOR mode. As in the the MONITOR mode, commands are usually entered with a single character. Some STEP mode commands (like command <M>) will use the bottom two lines of the display for their operations. To terminate these commands, hit the <BREAK> key. This will return you to the STEP mode command level and restore the STEP mode display we have just described. To exit the STEP mode entirely and return to the MONITOR mode, hit <SHIFT BREAK>.

Many of the STEP mode commands perform the same functions as their counterparts in the MONITOR mode. Some STEP mode commands, however, have the same letter codes as MONITOR commands and yet perform entirely new operations. For this reason it is a good idea to pay attention to which mode you are in when selecting commands, but DEMON is quite forgiving. Usually <BREAK> will abort an incorrect command with no harm done.

The <ENTER> key is used to step DEMON through a program one instruction at a time, following CALLS, JUMPS, and RETURNS wherever they lead. To speed things up, <B> can be used except when the displayed instruction is RET, when you may want to use <ENTER> instead to see where the program returns. For a great further increase in speed when the displayed instruction is a CALL or RST, use <U>. These commands will be explained in detail in the following sections.

General Information

When the stepper comes to a bad code, it switches to disassembling, and stops if running in one of the continuous modes. This is either an indication that something is wrong with the target program or that you have gotten out of sync with the program flow. If you want to continue anyway, press <S> to resume stepping one byte forward. In some cases this produces the same result as the Z-80 microprocessor itself, but not when H or L are involved.

The stepper keeps interrupts disabled, and ignores EI instructions. In the case of RETI and RETN instructions (return from interrupts), the stepper gives you the option of continuing with the prompt "GO ON?". If you press <Y>, these instructions are processed as though they were both RET's.

If the stepper encounters a HALT instruction it stops with the message "HALT" displayed, and will not advance if <ENTER> is pressed. Hit <BREAK> to continue. (If HALT is encountered while a program is running normally, The TRS-80 is designed to execute a jump to the reset address 66H.)

3.2 STEPPER UTILITY COMMANDS <A>, <H>, <M>, <Q>

<A> - Hexadecimal arithmetic (X+Y, X-Y, Y-X)

This command is almost identical to the monitor mode command <A> which is described in Section 2.1. In the STEP mode, however, the prompts and display will appear on the bottom line of the screen. This command may be called from the main STEP mode command level or from the STEP mode disassemble command (see command <D> below). To return to the mode you were in before calling this command, hit <BREAK>.

<H> - Hexadecimal to decimal and decimal to hexadecimal number conversion

This command is almost identical to the monitor mode command <H> which is described in Section 2.1. In the STEP mode, however, the prompts and display will appear on the bottom line of the screen. This command may be called from the main STEP mode command level or from the STEP mode disassemble command (see command <D> below). To return to the mode you were in before calling this command, hit <BREAK>.

<M> - Memory display and edit

This is a two line version of the monitor mode command <M> which is described in Section 2.1. In the STEP mode, however, the display will only occupy the bottom two lines of the screen. Because of this, a "page" is defined as two lines instead of 16. This command may be called from the main STEP mode command level or from the STEP mode disassemble command (see command <D> below). To return to the mode you were in before calling this command, hit <BREAK>.

<Q> - Quick return to memory edit

This command is identical to the monitor mode command <Q> whcih is described in Section 2.1. It will return you to the memory edit function <M> with the same display as was last used.

NOTE

When STEP mode commands <A>, <H>, <M> or <Q> are used, line printing may be done as explained for the corresponding monitor commands by hitting <SHIFT RIGHT ARROW>. If printing isn't needed, just press <BREAK> to return to whatever you were doing before you called one of these commands (stepping or disassembling). Either way, you will return with the printer disabled (with a "?" prompt instead of a "!" prompt). This is true even if the prompt was "!" with printing enabled before using these commands. This is necessary to avoid multiple printing of the same step display.

3.3 STEPPER COMMAND <D>

<D> - Disassemble

This command allows inspection of routines anywhere in memory while in the middle of stepping, without disturbing the step count, the target register contents, or the target screen. If line printing is enabled, <D> disassembles to the printer.

After pressing <D>, "D ADDRESS: " will appear on the bottom line of the screen. Enter the address at which you want disassembly to begin. When this number is entered the STEP mode display will disappear except for the first three items and the command level prompt (see the sample display in Section 3.1). To advance the disassembly by one instruction, press <ENTER>. Press <D> to change the disassembly address. Press <S> to return to stepping at the original address.

