

Apple II

Reference Manual Addendum:
Monitor ROM Listings
For IIe Only



Notice

Apple Computer, Inc. reserves the right to make improvements in the product described in this manual at any time and without notice.

Disclaimer of All Warranties and Liabilities

Apple Computer, Inc. makes no warranties, either express or implied, with respect to this manual or with respect to the software described in this manual, its quality, performance, merchantability, or fitness for any particular purpose. Apple Computer, Inc. software is sold or licensed "as is." The entire risk as to its quality and performance is with the buyer. Should the programs prove defective following their purchase, the buyer (and not Apple Computer, Inc., its distributor, or its retailer) assumes the entire cost of all necessary servicing, repair, or correction and any incidental or consequential damages. In no event will Apple Computer, Inc. be liable for direct, indirect, incidental, or consequential damages resulting from any defect in the software, even if Apple Computer, Inc. has been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you.

This manual is copyrighted. All rights are reserved. This document may not, in whole or part, be copied, photocopied, reproduced, translated or reduced to any electronic medium or machine readable form without prior consent, in writing, from Apple Computer, Inc.

© 1982 by Apple Computer, Inc.
20525 Mariani Avenue
Cupertino, California 95014
(408) 996-1010

The word Apple and the Apple logo are registered trademarks of Apple Computer, Inc.

Simultaneously published in the U.S.A and Canada.

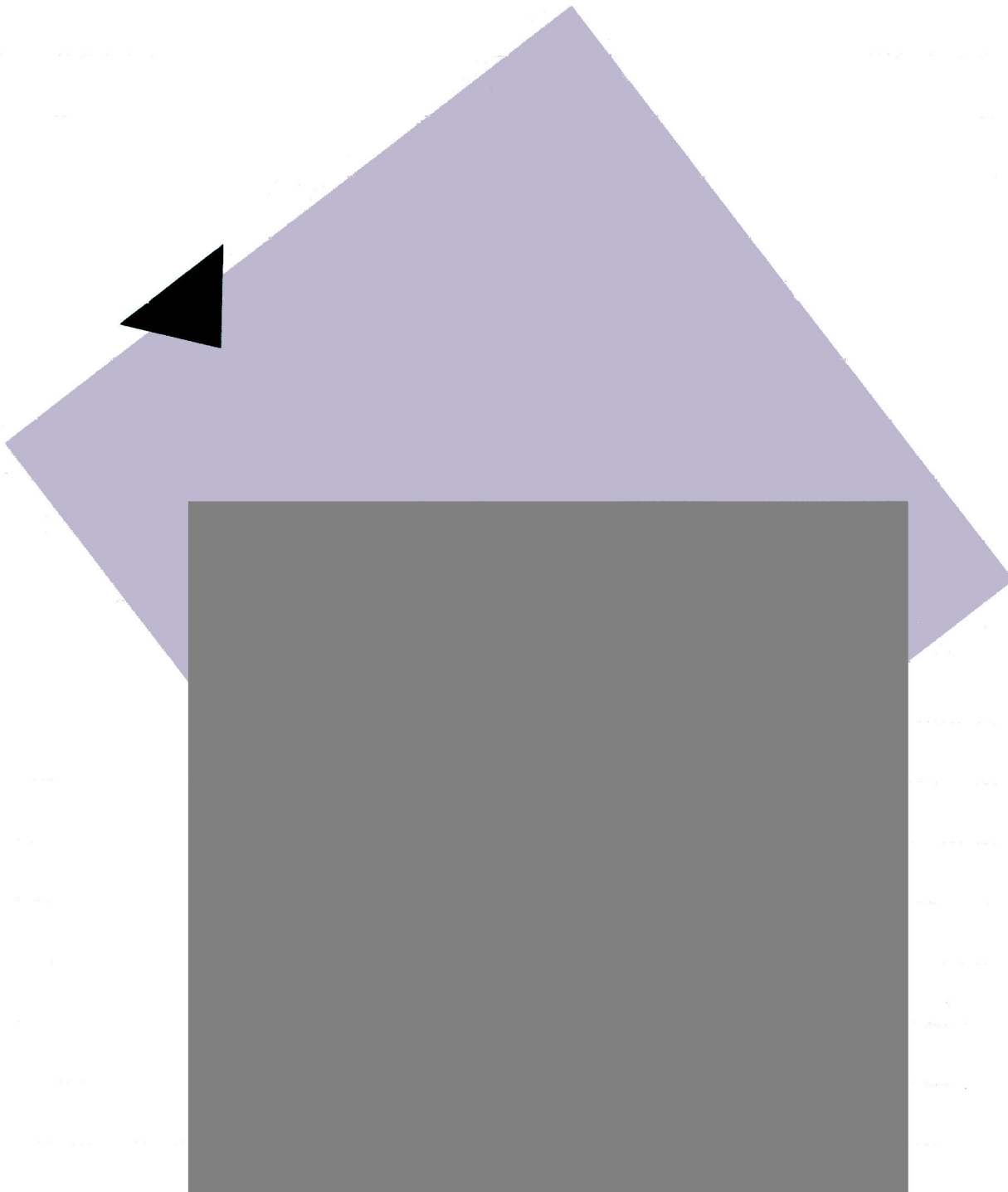


Warning

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

Apple IIe

Reference Manual Addendum: Monitor ROM Listings



Radio and Television Interference

The equipment described in this manual generates and uses radio-frequency energy. If it is not installed and used properly, that is, in strict accordance with our instructions, it may cause interference with radio and television reception.

This equipment has been tested and complies with the limits for a Class B computing device in accordance with the specifications in Subpart J, Part 15, of FCC rules. These rules are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that the interference will not occur in a particular installation, especially if you use a “rabbit ear” television antenna. (A “rabbit ear” antenna is the telescoping-rod type usually contained on TV receivers.)

You can determine whether your computer is causing interference by turning it off. If the interference stops, it was probably caused by the computer or its peripheral devices. To further isolate the problem:

- Disconnect the peripheral devices and their input/output cables one at a time. If the interference stops, it is caused by either the peripheral device or its I/O cable. These devices usually require shielded I/O cables. For Apple peripheral devices, you can obtain the proper shielded cable from your dealer. For non-Apple peripheral devices, contact the manufacturer or dealer for assistance.

If your computer does cause interference to radio or television reception, you can try to correct the interference by using one or more of the following measures:

- Turn the TV or radio antenna until the interference stops.
- Move the computer to one side or the other of the TV or radio.
- Move the computer farther away from the TV or radio.
- Plug the computer into an outlet that is on a different circuit than the TV or radio. (That is, make certain the computer and the radio or television set are on circuits controlled by different circuit breakers or fuses.)
- Consider installing a rooftop television antenna with coaxial cable lead-in between the antenna and TV.

Table of Contents

3

Monitor ROM Listings

- 3 Monitor Firmware Listing
- 19 Monitor Symbol Table, Sorted by Symbol
- 21 Monitor Symbol Table, Sorted by Address
- 23 80-Column Firmware Listing
- 51 80-Column Symbol Table, Sorted by Symbol
- 53 80-Column Symbol Table, Sorted by Address

Monitor Firmware Listing

```

0000:          2 *****
0000:          3 *
0000:          4 * APPLE II
0000:          5 * MONITOR II
0000:          6 *
0000:          7 * COPYRIGHT 1978 BY
0000:          8 * APPLE COMPUTER, INC.
0000:          9 *
0000:         10 * ALL RIGHTS RESERVED
0000:         11 *
0000:         12 * STEVE WOZNIAK
0000:         13 *
0000:         14 *****
0000:         15 *
0000:         16 * MODIFIED NOV 1978
0000:         17 * BY JOHN A
0000:         18 *
0000:         19 * MODIFIED SEP 1981
0000:         20 * BY RICK AURICCHIO
0000:         21 *   & BRYAN STEARNS
0000:         22 *   FOR APPLE2E BOCOLS
0000:         23 *
0000:         24 * CHANGES MARKED BY 'RRA09B1'
0000:         25 *
0000:         0001 26 APPLE2E      EQU    1          ; COND ASSM/RRA09B1
0000:         27 *
0000:         28 *****
----- NEXT OBJECT FILE NAME IS BJS SRC1 OBJO
FB00:         FB00 29          ORG    $FB00
FB00:         0000 30          OBJ    $2000
FB00:         31 *****
FB00:         0000 32 LOCO      EQU    $00
FB00:         0001 33 LOC1      EQU    $01
FB00:         0020 34 WNDLFT    EQU    $20
FB00:         0021 35 WNDWTH    EQU    $21
FB00:         0022 36 WNDTOP    EQU    $22
FB00:         0023 37 WNDBTM    EQU    $23
FB00:         0024 38 CH        EQU    $24
FB00:         0025 39 CV        EQU    $25
FB00:         0026 40 GBASL     EQU    $26
FB00:         0027 41 GBASH     EQU    $27
FB00:         0028 42 BASL      EQU    $28
FB00:         0029 43 BASH      EQU    $29
FB00:         002A 44 BAS2L     EQU    $2A
FB00:         002B 45 BAS2H     EQU    $2B
FB00:         002C 46 H2        EQU    $2C
FB00:         002C 47 LHMEM     EQU    $2C
FB00:         002D 48 V2        EQU    $2D
FB00:         002D 49 RMNEM     EQU    $2D
FB00:         002E 50 MASK      EQU    $2E
FB00:         002E 51 CHKSUM    EQU    $2E
FB00:         002E 52 FORMAT    EQU    $2E
FB00:         002F 53 LASTIN    EQU    $2F
FB00:         002F 54 LENGTH    EQU    $2F
FB00:         002F 55 SIGN      EQU    $2F
FB00:         0030 56 COLDR     EQU    $30
FB00:         0031 57 MODE      EQU    $31
FB00:         0032 58 INVFLG    EQU    $32
FB00:         0033 59 PROMPT    EQU    $33
FB00:         0034 60 YSAV      EQU    $34
FB00:         0035 61 YSAV1     EQU    $35
FB00:         0036 62 CSWL      EQU    $36
FB00:         0037 63 CSWH      EQU    $37
FB00:         0038 64 KSWL      EQU    $38
FB00:         0039 65 KSWH      EQU    $39
FB00:         003A 66 PCL       EQU    $3A
FB00:         003B 67 PCH       EQU    $3B
FB00:         003C 68 A1L       EQU    $3C
FB00:         003D 69 A1H       EQU    $3D
FB00:         003E 70 A2L       EQU    $3E
FB00:         003F 71 A2H       EQU    $3F
FB00:         0040 72 A3L       EQU    $40
FB00:         0041 73 A3H       EQU    $41
FB00:         0042 74 A4L       EQU    $42
FB00:         0043 75 A4H       EQU    $43
FB00:         0044 76 A5L       EQU    $44

