## System Monitor Commands

Apple II contains a powerful machine level monitor for use by the advanced programmer. To enter the monitor either press RESET button on keyboard or CALL-151 (Hex FF65) from Basic. Apple II will respond with an "*" (asterisk) prompt character on the TV display. This action will not kill current BASIC program which may be re-entered by a $C^C$ (control C). NOTE: "adrs" is a four digit hexidecimal number and "data" is a two digit hexidecimal number. Remember to press "return" button at the end of each line.

| Command Format | Example | Description |
|---|---|---|
| **Examine Memory** | | |
| adrs | *CØF2 | Examines (displays) single memory location of (adrs) |
| adrs1.adrs2 | *1Ø24.1Ø48 | Examines (displays) range of memory from (adrs1) thru (adrs2) |
| (return) | *(return) | Examines (displays) next 8 memory locations. |
| .adrs2 | *.4Ø96 | Examines (displays) memory from current location through location (adrs2) |
| **Change Memory** | | |
| adrs:data data data | *A256:EF 2Ø 43 | Deposits data into memory starting at location (adrs). |
| :data data data | *:FØ A2 12 | Deposits data into memory starting after (adrs) last used for deposits. |
| **Move Memory** | | |
| adrs1<adrs2. adrs3M | *1ØØ<BØ1Ø.B41ØM | Copy the data now in the memory range from (adrs2) to (adrs3) into memory locations starting at (adrs1). |
| **Verify Memory** | | |
| adsr1<adrs2 adrs3V | *1ØØ<BØ1Ø.B41ØV | Verify that block of data in memory range from (adrs2) to (adrs3) exactly matches data block starting at memory location (adrs1)and displays differences if any. |

| Command Format | Example | Description |
|---|---|---|
| **Cassette I/O** | | |
| adrsl.adrs2R | *3ØØ.4FFR | Reads cassette data into specified memory (adrs) range. Record length must be same as memory range or an error will occur. |
| adrsl.adrs2W | *8ØØ.9FFW | Writes onto cassette data from specified memory (adrs) range. |
| **Display** | | |
| I | *I | Set inverse video mode. (Black characters on white background) |
| M | *N | Set normal video mode. (White characters on black background) |
| **Dis-assembler** | | |
| adrsL | *C8ØØL | Decodes 2Ø instructions starting at memory (adrs) into 65Ø2 assembly nmenonic code. |
| L | *L | Decodes next 2Ø instructions starting at current memory address. |
| **Mini-assembler** | | |
| (Turn-on) | *F666G | Turns-on mini-assembler. Prompt character is now a "!" (exclamation point). |
| $(monitor: command) | $C8ØØL | Executes any monitor command from mini-assembler then returns control to mini-assembler. Note that many monitor commands change current memory address reference so that it is good practice to retype desired address reference upon return to mini-assembler. |
| adrs:(65Ø2 MNEMONIC instruction) | !CØ1Ø:STA 23FF | Assembles a mnemonic 65Ø2 instruction into machine codes. If error, machine will refuse instruction, sound bell, and reprint line with up arrow under error. |

| Command Format | Example | Description |
|---|---|---|
| (space) (65Ø2 mnemonic instruction) | ! STA Ø1FF | Assembles instruction into next available memory location. (Note space between "f" and instruction) |
| (TURN-OFF) | ! (Reset Button) | Exits mini-assembler and returns to system monitor. |

## Monitor Program Execution and Debuging

| | | |
|---|---|---|
| adrsG | *3ØØG | Runs machine level program starting at memory (adrs). |
| adrsT | *8ØØT | Traces a program starting at memory location (adrs) and continues trace until hitting a breakpoint. Break occurs on instruction ØØ (BRK), and returns control to system monitor. Opens 65Ø2 status registers (see note 1) |
| asrdS | *CØ5ØS | Single steps through program beginning at memory location (adrs). Type a letter S for each additional step that you want displayed. Opens 65Ø2 status registers (see Note 1). |
| (Control E) | *E$^C$ | Displays 65Ø2 status registers and opens them for modification (see Note 1) |
| (Control Y) | *Y$^C$ | Executes user specified machine language subroutine starting at memory location (3F8). |

Note 1:

65Ø2 status registers are open if they are last line displayed on screen. To change them type ":" then "data" for each register.

```
Example:  A = 3C  X = FF  Y = ØØ  P = 32  S = F2
          *: FF               Changes A register only
          *:FF ØØ 33          Changes A, X, and Y registers
```

To change S register, you must first retype data for A, X, Y and P.

## Hexidecimal Arithmetic

| | | |
|---|---|---|
| data1+data2 | *78+34 | Performs hexidecimal sum of data1 plus data2. |
| data1-data2 | *AE-34 | Performs hexidecimal difference of data1 minus data2. |

| Command Format | Example | Description |
|---|---|---|
| **Set Input/Output Ports** | | |
| (X) (Control P) | *5PC | Sets printer output to I/O slot number (X). (see Note 2 below) |
| (X) (Control K) | *2KC | Sets keyboard input to I/O slot number (X). (see Note 2 below) |

Note 2:

Only slots 1 through 7 are addressable in this mode. Address Ø (Ex: ØP$^{C}$ or ØK$^{C}$) resets ports to internal video display and keyboard. These commands will not work unless Apple II interfaces are plugged into specificed I/O slot.

## Multiple Commands

| Command Format | Example | Description |
|---|---|---|
| | *1ØØL 4ØØG AFFT | Multiple monitor commands may be given on same line if separated by a "space". |
| | *LLLL | Single letter commands may be repeated without spaces. |

## SPECIAL CONTROL AND EDITING CHARACTERS

"Control" characters are indicated by a super-scripted "C" such as $G^C$. They are obtained by holding down the CTRL key while typing the specified letter. Control characters are NOT displayed on the TV screen. $B^C$ and $C^C$ must be followed by a carriage return. Screen editing characters are indicated by a sub-scripted "E" such as $D_C$. They are obtained by pressing and releasing the ESC key then typing specified letter. Edit characters send information only to display screen and does not send data to memory. For example, $U^C$ moves to cursor to right and copies text while $A_E$ moves cursor to right but does not copy text.

| CHARACTER | DESCRIPTION OF ACTION |
|---|---|
| RESET key | Immediately interrupts any program execution and resets computer. Also sets all text mode with scrolling window at maximum. Control is transferred to System Monitor and Apple prompts with a "*" (asterisk) and a bell. Hitting RESET key does NOT destroy existing BASIC or machine language program. |
| Control B | If in System Monitor (as indicated by a "*"), a control B and a carriage return will transfer control to BASIC, scratching (killing) any existing BASIC program and set HIMEM: to maximum installed user memory and LOMEM: to 2048. |
| Control C | If in BASIC, halts program and displays line number where stop occurred*. Program may be continued with a CON command. If in System Monitor, (as indicated by "*"), control C and a carriage return will enter BASIC without killing current program. |
| Control G | Sounds bell (beeps speaker) |
| Control H | Backspaces cursor and deletes any overwritten characters from computer but not from screen. Apply supplied keyboards have special key "4-." on right side of keyboard that provides this functions without using control button. |
| Control J | Issues line feed only |
| Control V | Compliment to $H^C$. Forward spaces cursor and copies over written characters. Apple keyboards have "+" key on right side which also performs this function. |
| Control X | Immediately deletes current line. |

* If BASIC program is expecting keyboard input, you will have to hit carriage return key after typing control C.

## SPECIAL CONTROL AND EDITING CHARACTERS

(continued)

| CHARACTER | DESCRIPTION OF ACTION |
|---|---|
| $A_E$ | Move cursor to right |
| $B_E$ | Move cursor to left |
| $C_E$ | Move cursor down |
| $D_E$ | Move cursor up |
| $E_E$ | Clear text from cursor to end of line |
| $F_E$ | Clear text from cursor to end of page |
| $@_E$ | Home cursor to top of page, clear text to end of page. |

## Special Controls and Features

### Display Mode Controls

| Hex | BASIC Example | Description |
|---|---|---|
| CO5Ø | 1Ø POKE -163Ø4,Ø | Set color graphics mode |
| CO51 | 2Ø POKE -163Ø3,Ø | Set text mode |
| CO52 | 3Ø POKE -163Ø2,Ø | Clear mixed graphics |
| CO53 | 4Ø POKE -163Ø1,Ø | Set mixed graphics (4 lines text) |
| CO54 | 5Ø POKE -163ØØ,Ø | Clear display Page 2 (BASIC commands use Page 1 only) |
| CO55 | 6Ø POKE -16299,Ø | Set display to Page 2 (alternate) |
| CO56 | 7Ø POKE -16298,Ø | Clear HIRES graphics mode |
| CO57 | 8Ø POKE -16297,Ø | Set HIRES graphics mode |

### TEXT Mode Controls

| Hex | BASIC Example | Description |
|---|---|---|
| ØØ2Ø | 9Ø POKE 32,L1 | Set left side of scrolling window to location specified by L1 in range of Ø to 39. |
| ØØ21 | 1ØØ POKE 33,W1 | Set window width to amount specified by W1. L1+W1<4Ø. W1>Ø |
| ØØ22 | 11Ø POKE 34,11 | Set window top to line specified by T1 in range of Ø to 23 |
| ØØ23 | 12Ø POKE 35,B1 | Set window bottom to line specified by B1 in the range of Ø to 23. B1>T1 |
| ØØ24 | 13Ø CH=PEEK(36) | Read/set cusor horizontal position in the range of Ø to 39. If using TAB, you must add "1" to cusor position read value; Ex. 14Ø and 15Ø perform identical function. |
| | 14Ø POKE 36,CH | |
| | 15Ø TAB(CH+1) | |
| ØØ25 | 16Ø CV=PEEK(37) | Similar to above. Read/set cusor vertical position in the range Ø to 23. |
| | 17Ø POKE 37,CV | |
| | 18Ø VTAB(CV+1) | |
| ØØ32 | 19Ø POKE 5Ø,127 | Set inverse flag if 127 (Ex. 19Ø) |
| | 2ØØ POKE 5Ø,255 | Set normal flag if 255(Ex. 2ØØ) |
| FC58 | 21Ø CALL -936 | ($@_E$) Home cusor, clear screen |
| FC42 | 22Ø CALL -958 | ($F_E$) Clear from cusor to end of page |

| Hex | BASIC Example | Description |
|---|---|---|
| FC9C | 23Ø CALL -868 | ($E_E$) Clear from cusor to end of line |
| FC66 | 24Ø CALL -922 | ($J^C$) Line feed |
| FC7Ø | 25Ø CALL -912 | Scroll up text one line |

