8080 Apple Monitor

Apple V1.0 ECT

Copyright (c) 1979 ECT

ALL RIGHTS RESERVED

Electronic Control Technology

763 Ramsey Ave.

Hillside, NJ 07205

(201) 686-8080
Electronic Control Technology

8080 Apple Monitor

The Apple Monitor is a program for the 8080 or Z80 microprocessors with executive commands and I/O handling routines. The author of the Apple Monitor is Roger Amidon of Applezap Corp. who also authored the Zapple Monitor <Z80 only version of the Apple Monitor> for TDL/Xitan. NOTE: The Apple Monitor has nothing to do with the Apple Computer and early versions of the Apple Monitor probably existed before the Apple Computer.

The Apple Monitor can be utilized as a software equivalent of a front panel. Memory, registers and I/O can be displayed and substituted from the system terminal. Debugging of both hardware and software is possible by use of the memory test or verifying blocks of memory or use of breakpoints.

The Apple Monitor is expandable. The user may add special routines for special I/O devices and/or additional commands. All software programs may utilize the I/O routines of the Apple Monitor through the vectors at the beginning of the Apple Monitor and thereby take advantage of the dynamic assigning of the I/O ports and routines. The Apple Monitor also includes many useful subroutines that may be used by user written programs.

USER WRITTEN COMMAND ROUTINES

Three command letters are available for user written command routines - 'I', 'K' & 'O'. Apple vectors to the user jump vectors for these commands; 'I' to F812, 'K' to F815 & 'O' to F818. JMP's to the actual user written routines should be placed at these locations. A RET instruction at the end of the user written routine will return control back to the monitor displaying the prompt '>'.

The Branch 'B' command also allows additional user written command routines with the use of a letter A - Z. Control is passed to the routine at the address found in the user table at F880 to F8B3.

USER WRITTEN I/O ROUTINES

Occasionally I/O devices require special routines. The user I/O jump vectors should be located at F800 through F811. Be careful not to modify any register except those called for and do not upset the stack pointer. PUSH and match with POP's to restore registers. Use a RET at the end of the routine to return control back to the monitor.
Electronic Control Technology

Memory Map

Top of Memory

- Apple Stack
  - FFFF
- User
  - F8B4
- User Table For B
  - F880
- User
  - F81B
- User Jump Vectors
  - F800

Start of RAM

- Apple Monitor
  - F01E
- Apple Jump Vectors
  - F000

Start of ROM

Program Area

Memory Bottom

- RST 7 <Used Only During Debugging>
  - 0100
- 0038
- 0000
### Apple Jump Vectors

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F000</td>
<td>Begin Apple</td>
</tr>
<tr>
<td>F003</td>
<td>Console Input</td>
</tr>
<tr>
<td>F006</td>
<td>Reader Input</td>
</tr>
<tr>
<td>F009</td>
<td>Console Output</td>
</tr>
<tr>
<td>F00C</td>
<td>Punch Output</td>
</tr>
<tr>
<td>F00F</td>
<td>List Output</td>
</tr>
<tr>
<td>F012</td>
<td>Console Status</td>
</tr>
<tr>
<td>F015</td>
<td>I/O Assignment Check</td>
</tr>
<tr>
<td>F018</td>
<td>I/O Set</td>
</tr>
<tr>
<td>F01B</td>
<td>Memory Limit Check</td>
</tr>
</tbody>
</table>

### User Jump Vectors

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F800</td>
<td>Console Input</td>
</tr>
<tr>
<td>F803</td>
<td>Console Output</td>
</tr>
<tr>
<td>F806</td>
<td>Console Input Status</td>
</tr>
<tr>
<td>F809</td>
<td>User Defined Storage &lt;Input&gt;</td>
</tr>
<tr>
<td>F80C</td>
<td>User Defined Storage &lt;Output&gt;</td>
</tr>
<tr>
<td>F80F</td>
<td>User Defined Printer &lt;List&gt;</td>
</tr>
<tr>
<td>F812</td>
<td>I</td>
</tr>
<tr>
<td>F815</td>
<td>K</td>
</tr>
<tr>
<td>F818</td>
<td>O</td>
</tr>
<tr>
<td>F880</td>
<td>User Table For B</td>
</tr>
<tr>
<td>F8B3</td>
<td></td>
</tr>
</tbody>
</table>

### Assign

- **C**: Console
- **P**: Punch
- **D**: Data Transfer Device
- **P**: Printer
- **A**: Alternate <Parallel>
- **U**: User
- **R**: Reader
- **L**: List
- **C**: CRT
- **P**: Printer
- **D**: Data Transfer Device
- **U**: User
Electronic Control Technology

Apple V1.0 ECT

A - Assign I/O
B - Branch to user routine A-Z
C - Undefined
D - Display memory on console in Hex
E - End of file tag for Hex dumps
F - Fill memory with a constant
G - GOTO an address with breakpoints
H - Hex math sum & difference
I * User defined
J - Non-destructive memory test
K * User defined
L - Load a binary format file
M - Move memory area to another address
N - Nulls leader/trailer
O * User defined
P - Put ASCII into memory
Q - Query I/O ports QI<N>=read I/O; QO<N,V>=send I/O
R - Read a Hex file with checksum
S - Substitute/examine memory in Hex
T - Types the contents of memory in ASCII equivalent
U - Unload memory in Binary format
V - Verify memory block against another memory block
W - Write a checksummed Hex file
X - Examine/modify CPU registers
Y - 'Yes there' find 'N' Bytes in memory
Z - 'Z END' address of last R/W memory location
A - >A[d]=[$]

Assigns a device to be a particular unit.
First letter of specifier is all that's required.

device:= Console, Reader, Punch, List
unit:=
  if Console: CRT, Printer, Batch Mode, User
  if Reader: Data transfer, Printer, Alternate (parallel), User
  if Punch: Data transfer, Printer, Alternate (parallel), User
  if List: CRT, Printer, Data transfer, User

EXAMPLE:   AC=P Assign Console = Printer
            AP=P Assign Punch = Printer

B - >B.[a-z]

Branches into address table based on letter a-z.
If no command implemented, address will contain OFFFH, which aborts command.

EXAMPLE:   B.A

C - unused

D - >D[addr1],[addr2]<,byte>

Dumps memory from addr1 thru addr2, where <byte> is optional depending on line width desired.
NOTE- Defaults to 16 bytes per line.

EXAMPLE:   D0,IF
            0000 C3 07 F7 C3 24 F7 C3 32 F5 C3 84 F5 C3 53 F5 C3
            0010 65 F6 DB 76 C9 C3 CD F1 C3 DC F0 C3 38 F0 C3 38

E - >E<addr>

End of file is generated to assigned punch device.
<addr> is optional.

EXAMPLE:   E
            E1234

F - >F[addr1],[addr2],[byte]

Fills from addr1 thru addr2 with byte.

EXAMPLE:   F0,17FF,0
G -> G[addr1], addr2, addr3

Goes to addr1, and optionally set breakpoints at addr2 & addr3. If continuing from a breakpoint, the first parameter may be omitted. This will cause execution of whatever addr. is contained in the "P" register.

EXAMPLE: G1600, 163E

H -> H[va11], va12

Hex math of: va11+va12 & va11-val2 is displayed.

EXAMPLE: H2000, 102A 302A 0FD6

I - unused

J -> J[addr1], addr2

Justifies memory from addr1 thru addr2. Any errors are displayed as:

addr 00100000

where the "1" indicates a bad bit, in this case, bit 5, and addr is the location in memory the error occurred.

EXAMPLE: J800, 17FF

0F3D 00000010
0FB8 00000010
16FD 10000000

K - unused

L -> L[addr]

Loads a binary file, starting at addr. The address following the last byte loaded will then be displayed on the console.

EXAMPLE: L800 12A0

M -> M[addr1], addr2, addr3

Moves a block of memory starting at addr1, ending at addr2, to the block starting at addr3.

EXAMPLE: M0, 7FF, 1000

ELECTRONIC CONTROL TECHNOLOGY
Null simply causes 60 blanks (00) to be sent to the currently assigned punch device.

**EXAMPLE:**

```
N
```

G - unused

P - >P[addr]

Puts keyboard input directly into memory starting at addr. Inputting is terminated with a control-d. The address of the next byte that would have been loaded is displayed on the console.

**EXAMPLE:**

```
P1000
The quick brown fox jumped over a byte.
Boy was he surprised. ^D (control-d)
103D
```

Q - >QI[port]

>QO[port],[byte]

Q may be used to both display (QI) and send to (Q0) any of the 256 I/O ports. When inputting, the results are displayed in binary; 00001101 with bit zero on the right.

When outputting, [port] will be sent [byte].

**EXAMPLE:**

```
QI70 00000010
0071,7
```

R - >R<addr1>,<addr2>

Read will load a normal hex file, or a relocatable hex file. Addr1 is an optional bias, which will be added to the load address, and addr2 is the optional relocation base which is used only with relocatable files.

**EXAMPLE:**

```
R
R,300
R1000
```

S - >S[addr]

Substitutes memory, starting at addr.

**EXAMPLE:**

```
S1000 54- 68- 65-79 20- 71- 75- 69- 63-
1008 6B- 20- 62- 72-
```

ELECTRONIC CONTROL TECHNOLOGY
T - >T[addr1],[addr2]

Types out memory from addr1 thru addr2.

EXAMPLE: T1000,100F
1000 Thy quick brown
    TO,1F
    0000 C.2pC.vC6vC..tCDtC+tC..qC,uC

U - >U[addr1],[addr2]

Unloads memory from addr1 thru addr2.

EXAMPLE: U1000,17FF

V - >V[addr1],[addr2],[addr3]

Verifies the contents of memory from addr1 thru addr2
with memory starting at addr3.
If a difference is found, first the address of
the lower block is printed, then the byte found
at that address, then the byte found at the address
which would correspond relative to [addr3].

EXAMPLE: V0,402,F000

0400 FF ED
0401 FF 52
0402 FF 20

W - >W[addr1],[addr2],[byte>]

Writes hex file records of the contents of memory from
addr1 thru addr2, with the length of records
of the optional <byte>. The default maximum
length of records is 24 (18H).

EXAMPLE: W0,3FF
W100,13A,FF

X - >X<^>^r>

eXamines all the registers, or optionally, a single
register. Typing X, followed by a carriage return
displays the entire set, where X<r> followed by
a space bar, will examine the contents of a single
register, with the option of altering it's contents.
The technique is similar to the 'S' command.

EXAMPLE: X
A=18 B=AA C=28 D=A9 E=FA F=44 H=AC L=41 M=00 P=ADC2 S=AC96
X
A=18 B=AA C=28 D=10 E=FA F=44 H=AC L=41 M=00 P=ADC2 S=F000

ELECTRONIC CONTROL TECHNOLOGY
Y - >Y[byte],[byte],[byte],[byte]......

Y searches all of memory for a match on the series of <byte>s. The starting address of each occurrence is displayed on the console. Search string limit is 255 characters.

EXAMPLE: YCD,1E,F0
          0836
          0979
          1703
          231C

Z - >Z

Z alone causes the last R/W memory address to be displayed on the console. Remember, this is the last R/W location starting from the bottom. It is possible to have some memory above this point, separated by either non-R/W (ROM), or non-existant memory. In addition, the start of R/W does not have to be at zero. The command will first find R/W, and THEN the end of R/W.

EXAMPLE: Z
          17FF

ELECTRONIC CONTROL TECHNOLOGY
A - Assign I/O

This command allows dynamic re-assignment of I/O configuration from within the monitor via keyboard input. Operation is straightforward with the exception of the 'BATCH' mode. In this mode, the console input comes from the currently assigned reader, and any console output goes to the currently assigned list device. This allows batch processing; that is, you can store a series of commands in the reader device, and execute them automatically without keyboard intervention. The last command in such a series would normally be a re-assignment to a normal console.

B - Branch

This command allows user-defined commands to be executed directly from the monitor. The syntax requires a period (.) directly following the 'B', followed by a letter A-Z. Control is then passed to a command branch table located outside the P/ROM monitor. Routines are then written and their addresses placed in the table at the location corresponding to the appropriate letter. If the table address of any requested branch evaluates to OFFFH, it is considered an un-implemented command, and an error condition will occur. This would normally be the case when no memory exists in the branch table area.

C - unused

D - Display memory (in hex)

This command allows examination of memory in a hexadecimal format. An optional third parameter may be specified to allow different widths of printout. The default width is 16 bytes per line, with the address of each 16 byte block displayed on the left.