```

```

F800: 0045 77 A5H EGU $45
F800: 0045 78 ACC EGU $45 ; NOTE OVERLAP WITH A5H!
F800: 0046 79 XREG EGU $46
F800: 0047 80 YREG EGU $47
F800: 0048 81 STATUS EGU $48
F800: 0049 82 SPNT EGU $49
F800: 004E 83 RNDL EGU $4E
F800: 004F 84 RNDH EGU $4F
F800: 0095 85 PICK EGU $95
F800: 0200 86 IN EGU $0200
F800: 03F0 87 BRKV EGU $3F0 ; NEW VECTOR FOR BRK
F800: 03F2 88 SOFTEV EGU $3F2 ; VECTOR FOR WARM START
F800: 03F4 89 PWREDUP EGU $3F4 ; THIS MUST = EDR ##A5 OF SOFTEV+1
F800: 03F5 90 AMPERV EGU $3F5 ; APPLESOFT & EXIT VECTOR
F800: 03FB 91 USRADR EGU $03FB
F800: 03FB 92 NMI EGU $03FB
F800: 03FE 93 IRGLDC EGU $3FE
F800: 0400 94 LINE1 EGU $400
F800: 07FB 95 MSLDT EGU $07FB
F800: C000 96 IDADR EGU $C000
F800: C000 97 KBD EGU $C000
F800: C010 98 KBDSTRB EGU $C010
F800: C020 99 TAPEOUT EGU $C020
F800: C030 100 SPKR EGU $C030
F800: C030 101 TXTCLR EGU $C030
F800: C051 102 TXTSET EGU $C051
F800: C052 103 MIXCLR EGU $C052
F800: C053 104 MIXSET EGU $C053
F800: C054 105 LMSCLR EGU $C054
F800: C055 106 HISCR EGU $C055
F800: C056 107 LORES EGU $C056
F800: C057 108 HIRES EGU $C057
F800: C058 109 SETANO EGU $C058
F800: C059 110 CLRANO EGU $C059
F800: C05A 111 SETAN1 EGU $C05A
F800: C05B 112 CLRAN1 EGU $C05B
F800: C05C 113 SETAN2 EGU $C05C
F800: C05D 114 CLRAN2 EGU $C05D
F800: C05E 115 SETAN3 EGU $C05E
F800: C05F 116 CLRAN3 EGU $C05F
F800: C060 117 TAPEIN EGU $C060
F800: C064 118 PADDLO EGU $C064
F800: C070 119 PTRIG EGU $C070
F800: CFFF 120 CLRROM EGU $CFFF
F800: E000 121 BASIC EGU $E000
F800: E003 122 BASIC2 EGU $E003
F800: 4A 123 PLOT LSR A ; Y-COORD/2
F801: 08 124 PHP ; SAVE LSB IN CARRY
F802: 20 47 FB 125 JSR GBASCALC ; CALC BASE ADR IN GBASL,H
F803: 28 126 PLP ; RESTORE LSB FROM CARRY
F806: A9 0F 127 LDA ##0F ; MASK #0F IF EVEN
F808: 90 02 F80C 128 BCC RTMASK
F80A: 69 E0 129 ADC ##E0 ; MASK #FO IF ODD
F80C: 85 2E 130 RTMASK STA MASK
F80E: B1 26 131 PLOT1 LDA (GBASL),Y ; DATA
F810: 45 30 132 EDR COLOR ; XOR COLOR
F812: 25 2E 133 AND MASK ; AND MASK
F814: 51 26 134 EDR (GBASL),Y ; XOR DATA
F816: 91 26 135 STA (GBASL),Y ; TD DATA
F818: 60 136 RTS
F819: 20 00 FB 137 HLINE JSR PLOT ; PLOT SQUARE
F81C: C4 2C 138 HLINE1 H2 ; DONE?
F81E: 80 11 F831 139 BCS RTS1 ; YES, RETURN
F820: C8 140 INY ; NO, INCR INDEX (X-COORD)
F821: 20 0E FB 141 JSR PLOT1 ; PLOT NEXT SQUARE
F824: 90 F6 F81C 142 BCC HLINE1 ; ALWAYS TAKEN
F826: 69 01 143 VLINEZ ADC ##01 ; NEXT Y-COORD
F828: 48 144 VLINE PHA ; SAVE ON STACK
F829: 20 00 FB 145 JSR PLOT ; PLOT SQUARE
F82C: 6B 146 PLA
F82D: C5 2D 147 CMP V2 ; DONE?
F82F: 90 F5 F826 148 BCC VLINEZ ; NO, LOOP.
F831: 60 149 RTS1 RTS
F832: A0 2F 150 CLRSCR LDY ##2F ; MAX Y, FULL SCRN CLR
F834: D0 02 F83B 151 BNE CLRSC2 ; ALWAYS TAKEN
F836: A0 27 152 CLRTOP LDY ##27 ; MAX Y, TOP SCRN CLR
F838: 84 2D 153 CLRSC2 STY V2 ; STORE AS BOTTOM COORD
F83A: 154 FOR VLINE CALLS
F83A: A0 27 155 LDY ##27 ; RIGHTMOST X-COORD (COLUMN)
F83C: A9 00 156 CLRSC3 LDA ##00 ; TOP COORD FOR VLINE CALLS
F83E: 85 30 157 STA COLOR ; CLEAR COLOR (BLACK)
F840: 20 2B FB 158 JSR VLINE ; DRAW VLINE
F843: 8B 159 DEY ; NEXT LEFTMOST X-COORD
F844: 10 F6 F83C 160 BPL CLRSC3 ; LOOP UNTIL DONE.
F846: 60 161 RTS
F847: 4B 162 GBASCALC PHA ; FOR INPUT ODEFGH
F848: 4A 163 LSR A
F849: 29 03 164 AND ##03
F84B: 09 04 165 ORA ##04 ; GENERATE GBASH=000001FG
F84D: 85 27 166 STA GBASH
F84F: 6B 167 PLA ; AND GBASL=HDEDE000
F850: 29 1B 168 AND ##1B

```

```

F852: 90 02 F856 169 BCC GBALC
F854: 69 7F 170 ADC ##7F
F856: 85 26 171 GBALC STA GBASL
F858: 0A 172 ASL A
F859: 0A 173 ASL A
F85A: 05 26 174 ORA GBASL
F85C: 85 26 175 STA GBASL
F85E: 60 176 RTS
F85F: A5 30 177 NXTCOL LDA COLOR ; INCREMENT COLOR BY 3
F861: 18 178 CLC
F862: 69 03 179 ADC ##03
F864: 29 0F 180 SETCOL AND ##0F ; SETS COLOR=17*A MOD 16
F866: 85 30 181 STA COLOR
F868: 0A 182 ASL A ; BOTH HALF BYTES OF COLOR EQUAL
F869: 0A 183 ASL A
F86A: 0A 184 ASL A
F86B: 0A 185 ASL A
F86C: 05 30 186 ORA COLOR
F86E: 85 30 187 STA COLOR
F870: 60 188 RTS
F871: 4A 189 SCRN LSR A ; READ SCREEN Y-COORD/2
F872: 08 190 PHP ; SAVE LSB (CARRY)
F873: 20 47 F8 191 JSR GBASCALC ; CALC BASE ADDRESS
F876: B1 26 192 LDA (GBASL),Y ; GET BYTE
F878: 2B 193 PLP ; RESTORE LSB FROM CARRY
F879: 90 04 F87F 194 SCR2 BCC RTMSKZ ; IF EVEN, USE LD H
F87B: 4A 195 LSR A
F87C: 4A 196 LSR A
F87D: 4A 197 LSR A ; SHIFT HIGH HALF BYTE DOWN
F87E: 4A 198 LSR A
F87F: 29 0F 199 RTMSKZ AND ##0F ; MASK 4-BITS
F881: 60 200 RTS
F882: A6 3A 201 INSDS1 LDX PCL ; PRINT PCL, H
F884: A4 3B 202 LDY PCH
F886: 20 96 FD 203 JSR PRY2K
F889: 20 4B F9 204 JSR PRBLNK ; FOLLOWED BY A BLANK
F88C: A1 3A 205 INSDS2 LDA (PCL,X) ; GET OPCODE
F88E: AB 206 TAY
F88F: 4A 207 LSR A ; EVEN/ODD TEST
F890: 90 09 F89B 208 BCC IEVEN
F892: 6A 209 RDR A ; BIT 1 TEST
F893: B0 10 F8A5 210 BCS ERR ; XXXXX11 INVALID OP
F895: C9 A2 211 CMP ##A2
F897: F0 0C F8A5 212 BEQ ERR ; OPCODE $B9 INVALID
F899: 29 87 213 AND ##B7
F89B: 4A 214 IEVEN LSR A ; LSB INTO CARRY FOR L/R TEST
F89C: AA 215 TAX
F89D: BD 62 F9 216 LDA FMT1,X ; GET FORMAT INDEX BYTE
F8A0: 20 79 F8 217 JSR SCR2 ; R/L H-BYTE ON CARRY
F8A3: D0 04 F8A9 218 BNE GETFMT
F8A5: A0 80 219 ERR LDY ##B0 ; SUBSTITUTE $B0 FOR INVALID OPS
F8A7: A9 00 220 LDA ##00 ; SET PRINT FORMAT INDEX TO 0
F8A9: AA 221 GETFMT TAX
F8AA: BD A6 F9 222 LDA FMT2,X ; INDEX INTO PRINT FORMAT TABLE
F8AD: 85 2E 223 STA FORMAT ; SAVE FOR ADR FIELD FORMATTING
F8AF: 29 03 224 AND ##03 ; MASK FOR 2-BIT LENGTH
F8B1: 225 ; (0=1 BYTE, 1=2 BYTE, 2=3 BYTE)
F8B1: 85 2F 226 STA LENGTH
F8B3: 9B 227 TYA ; OPCODE
F8B4: 29 8F 228 AND ##BF ; MASK FOR 1XXX1010 TEST
F8B6: AA 229 TAX ; SAVE IT
F8B7: 9B 230 TYA ; OPCODE TO A AGAIN
F8B8: A0 03 231 LDY ##03
F8BA: E0 8A 232 CPX ##BA
F8BC: F0 08 F8C9 233 BEQ MNNDX3
F8BE: 4A 234 LSR A
F8BF: 90 08 F8C9 235 BCC MNNDX3 ; FORM INDEX INTO MNEMONIC TABLE
F8C1: 4A 236 LSR A
F8C2: 4A 237 MNNDX2
F8C3: 09 20 238 ORA ##20 ; 1) 1XXX1010 => 00101XXX
F8C5: 8B 239 DEY ; 2) XXXYY01 => 00111XXX
F8C6: D0 FA F8C2 240 BNE MNNDX2 ; 3) XXXYY10 => 00110XXX
F8C8: CB 241 INY ; 4) XXXYY100 => 00100XXX
F8C9: 8B 242 MNNDX3 ; 5) XXXXX000 => 000XXXXX
F8CA: D0 F2 F8BE 243 BNE MNNDX1
F8CC: 60 244 RTS
F8CD: FF FF FF 245 DFB
F8D0: 20 82 F8 246 INSTDSP JSR INSDS1 ; GEN FMT, LEN BYTES
F8D3: 4B 247 PHA ; SAVE MNEMONIC TABLE INDEX
F8D4: B1 3A 248 PRNTOP LDA (PCL),Y
F8D6: 20 DA FD 249 JSR PRBYTE
F8D9: A2 01 250 LDX ##01 ; PRINT 2 BLANKS
F8DB: 20 4A F9 251 PRNTBL JSR PRBL2
F8DE: C4 2F 252 CPY LENGTH ; PRINT INST (1-3 BYTES)
F8E0: C8 253 INY ; IN A 12 CHR FIELD
F8E1: 90 F1 F8D4 254 BCC PRNTOP
F8E3: A2 03 255 LDX ##03
F8E5: C0 04 256 CPY ##04 ; CHAR COUNT FOR MNEMONIC INDEX
F8E7: 90 F2 F8DB 257 BCC PRNTBL
F8E9: 68 258 PLA ; RECOVER MNEMONIC INDEX
F8EA: AB 259 TAY
F8EB: B9 C0 F9 260 LDA MNEML,Y