Miscellaneous

| | | |
|---|---|---|
| CØ3Ø | 36Ø X=PEEK(-16336)<br>365 POKE -16336,Ø | Toggle speaker |
| CØØØ | 37Ø X=PEEK(-16384 | Read keyboard; if X>127 then key was pressed. |
| CØ1Ø | 38Ø POKE -16368,Ø | Clear keyboard strobe - always after reading keyboard. |
| CØ61 | 39Ø X=PEEK(16287) | Read PDL(Ø) push button switch. If X>127 then switch is "on". |
| CØ62 | 4ØØ X=PEEK(-16286) | Read PDL(1) push button switch. |
| CØ63 | 41Ø X=PEEK(-16285 | Read PDL(2) push button switch. |
| CØ58 | 42Ø POKE -16296,Ø | Clear Game I/O ANØ output |
| CØ59 | 43Ø POKE -16295,Ø | Set Game I/O ANØ output |
| CØ5A | 44Ø POKE -16294,Ø | Clear Game I/O AN1 output |
| CØ5B | 45Ø POKE -16293,Ø | Set Game I/O AN1 output |
| CØ5C | 46Ø POKE -16292,Ø | Clear Game I/O AN2 output |
| CØ5D | 47Ø POKE -16291,Ø | Set Game I/O AN2 output |
| CØ5E | 48Ø POKE -1629Ø,Ø | Clear Game I/O AN3 output |
| CØ5F | 49Ø POKE -16289,Ø | Set Game I/O AN3 output |

```
***************************
*                         *
*        APPLE II         *
*     SYSTEM MONITOR      *
*                         *
*    COPYRIGHT 1977 BY    *
*   APPLE COMPUTER, INC.  *
*                         *
*   ALL RIGHTS RESERVED   *
*                         *
*       S. WOZNIAK        *
*        A. BAUM          *
*                         *
***************************
        TITLE            "APPLE II SYSTEM MONITOR"
LOC0     EPZ   $00
LOC1     EPZ   $01
WNDLFT   EPZ   $20
WNDWDTH  EPZ   $21
WNDTOP   EPZ   $22
WNDBTM   EPZ   $23
CH       EPZ   $24
CV       EPZ   $25
GBASL    EPZ   $26
GBASH    EPZ   $27
BASL     EPZ   $28
BASH     EPZ   $29
BAS2L    EPZ   $2A
BAS2H    EPZ   $2B
H2       EPZ   $2C
LMNEM    EPZ   $2C
RTNL     EPZ   $2C
V2       EPZ   $2D
RMNEM    EPZ   $2D
RTNH     EPZ   $2D
MASK     EPZ   $2E
CHKSUM   EPZ   $2E
FORMAT   EPZ   $2E
LASTIN   EPZ   $2F
LENGTH   EPZ   $2F
SIGN     EPZ   $2F
COLOR    EPZ   $30
MODE     EPZ   $31
INVFLG   EPZ   $32
PROMPT   EPZ   $33
YSAV     EPZ   $34
YSAV1    EPZ   $35
CSWL     EPZ   $36
CSWH     EPZ   $37
KSWL     EPZ   $38
KSWH     EPZ   $39
PCL      EPZ   $3A
PCH      EPZ   $3B
XQT      EPZ   $3C
A1L      EPZ   $3C
A1H      EPZ   $3D
A2L      EPZ   $3E
A2H      EPZ   $3F
A3L      EPZ   $40
A3H      EPZ   $41
A4L      EPZ   $42
A4H      EPZ   $43
A5L      EPZ   $44
A5H      EPZ   $45
```

```
                ACC      EQU   $45
                XREG     EQU   $46
                YREG     EQU   $47
                STATUS   EQU   $48
                SPNT     EQU   $49
                RNDL     EQU   $4E
                RNDH     EQU   $4F
                ACL      EQU   $50
                ACH      EQU   $51
                XTNDL    EQU   $52
                XTNDH    EQU   $53
                AUXL     EQU   $54
                AUXH     EQU   $55
                PICK     EQU   $95
                IN       EQU   $0200
                USRADR   EQU   $03F8
                NMI      EQU   $03FB
                IRQLOC   EQU   $03FE
                IOADR    EQU   $C000
                KBD      EQU   $C000
                KBDSTRB  EQU   $C010
                TAPEOUT  EQU   $C020
                SPKR     EQU   $C030
                TXTCLR   EQU   $C050
                TXTSET   EQU   $C051
                MIXCLR   EQU   $C052
                MIXSET   EQU   $C053
                LOWSCR   EQU   $C054
                HISCR    EQU   $C055
                LORES    EQU   $C056
                HIRES    EQU   $C057
                TAPEIN   EQU   $C060
                PADDL0   EQU   $C064
                PTRIG    EQU   $C070
                BASIC    EQU   $E000
                BASIC2   EQU   $E003
                         ORG   $F800     ROM START ADDRESS
F800: 4A        PLOT     LSR             Y-COORD/2
F801: 08                 PHP             SAVE LSB IN CARRY
F802: 20 47 F8           JSR   GBASCALC  CALC BASE ADR IN GBASL,H
F805: 28                 PLP             RESTORE LSB FROM CARRY
F806: A9 0F              LDA   #$0F      MASK $0F IF EVEN
F808: 90 02              BCC   RTMASK
F80A: 69 E0              ADC   #$E0      MASK $F0 IF ODD
F80C: 85 2E     RTMASK   STA   MASK
F80E: B1 26     PLOT1    LDA   (GBASL),Y DATA
F810: 45 30              EOR   COLOR     EOR COLOR
F812: 25 2E              AND   MASK       AND MASK
F814: 51 26              EOR   (GBASL),Y    XOR DATA
F816: 91 26              STA   (GBASL),Y     TO DATA
F818: 60                 RTS
F819: 20 00 F8  HLINE    JSR   PLOT      PLOT SQUARE
F81C: C4 2C     HLINE1   CPY   H2        DONE?
F81E: B0 11              BCS   RTS1       YES, RETURN
F820: C8                 INY             NO, INCR INDEX (X-COORD)
F821: 20 0E F8           JSR   PLOT1     PLOT NEXT SQUARE
F824: 90 F6              BCC   HLINE1    ALWAYS TAKEN
F826: 69 01     VLINEZ   ADC   #$01      NEXT Y-COORD
F828: 48        VLINE    PHA             SAVE ON STACK
F829: 20 00 F8           JSR   PLOT      PLOT SQUARE
F82C: 68                 PLA
F82D: C5 2D              CMP   V2        DONE?
F82F: 90 F5              BCC   VLINEZ     NO, LOOP
F831: 60        RTS1     RTS
F832: A0 2F     CLRSCR   LDY   #$2F      MAX Y, FULL SCRN CLR
F834: D0 02              BNE   CLRSC2    ALWAYS TAKEN
F836: A0 27     CLRTOP   LDY   #$27      MAX Y, TOP SCREEN CLR
F838: 84 2D     CLRSC2   STY   V2        STORE AS BOTTOM COORD
                          FOR VLINE CALLS
F83A: A0 27              LDY   #$27      RIGHTMOST X-COORD (COLUMN)
F83C: A9 00     CLRSC3   LDA   #$00      TOP COORD FOR VLINE CALLS
F83E: 85 30              STA   COLOR     CLEAR COLOR (BLACK)
F840: 20 28 F8           JSR   VLINE     DRAW VLINE
F843: 88                 DEY             NEXT LEFTMOST X-COORD
F844: 10 F6              BPL   CLRSC3    LOOP UNTIL DONE
F846: 60                 RTS
F847: 48        GBASCALC PHA             FOR INPUT 000DEFGH
F848: 4A                 LSR
F849: 29 03              AND   #$03
F84B: 09 04              ORA   #$04        GENERATE GBASH=000001FG
F84D: 85 27              STA   GBASH
F84F: 68                 PLA             AND GBASL=HDEDE000
F850: 29 18              AND   #$18
F852: 90 02              BCC   GBCALC
F854: 69 7F              ADC   #$7F
F856: 85 26     GBCALC   STA   GBASL
```

```
F858: 0A                ASL   A
F859: 0A                ASL   A
F85A: 05 26             ORA   GBASL
F85C: 85 26             STA   GBASL
F85E: 60                RTS
F85F: A5 30     NXTCOL  LDA   COLOR      INCREMENT COLOR BY 3
F861: 18                CLC
F862: 69 03             ADC   #$03
F864: 29 0F     SETCOL  AND   #$0F       SETS COLOR=17*A MOD 16
F866: 85 30             STA   COLOR
F868: 0A                ASL   A          BOTH HALF BYTES OF COLOR EQUAL
F869: 0A                ASL   A
F86A: 0A                ASL   A
F86B: 0A                ASL   A
F86C: 05 30             ORA   COLOR
F86E: 85 30             STA   COLOR
F870: 60                RTS
F871: 4A        SCRN    LSR   A          READ SCREEN Y-COORD/2
F872: 08                PHP              SAVE LSB (CARRY)
F873: 20 47 F8          JSR   GBASCALC   CALC BASE ADDRESS
F876: B1 26             LDA   (GBASL),Y  GET BYTE
F878: 28                PLP              RESTORE LSB FROM CARRY
F879: 90 04     SCRN2   BCC   RTMSKZ     IF EVEN, USE LO H
F87B: 4A                LSR   A
F87C: 4A                LSR   A
F87D: 4A                LSR   A          SHIFT HIGH HALF BYTE DOWN
F87E: 4A                LSR   A
F87F: 29 0F     RTMSKZ  AND   #$0F       MASK 4-BITS
F881: 60                RTS
F882: A6 3A     INSDS1  LDX   PCL        PRINT PCL,H
F884: A4 3B             LDY   PCH
F886: 20 96 FD          JSR   PRYX2
F889: 20 48 F9          JSR   PRBLNK     FOLLOWED BY A BLANK
F88C: A1 3A             LDA   (PCL,X)    GET OP CODE
F88E: A8        INSDS2  TAY
F88F: 4A                LSR   A          EVEN/ODD TEST
F890: 90 09             BCC   IEVEN
F892: 6A                ROR              BIT 1 TEST
F893: B0 10             BCS   ERR        XXXXXX11 INVALID OP
F895: C9 A2             CMP   #$A2
F897: F0 0C             BEQ   ERR        OPCODE $89 INVALID
F899: 29 87             AND   #$87       MASK BITS
F89B: 4A        IEVEN   LSR   A          LSB INTO CARRY FOR L/R TEST
F89C: AA                TAX
F89D: BD 62 F9          LDA   FMT1,X     GET FORMAT INDEX BYTE
F8A0: 20 79 F8          JSR   SCRN2      R/L H-BYTE ON CARRY
F8A3: D0 04             BNE   GETFMT
F8A5: A0 80     ERR     LDY   #$80       SUBSTITUTE $80 FOR INVALID OPS
F8A7: A9 00             LDA   #$00       SET PRINT FORMAT INDEX TO 0
F8A9: AA        GETFMT  TAX
F8AA: BD A6 F9          LDA   FMT2,X     INDEX INTO PRINT FORMAT TABLE
F8AD: 85 2E             STA   FORMAT     SAVE FOR ADR FIELD FORMATTING
F8AF: 29 03             AND   #$03       MASK FOR 2-BIT LENGTH
                              (P=1 BYTE, 1=2 BYTE, 2=3 BYTE)
F8B1: 85 2F             STA   LENGTH
F8B3: 98                TYA              OPCODE
F8B4: 29 8F             AND   #$8F       MASK FOR 1XXX1010 TEST
F8B6: AA                TAX              SAVE IT
F8B7: 98                TYA              OPCODE TO A AGAIN
F8B8: A0 03             LDY   #$03
F8BA: E0 8A             CPX   #$8A