Some of the STEP mode commands are still available while you are in the disassembly mode. Commands that might interfere with stepping, however, are disabled. These include <J>, <K>, <U> and <P>. Commands that are active but should be avoided include <B>, <C>, <E>, <F>, <T>, <V>, and <Z>. For more details on these commands see their respective sections.

The disassembler ignores <E> breaks, but stops at <I>, <G> and <N> breaks. The commands that affect these breaks are still active, but any changes that are made in the disassembly mode will still be in effect when you return to stepping. For more details on the <E>, <I>, and <N> breaks see Section 3.4.

Commands <A>, <H> and <M> are still available in the disassembly mode. When you exit any of these auxiliary command routines by pressing <BREAK>, the disassembly display returns.

Disassembly to the line printer

If clearing the display screen of your target program doesn't matter, and if a series of program instructions are to be disassembled with line printing, it is easiest to use the MONITOR mode command <SHIFT D> to disassemble to the printer. To do this, press <SHIFT BREAK> to return to the MONITOR mode, use the <SHIFT D> command, and return to the STEP mode with the <C> command. If you must preserve the target program screen display, use the following procedure:

Enter the disassembly mode with stepper command <D> and enter the starting address. Next, press <SHIFT RIGHT ARROW> to enable line printing. This will cause the current instruction to be printed on the line printer. It also enables printing, which you will know by the fact that the command prompt changes from a question mark to an exclamation point. If this is all you want to print, press <BREAK>. If you want to print more disassembled instructions, press <R>. You will then be asked for an "R LAST ADDRESS:". Enter the address at which you want the disassembly to stop. Printing will begin as soon as you enter this number. It will continue until the last address is reached or exceeded unless stopped by an <I>, <G>, or <N> break.

Printing will still be enabled after the printer stops (the command prompt will still be "!"), so press <BREAK> unless you have more to print. This disables printing and restores the "?" prompt at the bottom left of the screen. To stop the printer at any time, press and hold <BREAK> until the printer stops.

3.4 BREAK SETTING COMMANDS <E>, <G>, <I>, <N>

Breakpoints are very useful functions in a debugger like DEMON. In general, a breakpoint is a condition under which program execution or single stepping will stop. DEMON has four kinds of breakpoints in addition to the "LAST ADDRESS" of the <R> command.

<E> - External break

The <E> break stands for external break. It will only function in RAM. When you select <E> from the command level, you will be given the following prompt on the bottom line of the display: "E BREAK ADDRESS:". Enter the address at which you want stepping to stop. This address should be the beginning of a known instruction, and ideally this instruction should be three or four bytes long. After this address is entered the STEP mode display will return, but a jump to DEMON will have been inserted in the target program at the break address. (This is why the E BREAK will only function in RAM. If you try to enter an E BREAK in ROM you will be given the error message "E ROM".) The three bytes that originally occupied the E BREAK address are saved within DEMON.

Next, the usual procedure is to jump into the target program by pressing <J>, and answer the "J OK?" question by pressing <Y>. At this point, the target program will resume normal operation at full speed. You will no longer be running DEMON - your target program will be in full control. If you have placed your E BREAK properly, however, the target program will eventually encounter it. What it will find, in place of the original three bytes in the target program, is a JUMP to the E BREAK entry point in DEMON. DEMON will then resume control, restore the original three bytes in the target program, and stop in the STEP mode command level at this instruction. The Z-80 register display will show the exact contents each register had at that instant in your target program. You may continue stepping through the program by pressing <ENTER>.

This kind of break is essential for debugging, but many things can go wrong, for example:

1) The target program may never encounter your E BREAK.
2) The target program might alter the codes that cause the jump to DEMON.
3) The target program might enter the middle of the E BREAK jump instruction.
4) The target program might alter DEMON before encountering the E BREAK.

Problems 1 and 3 can usually be avoided by careful placement of the E BREAK. Problems 2 and 4 are unlikely to occur, but may be avoided by knowing something about the operation of your target program. After an E BREAK is encountered, the break address is saved automatically. It will be referred to as the "old break" in the discussion which follows. The same break may be reset and used again by way of the <K> command (see Section 3.9).