E - Eof

This generates an "end of file" string of data, and is used in conjunction with the "w" command. A 16 bit (two byte) address may be specified, which will, upon loading with the "R" command, be placed into the "P" (PC counter) storage area. This would then be used as the execution address of the object module, evoked with a simple "G<cr>".
F - Fill memory

This is used to initialize blocks of memory with a constant. Especially useful for clearing all of memory to zero after power-up.

G - Go

This command allows transferring of control to any location in memory. Also, two additional addresses may be specified as "breakpoints". If during the "fetch" cycle of operation an address that was specified as a breakpoint is encountered, system control will be transferred back to the monitor system. The contents of all processor registers may then be examined, modified, cleared, etc. Program execution may then be continued either with or without additional breakpoints. Again, the "P" counter will contain the address where the execution stopped, and a simple "G<cr>" would continue program execution.

H - Hex math

This is useful for calculating relative jump offsets, or index register pointers, etc. Overflow is ignored.

I - unused

J - Justify memory

This command is a non-destructive memory test. It is useful to be sure that a block of memory is where you thought it was. It would also spot accidentally protected memory. Because of its quick and non-destructive nature, it may not always spot "intermittent" or "nervous" memory, but any hard failures will always be detected.

K - unused
L - Load binary file

In many applications, a straight binary dump & load of memory are useful. This provides that ability, and yet does retain some degree of control. The start address is specified in the command, and the end will be determined by the file itself. This is then printed on the console. It uses the unlikely occurrence of the "OFFH" character (all bits one) appearing eight times in a row within a file. The start of file is 8 OFFH's, as is the end of file. When loading a file, when the 8 starting "OFFH"s are found, the console bell will ring and loading begins. Loading stops when the next 8 OFFH's are read.

M - Move memory

This command will allow moving mass amounts of memory from one area to another. Care should be used so as not to crash the data during a move. When moving up, say from 100H to 800H, the amount of the move (the second address in the command) must be below 800H. If that is not the case, the block should be moved well beyond it's intended place, and then moved downwards.

N - Nulls to punch

This command is most useful when using a paper tape punch for data storage. It will send 60 blanks to the punch for use as leader/trailer. It is also useful with a cassette tape system to preface any write operations. This allows the cassette to "synch up" quickly during playback.

O - unused

P - Place text to memory

This command allows typing from the keyboard ascii text directly into memory. Useful for modifying text in memory, etc.

Q - Query I/O

This allows direct inputting or outputting to the 256 I/O ports in the system.

ELECTRONIC CONTROL TECHNOLOGY
R - Read a hex file.

This will read into memory an INTEL formatted hex file. A bias may be added, which would cause the program to be loaded into memory at an address other than specified in it's loading data. This monitor also has the ability to load TDL formatted relocatable files which were generated on the TDL macro assembler. In normal usage, it is mainly meant to read in files that were generated by the "W" command.

S - Substitute memory

This allows a byte by byte examination of memory with the option of altering the data there. It will print the address on the left every 8 bytes in order to keep track of the current memory location being examined. An underline (or left arrow) will back the location to the previous byte. The command is exited by a carriage return. A space bar steps to the next location.

T - Type out memory

It is sometimes useful to examine memory in an ascii format. This command provides that ability. Any non-printing characters will be converted to periods prior to printing. A third parameter is allowed in this command, which defines the maximum characters per line. The default is 64.

U - Unload memory

This routine will dump a continuous block of memory. It is a full 8-bit binary dump, and is formatted with a blank leader, followed by 8 OFFH characters, followed by the first byte of the memory location being dumped. It continues until the range requested has been satisfied, and then dumps 8 more OFFH's, followed by some blank trailer. Files generated by this command are meant to be read by the "L" command. The formatting scheme used here relies on the fact that a OFFH is not normally found in a file, at least not 8 in a row. In order that this scheme perform correctly, it is advisable to initialize memory to zero, or some other such character. This eliminates the chance of accidentally dumping 8 or more OFFH's, which would cause an early termination during read-in with the "L" command.
V - Verify memory

This is a block to block comparison of memory. Useful to see if a program is still as it was when first loaded. You would make a 'copy' first, using the "M" command, at some safe location in memory. Then, if during running of the program you wanted to see if it had altered itself, or if the memory had dropped bits, etc., you would verify the two blocks against each other. Any changes will be printed with the address on the console.

W - Write a hex file

This will do a dump of a specific memory block, similar to the "U" command. However, this is formatted with checksums, and is in 7 bit ascii, which allows transmission over modems, or use with 7 bit storage devices, etc. A third parameter is allowed here, which defines the maximum number of bytes per record. A record defaults to 24 bytes per record, but may be optionally set to a maximum of 255 bytes. Files generated by this command may not be read by the "U" command, but must be read by the "R" command. Also, after all sections of memory have been written out, an "EDF" record must be generated, using the "E" command. This terminates the "R" command.

X - Examine registers.

The "X" command allows quick examination and modification of all 8080 8 bit temporary registers, and the 16 bit stack pointer and pc counter. The values in these registers are only valid while executing a user program via the "G" command, and are initialized to zero on powerup. If the monitor is entered by either a "BREAKPOINT" or by a "CALL TRAP", all registers will be saved, and are displayable by the "X" command. Upon continuing execution ("G<cr>"), the values are restored to the appropriate registers, and execution resumes from whence it came.
Y - Y is there.

This is a memory search routine, used to find collections of hex bytes in memory. All of memory is searched, with every occurrence of the string printed on the console. It is desirable to look for at least 2 bytes at a time, with 3-4 the usual case. There is no limit to the length of search string, but more than 12 would be unusual.

Z - Zend

This routine looks for the "TOP" of the first continuous block of memory in the system, and prints the value on the console. This represents the last R/W location in memory.

Note that the monitor places a value in the "S" register (stack pointer). This value is to be used as the highest location a user should place his stack pointer to avoid any conflict with the monitor’s stack. It is initiated on powerup, and therefore a user does not need to set the stack pointer unless he desires to do so.
; "APPLE MONITOR" COPYRIGHT 1975,1976,1977
; BY ROGER AMIDON

; #8080 ;THIS MONITOR IS 8080 CODE ONLY
; #PROGID APPLE,1,0
; #IDENT APPLE

BASE =""Rom starting address?"
PABS #THIS MONITOR IN ABSOLUTE FORMAT
XLINK #NO LINKING IN THIS PROGRAM
PHEX

; THIS VERSION WRITTEN FOR ELECTRONIC CONTROL TECHNOLOGY
; ALL RIGHTS RESERVED

F800

USER = BASE+800H

0000 IO ="I/O Port base?"

0000 CONFIG = 0 ;INITIAL CONFIGURATION

0038 RST7 = 38H ;RST 7 (LOCATION FOR TRAP)

; <I/O DEVICES>

;--C.R.T. SYSTEM

0001 CRTI = IO+1H ;DATA PORT (IN)
0000 CRTS = IO+0H ;STATUS PORT (IN)
0001 CRTD = IO+1H ;DATA PORT (OUT)
0001 CRTDA = 1 ;DATA AVAILABLE MASK
0080 CRTBE = 80H ;XMTR BUFFER EMPTY MASK

;--PRINTER

0003 TTI = IO+3H ;DATA IN PORT
0003 TTO = IO+3H ;DATA OUT PORT
0002 TTS = IO+2H ;STATUS PORT (IN)
0001 TTYDA = 1 ;DATA AVAILABLE MASK BIT
0080 TTYBE = 80H ;XMTR BUFFER EMPTY MASK

;--DATA TRANSFER SYSTEM

0005 RCSD = IO+5H ;DATA IN PORT
0004 RCSS = IO+4H ;STATUS PORT (IN)
0001 RCD = 1 ;DATA AVAILABLE MASK
0005 PCASO = IO+5H ;DATA PORT (OUT)
0080 PCSBE = 80H ;XMTR BUFFER EMPTY MASK

; PARALLEL PORT

0080 PORT
0007  PPDATA = 10+7 ;PARALLEL DATA PORT
0006  PPSTAT = 10+6 ;PARALLEL STATUS PORT
0001  PPAK = 1 ;DATA AVAILABLE
0080  PPBE = 80H ;CLEAR TO TRANSMIT DATA
;
;
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
0000  FIL = 00 ;FILL CHARACTERS AFTER CRLF
0007  MAX = 7 ;NUMBER OF QUEST IN EDU
;
;
00FC  CMSK = 11111100B ;CONSOLE DEVICE
00F3  RSMK = 11110011B ;STORAGE DEVICE (IN)
00CF  PMSK = 11001111B ;STORAGE DEVICE (OUT)
003F  LMSK = 00111111B ;LIST DEVICE
;
;
0000  CCR = 11111111B ;C.R.T.
0001  CTY = 1 ;PRINTER
0002  BATCH = 2 ;READER FOR INPUT, LIST FOR OUTPUT
0003  CUSE = 3 ;USER DEFINED
;
;
0000  RPTR = 0 ;DATA TRANSFER DEVICE
0004  RTY = 4 ;PRINTER DEVICE
0008  RCAS = 8 ;PARALLEL PORT
000C  USER = 0CH ;USER DEFINED
;
;
0000  PTP = 0 ;DATA TRANSFER DEVICE
0010  PTTY = 10H ;PRINTER PUNCH
0020  PCAS = 20H ;PARALLEL PORT
0030  USER = 30H ;USER DEFINED
;
;
0000  LTTY = 0 ;CONSOLE DEVICE
0040  LCRT = 40H ;PRINTER
0080  LINE = 80H ;DATA TRANSFER DEVICE
00C0  USER = 0COH ;USER DEFINED
;
;
FACTORS FOR USER DEFINED ROUTINES
;
F800  .LOC USER
F800  .CLOC .BLKB 3 ;CONSOLE INPUT


F003
C3 F70B
JMP CI ;CONSOLE INPUT

F006
C3 F72F
JMP RI ;READER INPUT

F009
C3 F56A
JMP CD ;CONSOLE OUTPUT

F00C
C3 F6CC
JMP PD ;PUNCH OUTPUT

F00F
C3 F590
JMP LO ;LIST OUTPUT

---

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.

'IUCHK'-
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 'GICRI' on the console.

---

01/07/79 22:40:00
APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979
©RIGHT 1979 BY APPELZAP CORP.
F069 72 MOV M,D
F06A 23 INX H
F06B 79 MOV A,C
F06C B0 ORA B
F06D CA F072 JZ ..R4
F070 7E MOV A,M
F071 02 STAX B
F072 23 ..R4: INX H
F073 1D DCR E
F074 C2 F065 JNZ ..R3
F077 C3 F0FC JMP START
F07A 7E ..R5: MOV A,M
F07B 91 SUB C
F07C 23 INX H
F07D C0 RNZ H
F07E 7E MOV A,M
F07F 90 SUB B
F080 C9 RET
F081 21 FFFF MEMSIZ: LXI H,-1 ;START AT THE BOTTOM
F084 24 ..MO: INR H ;FIRST FIND R/W MEMORY
F085 7E MOV A,M
F086 2F CMA
F087 77 MOV M,A
F088 BE CMP M
F089 2F CMA
F08A 77 MOV M,A
F08B C2 F084 JNZ ..MO
F08E 24 ..M1: INR H ;NOW FIND NON-R/W
F08F 7E MOV A,M
F090 2F CMA
F091 77 MOV M,A
F092 BE CMP M
F093 2F CMA
F094 77 MOV M,A
F095 CA F08E JZ ..M1
F098 25 DCR H
F099 C9 RET
F09A E5 MEMCK: PUSH H
F09B CD F081 CALL MEMSIZ
F09E 44 MOV B,H ;USER'S HIGH BYTE
F09F E1 POP H
F0A0 3EC0 MVI A,OCOH ;USER'S LOW BYTE
F0A2 C9 RET
F0A3 21 F0C5 TOM: LXI H,MS6
F0A6 4E TOM1: MOV C,M
F0A7 23 INX H
F0A8 CD F56A CALL CO
F0AB 05 DCR B
F0AC C2 F0A6 JNZ TOM1
F0AF CD F5D6 CALL CSTS
ANNOUNCEMENT OF MONITOR NAME & VERSION

LET US BEGIN

BEGIN: LXI H,65535-(ENDX-EXIT)