F8BC: F0 0B             BEQ   MNNDX3
F8BE: 4A        MNNDX1  LSR   A
F8BF: 90 08             BCC   MNNDX3     FORM INDEX INTO MNEMONIC TABLE
F8C1: 4A                LSR   A
F8C2: 4A        MNNDX2  LSR   A          1) 1XXX1010-&gt00101XXX
F8C3: 09 20             ORA   #$20          2) XXXYYY01-&gt00111XXX
F8C5: 88                DEY              3) XXXYYY10-&gt00110XXX
F8C6: D0 FA             BNE   MNNDX2        4) XXXYY100-&gt00100XXX
F8C8: C8                INY              5) XXXXX000-&gt000XXXXX
F8C9: 88        MNNDX3  DEY
F8CA: D0 F2             BNE   MNNDX1
F8CC: 60                RTS
F8CD: FF FF FF          DFB   $FF,$FF,$FF
F8D0: 20 82 F8  INSTDSP JSR   INSDS1     GEN FMT, LEN BYTES
F8D3: 48                PHA              SAVE MNEMONIC TABLE INDEX
F8D4: B1 3A     PRNTOP  LDA   (PCL),Y
F8D6: 20 DA FD          JSR   PRBYTE
F8D9: A2 01             LDX   #$01       PRINT 2 BLANKS
F8DB: 20 4A F9  PRNTBL  JSR   PRBL2
F8DE: C4 2F             CPY   LENGTH     PRINT INST (1-3 BYTES)
F8E0: C8                INY              IN A 12 CHR FIELD
F8E1: 90 F1             BCC   PRNTOP
F8E3: A2 03             LDX   #$03       CHAR COUNT FOR MNEMONIC PRINT
F8E5: C0 04             CPY   #$04
```

```
F8E7: 90 F2              BCC   PRNTBL
F8E9: 68                 PLA              RECOVER MNEMONIC INDEX
F8EA: A8                 TAY
F8EB: B9 C0 F9           LDA   MNEML,Y
F8EE: 85 2C              STA   LMNEM      FETCH 3-CHAR MNEMONIC
F8F0: B9 00 FA           LDA   MNEMR,Y      (PACKED IN 2-BYTES)
F8F3: 85 2D              STA   RMNEM
F8F5: A9 00     PRMN1    LDA   #$00
F8F7: A0 05              LDY   #$05
F8F9: 06 2D     PRMN2    ASL   RMNEM      SHIFT 5 BITS OF
F8FB: 26 2C              ROL   LMNEM        CHARACTER INTO A
F8FD: 2A                 ROL                  (CLEARS CARRY)
F8FE: 88                 DEY
F8FF: D0 F8              BNE   PRMN2
F901: 69 BF              ADC   #$BF       ADD "?" OFFSET
F903: 20 ED FD           JSR   COUT       OUTPUT A CHAR OF MNEM
F906: CA                 DEX
F907: D0 EC              BNE   PRMN1
F909: 20 48 F9           JSR   PRBLNK     OUTPUT 3 BLANKS
F90C: A4 2F              LDY   LENGTH
F90E: A2 06              LDX   #$06       CNT FOR 6 FORMAT BITS
F910: E0 03     PRADR1   CPX   #$03
F912: F0 1C              BEQ   PRADR5     IF X=3 THEN ADDR.
F914: 06 2E     PRADR2   ASL   FORMAT
F916: 90 0E              BCC   PRADR3
F918: BD B3 F9           LDA   CHAR1-1,X
F91B: 20 ED FD           JSR   COUT
F91E: BD B9 F9           LDA   CHAR2-1,X
F921: F0 03              BEQ   PRADR3
F923: 20 ED FD           JSR   COUT
F926: CA        PRADR3   DEX
F927: D0 E7              BNE   PRADR1
F929: 60                 RTS
F92A: 88        PRADR4   DEY
F92B: 30 E7              BMI   PRADR2
F92D: 20 DA FD           JSR   PRBYTE
F930: A5 2E     PRADR5   LDA   FORMAT
F932: C9 E8              CMP   #$E8       HANDLE REL ADR MODE
F934: B1 3A              LDA   (PCL),Y    SPECIAL (PRINT TARGET,
F936: 90 F2              BCC   PRADR4       NOT OFFSET)
F938: 20 56 F9  RELADR   JSR   PCADJ3
F93B: AA                 TAX              PCL,PCH+OFFSET+1 TO A,Y
F93C: E8                 INX
F93D: D0 01              BNE   PRNTYX     +1 TO Y,X
F93F: C8                 INY
F940: 98        PRNTYX   TYA
F941: 20 DA FD  PRNTAX   JSR   PRBYTE     OUTPUT TARGET ADR
F944: 8A        PRNTX    TXA                OF BRANCH AND RETURN
F945: 4C DA FD           JMP   PRBYTE
F948: A2 03     PRBLNK   LDX   #$03       BLANK COUNT
F94A: A9 A0     PRBL2    LDA   #$A0       LOAD A SPACE
F94C: 20 ED FD  PRBL3    JSR   COUT       OUTPUT A BLANK
F94F: CA                 DEX
F950: D0 F8              BNE   PRBL2      LOOP UNTIL COUNT=0
F952: 60                 RTS
F953: 38        PCADJ    SEC              0=1-BYTE, 1=2-BYTE
F954: A5 2F     PCADJ2   LDA   LENGTH       2=3-BYTE
F956: A4 3B     PCADJ3   LDY   PCH
F958: AA                 TAX              TEST DISPLACEMENT SIGN
F959: 10 01              BPL   PCADJ4       (FOR REL BRANCH)
F95B: 88                 DEY              EXTEND NEG BY DEC PCH
F95C: 65 3A     PCADJ4   ADC   PCL
F95E: 90 01              BCC   RTS2       PCL+LENGTH(OR DISPL)+1 TO A
F960: C8                 INY                CARRY INTO Y (PCH)
F961: 60        RTS2     RTS
                *         FMT1 BYTES:               XXXXXXY0 INSTRS
                *         IF Y=0                    THEN LEFT HALF BYTE
                *         IF Y=1                    THEN RIGHT HALF BYTE
                *                                        (X=INDEX)
F962: 04 20 54
F965: 30 0D     FMT1     DFB   $04,$20,$54,$30,$0D
F967: 80 04 90
F96A: 03 22              DFB   $80,$04,$90,$03,$22
F96C: 54 33 0D
F96F: 80 04              DFB   $54,$33,$0D,$80,$04
F971: 90 04 20
F974: 54 33              DFB   $90,$04,$20,$54,$33
F976: 0D 80 04
F979: 90 04              DFB   $0D,$80,$04,$90,$04
F97B: 20 54 3B
F97E: 0D 80              DFB   $20,$54,$3B,$0D,$80
F980: 04 90 00
F983: 22 44              DFB   $04,$90,$00,$22,$44
F985: 33 0D C8
F988: 44 00              DFB   $33,$0D,$C8,$44,$00
```

```
F98A: 11 22 44
F98D: 33 0D            DFB   $11,$22,$44,$33,$0D
F98F: C8 44 A9
F992: 01 22            DFB   $C8,$44,$A9,$01,$22
F994: 44 33 0D
F997: 80 04            DFB   $44,$33,$0D,$80,$04
F999: 90 01 22
F99C: 44 33            DFB   $90,$01,$22,$44,$33
F99E: 0D 80 04
F9A1: 90               DFB   $0D,$80,$04,$90
F9A2: 26 31 87
F9A5: 9A               DFB   $26,$31,$87,$9A $ZZXXXY01 INSTR'S
F9A6: 00        FMT2   DFB   $00      ERR
F9A7: 21               DFB   $21      IMM
F9A8: 81               DFB   $81      Z-PAGE
F9A9: 82               DFB   $82      ABS
F9AA: 00               DFB   $00      IMPLIED
F9AB: 00               DFB   $00      ACCUMULATOR
F9AC: 59               DFB   $59      (ZPAG,X)
F9AD: 4D               DFB   $4D      (ZPAG),Y
F9AE: 91               DFB   $91      ZPAG,X
F9AF: 92               DFB   $92      ABS,X
F9B0: 86               DFB   $86      ABS,Y
F9B1: 4A               DFB   $4A      (ABS)
F9B2: 85               DFB   $85      ZPAG,Y
F9B3: 9D               DFB   $9D      RELATIVE
F9B4: AC A9 AC
F9B7: A3 A8 A4
                CHAR1  ASC   ",),#($"
F9BA: D9 00 D8
F9BD: A4 A4 00  CHAR2  DFB   $D9,$00,$D8,$A4,$A4,$00
                *CHAR2:  "Y",0,"X$$",0
                *        MNEML        IS OF FORM:
                *        (A) XXXXX000
                *        (B) XXXYY100
                *        (C) 1XXX1010
                *        (D) XXXYYY10
                *        (E) XXXYYY01
                *            (X=INDEX)
F9C0: 1C 8A 1C
F9C3: 23 5D 8B  MNEML  DFB   $1C,$8A,$1C,$23,$5D,$
F9C6: 1B A1 9D
F9C9: 8A 1D 23         DFB   $1B,$A1,$9D,$8A,$1D,$23
F9CC: 9D 8B 1D
F9CF: A1 00 29         DFB   $9D,$8B,$1D,$A1,$00,$29
F9D2: 19 AE 69
F9D5: A8 19 23         DFB   $19,$AE,$69,$A8,$19,$23
F9D8: 24 53 1B
F9DB: 23 24 53         DFB   $24,$53,$1B,$23,$24,$53
F9DE: 19 A1            DFB   $19,$A1    (A) FORMAT ABOVE
F9E0: 00 1A 5B
F9E3: 5B A5 69         DFB   $00,$1A,$5B,$5B,$A5,$69
F9E6: 24 24            DFB   $24,$24    (B) FORMAT
F9E8: AE AE A8
F9EB: AD 29 00         DFB   $AE,$AE,$A8,$AD,$29,$00
F9EE: 7C 00            DFB   $7C,$00    (C) FORMAT
F9F0: 15 9C 6D
F9F3: 9C A5 69         DFB   $15,$9C,$6D,$9C,$A5,$69
F9F6: 29 53            DFB   $29,$53    (D) FORMAT
F9F8: 84 13 34
F9FB: 11 A5 69         DFB   $84,$13,$34,$11,$A5,$69
F9FE: 23 A0            DFB   $23,$A0    (E) FORMAT
FA00: D8 62 5A
FA03: 48 26 62  MNEMR  DFB   $D8,$62,$5A,$48,$26,$62
FA06: 94 88 54
FA09: 44 C8 54         DFB   $94,$88,$54,$44,$C8,$54
FA0C: 68 44 E8
FA0F: 94 00 B4         DFB   $68,$44,$E8,$94,$00,$B4
FA12: 08 84 74
FA15: B4 28 6E         DFB   $08,$84,$74,$B4,$28,$6E
FA18: 74 F4 CC
FA1B: 4A 72 F2         DFB   $74,$F4,$CC,$4A,$72,$F2
FA1E: A4 8A            DFB   $A4,$8A    (A) FORMAT
FA20: 00 AA A2
FA23: A2 74 74         DFB   $00,$AA,$A2,$A2,$74,$74
FA26: 74 72            DFB   $74,$72    (B) FORMAT
FA28: 44 68 B2
FA2B: 32 B2 00         DFB   $44,$68,$B2,$32,$B2,$00
FA2E: 22 00            DFB   $22,$00    (C) FORMAT
FA30: 1A 1A 26
FA33: 26 72 72         DFB   $1A,$1A,$26,$26,$72,$72
FA36: 88 C8            DFB   $88,$C8    (D) FORMAT
FA38: C4 CA 26
FA3B: 48 44 44         DFB   $C4,$CA,$26,$48,$44,$44
FA3E: A2 C8            DFB   $A2,$C8    (E) FORMAT
```

```
FA40: FF FF FF          DFB   $FF,$FF,$FF
FA43: 20 D0 F8  STEP    JSR   INSTDSP    DISASSEMBLE ONE INST
FA46: 68                PLA              AT (PCL,H)
FA47: 85 2C             STA   RTNL       ADJUST TO USER
FA49: 68                PLA              STACK. SAVE
FA4A: 85 2D             STA   RTNH         RTN ADR.
FA4C: A2 08             LDX   #$08
FA4E: BD 10 FB  XQINIT  LDA   INITBL-1,X INIT XEQ AREA
FA51: 95 3C             STA   XQT,X
FA53: CA                DEX
FA54: D0 F8             BNE   XQINIT
FA56: A1 3A             LDA   (PCL,X)    USER OPCODE BYTE
FA58: F0 42             BEQ   XBRK       SPECIAL IF BREAK
FA5A: A4 2F             LDY   LENGTH     LEN FROM DISASSEMBLY
FA5C: C9 20             CMP   #$20
FA5E: F0 59             BEQ   XJSR       HANDLE JSR, RTS, JMP,