```

```

FBEE: 85 2C      261      STA  LMNEM      ; FETCH 3-CHAR MNEMONIC
FBF0: B9 00 FA   262      LDA  MNEMR, Y   ; (PACKED INTO 2-BYTES)
FBF3: 85 2D      263      STA  RMNEM
FBF5: A9 00      264      LDA  #00
FBF7: A0 05      265      LDY  #05
FBF9: 06 2D      266      PRMN2 ASL  RMNEM      ; SHIFT 5 BITS OF CHARACTER INTO A
FBFB: 26 2C      267      ROL  LMNEM
FBFD: 2A         268      ROL  A          ; (CLEARS CARRY)
FBFE: B8         269      DEY
FBFF: D0 FB FBF9 270      BNF  PRMN2
F901: 69 BF      271      ADC  #0BF
F903: 20 ED FD   272      JSR  COUT      ; ADD "?" OFFSET
F906: CA         273      DEX          ; OUTPUT A CHAR OF MNEM
F907: D0 EC FBF5 274      BNE  PRMN1
F909: 20 4B F9   275      JSR  PRBLNK   ; OUTPUT 3 BLANKS
F90C: A4 2F      276      LDY  LENGTH
F90E: A2 06      277      LDX  #06
F910: E0 03      278      PRADR1 CPX  #03
F912: F0 1C F930 279      BEQ  PRADR5   ; IF X=3 THEN ADDR.
F914: 06 2E      280      PRADR2 ASL  FORMAT
F916: 90 0E F926 281      BCC  PRADR3
F918: BD B3 F9   282      LDA  CHAR1-1, X
F91B: 20 ED FD   283      JSR  COUT
F91E: BD B7 F9   284      LDA  CHAR2-1, X
F921: F0 03 F926 285      BEQ  PRADR3
F923: 20 ED FD   286      JSR  COUT
F926: CA         287      PRADR3 DEX
F927: D0 E7 F910 288      BNE  PRADR1
F929: 60         289      RTS
F92A: B8         290      PRADR4 DEY
F92B: 30 E7 F914 291      BMT  PRADR2
F92D: 20 DA FD   292      JSR  PRBYTE
F930: A5 2E      293      PRADR5 LDA  FORMAT
F932: C9 E8      294      CMP  #0EB
F934: B1 3A      295      LDA  (PCL), Y ; HANDLE REL ADR MODE
F936: 90 F2 F92A 296      BCC  PRADR4   ; SPECIAL (PRINT TARGET,
F938: 20 56 F9   297      RELADR JSR  PCADJ3   ; NOT OFFSET)
F93B: AA         298      TAX
F93C: EB         299      INX
F93D: D0 01 F940 300      BNE  PRNTYX   ; +1 TO Y, X
F93F: CB         301      INY
F940: 98         302      PRNTYX TYA
F941: 20 DA FD   303      PRNTAX JSR  PRBYTE   ; OUTPUT TARGET ADR
F944: BA         304      PRNTX  TXA          ; OF BRANCH AND RETURN
F945: 4C DA FD   305      JMP  PRBYTE
F948: A2 03      306      PRBLNK LDX  #03
F94A: A9 A0      307      PRBL2 LDA  #0A0
F94C: 20 ED FD   308      PRBL3 JSR  COUT
F94F: CA         309      DEX
F950: D0 FB F94A 310      BNE  PRBL2   ; LOOP UNTIL COUNT=0
F952: 60         311      RTS
F953: 38         312      PCADJ SEC
F954: A5 2F      313      PCADJ2 LDA  LENGTH ; 0=1 BYTE, 1=2 BYTE,
F956: A4 3B      314      PCADJ3 LDY  PCH      ; 2=3 BYTE
F958: AA         315      TAX
F959: 10 01 F95C 316      BPL  PCADJ4   ; TEST DISPLACEMENT SIGN
F95B: B8         317      DEY          ; (FOR REL BRANCH)
F95C: 65 3A      318      PCADJ4 ADC  PCL
F95E: 90 01 F961 319      BCC  RTS2
F960: C8         320      INY
F961: 60         321      RTS2
F962:          322      ; FMT1 BYTES:
F962:          323      ; IF Y=0
F962:          324      ; IF Y=1
F962:          325      ; (X=INDEX)
F962: 04         326      FMT1  DFB  #04
F963: 20         327      DFB  #20
F964: 54         328      DFB  #54
F965: 30         329      DFB  #30
F966: 0D         330      DFB  #0D
F967: 80         331      DFB  #80
F968: 04         332      DFB  #04
F969: 90         333      DFB  #90
F96A: 03         334      DFB  #03
F96B: 22         335      DFB  #22
F96C: 54         336      DFB  #54
F96D: 33         337      DFB  #33
F96E: 0D         338      DFB  #0D
F96F: 80         339      DFB  #80
F970: 04         340      DFB  #04
F971: 90         341      DFB  #90
F972: 04         342      DFB  #04
F973: 20         343      DFB  #20
F974: 54         344      DFB  #54
F975: 33         345      DFB  #33
F976: 0D         346      DFB  #0D
F977: 80         347      DFB  #80
F978: 04         348      DFB  #04
F979: 90         349      DFB  #90
F97A: 04         350      DFB  #04
F97B: 20         351      DFB  #20
F97C: 54         352      DFB  #54

```

F97D: 3B	353	DFB	#3B	
F97E: 0D	354	DFB	#0D	
F97F: 80	355	DFB	#80	
F980: 04	356	DFB	#04	
F981: 90	357	DFB	#90	
F982: 00	358	DFB	#00	
F983: 22	359	DFB	#22	
F984: 44	360	DFB	#44	
F985: 33	361	DFB	#33	
F986: 0D	362	DFB	#0D	
F987: C8	363	DFB	#C8	
F988: 44	364	DFB	#44	
F989: 00	365	DFB	#00	
F98A: 11	366	DFB	#11	
F98B: 22	367	DFB	#22	
F98C: 44	368	DFB	#44	
F98D: 33	369	DFB	#33	
F98E: 0D	370	DFB	#0D	
F98F: C8	371	DFB	#C8	
F990: 44	372	DFB	#44	
F991: A9	373	DFB	#A9	
F992: 01	374	DFB	#01	
F993: 22	375	DFB	#22	
F994: 44	376	DFB	#44	
F995: 33	377	DFB	#33	
F996: 0D	378	DFB	#0D	
F997: 80	379	DFB	#80	
F998: 04	380	DFB	#04	
F999: 90	381	DFB	#90	
F99A: 01	382	DFB	#01	
F99B: 22	383	DFB	#22	
F99C: 44	384	DFB	#44	
F99D: 33	385	DFB	#33	
F99E: 0D	386	DFB	#0D	
F99F: 80	387	DFB	#80	
F9A0: 04	388	DFB	#04	
F9A1: 90	389	DFB	#90	
F9A2: 26	390	DFB	#26	
F9A3: 31	391	DFB	#31	
F9A4: 87	392	DFB	#87	
F9A5: 9A	393	DFB	#9A	
F9A6: 00	394 ; ZZXXXY01 INSTR 'S			
F9A6: 00	395 FMT2	DFB	#00	; ERR
F9A7: 21	396	DFB	#21	; IMM
F9A8: 81	397	DFB	#81	; Z-PAGE
F9A9: 82	398	DFB	#82	; ABS
F9AA: 00	399	DFB	#00	; IMPLIED
F9AB: 00	400	DFB	#00	; ACCUMULATOR
F9AC: 59	401	DFB	#59	; (ZPAG, X)
F9AD: 4D	402	DFB	#4D	; (ZPAG, Y)
F9AE: 91	403	DFB	#91	; ZPAG, X
F9AF: 92	404	DFB	#92	; ABS, X
F9B0: 86	405	DFB	#86	; ABS, Y
F9B1: 4A	406	DFB	#4A	; (ABS)
F9B2: 85	407	DFB	#85	; ZPAG, Y
F9B3: 9D	408	DFB	#9D	; RELATIVE
F9B4: AC	409 CHAR1	DFB	#AC	; ' '
F9B5: A9	410	DFB	#A9	; ' ' '
F9B6: AC	411	DFB	#AC	; ' ' '
F9B7: A3	412	DFB	#A3	; ' # ' '
F9B8: AB	413	DFB	#AB	; ' (' '
F9B9: A4	414	DFB	#A4	; ' \$ ' '
F9BA: D9	415 CHAR2	DFB	#D9	; ' Y ' '
F9BB: 00	416	DFB	#00	; ' ' ' '
F9BC: DB	417	DFB	#DB	; ' Y ' '
F9BD: A4	418	DFB	#A4	; ' \$ ' '
F9BE: A4	419	DFB	#A4	; ' \$ ' '
F9BF: 00	420	DFB	#00	
F9C0: 1C	421 MNEML	DFB	#1C	
F9C1: BA	422	DFB	#BA	
F9C2: 1C	423	DFB	#1C	
F9C3: 23	424	DFB	#23	
F9C4: 5D	425	DFB	#5D	
F9C5: 8B	426	DFB	#8B	
F9C6: 1B	427	DFB	#1B	
F9C7: A1	428	DFB	#A1	
F9C8: 9D	429	DFB	#9D	
F9C9: 8A	430	DFB	#8A	
F9CA: 1D	431	DFB	#1D	
F9CB: 23	432	DFB	#23	
F9CC: 9D	433	DFB	#9D	
F9CD: 8B	434	DFB	#8B	
F9CE: 1D	435	DFB	#1D	
F9CF: A1	436	DFB	#A1	
F9D0: 00	437	DFB	#00	
F9D1: 29	438	DFB	#29	
F9D2: 19	439	DFB	#19	
F9D3: AE	440	DFB	#AE	
F9D4: 69	441	DFB	#69	
F9D5: AB	442	DFB	#AB	
F9D6: 19	443	DFB	#19	
F9D7: 23	444	DFB	#23	

