Zapple

Monitor

Quality Software From
Computer Design Labs
342 Columbus Ave. Trenton, N.J. 08629

COMMANDS

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

A - ASSIGN reader, punch, console or list device options from the console.
B - BYE (system shut down).
C - COMPARE the contents of memory with the reader input and display any differences.
D - DISPLAY the contents of any defined memory area in Hex.
E - END OF FILE statement generator.
F - FILL any defined area of memory with a constant.
G - GOTO an address and execute. With breakpointing.
H - HEX MATH. Gives the sum and difference of two Hex numbers.
I * USER DEFINED.
J - JUSTIFY MEMORY - a non-destructive test for hard memory failures.
K * USER DEFINED.
L - LOAD a binary file.
M - MOVE a defined memory area to another starting address.
N - NULLS to the punch device.
O * USER DEFINED.
P - PUT ASCII characters into memory from the keyboard.
Q - QUERY I/O ports - may output or input any value to or from any I/O port.
R - READ a Hex file. Performs checksum, relocating, offsetting, etc.
S - SUBSTITUTE and/or examine any value at any address (in hex).
T - TYPEs the contents of a defined memory block in their ASCII equivalent.
U - UNLOAD a binary tape to the punch device.
V - VERIFY the contents of a defined memory block against that of another block and display the differences.
W - WRITE a checksummed hex file to the punch device.
X - eXAMINE and/or modify any or all registers including the special Z-80 registers.
Y - "Yis there". Search memory for defined byte strings and display all the addresses where they are found.
Z - "Z end". Locate and display the highest address in memory.

D. COMMAND SET USAGE

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

COMMAND — DESCRIPTION

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

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

The default mode for each logical device is always the teleprinter.

Assignments are made using the following format:

EXAMPLE: AC=C(cr)

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

EXAMPLE: AR=T(cr)

assigns the reader device to be the teleprinter.

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

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

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

EXAMPLE: AL=L(cr)

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

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

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

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

EXAMPLE: B

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

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

EXAMPLE: C1000,2000(cr,start reader)

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

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

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

D DISPLAY memory contents. This command displays the contents of memory in Hex. Memory is displayed

16 bytes per line, with the starting address of the line given as the first piece of data on the line.

EXAMPLE: D100,1FF(cr)

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

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

EXAMPLE: E(cr)

will generate an "end of file marker".

EXAMPLE: E100(cr)

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

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

The format for the command is:

EXAMPLE: F100,1FF,FF

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

G GOTO command. This command allows the user to cause the processor to GOTO an address and execute the program from that address. In the actual performing of the G command, a program, which has been placed in the stack area during the sign-on of the monitor, is executed. This program will first take all of the values in the register storage area (displayed with the X command), and stuff them in their correct registers in the CPU, and finally JMP to the program address being requested by the

operator. If this short program up in the stack has been destroyed (as a result of a "blow-up", or the F or M commands, etc.) the monitor will not be able to GO anywhere, and a manual restart of the monitor will be required. Whenever the monitor is restarted at the initialization point (first address I.E. 0F000H), the contents of the registers are set to ZERO with the exception of the S (stack), which contains a valid stack address. This actual value depends on the amount of memory in the system, etc. In its simplest form, the letter "G" accompanied by a parameter causes the processor to go to that address and start execution.

EXAMPLE: G1000

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

Additionally, one or two breakpoints may be set.

EXAMPLE: G1000,1005,1010

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

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

```
EXAMPLE:    *C2   JNZ   1234H
             34
             12
            *3E   MVI   A,CR
             0D
            *21   LXI   H,1000H
             00
             10
            *77   MOV   M,A
            *23   INX   H
            *CD   CALL  5678H
             78
             56
```

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

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

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

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

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

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

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

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

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

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

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

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

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

would load the tape at address 1000H. This would require that the program be an absolute program, designed for address 1000H. The start-of-file mark (automatically generated by the "U" command) is a series of 8 0FFH's (rubouts). When this is detected at the start of file, the bell will ring on the TTY to indicate the start of the load process. When the end-of-file is detected (again, a series of 8 rubouts) the load is terminated, and the address of

the NEXT location that would have been loaded is printed on the console. There are two constraints on this type of file system. The middle of the program cannot contain more than 6 0FF's (11111111) in a row (an unusual occurence), and if 0FFH is the LAST data byte in the file, it will be ignored. This too is unusual, and only a minor inconvenience.
Binary programs loaded at other than their design address will not run. The "L" command does not perform checksum functions, and cannot handle relocatable files. This is a pure and simple byte-for-byte binary loader (see "U" command).

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

EXAMPLE: M1000,1FFF,2000(cr)

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

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

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

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

It will punch 72 nulls to the punch device.

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

EXAMPLE: P1000(cr)

activates the command, and any further inputs via the keyboard would be placed into memory in their ASCII equivalent. The command is terminated by a CONTROL D character, with the address of the

location following the last entry printed on the console (the Control-D is NOT stored). Recovery of the input data is affected by use of the "T" or "U" command.

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

EXAMPLE: Q01,7(cr)

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

EXAMPLE: QI1(cr) 00001101

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

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

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

EXAMPLE: R(cr, start reader)

will load a hex file at its absolute address.

EXAMPLE: R,1000(cr,start reader)

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

EXAMPLE: R1000,100(cr,start reader)

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

EXAMPLE: R1000(cr)

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

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

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

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

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

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

EXAMPLE: SF810(sp)FF-

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

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

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

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

EXAMPLE: T1000,2000(cr)

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

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

EXAMPLE: U00,FF(cr,start reader)

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

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

EXAMPLE: V1000,2000,3000

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

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

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

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

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

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

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

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

EXAMPLE: X(cr)

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

EXAMPLE: X'(cr)

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

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

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

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

Y SEARCH. This command allows unique byte strings, from one up to 255 bytes to be searched for in

memory, and the addresses where they are found to be displayed. It is advisable to search for unique patterns rather than single bytes. The search operation may be stopped with a control-C.

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

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

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

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

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

ZAPPLE SOURCE DOCUMENTATION

ZAPPLE was assembled using CDL's Relocating Macro Assembler. In the event that you are not familiar with it's format, here is a brief description.

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

EXPERIMENTING WITH ZAPPLE

One thing that is rather nice about playing with computer programs is that you can experiment, manipulate, discect, make mistakes, 'blow them up', etc., and when the patient dies (or is "POKED TO DEATH"), he can be bought back to life by simply re-loading the program!

Please feel free to examine and modify this monitor to suit your tastes and needs. The most important thing to avoid changing however is the monitor VECTORS, and the RULES regarding them. They are:

1. Any I/O operation (CI, RI, CO, PO, etc.) should modify only the "A" register. When outputting, the character is passed in "C", and should be in "A" upon returning. When inputting, the character is returned in "A" register. *NOTE: On the "RI" Vector, the carry is normally cleared unless there is no more data to be obtained from the reader device, at which time the carry is SET to indicate an OUT OF DATA condition.

2. CSTS. This routine modifies only the contents of "A" register. It will make "A" equal to ZERO if there are no characters waiting at the assigned console input, and 0FFH if there ARE characters waiting. We are talking about the CONTENTS of "A", not the flags. The calling program would then test the contents of "A" with perhaps an "ORA A" instruction, for example, and if the result was non-zero, it would indicate a CHARACTER WAITING condition at the console keyboard.

3. IOCHK/IOSET. Allows applications software to dynamically change the I/O configuration. Any new configuration is passed in "C" reg. when IOSET is called, and the current configuration is returned in "A" reg. when IOCHK is called. *NOTE: The program in the monitor that allows modifying and assigning various I/O devices uses a R/W I/O port (one I/O port with the input tied to the output). However, the program may be modified to use a specific RAM location to store the 8-bit value. The later involves changing the IOSET/IOCHK routines accordingly. For example: "CMA, OUT 2" becomes "STA 0F8FFH", and "IN 2, CMA" becomes "LDA 0F8FFH". The use of the R/W I/O port is preffered, as it is much less sensitive to being accidentally altered during a de-bugging session, or if the program goes nuts, etc. Also, the port just above the R/W one is used (hardwired) to indicate the I/O configuration desired upon monitor initialization (may be changed to a "MVI A,XX", where XX is the desired asssignment pattern.)

This whole scheme is easily accomplished using a "3P+S" board or equivalent. (see listing for any software

details).

4. MEMCK. This routine modifies only the "A" & "B" registers. It is used to allow an applications program to find out how much memory it may use. It will load the A & B registers with the highest value of CONTINOUS memory (starting from zero) MINUS the area needed for the monitor to function properly. (A=low byte, B=high byte). This value is also placed in the STACK register when the monitor is initialized. This is then used as an initial stack value (when a "GO" command is first issued), in case the programmer has forgotten to initialize the stack. (also see "X" command).

USER WRITTEN COMMAND ROUTINES.

There are 3 command letters left open for your use. They are "I","K", & "O". Both "I" & "O" are naturals for implementing custom I/O routines. (That's what this monitor is all about.) "K" is left for your own imagination. The locations in the command table NOW contain the vector for the ERROR routine. However, in the listing, vectors to the 0F800H block are given, and should be patched to those vectors as the commands are implemented. Then, JMPs to the ACTUAL routines should be placed in the 0F800H portion. At the conclusion of the CUSTOM COMMAND, a RET instruction will return to the normal monitor command loop, printing the ">" prompt. The ideal situation, once you have settled on your own customizing of the monitor, is for the monitor to be in ROM from 0F000H to 0F7FFH (2-K ROM BOARD), and then RAM from 0F800H on upward to a maximum of 0FFFFH. (This sounds like a good use for those old 1-K static memory cards!)

USER WRITTEN I/O ROUTINES.

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

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

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

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

There are many variations of the above, and will depend on the configuration of your system, etc.

Any reasonable SPECIFIC questions regarding interfacing other devices, software, etc., which are sent to TDL, IN WRITING, will be looked at and answered within a reasonable period of time, either by return mail, or in the USER'S GROUP newsletter.

It is an almost impossible task to fully cover all of the intricate details involved in the operation of ZAPPLE. The best thing you can do now is re-read this entire manual, and then start experimenting on your own. You will have to use some common-sense if a particular subject has not been fully explained. As any lackings in this manual become evident, they WILL be covered in the NEWSLETTERS to follow. We also appreciate your feedback, and feel free to write and complain (or praise!) us about this manual or any other TDL product. YOU help US, and we'll help YOU. But most of all.....

HAVE FUN!

Roger Amidon,

Computer Design Labs
342 Columbus Ave. Trenton, N.J. 08629

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
                        ;       << ZAPPLE 2-K MASKED ROM MONITOR SYSTEM >>
                        ;                       by
                        ;
                        ;                   Roger Amidon
                        ;
                        .PABS   ;THIS MONITOR IN ABSOLUTE FORMAT
                        ;
                        ;
F000                    BASE    = 0F000H
F800                    USER    = BASE+800H
                        ;
                        ;
0038                    RST7    = 38H   ;RST 7 (LOCATION FOR TRAP)
0076                    IOBYT   = 76H   ;R/W PORT FOR TEMP. STORAGE
007A                    SENSE   = 7AH   ;SWITCH WORD FOR INITIAL DEFAULT
00FF                    SWITCH  = 0FFH  ;TEST PORT TO ABORT READ OPERATION
007A                    RCP     = 7AH   ;READER CONTROL PORT (OUT)
00F8                    NN      = 0F8H  ;"I" REGISTER INITIAL VALUE
                        ;
                        ;       <I/O DEVICES>
                        ;
                        ;-TELEPRINTER
                        ;
0071                    TTI     = 71H   ;DATA IN PORT
0071                    TTO     = 71H   ;DATA OUT PORT
0070                    TTS     = 70H   ;STATUS PORT (IN)
0001                    TTYDA   = 1     ;DATA AVAILABLE MASK BIT
0002                    TTYBE   = 02    ;XMTR BUFFER EMPTY MASK
                        ;
                        ;-C.R.T. SYSTEM
                        ;
0073                    CRTI    = 73H   ;DATA PORT (IN)
0072                    CRTS    = 72H   ;STATUS PORT (IN)
0073                    CRTO    = 73H   ;DATA PORT (OUT)
0001                    CRTDA   = 1     ;DATA AVAILABLE MASK
0002                    CRTBE   = 02    ;XMTR BUFFER EMPTY MASK
                        ;
                        ;-CASSETTE SYSTEM
                        ;
0075                    RCSD    = 75H   ;DATA IN PORT
0074                    RCSS    = 74H   ;STATUS PORT (IN)
0001                    RCSDA   = 1     ;DATA AVAILABLE MASK
0075                    PCASO   = 75H   ;DATA PORT (OUT)
0074                    PCASS   = 74H   ;CONTROL PORT (OUT)
0002                    PCSBE   = 02    ;XMTR BUFFER EMPTY MASK
                        ;
                        ;       <CONSTANTS>
                        ;
0000                    FALSE   = 0             ;ISN'T SO
FFFF                    TRUE    = # FALSE       ;IT IS SO
000D                    CR      = 0DH           ;ASCII CARRIAGE RETURN
000A                    LF      = 0AH           ;ASCII LINE FEED
0007                    BELL    = 7             ;DING
00FF                    RUB     = 0FFH          ;RUB OUT
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
0000                    FIL     = 00            ;FILL CHARACTERS AFTER CRLF
0007                    MAX     = 7             ;NUMBER OF QUES IN EOF
                        ;
                        ;       <I/O CONFIGURATION MASKS>
                        ;
00FC                    CMSK    = 11111100B     ;CONSOLE DEVICE
00F3                    RMSK    = 11110011B     ;STORAGE DEVICE (IN)
00CF                    PMSK    = 11001111B     ;STORAGE DEVICE (OUT)
003F                    LMSK    = 00111111B     ;LIST DEVICE
                        ;
                        ;
                        ;-CONSOLE CONFIGURATION
0000                    CTTY    = 0     ;TELEPRINTER
0001                    CCRT    = 1     ;C.R.T.
0002                    BATCH   = 2     ;READER FOR INPUT, LIST FOR OUTPUT
0003                    CUSE    = 3     ;USER DEFINED
                        ;
                        ;-STORAGE INPUT CONFIGURATION
0000                    RTTY    = 0     ;TELEPRINTER READER
0004                    RPTR    = 4     ;HIGH-SPEED RDR (EXTERNAL ROUTINE)
0008                    RCAS    = 8     ;CASSETTE
000C                    RUSER   = 0CH   ;USER DEFINED
                        ;
                        ;-STORAGE OUTPUT CONFIGURATION
0000                    PTTY    = 0     ;TELEPRINTER PUNCH
0010                    PPTP    = 10H   ;HIGH-SPEED PUNCH (EXTERNAL ROUTINE)
0020                    PCAS    = 20H   ;CASSETTE
0030                    PUSER   = 30H   ;USER DEFINED
                        ;
                        ;-LIST DEVICE CONFIGURATION
0000                    LTTY    = 0     ;TELEPRINTER PRINTER
0040                    LCRT    = 40H   ;C.R.T. SCREEN
0080                    LINE    = 80H   ;LINE PRINTER (EXTERNAL ROUTINE)
00C0                    LUSER   = 0C0H  ;USER DEFINED
                        ;
                        ;
                        ;       VECTORS FOR USER DEFINED ROUTINES
                        ;
F800                    .LOC    USER
F800                    CILOC:  .BLKB 3 ;CONSOLE INPUT
F803                    COLOC:  .BLKB 3 ;CONSOLE OUTPUT
F806                    RPTPL:  .BLKB 3 ;HIGH-SPEED READER
F809                    RULOC:  .BLKB 3 ;USER DEFINED STORAGE (INPUT)
F80C                    PTPL:   .BLKB 3 ;HIGH-SPEED PUNCH
F80F                    PULOC:  .BLKB 3 ;USER DEFINED STORAGE (OUTPUT)
F812                    LNLOC:  .BLKB 3 ;LINE PRINTER
F815                    LULOC:  .BLKB 3 ;USER DEFINED PRINTER
F818                    CSLOC:  .BLKB 3 ;CONSOLE INPUT STATUS ROUTINE
F81B                    J =.
                        ;
                        ;       PROGRAM CODE BEGINS HERE
                        ;
F000                    .LOC    BASE
F000    C3 F032                 JMP     BEGIN   ;GO AROUND VECTORS
```