SET UP A STACK

SPHL

MOV B,ENDX-EXIT

LDAX D

MOV M,A

INX H

INX D

DCR B

JNZ ..BG1

CALL MEMSZ

GET USER'S STACK

PUSH H

MOV H,B

Zero out HL

MOV L,B

PUSH H

PUSH H

MVI A,CONFIG

STA -1

MVI B,MSG

CALL TOH

PRINT SIGN-ON

START: LXI D,START

PUSH D

CALL CRLF

CALL CPI

CALL TI

CALL CO

CALL H,TBL

CALL TI

JZ STARO

CONTROL?
<table>
<thead>
<tr>
<th>F113</th>
<th>DA F10B</th>
<th>JC</th>
<th>STAM</th>
<th>;IGNORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F116</td>
<td>D641</td>
<td>SUI</td>
<td>&quot;A&quot;</td>
<td></td>
</tr>
<tr>
<td>F118</td>
<td>D8</td>
<td>RC</td>
<td>&quot;&lt;A&quot;</td>
<td></td>
</tr>
<tr>
<td>F119</td>
<td>FE1A</td>
<td>CPI</td>
<td>&quot;Z&quot;-&quot;A&quot;+1</td>
<td></td>
</tr>
<tr>
<td>F11B</td>
<td>D0</td>
<td>RNC</td>
<td>;&quot;Z&quot;</td>
<td></td>
</tr>
<tr>
<td>F11C</td>
<td>87</td>
<td>ADD</td>
<td>A</td>
<td>;&quot;A*2&quot;</td>
</tr>
<tr>
<td>F11D</td>
<td>85</td>
<td>ADD</td>
<td>L</td>
<td>;&quot;TBL&quot;</td>
</tr>
<tr>
<td>F11E</td>
<td>6F</td>
<td>MOV</td>
<td>L,A</td>
<td></td>
</tr>
<tr>
<td>F11F</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td></td>
</tr>
<tr>
<td>F120</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>F121</td>
<td>66</td>
<td>MOV</td>
<td>H,M</td>
<td></td>
</tr>
<tr>
<td>F122</td>
<td>6F</td>
<td>MOV</td>
<td>L,A</td>
<td></td>
</tr>
<tr>
<td>F123</td>
<td>A4</td>
<td>ANA</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>F124</td>
<td>3C</td>
<td>INR</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>F125</td>
<td>CA FOBA</td>
<td>JZ</td>
<td>ERROR</td>
<td>;DON'T GO TO OFFFFH</td>
</tr>
<tr>
<td>F128</td>
<td>E9</td>
<td>PCHL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
; TBL:
F129  F15D  .WORD  ASSIGN  $A  -  ASSIGN I/O
F12B  F1A6  .WORD  BRANCH  $B  -  BRANCH TO USER ROUTINE A-Z
F12D  FFFF  .WORD  OFF  $C  -  UNDEFINED
F12F  F1B4  .WORD  DISP  $D  -  DISPLAY MEMORY ON CONSOLE IN HEX
F131  F1D4  .WORD  EOF  $E  -  END OF FILE TAG FOR HEX DUMPS
F133  F1F0  .WORD  FILL  $F  -  FILL MEMORY WITH CONSTANT
F135  F1FE  .WORD  GOTO  $G  -  GOTO <ADDRESS>, W/BKPNTS (2)
F137  F656  .WORD  HEXN  $H  -  HEX MATH <SUM> <DIFFERENCE>
F139  F812  .WORD  J  $I  -  *** USER DEFINED
F13B  F24C  .WORD  TEST  $J  -  NON-DESTRUCTIVE MEMORY TEST
F13D  F815  .WORD  J+3  $K  -  *** USER DEFINED
F13F  F267  .WORD  LOAD  $L  -  LOAD A BINARY FORMAT FILE
F141  F2AD  .WORD  MOVE  $M  -  MOVE MASS MEMORY
F143  F702  .WORD  NULL  $N  -  PUNCH LEADER/TRAILER
F145  F818  .WORD  J+6  $O  -  *** USER DEFINED
F147  F2B9  .WORD  PUTA  $P  -  'PUT' ASCII INTO MEMORY.
F149  F535  .WORD  QUERY  $Q  -  QI(N)=READ I/O; QO(N,V)=SEND I/O
F14B  F31F  .WORD  READ  $R  -  READ A HEX FILE (W/CHECKSUM)
F14D  F3F5  .WORD  SUBS  $S  -  EXAMINE/SUBSTITUTE MEMORY
F14F  F420  .WORD  TYPE  $T  -  DISPLAY MEMORY IN ASCII
F151  F6E9  .WORD  UNLD  $U  -  DUMP MEMORY IN BINARY FORMAT
F153  F4AC  .WORD  VERIFY  $V  -  COMPARE MEMORY TO MEMORY
F155  F472  .WORD  WRITE  $W  -  DUMP MEMORY IN HEX FILE FORMAT
F157  F4BA  .WORD  XAM  $X  -  EXAMINE/MODIFY CPU REGISTERS
F159  F2DB  .WORD  WHERE  $Y  -  FIND 'N' BYTES IN MEMORY
F15B  F55F  .WORD  SIZE  $Z  -  ADOR OF LAST R/W MEMORY LOCATION