F9D8: 24	445	DFB	\$24	
F9D9: 53	446	DFB	\$53	
F9DA: 1B	447	DFB	\$1B	
F9DB: 23	448	DFB	\$23	
F9DC: 24	449	DFB	\$24	
F9DD: 53	450	DFB	\$53	
F9DE: 19	451	DFB	\$19	; (A) FORMAT ABOVE
F9DF: A1	452	DFB	\$A1	
F9E0: 00	453	DFB	\$00	
F9E1: 1A	454	DFB	\$1A	
F9E2: 5B	455	DFB	\$5B	
F9E3: 5B	456	DFB	\$5B	
F9E4: A5	457	DFB	\$A5	
F9E5: 69	458	DFB	\$69	
F9E6: 24	459	DFB	\$24	; (B) FORMAT
F9E7: 24	460	DFB	\$24	
F9E8: AE	461	DFB	\$AE	
F9E9: AE	462	DFB	\$AE	
F9EA: AB	463	DFB	\$AB	
F9EB: AD	464	DFB	\$AD	
F9EC: 29	465	DFB	\$29	
F9ED: 00	466	DFB	\$00	
F9EE: 7C	467	DFB	\$7C	; (C) FORMAT
F9EF: 00	468	DFB	\$00	
F9F0: 15	469	DFB	\$15	
F9F1: 9C	470	DFB	\$9C	
F9F2: 6D	471	DFB	\$6D	
F9F3: 9C	472	DFB	\$9C	
F9F4: A5	473	DFB	\$A5	
F9F5: 69	474	DFB	\$69	
F9F6: 29	475	DFB	\$29	; (D) FORMAT
F9F7: 53	476	DFB	\$53	
F9F8: B4	477	DFB	\$B4	
F9F9: 13	478	DFB	\$13	
F9FA: 34	479	DFB	\$34	
F9FB: 11	480	DFB	\$11	
F9FC: A5	481	DFB	\$A5	
F9FD: 69	482	DFB	\$69	
F9FE: 23	483	DFB	\$23	; (E) FORMAT
F9FF: A0	484	DFB	\$A0	
FA00: DB	485	MNEMR	DFB	\$DB
FA01: 62	486	DFB	\$62	
FA02: 5A	487	DFB	\$5A	
FA03: 4B	488	DFB	\$4B	
FA04: 26	489	DFB	\$26	
FA05: 62	490	DFB	\$62	
FA06: 94	491	DFB	\$94	
FA07: 8B	492	DFB	\$8B	
FA08: 54	493	DFB	\$54	
FA09: 44	494	DFB	\$44	
FA0A: CB	495	DFB	\$CB	
FA0B: 54	496	DFB	\$54	
FA0C: 6B	497	DFB	\$6B	
FA0D: 44	498	DFB	\$44	
FA0E: EB	499	DFB	\$EB	
FA0F: 94	500	DFB	\$94	
FA10: 00	501	DFB	\$00	
FA11: B4	502	DFB	\$B4	
FA12: 0B	503	DFB	\$0B	
FA13: B4	504	DFB	\$B4	
FA14: 74	505	DFB	\$74	
FA15: B4	506	DFB	\$B4	
FA16: 2B	507	DFB	\$2B	
FA17: 6E	508	DFB	\$6E	
FA18: 74	509	DFB	\$74	
FA19: F4	510	DFB	\$F4	
FA1A: CC	511	DFB	\$CC	
FA1B: 4A	512	DFB	\$4A	
FA1C: 72	513	DFB	\$72	
FA1D: F2	514	DFB	\$F2	
FA1E: A4	515	DFB	\$A4	; (A) FORMAT
FA1F: 8A	516	DFB	\$8A	
FA20: 00	517	DFB	\$00	
FA21: AA	518	DFB	\$AA	
FA22: A2	519	DFB	\$A2	
FA23: A2	520	DFB	\$A2	
FA24: 74	521	DFB	\$74	
FA25: 74	522	DFB	\$74	
FA26: 74	523	DFB	\$74	; (B) FORMAT
FA27: 72	524	DFB	\$72	
FA28: 44	525	DFB	\$44	
FA29: 6B	526	DFB	\$6B	
FA2A: B2	527	DFB	\$B2	
FA2B: 32	528	DFB	\$32	
FA2C: B2	529	DFB	\$B2	
FA2D: 00	530	DFB	\$00	
FA2E: 22	531	DFB	\$22	; (C) FORMAT
FA2F: 00	532	DFB	\$00	
FA30: 1A	533	DFB	\$1A	
FA31: 1A	534	DFB	\$1A	
FA32: 26	535	DFB	\$26	
FA33: 26	536	DFB	\$26	

```

FA34: 72          537          DFB  #72
FA35: 73          538          DFB  #72
FA36: 8B          539          DFB  #8B
FA37: CB          540          DFB  #CB      ; (D) FORMAT
FA38: C4          541          DFB  #C4
FA39: CA          542          DFB  #CA
FA3A: 26          543          DFB  #26
FA3B: 4B          544          DFB  #4B
FA3C: 44          545          DFB  #44
FA3D: 44          546          DFB  #44
FA3E: A2          547          DFB  #A2      ; (E) FORMAT
FA3F: CB          548          DFB  #CB
FA40: 85 45      549          STA  ACC      ; *** IRQ HANDLER
FA41: 85 45      549          PLA
FA42: 68          550          PLA
FA43: 48          551          PHA
FA44: 0A          552          ASL  A
FA45: 0A          553          ASL  A
FA46: 0A          554          ASL  A
FA47: 30 03      555          BMI  BREAK    ; TEST FOR 'BRK'
FA48: 6C FE 03   556          JMP  (IRGLOC) ; USER ROUTINE VECTOR IN RAM
FA49: 2B          557          PLP
FA4A: 20 4C FF   558          JSR  SAV1     ; SAVE REG'S ON BREAK
FA4B: 68          559          PLA         ; INCLUDING PC
FA4C: 85 3A      560          STA  PCL
FA4D: 68          561          PLA
FA4E: 85 3B      562          STA  PCH
FA4F: 6C FO 03   563          JMP  (BRKV)   ; BRKV WRITTEN OVER BY DISK BOOT
FA50: 20 B2 FB   564          JSR  INSDS1   ; PRINT USER PC
FA51: 20 DA FA   565          JSR  RGDSP1   ; AND REGS
FA52: 4C 65 FF   566          JMP  MDN      ; GO TO MONITOR (NO PASS GO, NO #200!)
FA53: DB          567          CLD         ; DO THIS FIRST THIS TIME
FA54: 20 B4 FE   568          JSR  SETNORM
FA55: 20 2F FB   569          JSR  INIT
FA56: 20 93 FE   570          JSR  SETVID
FA57: 20 89 FE   571          JSR  SETKBD
FA58: AD 5B C0    572          LDA  SETANO   ; AN0 = TTL HI
FA59: AD 5A C0    573          LDA  SETAN1   ; AN1 = TTL HI
FA60: 0001       574          DO   APPLE2E  ; /RRA09B1
FA61: A0 05      575          LDY  #5       ; CODE=INIT/RRA09B1
FA62: 20 B4 FB   576          JSR  GOTOCX   ; DO APPLE2E INIT/RRA09B1
FA63: EA        577          NOP
FA64: S          578          ELSE
FA65: S          579          LDA  CLRAN2   ; AN2 = TTL LO
FA66: S          580          LDA  CLRAN3   ; AN3 = TTL LO
FA67: S          581          FIN
FA68: AD FF CF   582          LDA  CLRROM   ; TURN OFF EXTNSN ROM
FA69: 2C 10 C0   583          BIT  KBDSTRB  ; CLEAR KEYBOARD
FA70: DB          584          NEWMON      CLD
FA71: 20 3A FF   585          JSR  BELL     ; CAUSES DELAY IF KEY BOUNCES
FA72: AD F3 03   586          LDA  SFTEV+1 ; IS RESET HI
FA73: 49 A5      587          EOR  #A5     ; A FUNNY COMPLEMENT OF THE
FA74: CD F4 03   588          CMP  PWRDUP   ; PWR UP BYTE ???
FA75: D0 17 FAA6 589          BNE  PWRUP    ; NO SO PWRUP
FA76: AD F2 03   590          LDA  SDFTEV   ; YES SEE IF COLD START
FA77: D0 9F FAA3 591          BNE  NOFIX    ; HAS BEEN DONE YET?
FA78: A9 E0      592          LDA  #E0     ; #E0
FA79: CD F3 03   593          CMP  SDFTEV+1 ; DOES SOFT ENTRY VECTOR POINT AT BASIC?
FA80: D0 08 FAA3 594          BNE  NOFIX    ; YES SO REENTER SYSTEM
FA81: A0 03      595          LDY  #3      ; NO SO POINT AT WARM START
FA82: 8C F2 03   596          STY  SDFTEV  ; FOR NEXT RESET
FA83: 4C 00 E0   597          JMP  BASIC    ; AND DO THE COLD START
FA84: 6C F2 03   598          JMP  (SDFTEV) ; SOFT ENTRY VECTOR
FA85: A6         599          *****
FA86: 20 60 FB   600          JSR  APPLE11  ; APPLE11
FA87: A9 FAA9    601          EQU  *
FA88: A2 05      602          LDX  #5
FA89: BD FC FA   603          SETPLP      ; WITH CNTRL B ADRS
FA90: 9D EF 03   604          STA  BRKV-1.X ; OF CURRENT BASIC
FA91: CA        605          DEX
FA92: D0 F7 FAAB 606          BNE  SETPLP
FA93: A9 C8      607          LDA  #CB     ; LOAD HI SLOT +1
FA94: 86 00      608          STX  LOCO    ; SETPG3 MUST RETURN X=0
FA95: 85 01      609          STA  LOC1    ; SET PTR H
FA96: A0 07      610          LDY  #7      ; Y IS BYTE PTR
FA97: C6 01      611          DEC  LOC1
FA98: A5 01      612          LDA  LOC1
FA99: C9 C0      613          CMP  #C0     ; AT LAST SLOT YET?
FA100: F0 D7 FA9B 614          BEQ  FIXSEV  ; YES AND IT CAN'T BE A DISK
FA101: 8D FB 07   615          STA  MSL0T
FA102: B1 00      616          LDA  (LOC0),Y ; FETCH A SLOT BYTE
FA103: D9 01 FB   617          CMP  DISKID-1,Y ; IS IT A DISK ??
FA104: D0 EC FABA 618          BNE  SLODP   ; NO, SO NEXT SLOT DOWN
FA105: 88         619          DEY
FA106: 8B        620          DEY
FA107: 10 F5 FAC7 621          BPL  NXTBYT  ; YES, SO CHECK NEXT BYTE
FA108: 6C 00 00   622          JMP  (LOCO)  ; UNTIL 4 BYTES CHECKED
FA109: EA        623          NOP
FA110: EA        624          NOP
FA111: 20 BE FD   625          * REGDSP MUST DRG #FAD7
FA112: A9 45      626          JSR  CROUT   ; DISPLAY USER REG CONTENTS
FA113: 6C 00 00   627          JMP  (LOCO)  ; WITH LABELS
FA114: EA        628          STA  A3L