```
                        ;
                        ;       <VECTORS FOR CALLING PROGRAMS>
                        ;
                        ; THESE VECTORS MAY BE USED BY USER WRITTEN
                        ; PROGRAMS TO SIMPLIFY THE HANDLING OF I/O
                        ; FROM SYSTEM TO SYSTEM.  WHATEVER THE CURRENT
                        ; ASSIGNED DEVICE, THESE VECTORS WILL PERFORM
                        ; THE REQUIRED I/O OPERATION, AND RETURN TO
                        ; THE CALLING PROGRAM. (RET)
                        ;
                        ; THE REGISTER CONVENTION USED FOLLOWS-
                        ;
                        ; ANY INPUT OR OUTPUT DEVICE-
                        ;       CHARACTER TO BE OUTPUT IN 'C' REGISTER.
                        ;       CHARACTER WILL BE IN 'A' REGISTER UPON
                        ;       RETURNING FROM AN INPUT OR OUTPUT.
                        ; 'CSTS'-
                        ;       RETURNS TRUE (OFFH IN 'A' REG.) IF THERE IS
                        ;       SOMETHING WAITING, AND ZERO (00) IF NOT.
                        ; 'IOCHK'-
                        ;       RETURNS WITH THE CURRENT I/O CONFIGURATION
                        ;       BYTE IN 'A' REGISTER.
                        ; 'IOSET'-
                        ;       ALLOWS A PROGRAM TO DYNAMICALLY ALTER THE
                        ;       CURRENT I/O CONFIGURATION, AND REQUIRES
                        ;       THE NEW BYTE IN 'C' REGISTER.
                        ; 'MEMCK'-
                        ;       RETURNS WITH THE HIGHEST ALLOWED USER
                        ;       MEMORY LOCATION. 'B'=HIGH BYTE, 'A'=LOW.
                        ; 'TRAP'-
                        ;       THIS IS THE 'BREAKPOINT' ENTRY POINT,
                        ;       BUT MAY BE 'CALLED'. IT WILL SAVE
                        ;       THE MACHINE STATE. RETURN CAN BE MADE WITH
                        ;       A SIMPLE 'GOCR3' ON THE CONSOLE.
                        ;
F003    C3 F619                 JMP     CI      ;CONSOLE INPUT
F006    C3 F636                 JMP     RI      ;READER INPUT
F009    C3 F48A                 JMP     CO      ;CONSOLE OUTPUT
F00C    C3 F4C4                 JMP     PO      ;PUNCH OUTPUT
F00F    C3 F4AB                 JMP     LO      ;LIST OUTPUT
F012    C3 F51A                 JMP     CSTS    ;CONSOLE STATUS
F015    DB76                    IN      IOBYT   ;I/O CHECK
F017    C9                      RET
F018    C3 F11D                 JMP     IOSET   ;I/O SET
F01B    C3 F5AC                 JMP     MEMCK   ;MEMORY LIMIT CHECK
F01E    C3 F6BE         TRAP:   JMP     RESTART ;BREAKPOINT
                        ;
                        ;       ANNOUNCEMENT OF MONITOR NAME & VERSION
                        ;
F021    0D0A000000      MSG:    .BYTE   CR,LF,FIL,FIL,FIL
F026    5A6170706C65            .ASCII  'Zapple V'
F02E    312E3052                .ASCII  '1.0R'
0011                    MSGL    = .-MSG
                        ;
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
                        ;       LET US BEGIN
                        ;
F032    3E53    BEGIN:  MVI     A,053H  ;INITIALIZE THE HARDWARE
F034    D370            OUT     TTS     Teletype
F036    D372            OUT     CRTS    CRT
F038    D374            OUT     RCSS    cassette
F03A    3E51            MVI     A,051H
F03C    D370            OUT     TTS
F03E    D372            OUT     CRTS
F040    3D              DCR     A       (A=50H)
F041    D374            OUT     RCSS
F043    AF              XRA     A       (zero Acc.)
F044    D377            OUT     IOBYT+1 - IO byte status.
F046    D37A            OUT     RCP     ;CLEAR RDR CONTROL PORT
F048    3D              DCR     A       A=FF
F049    D376            OUT     IOBYT
F04B    3E04            MVI     A,4
F04D    D377            OUT     IOBYT+1 ;WHEW!
                        ;
F04F    DB7A            IN      SENSE   ;INITIALIZE I/O CONFIGURATION
F051    D376            OUT     IOBYT
F053    3EF8            MVI     A,NN    ;INITIAL 'I' REG. CONFIGURATION
F055    ED47            STAI            ;SET FOR PAGE 'NN' ON INTERUPT
F057    31 F05B         LXI     SP,AHEAD-4      ;SET UP A FAKE STACK
F05A    C3 F5BA         JMP     MEMSIZ+1        ;GET MEMORY SIZE
F05D    F05F            .WORD   AHEAD
F05F    F9      AHEAD:  SPHL            ;SET TRUE STACK
F060    EB              XCHG
F061    01 0023         LXI     B,ENDX-EXIT
F064    21 F7A8         LXI     H,EXIT
F067    EDB0            LDIR            ;MOVE TO RAM
F069    EB              XCHG
F06A    01 FFA1         LXI     B,-5FH  ;SET UP A USER'S STACK VALUE
F06D    09              DAD     B
F06E    E5              PUSH    H       ;PRE-LOAD STACK VALUE
F06F    21 0000         LXI     H,0     ;INITIALIZE OTHER REGISTERS
F072    060A            MVI     B,10    ; (20 OF THEM)
F074    E5      STKIT:  PUSH    H       ; TO ZERO
F075    10FD            DJNZ    STKIT
F077    0611    HELLO:  MVI     B,MSGL  ;SAY HELLO TO THE FOLKS
F079    CD F44F         CALL    TOM1    ;OUTPUT SIGN-ON MSG
F07C    11 F07C START:  LXI     D,START ;MAIN 'WORK' LOOP
F07F    D5              PUSH    D       ;SET UP A RETURN TO HERE
F080    CD F512         CALL    CRLF
F083    0E3E            MVI     C,'>'
F085    CD F48A         CALL    CO
F088    CD F736 STAR0:  CALL    TI      ;GET A CONSOLE CHARACTER
F08B    E67F            ANI     7FH     ;IGNORE NULLS
F08D    28F9            JRZ     STAR0   ;GET ANOTHER
F08F    D641            SUI     'A'     ;QUALIFY THE CHARACTER
F091    F8              RM              ;<A
F092    FE1A            CPI     'Z'-'A'+1
F094    D0              RNC             ;INVALID CHARACTER
F095    87              ADD     A       ;A*2
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F096    21 F0A2             LXI     H,TBL   ;POINT TO COMMAND TABLE
F099    85                  ADD     L       ;ADD IN DISPLACEMENT
F09A    6F                  MOV     L,A
F09B    7E                  MOV     A,M
F09C    23                  INX     H
F09D    66                  MOV     H,M
F09E    6F                  MOV     L,A
F09F    0E02                MVI     C,2     ;SET C UP
F0A1    E9                  PCHL            ;GO EXECUTE COMMAND.
                    ;
                    ;               <COMMAND BRANCH TABLE>
                    ;
F0A2                TBL:
F0A2    F0D6        .WORD   ASSIGN  ;A - ASSIGN I/O
F0A4    F121        .WORD   BYE     ;B - SYSTEM SHUT-DOWN
F0A6    F14E        .WORD   COMP    ;C - COMPARE MEMORY VS. READER INPUT
F0A8    F16F        .WORD   DISP    ;D - DISPLAY MEMORY ON CONS. IN HEX
F0AA    F186        .WORD   EOF     ;E - END OF FILE TAG FOR HEX DUMPS
F0AC    F1A2        .WORD   FILL    ;F - FILL MEMORY WITH A CONSTANT
F0AE    F1AF        .WORD   GOTO    ;G - GOTO [ADDR]<,>BREAKPOINTS (2)
F0B0    F57E        .WORD   HEXN    ;H - HEX MATH. <SUM>,<DIFFERENCE>
F0B2    F81B        .WORD   J       ;I * USER DEFINED
F81E                        J=J+3   ;INCREMENT VECTOR ADDR
F0B4    F1FD        .WORD   TEST    ;J - NON-DESTRUCTIVE MEMORY TEST
F0B6    F81E        .WORD   J       ;K * USER DEFINED
F821                        J=J+3   ;INCREMENT VECTOR ADDR
F0B8    F681        .WORD   LOAD    ;L - LOAD A BINARY FORMAT FILE
F0BA    F21B        .WORD   MOVE    ;M - MOVE BLOCKS OF MEMORY
F0BC    F4F8        .WORD   NULL    ;N - PUNCH NULLS ON PUNCH DEVICE
F0BE    F821        .WORD   J       ;O * USER DEFINED
F0C0    F12F        .WORD   PUTA    ;P - 'PUT' ASCII INTO MEMORY.
F0C2    F757        .WORD   QUERY   ;Q - QI(N)=DISP. N; QO(N,V)=OUT N,V
F0C4    F226        .WORD   READ    ;R - READ A HEX FILE (W/CHECKSUMS)
F0C6    F2DF        .WORD   SUBS    ;S - SUBSTITUTE &/OR EXAMINE MEMORY
F0C8    F308        .WORD   TYPE    ;T - TYPE MEMORY IN ASCII
F0CA    F4E0        .WORD   UNLD    ;U - MEMORY TO PUNCH (BINARY FORMAT)
F0CC    F782        .WORD   VERIFY  ;V - COMPARE MEMORY AGAINST MEMORY
F0CE    F370        .WORD   WRITE   ;W - MEMORY TO PUNCH (HEX FORMAT)
F0D0    F3B0        .WORD   XAM     ;X - EXAMINE & MODIFY CPU REGISTERS
F0D2    F328        .WORD   WHERE   ;Y - FIND SEQUENCE OF BYTES IN MEM.
F0D4    F47B        .WORD   SIZE    ;Z - ADDRESS OF LAST R/W LOCATION
                    ;
                    ;
                    ;
                    ;       THIS ROUTINE CONTROLS THE CONFIGURATION
                    ; OF THE VARIOUS I/O DRIVERS & DEVICES. THIS IS
                    ; ACCOMPLISHED VIA A HARDWARE READ/WRITE PORT.
                    ;       THIS PORT IS INITIALIZED UPON SIGN-ON
                    ; BY THE VALUE READ ON PORT 'SENSE'.  IT MAY BE
                    ; DYNAMICALLY MODIFIED THROUGH CONSOLE COMMANDS.
                    ;
                    ; THE VALUE ON THE 'IOBYT' PORT REPRESENTS THE
                    ; CURRENT CONFIGURATION.  IT IS STRUCTURED THUSLY:
                    ;
```

```
                        ; 000000XX - WHERE XX REPRESENTS THE CURRENT CONSOLE.
                        ; 0000XX00 - WHERE XX REPRESENTS THE CURRENT READER.
                        ; 00XX0000 - WHERE XX REPRESENTS THE CURRENT PUNCH.
                        ; XX000000 - WHERE XX REPRESENTS THE CURRENT LISTER.
                        ;
                        ; WHEN XX = 00, THE DEVICE IS ALWAYS THE
                        ; TELEPRINTER.  WHEN XX = 11, THE DEVICE IS ALWAYS THE
                        ; USER DEFINED.  SEE OPERATORS MANUAL FOR FURTHER
                        ; DETAILS.
                        ;
F0D6    CD F736         ASSIGN: CALL    TI          ;GET DEVICE NAME
F0D9    21 F794                 LXI     H,LTBL      ;POINT TO DEVICE TABLE
F0DC    01 0400                 LXI     B,400H      ;4 DEVICES TO LOOK FOR
F0DF    11 0005                 LXI     D,5         ;IDENTIFIER + 4 DEV. IN TABLE
F0E2    BE              ..A0:   CMP     M           ;LOOK FOR MATCH
F0E3    2806                    JRZ     ..A1
F0E5    19                      DAD     D           ;GO THRU TABLE
F0E6    0C                      INR     C           ;KEEP TRACK OF DEVICE
F0E7    10F9                    DJNZ    ..A0
F0E9    1815                    JMPR    ..ERR       ;WRONG IDENTIFIER
F0EB    59              ..A1:   MOV     E,C         ;SAVE DEVICE NUMBER
F0EC    CD F736         ..A2:   CALL    TI          ;SCAN PAST '='
F0EF    FE3D                    CPI     '='
F0F1    20F9                    JRNZ    ..A2
F0F3    CD F736                 CALL    TI          ;GET NEW ASSIGNMENT
F0F6    01 0400                 LXI     B,400H      ;4 POSSIBLE ASSIGNMENTS
F0F9    23              ..A3:   INX     H           ;POINT TO ASSIGNMENT NAME
F0FA    BE                      CMP     M           ;LOOK FOR PROPER MATCH
F0FB    2806                    JRZ     ..A4        ;MATCH FOUND
F0FD    0C                      INR     C           ;KEEP TRACK OF ASSIGNMENT NMBR
F0FE    10F9                    DJNZ    ..A3
F100    C3 F464         ..ERR:  JMP     ERROR       ;NO MATCH, ERROR
F103    3E03            ..A4:   MVI     A,3         ;SET UP A MASK
F105    1C                      INR     E
F106    1D              ..A5:   DCR     E           ;DEVICE IN E
F107    2808                    JRZ     ..A6        ;GOT IT
F109    CB21                    SLAR    C           ;ELSE MOVE MASKS
F10B    CB21                    SLAR    C
F10D    17                      RAL
F10E    17                      RAL                 ;A=DEVICE MASK
F10F    18F5                    JMPR    ..A5
F111    2F              ..A6:   CMA                 ;INVERT FOR AND'ING
F112    57                      MOV     D,A         ;SAVE IN D
F113    CD F60A         ..A7:   CALL    PCHK        ;WAIT FOR [CR]
F116    30FB                    JRNC    ..A7
F118    DB76                    IN      IOBYT       ;GET PRESENT CONFIGURATION
F11A    A2                      ANA     D           ;MODIFY ONLY SELECTED DEVICE
F11B    B1                      ORA     C           ;'OR' IN NEW BIT PATTERN
F11C    4F                      MOV     C,A         ;NEW CONFIGURATION
                        ;
                        ; THIS ALLOWS USER PROGRAMS TO MODIFY
                        ; THE I/O CONFIGURATION DYNAMICALLY
                        ; DURING EXECUTION.
                        ;
```