FFFF  .OFF  =  -1

F880  .UTAB  =  USER+80H
```
ELECTRONIC CONTROL TECHNOLOGY

F15D   CD F793  ASSIGN: CALL TI ;GET A DEVICE
F160   21 F7AC  LXI H,LTBL-1 ;POINT TO TABLE
F163   01 0004  LXI B,4 ;TO SKIP THRU TABLE
F166   CD F186  CALL ..A3 ;GET DEVICE COUNT
F169   D5      PUSH D ;SAVE IN STACK
F16A   CD F793  ..A1: CALL TI
F16D   D63D    SUI '
F16F   C2 F16A  JNZ ..A1
F172   4F      MOV C,A ;C=0
F173   CD F793  CALL TI ;GET ASSIGNMENT
F176   CD F186  CALL ..A3
F179   F1      POP PSW ;A=DEVICE
F17A   6A      MOV L,D ;L=ASSIGNMENT
F17B   2603    MVI H,3 ;SETUP A MASK
F17D   3D      DCR A ;ZERO=DONE
F17E   FA F195  JM ..A5
F181   29      DAD H ;DOUBLE SHIFT LEFT
F182   29      DAD H
F183   C3 F17D  JMP ..A2
F186   11 0004  ..A3: LXI 0,4 ;GO THRU THIS 4 TIMES
F189   23      CMP M ;MATCH?
F18A   BE      RZ  ;YES
F18B   C8      DAD B ;BUMP HL
F18C   09      INK D
F18D   14      DCR E ;COUNT DOWN
F18E   1D      DCR E
F18F   C2 F189  JNZ ..A4
F192   C3 F0BA  JMP ERROR ;CAN'T FIND IT
F195   AC      ..A5: XRA H ;COMPLIMENT H
F196   67      MOV H,A
F197   CD F1A2  CALL IDCHK ;GET CURRENT CONFIGURATION
F19A   A4      ANA H ;KILL ASSIGNMENT BITS
F19B   B5      ORA L ;MODIFY TO NEW DEVICE
F19C   4F      MOV C,A ;PUT NEW IOBYT IN C
0040
000E
000E
; INC=--

; 0000
F1B4 0E10 DISP: MVI C, 16 ; SET A DEFAULT
F1B6 CD F5F7 CALL EXPF
F1B9 F5 PUSH PSW
F1BA CD F562 .DO: CALL, LFADDR
F1BD F1 POP PSW
F1BE F5 PUSH PSW ; GET SIZE
F1BF 47 MOV B, A ; IN B
F1C0 CD F568 .DI: CALL BLK
F1C3 7E MOV A, M
F1C4 CD F66A CALL BYTE
F1C7 CD F64C CALL HIL0
F1CA DA F64A JC PRET
F1CD 05 DCR B
F1CE C2 F1C0 JNZ .DI
F1D1 C3 F1BA JMP .DO

; 0020
F1D4 CD F623 ED: CALL FEOL
F1D7 CD F6C5 CALL C', :-
F1DA 0E3A MVI C, ':
F1DC CD F6CC CALL PO
F1DF AF XRA A
F1E0 CD F6A2 CALL PBYTE
F1E3 E1 POP H
F1E4 CD F69D CALL PADR
F1E7 21 0000 LXI H, 0
F1EA CD F69D CALL PADR
F1ED C3 F702 JMP NULL

; 001C
F1F0 CD F5F7 FILL: CALL EXPF
F1F3 71 .FI: MOV M, C
F1F4 CD F64C CALL HIL0
F1F7 D2 F1F3 JNC .FI
F1FA D1 POP D
F1FB C3 F0FC JMP START

; 000E
F1FE CD F6B6 GOTO: CALL PCHK
F201 CA F20F JZ .GO ; DELIMITER ENTERED
F204 CD F626 CALL EXF ; ELSE GET A 'GO' ADDR
F207 D1 POP D
F20B 21 0015 LXI H, FLOC
F20B 39 DAD SP
F20C 72 MOV M, D ; PLACE IN EXIT TEMPLATE
F20D 28 DCX H
F20E 73 MOV M, E ; TEST DELIMITER
F20F FE0D .GO: CPI CR
ELE.TR'NI. cnNTP.L TFcHN.l .GY

FAGL .i...

0l/07/79:lr40ro0

C! {RIGHT 1979 BY AF'FLEZAT? CBT'iF "

F?14 160:
F?16 ?1 00:
F2L9 39
F?14 E5
F:19 C?' F623
F21E C1
F21F E1
F220 F5
F221 78
F222 B1
F223 CA F230
F226 71
F227 23
F228 70
F229 23
F22A 0A
F22B 77
F22C 23
F22D 3EFF
F22E 02
F231 DA F238
F233 15
F235 C2 F21A
F238 3EC3
F23A 32 0038
F23B 21 F01E
F23D 22 0039
F243 CD F5CC
F246 D1
F247 21 0008
F24A 39
F24B E9
004E
001B

01/07/79 22:41:00
<APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979>

F211 CA F243 JZ ..G4 ;NO BREAKPOINTS, JUST GO
F214 1602 MVI D,2 ;2 POSSIBLE BREAKPOINTS
F216 21 0016 LXI H,TLOCX
F219 39 DAD SP
F21A E5 ..G1: PUSH H
F21B CD F623 CALL EXPR ;GET AN ADDRESS
F21E C1 POP B ;IN BC
F21F E1 POP H
F220 F5 PUSH PSW ;SAVE DELIMITER
F221 78 MOV A,B ;CAN'T ALLOW ANY
F222 B1 ORA C ; BREAKPOINTS AT ZERO
F223 CA F230 JZ ..G2 ;DO NOTHING
F226 71 MOV M,C
F227 23 INX H
F228 70 MOV M,B ;ELSE SAVE BKPT ADDRESS
F229 23 INX H
F22A 0A LDAX B ;AND_OPCODE THERE
F22B 77 MOV M,A
F22C 23 INX H
F22D 3EFF MVI A,0FFH ;RST 7
F22E 02 STAX B ;REPLACE_OPCODE
F231 DA F238 JC ..G3 ;LOOK AT DELIMITER
F233 15 DCR D
F235 C2 F21A JNZ ..G1
F238 3EC3 ..G3: MVI A,JMP ;SET A 'JMP' AT RST7
F23A 32 0038 STA RST7
F23B 21 F01E LXI H,TRAP
F240 22 0039 SHLD RST7+1
F243 CD F5CC ..G4: CALL CRLF
F246 D1 POP D ;THROW AWAY RETURN
F247 21 0008 LXI H,8
F24A 39 DAD SP
F24B E9 PCHL

004E
001B

; ZG=-GOTO
; ZJ=-TEST

F24C CD F5F7 TEST: CALL EXPC
F24E 7E ..T1: MOV A,M
F250 47 MOV B,A ;SAVE CHAR IN 'B'
F251 2F CMA
F252 77 MOV M,A
F253 AE XRA M
F254 70 MOV M,B ;REPLACE_BYTE
F255 CA F261 JZ ..T2
F258 D5 PUSH D ;SAVE END_POINTER
F259 5F MOV E,A ;SAVE_ERROR_MASK
F25A CD F565 CALL HLSP ;DISPLAY BAD_ADDRESS
F25D CD F5B8 CALL BITS+1 ;DISPLAY BAD_BIT(S)
F260 D1 POP D ;RESTORE_DE
F261 CD F646 ..T2: CALL HILOX
F264 C3 F24F JMP ..T1
; LOAD: CALL EXPR
F267 CD F623
F26A CD F5CC
F26D 1 E1
F26E 16 FF
F270 01 0407 ..L0: LXI B, 407H ; B=4 MATCHES, C=BELL
F273 CD F785 ..L1: CALL RIFF
F276 C2 F270 JNZ ..L0
F279 05 DCR B
F27A C2 F273 JNZ ..L1
F27D CD F785 ..L2: CALL RIFF
F280 CA F27D JZ ..L2
F283 77 MOV M, A
F284 CD F56A CALL CD ; TELL CONSOLE
F287 23 ..L3: INX H
F288 CD F785 CALL RIFF
F28B CA F292 JZ ..L5
F28E 77 ..L4: MOV M, A
F28F C3 F287 JMP ..L3
F292 1 E01 ..L5: MVI E, 1 ; INITIALIZE
F294 CD F785 ..L6: CALL RIFF
F297 C2 F2A4 JNZ ..L7
F29A 1 C INR E
F29B 3 E07 MVI A, MAX
F29D BB CMP E
F29E C2 F294 JNZ ..L6
F2A1 C3 F562 JMP LFA D R
F2A4 72 ..L7: MOV M, D
F2A5 23 INX H
F2A6 1 D ICR E
F2A7 C2 F2A4 JNZ ..L7
F2AA C3 F2BE JMP ..L4

0046 $ X L= .. LOAD

; MOVE: CALL EXPR
F2AD CD F5F7 MOVE: CALL EXPC
F2B0 7 E .. M: MOV A, M
F2B1 02 STAX B
F2B2 03 INX B
F2B3 CD F646 CALL HILOX
F2B6 C3 F2B0 JMP .. M

000C $ X M= .. MOVE

; P U T A: CALL EXPR
F2B9 CD F623 PUTA: CALL EXPR
F2BC CD F5CC CALL CR LF
F2BF E1 POP H
F2C0 CD F78D .. PO: CALL K1
F2C3 FE04 CPI 4 ; EOT?
F2C5 CA F562 JZ LFA D R ; PRINT ADDRESS & QUIT
F2C8 FE7F CPI 7FH ; RUB-OUT?
F2CA CA F2D6 JZ ..F2 ; YES
F2CD 77 MOV M, A ; PUT CHARACTER INTO MEMORY
F2CE 4F MOV C, A ;
ELECTRONIC CONTROL TECHNOLOGY

F31C C3 F2F4
JMP ..Y2

XY=. WHERE

0044

F31F CD F623 READ: CALL EXPR ;GET 16 BIT VALUE
F322 D1 POP D ;DE=BIAS
F323 21 0000 LXI H,0 ;SET-UP DEFAULT BASE[1]
F326 ES PUSH H ;AND DEFAULT BASE[2]
F327 DA F337 JC ..RO ;CR
F32A CD F623 CALL EXPR ;GET ACTUAL BASE[1]
F32D E1 POP H ;HL=BASE[1]
F32E DA F337 JC ..RO ;CR
F331 E3 XTHL ;GET DEFAULT BASE[2]
F332 CD F623 CALL EXPR ;GET ACTUAL BASE[2]
F335 E1 POP H
F336 E3 XTHL ;(SP)=BASE[2]
F337 ES PUSH H ;HL=BASE[1]
F338 D5 PUSH D ;DE=BIAS
F339 CD F5CC CALL CRLF ;BEGIN READING FILE
F33C CD F77F ..R1: CALL RX ;GET READER CHARACTER
F33F D63A SUI ..R1 ;GET FILE TYPE CUE
F341 47 MOV B,A ;SAVE CUE CUE
F342 E6FE CALL ..BYE ;GET FILE CUE
F344 C2 F33C JNZ ..R1 ;NOT ..' ' OR ..' ?'
F347 57 MOV D,A ;ZERO CHECKSUM STORAGE
F348 CD F3D4 CALL ..BYTE ;GET FILE LENGTH
F34B SF MOV E,A ;SAVE IN E
F34C CD F3D4 CALL ..BYTE ;GET LOAD MSB
F34F F5 PUSH PSW ;SAVE IN STACK
F350 CD F3D4 CALL ..BYTE ;GET LOAD LSB
F353 E1 POP H ;H=MSB
F354 6F MOV L,A ;HL=LOAD ADDR
F355 CD F3D4 CALL ..BYTE ;GET FILE TYPE
F358 B7 DRA A ;TEST FILE TYPE
F359 78 MOV A,B ;GET CUE
F35A C1 POP B ;BC=BIAS
F35B CA F365 JZ ..R2 ;ABSOLUTE LOAD
F35E EB XCHG ;RELOCATE LOAD ADDR.
F35F E3 XTHL
F360 EB XCHG
F361 19 DAD D ;DO IT
F362 EB XCHG
F363 E3 XTHL
F364 EB XCHG ;HL=LOAD+BASE[1]
F365 1C ..R2: INR E ;TEST LENGTH
F366 1D DCR E ;ZERO?
F367 CA F3E7 JZ ..DONE
F36A 09 DAD B ;ADD BIAS TO LOAD
F36B C5 PUSH B ;SAVE BIAS
F36C 47 MOV B,A ;SET-UP B
F36D 3D DCR A ;TEST CUE CUE
F36E CA F386 JZ ..R6 ;Z=REL. FILE, NZ=ABS.
F371 CD F3D4 ..R3: CALL ..BYTE ;GET NEXT DATA BYTE
F374 77 MOV M,A ;WRITE TO MEMORY
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00

APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979

COPYRIGHT 1979 BY APPLEZAP CORP.

F375 23 INX H ;BUMP UP LOAD POINT
F376 1D DCR E ;BUMP DOWN BYTE COUNT
F377 C2 F371 JNZ .R3 ;CONTINUE
F37A CD F3D4 ..R4: CALL ..BYTE ;TEST CHECKSUM
F37D CA F33C JZ ..R1 ;OK, CONTINUE W/NEXT
F380 CD F665 ..R5: CALL LADR ;ELSE PRINT LOAD ADDR
F383 C3 F08A JMP ERROR ;& ABORT
F386 CD F3BE ..R6: CALL ..R10 ;GET NEXT DATA BYTE
F389 77 MOV M,A ;STORE IT
F38A D2 F3B6 JNC ..R9 ;NORMAL BYTE
F38D E5 PUSH H ;CARRY=RELOCATE NEXT WORD
F3BE 21 0005 LXI H,5 ;POINT TO BASE[13
F391 39 DAD SP ;IN STACK
F392 CD F3BE ..R7: CALL ..R10 ;GET HIGH BYTE
F395 D2 F3A5 JNC ..R8 ;USE BASE[13
F399 1D DCR E ;COUNT EXTRA BYTE
F39A E3 XTHL ;GET LOAD ADDR
F39B 35 DCR M ;TEST FOR BASE[13
F39C E3 MOV M,A ;NEW LOW BYTE
F39D CA F392 XTHL ;SAVE LOAD AGAIN
F3A0 23 INX H ;BASE[1]
F3A1 23 INX H ;POINT TO BASE[2]
F3A2 C3 F392 JMP ..R7 ;AND TRY AGAIN
F3A5 86 ..R8: ADD M ;ADD IN MSB
F3A6 E3 XTHL ;STICK AT LOAD+1
F3A7 23 INX H ;GET LOAD BYTE
F3A8 77 MOV M,A ;STORE IT
F3A9 2B DCX H ;GET MSB
F3AA 7E MOV A,M ;IN A
F3AB E3 XTHL ;RELOCATE LSB
F3AC 2B DCX H ;GET LOAD ADDR
F3AD 86 ADD M ;STORE IT
F3AE 1E POP H ;GET MSB
F3AF 77 MOV M,A ;IN A
F3B0 23 INX H ;CARRY=RELOCATE NEXT WORD
F3B1 7E MOV A,M ;STORE IT
F3B2 CE00 ACI O ;MORE?
F3B4 77 MOV M,A ;COUNT IT
F3B5 1D DCR E ;NEXT IS DATA BYTE
F3B6 23 ..R9: INX H ;GET RELOC. MAP
F3B7 1D DCR E ;NEXT IS DATA BYTE
F3B8 C2 F386 JNZ ..R6 ;& CONTINUE
F3BB C3 F37A JMP ..R4 ;TEST CHECKSUM
F3BE 05 ..R10: DCR B ;COUNT BITS/BYTES
F3BF C2 F3C9 JNZ ..R11 ;NEXT IS DATA BYTE
F3C2 CD F3D4 CALL ..BYTE ;GET RELOC. MAP
F3C5 1D DCR E ;BUMP DOWN BYTE COUNT
F3C6 4F MOV C,A ;MAP IN C
F3C7 0608 MVI B,8 ;RESET FOR NEXT 8
F3C9 CD F3D4 ..R11: CALL ..BYTE ;NEXT DATA BYTE
F3CC D5 PUSH D ;SAVE DE
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00
APPLE MONITOR, ECT ROM* V1.0 JAN 07, 1979
COPYRIGHT 1979 BY APPLEZAP CORP.

F3CD 57 MOV D,A ;SAVE DATA BYTE
F3CE 79 MOV A,C ;TEST FOR RELOC.
F3CF 17 RAL ;IN CARRY FLAG
F3D0 4F MOV C,A ;UPDATE C
F3D1 7A MOV A,D ;RESTORE DATA BYTE
F3D2 D1 POP D ;RESTORE DE
F3D3 C9 RET ;CONTINUE

F3D4 C5 ..BYTE: PUSH B ;SAVE BC
F3D5 CD F68A CALL RIBBLE ;GET A CONVERTED CHAR.
F3D6 07 RLC
F3D9 07 RLC
F3DA 07 RLC
F3DB 07 RLC
F3DC 4F MOV C,A ;SAVE IT
F3DD CD F68A CALL RIBBLE ;GET OTHER HALF
F3E0 B1 ORA C ;MAKE WHOLE
F3E1 4F MOV C,A ;SAVE IN C
F3E2 82 ADD D ;UPDATE CHECKSUM
F3E3 57 MOV D,A ;NEW CHECKSUM
F3E4 79 MOV A,C ;RESTORE DATA BYTE
F3E5 C1 POP B ;RESTORE BC
F3E6 C9 RET ;CONTINUE

F3E7 C1 ..DONE: POP B ;BASE[1]
F3E8 C1 POP B ;BASE[2]
F3E9 7C MOV A,H ;TEST EOF
F3EA B5 ORA L ;FOR ZERO
F3EB C8 RZ
F3EC EB XCHG ;ELSE STORE IT IN 'P'
F3ED 21 0015 LXI H,PLOC
F3F0 39 DAD SP
F3F1 72 MOV M,D ;IN 'EXIT' TEMPLATE
F3F2 2B DCX H
F3F3 73 MOV M,E
F3F4 C9 RET ;REALLY DONE.

0006 ;XR=.READ

F3F5 CD F623 SUBS: CALL EXPR
F3F8 E1 POP H ;QUIT
F3F9 D8 RC
F3FA 7E ..SO: MOV A,M
F3FB CD F66A CALL LBYTE
F3FE CD F681 CALL COPCK
F401 D8 RC
F402 CA F412 JZ ..S1
F405 FE5F CPI ..''
F407 CA F41C JZ ..S3
F40A E5 PUSH H
F40B CD F626 CALL EXF
F40E D1 POP D
F40F E1 POP H
F410 73 MOV M,E
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00
APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979
COPYRIGHT 1979 BY APPELZAP CORP.