Safety Features

DEMON will remove an existing E BREAK if you use the <E> command to set a new one, if you return to the monitor, or if you delete all breaks with <C>. In case you have to reset the computer and re-enter DEMON by jumping to its starting address, an E BREAK will usually be removed on reentry. In the worst case, where a break remains that DEMON doesn't expect, you will be given the error message "BAD E BREAK = xxxx" when the target program encounters the break. The xxxx will be replaced by the address of the unexpected E BREAK. You will find a jump to DEMON's E BREAK entry point at that address instead of the correct codes.

CAUTION: If you single step into an E BREAK, you will be given the error message "E-IN". This means that stepping is about to invade DEMON. If this occurs you will have to hit <BREAK> which will remove the E BREAK and return you to the MONITOR mode. You may continue stepping with the <C> command.

The following example will demonstrate the use of an E BREAK. It will use the sample program described in Appendix C. You must first either type this program into memory at address 5500, or have done so earlier and load it now from tape or disk. Enter the STEP mode with <S>. When asked for the "S ADDRESS", enter 5500. You will then be in the STEP mode, with the first instruction of the sample program displayed (CALL 0049H). Next type <E> to enter an E BREAK, and use 5511 for the E BREAK ADDRESS. Now type <J> to jump into the sample program. When asked "J OK?", type <Y>.

The screen will go blank and the sample program will wait for a keystroke with the CALL to 0049H. Type the letter "A", and the screen will immediately fill with this character. The program will then reach address 5511 and encounter the E BREAK placed there. At this point DEMON will once again regain control, and the STEP mode display will return to the bottom of the screen. The next instruction will be a JUMP to 5500 at address 5511, which is where we placed the E BREAK. While this took only a fraction of a second, the Z-80 actually executed over 20,000 instructions before we regained control. The E BREAK is very useful because it allows the target program to run at full speed until the break is encountered.

<u><G> - Break after a certain number of instructions</u>

The G BREAK is another way of stopping the target program during execution. It allows you to specify a limit to the number of instructions that will be executed in the continuous step modes (see STEP mode command <R>). Because it works in conjunction with the <R> command, the G BREAK won't work if the target program is running at full speed. Since it stores nothing in memory, however, it will work as readily in ROM routines as in RAM.

To use the G BREAK, type <G> from the STEP mode command level. You will then be asked for a "COUNT:". Enter the number of instructions you want to execute before the G BREAK is activated. This number must be a decimal number, so you must end it with a decimal point instead of the <ENTER> key. If you have forgotten how to enter decimal numbers, refer to the explanation under GROUND RULES at the very beginning of these instructions. Once this number has been entered, use the <R> command to run the target program under DEMON control. After the specified number of instructions have been executed, stepping will stop. The G BREAK, however, is still set. Each time <R> is pressed it starts a new G BREAK count, so the G BREAK can be used over and over. To clear a G BREAK, press <G> and hit <ENTER>. Any use of the <B> command will also clear the G BREAK, as will the <C> command.

To demonstrate the G BREAK, we will use the sample program described in Appendix C. You must first either type this program into memory at address 5500, or have done so earlier and load it now from tape or disk. Enter the STEP mode with <S>. When asked for the "S ADDRESS", enter 5500. You will then be in the STEP mode, with the first instruction of the sample program displayed (CALL 0049H). Type <U> to execute this ROM call at full speed. The screen will go blank, and you should type the letter "A". The step mode display will return, and the ASCII value of the letter A (41H) will be in the A register. Hit <ENTER> three times to take three steps. The step count in the bottom right corner of the screen should be 5. Next type <G> to enter a G BREAK and enter "384." (a decimal number) for the "COUNT:". You should then be back at the command level of the STEP mode. Now type <R> to run target DEMON, and when asked for an "R LAST ADDRESS:", just hit <ENTER>.

You will see the screen begin to fill with the letter "A". The step mode display will change very rapidly at the bottom of the screen as each instruction is executed. When 384 instructions have been executed, the program will stop. You will find that only the top line of the screen has been filled with the letter "A". The step counter in the bottom right corner of the screen will show 389, which is 384 more than the starting count of 5. The G BREAK is still set. To verify this, press <R> again, and hit <ENTER> when asked for an "R LAST ADDRESS". You will see the letter "A" fill the second line of the screen, and the step counter will stop at 773, which is 384 more than the last count of 389.