```
F11D   79          IOSET:  MOV     A,C     ;NEW I/O BYTE PASSED IN C REG
F11E   D376                OUT     IOBYT   ;IN AN I/O PORT LATCH
F120   C9                  RET
                   ;
                   ; THIS ROUTINE IS USED AS A SIMPLE MEANS TO PREVENT
                   ; UNAUTHORIZED SYSTEM OPERATION. THE SYSTEM LOCKS UP,
                   ; MONITORING FOR A 'CONT.-SHIFT-N', AT WHICH TIME IT
                   ; WILL SIGN-ON AGAIN. NO REGISTER ASSIGNMENTS OR I/O
                   ; CONFIGURATIONS WILL BE ALTERED.
                   ;
F121   CD F512     BYE:    CALL    CRLF
F124   CD F730     ..BY:   CALL    KI
F127   FE1E                CPI     1EH     ;CONTROL-SHIFT-N
F129   20F9                JRNZ    ..BY
F12B   D1                  POP     D       ;REMOVE THE RETURN
F12C   C3 F077             JMP     HELLO   ;AND SIGN-ON AGAIN
                   ;
                   ; THIS ALLOWS ENTERING OF ASCII TEXT INTO MEMORY
                   ; FROM THE CONSOLE DEVICE. THE PARITY BIT IS CLEARED,
                   ; AND ALL WILL BE STORED EXCEPT THE BACK-ARROW [_]
                   ; WHICH DELETES THE PREVIOUS CHARACTER, AND
                   ; CONTROL-D, WHICH RETURNS CONTROL TO THE MONITOR.
                   ; THIS COMMAND, COMBINED WITH THE 'Y' COMMAND,
                   ; PROVIDES A RUDIMENTARY TEXT PROCESSING ABILITY.
                   ;
F12F   CD F540     PUTA:   CALL    EXPR1   ;GET THE STARTING ADDR.
F132   CD F512             CALL    CRLF
F135   E1                 POP     H
F136   CD F730     ..A1:   CALL    KI      ;GET A CHARACTER
F139   FE04                CPI     4       ;CONTROL-D? (EOT)
F13B   CA F482             JZ      LFADR   ;YES, STOP & PRINT ADDR.
F13E   FE5F                CPI     '_'     ;ERASE MISTAKE?
F140   2808                JRZ     ..A3    ; YES.
F142   77                  MOV     M,A     ;ELSE STORE IT IN MEMORY
F143   4F                  MOV     C,A
F144   23                  INX     H
F145   CD F48A     ..A2:   CALL    CO      ;ECHO ON CONSOLE
F148   18EC                JMPR    ..A1
F14A   2B          ..A3:   DCX     H       ;BACK UP POINTER
F14B   4E                  MOV     C,M
F14C   18F7                JMPR    ..A2    ;ECHO & CONTINUE
                   ;
                   ; THIS ROUTINE COMPARES THE READER INPUT
                   ; DEVICE WITH THE MEMORY BLOCK SPECIFIED.
                   ; IT TESTS ALL EIGHT BITS, AND ANY DISCREPENCIES
                   ; WILL BE OUTPUT TO THE CONSOLE. THIS IS USEFUL
                   ; WHEN USED WITH THE BINARY DUMP FORMAT TO BOTH
                   ; VERIFY PROPER READING & STORAGE, OR TO DETECT
                   ; PROGRAM CHANGES SINCE IT WAS LAST LOADED.
                   ;
F14E   CD F50D     COMP:   CALL    EXLF    ;GET START ' STOP ADDR.
F151   CD F474     ..C:    CALL    RIFF    ;GET A FULL READER BYTE
F154   BE                  CMP     M       ;8 BIT COMAPARE
F155   C4 F15D             CNZ     CERR    ;CALL IF INVALID COMPARE
```

```
F158  CD F56E            CALL    HILOX   ;SEE IF RANGE SATISFIED
F15B  18F4               JMPR    ..C
                  ;
                  ; THIS SUBOUTINE IS USED TO DISPLAY THE
                  ; CURRENT LOCATION OF THE 'M' REGISTER POINTERS (HL),
                  ; AND THE VALUE AT THE LOCATION, AND THE CONTENTS
                  ; OF THE ACCUMULATOR. USED BY TWO ROUTINES.
                  ;
F15D  47          CERR:  MOV     B,A     ;SAVE ACC.
F15E  CD F485            CALL    HLSP    ;DISPLAY H&L
F161  7E                 MOV     A,M
F162  CD F58F            CALL    LBYTE   ;PRINT 'M'
F165  CD F488            CALL    BLK     ;SPACE OVER
F168  78                 MOV     A,B
F169  CD F58F            CALL    LBYTE   ;PRINT ACC.
F16C  C3 F512            JMP     CRLF    ;CRLF & RETURN
                  ;
                  ; THIS DISPLAYS THE CONTENTS OF MEMORY IN BASE HEX
                  ; WITH THE STARTING LOCATION ON EACH LINE.(BETWEEN
                  ; THE TWO PARAMETERS GIVEN). 16 BYTES PER LINE MAX.
                  ;
F16F  CD F50D     DISP:  CALL    EXLF    ;GET DISPLAY RANGE
F172  CD F482     ..D0:  CALL    LFADR   ;CRLF & PRINT ADDR.
F175  CD F488     ..D1:  CALL    BLK     ;SPACE OVER
F178  7E                 MOV     A,M
F179  CD F58F            CALL    LBYTE
F17C  CD F56E            CALL    HILOX   ;RANGE CHECK
F17F  7D                 MOV     A,L
F180  E60F               ANI     0FH     ;SEE IF TIME TO CRLF
F182  20F1               JRNZ    ..D1
F184  18EC               JMPR    ..D0
                  ;
                  ; THIS OUTPUTS THE END OF FILE (EOF) PATTERN
                  ; FOR THE CHECKSUM LOADER. IT IS USED AFTER
                  ; PUNCHING A BLOCK OF MEMORY WITH THE 'W'
                  ; COMMAND.  AN ADDRESS PARAMETER MAY BE GIVEN,
                  ; AND UPON READING, THIS ADDRESS WILL BE
                  ; AUTOMATICALLY PLACED IN THE 'P' COUNTER. THE
                  ; PROGRAM CAN THEN BE RUN WITH A SIMPLE 'GCCR]'
                  ; COMMAND.
                  ;
F186  CD F540     EOF:   CALL    EXPR1   ;GET OPTIONAL ADDR.
F189  CD F4BD            CALL    PEOL    ;CRLF TO PUNCH
F18C  0E3A               MVI     C,':'   ;FILE MARKER CUE
F18E  CD F4C4            CALL    PO
F191  AF                 XRA     A       ;ZERO LENGTH
F192  CD F5EE            CALL    PBYTE
F195  E1                 POP     H
F196  CD F5E9            CALL    PADR    ;PUNCH OPTIONAL ADDR.
F199  21 0000            LXI     H,0     ;FILE TYPE=0
F19C  CD F5E9            CALL    PADR    ;PUNCH IT
F19F  C3 F4F8            JMP     NULL    ;TRAILER & RETURN
                  ;
                  ; THIS COMMAND WILL FILL A BLOCK OF MEMORY
```

```
                        ; WITH A VALUE. IE: F0,1FFF,0  FILLS FROM
                        ; <1> TO <2> WITH THE BYTE <3>. HANDY FOR
                        ; INITIALIZING A BLOCK TO A SPECIFIC VALUE, OR
                        ; MEMORY TO A CONSTANT VALUE BEFORE LOADING
                        ; A PROGRAM. (ZERO IS ESPECIALLY USEFUL.)
                        ;
F1A2    CD F535         FILL:   CALL    EXPR3   ;GET 3 PARAMETERS
F1A5    71              ..F:    MOV     M,C     ;STORE THE BYTE
F1A6    CD F574                 CALL    HILO
F1A9    30FA                    JRNC    ..F
F1AB    D1                      POP     D       ;RESTORE STACK
F1AC    C3 F07C                 JMP     START   ; IN CASE OF ACCIDENTS
                        ;
                        ; THIS COMMAND ALLOWS EXECUTION OF ANOTHER
                        ; PROGRAM WHILE RETAINING SOME MONITOR
                        ; CONTROL BY SETTING BREAKPOINTS.
                        ;
                        ; TO SIMPLY EXECUTE, TYPE 'G<ADDR>[CR]'. TO SET
                        ; A BREAKPOINT TRAP, ADD THE ADDRESS(ES) TO THE
                        ; COMMAND. IE: G<ADDR>,<BKPT>[CR]. TWO BREAKPOINTS
                        ; ARE ALLOWED, ENOUGH TO SATISFY MOST REQUIREMENTS.
                        ; ONCE A BREAKPOINT HAS BEEN REACHED, THE
                        ; REGISTERS MAY BE EXAMINED OR MODIFIED. THE
                        ; PROGRAM CAN THEN BE CONTINUED BY TYPING ONLY
                        ; A 'G[CR]'. OR ANOTHER BREAKPOINT COULD BE
                        ; IMPLEMENTED AT THAT TIME BY TYPING 'G,<BKPT>[CR]'.
                        ;
                        ; *NOTE: THIS IS SOFTWARE CONTROLLED, AND THE
                        ; BREAKPOINT MUST OCCUR ON AN INTRUCTION
                        ; BYTE.
                        ;
F1AF    CD F60A         GOTO:   CALL    PCHK    ;GET A POSSIBLE ADDRESS
F1B2    3840                    JRC     ..G3    ;CR ENTERED
F1B4    2810                    JRZ     ..G0    ;DELIMETER ENTERED
F1B6    CD F567                 CALL    EXF     ;GET ONE EXPRESSION
F1B9    D1                      POP     D
F1BA    21 0034                 LXI H,  PLOC    ;PLACE ADDRESS IN 'P' LOCATION
F1BD    39                      DAD     SP
F1BE    72                      MOV     M,D     ;HIGH BYTE
F1BF    2B                      DCX     H
F1C0    73                      MOV     M,E     ;LOW BYTE
F1C1    78                      MOV     A,B
F1C2    FE0D                    CPI     CR      ;SEE IF LAST CHARACTER WAS CR
F1C4    282E                    JRZ     ..G3    ;YES, LEAVE
F1C6    1602            ..G0:   MVI     D,2     ;TWO BREAKPOINTS MAX
F1C8    21 0035                 LXI H,  TLOC    ;POINT TO TRAP STORAGE
F1CB    39                      DAD     SP
F1CC    E5              ..G1:   PUSH    H       ;SAVE STORAGE POINTER
F1CD    CD F540                 CALL    EXPR1   ;GET A TRAP ADDRESS
F1D0    58                      MOV     E,B     ;SAVE DELIMETER
F1D1    C1                      POP     B       ;TRAP ADDR.
F1D2    E1                      POP     H       ;STORAGE
F1D3    78                      MOV     A,B     ;LOOK AT TRAP ADDR
F1D4    B1                      ORA     C
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F1D5    280A                    JRZ     ..G2    ;DON'T SET A TRAP AT 0
F1D7    71                      MOV     M,C     ;SAVE BKPT ADDR
F1D8    23                      INX     H
F1D9    70                      MOV     M,B
F1DA    23                      INX     H
F1DB    0A                      LDAX    B       ;PICK UP INST. BYTE
F1DC    77                      MOV     M,A     ;SAVE THAT TOO
F1DD    23                      INX     H
F1DE    3EFF                    MVI     A,0FFH  ;RST 7
F1E0    02                      STAX    B       ;SOFTWARE INTERUPT
F1E1    7B              ..G2:   MOV     A,E     ;LOOK AT DELIMITER
F1E2    FE0D                    CPI     CR
F1E4    2803                    JRZ     ..G2A
F1E6    15                      DCR     D       ;COUNT BKPTS
F1E7    20E3                    JRNZ    ..G1    ;GET ONE MORE
F1E9    3EC3            ..G2A:  MVI     A,JMP   ;SET UP JMP INSTRUCTION
F1EB    32 0038                 STA     RST7    ; AT RESTART TRAP LOC.
F1EE    21 F01E                 LXI     H,TRAP  ; TO MONITOR VECTOR
F1F1    22 0039                 SHLD    RST7+1
F1F4    CD F512         ..G3:   CALL    CRLF
F1F7    D1                      POP     D       ;CLEAR SYSTEM RETURN
F1F8    21 0016                 LXI     H,22    ;FIND 'EXIT' ROUTINE
F1FB    39                      DAD     SP      ;UP IN STACK
F1FC    E9                      PCHL            ;GOOD LUCK.
                        ;
                        ; THIS IS A 'QUICKIE' MEMORY TEST TO SPOT
                        ; HARD MEMORY FAILURES, OR ACCIDENTLY
                        ; PROTECTED MEMORY LOCATIONS. IT IS NOT
                        ; MEANT TO BE THE DEFINITIVE MEMORY DIAGNOSTIC.
                        ; IT IS, HOWEVER, NON-DESTRUCTIVE. ERRORS ARE
                        ; PRINTED ON THE CONSOLE AS FOLLOWS-
                        ; <ADDR> 00000100 WHERE <1> IS THE BAD BIT.
                        ; BIT LOCATION OF THE FAILURE IS EASILY
                        ; DETERMINED. NON-R/W MEMORY WILL RETURN
                        ; WITH-  11111111
                        ;
F1FD    CD F50D         TEST:   CALL    EXLF    ;GET TWO PARAMS
F200    7E              ..T1:   MOV     A,M     ;READ A BYTE
F201    47                      MOV     B,A     ;SAVE IN B REG.
F202    2F                      CMA
F203    77                      MOV     M,A     ;READ/COMPLIMENT/WRITE
F204    AE                      XRA     M       ; & COMPARE
F205    280E                    JRZ     ..T2    ;SKIP IF ZERO (OK)
F207    D5                      PUSH    D       ;SAVE END POINTER
F208    50                      MOV     D,B     ;SAVE BYTE
F209    5F                      MOV     E,A     ;SET-UP TO DISPLAY
F20A    CD F485                 CALL    HLSP    ;PRINT BAD ADDR
F20D    CD F769                 CALL    BITS    ;PRINT BAD BIT LOC.
F210    CD F512                 CALL    CRLF
F213    42                      MOV     B,D     ;RESTORE BYTE
F214    D1                      POP     D       ;RESTORE DE
F215    70              ..T2:   MOV     M,B     ;REPLACE BYTE
F216    CD F56E                 CALL    HILOX   ;RANGE TEST
```