FA60: C9 60             CMP   #$60         JMP (), RTI SPECIAL
FA62: F0 45             BEQ   XRTS
FA64: C9 4C             CMP   #$4C
FA66: F0 5C             BEQ   XJMP
FA68: C9 6C             CMP   #$6C
FA6A: F0 59             BEQ   XJMPAT
FA6C: C9 40             CMP   #$40
FA6E: F0 35             BEQ   XRTI
FA70: 29 1F             AND   #$1F
FA72: 49 14             EOR   #$14
FA74: C9 04             CMP   #$04       COPY USER INST TO XEQ AREA
FA76: F0 02             BEQ   XQ2          WITH TRAILING NOPS
FA78: B1 3A     XQ1     LDA   (PCL),Y    CHANGE REL BRANCH
FA7A: 99 3C 00  XQ2     STA   XQT,Y        DISP TO 4 FOR
FA7D: 88                DEY                JMP TO BRANCH OR
FA7E: 10 F8             BPL   XQ1          NBRANCH FROM XEQ.
FA80: 20 3F FF          JSR   RESTORE    RESTORE USER REG CONTENTS.
FA83: 4C 3C 00          JMP   XQT        XEQ USER OP FROM RAM
FA86: 85 45     IRQ     STA   ACC          (RETURN TO NBRANCH)
FA88: 68                PLA
FA89: 48                PHA              **IRQ HANDLER
FA8A: 0A                ASL   A
FA8B: 0A                ASL   A
FA8C: 0A                ASL   A
FA8D: 30 03             BMI   BREAK      TEST FOR BREAK
FA8F: 6C FE 03          JMP   (IRQLOC)   USER ROUTINE VECTOR IN RAM
FA92: 28        BREAK   PLP
FA93: 20 4C FF          JSR   SAV1       SAVE REG'S ON BREAK
FA96: 68                PLA              INCLUDING PC
FA97: 85 3A             STA   PCL
FA99: 68                PLA
FA9A: 85 3B             STA   PCH
FA9C: 20 82 F8  XBRK    JSR   INSDS1     PRINT USER PC.
FA9F: 20 DA FA          JSR   RGDSP1       AND REG'S
FAA2: 4C 65 FF          JMP   MON        GO TO MONITOR
FAA5: 18        XRTI    CLC
FAA6: 68                PLA              SIMULATE RTI BY EXPECTING
FAA7: 85 48             STA   STATUS       STATUS FROM STACK, THEN RTS
FAA9: 68        XRTS    PLA              RTS SIMULATION
FAAA: 85 3A             STA   PCL          EXTRACT PC FROM STACK
FAAC: 68                PLA              AND UPDATE PC BY 1 (LEN=0)
FAAD: 85 3B     PCINC2  STA   PCH
FAAF: A5 2F     PCINC3  LDA   LENGTH     UPDATE PC BY LEN
FAB1: 20 56 F9          JSR   PCADJ3
FAB4: 84 3B             STY   PCH
FAB6: 18                CLC
FAB7: 90 14             BCC   NEWPCL
FAB9: 18        XJSR    CLC
FABA: 20 54 F9          JSR   PCADJ2     UPDATE PC AND PUSH
FABD: AA                TAX              ONTO STACH FOR
FABE: 98                TYA              JSR SIMULATE
FABF: 48                PHA
FAC0: 8A                TXA
FAC1: 48                PHA
FAC2: A0 02             LDY   #$02
FAC4: 18        XJMP    CLC
FAC5: B1 3A     XJMPAT  LDA   (PCL),Y
FAC7: AA                TAX              LOAD PC FOR JMP,
FAC8: 88                DEY              (JMP) SIMULATE.
FAC9: B1 3A             LDA   (PCL),Y
FACB: 86 3B             STX   PCH
FACD: 85 3A     NEWPCL  STA   PCL
FACF: B0 F3             BCS   XJMP
FAD1: A5 2D     RTNJMP  LDA   RTNH
FAD3: 48                PHA
FAD4: A5 2C             LDA   RTNL
FAD6: 48                PHA
FAD7: 20 8E FD  REGDSP  JSR   CROUT      DISPLAY USER REG
FADA: A9 45     RGDSP1  LDA   #ACC         CONTENTS WITH
FADC: 85 40             STA   A3L          LABELS
```

```
FADE: A9 00             LDA   #ACC/256
FAE0: 85 41             STA   A3H
FAE2: A2 FB             LDX   #$FB
FAE4: A9 A0     RDSP1   LDA   #$A0
FAE6: 20 ED FD          JSR   COUT
FAE9: BD 1E FA          LDA   RTBL-$FB,X
FAEC: 20 ED FD          JSR   COUT
FAEF: A9 BD             LDA   #$BD
FAF1: 20 ED FD          JSR   COUT
FAF4: B5 4A             LDA   ACC+5,X
FAF6: 20 DA FD          JSR   PRBYTE
FAF9: E8                INX
FAFA: 30 E8             BMI   RDSP1
FAFC: 60                RTS
FAFD: 18        BRANCH  CLC              BRANCH TAKEN,
FAFE: A0 01             LDY   #$01         ADD LEN+2 TO PC
FB00: B1 3A             LDA   (PCL),Y
FB02: 20 56 F9          JSR   PCADJ3
FB05: 85 3A             STA   PCL
FB07: 98                TYA
FB08: 38                SEC
FB09: B0 A2             BCS   PCINC2
FB0B: 20 4A FF  NBRNCH  JSR   SAVE       NORMAL RETURN AFTER
FB0E: 38                SEC                XEQ USER OF
FB0F: B0 9E             BCS   PCINC3     GO UPDATE PC
FB11: EA        INITBL  NOP
FB12: EA                NOP              DUMMY FILL FOR
FB13: 4C 0B FB          JMP   NBRNCH       XEQ AREA
FB16: 4C FD FA          JMP   BRANCH
FB19: C1        RTBL    DFB   $C1
FB1A: D8                DFB   $D8
FB1B: D9                DFB   $D9
FB1C: D0                DFB   $D0
FB1D: D3                DFB   $D3
FB1E: AD 70 C0  PREAD   LDA   PTRIG      TRIGGER PADDLES
FB21: A0 00             LDY   #$00       INIT COUNT
FB23: EA                NOP              COMPENSATE FOR 1ST COUNT
FB24: EA                NOP
FB25: BD 64 C0  PREAD2  LDA   PADDL0,X   COUNT Y-REG EVERY
FB28: 10 04             BPL   RTS2D        12 USEC
FB2A: C8                INY
FB2B: D0 F8             BNE   PREAD2       EXIT AT 255 MAX
FB2D: 88                DEY
FB2E: 60        RTS2D   RTS
FB2F: A9 00     INIT    LDA   #$00       CLR STATUS FOR DEBUG
FB31: 85 48             STA   STATUS       SOFTWARE
FB33: AD 56 C0          LDA   LORES
FB36: AD 54 C0          LDA   LOWSCR     INIT VIDEO MODE
FB39: AD 51 C0  SETTXT  LDA   TXTSET     SET FOR TEXT MODE
FB3C: A9 00             LDA   #$00         FULL SCREEN WINDOW
FB3E: F0 0B             BEQ   SETWND
FB40: AD 50 C0  SETGR   LDA   TXTCLR     SET FOR GRAPHICS MODE
FB43: AD 53 C0          LDA   MIXSET       LOWER 4 LINES AS
FB46: 20 36 F8          JSR   CLRTOP       TEXT WINDOW
FB49: A9 14             LDA   #$14
FB4B: 85 22     SETWND  STA   WNDTOP     SET FOR 40 COL WINDOW
FB4D: A9 00             LDA   #$00         TOP IN A-REG,
FB4F: 85 20             STA   WNDLFT       BTTM AT LINE 24
FB51: A9 28             LDA   #$28
FB53: 85 21             STA   WNDWDTH
FB55: A9 18             LDA   #$18
FB57: 85 23             STA   WNDBTM       VTAB TO ROW 23
FB59: A9 17             LDA   #$17
FB5B: 85 25     TABV    STA   CV         VTABS TO ROW IN A-REG
FB5D: 4C 22 FC          JMP   VTAB
FB60: 20 A4 FB  MULPM   JSR   MD1        ABS VAL OF AC AUX
FB63: A0 10     MUL     LDY   #$10       INDEX FOR 16 BITS
FB65: A5 50     MUL2    LDA   ACL        ACX * AUX + XTND
FB67: 4A                LSR   A           TO AC, XTND
FB68: 90 0C             BCC   MUL4       IF NO CARRY,
FB6A: 18                CLC               NO PARTIAL PROD.
FB6B: A2 FE             LDX   #$FE
FB6D: B5 54     MUL3    LDA   XTNDL+2,X  ADD MPLCND (AUX)
FB6F: 75 56             ADC   AUXL+2,X    TO PARTIAL PROD
FB71: 95 54             STA   XTNDL+2,X   (XTND)
FB73: E8                INX
FB74: D0 F7             BNE   MUL3
FB76: A2 03     MUL4    LDX   #$03
FB78: 76        MUL5    DFB   $76
FB79: 50                DFB   $50
FB7A: CA                DEX
FB7B: 10 FB             BPL   MUL5
FB7D: 88                DEY
FB7E: D0 E5             BNE   MUL2
FB80: 60                RTS
```

```
FB81: 20 A4 FB  DIVPM    JSR   MD1        ABS VAL OF AC, AUX.
FB84: A0 10     DIV      LDY   #$10       INDEX FOR 16 BITS
FB86: 06 50     DIV2     ASL   ACL
FB88: 26 51              ROL   ACH
FB8A: 26 52              ROL   XTNDL      XTND/AUX
FB8C: 26 53              ROL   XTNDH        TO AC.
FB8E: 38                 SEC
FB8F: A5 52              LDA   XTNDL
FB91: E5 54              SBC   AUXL       MOD TO XTND.
FB93: AA                 TAX
FB94: A5 53              LDA   XTNDH
FB96: E5 55              SBC   AUXH
FB98: 90 06              BCC   DIV3
FB9A: 86 52              STX   XTNDL
FB9C: 85 53              STA   XTNDH
FB9E: E6 50              INC   ACL
FBA0: 88        DIV3     DEY
FBA1: D0 E3              BNE   DIV2
FBA3: 60                 RTS
FBA4: A0 00     MD1      LDY   #$00       ABS VAL OF AC, AUX
FBA6: 84 2F              STY   SIGN         WITH RESULT SIGN
FBA8: A2 54              LDX   #AUXL        IN LSB OF SIGN.
FBAA: 20 AF FB           JSR   MD3
FBAD: A2 50              LDX   #ACL
FBAF: B5 01     MD3      LDA   LOC1,X     X SPECIFIES AC OR AUX
FBB1: 10 0D              BPL   MDRTS
FBB3: 38                 SEC
FBB4: 98                 TYA
FBB5: F5 00              SBC   LOC0,X     COMPL SPECIFIED REG
FBB7: 95 00              STA   LOC0,X       IF NEG.
FBB9: 98                 TYA
FBBA: F5 01              SBC   LOC1,X
FBBC: 95 01              STA   LOC1,X
FBBE: E6 2F              INC   SIGN
FBC0: 60        MDRTS    RTS
FBC1: 48        BASCALC  PHA              CALC BASE ADR IN BASL,H
FBC2: 4A                 LSR   A            FOR GIVEN LINE NO
FBC3: 29 03              AND   #$03         0&lt=LINE NO.&lt=$17
FBC5: 09 04              ORA   #$04       ARG=000ABCDE, GENERATE
FBC7: 85 29              STA   BASH         BASH=000001CD
FBC9: 68                 PLA              AND
FBCA: 29 18              AND   #$18         BASL=EABAB000
FBCC: 90 02              BCC   BSCLC2
FBCE: 69 7F              ADC   #$7F
FBD0: 85 28     BSCLC2   STA   BASL
FBD2: 0A                 ASL
FBD3: 0A                 ASL
FBD4: 05 28              ORA   BASL
FBD6: 85 28              STA   BASL
FBD8: 60                 RTS
FBD9: C9 87     BELL1    CMP   #$87       BELL CHAR? (CNTRL-G)