<I> - Internal break

This breakpoint is similar to the E BREAK, except that it works in conjunction with the <R> and <B> commands to stop when an exact address is reached in program execution. It does not allow the target program to run at full speed, but it will work in RAM or ROM. The address specified must be the beginning of a valid instruction, or DEMON will not stop.

To set an I BREAK, type <I> from the STEP mode command level. You will then be asked for an "I BREAK ADDRESS:". You may enter any valid hex or decimal number. You should already know, however, that the address you enter is the beginning of an instruction you are likely to encounter when running. After this number is entered, the STEP mode display will return. You may then begin continuous stepping of your target program with the <R> command, and when program flow reaches your I BREAK address, stepping will stop.

To demonstrate the I BREAK, we will use the sample program described in Appendix C. You must first either type this program into memory at address 5500, or have done so earlier and load it now from tape or disk. Enter the STEP mode with <S>. When asked for the "S ADDRESS", enter 5500. You will then be in the STEP mode, with the first instruction of the sample program displayed (CALL 0049H). Type <U> to execute this CALL at full speed. The screen will go blank, and you should type the letter "A". The step mode display will return and you will be at the second instruction of the sample program.

Now type <I> to enter an I BREAK, and enter 5511 for the address. You should then be back at the command level of the STEP mode. Now type <R> for continuous stepping, and just hit <ENTER> when asked for an "R LAST ADDRESS". DEMON will immediately begin stepping through the sample program. You will see the letter "A" begin to fill the screen. When the screen is completely filled the stepping will stop. The next instruction will be a JUMP to 5500 at address 5511, which is where we placed the I BREAK.

<N> - Break on numbers in given range

This command will allow you to stop continuous stepping with STEP mode commands <R> or <B> when a number within a defined range is refenced by one of the registers. Because it works in conjunction with the <R> and <B> commands, it does not allow the target program to execute at full speed, but it will work equally well in ROM routines as in RAM.

When you type <N> from the STEP mode command level, you will be asked for an "N FIRST ADDRESS:". Enter the lower limit of the range you want to specify. You will then be asked for a "LAST ADDRESS:". Enter the upper limit of the range you are interested in. You will then return to the STEP mode command level. You may now select command <R> to begin continuous stepping, and if any register or instruction references a memory address in the defined range, stepping will stop. To remove an N BREAK, simply enter zero for the two addresses requested (also see STEP mode command <C>).

To demonstrate the N BREAK, we will use the sample program described in Appendix C. You must first either type this program into memory at address 5500, or have done so earlier and load it now from tape or disk. Enter the STEP mode with <S>. When asked for the "S ADDRESS", enter 5500. You will then be in the STEP mode, with the first instruction of the sample program displayed (CALL 0049H). Type <U> to execute this CALL at full speed. The screen will go blank, and you should type the letter <A>. The step mode display will then return.

To enter the N BREAK, type <N> and enter 3D00 for the "N FIRST ADDRESS:". Enter 3D10 for the "LAST ADDRESS:". These numbers define a range of memory that is in the middle of the video screen. Type <R> to begin continuous stepping, and hit <ENTER> for "R LAST ADDRESS:". You will then see the screen begin to fill with the letter "A". If you watch the contents of the HL register, you will see it approaching the range we have defined. When it reaches 3D00, stepping will stop.

As further example, if the defined range were 4200 to 4500, stepping would stop if one of the following instructions were reached:

```
1) LD  HL,4350
2) JR 4450
3) LD  BC,(A000)  (If memory location A000 contained an address in the defined
                   range.)
```

3.5 CLEARING AND LISTING COMMANDS <C>, <L>, <O>

<C> - Clear all breaks

This command is quite simple. It just clears all breakpoints that have been set. When you type <C>, you will be asked "C OK?". If you type <N>, you will return to the command level with any breaks still intact. If you type <Y>, all breaks will be cleared. This includes E BREAK, G BREAK, I BREAK, and N BREAK. It also clears the value stored as the "old break" which is used by the STEP mode command <K>.