```

```

FADE: A9 00      629      LDA    ##00
FAE0: 85 41      630      STA    ACH
FAE2: A2 FB      631      LDX    ##FB
FAE4: A9 A0      632      LDA    ##A0
FAE6: 20 ED FD   633      JSR    CDUT
FAE9: BD 1E FA   634      LDA    RTBL-251.X
FAEC: 20 ED FD   635      JSR    CDUT
FAEF: A9 BD      636      LDA    ##BD
FAF1: 20 ED FD   637      JSR    CDUT
FAF4:             638      * LDA ACC+9, X
FAF4: B5 4A      639      DFB   $B5, $4A
FAF6: 20 DA FD   640      JSR    PRBYTE
FAF9: EB         641      INX
FAFA: 30 E8 FAE4 642      BPL   RDSP1
FAFC: 60         643      RTS
FAFD: 59 FA      644      PWRCON DW    QLDDBRK
FAFF: 00 E0 45   645      DFB   $00, $E0, $45
FB02: 20 FF 00 FF 646      DISKID DFB   $20, $FF, $00, $FF
FB06: 03 FF 3C   647      DFB   $03, $FF, $3C
FB09: C1 F0 EC   648      TITLE ASC    'Apple' II'
FB11:             FB11 649      XLTLB EQU    *
FB11: C4 C2 C1   650      DFB   $C4, $C2, $C1
FB14: FF C3      651      DFB   $FF, $C3
FB16: FF FF FF   652      DFB   $FF, $FF, $FF
FB19:             653      * MUST ORG $FB19
FB19: C1 D8 D9   654      RTBL  DFB   $C1, $D8, $D9 ; REGISTER NAMES FOR REGDSP:
FB1C: D0 D3      655      DFB   $D0, $D3 ; 'AXYPS'
FB1E: AD 70 C0   656      PREAD LDA    PTRIG ; TRIGGER PADDLES
FB21: A0 00      657      LDY    ##00 ; INIT COUNT
FB23: EA         658      NOP ; COMPENSATE FOR 1ST COUNT
FB24: EA         659      NOP
FB25: BD 64 C0   660      PREAD2 LDA   PADDLO, X ; COUNT Y-REG EVERY 12 USEC.
FB28: 10 04 FB2E 661      BPL   RTS2D
FB2A: CB         662      INY
FB2B: D0 F8 FB25 663      BNE   PREAD2 ; EXIT AT 255 MAX
FB2D: BB         664      DEY
FB2E: 60         665      RTS2D
FB2F:             666      CHN    BJS. SRC2
FB2F:             1 *
FB2F: A9 00      2 INIT  LDA    ##00 ; CLR STATUS FOR DEBUG SOFTWARE
FB31: 85 48      3      STA    STATUS
FB33: AD 56 C0   4      LDA    LDRS
FB36: AD 54 C0   5      LDA    LOWSCR ; INIT VIDED MODE
FB39: AD 51 C0   6      SETTXT LDA   TXTSET ; SET FOR TEXT MODE
FB3C: A9 00      7      LDA    ##90 ; FULL SCREEN WINDOW
FB3E: F0 08 FB4B 8      BEQ   SETWND
FB40: AD 50 C0   9      SETOR  LDA   TXTCLR ; SET FOR GRAPHICS MODE
FB43: AD 53 C0   10     LDA    MIXSET ; LOWER 4 LINES AS TEXT WINDOW
FB46: 20 36 F8   11     JSR    CLRTOP
FB49: A9 14      12     LDA    ##14
FB4B: 85 22      13     SETWND STA   WNDDTOP ; SET FOR 40 CDL WINDOW
FB4D: A9 00      14     LDA    ##00 ; TOP IN A-REG,
FB4F: 85 20      15     STA   WNDLFT ; BOTTOM AT LINE #24
FB51:             16     DD    APPLEZE ; /RRA09B1
FB51: A0 08      17     LDY    ; CODE-SETWND /RRA09B1
FB53: D0 5F FB44 18     BNE   GOTOCX ; DD 40/80 /RRA09B1
FB55:             19     ELSE  ; /RRA09B1
S             20     LDA    ##28
S             21     STA   WNDWDTH
FB55:             22     FIN ; /RRA09B1
FB55: A9 18      23     LDA    ##18
FB57: 85 23      24     STA   WNDBTM
FB59: A9 17      25     LDA    ##17 ; VTAB TO ROW 23
FB5B: 85 25      26     TABV  STA   CV ; VTABS TO ROW IN A-REG
FB5D: 4C 22 FC   27     JMP   VTB ; CLEAR THE SCRN
FB60: 20 58 FC   28     APPLEII JSR   HDME ;
FB63: A0 08      29     LDY    #8 ; GET A CHAR
FB65: B9 08 FB   30     STITLE LDA   TITLE-1, Y ; PUT IT AT TOP CENTER OF SCREEN
FB68: 99 0E 04   31     STA   LINE+14, Y
FB6B: 88         32     DEY
FB6C: D0 F7 FB65 33     BNE   STITLE
FB6E: 60         34     RTS
FB6F: AD F3 03   35     SETPWRC LDA   SOFTEV+1 ; ROUTINE TO CALCULATE THE 'FUNNY
FB72: 49 A5      36     EOR   ##A5 ; COMPLEMENT' FOR THE RESET VECTOR
FB74: BD F4 03   37     STA   PWREDUP
FB77: 60         38     RTS
FB7B:             39     EQU    * ; CHECK FOR A PAUSE (CONTROL-S).
FB7B: C9 8D FB7B 40     CMP   ##BD ; ONLY WHEN I HAVE A CR
FB7A: D0 18 FB94 41     BNE   NOWAIT ; NOT SO, DO REGULAR
FB7C: AC 00 C0   42     LDY    KBD ; IS KEY PRESSED?
FB7F: 10 13 FB94 43     BPL   NOWAIT ; NO.
FB81: C0 93      44     CPY    ##93 ; YES -- IS IT CTRL-S?
FB83: D0 0F FB94 45     BNE   NOWAIT ; NOPE - IONORE
FB85: 2C 10 C0   46     BIT   KBDSTRB ; CLEAR STROBE
FB8B: AC 00 C0   47     KBDWAIT LDY   KBD ; WAIT TILL NEXT KEY TO RESUME
FB8B: 10 FB FB8B 48     BPL   KBDWAIT ; WAIT FOR KEYPRESS
FB8D: C0 83      49     CPY    ##83 ; IS IT CONTROL-C?
FB8F: F0 03 FB94 50     BEQ   NOWAIT ; YES, SO LEAVE IT
FB91: 2C 10 C0   51     BIT   KBDSTRB ; CLR STROBE
FB94: 4C FD FB   52     JMP   VIDDUT ; DO AS BEFORE
FB97: 38         53     ESCOLD SEC ; INSURE CARRY SET