```assembly
F411  DB  RC
F412  23  ..S1:  INX  H
F413  7D  ..S2:  MOV  A,L
F414  E607  ANI  7
F416  CC  F562  CZ  LFAADR
F419  C3  F3FA  JMP  ..S0
F41C  2B  ..S3:  DCX  H  ;BACK-UP
F41D  C3  F413  JMP  ..S2

002B  %S=-.-SUBS

F420  0E40  TYPE:  MVI  C,64  ;SET UP A DEFAULT
F422  CD  F5F7  CALL  EXPC
F425  F5  PUSH  PSW
F426  CD  F562  ..TO:  CALL  LFAADR
F429  F1  POP  PSW
F42A  F5  PUSH  PSW
F42B  47  MOV  B,A  ;RESET LENGTH
F42C  7E  ..T1:  MOV  A,M
F42D  E67F  ANI  7FH
F42F  FE20  CPI  ' '  ;TEST LOWER END
F431  D2  F436  JNC  ..T3
F434  3E2E  ..T2:  MVI  A,' '  ;PRINT PERIODS INSTEAD
F436  FE7D  ..T3:  CPI  7DH  ;TEST UPPER END
F438  D2  F434  JNC  ..T2
F43B  4F  MOV  C,A  ;PUT WHATEVER INTO C
F43C  CD  F56A  CALL  CO
F43F  CD  F64C  CALL  HILO
F442  DA  F64A  JC  PRET
F445  05  DCR  B
F446  C2  F42C  JNZ  ..T1
F449  C3  F426  JMP  ..TO

002C  %T=-.-TYPE

F44C  CD  F5F7  VERIFY:  CALL  EXPC
F44F  0A  ..VO:  LDA X  B
F450  D5  PUSH  D  ;SAVE END POINTER
F451  SE  MOV  E,M  ;GET MEMORY DATA
F452  BB  CMP  E  ;TEST FOR MATCH
F453  CA  F46A  JZ  ..V1  ;MATHES
F456  C5  PUSH  B
F457  47  MOV  B,A
F458  CD  F565  CALL  HLSP
F45B  7B  MOV  A,E  ;GET MISMATCH
F45C  CD  F66A  CALL  LBYTE  ;PRINT IT
F45F  CD  F568  CALL  BLK  ;SPACE OVER
F462  7B  MOV  A,B  ;GET OTHER MISMATCH
F463  CD  F66A  CALL  LBYTE  ;PRINT THAT TOO
F466  CD  F5CC  CALL  CRLF  ;PREPARE FOR ANOTHER
F469  C1  POP  B
F46A  D1  ..V1:  POP  D  ;RESTORE END POINTER
F46B  03  INX  B
F46C  CD  F646  CALL  HILOX
```
F4CB  05  DCR  B  $END OF TABLE?
F4CC  CA  F0BA  JZ  ERROR  $YES
F4CF  C3  F4C5  JMP  ..X0  $ELSE KEEP LOOKING
F4D2  CD  F568  ..X1:  CALL  BLK  $GET & PRINT REG(S)
F4D5  CD  F511  ..X2:  CALL  ..X8  $MODIFY?
F4D8  CD  F6B1  ..X3:  CALL  COPCK  $NO, DELIMITER ENTERED
F4DB  CA  F4F2  JZ  ..X5  $SAVE TABLE POINTER
F4DE  E5  PUSH  H  $SAVE FLAG TEST (B)
F4DF  C5  PUSH  B  $GET NEW VALUE
F4E0  CD  F626  CALL  EXP  $IN HL
F4E3  E1  POP  H  $B=FLAG BYTE
F4E4  C1  POP  B  $A=DELIMITER
F4E5  F5  PUSH  PSW  $L=LOW BYTE
F4E6  7D  MOV  A,L  $STORE IT
F4E7  12  STAX  D  $GET FLAG
F4E8  78  MOV  A,B  $TEST BIT 7
F4E9  17  RAL  $TEST BIT 7
F4EA  D2  F4F0  JNC  ..X4  $SINGLE BYTE
F4ED  13  INX  D  $ELSE
F4EE  7C  MOV  A,H  $SAVE
F4EF  12  STAX  D  $HIGH BYTE
F4F0  F1  POP  H  $GET DELIMITER
F4F2  E1  POP  PSW  $RESTORE TABLE POINTER
F4F3  7D  MOV  A,M  $CR=DONE
F4F4  7E  MOV  A,M  $END OF TABLE?
F4F5  7F  ORA  A  $TEST BIT 7
F4F6  C3  F4D5  JMP  ..X2  $YES, DONE
F4F9  CD  F5CC  ..X6:  CALL  CRIF  $ELSE CONTINUE
F4FC  CD  F568  ..X7:  CALL  BLK  $FULL REGISTER DISPLAY
F4FF  7E  MOV  A,M  $SPACE OVER
F500  B7  ORA  A  $GET REGISTER NAME
F501  F8  RM  $END OF TABLE?
F502  4F  MOV  C,A  $YES, RETURN
F503  FD  F56A  CALL  CO  $ELSE PRINT IDENTIFIER
F506  0E3D  MVI  C,='7  $ON CONSOLE
F508  CD  F56A  CALL  CO  $FOR READABILITY
F50B  CD  F511  CALL  ..X8  $GET & PRINT REG(S)
F50E  C3  F4FC  JMP  ..X7  $POINT TO DISPLACEMENT
F511  23  INX  H  $GET IT
F512  7E  MOV  A,M  $POINT TO NEXT IN TABLE
F513  23  INX  H  $SAVE IN DE
F514  E8  XCHG  $SAVE FOR FLAGS
F515  47  MOV  B,A  $KILL FLAGS
F516  E63F  ANI  3FH  $UP IN STACK
F517  2600  MOV  L,A  $CALCULATE DISPLACEMENT
F518  6F  MOV  H,O  $UP IN STACK
F519  39  BAD  $SP.
F51C  23  INX  H  $ADJUST FOR RET IN STACK
F51D  23  INX  H  $TEST FOR "M"
F51E  7B  MOV  A,B  $BIT 6
F51F  E640  ANI  40H  $NO, NOT "M"
F521  CA  F528  JZ  ..X9
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00
<APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979>
(C)RIGHT 1979 BY APPLEZAP CORP.

F524 7E MOV A,M ;ELSE GET "M" POINTER
F525 2B DCX H ; INSTEAD
F526 6E MOV L,M ; WHERE ELSE
F527 67 MOV A,H (WHERE)
F528 7E CALL LBYTE ; GET THE VALUE
F529 CD F66A CALL LBYTE ; AND PRINT IT
F52C EB XCHG LBYTE $ ; SWITCH POINTERS
F52D 7B MOV A,B ; TEST FLAG
F52E 17 RAL ; SINGLE OR DOUBLE?
F52F D0 RNC ; SINGLE
F530 1B DCX D ; DOUBLE
F531 1A LDAX D ; GET IT
F532 C3 F66A JMP LBYTE ; PRINT IT & RETURN

007B

F535 CD F793 QUERY: CALL TI ; SEE IF IN OR OUT
F538 21 001D LXI H,0LOC ; PRESET
F53B 39 ADD SP ; TO ROUTINE IN EXIT AREA
F53C E5 PUSH H ; FOR BOTH Routines
F53D FE4F CPI 'O' ; OUT?
F53F C2 F540 JNZ ..QI ; NO, MUST BE IN
F542 CD F5F7 CALL EXPC ; GET PORT & VALUE
F545 7B MOV A,E ; L=PORT E=VALUE
F546 40 MOV C,L
F547 E1 POP H
F548 71 MOV M,C
F549 2B DCX H
F54A 36D3 MVI M,(OUT)
F54C E9 PHCH ; DO IT & RETURN

F54D FE49 ..QI: CPI 'I'
F54F C2 F0BA JNZ ERROR
F552 CD F623 CALL EXPR
F555 C1 POP B
F556 21 F5B7 LXI H,BITS ; SET-UP A RETURN
F559 E3 XTDL
F55A 71 MOV M,C ; SET PORT NUMBER
F55B 2B DCX H
F55C 36DB MVI M,(IN) ; SET FOR INPUT
F55E E9 PHCH ; DO IT

002A

F55F CD F081 SIZE: CALL MEMSIZ
F562 CD F5CC LFADDR: CALL CRLF
F565 CD F665 HLSP: CALL LADR
F568 0E20 BLK: MVI C,''
F56A 3A FFFF CD: LDA -1
F56D E603 ANI CMSK
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00
<APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979>
©RIGHT 1979 BY APPLEZAP CORP.

PAGE 21

```
F56F CA F581 JZ CRTOUT
F572 3D DCR A ;
F573 C2 F58C JNZ COU
F576 DB02 TTYOUT: IN TTS
F578 E680 ANI TTYBE
F57A C2 F576 JNZ TTYOUT
F57D 79 MOV A,C
F57E D303 OUT TTO
F580 C9 RET
F581 DB00 CRTOUT: IN CRTS
F583 E680 ANI CRTBE
F585 C2 F581 JNZ CRTOUT
F588 79 MOV A,C
F589 D301 OUT CRTD
F58B C9 RET
F58C 3D COU: DCR A ;BATCH
F58D C2 F803 JNZ COLOC ;NO
F590 3A FFFF L0: LDA -1
F593 E6C0 ANI * LMSK
F595 CA F581 JZ CRTOUT ;USE MAIN CONSOLE
F598 FE40 CPI LCR
F59A CA F576 JZ TTYOUT ;USE PRINTER
F59D FE80 CPI LINE
F59F C2 F80F JNZ LULOC ;MUST BE USER DEFINED
                           ;ELSE USE DATA TRANSFER
F5A2 DB04 LNLOC: IN RCSS
F5A4 E680 ANI PCSBE
F5A6 C2 F5A2 JNZ LNLOC
F5A9 79 MOV A,C
F5AA D305 OUT PCSAS
F5AC C9 RET
F5AD E60F CONV: ANI 0FH
F5AF C690 ADI 90H
F5B1 27 DAA
F5B2 CE40 ACI 40H
F5B4 27 DAA
F5B5 4F MOV C,A
F5B6 C9 RET
F5B7 5F BITS: MOV E,A
F5B8 1608 MVI D,B
F5BA CD F568 CALL BLK
F5BD 78 ..BI: MOV A,E
F5BE 17 RAL
F5BF 5F MOV E,A
F5C0 3E00 MVI A,0
F5C2 CE30 ACI '0'
F5C4 4F MOV C,A
F5C5 CD F56A CALL CD
```
THIS ROUTINE WILL GET TWO PARAMETERS FROM THE KEYBOARD, AND RETURN WITH THE 'C' REGISTER IN A, & CARRY SET IF THE TERMINATOR WAS A CARRIAGE RETURN. OTHERWISE, IT WILL GET THE THIRD PARAMETER. IF THE THIRD PARAMETER IS NON-ZERO, IT WILL RETURN WITH THE THIRD PARAMETER IN 'A'. IF IT IS ZERO, IT WILL RETURN WITH THE DEFAULT PARAM. IN EITHER CASE, IF THREE PARAMETERS WERE ENTERED, IT WILL RETURN WITH THE CARRY CLEAR.

```
F5C8  15  DCR  D
F5C9  C2 F5BD  JNZ .BI
F5CC  E5  CRLF:  PUSH  H
F5CD  C5  PUSH  B  ;SAVE BC
F5CE  0605  MVI  B,5
F5D0  CD F0A3  CALL  TOM
F5D3  C1  POP  B
F5D4  E1  POP  H
F5D5  C9  RET
F5D6  3A FFFF  CSTS:  LDA  -1
F5D9  E603  ANI  # CMSK
F5DB  CA F5EE  JZ .CS1  ;CRT
F5DE  3D  DCR  A
F5DF  CA F5E7  JZ .CS0  ;TTY
F5E2  3D  DCR  A
F5E3  C8  RZ  ;BATCH MODE
F5E4  C3 F806  JMP  CSLOC ;USER
F5E7  DB02  ..CS0:  IN  TTS
F5E9  E601  ANI  TTYDA
F5EB  C3 F5F2  JMP .CS2
F5EE  DB00  ..CS1:  IN  CRTS
F5F0  E601  ANI  CRTDA
F5F2  3EFF  ..CS2:  MVI  A, TRUE
F5F4  C8  RZ
F5F5  2F  CMA
F5F6  C9  RET
F5F7  C5  EXPC:  PUSH  B  ;SAVE DEFAULT PARAMETER
F5F8  CD F623  CALL  EXPR  ;GET 1st.
F5FB  DA F0BA  JC  ERROR  ;CR ENTERED TOO SOON
F5FE  CD F623  CALL  EXPR  ;GET 2nd. PARAMETER
F601  D1  POP  D  ;2nd. IN DE
F602  E1  POP  H  ;1st. IN HL
F603  C1  POP  B  ;REMOVE DEFAULT
F604  E5  PUSH  H  ;SAVE 1st. PARAMETER
F605  79  MOV  A,C  ;USE DEFAULT
F606  DA F615  JC .E1  ;NO THIRD PARAMETER
F609  C5  PUSH  B  ;SAVE DEFAULT AGAIN
F60A  CD F623  CALL  EXPR  ;GET 3rd. PARAMETER
```
01/07/79 22:40:00
APPLE MONITOR, 8ECT ROM* V1.0 JAN 07, 1979
Copyright 1979 by Applezap Corp.

ELECTRONIC CONTROL TECHNOLOGY

F60D  C1  POP  B  ;BC=TRUE 3rd. PARAMETER
F60E  79  MOV  A,C  ;TEST IT
F60F  E1  POP  H  ;HL=DEFAULT
F610  B7  ORA  A  ;TEST LOW BYTE
F611  C2  F615  JNZ  ..E1  ;OK, TAKE IT
F614  7D  MOV  A,L  ;ELSE USE DEFAULT
F615  E1  ..E1:  POP  H  ;GET 1st. PARAM
F616  F5  PUSH  PSW  ;SAVE ACC & FLAGS
F617  CD  F5CC  CALL  CRLF
F61A  F1  POP  PSW
F61B  C9  RET

; THIS ROUTINE RETURNS ONLY IF THREE PARAMETERS
; WERE ENTERED. LESS THAN THREE RESULTS IN AN
; ERROR CONDITION.

F61C  CD  F5F7  EXP3:  CALL  EXPC  ;GET THREE PARAMETERS
F61F  DA  F0BA  JC  ERROR  ;I SAID 3
F622  C9  RET

F623  CD  F793  EXPR:  CALL  TI  ;GET KEYBOARD
F626  21  0000  EXF:  LXI  H,0  ;INITIALIZE HL
F629  47  ..E1:  MOV  B,A  ;SAVE KEYBOARD
F62A  CD  F68D  CALL  NIBBLE  ;CONVERT ASCII TO HEX
F62D  DA  F63C  JC  ..E2  ;NOT LEGAL
F630  29  DAD  H  ;HL#16
F631  29  DAD  H
F632  29  DAD  H
F633  29  DAD  H
F634  B5  ORA  L  ;ADD IN NIBBLE
F635  6F  MOV  L,A
F636  CD  F793  CALL  TI  ;GET NEXT KEYBOARD
F639  C3  F629  JMP  ..E1  ;AND CONTINUE
F63C  E3  ..E2:  XTHL  ;STICK PARAMETER IN STACK
F63D  E5  PUSH  H  ;REPLACE RETURN
F63E  78  MOV  A,B  ;TEST CHARACTER
F63F  CD  F6B9  CALL  QCHK  ;FOR DELIMITERS
F642  C2  F0BA  JNZ  ERROR  ;ILLEGAL
F645  C9  RET

F646  CD  F64C  HILOX:  CALL  HILO  ;RETURN IF OK
F649  D0  RNC
F64A  D1  PRET:  POP  D  ;ELSE RETURN
F64B  C9  RET  ;ONE LEVEL BACK

F64C  23  HILO:  INX  H
F64D  7C  MOV  A,H
F64E  B5  ORA  L
F64F  37  STC
F650  C8  RZ
F651  7B  MOV  A,E
F652  95  SUB  L
F653  7A  MOV  A,D
F654  9C  SBB  H
```
F655 C9   RET
F656 CD F5F7 HEXN: CALL EXPC
F659 E5   PUSH H
F65A 19   DADD D
F65B CD F565 CALL HLSP
F65E E1   POP H
F65F 7D   MOV A,L
F660 93   SUB E
F661 6F   MOV L,A
F662 7C   MOV A,H
F663 9A   SBB D
F664 67   MOV H,A