FBDB: D0 12              BNE   RTS2B        NO, RETURN
FBDD: A9 40              LDA   #$40       DELAY .01 SECONDS
FBDF: 20 A8 FC           JSR   WAIT
FBE2: A0 C0              LDY   #$C0
FBE4: A9 0C     BELL2    LDA   #$0C       TOGGLE SPEAKER AT
FBE6: 20 A8 FC           JSR   WAIT         1 KHZ FOR .1 SEC.
FBE9: AD 30 C0           LDA   SPKR
FBEC: 88                 DEY
FBED: D0 F5              BNE   BELL2
FBEF: 60        RTS2B    RTS
FBF0: A4 24     STOADV   LDY   CH         CURSOR H INDEX TO Y-REG
FBF2: 91 28              STA   (BASL),Y   STORE CHAR IN LINE
FBF4: E6 24     ADVANCE  INC   CH         INCREMENT CURSOR H INDEX
FBF6: A5 24              LDA   CH           (MOVE RIGHT)
FBF8: C5 21              CMP   WNDWDTH    BEYOND WINDOW WIDTH?
FBFA: B0 66              BCS   CR           YES CR TO NEXT LINE
FBFC: 60        RTS3     RTS                NO,RETURN
FBFD: C9 A0     VIDOUT   CMP   #$A0       CONTROL CHAR?
FBFF: B0 EF              BCS   STOADV       NO,OUTPUT IT.
FC01: A8                 TAY              INVERSE VIDEO?
FC02: 10 EC              BPL   STOADV       YES, OUTPUT IT.
FC04: C9 8D              CMP   #$8D       CR?
FC06: F0 5A              BEQ   CR           YES.
FC08: C9 8A              CMP   #$8A       LINE FEED?
FC0A: F0 5A              BEQ   LF           IF SO, DO IT.
FC0C: C9 88              CMP   #$88       BACK SPACE? (CNTRL-H)
FC0E: D0 C9              BNE   BELL1        NO, CHECK FOR BELL.
FC10: C6 24     BS       DEC   CH         DECREMENT CURSOR H INDEX
FC12: 10 E8              BPL   RTS3       IF POS, OK. ELSE MOVE UP
FC14: A5 21              LDA   WNDWDTH    SET CH TO WNDWDTH-1
FC16: 85 24              STA   CH
FC18: C6 24              DEC   CH         (RIGHTMOST SCREEN POS)
FC1A: A5 22     UP       LDA   WNDTOP     CURSOR V INDEX
FC1C: C5 25              CMP   CV
```

```
FC1E: B0 0B              BCS   RTS4        IF TOP LINE THEN RETURN
FC20: C6 25              DEC   CV          DEC CURSOR V-INDEX
FC22: A5 25     VTAB     LDA   CV          GET CURSOR V-INDEX
FC24: 20 C1 FB  VTABZ    JSR   BASCALC     GENERATE BASE ADR
FC27: 65 20              ADC   WNDLFT      ADD WINDOW LEFT INDEX
FC29: 85 28              STA   BASL        TO BASL
FC2B: 60        RTS4     RTS
FC2C: 49 C0     ESC1     EOR   #$C0        ESC?
FC2E: F0 28              BEQ   HOME          IF SO, DO HOME AND CLEAR
FC30: 69 FD              ADC   #$FD        ESC-A OR B CHECK
FC32: 90 C0              BCC   ADVANCE       A, ADVANCE
FC34: F0 DA              BEQ   BS            B, BACKSPACE
FC36: 69 FD              ADC   #$FD        ESC-C OR D CHECK
FC38: 90 2C              BCC   LF            C, DOWN
FC3A: F0 DE              BEQ   UP            D, GO UP
FC3C: 69 FD              ADC   #$FD        ESC-E OR F CHECK
FC3E: 90 5C              BCC   CLREOL        E, CLEAR TO END OF LINE
FC40: D0 E9              BNE   RTS4          NOT F, RETURN
FC42: A4 24     CLREOP   LDY   CH          CURSOR H TO Y INDEX
FC44: A5 25              LDA   CV          CURSOR V TO A-REGISTER
FC46: 48        CLEOP1   PHA               SAVE CURRENT LINE ON STK
FC47: 20 24 FC           JSR   VTABZ       CALC BASE ADDRESS
FC4A: 20 9E FC           JSR   CLEOLZ      CLEAR TO EOL, SET CARRY
FC4D: A0 00              LDY   #$00        CLEAR FROM H INDEX=0 FOR REST
FC4F: 68                 PLA               INCREMENT CURRENT LINE
FC50: 69 00              ADC   #$00        (CARRY IS SET)
FC52: C5 23              CMP   WNDBTM      DONE TO BOTTOM OF WINDOW?
FC54: 90 F0              BCC   CLEOP1        NO, KEEP CLEARING LINES
FC56: B0 CA              BCS   VTAB          YES, TAB TO CURRENT LINE
FC58: A5 22     HOME     LDA   WNDTOP      INIT CURSOR V
FC5A: 85 25              STA   CV            AND H-INDICES
FC5C: A0 00              LDY   #$00
FC5E: 84 24              STY   CH          THEN CLEAR TO END OF PAGE
FC60: F0 E4              BEQ   CLEOP1
FC62: A9 00     CR       LDA   #$00        CURSOR TO LEFT OF INDEX
FC64: 85 24              STA   CH          (RET CURSOR H=0)
FC66: E6 25     LF       INC   CV          INCR CURSOR V(DOWN 1 LINE)
FC68: A5 25              LDA   CV
FC6A: C5 23              CMP   WNDBTM      OFF SCREEN?
FC6C: 90 B6              BCC   VTABZ         NO, SET BASE ADDR
FC6E: C6 25              DEC   CV          DECR CURSOR V (BACK TO BOTTOM)
FC70: A5 22     SCROLL   LDA   WNDTOP      START AT TOP OF SCRL WNDW
FC72: 48                 PHA
FC73: 20 24 FC           JSR   VTABZ       GENERATE BASE ADR
FC76: A5 28     SCRL1    LDA   BASL        COPY BASL,H
FC78: 85 2A              STA   BAS2L         TO BAS2L,H
FC7A: A5 29              LDA   BASH
FC7C: 85 2B              STA   BAS2H
FC7E: A4 21              LDY   WNDWDTH     INIT Y TO RIGHTMOST INDEX
FC80: 88                 DEY                 OF SCROLLING WINDOW
FC81: 68                 PLA
FC82: 69 01              ADC   #$01        INCR LINE NUMBER
FC84: C5 23              CMP   WNDBTM      DONE?
FC86: B0 0D              BCS   SCRL3         YES, FINISH
FC88: 48                 PHA
FC89: 20 24 FC           JSR   VTABZ       FORM BASL,H (BASE ADDR)
FC8C: B1 28     SCRL2    LDA   (BASL),Y    MOVE A CHR UP ON LINE
FC8E: 91 2A              STA   (BAS2L),Y
FC90: 88                 DEY               NEXT CHAR OF LINE
FC91: 10 F9              BPL   SCRL2
FC93: 30 E1              BMI   SCRL1       NEXT LINE (ALWAYS TAKEN)
FC95: A0 00     SCRL3    LDY   #$00        CLEAR BOTTOM LINE
FC97: 20 9E FC           JSR   CLEOLZ      GET BASE ADDR FOR BOTTOM LINE
FC9A: B0 86              BCS   VTAB        CARRY IS SET
FC9C: A4 24     CLREOL   LDY   CH          CURSOR H INDEX
FC9E: A9 A0     CLEOLZ   LDA   #$A0
FCA0: 91 28     CLEOL2   STA   (BASL),Y    STORE BLANKS FROM 'HERE'
FCA2: C8                 INY                 TO END OF LINES (WNDWDTH)
FCA3: C4 21              CPY   WNDWDTH
FCA5: 90 F9              BCC   CLEOL2
FCA7: 60                 RTS
FCA8: 38        WAIT     SEC
FCA9: 48        WAIT2    PHA
FCAA: E9 01     WAIT3    SBC   #$01
FCAC: D0 FC              BNE   WAIT3       1.0204 USEC
FCAE: 68                 PLA               (13+27/2*A+5/2*A*A)
FCAF: E9 01              SBC   #$01
FCB1: D0 F6              BNE   WAIT2
FCB3: 60                 RTS
FCB4: E6 42     NXTA4    INC   A4L         INCR 2-BYTE A4
FCB6: D0 02              BNE   NXTA1         AND A1
FCB8: E6 43              INC   A4H
FCBA: A5 3C     NXTA1    LDA   A1L         INCR 2-BYTE A1.
FCBC: C5 3E              CMP   A2L
FCBE: A5 3D              LDA   A1H           AND COMPARE TO A2
```

```
FCC0: E5 3F             SBC   A2H
FCC2: E6 3C             INC   A1L          (CARRY SET IF &gt=)
FCC4: D0 02             BNE   RTS4B
FCC6: E6 3D             INC   A1H
FCC8: 60        RTS4B   RTS
FCC9: A0 4B     HEADR   LDY   #$4B       WRITE A*256 'LONG 1'
FCCB: 20 DB FC          JSR   ZERDLY       HALF CYCLES
FCCE: D0 F9             BNE   HEADR        (650 USEC EACH)
FCD0: 69 FE             ADC   #$FE
FCD2: B0 F5             BCS   HEADR      THEN A 'SHORT 0'
FCD4: A0 21             LDY   #$21         (400 USEC)
FCD6: 20 DB FC  WRBIT   JSR   ZERDLY     WRITE TWO HALF CYCLES
FCD9: C8                INY                OF 250 USEC ('0')
FCDA: C8                INY                OR 500 USEC ('0')
FCDB: 88        ZERDLY  DEY
FCDC: D0 FD             BNE   ZERDLY
FCDE: 90 05             BCC   WRTAPE     Y IS COUNT FOR
FCE0: A0 32             LDY   #$32         TIMING LOOP
FCE2: 88        ONEDLY  DEY
FCE3: D0 FD             BNE   ONEDLY
FCE5: AC 20 C0  WRTAPE  LDY   TAPEOUT
FCE8: A0 2C             LDY   #$2C

FCEA: CA                DEX
FCEB: 60                RTS
FCEC: A2 08     RDBYTE  LDX   #$08       8 BITS TO READ
FCEE: 48        RDBYT2  PHA              READ TWO TRANSITIONS
FCEF: 20 FA FC          JSR   RD2BIT       (FIND EDGE)
FCF2: 68                PLA
FCF3: 2A                ROL              NEXT BIT
FCF4: A0 3A             LDY   #$3A       COUNT FOR SAMPLES
FCF6: CA                DEX
FCF7: D0 F5             BNE   RDBYT2
FCF9: 60                RTS
FCFA: 20 FD FC  RD2BIT  JSR   RDBIT
FCFD: 88        RDBIT   DEY              DECR Y UNTIL
FCFE: AD 60 C0          LDA   TAPEIN      TAPE TRANSITION
FD01: 45 2F             EOR   LASTIN
FD03: 10 F8             BPL   RDBIT
FD05: 45 2F             EOR   LASTIN
FD07: 85 2F             STA   LASTIN
FD09: C0 80             CPY   #$80       SET CARRY ON Y
FD0B: 60                RTS
FD0C: A4 24     RDKEY   LDY   CH
FD0E: B1 28             LDA   (BASL),Y   SET SCREEN TO FLASH
FD10: 48                PHA
FD11: 29 3F             AND   #$3F
FD13: 09 40             ORA   #$40
FD15: 91 28             STA   (BASL),Y
FD17: 68                PLA
FD18: 6C 38 00          JMP   (KSWL)     GO TO USER KEY-IN
FD1B: E6 4E     KEYIN   INC   RNDL
FD1D: D0 02             BNE   KEYIN2     INCR RND NUMBER
FD1F: E6 4F             INC   RNDH
FD21: 2C 00 C0  KEYIN2  BIT   KBD        KEY DOWN?
FD24: 10 F5             BPL   KEYIN        LOOP
FD26: 91 28             STA   (BASL),Y   REPLACE FLASHING SCREEN
FD28: AD 00 C0          LDA   KBD        GET KEYCODE
FD2B: 2C 10 C0          BIT   KBDSTRB    CLR KEY STROBE