```
F219    18E5                    JMPR    ..T1
                        ;
                        ; THIS COMMAND MOVES MASS AMOUNTS OF MEMORY
                        ; FROM <1> THRU <2> TO THE ADDRESS STARTING
                        ; AT <3>.  THIS ROUTINE SHOULD BE USED WITH
                        ; SOME CAUTION, AS IT COULD SMASH MEMORY IF
                        ; CARELESSLY IMPLEMENTED.
                        ;
                        ;       M<1>,<2>,<3>
                        ;
F21B    CD F535         MOVE:   CALL    EXPR3   ;GET 3 PARAMETERS
F21E    7E              ..M:    MOV     A,M     ;PICK UP
F21F    02                      STAX    B       ;PUT DOWN
F220    03                      INX     B       ;MOVE UP
F221    CD F56E                 CALL    HILOX   ;CHECK IF DONE
F224    18F8                    JMPR    ..M
                        ;
                        ; THIS COMMAND READS THE CHECK-SUMMED HEX FILES
                        ; FOR BOTH THE NORMAL INTEL FORMAT AND THE TDL/CDL
                        ; RELOCATING FORMAT. ON BOTH FILES, A 'BIAS' MAY
                        ; BE ADDED, WHICH WILL CAUSE THE OBJECT CODE TO
                        ; BE PLACED IN A LOCATION OTHER THAN ITS
                        ; INTENDED EXECUTION LOCATION. THE BIAS IS ADDED TO
                        ; WHAT WOULD HAVE BEEN THE NORMAL LOADING
                        ; LOCATION, AND WILL WRAP AROUND TO ENABLE
                        ; LOADING ANY PROGRAM ANYWHERE IN MEMORY.
                        ;
                        ; WHEN LOADING A RELOCATABLE FILE, AN ADDITIONAL
                        ; PARAMETER MAY BE ADDED, WHICH REPRESENTS THE
                        ; ACTUAL EXECUTION ADDRESS DESIRED. THIS ALSO MAY
                        ; BE ANY LOCATION IN MEMORY.
                        ;
                        ; EXAMPLES:
                        ;
                        ; R[CR] =0 BIAS, 0 EXECUTION ADDR.
                        ; R<ADDR1>[CR] =<1>BIAS, 0 EXECUTION ADDR.
                        ; R,<ADDR1>[CR] =0 BIAS, <1> EXECUTION ADDR.
                        ; R<ADDR1>,<ADDR2>[CR] =<1>BIAS, <2> EXECUTION ADDR.
                        ;
F226    CD F540         READ:   CALL    EXPR1   ;GET BIAS, IF ANY
F229    78                      MOV     A,B     ;LOOK AT DELIMITER
F22A    D60D                    SUI     CR      ;ALL DONE?
F22C    47                      MOV     B,A     ;SET UP RELOCATION OF 0
F22D    4F                      MOV     C,A     ; IF CR ENTERED
F22E    D1                      POP     D       ;BIAS AMOUNT
F22F    2804                    JRZ     ..R0    ;CR ENTERED
F231    CD F540                 CALL    EXPR1   ;GET RELOCATION
F234    C1                      POP     B       ;ACTUAL RELOCATION VALUE
F235    EB              ..R0:   XCHG
F236    D9                      EXX             ;HL'=BIAS, BC'=RELOCATION
F237    CD F512                 CALL    CRLF
F23A    CD F67B         LOD0:   CALL    RIX     ;GET A CHARACTER
F23D    D63A                    SUI     ':'     ;ABSOLUTE FILE CUE?
F23F    47                      MOV     B,A     ;SAVE CUE CLUE
```

```
F240    E6FE                    ANI     0FEH    ;KILL BIT 0
F242    20F6                    JRNZ    LOD0    ; NO, KEEP LOOKING
F244    57                      MOV     D,A     ;ZERO CHECKSUM
F245    CD F2C0                 CALL    SBYTE   ;GET FILE LENGTH
F248    5F                      MOV     E,A     ;SAVE IN E REG.
F249    CD F2C0                 CALL    SBYTE   ;GET LOAD MSB
F24C    F5                      PUSH    PSW     ;SAVE IT
F24D    CD F2C0                 CALL    SBYTE   ;GET LOAD LSB
F250    D9                      EXX             ;CHANGE GEARS
F251    D1                      POP     D       ;RECOVER MSB
F252    5F                      MOV     E,A     ;FULL LOAD ADDR
F253    C5                      PUSH    B       ;BC'=RELOCATION
F254    D5                      PUSH    D       ;DE'=LOAD ADDR
F255    E5                      PUSH    H       ; HL'=BIAS
F256    19                      DAD     D       ; BIAS+LOAD
F257    E3                      XTHL            ;RESTORE HL'
F258    DDE1                    POP     X       ; X=BIAS+LOAD
F25A    D9                      EXX             ;DOWNSHIFT
F25B    E1                      POP     H       ;HL=LOAD ADDR
F25C    CD F2C0                 CALL    SBYTE   ;GET FILE TYPE
F25F    3D                      DCR     A       ;1=REL. FILE, 0=ABS.
F260    78                      MOV     A,B     ;SAVE CUE BIT
F261    C1                      POP     B       ;BC=RELOCATION
F262    2003                    JRNZ    ..A     ;ABSOLUTE FILE
F264    09                      DAD     B       ;ELSE RELOCATE
F265    DD09                    DADX    B       ;BOTH X & HL
F267    1C              ..A:    INR     E       ;TEST LENGTH
F268    1D                      DCR     E       ;0=DONE
F269    2819                    JRZ     DONE
F26B    3D                      DCR     A       ;TEST CUE
F26C    2822                    JRZ     LODR    ;RELATIVE
F26E    CD F2C0         ..L1:   CALL    SBYTE   ;NEXT
F271    CD F2D3                 CALL    STORE   ;STORE IT
F274    20F8                    JRNZ    ..L1    ;MORE COMING
F276    CD F2C0         LOD4:   CALL    SBYTE   ;GET CHECKSUM
F279    28BF                    JRZ     LOD0    ;GOOD CHECKSUM
F27B    DDE5            ERR2:   PUSH    X
F27D    E1                      POP     H       ;TRANSFER
F27E    CD F58A                 CALL    LADR    ;PRINT CURRENT LOAD ADDR
F281    C3 F464                 JMP     ERROR   ;ABORT
F284    7C              DONE:   MOV     A,H
F285    B5                      ORA     L       ;DON'T MODIFY IF ZERO
F286    C8                      RZ
F287    EB                      XCHG            ;STORE PC
F288    21 0034                 LXI H,  PLOC
F28B    39                      DAD     SP
F28C    72                      MOV     M,D     ;IN STACK AREA
F28D    2B                      DCX     H
F28E    73                      MOV     M,E
F28F    C9                      RET
F290    2E01            LODR:   MVI     L,1     ;SET-UP BIT COUNTER
F292    CD F2B0         ..L1:   CALL    LODCB   ;GET THE BIT
F295    3807                    JRC     ..L3    ;DOUBLE BIT
F297    CD F2D3         ..L5:   CALL    STORE   ;WRITE IT
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F29A    20F6                    JRNZ    ..L1
F29C    18D8                    JMPR    LOD4        ;TEST CHECKSUM
F29E    4F              ..L3:   MOV     C,A         ;SAVE LOW BYTE
F29F    CD F2B0                 CALL    LODCB       ;NEXT CONTROL BIT
F2A2    47                      MOV     B,A         ;SAVE HIGH BYTE
F2A3    D9                      EXX
F2A4    C5                      PUSH    B           ;GET RELOCATION
F2A5    D9                      EXX
F2A6    E3                      XTHL                ;INTO HL
F2A7    09                      DAD     B           ;RELOCATE
F2A8    7D                      MOV     A,L         ;LOW BYTE
F2A9    CD F2D3                 CALL    STORE       ;STORE IT
F2AC    7C                      MOV     A,H         ;HIGH BYTE
F2AD    E1                      POP     H           ;RESTORE HL
F2AE    18E7                    JMPR    ..L5        ;DO THIS AGAIN
F2B0    2D              LODCB:  DCR     L           ;COUNT BITS
F2B1    2007                    JRNZ    ..LC1       ;MORE LEFT
F2B3    CD F2C0                 CALL    SBYTE       ;GET NEXT
F2B6    1D                      DCR     E           ;COUNT BYTES
F2B7    67                      MOV     H,A         ;SAVE THE BITS
F2B8    2E08                    MVI     L,8         ;8 BITS/BYTE
F2BA    CD F2C0         ..LC1:  CALL    SBYTE       ;GET A DATA BYTE
F2BD    CB24                    SLAR    H           ;TEST NEXT BIT
F2BF    C9                      RET
F2C0    C5              SBYTE:  PUSH    B           ;PRESERVE BC
F2C1    CD F5D6                 CALL    RIBBLE      ;GET A CONVERTED ASCII CHAR.
F2C4    07                      RLC
F2C5    07                      RLC
F2C6    07                      RLC
F2C7    07                      RLC                 ;MOVE IT TO HIGH NIBBLE
F2C8    4F                      MOV     C,A         ;SAVE IT
F2C9    CD F5D6                 CALL    RIBBLE      ;GET OTHER HALF
F2CC    B1                      ORA     C           ;MAKE WHOLE
F2CD    4F                      MOV     C,A         ;SAVE AGAIN IN C
F2CE    82                      ADD     D           ;UPDATE CHECKSUM
F2CF    57                      MOV     D,A         ;NEW CHECKSUM
F2D0    79                      MOV     A,C         ;CONVERTED BYTE
F2D1    C1                      POP     B
F2D2    C9                      RET
F2D3    DD7700          STORE:  MOV     0(X),A      ;WRITE TO MEMORY
F2D6    DDBE00                  CMP     0(X)        ;VALID WRITE?
F2D9    20A0                    JRNZ    ERR2        ; NO.
F2DB    DD23                    INX     X           ;ADVANCE POINTER
F2DD    1D                      DCR     E           ;COUNT DOWN
F2DE    C9                      RET

                        ;
                        ; THIS ROUTINE ALLOWS BOTH INSPECTION OF &
                        ; MODIFICATION OF MEMORY ON A BYTE BY BYTE
                        ; BASIS. IT TAKES ONE ADDRESS PARAMETER,
                        ; FOLLOWED BY A SPACE.  THE DATA AT THAT
                        ; LOCATION WILL BE DISPLAYED. IF IT IS
                        ; DESIRED TO CHANGE IT, THE VALUE IS THEN
                        ; ENTERED.  A FOLLOWING SPACE WILL DISPLAY
                        ; THE NEXT BYTE.  A CARRIAGE RETURN [CR]
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
                        ; WILL TERMINATE THE COMMAND.  THE SYSTEM
                        ; ADDS A CRLF AT LOCATIONS ENDING WITH EITHER
                        ; XXX0 OR XXX8. TO AID IN DETERMINING THE
                        ; PRESENT ADDRESS, IT IS PRINTED AFTER
                        ; EACH CRLF.  A BACKARROW [_] WILL BACK
                        ; UP THE POINTER AND DISPLAY THE
                        ; PREVIOUS LOCATION.
                        ;
F2DF    CD F540         SUBS:   CALL    EXPR1   ;GET STARTING ADDR.
F2E2    E1                      POP     H
F2E3    7E              ..S0:   MOV     A,M
F2E4    CD F58F                 CALL    LBYTE   ;DISPLAY THE BYTE
F2E7    CD F605                 CALL    COPCK   ;MODIFY?
F2EA    D8                      RC              ; NO, ALL DONE
F2EB    280F                    JRZ     ..S1    ;DON'T MODIFY
F2ED    FE5F                    CPI     '_'     ;BACKUP?
F2EF    2814                    JRZ     ..S2
F2F1    E5                      PUSH    H       ;SAVE POINTER
F2F2    CD F567                 CALL    EXF     ;GET NEW VALUE
F2F5    D1                      POP     D       ;VALUE IN E
F2F6    E1                      POP     H
F2F7    73                      MOV     M,E     ;MODIFY
F2F8    78                      MOV     A,B     ;TEST DELIMITER
F2F9    FE0D                    CPI     CR
F2FB    C8                      RZ              ;DONE
F2FC    23              ..S1:   INX     H
F2FD    7D              ..S3:   MOV     A,L     ;SEE IF TIME TO CRLF
F2FE    E607                    ANI     7
F300    CC F482                 CZ      LFADR   ;TIME TO CRLF
F303    18DE                    JMPR    ..S0
F305    2B              ..S2:   DCX     H       ;DECREMENT POINTER
F306    18F5                    JMPR    ..S3    ;AND PRINT DATA THERE.
                        ;
                        ; THIS ROUTINE TRANSLATES THE DATA IN
                        ; MEMORY TO AN ASCII FORMAT.  ALL NON-
                        ; PRINTING CHARACTERS ARE CONVERTED TO
                        ; PERIODS. [.]
                        ; THERE ARE 64 CHARACTERS PER LINE.
                        ;
F308    CD F50D         TYPE:   CALL    EXLF    ;DISPLAY RANGE
F30B    CD F482         ..T0:   CALL    LFADR   ;DISPLAY ADDRESS
F30E    0640                    MVI     B,64    ;CHARACTERS PER LINE
F310    7E              ..T1:   MOV     A,M
F311    E67F                    ANI     7FH     ;KILL PARITY BIT
F313    FE20                    CPI     ' '     ;RANGE TEST
F315    3002                    JRNC    ..T3    ;=>SPACE
F317    3E2E            ..T2:   MVI     A,'.'   ;REPLACE NON-PRINTING
F319    FE7C            ..T3:   CPI     07CH    ;ABOVE LOWER CASE z
F31B    30FA                    JRNC    ..T2
F31D    4F                      MOV     C,A     ;SEND IT
F31E    CD F48A                 CALL    CO
F321    CD F56E                 CALL    HILOX   ;MORE TO GO?
F324    10EA                    DJNZ    ..T1    ;SEE IF TIME TO CRLF
F326    18E3                    JMPR    ..T0    ; YES.
```