000F

CALL
PUSH PSW

LADR: MOV A,H
CALL LBYTE
MOV A,L

LBYTE: PUSH PSW

C3 F66A CALL .L
POP PSW

F676 C3 F56A JMP CO

01 0BFF MARK: LXI B,08FFH ;Preset for rub-outs
JMP LEED

F67C C3 F682 LEAD: LXI B,4800H ;Preset for NULLs
CALL PO

F682 CD F6CC LEED: DCR B
JNZ LEED

F689 C9   RET

CD F77F RIBBLE: CALL RIX

D630 NIBBLE: SUI '0'

F68F DB RC
F690 FE17 CPI 'G'-'0'
F692 3F CMC
F693 D8 RC
F694 FE0A CPI 10
F696 3F CMC
F697 D0 RNC
F698 D607 SUI 'A'-'9'-1
F69C FE0A CPI 10

F699 C9   RET

7C PADR: MOV A,H
CD F6A2 CALL PBYTE
```
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00
APPLE MONITOR, ECT ROM, V1.0 JAN 07, 1979
RIGHT 1979 BY APPLEZAP CORP.

F6A1 7D MOV A,L
F6A2 F5 PBYTE: PUSH PSW
F6A3 0F RRC
F6A4 0F RRC
F6A5 0F RRC
F6A6 0F RRC
F6A7 CD F6AB CALL ..L
F6AA F1 POP PSW
F6AB CD F5AD ..L: CALL CONV
F6AE C3 F6CC JMP PO
F6B1 0E2D COPCK: MVI C,'-''
F6B3 CD F56A CALL CO
F6B6 CD F793 PCHK: CALL TI
F6B9 FE20 QCHK: CPI ','
F6BB C8 RZ
F6BC FE2C CPI','
F6BE C8 RZ
F6BF FE0D CPI CR
F6C1 37 STC
F6C2 C8 RZ
F6C3 3F CMC
F6C4 C9 RET
F6C5 0E0D PEOL: MVI C,CR
F6C7 CD F6CC CALL PO
F6CA 000A MVI C,LF
F6CC 3A FFFF PO: LDA -1
F6CF E630 ANI # PMSK
F6D1 CA F5A2 JZ LNLOC ;DATA XFER DEVICE
F6D4 FE10 CPI PTTY
F6D6 CA F576 JZ TTYOUT ;PRINTER DEVICE
F6D9 FE20 CPI FCAS
F6DB C2 F80C JNZ PULOC ;USER DEFINED
F6DE DB06 PTPL: IN PPSTAT ;PARALLEL PORT
F6EO E680 ANI PPBE
F6E2 C2 F6DE JNZ PTPL
F6E5 79 MOV A,C
F6E6 D307 OUT PPIDATA
F6E8 C9 RET
F6E9 CD F5F7 UNLD: CALL EXPC
F6EC CD F705 CALL WAIT
F6EF CD F67F CALL LEAD
F6F2 CD F679 CALL MARK
F6F5 4E .UI: MOV C,M
F6F6 CD F6CC CALL PO
F6F9 CD F64C CALL HILO
F6FC D2 F6FS JNC .UI
<table>
<thead>
<tr>
<th>F6FF</th>
<th>CD F679</th>
<th>CALL</th>
<th>MARK</th>
</tr>
</thead>
<tbody>
<tr>
<td>0019</td>
<td></td>
<td></td>
<td>ZU=--UNLD</td>
</tr>
<tr>
<td>F702</td>
<td>CD F67F</td>
<td>NULL</td>
<td>CALL</td>
</tr>
<tr>
<td>0003</td>
<td></td>
<td></td>
<td>ZN=--NULL</td>
</tr>
<tr>
<td>F705</td>
<td>3A FFFF</td>
<td>WAIT</td>
<td>LDA</td>
</tr>
<tr>
<td>F708</td>
<td>E603</td>
<td>ANI</td>
<td>CMSK</td>
</tr>
<tr>
<td>F70A</td>
<td>C8</td>
<td></td>
<td>RZ</td>
</tr>
<tr>
<td>F70B</td>
<td>3A FFFF</td>
<td>CI</td>
<td>LDA</td>
</tr>
<tr>
<td>F70E</td>
<td>E603</td>
<td>ANI</td>
<td>CMSK</td>
</tr>
<tr>
<td>F710</td>
<td>CA F721</td>
<td>JZ</td>
<td>CRTIN</td>
</tr>
<tr>
<td>F713</td>
<td>3D</td>
<td>DCR</td>
<td>A</td>
</tr>
<tr>
<td>F714</td>
<td>C2 F72B</td>
<td>JNZ</td>
<td>CIU</td>
</tr>
<tr>
<td>F717</td>
<td>DB02</td>
<td>TTYIN</td>
<td>IN</td>
</tr>
<tr>
<td>F719</td>
<td>E601</td>
<td>ANI</td>
<td>TTYDA</td>
</tr>
<tr>
<td>F71B</td>
<td>C2 F717</td>
<td>JNZ</td>
<td>TTYIN</td>
</tr>
<tr>
<td>F71E</td>
<td>DB03</td>
<td>IN</td>
<td>TTI</td>
</tr>
<tr>
<td>F720</td>
<td>C9</td>
<td>RET</td>
<td></td>
</tr>
<tr>
<td>F721</td>
<td>DB00</td>
<td>CRTIN</td>
<td>IN</td>
</tr>
<tr>
<td>F723</td>
<td>E601</td>
<td>ANI</td>
<td>CRTDA</td>
</tr>
<tr>
<td>F725</td>
<td>C2 F721</td>
<td>JNZ</td>
<td>CRTIN</td>
</tr>
<tr>
<td>F728</td>
<td>DB01</td>
<td>IN</td>
<td>CRTI</td>
</tr>
<tr>
<td>F72A</td>
<td>C9</td>
<td>RET</td>
<td></td>
</tr>
<tr>
<td>F72B</td>
<td>3D</td>
<td>CIU</td>
<td>DCR</td>
</tr>
<tr>
<td>F72C</td>
<td>C2 F800</td>
<td>JNZ</td>
<td>CILOC</td>
</tr>
<tr>
<td>F72F</td>
<td>3A FFFF</td>
<td>RI</td>
<td>LDA</td>
</tr>
<tr>
<td>F732</td>
<td>E60C</td>
<td>ANI</td>
<td>RMSK</td>
</tr>
<tr>
<td>F734</td>
<td>D302</td>
<td>OUT</td>
<td>TTS</td>
</tr>
<tr>
<td>F736</td>
<td>C2 F746</td>
<td>JNZ</td>
<td>R3</td>
</tr>
<tr>
<td>F739</td>
<td>CD F76A</td>
<td>DATA XFER</td>
<td>CALL</td>
</tr>
<tr>
<td>F73C</td>
<td>DB04</td>
<td>IN</td>
<td>RCSS</td>
</tr>
<tr>
<td>F73E</td>
<td>E601</td>
<td>ANI</td>
<td>RSCDA</td>
</tr>
<tr>
<td>F740</td>
<td>C2 F739</td>
<td>JNZ</td>
<td>R4</td>
</tr>
<tr>
<td>F743</td>
<td>DB05</td>
<td>IN</td>
<td>RCSD</td>
</tr>
<tr>
<td>F745</td>
<td>C9</td>
<td>RET</td>
<td></td>
</tr>
<tr>
<td>F746</td>
<td>FE04</td>
<td>.R3</td>
<td>CPI</td>
</tr>
<tr>
<td>F748</td>
<td>C2 F758</td>
<td>.RS</td>
<td>JNZ</td>
</tr>
<tr>
<td>F74B</td>
<td>CD F76A</td>
<td>.R1</td>
<td>CALL</td>
</tr>
<tr>
<td>F74E</td>
<td>DB02</td>
<td>IN</td>
<td>TTS</td>
</tr>
<tr>
<td>F750</td>
<td>E601</td>
<td>ANI</td>
<td>TTYDA</td>
</tr>
<tr>
<td>F752</td>
<td>C2 F74B</td>
<td>JNZ</td>
<td>R1</td>
</tr>
<tr>
<td>F755</td>
<td>DB03</td>
<td>IN</td>
<td>TTI</td>
</tr>
<tr>
<td>F757</td>
<td>C9</td>
<td>RET</td>
<td></td>
</tr>
</tbody>
</table>
ELECTRONIC CONTROL TECHNOLOGY

01/07/79 22:40:00
APPLE MONITOR, ECT ROM* V1.0 JAN 07, 1979
COPYRIGHT 1979 BY APPLEZAP CORP.

PAGE 27

F758 FE08 ..R5: CPI RCAS
F75A C2 F809 JNZ RULOC ;USER DEFINED

;PARALLEL PORT
F75D CD F76A ..R6: CALL ..R2
F760 DB06 IN PPSTAT
F762 E601 ANI PPDA
F764 C2 F75D JNZ ..R6
F767 DB07 IN PPDATA
F769 C9 RET

F76A 3A FFFF ..R2: LDA -1 ;MAKE SURE CONSOLE=0
F76D E603 ANI ~ CMSK
F76F C0 RNZ
F770 CD F5D6 CALL CSTS ;ANYTHING WAITING THERE?
F773 B7 ORA A
F774 C8 RZ ;NO, CONTINUE
F775 CD F78D CALL KI ;ELSE GET IT
F778 FE03 CPI 3 ;CONTROL-C?
F77A C0 RNZ
F77B F1 POP PSW ;ELSE RETURN
F77C AF XRA A ;WITH CARRY SET
F77D 37 STC
F77E C9 RET

F77F CD F785 RIX: CALL RIFF
F782 E67F ANI 7FH
F784 C9 RET

F785 CD F72F RIFF: CALL RI
F788 DA F0BA JC ERROR
F78B BA CMP D
F78C C9 RET

F78D CD F70B KI: CALL CI ;GET CONSOLE CHARACTER
F790 E67F ANI 7FH ;KILL PARITY BIT
F792 C9 RET

F793 CD F78D TI: CALL KI
F796 C8 RZ
F797 FE7F CPI 7FH
F799 C8 RZ ;TEST FOR RUB-OUT
F79A FE0D CPI CR ;IGNORE CR'S
F79C C8 RZ
F79D C5 PUSH B
F79E 4F MOV C,A
F79F CD F56A CALL CO
F7A2 79 MOV A,C
F7A3 C1 POP B
F7A4 FE40 CPI "A"-1 ;CONVERT TO UPPER CASE
F7A6 D8 RC
F7A7 FE7B CPI "Z"+1
F7A9 D0 RNC
F7AA E65F ANI 05FH
07/01/79 22:40:00
APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979
Copyright 1979 by Applezap Corp.

ELECTRONIC CONTROL TECHNOLOGY

PAGE 28

F7AC C9

RET

; <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.

F7AD LBL:

F7AD 43

.CODE 'C' ;CONSOLE ASSIGNMENTS

F7AE 43

.CODE 'C' ;CRT

F7AF 50

.CODE 'P' ;PRINTER

F7B0 42

.CODE 'B' ;BATCH= COMMANDS FROM READER

F7B1 55

.CODE 'U' ;USER

F7B2 52

.CODE 'R' ;READER ASSIGNMENTS

F7B3 44

.CODE 'D' ;DATA TRANSFER DEVICE