<L> - List breaks

This command allows you to see what values have been set for the various breakpoints. After typing <L>, you will be shown a display similar to the following:

```
E = 0000, (0000)   I = 0000   N = 0000, 0000   G = 0.
```

The first number displayed is the E BREAK address. The second number, in parentheses, is the old E BREAK address used by the STEP mode command <K>. The third number is the current I BREAK address. The next two numbers are the first and last N BREAK addresses. The last number, followed by a decimal point to indicate that it is a decimal number, is the G BREAK count. Once the break display is in place, hit <SHIFT RIGHT ARROW> to print it on the line printer. Hit <BREAK> to restore the full STEP mode display.

<O> - Reset step counter

This command very simply resets the step counter in the bottom right corner of the screen to 1. When you type <O>, you will be asked "O OK?". Type <N> and you will return to the command level with the step counter unaltered. Type <Y> and you will return to the command level with the step counter set to 1.

3.6 STEPPER DISPLAY COMMANDS <S> AND <T>

<S> - Restore STEP mode display

If the STEP mode command <D> has been used, the bottom two lines will have the disassembling mode display. To return to the STEP mode display, press <S>. You will return to the address where stepping was interrupted, and DEMON will continue just as though <D> had not been used to examine other parts of memory.

<T> - Restore target program's screen

Even though you see the STEP mode display on the bottom two lines of the screen, DEMON knows what your target program has placed on these lines. In fact, when DEMON is stepping, control of these two lines is returned to the target program. The information displayed there is saved by DEMON every time it displays its own STEP mode display. To see what your target program is doing with these two lines, hit <T>. The STEP mode display will disappear entirely so you can see the entire target screen. Hit <BREAK> to return to the STEP mode display.

3.7 CONTINUOUS STEPPING COMMANDS <R>, <B>, <V>

<R> - Run with continuous stepping

This command is used to begin continuous stepping at the current address. In this mode it is not necessary to hit <ENTER> for each new step - each step will be taken automatically until an error occurs, a breakpoint is reached, or the <BREAK> key is pressed.

To use this command, type <R> from the STEP mode command level. You will then be asked for an "R LAST ADDRESS:". This address is yet another type of breakpoint. It allows you to enter an address that is the upper limit to which you are willing to step. This address, however, does not need to be exact. Continuous stepping will stop if it reaches an address that is equal to or greater than the defined LAST ADDRESS. If you do not want to put an upper limit on continuous stepping, just hit <ENTER> for this address, and a default value of FFFF will be used.

Continuous stepping will stop if an I BREAK is reached, on the G BREAK count, and when numbers appear in the N BREAK range. If printing is enabled, every step will also be sent to the line printer. <R> resets the G BREAK counter each time it is used (see stepper command <G>).

<B> - Branch step

This command is very much like the command <R> described above. When continous stepping is requested with the <B> command, however, no LAST ADDRESS is asked for. Instead, the <B> command will step continuously until it encounters a branching instruction that branches. This may sound redundant, but all branching instructions don't branch. What we mean by a branching instruction is one that directs program flow to some new address. Examples of branching instructions are JP, CALL, RST, and RET. An example of a branching instruction that doesn't branch is "JP NZ,5200" if the "NZ" condition is not met.

This method of continuous stepping will also stop at I BREAKs and N BREAKs, if an error is encountered, or if the <BREAK> key is pressed. If printing is enabled, the display will also be sent to the line printer for every step.

<V> - Vary running speed

This command will allow you to vary the rate at which the commands <R> and <B> step. When you type <V>, the command level prompt will change to the letter "V". You may then enter any valid hexadecimal digit from 1 to F. Enter 1 for the slowest rate. Enter F (or zero) for the fastest rate. The rate of stepping approximately doubles with each increment of 1. The slowest rate displays new instructions about once each second, the fastest rate at about 90 per second. If line printing is enabled, these rates will be much slower.

3.8 REGISTER EDIT COMMANDS <F>, <Z>

<F> - Flag register edit

This command will allow you to directly change the status of the flag bits in the F register. A flag can be 0 or 1. If it is 1, or set, the flag's letter will appear in the STEP mode flag display (see the description of the STEP mode display in Section 3.1). If the flag is 0, or not set, a hyphen will appear in the flag display instead of the letter. DEMON will allow you to change the SIGN FLAG (S), the ZERO FLAG (Z), the PARITY/OVERFLOW FLAG (V), and the CARRY FLAG (C).