```
                        ;
                        ; THIS IS A HEXADECIMAL SEARCH ROUTINE. IT
                        ; TAKES NO ADDRESS PARAMETERS. AS MANY
                        ; BYTES MAY BE ENTERED, SEPARATED BY A COMMA,
                        ; AS DESIRED.  THE MAXIMUM IS 255, BUT 3-4 IS
                        ; TYPICAL, AND MORE THAN 12 WOULD BE UNUSUAL.
                        ; THE ENTIRE MEMORY IS SEARCHED, STARTING
                        ; FROM ZERO, AND ALL STARTING ADDRESSES OF EACH
                        ; OCCURENCE OF THE REQUESTED STRING ARE PRINTED
                        ; ON THE CONSOLE DEVICE.
                        ;
F328    1600            WHERE:  MVI     D,0     ;COUNT SEARCH BYTES
F32A    CD F540         ..W0:   CALL    EXPR1   ;GET ONE BYTE
F32D    E1                      POP     H       ;PICK IT UP
F32E    65                      MOV     H,L     ;STICK IN HIGH BYTE
F32F    E5                      PUSH    H       ;PUT IT IN STACK
F330    33                      INX     SP      ;ADJUST STACK
F331    14                      INR     D       ;COUNT UP
F332    78                      MOV     A,B     ;TEST DELIMITER
F333    D60D                    SUI     CR
F335    20F3                    JRNZ    ..W0    ;MORE TO GO
F337    47                      MOV     B,A     ;CHEAP ZEROES
F338    4F                      MOV     C,A
F339    67                      MOV     H,A
F33A    6A                      MOV     L,D     ;GET BYTE COUNT IN L
F33B    2D                      DCR     L       ;-1
F33C    39                      DAD     SP      ;BYTES STORED IN STACK
F33D    E5                      PUSH    H
F33E    C5                      PUSH    B
F33F    C5              FINDC:  PUSH    B       ;SAVE THAT POINTER
F340    CD F512                 CALL    CRLF
F343    C1                      POP     B       ;RESTORE
F344    E1              FIND:   POP     H       ;HL=SEARCH ADDR
F345    DDE1                    POP     X       ;X=SEARCH BYTE POINTER
F347    5A                      MOV     E,D     ;RESET COUNT
F348    DD7E00                  MOV     A,0(X)  ;GET THE FIRST SEARCH BYTE
F34B    EDB1                    CCIR            ;COMPARE, INCR., & REPEAT
F34D    E2 F36B                 JPO     DONE2   ;ODD PARITY=DONE
F350    DDE5                    PUSH    X       ;SAVE POINTERS
F352    E5                      PUSH    H
F353    1D              FOUND:  DCR     E
F354    280B                    JRZ     TELL    ;FOUND ALL
F356    DD7EFF                  MOV     A,-1(X) ;LOOK AT NEXT MATCH
F359    BE                      CMP     M       ;TEST NEXT
F35A    20E8                    JRNZ    FIND    ;NO MATCH
F35C    23                      INX     H       ;BUMP POINTERS
F35D    DD2B                    DCX     X
F35F    18F2                    JMPR    FOUND   ;TEST NEXT MATCH
F361    E1              TELL:   POP     H
F362    E5                      PUSH    H
F363    2B                      DCX     H
F364    C5                      PUSH    B       ;SAVE SEARCH COUNT LIMIT
F365    CD F58A                 CALL    LADR    ;TELL CONSOLE
F368    C1                      POP     B       ;RESTORE
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F369  18D4              JMPR    FINDC
F36B  33          DONE2:INX     SP
F36C  1D                DCR     E       ;RESET STACK
F36D  20FC              JRNZ    DONE2
F36F  C9                RET
                        ;
                        ; THIS ROUTINE DUMPS MEMORY IN THE STANDARD
                        ; INTEL HEX-FILE FORMAT.  A START & END
                        ; PARAMETER IS REQUIRED. AT THE CONCLUSION
                        ; OF THE DUMP, AN "END OF FILE" SHOULD BE
                        ; GENERATED WITH THE "E" COMMAND.
                        ;
F370  CD F50D     WRITE:CALL    EXLP    ;GET TWO PARAMETERS
F373  CD F4FB           CALL    WAIT    ;PAUSE IF TTY CONFIGURATION
F376  CD F4BD     ..W0: CALL    PEOL    ;CRLF TO PUNCH
F379  01 003A           LXI     B,':'   ;START-OF-FILE CUE
F37C  CD F4C4           CALL    PO      ;PUNCH IT
F37F  D5                PUSH    D       ;SAVE
F380  E5                PUSH    H       ; POINTERS
F381  04          ..W1: INR     B       ;CALCULATE FILE LENGTH
F382  CD F574           CALL    HILO
F385  3824              JRC     ..W4    ;SHORT FILE
F387  3E18              MVI     A,24    ;24 BYTES PER FILE
F389  90                SUB     B       ;ENOUGH YET?
F38A  20F5              JRNZ    ..W1    ; NO.
F38C  E1                POP     H       ;GET START ADDR BACK.
F38D  CD F393           CALL    ..W2    ;SEND THE BLOCK
F390  D1                POP     D       ;RESTORE END OF FILE POINTER
F391  18E3              JMPR    ..W0    ;KEEP GOING
F393  57          ..W2: MOV     D,A     ;INITIALIZE CHECKSUM
F394  78                MOV     A,B     ;FILE LENGTH
F395  CD F5EE           CALL    PBYTE   ;PUNCH IT
F398  CD F5E9           CALL    PADR    ;PUNCH ADDRESS
F39B  AF                XRA     A       ;FILE TYPE=0
F39C  CD F5EE           CALL    PBYTE   ;PUNCH IT
F39F  7E          ..W3: MOV     A,M     ;GET A DATA BYTE
F3A0  CD F5EE           CALL    PBYTE   ;PUNCH IT
F3A3  23                INX     H       ;POINT TO NEXT BYTE
F3A4  10F9              DJNZ    ..W3    ;DECREMENT FILE COUNT
F3A6  AF                XRA     A
F3A7  92                SUB     D       ;CALCULATE CHECKSUM
F3A8  C3 F5EE           JMP     PBYTE   ;PUNCH IT, RETURN
F3AB  E1          ..W4: POP     H       ;CLEAR STACK
F3AC  D1                POP     D       ; OF POINTERS
F3AD  AF                XRA     A       ;SET-UP A
F3AE  18E3              JMPR    ..W2    ;FINISH UP & RETURN
                        ;
                        ;
                        ;       THIS ROUTINE ALLOWS DISPLAYING THE
                        ; USER'S CPU REGISTERS. THEY ALSO MAY BE
                        ; USING THE REGISTER NAME AFTER TYPING THE "X".
                        ; I.E.  XA 00-
                        ; THE REGISTER MAY BE SKIPPED OVER, OR MODIFIED,
                        ; SIMILARLY TO THE "S" COMMAND.
```

```
                        ;
                        ; TO DISPLAY THE "NORMAL" SYSTEM STATUS,
                        ; SIMPLY TYPE "X[CR]".  TO DISPLAY THE
                        ; ADDITIONAL Z-80 REGISTERS, FOLLOW
                        ; THE "X" WITH AN APOSTROPHE.  I.E. "X'[CR]",
                        ; OR TO EXAMINE A SINGLE "PRIME" REGISTER,
                        ; TYPE THE REGISTER IDENTIFIER AFTER THE
                        ; APOSTROPHE.  I.E.  X'X 0000-
                        ;
                        ; THESE REGISTER VALUES ARE PLACED INTO THE CPU
                        ; UPON EXECUTING ANY "GO" COMMAND. [G]
                        ;
F3B0    CD F736         XAM:    CALL    TI
F3B3    21 F7CB                 LXI     H,ACTBL
F3B6    FE0D                    CPI     CR          ;FULL REG. DISPLAY
F3B8    285A                    JRZ     ..X6
F3BA    FE27                    CPI     "'"         ;SEE IF PRIMES WANTED
F3BC    200A                    JRNZ    ..X0
F3BE    21 F7E7                 LXI     H,PRMTB
F3C1    CD F736                 CALL    TI
F3C4    FE0D                    CPI     CR          ;FULL REG. DISPLAY
F3C6    284C                    JRZ     ..X6
F3C8    BE              ..X0:   CMP     M           ;TEST FOR REGISTER NAME
F3C9    2809                    JRZ     ..X1
F3CB    CB7E                    BIT     7,M         ;SEE IF END OF TABLE
F3CD    C2 F464                 JNZ     ERROR
F3D0    23                      INX     H
F3D1    23                      INX     H
F3D2    18F4                    JMPR    ..X0
F3D4    CD F488         ..X1:   CALL    BLK
F3D7    23              ..X2:   INX     H
F3D8    7E                      MOV     A,M
F3D9    47                      MOV     B,A         ;SAVE FOR FLAGS
F3DA    E63F                    ANI     3FH         ;CLEAR FLAGS FOR BIAS
F3DC    EB                      XCHG
F3DD    6F                      MOV     L,A         ;DISPLACEMENT FROM STACK
F3DE    2600                    MVI     H,0
F3E0    39                      DAD     SP
F3E1    EB                      XCHG
F3E2    23                      INX     H
F3E3    1A                      LDAX    D           ;PICK UP REG. VALUE
F3E4    CD F58F                 CALL    LBYTE       ;PRINT IT
F3E7    CB78                    BIT     7,B
F3E9    2805                    JRZ     ..X3
F3EB    1B                      DCX     D
F3EC    1A                      LDAX    D
F3ED    CD F58F                 CALL    LBYTE
F3F0    CD F605         ..X3:   CALL    COPCK       ;ASK CONSOLE
F3F3    D8                      RC                  ;CR ENTERED, ALL DONE
F3F4    2819                    JRZ     ..X5        ;SKIP TO NEXT REG.
F3F6    E5                      PUSH    H
F3F7    C5                      PUSH    B
F3F8    CD F567                 CALL    EXF         ;GET NEW VALUE
F3FB    E1                      POP     H
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F3FC   F1                  POP     PSW
F3FD   C5                  PUSH    B
F3FE   F5                  PUSH    PSW
F3FF   7D                  MOV     A,L
F400   12                  STAX    D
F401   C1                  POP     B
F402   CB78                BIT     7,B       ;SEE IF 8 BIT OR 16 BIT REG.
F404   2803                JRZ     ..X4      ;8 BIT
F406   13                  INX     D
F407   7C                  MOV     A,H       ;HIGH BYTE OF 16 BIT REG.
F408   12                  STAX    D
F409   C1          ..X4:   POP     B
F40A   E1                  POP     H
F40B   78                  MOV     A,B       ;TEST KEYBOARD
F40C   FE0D                CPI     CR
F40E   C8                  RZ                ;ALL DONE
F40F   CB7E        ..X5:   BIT     7,M       ;SEE IF END OF TABLE
F411   C0                  RNZ               ;RETURN IF SO
F412   18C3                JMPR    ..X2
F414   CD F512     ..X6:   CALL    CRLF
F417   CD F488     ..X7:   CALL    BLK
F41A   7E                  MOV     A,M
F41B   23                  INX     H
F41C   B7                  ORA     A
F41D   F8                  RM
F41E   4F                  MOV     C,A
F41F   CD F48A             CALL    CO
F422   0E3D                MVI     C,'='
F424   CD F48A             CALL    CO
F427   7E                  MOV     A,M
F428   47                  MOV     B,A       ;SAVE FLAGS
F429   E63F                ANI     3FH       ;CLEAN UP FOR OFFSET
F42B   23                  INX     H
F42C   EB                  XCHG
F42D   6F                  MOV     L,A
F42E   2600                MVI     H,0
F430   39                  DAD     SP
F431   EB                  XCHG
F432   CB70                BIT     6,B       ;TEST FOR SPECIAL "M"
F434   200F                JRNZ    ..X9      ;PRINT OUT ACTUAL "M"
F436   1A                  LDAX    D
F437   CD F58F             CALL    LBYTE     ;PRINT REG. VALUE
F43A   CB78                BIT     7,B       ;SINGLE OR DOUBLE?
F43C   28D9                JRZ     ..X7      ;SINGLE.
F43E   1B                  DCX     D
F43F   1A                  LDAX    D
F440   CD F58F     ..X8:   CALL    LBYTE
F443   18D2                JMPR    ..X7
F445   E5          ..X9:   PUSH    H         ;SAVE HL
F446   1A                  LDAX    D         ;GET REG. POINTER
F447   67                  MOV     H,A       ;HIGH BYTE
F448   1B                  DCX     D
F449   1A                  LDAX    D
F44A   6F                  MOV     L,A       ;LOW BYTE
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F44B   7E                MOV     A,M     ;GET VALUE
F44C   E1                POP     H       ;RESTORE HL
F44D   18F1              JMPR    ..X8    ;PRINT VALUE & CONTINUE
                         ;
                         ; THIS IS A MESSAGE OUTPUT ROUTINE.
                         ; IT IS USED BY THE SIGN-ON AND CRLF.
                         ; POINTER IS IN HL (WHEN ENTERED AT
                         ; TOM) AND LENGTH IN B REG.
                         ;
F44F   21 F021           TOM1:   LXI     H,MSG
F452   4E                TOM:    MOV     C,M     ;GET A CHARACTER
F453   23                INX     H       ;MOVE POINTER
F454   CD F48A           CALL    CO      ;OUTPUT IT
F457   10F9              DJNZ    TOM     ;KEEP GOING TILL B=0
F459   CD F51A           CALL    CSTS    ;SEE IF AN ABORT REQUEST
F45C   B7                ORA     A       ; WAITING.
F45D   C8                RZ              ;NO.
                         ;
                         ; SEE IF CONTROL-C IS WAITING
                         ; ABORT IF SO.
                         ;
F45E   CD F730           CCHK:   CALL    KI
F461   FE03              CPI     3       ;CONTROL-C?
F463   C0                RNZ
                         ;
                         ; SYSTEM ERROR ROUTINE. THIS
                         ; WILL RESTORE THE SYSTEM AFTER
                         ; A SYSTEM ERROR HAS BEEN TAKEN.
                         ; THE I/O CONFIGURATION IS NOT
                         ; AFFECTED.
                         ;
F464   CD F5B9           ERROR:  CALL    MEMSIZ
F467   11 FFEA           LXI     D,-22   ;STACK POINTER OFFSET
F46A   19                DAD     D
F46B   F9                SPHL            ;RESET STACK
F46C   0E2A              MVI     C,'*'   ;ANNOUNCE ERROR
F46E   CD F48A           CALL    CO
F471   C3 F07C           JMP     START   ;BACK TO WORK
                         ;
                         ; THIS GETS A READER CHARACTER,
                         ; AND ALSO COMPARES IT WITH "D" REG.
                         ; IT WILL ABORT ON AN "OUT-OF-DATA"
                         ; CONDITION.
                         ;
F474   CD F636           RIFF:   CALL    RI      ;NORMAL READER ROUTINE
F477   38EB              JRC     ERROR   ;ABORT ON A CARRY
F479   BA                CMP     D       ;COMPARE W/"D" REG.
F47A   C9                RET
                         ;
                         ; THIS ROUTINE WILL RETURN THE
                         ; CURRENT VALUE OF THE HIGHEST
                         ; READ/WRITE MEMORY LOCATION THAT
                         ; IS AVAILABLE ON THE SYSTEM.
                         ; IT WILL "SEARCH" FOR MEMORY
```