F7B4 50

.CODE 'P' ;PRINTER

F7B5 41

.CODE 'A' ;ALTERNATE (PARALLEL)

F7B6 55

.CODE 'U' ;USER

F7B7 50

.CODE 'P' ;PUNCH ASSIGNMENTS

F7B8 44

.CODE 'D' ;DATA TRANSFER DEVICE

F7B9 50

.CODE 'P' ;PRINTER

F7BA 41

.CODE 'A' ;ALTERNATE (PARALLEL)

F7BB 55

.CODE 'U' ;USER

F7BC 4C

.CODE 'L' ;LIST ASSIGNMENTS

F7BD 43

.CODE 'C' ;CRT

F7BE 50

.CODE 'P' ;PRINTER

F7BF 44

.CODE 'D' ;DATA TRANSFER DEVICE

F7C0 55

.CODE 'U' ;USER

F7C1 EXIT:

F7C1 D1

.POP D

F7C2 C1

.POP B

F7C3 F1

.POP PSW

F7C4 E1

.POP H

F7C5 F9

.SP HL

F7C6 00

.NOP

F7C7 21 0000

.LXI H, 0

F7C8 00

.HLX = .2

F7CA C3 0000

.JMP 0

F7CB

.PCX = .2
ELECTRONIC CONTROL TECHNOLOGY

PAGE 2

01/07/79 22:40:00
APPLE MONITOR, *ECT ROM* V1.0 JAN 07, 1979
© COPYRIGHT 1979 BY APPLEZAP CORP.

F7CD 0000 T1A: .WORD 0
F7CF 00 .BYTE 0
F7D0 0000 .WORD 0
F7D2 00 .BYTE 0
F7D3 QIO:
F7D3 DB00 IN 0
F7D5 C9 RET

F7D6 3 ENDX:

0007 ALOC = 7
0005 BLOC = 5
0004 CLOC = 4
0003 DLOC = 3
0002 ELOC = 2
0006 FLOC = 6
0012 HLOC = HLX-EXIT+11
000F LLOC = HLX-EXIT+8
0015 PLOC = PCX-EXIT+11
0009 SLOC = 9
0014 TLOC = T1A-EXIT+8
0016 TLOCX = TLOC+2
001D QLOC = QIO-EXIT+11

F7D6 4 ACTBL:
F7D6 4107 .BYTE 'A', ALOC
F7D8 4205 .BYTE 'B', BLOC
F7DA 4304 .BYTE 'C', CLOC
F7DC 4403 .BYTE 'D', DLOC
F7DE 4502 .BYTE 'E', ELOC
F7E0 4606 .BYTE 'F', FLOC
F7E2 4812 .BYTE 'H', HLOC
F7E4 4C11 .BYTE 'L', LLOC+2
F7E6 4D52 .BYTE 'M', HLOC 1040H
F7E8 5095 .BYTE 'P', PLOC 1080H
F7EA 5389 .BYTE 'S', SLOC 1080H

000B ACTSZ = (.-ACTBL)/2
F7EC FF .BYTE -1 ;TABLE DELIMITER
F7ED 525741 .ASCII 'RWA' ;AUTHOR
F7FF .ASCII '(C) 1979 ECT'

F7FF Z: ;END OF PROGRAM

F000 .END APPLE
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTBL</td>
<td>F7D6</td>
</tr>
<tr>
<td>ACTSZ</td>
<td>000B</td>
</tr>
<tr>
<td>ACTSZ</td>
<td>0000</td>
</tr>
<tr>
<td>ALOC</td>
<td>0007</td>
</tr>
<tr>
<td>BATCH</td>
<td>0002</td>
</tr>
<tr>
<td>BEGIN</td>
<td>0000</td>
</tr>
<tr>
<td>BELL</td>
<td>0007</td>
</tr>
<tr>
<td>BASE</td>
<td>F000</td>
</tr>
<tr>
<td>BITS</td>
<td>F5B7</td>
</tr>
<tr>
<td>BLK</td>
<td>F568</td>
</tr>
<tr>
<td>BRANCH</td>
<td>F1A6</td>
</tr>
<tr>
<td>CCHK</td>
<td>F0B4</td>
</tr>
<tr>
<td>CCRT</td>
<td>0000</td>
</tr>
<tr>
<td>CLOC</td>
<td>0004</td>
</tr>
<tr>
<td>CO</td>
<td>F56A</td>
</tr>
<tr>
<td>COLOC</td>
<td>F803</td>
</tr>
<tr>
<td>CONFIG</td>
<td>0000</td>
</tr>
<tr>
<td>CONV</td>
<td>F5AD</td>
</tr>
<tr>
<td>COPY</td>
<td>F681</td>
</tr>
<tr>
<td>CTIB</td>
<td>0000</td>
</tr>
<tr>
<td>CRTI</td>
<td>0001</td>
</tr>
<tr>
<td>CSR</td>
<td>0000</td>
</tr>
<tr>
<td>CTS</td>
<td>F5D6</td>
</tr>
<tr>
<td>CTTY</td>
<td>0001</td>
</tr>
<tr>
<td>CUSE</td>
<td>0003</td>
</tr>
<tr>
<td>DLOC</td>
<td>0003</td>
</tr>
<tr>
<td>EXF</td>
<td>F626</td>
</tr>
<tr>
<td>EXIT</td>
<td>F7C1</td>
</tr>
<tr>
<td>EXP</td>
<td>F5F7</td>
</tr>
<tr>
<td>EXPR</td>
<td>F623</td>
</tr>
<tr>
<td>FALSE</td>
<td>0000</td>
</tr>
<tr>
<td>FILL</td>
<td>F1F0</td>
</tr>
<tr>
<td>FLOC</td>
<td>0006</td>
</tr>
<tr>
<td>GOTO</td>
<td>F1FE</td>
</tr>
<tr>
<td>FILE</td>
<td>F64C</td>
</tr>
<tr>
<td>H FILE</td>
<td>F646</td>
</tr>
<tr>
<td>HLOC</td>
<td>0012</td>
</tr>
<tr>
<td>IHK</td>
<td>F7C8</td>
</tr>
<tr>
<td>I0</td>
<td>0000</td>
</tr>
<tr>
<td>I CHK</td>
<td>F1A2</td>
</tr>
<tr>
<td>J</td>
<td>F812</td>
</tr>
<tr>
<td>KI</td>
<td>F78D</td>
</tr>
<tr>
<td>L</td>
<td>LEAD</td>
</tr>
<tr>
<td>LINE</td>
<td>0000</td>
</tr>
<tr>
<td>LOAD</td>
<td>000F</td>
</tr>
<tr>
<td>LOAD</td>
<td>F267</td>
</tr>
<tr>
<td>LLOC</td>
<td>0080</td>
</tr>
<tr>
<td>LEED</td>
<td>F682</td>
</tr>
<tr>
<td>LINLOC</td>
<td>F5A2</td>
</tr>
<tr>
<td>LO</td>
<td>F590</td>
</tr>
<tr>
<td>LOAD</td>
<td>0008</td>
</tr>
<tr>
<td>LOG</td>
<td>F80F</td>
</tr>
<tr>
<td>LUSER</td>
<td>00C0</td>
</tr>
<tr>
<td>MAX</td>
<td>0007</td>
</tr>
<tr>
<td>MEHCK</td>
<td>F09A</td>
</tr>
<tr>
<td>MECN</td>
<td>00S7</td>
</tr>
<tr>
<td>MSGL</td>
<td>0013</td>
</tr>
<tr>
<td>MNB</td>
<td>F68D</td>
</tr>
<tr>
<td>PC</td>
<td>F6C5</td>
</tr>
<tr>
<td>PCASS</td>
<td>0004</td>
</tr>
<tr>
<td>PCHK</td>
<td>F686</td>
</tr>
<tr>
<td>PLOC</td>
<td>0015</td>
</tr>
<tr>
<td>PCX</td>
<td>F7CB</td>
</tr>
<tr>
<td>PEDL</td>
<td>F6C5</td>
</tr>
<tr>
<td>PD</td>
<td>F6CC</td>
</tr>
<tr>
<td>PPBE</td>
<td>0080</td>
</tr>
<tr>
<td>PPDATA</td>
<td>0007</td>
</tr>
<tr>
<td>PPSTAT</td>
<td>0006</td>
</tr>
<tr>
<td>PPTP</td>
<td>0000</td>
</tr>
<tr>
<td>PUSER</td>
<td>0000</td>
</tr>
<tr>
<td>PVT</td>
<td>0010</td>
</tr>
<tr>
<td>PLOC</td>
<td>000C</td>
</tr>
<tr>
<td>Q10</td>
<td>F689</td>
</tr>
<tr>
<td>Q2O</td>
<td>F7D3</td>
</tr>
<tr>
<td>QUERY</td>
<td>F535</td>
</tr>
<tr>
<td>RCAS</td>
<td>0008</td>
</tr>
<tr>
<td>RCSD</td>
<td>0005</td>
</tr>
<tr>
<td>RCSDA</td>
<td>0001</td>
</tr>
<tr>
<td>RCSS</td>
<td>0004</td>
</tr>
<tr>
<td>READ</td>
<td>F31F</td>
</tr>
<tr>
<td>R-I</td>
<td>F72F</td>
</tr>
<tr>
<td>RFF</td>
<td>F7B5</td>
</tr>
<tr>
<td>RTX</td>
<td>F77F</td>
</tr>
<tr>
<td>RMSK</td>
<td>00F3</td>
</tr>
<tr>
<td>RST</td>
<td>0038</td>
</tr>
<tr>
<td>RTTY</td>
<td>0004</td>
</tr>
<tr>
<td>RUB</td>
<td>00FF</td>
</tr>
<tr>
<td>RUSER</td>
<td>000C</td>
</tr>
<tr>
<td>SIZE</td>
<td>F55F</td>
</tr>
<tr>
<td>SLOC</td>
<td>0009</td>
</tr>
<tr>
<td>START</td>
<td>00FC</td>
</tr>
<tr>
<td>SUBS</td>
<td>F3F5</td>
</tr>
<tr>
<td>SUBS</td>
<td>0017</td>
</tr>
<tr>
<td>TIA</td>
<td>F7CD</td>
</tr>
<tr>
<td>TEST</td>
<td>F24C</td>
</tr>
<tr>
<td>TI</td>
<td>F793</td>
</tr>
<tr>
<td>TLOC</td>
<td>0014</td>
</tr>
<tr>
<td>TRAP</td>
<td>F01E</td>
</tr>
<tr>
<td>TP</td>
<td>0003</td>
</tr>
<tr>
<td>TTS</td>
<td>0002</td>
</tr>
<tr>
<td>TTYDA</td>
<td>0001</td>
</tr>
<tr>
<td>TTYIN</td>
<td>F717</td>
</tr>
<tr>
<td>TTYOUT</td>
<td>F576</td>
</tr>
<tr>
<td>UNL</td>
<td>F6E9</td>
</tr>
<tr>
<td>USER</td>
<td>F80B</td>
</tr>
<tr>
<td>WRITE</td>
<td>F472</td>
</tr>
<tr>
<td>Z</td>
<td>F7F0</td>
</tr>
<tr>
<td>XA</td>
<td>0040</td>
</tr>
<tr>
<td>XB</td>
<td>000E</td>
</tr>
<tr>
<td>ZB</td>
<td>000E</td>
</tr>
<tr>
<td>ZC</td>
<td>0000</td>
</tr>
<tr>
<td>ZD</td>
<td>0020</td>
</tr>
<tr>
<td>ZF</td>
<td>000E</td>
</tr>
<tr>
<td>ZG</td>
<td>004E</td>
</tr>
<tr>
<td>ZH</td>
<td>000F</td>
</tr>
<tr>
<td>ZI</td>