When you enter the command <F>, you will see a display similar to the following on the bottom line of your screen:

```
SZVC = - Z - -          FLAG?
```

You may now select the flag you want to change by typing a single letter (S, Z, P, V, or C). If you typed Z, the display would become:

```
SZVC = - Z - C          FLAG? Z =
```

You may now type a zero to reset the zero flag, or a one to set the flag. You will then be asked again for a flag to change. To escape this mode and return to the command level, press <BREAK>.

<Z> - Z-80 register display and edit

This command allows editing of all the target program's Z-80 registers. It will also allow you to examine the alternate register set. To set a register value, first type <Z>. You will then be given the following prompt on the bottom line of the screen: "Z80 REG = ". You may now enter any double register and hit <ENTER>. Select the alternate registers by adding an apostrophe after the register letters (BC becomes BC'). You will then be shown the current value of the register. To leave it alone and return to the command level, hit <BREAK>. To change the current value, enter a new number in either decimal or hex. After you have entered a new value, the cursor will disappear but nothing else will change on the screen. You may now hit <BREAK> to return to the command level, or hit <SHIFT RIGHT ARROW> to line print the bottom line of the display.

In the following example, the text that is not underlined is supplied by DEMON. The underlined characters are your responses. Enter the register mode by typing <Z>:

```
0063 20FB      JR  NZ,0060     AF=FF28 BC=00FF DE=FF00 HL=7F53
Z80 REG = BC <ENTER>  => 00FF 1100 <BREAK>
```

3.9 JUMP COMMANDS <J>, <K>, <U>

<J> - Jump to the stepping address

This command will cause DEMON to jump to the STEP mode address currently displayed on the screen. Full control will be transferred to the target program. Unless an E BREAK has been set first, there will be no return to DEMON. As a safeguard, the message "J OK?" will appear after this command is selected. Press <N> or <BREAK> to return to the STEP mode. Press <Y> to jump to the displayed address and exit DEMON.

<K> - Restore E BREAK and jump

This command resets the last E BREAK (if it isn't still set), and jumps to the target program just as the STEP mode command <J> does. After typing <K> from the command level, you will be asked "K OK?". Press <N> or <BREAK> to return to the STEP mode. Press <Y> to jump to the displayed address and exit DEMON.

CAUTION: Because the E BREAK and K BREAK replace codes in your target program, they should only be placed at three byte instructions. This is especially true of the K BREAK. If the K BREAK is used at a one byte or two byte instruction, it is possible to crash DEMON when the <K> function is used. For this reason, be sure to place it at the beginning of a three byte instruction. If this is not possible, be sure to take at least two steps with the <ENTER> key after an E BREAK or K BREAK before using the <K> function. This will allow DEMON to step past the breakpoint and prevent it from crashing itself by running into the middle of the K BREAK code.

<U> - Jump "Under" CALLS and RST's

This command allows you to execute target program routines at full program speed if the current STEP instruction is a CALL or RST in RAM. If the current instruction is not a CALL or RST, nothing will be done (not even a step). If the CALL is conditional and the condition is not met, DEMON will just step to the next instruction. If the condition is met, DEMON will execute the CALL at full speed and return. If the CALL or RST is in ROM, the error message "E ROM" will be displayed, and you will have to press <BREAK> to continue.

The <U> command clears any existing E BREAK, sets an E BREAK at the next instruction (the return address), and then jumps to the target program just like stepper command <J>. This command does not affect the old E BREAK address that is used by the <K> command. The new break is erased as soon as stepping resumes.

NOTE: Since <U> sets a break, it alters the target program and will not work properly if the target program uses the next three codes in any way before returning. Also, DEMON will not regain control if the CALL or RST does not return.

3.10 LINE PRINT COMMANDS <P>, <SHIFT RIGHT ARROW>, <SHIFT P>

<P> - Enable line printing (2 lines)

This command puts DEMON into the line printer mode. When you type <P> from the STEP mode command level, you will be asked "P OK?". If you don't really want to print, press <N> or <BREAK>. If you do want to print, press <Y>. This will cause the bottom two lines of the screen to be immediately sent to the line printer. In addition, DEMON will stay in the printer mode (the command level prompt will be "!" instead of "?") so that both lines will continue to print on every step.

<SHIFT RIGHT ARROW> - Enable line printing (line 15 only)

This command functions exactly like stepper command <P> except that no verification is asked ("P OK?" does not display) and only the top line of the two STEP mode display lines is printed.