```
                        ; STARTING AT THE BOTTOM OF MEMORY
                        ; AND GO UPWARDS UNTIL NON-R/W MEMORY
                        ; IS FOUND.
                        ;
F47B    CD F5B9         SIZE:   CALL    MEMSIZE ;GET THE VALUE
F47E    01 0023                 LXI     B,(ENDX-EXIT)
F481    09                      DAD     B       ;ADJUST IT
                        ;
                        ;
                        ; CRLF BEFORE HLSP ROUTINE
                        ;
F482    CD F512         LFADR:  CALL    CRLF
                        ;
                        ; PRINT THE CURRENT VALUE OF H&L,
                        ; AND A SPACE.
                        ;
F485    CD F58A         HLSP:   CALL    LADR
                        ;
                        ; PRINT A SPACE ON THE CONSOLE
                        ;
F488    0E20            BLK:    MVI     C,' '
                        ;
                        ; THIS IS THE MAIN CONSOLE
                        ; OUTPUT ROUTINE.
                        ;
F48A    DB76            CO:     IN      IOBYT
F48C    E603                    ANI     # CMSK
F48E    200A                    JRNZ    COO
                        ;
                        ; TELEPRINTER CONFIGURATION
                        ; I/O DRIVER.
                        ;
F490    DB70            TTYOUT: IN      TTS
F492    E602                    ANI     TTYBE
F494    28FA                    JRZ     TTYOUT
F496    79                      MOV     A,C
F497    D371                    OUT     TTO
F499    C9                      RET
F49A    3D              COO:    DCR     A       ;CCRT?
F49B    200A                    JRNZ    CO1     ; NO.
                        ;
                        ; C.R.T. CONFIGURATION DRIVER.
                        ;
F49D    DB72            CRTOUT: IN      CRTS
F49F    E602                    ANI     CRTBE
F4A1    28FA                    JRZ     CRTOUT
F4A3    79                      MOV     A,C
F4A4    D373                    OUT     CRTO
F4A6    C9                      RET
                        ;
F4A7    3D              CO1:    DCR     A       ;BATCH?
F4A8    C2 F803                 JNZ     COLOC   ; NO, MUST BE USER
                        ;
                        ; LIST OUTPUT DRIVER ROUTINE
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
                        ; -AN EXTERNALLY VECTORED ROUTINE,
                        ; USED BY THE ASSEMBLER, ETC. ALSO,
                        ; WHEN THE ASSIGNED MODE IS "BATCH",
                        ; THIS IS THE ROUTINE USED FOR THE
                        ; MONITOR OUTPUT THAT WOULD NORMALLY
                        ; GO TO THE "CONSOLE".
                        ;
F4AB    DB76            LO:     IN      IOBYT
F4AD    E6C0                    ANI     # LMSK
F4AF    28DF                    JRZ     TTYOUT
F4B1    FE40                    CPI     LCRT
F4B3    28E8                    JRZ     CRTOUT
F4B5    FE80                    CPI     LINE
F4B7    CA F812                 JZ      LNLOC   ;EXTERNAL VECTOR
F4BA    C3 F815                 JMP     LULOC   ;USER DEFINED VECTOR
                        ;
                        ; SEND CRLF TO PUNCH DEVICE
                        ;
F4BD    0E0D            PEOL:   MVI     C,CR
F4BF    CD F4C4                 CALL    PO
F4C2    0E0A                    MVI     C,LF
                        ;
                        ; PUNCH OUTPUT DRIVER ROUTINE
                        ;
F4C4    DB76            PO:     IN      IOBYT
F4C6    E630                    ANI     # PMSK
F4C8    28C6                    JRZ     TTYOUT  ;PUNCH=TELEPRINTER
F4CA    FE20                    CPI     PCAS    ;CASSETTE?
F4CC    200A                    JRNZ    PO1     ; NO.
                        ;
F4CE    DB74            PO0:    IN      PCASS   ;CASSETTE DRIVER
F4D0    E602                    ANI     PCSBE
F4D2    28FA                    JRZ     PO0
F4D4    79                      MOV     A,C
F4D5    D375                    OUT     PCASO
F4D7    C9                      RET
                        ;
F4D8    FE10            PO1:    CPI     PPTP
F4DA    CA F80C                 JZ      PTPL    ;EXTERNAL VECTOR
F4DD    C3 F80F                 JMP     PULOC   ;USER VECTOR
                        ;
                        ;
                        ; THIS IS A BINARY DUMP ROUTINE THAT MAY BE
                        ; USED WITH BOTH PAPER-TAPE AND/OR CASSETTE
                        ; SYSTEMS.  IT PUNCHES A START-OF-FILE MARK
                        ; AND THEN PUNCHES IN FULL 8-BITS DIRECTLY
                        ; FROM MEMORY.  IT IS FOLLOWED BY AN END-OF-
                        ; FILE MARKER. THESE DUMPS MAY BE LOADED
                        ; USING THE "L" COMMAND. THEY ARE USEFUL
                        ; FOR FAST LOADING, AND MAY BE VERIFIED
                        ; USING THE "C" (COMPARE) COMMAND.
                        ;
                        ;   U<A1>,<A2>[CR]
                        ; PUNCHES FROM <A1> THRU <A2>
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
                        ;
F4E0    CD F50D         UNLD:   CALL    EXLF    ;GET TWO PARAMETERS
F4E3    CD F4FB                 CALL    WAIT    ;PAUSE FOR PUNCH-ON (TTY)
F4E6    CD F5A3                 CALL    LEAD    ;PUNCH LEADER
F4E9    CD F59E                 CALL    MARK    ;PUNCH FILE MARKER
F4EC    4E              ..U:    MOV     C,M     ;GET MEMORY BYTE
F4ED    CD F4C4                 CALL    PO      ;PUNCH IT
F4F0    CD F574                 CALL    HILO    ;SEE IF DONE
F4F3    30F7                    JRNC    ..U
F4F5    CD F59E                 CALL    MARK    ;PUNCH END FILE MARKER
                        ;
                        ; THIS PUNCHES NULLS (LEADER/TRAILER).
                        ; IT RETURNS "QUIET" IN CASE THE PUNCH
                        ; AND CONSOLE ARE THE SAME.
                        ;
F4F8    CD F5A3         NULL:   CALL    LEAD    ;PUNCH NULLS
                        ;
                        ; THIS ROUTINE WILL PAUSE FOR
                        ; A KEYBOARD CHARACTER. IT IS
                        ; USED AS A DELAY TO GIVE THE
                        ; OPERATOR TIME TO TURN ON THE
                        ; TELEPRINTER PUNCH BEFORE SENDING
                        ; A HEX FILE OR BINARY FILE TO
                        ; THE PUNCH.  IT WILL SIMPLY
                        ; RETURN IF THE PUNCH & CONSOLE
                        ; ARE NOT BOTH ASSIGNED TO THE
                        ; DEFAULT. (TELEPRINTER).
                        ;
F4FB    DB76            WAIT:   IN      IOBYT
F4FD    E633                    ANI     # CMSK ! # PMSK
F4FF    C0                      RNZ
F500    C3 F088                 JMP     STAR0   ;RETURN "QUIET"
                        ;
                        ; CONVERT HEX TO ASCII
                        ;
F503    E60F            CONV:   ANI     0FH     ;LOW NIBBLE ONLY
F505    C690                    ADI     90H
F507    27                      DAA
F508    CE40                    ACI     40H
F50A    27                      DAA
F50B    4F                      MOV     C,A
F50C    C9                      RET
                        ;
                        ; GET TWO PARAMETERS, PLACE
                        ; THEM IN DE & HL, AND THEN
                        ; CRLF.
                        ;
F50D    CD F542         EXLF:   CALL    EXPR
F510    D1                      POP     D
F511    E1                      POP     H
                        ;
                        ; CONSOLE CARRIAGE RETURN &
                        ; LINE FEED ROUTINE.
                        ;
```

```
                        ; THE NUMBER OF FILL CHARACTERS
                        ; IS SET TO 3 TO ALLOW A
                        ; LARGER NUMBER OF TERMINALS TO
                        ; BE USED WITH THIS MONITOR.
                        ; THE NUMBER OF FILLS MAY NOT BE
                        ; ADJUSTED.
                        ;
F512    E5              CRLF:   PUSH    H       ;SAVE HL
F513    0605                    MVI     B,5     ;CRLF LENGTH
F515    CD F44F                 CALL    TOM1    ;SEND CRLF
F518    E1                      POP     H
F519    C9                      RET
                        ;
                        ; TEST THE CURRENT CONSOLES
                        ; KEYBOARD FOR A KEY-PRESS.
                        ; RETURN TRUE (0FFH IN A REG)
                        ; IF THERE IS A CHARACTER
                        ; WAITING IN THE UART.
                        ;
F51A    DB76            CSTS:   IN      IOBYT
F51C    E603                    ANI     # CMSK
F51E    2004                    JRNZ    CS0
F520    DB70                    IN      TTS
F522    1805                    JMPR    CS1
F524    3D              CS0:    DCR     A       ;CCRT
F525    2009                    JRNZ    CS3
F527    DB72                    IN      CRTS
F529    E601            CS1:    ANI     TTYDA
F52B    3EFF                    MVI     A,TRUE
F52D    C0              CS2:    RNZ
F52E    2F                      CMA
F52F    C9                      RET
F530    3D              CS3:    DCR     A       ;BATCH
F531    C8                      RZ
F532    C3 F818                 JMP     CSLOC   ;USED DEFINED VECTOR
                        ;
                        ; GET THREE PARAMETERS AND
                        ; CRLF.
                        ;
F535    0C              EXPR3:  INR     C
F536    CD F542                 CALL    EXPR
F539    CD F512                 CALL    CRLF
F53C    C1                      POP     B
F53D    D1                      POP     D
F53E    E1                      POP     H
F53F    C9                      RET
                        ;
                        ; GET ONE PARAMETER.
                        ; NO CRLF.
                        ;
F540    0E01            EXPR1:  MVI     C,1
                        ;
                        ; THIS IS THE MAIN "PARAMETER-GETTING" ROUTINE.
                        ; THIS ROUTINE WILL ABORT ON A NON-HEX CHARACTER.
```

```
                        ; IT TAKES THE MOST RECENTLY TYPED FOUR VALID
                        ; HEX CHARACTERS, AND PLACES THEM UP ON THE STACK.
                        ; (AS ONE 16 BIT VALUE, CONTAINED IN TWO
                        ; 8-BIT BYTES.)  IF A CARRIAGE RETURN IS ENTERED,
                        ; IT WILL PLACE THE VALUE OF "0000" IN THE STACK.
                        ;
F542    21 0000         EXPR:   LXI     H,0     ;INITIALIZE HL TO ZERO
F545    CD F736         EX0:    CALL    TI      ;GET SOMETHING FROM CONSOLE
F548    47              EX1:    MOV     B,A     ;SAVE IT
F549    CD F5D9                 CALL    NIBBLE  ;CONVERT ASCII TO HEX.
F54C    3808                    JRC     ..EX2   ;ILLEGAL CHARACTER DETECTED
F54E    29                      DAD     H       ;MULTIPLY BY 16
F54F    29                      DAD     H
F550    29                      DAD     H
F551    29                      DAD     H
F552    B5                      ORA     L       ;OR IN THE SINGLE NIBBLE
F553    6F                      MOV     L,A
F554    18EF                    JMPR    EX0     ;GET SOME MORE
F556    E3              ..EX2:  XTHL            ;SAVE UP IN STACK
F557    E5                      PUSH    H       ;REPLACE THE RETURN
F558    78                      MOV     A,B     ;TEST THE DELIMITER
F559    CD F60D                 CALL    QCHK
F55C    3002                    JRNC    ..EX3   ;CR ENTERED
F55E    0D                      DCR     C       ;SHOULD GO TO ZERO
F55F    C8                      RZ              ; RETURN IF IT DOES
F560    C2 F464         ..EX3:  JNZ     ERROR   ;SOMETHING WRONG
F563    0D                      DCR     C       ;DO THIS AGAIN?
F564    20DC                    JRNZ    EXPR    ; YES.
F566    C9                      RET             ;ELSE RETURN
F567    0E01            EXF:    MVI     C,1
F569    21 0000                 LXI     H,0
F56C    18DA                    JMPR    EX1

                        ;
                        ; RANGE TESTING ROUTINES.
                        ; CARRY SET INDICATES RANGE EXCEEDED.
                        ;
F56E    CD F574         HILOX:  CALL    HILO
F571    D0                      RNC             ;OK
F572    D1                      POP     D       ;RETURN ONE LEVEL BACK
F573    C9                      RET

                        ;
F574    23              HILO:   INX     H       ;INCREMENT HL
F575    7C                      MOV     A,H     ;TEST FOR CROSSING 64K BORDER
F576    B5                      ORA     L
F577    37                      STC             ;CARRY SET=STOP
F578    C8                      RZ              ;YES, BORDER CROSSED
F579    7B                      MOV     A,E     ;NOW, TEST HL VS. DE
F57A    95                      SUB     L
F57B    7A                      MOV     A,D
F57C    9C                      SBB     H
F57D    C9                      RET             ;IF CARRY WAS SET, THEN STOP

                        ;
                        ;       HEXADECIMAL MATH ROUTINE
                        ;
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
                        ; THIS ROUTINE IS USEFUL FOR
                        ; DETERMINING RELATIVE JUMP
                        ; OFFSETS.  IT RETURNS THE SUM
                        ; & DIFFERENCE OF TWO PARAMETERS.
                        ;
                        ;   H<X>,<Y>
                        ;
                        ;  X+Y   X-Y
                        ;
F57E    CD F50D         HEXN:   CALL    EXLF
F581    E5                      PUSH    H       ;SAVE HL FOR LATER
F582    19                      DAD     D       ;GET SUM
F583    CD F485                 CALL    HLSP    ;PRINT IT
F586    E1                      POP     H       ;THIS IS LATER
F587    B7                      ORA     A       ;CLEAR CARRY
F588    ED52                    DSBC    D       ;GET DIFFERENCE & PRINT IT
                        ;
                        ; PRINT H&L ON CONSOLE
                        ;
F58A    7C              LADR:   MOV     A,H
F58B    CD F58F                 CALL    LBYTE
F58E    7D                      MOV     A,L
F58F    F5              LBYTE:  PUSH    PSW
F590    0F                      RRC
F591    0F                      RRC
F592    0F                      RRC
F593    0F                      RRC
F594    CD F598                 CALL    ..2
F597    F1                      POP     PSW
F598    CD F503         ..2:    CALL    CONV
F59B    C3 F48A                 JMP     CO
                        ; THIS ROUTINE SENDS EIGHT RUBOUTS
                        ; TO THE PUNCH DEVICE.
                        ;
F59E    01 08FF         MARK:   LXI     B,08FFH ;SET-UP B&C
F5A1    1803                    JMPR    LE0
                        ;
                        ; THIS ROUTINE SENDS BLANKS TO THE
                        ; PUNCH DEVICE.
                        ;
F5A3    01 4800         LEAD:   LXI     B,4800H ;PRESET FOR SOME NULLS
F5A6    CD F4C4         LE0:    CALL    PO
F5A9    10FB                    DJNZ    LE0
F5AB    C9                      RET
                        ;
                        ; THIS ROUTINE RETURNS TO A USER
                        ; PROGRAM THE CURRENT TOP OF
                        ; MEMORY VALUE MINUS WORKSPACE
                        ; AREA USED BY THE MONITOR.
                        ;
F5AC    E5              MEMCK:  PUSH    H
F5AD    CD F5B9                 CALL    MEMSIZ
F5B0    7D                      MOV     A,L
F5B1    D63C                    SUI     3CH
```