<td>001B</td>
</tr>
<tr>
<td>ZL</td>
<td>0046</td>
</tr>
<tr>
<td>ZK</td>
<td>000C</td>
</tr>
<tr>
<td>ZN</td>
<td>0022</td>
</tr>
<tr>
<td>ZR</td>
<td>002A</td>
</tr>
<tr>
<td>ZS</td>
<td>002B</td>
</tr>
<tr>
<td>ZT</td>
<td>002C</td>
</tr>
<tr>
<td>ZU</td>
<td>0019</td>
</tr>
<tr>
<td>ZV</td>
<td>0026</td>
</tr>
<tr>
<td>ZW</td>
<td>0048</td>
</tr>
<tr>
<td>ZX</td>
<td>007B</td>
</tr>
<tr>
<td>ZY</td>
<td>0044</td>
</tr>
</tbody>
</table>
APPLE V1.0 ECT PAGE 31

F000 C3 D8 F0 C3 OE F7 C3 2F F7 C3 6A F5 C3 CC F6 C3
F010 F0 F5 C3 D6 F5 C3 A2 F1 C3 9D F1 C3 9A F0 E5 D5
F020 C5 F5 11 EA FF 21 QA 00 39 06 04 EE 2E 72 2E 73
F030 D1 05 C2 2C F0 C1 OB F9 21 14 00 39 CD 7A F0 23
F040 23 C4 7A F0 CA 48 F0 03 21 0F 00 39 73 23 72 23
F050 23 71 23 70 C5 0E 40 CD 6A F5 E1 CD 65 F6 21 14
F060 00 39 11 02 00 2D 72 23 46 72 23 79 B0 CA 72 F0
F070 7E 02 23 1D C2 65 F0 C3 FC F0 7E 91 23 C0 7E 90
F080 C9 21 FF FF 24 7E 2F 77 BE 2F 77 C2 84 F0 24 7E
F090 2F 77 BE 2F 77 CA 6E F0 25 C9 E5 CD 81 F0 44 E1
F0A0 3E C0 C9 21 C5 F0 4E 23 CD 6A F5 05 C2 A6 F0 CD
F0B0 D6 F5 B7 C8 CD 8D F7 FE 03 C0 31 E2 FF 0E 2A CD
F0C0 6A F5 C3 FC F0 0D 0A 00 00 00 41 70 70 6C 65 20
F0D0 31 3E 30 20 45 43 54 21 EA FF F9 06 15 11 C1
F0E0 F7 1A 77 23 13 05 C2 E1 F0 CD 81 F0 F0 E5 60 E5 E5
F0F0 E5 E5 3E 00 32 FF FF 06 13 CD A3 F0 11 FC F0 D5
F100 CD CC F5 0E 3E CD 6A F5 21 29 F1 CD 93 F7 CA 0B
F110 F1 FE 20 DA 0B F1 D6 41 D8 FE 1A D0 87 85 6F 7E
F120 23 66 6F A4 3C CA BA F0 E9 5D F1 A6 F1 FF FF E4
F130 F1 D4 F1 F0 FE F1 56 F6 12 F8 4C F2 15 F8 E3
F140 F2 AD F2 02 F7 18 F8 B9 F2 35 F5 1F F3 F5 F3 20
F150 F4 E9 F6 4C F4 72 F4 BA F4 DB F2 5F F5 CD 93 F7
F160 21 AC F7 01 04 00 CD 86 F1 D5 CD 93 F7 D6 3D C2
F170 6A F1 4F CD 93 F7 CD 86 F1 F1 6A 26 03 3D FA 95
F180 F1 29 29 C3 7D F1 11 04 00 23 BE C8 09 14 1D C2
F190 89 F1 C3 BA F0 AC 67 CD A2 F1 A4 E5 4F 79 32 FF
F1A0 FF C9 3A FF FF C9 CD 93 F7 FE 2E C2 BA F0 21 80
F1B0 F8 C3 0E F1 0E 10 CD F7 F5 F5 CD 62 F5 F1 F5 47
F1C0 CD 68 F5 7E CD 6A F6 CD 4C F6 DA 4A F6 05 C2 C0
F1D0 F1 C3 BA F1 CD 23 F6 CD C5 F6 0E 3A CD CC F6 AF
F1E0 CD A2 F6 E1 CD 9D F6 21 00 00 CD 9D F6 C3 02 F7
F1F0 CD F7 F5 71 CD 4C F6 D2 F3 F1 D1 C3 FC F0 CD E6
F200 F6 CA 0F F2 CD 26 F6 D1 21 15 00 39 72 2B 73 FE
F210 0D CA 43 F2 16 02 21 16 00 39 E5 CD 23 F6 C1 E1
F220 F5 7B B1 CA 30 F2 71 23 70 23 0A 77 23 3E FF 02
F230 F1 DA 38 F2 15 C2 1A F2 3E C3 32 38 00 21 1E F0
F240 22 39 00 CD CC F5 D1 21 08 00 39 E9 CD F7 F5 7E
F250 47 2F 77 AE 70 CA 61 F2 D5 5F CD 65 F5 CD B8 F5
F260 D1 CD 4G 6F C3 4F F2 CD 23 F6 CD CC F5 E1 16 FF
F270 01 04 CD 85 F7 C2 70 F2 05 C2 73 F2 CD 85 F7
F280 CA 7D F2 77 CD 6A F5 23 CD 85 F7 CA 92 F2 77 C3
F290 87 F2 16 01 CD 85 F7 C2 A4 F2 1C 3E 07 BE C2 94
F2A0 F2 C3 62 F5 72 23 1D C2 A4 F2 C3 8E F2 CD F7 F5
F2B0 7E 02 03 CD 46 F6 C3 E0 F2 CD 23 F6 CD CC F5 E1
F2C0 CD 8D F7 FE 04 CA 62 F5 FE 7F CA D6 F2 77 4F 23
F2D0 CD 6A F5 C3 C0 F2 2B 4E C3 D0 F2 21 00 00 4D 39
F2E0 2E EB CD 23 F6 E1 65 E5 33 OC D2 E2 F2 F2 EB 51 E5
F2F0 01 00 00 C5 CD CC F5 C1 E1 5A 78 A1 3C C2 03 F3

ELECTRONIC CONTROL TECHNOLOGY
Apple V1.0 ECT

<table>
<thead>
<tr>
<th>Characters</th>
<th>Hexadecimal</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>F600</td>
<td>F6 D1 E1 C1 E5 79 DA 15 F6 C5 CD 23 F6 C1 79 E1</td>
<td></td>
</tr>
<tr>
<td>F610</td>
<td>F7 C2 15 F6 7D E1 F5 CD CC F5 F1 C9 CD F7 F5 DA</td>
<td></td>
</tr>
<tr>
<td>F620</td>
<td>EA F0 C9 CD 93 F7 21 00 00 47 CD 8D F6 DA 3C F6</td>
<td></td>
</tr>
<tr>
<td>F630</td>
<td>29 29 29 29 E5 6F CD 93 F7 C3 29 F6 E3 E5 7B CD</td>
<td></td>
</tr>
<tr>
<td>F640</td>
<td>B9 F6 C2 EA F0 C9 CD 4C F6 D0 D1 C9 23 7C E5 37</td>
<td></td>
</tr>
<tr>
<td>F650</td>
<td>C8 7B 95 7A 9C C9 CD F7 F5 E5 19 CD 65 F5 E1 7D</td>
<td></td>
</tr>
<tr>
<td>F660</td>
<td>93 6F 7C 9A 67 7C CD 6A F6 7D F5 0F 0F 0F CD</td>
<td></td>
</tr>
<tr>
<td>F670</td>
<td>73 F6 F1 CD AD F5 C3 6A F5 01 FF 08 C3 82 F6 01</td>
<td></td>
</tr>
<tr>
<td>F680</td>
<td>0F 48 CD CC F6 05 C2 82 F6 C9 CD 7F F7 D6 30 D8</td>
<td></td>
</tr>
<tr>
<td>F690</td>
<td>FE 17 3F D8 FE 0A 3F D0 D6 07 FE 0A C9 7C CD A2</td>
<td></td>
</tr>
<tr>
<td>F6A0</td>
<td>F6 7D F5 0F 0F 0F CD AB F6 F1 CD AD F5 C3 CC</td>
<td></td>
</tr>
<tr>
<td>F6B0</td>
<td>F6 0E 2D CD 6A F5 CD 93 F7 FE 20 C8 FE 2C C8 FE</td>
<td></td>
</tr>
<tr>
<td>F6C0</td>
<td>0D 37 C8 3F C9 0E 0D CD CC F6 0E 0A 3A FF FF E6</td>
<td></td>
</tr>
<tr>
<td>F6D0</td>
<td>30 CA A2 F5 FE 10 CA 76 F5 FE 20 C2 0C F8 DB 06</td>
<td></td>
</tr>
<tr>
<td>F6E0</td>
<td>E6 80 C2 DE F6 79 D3 07 C9 CD F7 F5 CD 05 F7 CD</td>
<td></td>
</tr>
<tr>
<td>F6F0</td>
<td>7F F6 CD 79 F6 4E CD DC F6 CD 4C F6 D2 F5 F6 CD</td>
<td></td>
</tr>
<tr>
<td>F700</td>
<td>79 F6 CD 7F F6 3A FF FF EE 03 C8 3A FF FF E6 03</td>
<td></td>
</tr>
<tr>
<td>F710</td>
<td>CA 21 F7 3D C2 2B F7 DB 02 E6 01 C2 17 F7 DB 03</td>
<td></td>
</tr>
<tr>
<td>F720</td>
<td>C9 DB 00 E6 01 C2 21 F7 DB 01 C9 3D C2 00 F8 3A</td>
<td></td>
</tr>
<tr>
<td>F730</td>
<td>FF FF E6 0C D3 02 C2 46 F7 CD 6A F7 DB 04 E6 01</td>
<td></td>
</tr>
<tr>
<td>F740</td>
<td>C2 39 F7 DB 05 C9 FE 04 C2 58 F7 CD 6A F7 DB 02</td>
<td></td>
</tr>
<tr>
<td>F750</td>
<td>E6 01 C2 4B F7 DB 03 C9 FE 08 C2 09 F8 CD 6A F7</td>
<td></td>
</tr>
<tr>
<td>F760</td>
<td>DB 06 E6 01 C2 5D F7 DB 07 C9 3A FF FF EE 03 C0</td>
<td></td>
</tr>
<tr>
<td>F770</td>
<td>CD D6 F5 B7 C8 CD 8D F7 FE 03 C0 F1 AF 37 C9 CD</td>
<td></td>
</tr>
<tr>
<td>F780</td>
<td>85 F7 E6 7F C9 CD 2F F7 DA EA F0 BA C9 CD 0E F7</td>
<td></td>
</tr>
<tr>
<td>F790</td>
<td>E6 7F C9 CD 8D F7 C8 FE 7F C8 FE 0D C8 C5 4F CD</td>
<td></td>
</tr>
<tr>
<td>F7A0</td>
<td>6A F5 79 C1 FE 40 D8 FE 7B D0 E6 5F C9 43 43 50</td>
<td></td>
</tr>
<tr>
<td>F7B0</td>
<td>42 55 52 44 50 41 55 50 44 50 41 55 4C 43 50 44</td>
<td></td>
</tr>
<tr>
<td>F7C0</td>
<td>55 D1 C1 F1 E1 F9 00 21 00 00 C3 00 00 00 00 00</td>
<td></td>
</tr>
<tr>
<td>F7D0</td>
<td>00 00 00 DB 00 C9 41 07 42 05 43 04 44 03 45 02</td>
<td></td>
</tr>
<tr>
<td>F7E0</td>
<td>46 06 48 12 4C 11 4D 52 50 95 53 89 FF 52 57 41</td>
<td></td>
</tr>
<tr>
<td>F7FC</td>
<td>0D 0A 28 43 29 20 31 39 37 39 20 45 43 54 0D 0A</td>
<td></td>
</tr>
</tbody>
</table>