NOTE: <SHIFT RIGHT ARROW> and <P> change the "?" prompt at bottom left corner of the screen to "!" and enable line printing so that every step will line print automatically. <BREAK> disables line printing, restoring the "?" prompt. Commands <A>, <H>, <L>, <M>, and <Z> line print independently with <SHIFT RIGHT ARROW>.

<SHIFT P> - Line printer control

This command functions identically to the monitor mode <SHIFT P> command. See Section 2.7.

3.11 TRACING WITH <B>, <U>, AND A LINE PRINTER

As has already been mentioned, <B> and <U> can be used very effectively together. Press <B> instead of <ENTER> except at a RET (where you should press <ENTER> if you want to follow the program flow) or at a CALL or RST where you may want to use <U> for full speed program execution. Remember, however, that <U> can be dangerous if you are in unknown territory. If the program crashes, reset and reload everything, and repeat the procedure using <B> instead of <U>.

When you are tracing with output to the line printer, there is an unavoidable speed penalty for printing every line. It may be useful to only line print each time the stepping stops after <B> is pressed. Do this with <P> or <SHIFT RIGHT ARROW> followed by <BREAK> (to disable further printing) each time the display stops. It is also useful to list new breakpoints with <L> and line print them for reference, and a disassembled listing of the entire program area may be very helpful.

APPENDIX A: TECHNICAL INFORMATION

In its original location, DEMON loads to 5F00-7F68, but uses memory to 7FFF. The monitor commands <E>, <L> and <W> use 8000-80FF as a disk I/O buffer. The <E> command places its symbol table above 8100. (The table is 1/8 as long as the program being disassembled.) All of these addresses relocate with DEMON so their new locations in relocated versions of DEMON will have to be calculated. Only the high bytes change, however.

The monitor command <M> will edit DEMON itself, and can be used to make useful changes. The following information refers to DEMON in its original 5F00-7FFF-80FF location:

LINE PRINTER CONTROL

Blank line count:	(79A3) = 06	(6 blank lines/page)
Printing line count:	(79A4) = 60	(60 printing lines/page)
Line feed character:	(79A6) = 0D	
Margin:	(79A7) = 00	(Set for 0 margin)

The MONITOR mode command <J> will automatically jump to address 402DH when 1 is entered. The address 402DH is stored at 734FH. If you want the <J> command to jump to some other address when 1 is entered, place that address here (low byte first).

To remove the colon after line labels in the MONITOR mode command <E> output: Change the bytes 3E 3A CD 27 76 at location 722F to 00 00 00 00 00.

DEMON has limits on the address to which programs may be loaded with the MONITOR mode command <L>. This limit is to prevent you from crashing your operating system by loading programs on top of it. This limit may, however, be changed. The 42H at 788DH prevents loading below 4300H. Change this byte to one less than the high byte of the lower limit you want to assign. If, for example, you want the lower limit to be 3C00H, change the byte at 788DH to a 3BH.

The lower limit to which DEMON itself may be relocated is stored at 5FF8H. This location currently contains 45H which prevents the <R> command from loading DEMON lower than 4500H.

Tape loads are not allowed into the disk buffer. If you don't have disks, this is a pointless limit. For tape loads only, it is possible to load into the disk I/O buffer by changing the contents of address 789BH from 3CH to 00.

To display periods for codes less than 20H when using the memory edit function <M> in ASCII mode: Change the byte at 6ECDH from 04 to 00.

TARGET STACK: The target program ~~may~~ move the target stack, but initially the region from 7FB4 to 7FFF is cleared for easy inspection, and the target stack pointer is set to 7FFC. "E STACK" will display and stepping will stop if this stack invades the region from 5F00 to 7FB3 in DEMON.

STORAGE LOCATIONS FOR TARGET REGISTERS WHILE STEPPING

AF' = (7E98)	BC' = (7E9A)	DE' = (7E9C)	HL' = (7E9E)
AF = (7EA0)	BC = (7EA2)	DE = (7EA4)	HL = (7EA6)
IX = (7EA8)	IY = (7EAA)	SP = (7EAC)	PC = (7ECD)

COMMAND INDEX

MONITOR COMMANDS
(Prompt = "CMD?")

STEPPER COMMANDS
(Prompt = "?" or "!")

COMMAND INDEX