FD2E: 60                RTS
FD2F: 20 0C FD  ESC     JSR   RDKEY      GET KEYCODE
FD32: 20 2C FC          JSR   ESC1         HANDLE ESC FUNC.
FD35: 20 0C FD  RDCHAR  JSR   RDKEY      READ KEY
FD38: C9 9B             CMP   #$9B       ESC?
FD3A: F0 F3             BEQ   ESC          YES, DON'T RETURN
FD3C: 60                RTS
FD3D: A5 32     NOTCR   LDA   INVFLG
FD3F: 48                PHA
FD40: A9 FF             LDA   #$FF
FD42: 85 32             STA   INVFLG     ECHO USER LINE
FD44: BD 00 02          LDA   IN,X         NON INVERSE
FD47: 20 ED FD          JSR   COUT
FD4A: 68                PLA
FD4B: 85 32             STA   INVFLG
FD4D: BD 00 02          LDA   IN,X
FD50: C9 88             CMP   #$88       CHECK FOR EDIT KEYS
FD52: F0 1D             BEQ   BCKSPC       BS, CTRL-X
FD54: C9 98             CMP   #$98
FD56: F0 0A             BEQ   CANCEL
FD58: E0 F8             CPX   #$F8       MARGIN?
FD5A: 90 03             BCC   NOTCR1
FD5C: 20 3A FF          JSR   BELL         YES, SOUND BELL
FD5F: E8        NOTCR1  INX              ADVANCE INPUT INDEX
FD60: D0 13             BNE   NXTCHAR
FD62: A9 DC     CANCEL  LDA   #$DC       BACKSLASH AFTER CANCELLED LINE
FD64: 20 ED FD          JSR   COUT
```

```
FD67: 20 8E FD  GETLNZ   JSR   CROUT      OUTPUT CR
FD6A: A5 33     GETLN    LDA   PROMPT
FD6C: 20 ED FD           JSR   COUT       OUTPUT PROMPT CHAR
FD6F: A2 01              LDX   #$01       INIT INPUT INDEX
FD71: 8A        BCKSPC   TXA                WILL BACKSPACE TO 0
FD72: F0 F3              BEQ   GETLNZ
FD74: CA                 DEX
FD75: 20 35 FD  NXTCHAR  JSR   RDCHAR
FD78: C9 95              CMP   #PICK      USE SCREEN CHAR
FD7A: D0 02              BNE   CAPTST       FOR CTRL-U
FD7C: B1 28              LDA   (BASL),Y

FD7E: C9 E0     CAPTST   CMP   #$E0
FD80: 90 02              BCC   ADDINP     CONVERT TO CAPS
FD82: 29 DF              AND   #$DF
FD84: 9D 00 02  ADDINP   STA   IN,X       ADD TO INPUT BUF
FD87: C9 8D              CMP   #$8D
FD89: D0 B2              BNE   NOTCR
FD8B: 20 9C FC           JSR   CLREOL     CLR TO EOL IF CR
FD8E: A9 8D     CROUT    LDA   #$8D
FD90: D0 5B              BNE   COUT
FD92: A4 3D     PRA1     LDY   A1H        PRINT CR,A1 IN HEX
FD94: A6 3C              LDX   A1L
FD96: 20 8E FD  PRYX2    JSR   CROUT
FD99: 20 40 F9           JSR   PRNTYX
FD9C: A0 00              LDY   #$00
FD9E: A9 AD              LDA   #$AD       PRINT '-'
FDA0: 4C ED FD           JMP   COUT
FDA3: A5 3C     XAM8     LDA   A1L
FDA5: 09 07              ORA   #$07       SET TO FINISH AT
FDA7: 85 3E              STA   A2L          MOD 8=7
FDA9: A5 3D              LDA   A1H
FDAB: 85 3F              STA   A2H
FDAD: A5 3C     MODSCHK  LDA   A1L
FDAF: 29 07              AND   #$07
FDB1: D0 03              BNE   DATAOUT
FDB3: 20 92 FD  XAM      JSR   PRA1
FDB6: A9 A0     DATAOUT  LDA   #$A0
FDB8: 20 ED FD           JSR   COUT       OUTPUT BLANK
FDBB: B1 3C              LDA   (A1L),Y
FDBD: 20 DA FD           JSR   PRBYTE     OUTPUT BYTE IN HEX
FDC0: 20 BA FC           JSR   NXTA1
FDC3: 90 E8              BCC   MODSCHK    CHECK IF TIME TO,
FDC5: 60        RTS4C    RTS                PRINT ADDR
FDC6: 4A        XAMPM    LSR   A          DETERMINE IF MON
FDC7: 90 EA              BCC   XAM          MODE IS XAM
FDC9: 4A                 LSR   A            ADD, OR SUB
FDCA: 4A                 LSR   A
FDCB: A5 3E              LDA   A2L
FDCD: 90 02              BCC   ADD
FDCF: 49 FF              EOR   #$FF       SUB: FORM 2'S COMPLEMENT
FDD1: 65 3C     ADD      ADC   A1L
FDD3: 48                 PHA
FDD4: A9 BD              LDA   #$BD
FDD6: 20 ED FD           JSR   COUT       PRINT '=', THEN RESULT
FDD9: 68                 PLA
FDDA: 48        PRBYTE   PHA              PRINT BYTE AS 2 HEX
FDDB: 4A                 LSR   A            DIGITS, DESTROYS A-REG
FDDC: 4A                 LSR   A
FDDD: 4A                 LSR   A
FDDE: 4A                 LSR   A
FDDF: 20 E5 FD           JSR   PRHEXZ
FDE2: 68                 PLA
FDE3: 29 0F     PRHEX    AND   #$0F       PRINT HEX DIG IN A-REG
FDE5: 09 B0     PRHEXZ   ORA   #$B0         LSB'S
FDE7: C9 BA              CMP   #$BA
FDE9: 90 02              BCC   COUT
FDEB: 69 06              ADC   #$06
FDED: 6C 36 00  COUT     JMP   (CSWL)     VECTOR TO USER OUTPUT ROUTINE
FDF0: C9 A0     COUT1    CMP   #$A0
FDF2: 90 02              BCC   COUTZ      DON'T OUTPUT CTRL'S INVERSE
FDF4: 25 32              AND   INVFLG     MASK WITH INVERSE FLAG
FDF6: 84 35     COUTZ    STY   YSAV1      SAV Y-REG
FDF8: 48                 PHA              SAV A-REG
FDF9: 20 FD FB           JSR   VIDOUT     OUTPUT A-REG AS ASCII
FDFC: 68                 PLA              RESTORE A-REG
FDFD: A4 35              LDY   YSAV1        AND Y-REG
FDFF: 60                 RTS                THEN RETURN
FE00: C6 34     BL1      DEC   YSAV
FE02: F0 9F              BEQ   XAM8
FE04: CA        BLANK    DEX              BLANK TO MON
FE05: D0 16              BNE   SETMDZ     AFTER BLANK
FE07: C9 BA              CMP   #$BA       DATA STORE MODE?
FE09: D0 BB              BNE   XAMPM        NO, XAM, ADD, OR SUB
FE0B: 85 31     STOR     STA   MODE       KEEP IN STORE MODE
FE0D: A5 3E              LDA   A2L
```

```
FE0F: 91 40             STA   (A3L),Y    STORE AS LOW BYTE AS (A3)
FE11: E6 40             INC   A3L
FE13: D0 02             BNE   RTS5       INCR A3, RETURN
FE15: E6 41             INC   A3H
FE17: 60        RTS5    RTS
FE18: A4 34     SETMODE LDY   YSAV       SAVE CONVERTED ':', '+',
FE1A: B9 FF 01          LDA   IN-1,Y       '-', '.' AS MODE.
FE1D: 85 31     SETMDZ  STA   MODE
FE1F: 60                RTS
FE20: A2 01     LT      LDX   #$01
FE22: B5 3E     LT2     LDA   A2L,X      COPY A2 (2 BYTES) TO
FE24: 95 42             STA   A4L,X        A4 AND A5
FE26: 95 44             STA   A5L,X
FE28: CA                DEX
FE29: 10 F7             BPL   LT2
FE2B: 60                RTS
FE2C: B1 3C     MOVE    LDA   (A1L),Y    MOVE (A1 TO A2) TO
FE2E: 91 42             STA   (A4L),Y      (A4)
FE30: 20 B4 FC          JSR   NXTA4
FE33: 90 F7             BCC   MOVE
FE35: 60                RTS
FE36: B1 3C     VFY     LDA   (A1L),Y    VERIFY (A1 TO A2) WITH
FE38: D1 42             CMP   (A4L),Y      (A4)
FE3A: F0 1C             BEQ   VFYOK
FE3C: 20 92 FD          JSR   PRA1
FE3F: B1 3C             LDA   (A1L),Y
FE41: 20 DA FD          JSR   PRBYTE
FE44: A9 A0             LDA   #$A0
FE46: 20 ED FD          JSR   COUT
FE49: A9 A8             LDA   #$A8
FE4B: 20 ED FD          JSR   COUT
FE4E: B1 42             LDA   (A4L),Y
FE50: 20 DA FD          JSR   PRBYTE
FE53: A9 A9             LDA   #$A9
FE55: 20 ED FD          JSR   COUT
FE58: 20 B4 FC  VFYOK   JSR   NXTA4
FE5B: 90 D9             BCC   VFY
FE5D: 60                RTS
FE5E: 20 75 FE  LIST    JSR   A1PC       MOVE A1 (2 BYTES) TO
FE61: A9 14             LDA   #$14         PC IF SPEC'D AND
FE63: 48        LIST2   PHA                DISEMBLE 20 INSTRS
FE64: 20 D0 F8          JSR   INSTDSP
FE67: 20 53 F9          JSR   PCADJ      ADJUST PC EACH INSTR
FE6A: 85 3A             STA   PCL
FE6C: 84 3B             STY   PCH
FE6E: 68                PLA
FE6F: 38                SEC
FE70: E9 01             SBC   #$01       NEXT OF 20 INSTRS
FE72: D0 EF             BNE   LIST2
FE74: 60                RTS
FE75: 8A        A1PC    TXA              IF USER SPEC'D ADR
FE76: F0 07             BEQ   A1PCRTS      COPY FROM A1 TO PC
FE78: B5 3C     A1PCLP  LDA   A1L,X
FE7A: 95 3A             STA   PCL,X
FE7C: CA                DEX
FE7D: 10 F9             BPL   A1PCLP
FE7F: 60        A1PCRTS RTS
FE80: A0 3F     SETINV  LDY   #$3F       SET FOR INVERSE VID
FE82: D0 02             BNE   SETIFLG     VIA COUT1
FE84: A0 FF     SETNORM LDY   #$FF       SET FOR NORMAL VID
FE86: 84 32     SETIFLG STY   INVFLG
FE88: 60                RTS
FE89: A9 00     SETKBD  LDA   #$00       SIMULATE PORT #0 INPUT
FE8B: 85 3E     INPORT  STA   A2L          SPECIFIED (KEYIN ROUTINE)
FE8D: A2 38     INPRT   LDX   #KSWL
FE8F: A0 1B             LDY   #KEYIN
FE91: D0 08             BNE   IOPRT
FE93: A9 00     SETVID  LDA   #$00       SIMULATE PORT #0 OUTPUT
FE95: 85 3E     OUTPORT STA   A2L          SPECIFIED (COUT1 ROUTINE)
FE97: A2 36     OUTPRT  LDX   #CSWL
FE99: A0 F0             LDY   #COUT1
FE9B: A5 3E     IOPRT   LDA   A2L        SET RAM IN/OUT VECTORS
FE9D: 29 0F             AND   #$0F
FE9F: F0 06             BEQ   IOPRT1
FEA1: 09 C0             ORA   #IOADR/256
FEA3: A0 00             LDY   #$00
FEA5: F0 02             BEQ   IOPRT2
FEA7: A9 FD     IOPRT1  LDA   #COUT1/256
FEA9: 94 00     IOPRT2  STY   LOC0,X
FEAB: 95 01             STA   LOC1,X
FEAD: 60                RTS
FEAE: EA                NOP
FEAF: EA                NOP
FEB0: 4C 00 E0  XBASIC  JMP   BASIC      TO BASIC WITH SCRATCH
FEB3: 4C 03 E0  BASCONT JMP   BASIC2     CONTINUE BASIC
```

```
FEB6: 20 75 FE  GO       JSR   A1PC       ADR TO PC IF SPEC'D
FEB9: 20 3F FF           JSR   RESTORE    RESTORE META REGS
FEBC: 6C 3A 00           JMP   (PCL)      GO TO USER SUBR
FEBF: 4C D7 FA  REGZ     JMP   REGDSP     TO REG DISPLAY