```



```

FB98: 4C 2C FC      54      JMP      ESC1
FB98: AB           55      ESCNDW   TAY
FB9C: 09 4B FA      56      LDA      XLTBL-#C9,Y ; USE CHAR AS INDEX
FB9F: 20 97 FB      57      JSR      ESCOLD ; TRANSLATE IJKM TO CBAD
FBA2:           0001 58      DO      APPLE2E ; DO THE CURSOR MOTION
FBA2: 20 21 FD      59      JSR      RDESC ; FOR FULL ESCAPES/RAA09B1
FBA5:           60      ELSE
S           61      JSR      RDKEY ; GET IJKM, IJkm, ARROWS/RAA09B1
FBA5:           62      FIN ; /RAA09B1
FBA5: C9 CE         63      ESCNEW  CMP      ##CE ; AND GET NEXT
FBA7: B0 EE       FB97 64      BCS     ESCOLD ; /RAA09B1
FBA9: C9 C9       FB97 65      CMP     #*C9 ; IS THIS AN 'N'?
FBA9: 90 EA       FB97 66      BCC     ESCOLD ; 'N' OR GREATER -- DO IT!
FBA9: C9 CC       FB97 67      CMP     #*CC ; LESS THAN 'I'?
FBAF: F0 E6       FB97 68      BEQ     ESCOLD ; YES, SO DO OLD WAY
FBB1: D0 E8       FB9B 69      BNE     ESCNDW ; IS IT AN 'L'?
FBB3:           0001 70      DO      APPLE2E ; DO NORMAL
FBB3: C006        71      SETSLOTXROM EQU ; GO DO IT
FBB3: C007        72      SETINTCXROM EQU ; /RAA09B1
FBB3: C015        73      RDCXROM EQU ; #C006
FBB3:           74      * ; /RAA09B1
FBB3: 06         FB94 75      VERSION DFB     #06 ; FOR IDCHECK/RAA09B1
FBB4:           76      GOTOCX  EQU     * ; /RAA09B1
FBB4: 08         77      PHP
FBB5: 7B         78      SEI
FBB6: 2C 15 CO    79      BIT     RDCXROM ; SAVE USER IRQ STATE/RAA09B1
FBB7: 0B         80      PHP
FBB8: 8D 07 CO    81      STA     SETINTCXROM ; INHBIT DURING BANKSWITCH/RAA09B1
FBB8: 4C 00 C1    82      JMP     #C100 ; GET CURRENT STATE/RAA09B1
FBC0:           83      * ; SAVE ROMBANK STATE/RAA09B1
FBC0: EA         84      NOP ; SET ROMS ON/RAA09B1
FBC1:           85      ELSE ; =>OFF TO CXSPACE/RAA09B1
S           86      NOP ; /RAA09B1
S           87      NOP ; "I GOT PLENTY OF NOTHING!"
S           88      NOP
S           89      NOP
S           90      NOP
S           91      NOP
S           92      NOP
S           93      NOP
S           94      NOP
S           95      NOP
S           96      NOP
S           97      NOP
S           98      NOP
S           99      NOP
FBC1:           100     FIN ; /RAA09B1
FBC1:           101     * MUST ORG #FBC1
FBC1: 4B         102     BASCALC PHA ; CALCBASE ADDR IN BASL,H
FBC2: 4A         103     LSR     A ; FOR GIVEN LINE NO.
FBC3: 29 03       104     AND     #*03 ; 0<=LINE NO.<=#17
FBC3: 09 04       105     ORA     #*04 ; ARG=000ABCDE, GENERATE
FBC7: B5 29       106     STA     BASH ; BASH=000001CD
FBC9: 5B         107     PLA
FBCA: 29 1B       108     AND     #*1B ; AND
FBCB: 90 02       FBDD 109     BCC     BASCLC2 ; BASL=EABAB000
FBCB: 69 7F       110     ADC     #*7F
FBD0: B5 2B       111     BASCLC2 STA     BASL
FBD2: 0A         112     ASL     A
FBD3: 0A         113     ASL     A
FBD4: 05 2B       114     ORA     BASL
FBD6: B5 2B       115     STA     BASL
FBD8: 60         116     RTS
FBD9: C9 B7       117     BELL1  CMP     #*B7 ; BELL CHAR? (CONTROL-G)
FBD9: D0 12       FBEB 118     BNE     RTS2B ; NO. RETURN.
FBD9: A9 40       119     LDA     #*40 ; YES.
FBD9: 20 AB FC    120     JSR     WAIT ; DELAY .01 SECONDS
FBE2: A0 C0       121     LDY     #*C0
FBE4: A9 0C       122     BELL2  LDA     #*0C ; TOGGLE SPEAKER AT 1 KHZ
FBE6: 20 AB FC    123     JSR     WAIT ; FOR .1 SEC.
FBE9: AD 30 C0    124     LDA     SPKR
FBE9: B8         125     DEY
FBE9: D0 F5       FBE4 126     BNE     BELL2
FBE9: 60         127     RTS2B  RTS
FBF0: A4 24       128     STORADV LDY     CH ; CURSOR H INDEX TO Y-REG
FBF2: 91 2B       129     STA     (BASL),Y ; STORE CHAR IN LINE
FBF4: E6 24       130     ADVANCE INC     CH ; INCREMENT CURSOR H INDEX
FBF6: A5 24       131     LDA     CH ; (MOVE CURSOR)
FBF8: C5 21       132     CMP     WNDWDTH ; BEYOND WINDOW WIDTH?
FBFA: B0 66       FC62 133     BCS     CR ; YES, CR TO NEXT LINE.
FBFC: 60         134     RTS3   RTS ; NO. RETURN.
FBFD: C9 A0       135     VIDOUT  CMP     #*A0 ; CONTROL CHAR?
FBFF: B0 EF       FBFO 136     BCS     STORADV ; NO. OUTPUT IT.
FC01: AB         137     TAY ; INVERSE VIDEO?
FC02: 10 EC       FBFO 138     BPL     STORADV ; YES, OUTPUT IT.
FC04: C9 BD       139     CMP     #*BD ; CR?
FC06: F0 5A       FC62 140     BEQ     CR ; YES.
FC08: C9 BA       141     CMP     #*BA ; LINE FEED?
FC0A: F0 5A       FC66 142     BEQ     LF ; IF SO, DO IT.
FC0C: C9 BB       143     CMP     #*BB ; BACK SPACE? (CONTROL-H)
FC0E: D0 C9       FBD9 144     BNE     BELL1 ; NO. CHECK FOR BELL.
FC10: C6 24       145     BS     DEC     CH ; DECREMENT CURSOR H INDEX

```

```

FC12: 10 EB FBFC 146 BPL RTS3 ; IF POSITIVE, OK; ELSE MOVE UP.
FC14: A5 21 147 ; SET CH TO WINDOW WIDTH - 1.
FC16: 85 24 148 STA CH
FC18: C6 24 149 DEC CH ; (RIGHTMOST SCREEN POS)
FC1A: A5 22 150 UP LDA WNDTOP ; CURSOR V INDEX
FC1C: C5 25 151 CMP CV
FC1E: B0 08 FC2B 152 BCS RTS4 ; IF TOP LINE THEN RETURN
FC20: C6 25 153 DEC CV ; DECR CURSOR V INDEX
FC22: A5 25 154 VTAB LDA CV ; GET CURSOR V INDEX
FC24: 20 C1 FB 155 VTABZ JSR BASCALC ; GENERATE BASE ADDRESS
FC27: 65 20 156 JSR WNDLFT ; ADD WINDOW LEFT OFFSET
FC29: B5 28 157 STA BASL ; TO BASL
FC2B: 60 158 RTS4 RTS
FC2C: 49 C0 159 ESC1 EOR ##C0 ; ESC '?
FC2E: F0 28 FC5B 160 BEQ HOME ; IF SO DO HOME AND CLEAR
FC30: 6F FD 161 ADC ##FD ; ESC-A OR B CHECK
FC32: 90 C0 FBF4 162 BCC ADVANCE ; A, ADVANCE
FC34: F0 DA FC10 163 BEQ BS ; B, BACKSPACE
FC36: 6F FD 164 ADC ##FD ; ESC-C OR D CHECK
FC38: 90 2C FC66 165 BCC LF ; C, DOWN
FC3A: F0 DE FC1A 166 BEQ UP ; D, GO UP
FC3C: 6F FD 167 ADC ##FD ; ESC-E OR F CHECK
FC3E: 90 5C FC9C 168 BCC CLREDL ; E, CLEAR TO END OF LINE
FC40: D0 E9 FC2B 169 BNE RTS4 ; ELSE NOT F.RETURN
FC42: 0001 170 DO APPLE2E ; /RAA09B1
FC42: FC42 171 CLREDP EGU * ; /RAA09B1
FC42: A0 00 172 LDY #0 ; CODE=CLREDP/RAA09B1
FC44: F0 2C FC72 173 BEG XGOTOCX ; DO 40/80 /RAA09B1
FC46: AB C3 A9 A0 174 ASC '(C) 1981-82, APPLE
FC5B: 175 ELSE ; /RAA09B1
S 176 CLREOP LDY CH ; ESC F IS CLR TO END OF PAGE
S 177 LDA CV
S 178 CLEDP1 PHA ; SAVE CURRENT LINE NO. ON STACK
S 179 JSR VTABZ ; CALC BASE ADDRESS
S 180 JSR CLEDLZ ; CLEAR TO EOL. (SETS CARRY)
S 181 LDY ##00 ; CLEAR FROM H INDEX=0 FOR REST
S 182 PLA ; INCREMENT CURRENT LINE NO.
S 183 ADC ##00 ; (CARRY IS STILL SET)
S 184 CMP WNDBTM ; DONE TO BOTTOM OF WINDOW?
S 185 BCC CLEDP1 ; NO, KEEP CLEARING LINES
S 186 BCS VTAB ; YES, VTAB TO CURRENT LINE.
FC5B: 187 FIN ; /RAA09B1
FC5B: 0001 188 DO APPLE2E ; /RAA09B1
FC5B: FC5B 189 HOME EGU * ; /RAA09B1
FC5B: A0 01 190 LDY #1 ; CODE=HOME/RAA09B1
FC5A: D0 16 FC72 191 BNE XGOTOCX ; DO 40/80 /RAA09B1
FC5C: D2 C9 C3 CB 192 ASC 'RICK A ; DUR HERD...
FC62: 193 ELSE ; /RAA09B1
S 194 HOME LDA WNDTOP ; INIT CURSOR V
S 195 STA CV ; AND H INDICES
S 196 LDY ##00
S 197 STY CH ; THEN CLEAR TO END OF PAGE.
S 198 BEQ CLEDP1 ; (ALWAYS TAKEN)
FC62: 199 FIN ; /RAA09B1
FC62: A9 00 200 CR LDA ##00 ; CURSOR TO LEFT OF INDEX
FC64: B5 24 201 STA CH ; (RET CURSOR H=0)
FC66: E6 25 202 LF INC CV ; INCR CURSOR V. (DOWN 1 LINE)
FC68: A5 25 203 LDA CV
FC6A: C5 23 204 CMP WNDBTM ; OFF SCREEN?
FC6C: 90 B6 FC24 205 BCC VTABZ ; NO, SET BASE ADDR
FC6E: C6 25 206 DEC CV ; DECR CURSOR V. (BACK TO BOTTOM)
FC70: 0001 207 DO APPLE2E ; /RAA09B1
FC70: FC70 208 SCROLL EGU * ; /RAA09B1
FC70: A0 02 209 LDY #2 ; CODE=SCROLL/RAA09B1
FC72: 4C B4 FB 210 XGOTOCX JMP GOTOCX ; DO 40/80 /RAA09B1
FC75: 211 *
FC75: 212 * IRQ SNIFFER FOR VIDEO CODE:
FC75: 213 *
FC75: C01B 214 RDBSTORE EGU #C01B ; /RAA09B1
FC75: C01C 215 RDPAGE2 EGU #C01C ; /RAA09B1
FC75: 4B 216 PHA ; PRESERVE AC/RAA09B1
FC75: AD 18 C0 217 LDA RDBSTORE ; FLAG->N /RAA09B1
FC79: DA 218 ASL A ; FLAG->C /RAA09B1
FC7A: 6B 219 PLA ; RESTORE AC/RAA09B1
FC7B: 2C 1C C0 220 BIT RDPAGE2 ; FLAG->N /RAA09B1
FC7E: 0B 221 PHP ; /RAA09B1
FC7F: 90 03 FC84 222 BCC RDCX ; NOT BANKSWITCHING/RAA09B1
FC81: BD 54 C0 223 STA #C054 ; FORCE MB TXTPAGE/RAA09B1
FC84: 2C 15 C0 224 RDCX BIT RDCXROM ; FLAG->N /RAA09B1
FC87: BD 06 C0 225 STA SETSLOTCXROM ; RESTORE BANK/RAA09B1
FC8A: 5B 226 CLI ; ENABLE IRG/RAA09B1
FC8B: 7B 227 SEI ; NDW DISABLE/RAA09B1
FC8C: 10 03 FC91 228 BPL ISSLOTS ; =>AS SLOTS/RAA09B1
FC8E: BD 07 C0 229 STA SETINTCXROM ; BANK-IN CX/RAA09B1
FC91: 2B 230 EQU * ; /RAA09B1
FC91: 2B 231 PLP ; WHAT VID BANK/PAGE?/RAA09B1
FC92: 90 05 FC99 232 BCC ISPAGE1 ; =>NOT BANKED/RAA09B1
FC94: 10 03 FC99 233 BPL ISPAGE1 ; IT'S PAGE1/RAA09B1
FC96: 2C 55 C0 234 BIT #C055 ; FORCE PAGE2/RAA09B1
FC99: FC99 235 EQU * ; /RAA09B1
FC99: 60 236 RTS ; CONTINUE VIDEO/RAA09B1