```
F5B3    3001                    JRNC    ..B
F5B5    25                      DCR     H
F5B6    44              ..B:    MOV     B,H
F5B7    E1                      POP     H
F5B8    C9                      RET
                        ;
                        ; THIS IS A CALLED ROUTINE USED
                        ; TO CALCULATE THE TOP OF MEMORY
                        ; STARTING FROM THE BOTTOM OF
                        ; MEMORY, AND SEARCHING UPWARD UNTIL
                        ; FIRST R/W MEMORY IS FOUND, AND THEN
                        ; CONTINUING UNTIL THE END OF THE R/W
                        ; MEMORY. THIS ALLOWS R.O.M. AT ZERO,
                        ; AND INSURES A CONTINUOUS MEMORY BLOCK
                        ; HAS BEEN FOUND.
                        ; IT IS USED BY THE ERROR ROUTINE TO
                        ; RESET THE STACK POINTER.
                        ;
F5B9    C5              MEMSIZ: PUSH    B
F5BA    21 FFFF                 LXI     H,-1    ;START AT THE BOTTOM
F5BD    24              ..M0:   INR     H       ;FIRST FIND R/W
F5BE    7E                      MOV     A,M
F5BF    2F                      CMA
F5C0    77                      MOV     M,A
F5C1    BE                      CMP     M
F5C2    2F                      CMA
F5C3    77                      MOV     M,A
F5C4    20F7                    JRNZ    ..M0    ;KEEP LOOKING FOR RAM
F5C6    24              ..M1:   INR     H       ;R/W FOUND, NOW FIND END
F5C7    7E                      MOV     A,M
F5C8    2F                      CMA
F5C9    77                      MOV     M,A
F5CA    BE                      CMP     M
F5CB    2F                      CMA
F5CC    77                      MOV     M,A
F5CD    28F7                    JRZ     ..M1    ;NOT THERE YET
F5CF    25              ..M2:   DCR     H       ;BACK UP, SUBTRACT WORKSPACE
F5D0    01 FFDD                 LXI     B,EXIT-ENDX
F5D3    09                      DAD     B
F5D4    C1                      POP     B       ;RESTORE BC
F5D5    C9                      RET             ;VALUE IN HL
                        ;
                        ;
F5D6    CD F67B         RIBBLE: CALL    RIX
F5D9    D630            NIBBLE: SUI     '0'     ;QUALIFY & CONVERT
F5DB    D8                      RC              ;<0
F5DC    FE17                    CPI     'G'-'0' ;>F?
F5DE    3F                      CMC             ;PERVERT CARRY
F5DF    D8                      RC
F5E0    FE0A                    CPI     10      ;NMBR?
F5E2    3F                      CMC             ;PERVERT AGAIN
F5E3    D0                      RNC             ;RETURN CLEAN
F5E4    D607                    SUI 'A'-'9'-1   ;ADJUST
F5E6    FE0A                    CPI     0AH     ;FILTER ":" THRU "@"
```

```
F5E8    C9                      RET
                        ;
                        ; SEND H&L VALUE TO PUNCH DEVICE
                        ;
F5E9    7C              PADR:   MOV     A,H
F5EA    CD F5EE                 CALL    PBYTE
F5ED    7D                      MOV     A,L
                        ;
                        ; PUNCH A SINGLE BYTE
                        ;
F5EE    F5              PBYTE:  PUSH    PSW     ;NIBBLE AT A TIME
F5EF    0F                      RRC
F5F0    0F                      RRC
F5F1    0F                      RRC
F5F2    0F                      RRC
F5F3    CD F503                 CALL    CONV
F5F6    CD F4C4                 CALL    PO
F5F9    F1                      POP     PSW     ;NEXT NIBBLE
F5FA    F5                      PUSH    PSW     ;SAVE FOR CHECKSUM
F5FB    CD F503                 CALL    CONV
F5FE    CD F4C4                 CALL    PO
F601    F1                      POP     PSW     ;ORIGINAL BYTE HERE
F602    82                      ADD     D       ;ADDED TO CHECKSUM
F603    57                      MOV     D,A     ;UPDATE CHECKSUM
F604    C9                      RET
                        ;
                        ;
F605    0E2D            COPCK:  MVI     C,'-'   ;PROMPT FOR CONSOLE
F607    CD F48A                 CALL    CO
                        ;
F60A    CD F736         PCHK:   CALL    TI
                        ;
                        ; TEST FOR DELIMITERS
                        ;
F60D    FE20            QCHK:   CPI     ' '     ;RETURN ZERO IF DELIMITER
F60F    C8                      RZ
F610    FE2C                    CPI     ','
F612    C8                      RZ
F613    FE0D                    CPI     CR      ;RETURN W/CARRY SET IF CR
F615    37                      STC
F616    C8                      RZ
F617    3F                      CMC             ;ELSE NON-ZERO, NO CARRY
F618    C9                      RET
                        ;
                        ; MAIN CONSOLE INPUT ROUTINE
                        ;
F619    DB76            CI:     IN      IOBYT
F61B    E603                    ANI     # CMSK
F61D    2008                    JRNZ    CI1
                        ;
                        ; TELEPRINTER ROUTINE
                        ;
F61F    DB70            TTYIN:  IN      TTS
F621    1F                      RAR
```

```
F622    30FB                    JRNC    TTYIN
F624    DB71                    IN      TTI
F626    C9                      RET
                        ;
F627    3D              CI1:    DCR     A       ;CONSOLE=CRT?
F628    2008                    JRNZ    CI2
                        ;
                        ; C.R.T. INPUT ROUTINE
                        ;
F62A    DB72            CRTIN:  IN      CRTS
F62C    1F                      RAR
F62D    30FB                    JRNC    CRTIN
F62F    DB73                    IN      CRTI
F631    C9                      RET
                        ;
F632    3D              CI2:    DCR     A       ;BATCH?
F633    C2 F800                 JNZ     CILOC   ;NO, MUST BE USER DEFINED
                        ;
                        ;
                        ; READER INPUT ROUTINE, WITH
                        ; TIME-OUT DELAY. INCLUDES
                        ; PULSING OF HARDWARE PORT
                        ; TO INDICATE REQUEST FOR
                        ; READER DATA.
                        ;
F636    E5              RI:     PUSH    H
F637    DB76                    IN      IOBYT
F639    E60C                    ANI     # RMSK
F63B    2F                      CMA
F63C    D37A                    OUT     RCP     ;INFORM OF DESIRE FOR INPUT
F63E    2F                      CMA
F63F    D37A                    OUT     RCP
F641    201A                    JRNZ    RI3
F643    67                      MOV     H,A     ;CLEAR FOR TIME-OUT TEST
F644    DB70            RI0:    IN      TTS
F646    1F                      RAR
F647    380F                    JRC     RI2
F649    C5                      PUSH    B
F64A    0600                    MVI     B,0
F64C    E3              DL0:    XTHL            ;WASTE TIME
F64D    E3                      XTHL            ;FOR DELAY
F64E    10FC                    DJNZ    DL0
F650    C1                      POP     B
F651    25                      DCR     H
F652    20F0                    JRNZ    RI0
F654    AF              RI1:    XRA     A
F655    37                      STC
F656    E1                      POP     H
F657    C9                      RET
F658    DB71            RI2:    IN      TTI
F65A    B7              RID:    ORA     A
F65B    E1                      POP     H
F65C    C9                      RET
F65D    FE08            RI3:    CPI     RCAS
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F65F  2011            JRNZ    RI6
F661  DBFF      RI4:  IN      SWITCH  ;TEST FOR AN ABORT
F663  6F              MOV     L,A     ;SAVE INITIAL STATUS
F664  DBFF      ..R4A:IN      SWITCH
F666  BD              CMP     L       ;SEE IF IT CHANGES
F667  20EB            JRNZ    RI1     ;YES, ABORT
F669  DB74            IN      RCSS    ;CASSETTE INPUT DRIVER
F66B  1F              RAR
F66C  30F6            JRNC    ..R4A
F66E  DB75      RI5:  IN      RCSD
F670  18E8            JMPR    RID
F672  E1        RI6:  POP     H
F673  FE04            CPI     RPTR
F675  CA F806         JZ      RPTPL   ;EXTERNAL ROUTINE
F678  C3 F809         JMP     RULOC   ;USER VECTOR
                      ;
                      ; THIS ROUTINE GETS READER INPUT
                      ; AND KILLS THE PARITY BIT.
                      ;
F67B  CD F474   RIX:  CALL    RIFF
F67E  E67F            ANI     7FH
F680  C9              RET
                      ;
                      ; THIS ROUTINE READS A BINARY FILE
                      ; IMAGE, IN THE FORM AS PUNCHED IN
                      ; THE "U" (UNLOAD) COMMAND.  IT TAKES
                      ; ONE PARAMETER, WHICH IS THE STARTING
                      ; ADDRESS OF THE LOAD, AND WILL PRINT
                      ; THE LAST ADDRESS(+1) LOADED ON THE
                      ; CONSOLE DEVICE.
                      ;
F681  CD F540   LOAD: CALL    EXPR1   ;INITIAL LOAD ADDRESS
F684  E1              POP     H
F685  CD F512         CALL    CRLF
F688  16FF            MVI     D,0FFH  ;START-OF-FILE TAG
F68A  0604      ..L0: MVI     B,4     ;FIND AT LEAST FOUR 0FFH'S
F68C  CD F474   ..L1: CALL    RIFF
F68F  20F9            JRNZ    ..L0
F691  10F9            DJNZ    ..L1
F693  CD F474   ..L2: CALL    RIFF    ;4 FOUND, NOW WAIT FOR NON-0FFH
F696  28FB            JRZ     ..L2
F698  77              MOV     M,A     ;FIRST REAL DATA BYTE
F699  3E07            MVI     A,BELL  ;TELL TTY
F69B  D371            OUT     TTO
F69D  23        ..L3: INX     H
F69E  CD F474         CALL    RIFF
F6A1  2803            JRZ     ..EL    ;POSSIBLE END OF FILE
F6A3  77              MOV     M,A
F6A4  18F7            JMPR    ..L3
F6A6  1E01      ..EL: MVI     E,1     ;INITIALIZE
F6A8  CD F474   ..EL0:CALL    RIFF
F6AB  2009            JRNZ    ..EL1
F6AD  1C              INR     E       ;COUNT QUES
F6AE  3E07            MVI     A,MAX   ;LOOK FOR EOF
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F6B0    BB                      CMP     E       ;FOUND MAX?
F6B1    20F5                    JRNZ    ..EL0   ;NOPE
F6B3    C3 F58A                 JMP     LADR    ;YEP, PRINT END ADDR
F6B6    72              ..EL1:  MOV     M,D
F6B7    23                      INX     H
F6B8    1D                      DCR     E       ;RESTORE
F6B9    20FB                    JRNZ    ..EL1
F6BB    77                      MOV     M,A     ;REAL BYTE
F6BC    18DF                    JMPR    ..L3
                        ;
                        ; THIS IS THE BREAKPOINT "TRAP" HANDLING
                        ; ROUTINE.  ALL USER REGISTERS ARE SAVED
                        ; FOR DISPLAY PURPOSES, AND THE CONTENTS
                        ; ARE RESTORED WHEN EXECUTING A "GO" (G)
                        ; COMMAND.
                        ;
F6BE    E5              RESTART: PUSH   H       ;PUSH ALL REGISTERS
F6BF    D5                       PUSH   D
F6C0    C5                       PUSH   B
F6C1    F5                       PUSH   PSW
F6C2    CD F5B9                  CALL   MEMSIZ  ;GET MONITOR'S STACK VALUE
F6C5    EB                       XCHG
F6C6    21 000A                  LXI H, 10      ;GO UP 10 BYTES IN STACK
F6C9    39                       DAD    SP
F6CA    0604                     MVI    B,4     ;PICK OFF REG.
F6CC    EB                       XCHG
F6CD    2B              ..R0:    DCX    H
F6CE    72                       MOV    M,D     ;SAVE IN WORKAREA
F6CF    2B                       DCX    H
F6D0    73                       MOV    M,E
F6D1    D1                       POP    D
F6D2    10F9                     DJNZ   ..R0
F6D4    C1                       POP    B
F6D5    0B                       DCX    B       ;ADJUST P.C. VALUE
F6D6    F9                       SPHL           ;SET MONITOR STACK
F6D7    21 0025                  LXI H, TLOCX
F6DA    39                       DAD    SP
F6DB    7E                       MOV    A,M
F6DC    91                       SUB    C       ;LOOK FOR A TRAP/MATCH
F6DD    23                       INX    H
F6DE    2004                     JRNZ   ..R1
F6E0    7E                       MOV    A,M
F6E1    90                       SUB    B
F6E2    280C                     JRZ    ..R3    ;NO TRAP HERE
F6E4    23              ..R1:    INX    H
F6E5    23                       INX    H
F6E6    7E                       MOV    A,M
F6E7    91                       SUB    C       ;TEST FOR 2ND TRAP
F6E8    2005                     JRNZ   ..R2
F6EA    23                       INX    H
F6EB    7E                       MOV    A,M
F6EC    90                       SUB    B
F6ED    2801                     JRZ    ..R3
F6EF    03              ..R2:    INX    B       ;NO TRAPS SET, RE-ADJUST P.C.
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F6F0    21 0020         ..R3:   LXI H,  LLOCX
F6F3    39                      DAD     SP
F6F4    73                      MOV     M,E     ;STORE USER H&L
F6F5    23                      INX     H
F6F6    72                      MOV     M,D
F6F7    23                      INX     H
F6F8    23                      INX     H
F6F9    71                      MOV     M,C     ;AND USER P.C.
F6FA    23                      INX     H
F6FB    70                      MOV     M,B
F6FC    C5                      PUSH    B
F6FD    0E40                    MVI     C,'@'   ;DISPLAY BREAK ADDRESS.
F6FF    CD F48A                 CALL    CO
F702    E1                      POP     H
F703    CD F58A                 CALL    LADR
F706    21 0025                 LXI H,  TLOCX
F709    39                      DAD     SP
F70A    01 0200                 LXI     B,200H
F70D    5E              ..R4:   MOV     E,M     ;REPLACE BYTES TAKEN FOR TRAP
F70E    71                      MOV     M,C     ;ZERO OUT STORAGE AREA
F70F    23                      INX     H
F710    56                      MOV     D,M
F711    71                      MOV     M,C
F712    23                      INX     H
F713    7B                      MOV     A,E
F714    B2                      ORA     D       ;DO NOTHING IF ZERO
F715    2802                    JRZ     ..R5
F717    7E                      MOV     A,M
F718    12                      STAX    D       ;STORE BYTE
F719    23              ..R5:   INX     H       ;SAME THING
F71A    10F1                    DJNZ    ..R4    ;FOR OTHER BREAKPOINT
F71C    08                      EXAF            ;GET ALTERNATE SET OF REG.'S
F71D    D9                      EXX
F71E    E5                      PUSH    H       ;AND STORE IN WORKSPACE
F71F    D5                      PUSH    D
F720    C5                      PUSH    B
F721    F5                      PUSH    PSW
F722    DDE5                    PUSH    X
F724    FDE5                    PUSH    Y
F726    ED57                    LDAI            ;GET INTERUPT VECTOR BYTE
F728    47                      MOV     B,A
F729    ED5F                    LDAR            ;GET REFRESH BYTE
F72B    4F                      MOV     C,A
F72C    C5                      PUSH    B       ;SAVE
F72D    C3 F07C                 JMP     START   ;BACK TO START
                        ;
                        ; THIS IS THE INTERNAL KEYBOARD
                        ; HANDLING ROUTINE. IT WILL IGNORE
                        ; RUBOUTS (0FFH) AND BLANKS (00),
                        ; AND IT WILL NOT ECHO CR'S & N'S.
                        ; (NO N'S FOR THE "NULL" COMMAND).
                        ; IT CONVERTS LOWER CASE TO UPPER
                        ; CASE FOR THE LOOK-UP OF COMMANDS.
                        ;
```