FEC2: C6 34     TRACE    DEC   YSAV
FEC4: 20 75 FE  STEPZ    JSR   A1PC       ADR TO PC IF SPEC'D
FEC7: 4C 43 FA           JMP   STEP       TAKE ONE STEP
FECA: 4C F8 03  USR      JMP   USRADR     TO USR SUBR AT USRADR
FECD: A9 40     WRITE    LDA   #$40
FECF: 20 C9 FC           JSR   HEADR      WRITE 10-SEC HEADER
FED2: A0 27              LDY   #$27
FED4: A2 00     WR1      LDX   #$00
FED6: 41 3C              EOR   (A1L,X)
FED8: 48                 PHA
FED9: A1 3C              LDA   (A1L,X)
FEDB: 20 ED FE           JSR   WRBYTE
FEDE: 20 BA FC           JSR   NXTA1
FEE1: A0 1D              LDY   #$1D
FEE3: 68                 PLA
FEE4: 90 EE              BCC   WR1
FEE6: A0 22              LDY   #$22
FEE8: 20 ED FE           JSR   WRBYTE
FEEB: F0 4D              BEQ   BELL
FEED: A2 10     WRBYTE   LDX   #$10
FEEF: 0A        WRBYT2   ASL   A
FEF0: 20 D6 FC           JSR   WRBIT
FEF3: D0 FA              BNE   WRBYT2
FEF5: 60                 RTS
FEF6: 20 00 FE  CRMON    JSR   BL1        HANDLE A CR AS BLANK
FEF9: 68                 PLA                THEN POP STACK
FEFA: 68                 PLA                AND RTN TO MON
FEFB: D0 6C              BNE   MONZ
FEFD: 20 FA FC  READ     JSR   RD2BIT     FIND TAPEIN EDGE
FF00: A9 16              LDA   #$16
FF02: 20 C9 FC           JSR   HEADR      DELAY 3.5 SECONDS
FF05: 85 2E              STA   CHKSUM     INIT CHKSUM=$FF
FF07: 20 FA FC           JSR   RD2BIT     FIND TAPEIN EDGE
FF0A: A0 24     RD2      LDY   #$24       LOOK FOR SYNC BIT
FF0C: 20 FD FC           JSR   RDBIT        (SHORT 0)
FF0F: B0 F9              BCS   RD2          LOOP UNTIL FOUND
FF11: 20 FD FC           JSR   RDBIT      SKIP SECOND SYNC H-CYCLE
FF14: A0 3B              LDY   #$3B       INDEX FOR 0/1 TEST
FF16: 20 EC FC  RD3      JSR   RDBYTE     READ A BYTE
FF19: 81 3C              STA   (A1L,X)    STORE AT (A1)
FF1B: 45 2E              EOR   CHKSUM
FF1D: 85 2E              STA   CHKSUM     UPDATE RUNNING CHKSUM
FF1F: 20 BA FC           JSR   NXTA1      INC A1, COMPARE TO A2
FF22: A0 35              LDY   #$35       COMPENSATE 0/1 INDEX
FF24: 90 F0              BCC   RD3        LOOP UNTIL DONE
FF26: 20 EC FC           JSR   RDBYTE     READ CHKSUM BYTE
FF29: C5 2E              CMP   CHKSUM
FF2B: F0 0D              BEQ   BELL       GOOD, SOUND BELL AND RETURN
FF2D: A9 C5     PRERR    LDA   #$C5
FF2F: 20 ED FD           JSR   COUT       PRINT "ERR", THEN BELL
FF32: A9 D2              LDA   #$D2
FF34: 20 ED FD           JSR   COUT
FF37: 20 ED FD           JSR   COUT
FF3A: A9 87     BELL     LDA   #$87       OUTPUT BELL AND RETURN
FF3C: 4C ED FD           JMP   COUT
FF3F: A5 48     RESTORE  LDA   STATUS     RESTORE 6502 REG CONTENTS
FF41: 48                 PHA                USED BY DEBUG SOFTWARE
FF42: A5 45              LDA   ACC
FF44: A6 46     RESTR1   LDX   XREG
FF46: A4 47              LDY   YREG
FF48: 28                 PLP
FF49: 60                 RTS
FF4A: 85 45     SAVE     STA   ACC        SAVE 6502 REG CONTENTS
FF4C: 86 46     SAV1     STX   XREG
FF4E: 84 47              STY   YREG
FF50: 08                 PHP
FF51: 68                 PLA
FF52: 85 48              STA   STATUS
FF54: BA                 TSX
FF55: 86 49              STX   SPNT
FF57: D8                 CLD
FF58: 60                 RTS
FF59: 20 84 FE  RESET    JSR   SETNORM    SET SCREEN MODE
FF5C: 20 2F FB           JSR   INIT         AND INIT KBD/SCREEN
FF5F: 20 93 FE           JSR   SETVID       AS I/O DEV'S
FF62: 20 89 FE           JSR   SETKBD
FF65: D8        MON      CLD              MUST SET HEX MODE!
FF66: 20 3A FF           JSR   BELL
FF69: A9 AA     MONZ     LDA   #$AA       '*' PROMPT FOR MON
FF6B: 85 33              STA   PROMPT
FF6D: 20 67 FD           JSR   GETLNZ     READ A LINE
```

```
FF70: 20 C7 FF           JSR   ZMODE      CLEAR MON MODE, SCAN IDX
FF73: 20 A7 FF  NXTITM   JSR   GETNUM     GET ITEM, NON-HEX
FF76: 84 34              STY   YSAV         CHAR IN A-REG
FF78: A0 17              LDY   #$17         X-REG=0 IF NO HEX INPUT
FF7A: 88        CHRSRCH  DEY
FF7B: 30 E8              BMI   MON        NOT FOUND, GO TO MON
FF7D: D9 CC FF           CMP   CHRTBL,Y   FIND CMND CHAR IN TEL
FF80: D0 F8              BNE   CHRSRCH
FF82: 20 BE FF           JSR   TOSUB      FOUND, CALL CORRESPONDING
FF85: A4 34              LDY   YSAV         SUBROUTINE
FF87: 4C 73 FF           JMP   NXTITM
FF8A: A2 03     DIG      LDX   #$03
FF8C: 0A                 ASL   A
FF8D: 0A                 ASL   A          GOT HEX DIG,
FF8E: 0A                 ASL   A            SHIFT INTO A2
FF8F: 0A                 ASL   A
FF90: 0A        NXTBIT   ASL   A
FF91: 26 3E              ROL   A2L
FF93: 26 3F              ROL   A2H
FF95: CA                 DEX              LEAVE X=$FF IF DIG
FF96: 10 F8              BPL   NXTBIT
FF98: A5 31     NXTBAS   LDA   MODE
FF9A: D0 06              BNE   NXTBS2     IF MODE IS ZERO
FF9C: B5 3F              LDA   A2H,X       THEN COPY A2 TO
FF9E: 95 3D              STA   A1H,X       A1 AND A3
FFA0: 95 41              STA   A3H,X
FFA2: E8        NXTBS2   INX
FFA3: F0 F3              BEQ   NXTBAS
FFA5: D0 06              BNE   NXTCHR
FFA7: A2 00     GETNUM   LDX   #$00       CLEAR A2
FFA9: 86 3E              STX   A2L
FFAB: 86 3F              STX   A2H
FFAD: B9 00 02  NXTCHR   LDA   IN,Y       GET CHAR
FFB0: C8                 INY
FFB1: 49 B0              EOR   #$B0
FFB3: C9 0A              CMP   #$0A
FFB5: 90 D3              BCC   DIG        IF HEX DIG, THEN
FFB7: 69 88              ADC   #$88
FFB9: C9 FA              CMP   #$FA
FFBB: B0 CD              BCS   DIG
FFBD: 60                 RTS
FFBE: A9 FE     TOSUB    LDA   #GO/256    PUSH HIGH-ORDER
FFC0: 48                 PHA                SUBR ADR ON STK
FFC1: B9 E3 FF           LDA   SUBTBL,Y   PUSH LOW-ORDER
FFC4: 48                 PHA                SUBR ADR ON STK
FFC5: A5 31              LDA   MODE
FFC7: A0 00     ZMODE    LDY   #$00       CLR MODE, OLD MODE

FFC9: 84 31              STY   MODE         TO A-REG
FFCB: 60                 RTS               GO TO SUBR VIA RTS
FFCC: BC        CHRTBL   DFB   $BC        F("CTRL-C")
FFCD: B2                 DFB   $B2        F("CTRL-Y")
FFCE: BE                 DFB   $BE        F("CTRL-E")
FFCF: ED                 DFB   $ED        F("T")
FFD0: EF                 DFB   $EF        F("V")
FFD1: C4                 DFB   $C4        F("CTRL-K")
FFD2: EC                 DFB   $EC        F("S")
FFD3: A9                 DFB   $A9        F("CTRL-P")
FFD4: BB                 DFB   $BB        F("CTRL-B")
FFD5: A6                 DFB   $A6        F("-")
FFD6: A4                 DFB   $A4        F("+")
FFD7: 06                 DFB   $06        F("M") (F=EX-OR $B0+$89)
FFD8: 95                 DFB   $95        F("&lt")
FFD9: 07                 DFB   $07        F("N")
FFDA: 02                 DFB   $02        F("I")
FFDB: 05                 DFB   $05        F("L")
FFDC: F0                 DFB   $F0        F("W")
FFDD: 00                 DFB   $00        F("G")
FFDE: EB                 DFB   $EB        F("R")
FFDF: 93                 DFB   $93        F(":")
FFE0: A7                 DFB   $A7        F(".")
FFE1: C6                 DFB   $C6        F("CR")
FFE2: 99                 DFB   $99        F(BLANK)
FFE3: B2        SUBTBL   DFB   BASCONT-1
FFE4: C9                 DFB   USR-1
FFE5: BE                 DFB   REGZ-1
FFE6: C1                 DFB   TRACE-1
FFE7: 35                 DFB   VFY-1
FFE8: 8C                 DFB   INPRT-1
FFE9: C3                 DFB   STEPZ-1
FFEA: 96                 DFB   OUTPRT-1
FFEB: AF                 DFB   XBASIC-1
FFEC: 17                 DFB   SETMODE-1
FFED: 17                 DFB   SETMODE-1
FFEE: 2B                 DFB   MOVE-1
FFEF: 1F                 DFB   LT-1
```

```
FFF0: 83                DFB   SETNORM-1
FFF1: 7F                DFB   SETINV-1
FFF2: 5D                DFB   LIST-1
FFF3: CC                DFB   WRITE-1
FFF4: B5                DFB   GO-1
FFF5: FC                DFB   READ-1
FFF6: 17                DFB   SETMODE-1
FFF7: 17                DFB   SETMODE-1
FFF8: F5                DFB   CRMON-1
FFF9: 03                DFB   BLANK-1
FFFA: FB                DFB   NMI        NMI VECTOR
FFFB: 03                DFB   NMI/256
FFFC: 59                DFB   RESET      RESET VECTOR
FFFD: FF                DFB   RESET/256
FFFE: 86                DFB   IRQ        IRQ VECTOR
FFFF: FA                DFB   IRQ/256
                XQTNZ   EQU   $3C
                *********************** 
                *                     * 
                *      APPLE-II       * 
                *   MINI-ASSEMBLER    * 
                *                     * 
                *  COPYRIGHT 1977 BY  * 
                * APPLE COMPUTER INC. * 
                *                     * 
                * ALL RIGHTS RESERVED * 
                *                     * 
                *     S. WOZNIAK      * 
                *      A. BAUM        * 
                *********************** 
                 TITLE "APPLE-II MINI-ASSEMBLER"
                FORMAT   EQU   $2E
                LENGTH   EQU   $2F