```

```

FC9A: EA      237      NOP      ; /RRA09B1
FC9B: EA      238      NOP      ; /RRA09B1
FC9C:         239      ELSE
S           240      SCROLL LDA      WNDTOP ; START AT TOP OF SCROLL WINDOW
S           241      PHA
S           242      JSR      VTABZ ; GENERATE BASE ADDRESS
S           243      SCRL1  LDA      BASL  ; COPY BASL,H
S           244      STA      BAS2L ; TO BAS2L,H
S           245      LDA      BASH
S           246      STA      BAS2H
S           247      LDY      WNDWDTH ; INIT Y TO RIGHTMOST INDEX
S           248      DEY      ; OF SCROLLING WINDOW
S           249      PLA
S           250      ADC      #*01 ; INCR LINE NUMBER
S           251      CMP      WNDBTM ; DONE?
S           252      BCS      SCRL3 ; YES, FINISH
S           253      PHA
S           254      JSR      VTABZ ; FORM BASL,H (BASE ADDR)
S           255      SCRL2  LDA      (BASL),Y ; MOVE A CHAR UP ONE LINE
S           256      STA      (BAS2L),Y
S           257      DEY      ; NEXT CHAR OF LINE
S           258      BPL      SCRL2
S           259      BMI      SCRL1 ; NEXT LINE
S           260      SCRL3  LDY      #*00 ; CLEAR BOTTOM LINE
S           261      JSR      CLEOLZ ; GET BASE ADDR FOR BOTTOM LINE
S           262      BCS      VTAB ; CARRY IS SET
FC9C:         263      FIN
FC9C:         0001 264      DO      APPLE2E ; /RRA09B1
FC9C:         FC9C 265      CLREOL EQU      * ; /RRA09B1
FC9C: 1B       266      CLC
FC9D: B0       267      DFB     #B0 ; SAY 'EOL' /RRA09B1
FC9E:         FC9E 268      CLREOLZ EQU     * ; /RRA09B1
FC9E: 3B       269      SEC
FC9F: B4 1F    270      STY     #1F ; VIDEO'S YSAV1/RRA09B1
FCA1: A0 03    271      LDY     #3 ; CODE=EOL/RRA09B1
FCA3: 90 CD    FC72 272      BCC     XGOTDCX ; ->IT'S EOL/RRA09B1
FCA5: CB       273      INY
FCA6: D0 CA    FC72 274      BNE     XGOTDCX ; CODE=EOLZ/RRA09B1
FCA8:         275      ELSE ; ->ALWAYS/RRA09B1
S           276      CLREOL CH ; /RRA09B1
S           277      CLEOLZ LDA     #*A0 ; CURSOR H INDEX
S           278      CLEOLZ STA     (BASL),Y ; STORE BLANKS FROM 'HERE'
S           279      INY      ; TO END OF LINES (WNDWDTH)
S           280      CFY
S           281      BCC     WNDWDTH
S           282      RTS     CLEOLZ
FCAB:         283      FIN ; /RRA09B1
FCAB: 3B       284      WAIT  SEC
FCA9: 4B       285      WAIT2 PHA
FCAA: E9 01    286      WAIT3 SBC     #*01
FCAC: D0 FC    FCAA 287      BNE     WAIT3 ; 1.0204 USEC
FCAD: 6B       288      PLA ; (13+2712*A+512*A*A)
FCAF: E9 01    289      SBC     #*01
FCB1: D0 F6    FCA9 290      BNE     WAIT2
FCB3: 60       291      RTS
FCB4: E6 42    292      NXTA4 INC     A4L ; INCR 2-BYTE A4
FCB6: D0 02    FCBA 293      BNE     NXTA1 ; AND A1
FCB8: E6 43    294      INC     A4H
FCBA: A5 3C    295      NXTA1 LDA     A1L ; INCR 2-BYTE A1
FCBC: C5 3E    296      CMP     A2L ; AND COMPARE TO A2
FCBE: A5 3D    297      LDA     A1H ; (CARRY SET IF >=)
FCC0: E5 3F    298      SBC     A2H
FCC2: E6 3C    299      INC     A1L
FCC4: D0 02    FCCB 300      BNE     RTS4B
FCC6: E6 3D    301      INC     A1H
FCC8: 60       302      RTS4B RTS
FCC9: A0 4B    303      HEADR LDY     #*4B ; WRITE A*256 'LONG 1'
FCCB: 20 DB FC 304      JSR     ZERDLY ; HALF CYCLES
FCCD: D0 F9    FCC9 305      BNE     HEADR ; (650 USEC EACH)
FCD0: 69 FE    306      ADC     #*FE
FCD2: B0 F5    FCC9 307      BCS     HEADR ; THEN A 'SHORT 0'
FCD4: A0 21    308      LDY     #*21 ; (400 USEC)
FCD6: 20 DB FC 309      WRBIT  JSR     ZERDLY ; WRITE TWO HALF CYCLES
FCD9: CB       310      INY     ; OF 250 USEC ('0')
FCDA: C9       311      INY     ; OR 500 USEC ('1')
FCDB: 8B       312      ZERDLY DEY
FCD0: D0 FD    FCDB 313      BNE     ZERDLY
FCDE: 90 05    FCE5 314      BCC     WRTAPE ; Y IS COUNT FOR
FCE0: A0 32    315      LDY     #*32 ; TIMING LOOP
FCE2: 8B       316      ONEDLY DEY
FCE3: D0 FD    FCE2 317      BNE     ONEDLY
FCE5: AC 20 CO 318      WRTAPE LDY
FCEB: A0 2C    319      LDY     #*2C
FCEA: CA       320      DEX
FCEB: 60       321      RTS
FCE0: A2 0B    322      RBBYTE LDX     #*0B ; B BITS TO READ
FCEE: 4B       323      RBBYT2 PHA ; READ TWO TRANSITIONS
FCEF: 20 FA FC 324      JSR     RD2BIT ; (FIND EDGE)
FCF2: 6B       325      PLA
FCF3: 2A       326      ROL     A ; NEXT BIT
FCF4: A0 3A    327      LDY     #*3A ; COUNT FOR SAMPLES
FCF6: CA       328      DEX