```
                        ; OTHER CHARACTERS ARE ECHOED AS THEY
                        ; ARE RECIEVED.
                        ;
F730    CD F619         KI:     CALL    C1      ;GET CHARACTER FROM CONSOLE
F733    E67F                    ANI     7FH     ;CLEAR PARITY BIT
F735    C9                      RET
                        ;
F736    CD F730         TI:     CALL    KI
F739    C8                      RZ              ;NULL
F73A    3C                      INR     A       ;TEST FOR RUBOUT
F73B    F8                      RM
F73C    3D                      DCR     A
F73D    FE0D                    CPI     CR      ;DON'T ECHO CR'S
F73F    C8                      RZ
F740    FE4E                    CPI     'N'     ;IGNORE N'S FOR NULL CMND
F742    C8                      RZ
F743    FE6E                    CPI     'n'
F745    280D                    JRZ     ..T
F747    C5                      PUSH    B
F748    4F                      MOV     C,A
F749    CD F48A                 CALL    CO
F74C    79                      MOV     A,C
F74D    C1                      POP     B
F74E    FE40                    CPI     'A'-1   ;CONVERT TO UPPER CASE
F750    D8                      RC
F751    FE7B                    CPI     'z'+1
F753    D0                      RNC
F754    E65F            ..T:    ANI     05FH
F756    C9                      RET
                        ;
                        ; THIS ROUTINE ALLOWS EXAMINATION OF
                        ; ANY INPUT PORT, OR THE SENDING OF
                        ; ANY VALUE TO ANY OUTPUT PORT.
                        ;
                        ; QO<N>,<V>[CR]
                        ;       OUTPUT TO PORT <N>, THE VALUE <V>
                        ;
                        ; QI<N>[CR]
                        ;       DISPLAY THE PORT <N>
                        ;
F757    CD F736         QUERY:  CALL    TI
F75A    FE4F                    CPI     'O'
F75C    281C                    JRZ     QUO
F75E    FE49                    CPI     'I'
F760    C2 F464                 JNZ     ERROR
F763    CD F540                 CALL    EXPR1
F766    C1                      POP     B
F767    ED58                    INP     E
F769    0608            BITS:   MVI     B,8     ;DISPLAY 8 BITS
F76B    CD F488                 CALL    BLK
F76E    CB23            ..Q2:   SLAR    E
F770    3E18                    MVI     A,'0' >1
F772    8F                      ADC     A       ;MAKE "0" OR "1"
F773    4F                      MOV     C,A
```

```
F774    CD F48A             CALL    CO
F777    10F5                DJNZ    ..Q2
F779    C9                  RET
F77A    CD F542     QUO:    CALL    EXPR
F77D    D1                  POP     D
F77E    C1                  POP     B
F77F    ED59                OUTP    E
F781    C9                  RET

                    ;
                    ; THIS ROUTINE VERIFIES THE CONTENTS
                    ; OF ONE MEMORY BLOCK WITH ANOTHER.
                    ;
                    ; V<ADDR1>,<ADDR2>,<ADDR3>
                    ;     VERIFY FROM <1> THRU <2> WITH
                    ; THE CONTENTS OF MEMORY BEGINNING AT <3>
                    ;
F782    CD F535     VERIFY: CALL    EXPR3   ;GET 3 PARAMETERS
F785    0A          VERIO:  LDAX    B
F786    BE                  CMP     M
F787    2805                JRZ     ..B
F789    C5                  PUSH    B
F78A    CD F15D             CALL    CERR    ;DISPLAY ERRORS
F78D    C1                  POP     B
F78E    03          ..B:    INX     B
F78F    CD F56E             CALL    HILOX
F792    18F1                JMPR    VERIO

                    ;
                    ; <SYSTEM I/O LOOK-UP TABLE>
                    ;
                    ; THE FIRST CHARACTER IS THE DEVICE NAME
                    ; (ONE LETTER) AND THE NEXT FOUR ARE THE
                    ; NAMES OF THE FOUR POSSIBLE DRIVERS TO BE
                    ; ASSIGNED.
                    ;
F794                LTBL:
F794    43          .BYTE   'C'     ;CONSOLE ASSIGNMENTS
F795    54          .BYTE   'T'     ;CTTY   T=TELEPRINTER
F796    56          .BYTE   'V'     ;CCRT   V=CRT (VIDEO MONITOR)
F797    42          .BYTE   'B'     ;CBATCH= COMMANDS FROM READER
F798    55          .BYTE   'U'     ;CUSE   USER
                    ;
F799    52          .BYTE   'R'     ;READER ASSIGNMENTS
F79A    54          .BYTE   'T'     ;RTTY
F79B    50          .BYTE   'P'     ;RPTR   P=PAPER TAPE
F79C    43          .BYTE   'C'     ;RCAS   C=CASSETTE
F79D    55          .BYTE   'U'     ;RUSER  USER
                    ;
F79E    50          .BYTE   'P'     ;PUNCH ASSIGNMENTS
F79F    54          .BYTE   'T'     ;PTTY
F7A0    50          .BYTE   'P'     ;PPTP
F7A1    43          .BYTE   'C'     ;PCAS   C=CASSETTE
F7A2    55          .BYTE   'U'     ;PUSER  USER
                    ;
F7A3    4C          .BYTE   'L'     ;LIST ASSIGNMENTS
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F7A4    54              .BYTE   'T'     ;LTTY   LIST=TELEPRINTER
F7A5    56              .BYTE   'V'     ;LCRT   LIST=CRT
F7A6    4C              .BYTE   'L'     ;LINE PRINTER
F7A7    55              .BYTE   'U'     ;LUSER  USER
                        ;
                        ;
                        ; THIS IS A SHORT PROGRAM, EXECUTED
                        ; UPON EXECUTING A 'GO' COMMAND. IT
                        ; IS PLACED IN THE WORK AREA WHEN
                        ; THE MONITOR IS INITIALIZED, AS IT
                        ; REQUIRES RAM FOR PROPER OPERATION.
                        ;
F7A8                    EXIT:           ;EXIT ROUTINE (LOADS ALL REGISTERS)
F7A8    C1                      POP     B
F7A9    79                      MOV     A,C
F7AA    ED4F                    STAR
F7AC    78                      MOV     A,B
F7AD    ED47                    STAI
F7AF    FDE1                    POP     Y
F7B1    DDE1                    POP     X
F7B3    F1                      POP     PSW
F7B4    C1                      POP     B
F7B5    D1                      POP     D
F7B6    E1                      POP     H
F7B7    08                      EXAF
F7B8    D9                      EXX
F7B9    D1                      POP     D
F7BA    C1                      POP     B
F7BB    F1                      POP     PSW
F7BC    E1                      POP     H
F7BD    F9                      SPHL
F7BE    00                      NOP             ;RESERVED FOR ENABLE INTERUPTS
F7BF    21 0000                 LXI     H,0
F7C2    C3 0000                 JMP     0
                        ;
F7C5    0000                    .WORD   0       ;STORAGE AREA FOR TRAP DATA
F7C7    00                      .BYTE   0
F7C8    ....                    ...
                        ;
                        ; DISPLACEMENTS OF REGISTER
                        ; STORAGE FROM NORMAL STACK
                        ; LOCATION.
                        ;
F7CB                    ENDX:
                        ;
0015                    ALOC    = 15H
0013                    BLOC    = 13H
0012                    CLOC    = 12H
0011                    DLOC    = 11H
0010                    ELOC    = 10H
0014                    FLOC    = 14H
0031                    HLOC    = 31H
0030                    LLOC    = 30H
```

```
0034                            PLOC    = 34H
0017                            SLOC    = 17H
0035                            TLOC    = 35H
0025                            TLOCX   = 25H
0020                            LLOCX   = 20H
                        ;
0009                            APLOC   = 09H
000B                            BPLOC   = 0BH
000A                            CPLOC   = 0AH
000D                            DPLOC   = 0DH
000C                            EPLOC   = 0CH
0008                            FPLOC   = 08H
000F                            HPLOC   = 0FH
000E                            LPLOC   = 0EH
0007                            XLOC    = 07
0005                            YLOC    = 05
0002                            RLOC    = 02
0003                            ILOC    = 03
                        ;
                        ;
                        ;
                        ; THIS IS THE TABLE USED TO DETERMINE
                        ; A VALID REGISTER IDENTIFIER, AND IT'S
                        ; DISPLACEMENT FROM THE STACK POINTER.
                        ;
                        ; POSITION ONE= REGISTER NAME, WITH BIT 7 INDICATING
                        ; END OF TABLE.
                        ;
                        ; POSITION TWO= BIAS FROM CURRENT STACK LEVEL OR'ED
                        ; WITH A TWO-BIT FLAG.  00XXXXXX=BYTE
                        ;                       10XXXXXX=WORD
                        ;                       11XXXXXX=SPECIAL FOR "M" REG.
                        ;
F7CB                    ACTBL:          ;NORMAL SET OF REGISTERS (8080)
                        ;               ;PLUS THE INTERUPT REGISTER ("I")
                        ;
F7CB    4115                    .BYTE   'A',    ALOC    !0
F7CD    4213                    .BYTE   'B',    BLOC    !0
F7CF    4312                    .BYTE   'C',    CLOC    !0
F7D1    4411                    .BYTE   'D',    DLOC    !0
F7D3    4510                    .BYTE   'E',    ELOC    !0
F7D5    4614                    .BYTE   'F',    FLOC    !0
F7D7    4831                    .BYTE   'H',    HLOC    !0
F7D9    4C30                    .BYTE   'L',    LLOC    !0
F7DB    4D81                    .BYTE   'M',    HLOC    !0C0H
F7DD    5084                    .BYTE   'P',    PLOC    !080H
F7DF    5397                    .BYTE   'S',    SLOC    !080H
F7E1    4903                    .BYTE   'I',    ILOC    !0
                        ;
F7E3    20522741        .ASCII  " RWA"
        F5CA3FC7
                        ;
F7E7                    PRMTB:          ;ADDITIONAL SET OF REGISTERS (Z-80)
                        ;
F7E7    4109                    .BYTE   'A',    APLOC   !0
```

MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>


```
F7E9    420B                    .BYTE   'B',    BPLOC   !0
F7EB    430A                    .BYTE   'C',    CPLOC   !0
F7ED    440D                    .BYTE   'D',    DPLOC   !0
F7EF    450C                    .BYTE   'E',    EPLOC   !0
F7F1    4608                    .BYTE   'F',    FPLOC   !0
F7F3    480F                    .BYTE   'H',    HPLOC   !0
F7F5    4C0E                    .BYTE   'L',    LPLOC   !0
F7F7    4DCF                    .BYTE   'M',    HPLOC   !0C0H
F7F9    5887                    .BYTE   'X',    XLOC    !080H
F7FB    5985                    .BYTE   'Y',    YLOC    !080H
F7FD    5202                    .BYTE   'R',    RLOC    !0
F7FF    C1                      .BYTE   0C1H
                        ;
F800                            Z:
                                        ;END OF PROGRAM
                        ;
                        ;
                        ;
F000                    .END    BASE
```

.MAIN. - <Zapple **MASKED ROM** Monitor, Version 1.05, Dec. 18 1976>
+++++ Symbol Table +++++

ACTBL	F7CB	AHEAD	F05F	ALOC	0015	APLOC	0009
ASSIGN	F0D6	BASE	F000	BATCH	0002	BEGIN	F032
BELL	0007	BITS	F769	BLK	F488	BLOC	0013
BPLOC	000B	BYE	F121	CCHK	F45E	CCRT	0001
CERR	F15D	CI	F619	CI1	F627	CI2	F632
CILOC	F800	CLOC	0012	CMSK	00FC	CO	F48A
CO0	F49A	CO1	F4A7	COLOC	F803	COMP	F14E
CONV	F503	COPCK	F605	CPLOC	000A	CR	000D
CRLF	F512	CRTBE	0002	CRTDA	0001	CRTI	0073
CRTIN	F62A	CRTO	0073	CRTOUT	F49D	CRTS	0072
CS0	F524	CS1	F529	CS2	F52D	CS3	F530
CSLOC	F818	CSTS	F51A	CTTY	0000	CUSE	0003
DISP	F16F	DLO	F64C	DLOC	0011	DONE	F284
DONE2	F36B	DPLOC	000D	ELOC	0010	ENDX	F7CB
EOF	F186	EPLOC	000C	ERR2	F27B	ERROR	F464
EX0	F545	EX1	F548	EXF	F567	EXIT	F7A8
EXLF	F50D	EXPR	F542	EXPR1	F540	EXPR3	F535
FALSE	0000	FIL	0000	FILL	F1A2	FIND	F344
FINDC	F33F	FLOC	0014	FOUND	F353	FPLOC	0008
GOTO	F1AF	HELLO	F077	HEXN	F57E	HILO	F574
HILOX	F56E	HLOC	0031	HLSP	F485	HPLOC	000F
ILOC	0003	IOBYT	0076	IOSET	F11D	J	F821
KI	F730	LADR	F58A	LBYTE	F58F	LCRT	0040
LE0	F5A6	LEAD	F5A3	LF	000A	LFADR	F482
LINE	0080	LLOC	0030	LLOCX	0020	LMSK	003F
LNLOC	F812	LO	F4AB	LOAD	F681	LODO	F23A
LOD4	F276	LODCB	F2B0	LODR	F290	LPLOC	000E
LTBL	F794	LTTY	0000	LULOC	F815	LUSER	00C0
MARK	F59E	MAX	0007	MEMCK	F5AC	MEMSIZ	F5B9
MOVE	F21B	MSG	F021	MSGL	0011	NIBBLE	F5D9
NN	00F8	NULL	F4F8	PADR	F5E9	PBYTE	F5EE
PCAS	0020	PCASO	0075	PCASS	0074	PCHK	F60A
PCSBE	0002	PEOL	F4BD	PLOC	0034	PMSK	00CF
PO	F4C4	PO0	F4CE	PO1	F4D8	PPTP	0010
PRMTB	F7E7	PTPL	F80C	PTTY	0000	PULOC	F80F
PUSER	0030	PUTA	F12F	QCHK	F60D	QU0	F77A
QUERY	F757	RCAS	0008	RCP	007A	RCSD	0075
RCSDA	0001	RCSS	0074	READ	F226	RESTAR	F6BE
RI	F636	RI0	F644	RI1	F654	RI2	F658
RI3	F65D	RI4	F661	RI5	F66E	RI6	F672
RIBBLE	F5D6	RID	F65A	RIFF	F474	RIX	F67B
RLOC	0002	RMSK	00F3	RPTPL	F806	RPTR	0004
RST7	0038	RTTY	0000	RUB	00FF	RULOC	F809
RUSER	000C	SBYTE	F2C0	SENSE	007A	SIZE	F47B
SLOC	0017	STAR0	F088	START	F07C	STKIT	F074
STORE	F2D3	SUBS	F2DF	SWITCH	00FF	TBL	F0A2
TELL	F361	TEST	F1FD	TI	F736	TLOC	0035
TLOCX	0025	TOM	F452	TOM1	F44F	TRAP	F01E
TRUE	FFFF	TTI	0071	TTO	0071	TTS	0070
TTYBE	0002	TTYDA	0001	TTYIN	F61F	TTYOUT	F490
TYPE	F308	UNLD	F4E0	USER	F800	VERIO	F785
VERIFY	F782	WAIT	F4FB	WHERE	F328	WRITE	F370
XAM	F3B0	XLOC	0007	YLOC	0005	Z	F800

.BLNK. 0000:03 X .DATA. 0000' X .PROG. 0000' X