                MODE     EQU   $31
                PROMPT   EQU   $33
                YSAV     EQU   $34
                L        EQU   $35
                PCL      EQU   $3A
                PCH      EQU   $3B
                A1H      EQU   $3D
                A2L      EQU   $3E
                A2H      EQU   $3F
                A4L      EQU   $42
                A4H      EQU   $43
                FMT      EQU   $44
                IN       EQU   $200
                INSDS2   EQU   $F88E
                INSTDSP  EQU   $F8D0
                PRBL2    EQU   $F94A
                PCADJ    EQU   $F953
                CHAR1    EQU   $F9B4
                CHAR2    EQU   $F9BA
                MNEML    EQU   $F9C0
                MNEMR    EQU   $FA00
                CURSUP   EQU   $FC1A
                GETLNZ   EQU   $FD67
                COUT     EQU   $FDED
                BL1      EQU   $FE00
                A1PCLP   EQU   $FE78
                BELL     EQU   $FF3A
                GETNUM   EQU   $FFA7
                TOSUB    EQU   $FFBE
                ZMODE    EQU   $FFC7
                CHRTBL   EQU   $FFCC
                         ORG   $F500
F500: E9 81     REL      SBC   #$81       IS FMT COMPATIBLE
F502: 4A                 LSR              WITH RELATIVE MODE?
F503: D0 14              BNE   ERR3       NO.
F505: A4 3F              LDY   A2H
F507: A6 3E              LDX   A2L        DOUBLE DECREMENT
F509: D0 01              BNE   REL2
F50B: 88                 DEY
F50C: CA        REL2     DEX
F50D: 8A                 TXA
F50E: 18                 CLC
F50F: E5 3A              SBC   PCL        FORM ADDR-PC-2
F511: 85 3E              STA   A2L
F513: 10 01              BPL   REL3
F515: C8                 INY
F516: 98        REL3     TYA
```

```
F517: E5 3B             SBC   PCH
F519: D0 6B     ERR3    BNE   ERR         ERROR IF >1-BYTE BRANCH
F51B: A4 2F     FINDOP  LDY   LENGTH
F51D: B9 3D 00  FNDOP2  LDA   A1H,Y       MOVE INST TO (PC)
F520: 91 3A             STA   (PCL),Y
F522: 88                DEY
F523: 10 F8             BPL   FNDOP2
F525: 20 1A FC          JSR   CURSUP
F528: 20 1A FC          JSR   CURSUP      RESTORE CURSOR
F52B: 20 D0 F8          JSR   INSTDSP     TYPE FORMATTED LINE
F52E: 20 53 F9          JSR   PCADJ       UPDATE PC
F531: 84 3B             STY   PCH
F533: 85 3A             STA   PCL
F535: 4C 95 F5          JMP   NXTLINE     GET NEXT LINE
F538: 20 BE FF  FAKEMON3 JSR  TOSUB       GO TO DELIM HANDLER
F53B: A4 34             LDY   YSAV        RESTORE Y-INDEX
F53D: 20 A7 FF  FAKEMON JSR   GETNUM      READ PARAM
F540: 84 34             STY   YSAV        SAVE Y-INDEX
F542: A0 17             LDY   #$17        INIT DELIMITER INDEX
F544: 88        FAKEMON2 DEY              CHECK NEXT DELIM
F545: 30 4B             BMI   RESETZ      ERR IF UNRECOGNIZED DELIM
F547: D9 CC FF          CMP   CHRTBL,Y    COMPARE WITH DELIM TABLE
F54A: D0 F8             BNE   FAKEMON2    NO MATCH
F54C: C0 15             CPY   #$15        MATCH, IS IT CR?
F54E: D0 E8             BNE   FAKEMON3    NO, HANDLE IT IN MONITOR
F550: A5 31             LDA   MODE
F552: A0 00             LDY   #$0
F554: C6 34             DEC   YSAV
F556: 20 00 FE          JSR   BL1         HANDLE CR OUTSIDE MONITOR
F559: 4C 95 F5          JMP   NXTLINE
F55C: A5 3D     TRYNEXT LDA   A1H         GET TRIAL OPCODE
F55E: 20 8E F8          JSR   INSDS2      GET FMT+LENGTH FOR OPCODE
F561: AA                TAX
F562: BD 00 FA          LDA   MNEMR,X     GET LOWER MNEMONIC BYTE
F565: C5 42             CMP   A4L         MATCH?
F567: D0 13             BNE   NEXTOP      NO, TRY NEXT OPCODE.
F569: BD C0 F9          LDA   MNEML,X     GET UPPER MNEMONIC BYTE
F56C: C5 43             CMP   A4H         MATCH?
F56E: D0 0C             BNE   NEXTOP      NO, TRY NEXT OPCODE
F570: A5 44             LDA   FMT
F572: A4 2E             LDY   FORMAT      GET TRIAL FORMAT
F574: C0 9D             CPY   #$9D        TRIAL FORMAT RELATIVE?
F576: F0 88             BEQ   REL         YES.
F578: C5 2E     NREL    CMP   FORMAT      SAME FORMAT?
F57A: F0 9F             BEQ   FINDOP      YES.
F57C: C6 3D     NEXTOP  DEC   A1H         NO, TRY NEXT OPCODE
F57E: D0 DC             BNE   TRYNEXT
F580: E6 44             INC   FMT         NO MORE, TRY WITH LEN=2
F582: C6 35             DEC   L           WAS L=2 ALREADY?
F584: F0 D6             BEQ   TRYNEXT     NO.
F586: A4 34     ERR     LDY   YSAV        YES, UNRECOGNIZED INST.
F588: 98        ERR2    TYA
F589: AA                TAX
F58A: 20 4A F9          JSR   PRBL2       PRINT ^ UNDER LAST READ
F58D: A9 DE             LDA   #$DE        CHAR TO INDICATE ERROR
F58F: 20 ED FD          JSR   COUT        POSITION.
F592: 20 3A FF  RESETZ  JSR   BELL
F595: A9 A1     NXTLINE LDA   #$A1        '!'
F597: 85 33             STA   PROMPT      INITIALIZE PROMPT
F599: 20 67 FD          JSR   GETLNZ      GET LINE.
F59C: 20 C7 FF          JSR   ZMODE       INIT SCREEN STUFF
F59F: AD 00 02          LDA   IN          GET CHAR
F5A2: C9 A0             CMP   #$A0        ASCII BLANK?
F5A4: F0 13             BEQ   SPACE       YES
F5A6: C8                INY
F5A7: C9 A4             CMP   #$A4        ASCII '$' IN COL 1?
F5A9: F0 92             BEQ   FAKEMON     YES, SIMULATE MONITOR
F5AB: 88                DEY               NO, BACKUP A CHAR
F5AC: 20 A7 FF          JSR   GETNUM      GET A NUMBER
F5AF: C9 93             CMP   #$93        ':' TERMINATOR?
F5B1: D0 D5     ERR4    BNE   ERR2        NO, ERR.
F5B3: 8A                TXA
F5B4: F0 D2             BEQ   ERR2        NO ADR PRECEDING COLON.
F5B6: 20 78 FE          JSR   A1PCLP      MOVE ADR TO PCL, PCH.
F5B9: A9 03     SPACE   LDA   #$3         COUNT OF CHARS IN MNEMONIC
F5BB: 85 3D             STA   A1H
F5BD: 20 34 F6  NXTMN   JSR   GETNSP      GET FIRST MNEM CHAR.
F5C0: 0A        NXTM    ASL   A
F5C1: E9 BE             SBC   #$BE        SUBTRACT OFFSET
F5C3: C9 C2             CMP   #$C2        LEGAL CHAR?
F5C5: 90 C1             BCC   ERR2        NO.
F5C7: 0A                ASL   A           COMPRESS-LEFT JUSTIFY
F5C8: 0A                ASL   A
F5C9: A2 04             LDX   #$4
F5CB: 0A        NXTM2   ASL   A           DO 5 TRIPLE WORD SHIFTS
```

```
F5CC: 26 42             ROL   A4L
F5CE: 26 43             ROL   A4H
F5D0: CA                DEX
F5D1: 10 F8             BPL   NXTM2
F5D3: C6 3D             DEC   A1H        DONE WITH 3 CHARS?
F5D5: F0 F4             BEQ   NXTM2      YES, BUT DO 1 MORE SHIFT
F5D7: 10 E4             BPL   NXTMN      NO
F5D9: A2 05    FORM1    LDX   #$5        5 CHARS IN ADDR MODE
F5DB: 20 34 F6 FORM2    JSR   GETNSP     GET FIRST CHAR OF ADDR
F5DE: 84 34             STY   YSAV
F5E0: DD B4 F9          CMP   CHAR1,X    FIRST CHAR MATCH PATTERN?
F5E3: D0 13             BNE   FORM3      NO
F5E5: 20 34 F6          JSR   GETNSP     YES, GET SECOND CHAR
F5E8: DD BA F9          CMP   CHAR2,X    MATCHES SECOND HALF?
F5EB: F0 0D             BEQ   FORM5      YES.
F5ED: BD BA F9          LDA   CHAR2,X    NO, IS SECOND HALF ZERO?
F5F0: F0 07             BEQ   FORM4      YES.
F5F2: C9 A4             CMP   #$A4       NO,SECOND HALF OPTIONAL?
F5F4: F0 03             BEQ   FORM4      YES.
F5F6: A4 34             LDY   YSAV
F5F8: 18       FORM3    CLC              CLEAR BIT-NO MATCH
F5F9: 88       FORM4    DEY              BACK UP 1 CHAR
F5FA: 26 44    FORM5    ROL   FMT        FORM FORMAT BYTE
F5FC: E0 03             CPX   #$3        TIME TO CHECK FOR ADDR.
F5FE: D0 0D             BNE   FORM7      NO
F600: 20 A7 FF          JSR   GETNUM     YES
F603: A5 3F             LDA   A2H
F605: F0 01             BEQ   FORM6      HIGH-ORDER BYTE ZERO
F607: E8                INX              NO, INCR FOR 2-BYTE
F608: 86 35    FORM6    STX   L          STORE LENGTH
F60A: A2 03             LDX   #$3        RELOAD FORMAT INDEX
F60C: 88                DEY              BACKUP A CHAR
F60D: 86 3D    FORM7    STX   A1H        SAVE INDEX
F60F: CA                DEX              DONE WITH FORMAT CHECK?
F610: 10 C9             BPL   FORM2      NO.
F612: A5 44             LDA   FMT        YES, PUT LENGTH
F614: 0A                ASL   A           IN LOW BITS
F615: 0A                ASL   A
F616: 05 35             ORA   L
F618: C9 20             CMP   #$20
F61A: B0 06             BCS   FORM8      ADD "$" IF NONZERO LENGTH
F61C: A6 35             LDX   L          AND DON'T ALREADY HAVE IT
F61E: F0 02             BEQ   FORM8
F620: 09 80             ORA   #$80
F622: 85 44    FORM8    STA   FMT
F624: 84 34             STY   YSAV
F626: B9 00 02          LDA   IN,Y       GET NEXT NONBLANK
F629: C9 BB             CMP   #$BB       '' START OF COMMENT?
F62B: F0 04             BEQ   FORM9      YES
F62D: C9 8D             CMP   #$8D       CARRIAGE RETURN?
F62F: D0 80             BNE   ERR4       NO, ERR.
F631: 4C 5C F5 FORM9    JMP   TRYNEXT
F634: B9 00 02 GETNSP   LDA   IN,Y
F637: C8                INY
F638: C9 A0             CMP   #$A0       GET NEXT NON BLANK CHAR
F63A: F0 F8             BEQ   GETNSP
F63C: 60                RTS
                        ORG   $F666
F666: 4C 92 F5 MINIASM  JMP   RESETZ
```