

```

FFE2:99      731      DFB $99      ;BLANK
FFE3:        732 *
FFE3:        733 * Table of low order monitor routine dispatch
FFE3:        734 * addresses. High byte always $FE
FFE3:        735 *
FFE3:B2      736 SUBTBL DFB >BASCON-1 ;'C (BASIC warm start)
FFE4:C9      737 DFB >USR-1 ;'Y (not used)
FFE5:BE      738 DFB >REGZ-1 ;'E (open and display registers)
FFE6:F0      739 DFB >MINI-1 ;mini assembler
FFE7:35      740 DFB >VFY-1 ;V (memory verify)
FFE8:8C      741 DFB >INPRT-1 ;'K (IN#SLOT)
FFE9:D6      742 DFB >SEARCH-1 ;search for pattern
FFEA:96      743 DFB >OUTPRT-1 ;'P (PR#SLOT)
FFEB:AF      744 DFB >XBASIC-1 ;'B (BASIC cold start)
FFEC:17      745 DFB >SETMODE-1 ;'-' (subtraction)
FFED:17      746 DFB >SETMODE-1 ;'+' (addition)
FFEE:2B      747 DFB >MOVE-1 ;'M (memory move)
FFEF:1F      748 DFB >LT-1 ;'<' (delim for move,vfy)
FFFO:83      749 DFB >SETNORM-1 ;'N (set normal video)
FFFI:7F      750 DFB >SETINV-1 ;'I (set inverse video)
FFF2:5D      751 DFB >LIST-1 ;'L (disassemble 20 instrs)
FFF3:CC      752 DFB >WRITE-1 ;'W (write to tape)
FFF4:B5      753 DFB >GO-1 ;'G (execute program)
FFF5:FC      754 DFB >READ-1 ;'R (read from tape)
FFF6:17      755 DFB >SETMODE-1 ;'.' (memory fill)
FFF7:17      756 DFB >SETMODE-1 ;'.' (address delimiter)
FFF8:F5      757 DFB >CRMON-1 ;'CR' (end of input)
FFF9:03      758 DFB >BLANK-1 ;BLANK
FFFA:        759 *
FFFA:FB 03   760 DW NMI ;NON-MASKABLE INTERRUPT VECTOR
FFFC:62 FA   761 DW RESET ;RESET VECTOR
FFFE:FA C3   762 DW IRQ ;INTERRUPT REQUEST VECTOR
0000:        19 INCLUDE MINI
0000:        1 *
0000:        2 * Apple //e Mini Assembler
0000:        3 *
0000:        4 * Got mnemonic, check address mode
0000:        5 *
C4C8:        6 ORG C3ORG+$1C8
C4C8:        7 *
C4C8:20 13 FF 8 AMOD1 JSR NNBL ;get next non-blank
C4C8:84 34 9 STY YSAV ;save Y
C4CD:DD B4 F9 10 CMP CHAR1,X
C4DD:DD 13 C4E5 11 BNE AMOD2
C4D2:20 13 FF 12 JSR NNBL ;get next non-blank
C4D5:DD BA F9 13 CMP CHAR2,X
C4D8:F0 0D C4E7 14 BEQ AMOD3
C4DA:BD BA F9 15 LDA CHAR2,X ;done yet?
C4DD:F0 07 C4E6 16 BEQ AMOD4
C4DE:C9 A4 17 CMP #$A4 ;if "$" then done
C4E1:F0 03 C4E6 18 BEQ AMOD4
C4E3:A4 34 19 LDY YSAV ;restore Y
C4E5:18 20 AMOD2 CLC
C4E6:88 21 AMOD4 DEY

```

```

C4E7:26 44 22 AMOD3 ROL A5L ;shift bit into format
C4E9:E0 03 23 CPX #$03
C4EB:D0 0D C4FA 24 BNE AMOD6
C4ED:20 A7 FF 25 JSR GETNUM
C4F0:A5 3F 26 LDA A2H ;get high byte of address
C4F2:F0 01 C4F5 27 BEQ AMOD5 ;=>
C4F4:E8 28 INX
C4F5:86 35 29 AMOD5 STX YSAV1
C4F7:A2 03 30 LDX #$03
C4F9:88 31 DEY
C4FA:86 3D 32 AMOD6 STX A1H
C4FC:CA 33 DEX
C4FD:10 C9 C4C8 34 BPL AMOD1
C4FF:60 35 RTS
C500: 36 *
CF3A: CF3A 37 ORG C8ORG+$73A
CF3A: 38 *
CF3A: 39 * Calculate offset byte for relative addresses
CF3A: 40 *
CF3A:E9 81 41 REL SBC #$81 ;calc relative address
CF3C:4A 42 LSR A
CF3D:D0 14 CF53 43 BNE GOERR ;bad branch
CF3F:A4 3F 44 LDY A2H
CF41:A6 3E 45 LDX A2L
CF43:D0 01 CF46 46 BNE REL1
CF45:88 47 DEY ;point to offset
CF46:CA 48 REL1 DEX ;displacement - 1
CF47:8A 49 TXA
CF48:18 50 CLC
CF49:E5 3A 51 SBC PCL ;subtract current PCL
CF4B:85 3E 52 STA A2L ;and save as displacement
CF4D:10 01 CF50 53 BPL REL2 ;check page
CF4F:C8 54 INY
CF50:98 55 REL2 TYA ;get page
CF51:E5 3B 56 SBC PCH ;check page
CF53:D0 40 CF95 57 GOERR BNE MINERR ;display error
CF55: 58 *
CF55: 59 * Move instruction to memory
CF55: 60 *
CF55:A4 2F 61 MOVINST LDY LENGTH ;get instruction length
CF57:B9 3D 00 62 MOVI LDA A1H,Y ;get a byte
CF5A:91 3A 63 STA (PCL),Y ;and move it
CF5C:88 64 DEY
CF5D:10 F8 CF57 65 BPL MOVI
CF5F: 66 *
CF5F: 67 * Display instruction
CF5F: 68 *
CF5F:20 48 F9 69 JSR PRELNK ;print blanks to make ProDOS work
CF62:20 1A FC 70 JSR UP ;move up 2 lines
CF65:20 1A FC 71 JSR UP
CF68:4C E3 FC 72 JMP DISLIN ;disassemble it, =>DOINST
CF6B: 73 *
CF6B: 74 * Compare disassembly of all known opcodes with
CF6B: 75 * the one typed in until a match is found

```