```

```

FCF7: D0 F5 FCEE 329      BNE  RDBYT2
FCF9: 60                RTS
FCFA: 20 FD FC         331 RD2BIT JSR  RDBIT
FCFD: 88                332 RDBIT  DEY
FCFE: AD 60 CO         333      LDA  TAPEIN  ; DECR Y UNTIL
FD01: 45 2F            334      EOR  LASTIN  ; TAPE TRANSITION
FD03: 10 F8 FCFD      335      BPL  RDBIT
FD05: 45 2F            336      EOR  LASTIN
FD07: 85 2F            337      STA  LASTIN
FD09: C0 80            338      CPY  ##80    ; SET CARRY ON Y-REG.
FD0B: 60                339      RTS
FD0C: A4 24            340 RDKEY  LDY  CH
FD0E: B1 28            341      LDA  (BASL),Y ; SET SCREEN TO FLASH
FD10: 48                342      PHA
FD11: 29 3F            343      AND  ##3F
FD13: 09 40            344      ORA  ##40
FD15: 91 28            345      STA  (BASL),Y
FD17: 68                346      PLA
FD18: 6C 38 00         347      JMP  (KSWL)   ; GO TO USER KEY-IN
FD18: 0001             348      DO  APPLE2E
FD1B: FD1B             349 KEYIN  EQU  *
FD1B: A0 06            350      LDY  #6      ; RDKEY/RRAO9B1
FD1D: 4C B4 F8         351      JMP  GOTDCX  ; /RRAO9B1
FD20: EA                352      NOP        ; /RRAO9B1
FD21: FD21             353 RDESC   EQU  *
FD21: 20 0C FD         354      JSR  RDKEY  ; GET A KEY
FD24: A0 07            355      LDY  #7      ; CODE=FIXIT
FD26: 4C B4 F8         356      JMP  GOTDCX
FD29:                    357 *
FD29:                    358 * RETURN FROM GOTDCX HERE:
FD29:                    359 *
FD29: BD 06 CO         360      STA  SETSL0TCXRD  ; RESTORE BANK/RRAO9B1
FD2C: 28                361      PLP        ; RESTORE IRG/RRAO9B1
FD2D: 60                362      RTS      ; RETURN TO CALLER/RRAO9B1
FD2E:                    363      ELSE    ; /RRAO9B1
S                    364 KEYIN  INC  RNDL   ; INCR RND NUMBER
S                    365      BNE  KEYIN2
S                    366      INC  RNDH
S                    367 KEYIN2 BIT  KBD     ; KEY DOWN?
S                    368      BPL  KEYIN  ; LOOP UNTIL WE GET ONE
S                    369      STA  (BASL),Y ; REPLACE FLASHING SCREEN
S                    370      LDA  KBD     ; GET KEYCODE
S                    371      BIT  KBDSTRB ; CLR KEY STROBE
FD2E:                    372      FIN      ; /RRAO9B1
FD2E: 60                373      RTS
FD2F:                    374      DO  APPLE2E ; /RRAO9B1
FD2F: 20 21 FD         375 ESC    JSR  RDESC ; /RRAO9B1
FD32:                    376      ELSE    ; /RRAO9B1
S                    377 ESC    JSR  RDKEY  ; GET KEYCODE
FD32:                    378      FIN      ; /RRAO9B1
FD32: 20 A5 FB         379      JSR  ESCNEW  ; HANDLE ESC FUNCTION.
FD35: 20 0C FD         380 RDCHAR JSR  RDKEY  ; GO READ KEY
FD38: C9 9B            381      CMP  ##9B   ; 'ESC'?
FD3A: F0 F3 FD2F      382      BEQ  ESC    ; YES, DON'T RETURN.
FD3C: 60                383      RTS
FD3D: A9 32            384 NOTCR  LDA  INVFLG
FD3F: 48                385      PHA
FD40: A9 FF            386      LDA  ##FF
FD42:                    387      DO  APPLE2E ; /RRAO9B1
FD42: EA                388      NOP        ; DON'T CHANGE INPUT/RRAO9B1
FD43: EA                389      NOP        ; TO NORMAL/RRAO9B1
FD44:                    390      ELSE    ; /RRAO9B1
S                    391      STA  INVFLG ; CONVERT TYPED CHAR TO 'NORMAL'
FD44:                    392      FIN      ; /RRAO9B1
FD44: BD 00 02         393      LDA  IN, X
FD47: 20 ED FD         394      JSR  COUT   ; ECHO TYPED CHAR
FD4A: 68                395      PLA
FD4B: 85 32            396      STA  INVFLG
FD4D: BD 00 02         397      LDA  IN, X
FD50: C9 8B            398      CMP  ##8B   ; CHECK FOR EDIT KEYS
FD52: F0 1D FD71      399      BEQ  BCKSPC ; - BACKSPACE
FD54: C9 9B            400      CMP  ##9B   ; - CONTROL-X
FD56: F0 0A FD62      401      BEQ  CANCEL ; - CONTROL-X
FD58: E0 FB            402      CPX  ##FB
FD5A: 90 03 FD5F      403      BCC  NOTCR1 ; MARGIN?
FD5C: 20 3A FF         404      JSR  BELL   ; YES, SOUND BELL
FD5F: EB                405      INX      ; ADVANCE INPUT INDEX
FD60: D0 13 FD75      406      BNE  NXTCHAR ; BACKSLASH AFTER CANCELLED LINE
FD62: A9 DC            407 CANCEL  LDA  COUT
FD64: 20 ED FD         408      JSR  COUT
FD67: 20 BE FD         409 GETLNZ  JSR  CROUT  ; OUTPUT 'CR'
FD6A: A5 33            410 GETLN  LDA  PROMPT ; OUTPUT PROMPT CHAR
FD6C: 20 ED FD         411      JSR  COUT
FD6F: A2 01            412      LDY  ##01   ; INIT INPUT INDEX
FD71: 8A                413 BCKSPC  TXA
FD72: F0 F3 FD67      414      BEQ  GETLNZ  ; WILL BACKSPACE TO 0
FD74: CA                415      DEX
FD75: 20 35 FD         416 NXTCHAR JSR  RDCHAR
FD78: C9 95            417      CMP  ##95   ; USE SCREEN CHAR
FD7A: D0 02 FD7E      418      BNE  CAPTST ; FOR CONTROL-U
FD7C: B1 28            419      LDA  (BASL),Y
FD7E: C9 E0            420 CAPTST  CMP  ##E0   ; LOWER CASE?

```

```

FB80: 90 02   FDB4 421      BCC   ADDINP
FB82:         0001 422      DO    APPLE2E
FB82: 29 FF   423      AND   *$FF
FB84:         424      ELSE
S         425      AND   *$DF
FB84:         426      FIN
FB84: 9D 00 02   427 ADDINP      STA   IN, X
FB87: C9 8D         428      CMP   *$BD
FB89: D0 B2   FDB3 429      BNE   NOTCR
FB8B: 20 9C FC   430      JSR   CLREOL
FB8E: A9 8D         431 CROUT      LDA   *$BD
FD70: D0 5B   FDED 432      BNE   COUT
FD72: A4 3D         433 PRA1      LDY   A1H
FD74: A6 3C         434          LDX   A1L
FD76: 20 8E FD   435 PRYX2     JSR   CROUT
FD79: 20 40 F9   436          JSR   PRNTYX
FD9C: A0 00         437          LDY   *$00
FD9E: A9 AD         438          LDA   *$AD
FDA0: 4C ED FD   439          JMP   COUT
FDA3: A5 3C         440 XAMB      LDA   A1L
FDA5: 09 07         441          DRA   *$07
FDA7: 85 3E         442          STA   A2L
FDA9: A5 3D         443          LDA   A1H
FDAB: 85 3F         444          STA   A2H
FDAD: A5 3C         445 MODBCHK  LDA   A1L
FDAF: 29 07         446          AND   *$07
FB11: D0 03   FDB6 447      BNE   DATADUT
FB13: 20 92 FD   448 XAM      JSR   PRA1
FB16: A9 A0         449 DATADUT  LDA   *$A0
FB18: 20 ED FD   450          JSR   COUT
FB1B: B1 3C         451          LDA   (A1L), Y
FB1D: 20 DA FD   452          JSR   PRBYTE
FD00: 20 BA FC   453          JSR   NXTA1
FD03: 90 EB   FDAD 454      BCC   MODBCHK
FDC5: 60         455 RTS4C     RTS
FDC6: 4A         456 XAMPM   LSR   A
FDC7: 90 EA   FDB3 457      BCC   XAM
FDC9: 4A         458          LSR   A
FDCA: 4A         459          LSR   A
FDCB: A5 3E         460          LDA   A2L
FDCD: 90 02   FDD1 461      BCC   ADD
FDCF: 49 FF         462          EOR   *$FF
FDD1: 65 3C         463 ADD
FDD3: 4B         464          PHA
FDD4: A9 BD         465          LDA   *$BD
FDD6: 20 ED FD   466          JSR   COUT
FDD9: 6B         467          PLA
FDDA: 4B         468 PRBYTE   PHA
FDDB: 4A         469          LSR   A
FDDC: 4A         470          LSR   A
FDDD: 4A         471          LSR   A
FDDE: 4A         472          LSR   A
FDDF: 20 E5 FD   473          JSR   PRHEXZ
FDE2: 6B         474          PLA
FDE3: 29 0F         475 PRHEX   AND   *$0F
FDE5: 09 B0         476 PRHEXZ  DRA   *$B0
FDE7: C9 BA         477          CMP   *$BA
FDE9: 90 02   FDED 478      BCC   COUT
FDEB: 69 06         479          ADC   *$06
FDED: 6C 36 00   480 COUT      JMP   (CSWL)
FDFO: C9 A0         481 COUT1    CMP   *$A0
FDF2: 90 02   FDF6 482      BCC   COUTZ
FDF4: 25 32         483          AND   INVFLG
FDF6: B4 35         484 COUTZ    STY   YSAV1
FDF8: 4B         485          PHA
FDF9: 20 7B FB   486          JSR   VIDWAIT
FDFC: 6B         487          PLA
FDFD: A4 35         488          LDY   YSAV1
FDFE: 60         489          RTS
FE00: C6 34         490 BL1     DEC   YSAV
FE02: F0 9F   FDA3 491      BEQ   XAMB
FE04: CA         492 BLANK    DEX
FE05: D0 16   FE1D 493      BNE   SETMDZ
FE07: C9 BA         494          CMP   *$BA
FE09: D0 BB   FDC6 495      BNE   XAMPM
FE0B: B5 31         496 STOR     STA   MDDE
FE0D: A5 3E         497          LDA   A2L
FE0F: 91 40         498          STA   (A3L), Y
FE11: E6 40         499          INC   A3L
FE13: D0 02   FE17 500      BNE   RTS5
FE15: E6 41         501          INC   A3H
FE17: 60         502 RTS5    RTS
FE18: A4 34         503 SETMDZE  LDY   YSAV
FE1A: B9 FF 01   504          LDA   IN-1, Y
FE1D: B5 31         505 SETMDZ  STA   MDDE
FE1F: 60         506          RTS
FE20: A2 01         507 LT      LDY   LT
FE22: B5 3E         508 LT2     LDA   A2L, X
FE24: 95 42         509          STA   A4L, X
FE26: 95 44         510          STA   A5L, X
FE28: CA         511          DEX
FE29: 10 F7   FE22 512      BPL   LT2

```

```

FE2B: 60          513          RTS
FE2C: B1 3C      514 MOVE      LDA (A1L),Y ; MOVE (A1) THRU (A2) TO (A4)
FE2E: 91 42      515          STA (A4L),Y
FE30: 20 B4 FC   516          JSR NXTA4
FE33: 90 F7 FE2C 517          BCC MOVE
FE35: 60          518          RTS
FE36: B1 3C      519 VFY      LDA (A1L),Y ; VERIFY (A1) THRU (A2)
FE3B: D1 42      520          CMP (A4L),Y ; WITH (A4)
FE3A: F0 1C FE9B 521          BEQ VFYOK
FE3C: 20 92 FD   522          JSR PR#1
FE3F: B1 3C      523          LDA (A1L),Y
FE41: 20 DA FD   524          JSR PRBYTE
FE44: A9 A0      525          LDA #A0
FE46: 20 ED FD   526          JSR COUT
FE49: A9 AB      527          LDA #A8
FE4B: 20 ED FD   528          JSR COUT
FE4E: B1 42      529          LDA (A4L),Y
FE50: 20 DA FD   530          JSR PRBYTE
FE53: A9 A9      531          LDA #A9
FE55: 20 ED FD   532          JSR COUT
FE5B: 20 B4 FC   533 VFYOK   JSR NXTA4
FE5B: 90 D9 FE36 534          BCC VFY
FE5D: 60          535          RTS
FE5E: 20 75 FE   536 LIST     JSR A1PC ; MOVE A1 (2 BYTES) TO
FE61: A9 14      537          LDA #B14 ; PC IF SPEC'D AND
FE63: 48          538 LIST2    PHA ; DISASSEMBLE 20 INSTRUCTIONS.
FE64: 20 D0 F8   539          JSR INSTDSP
FE67: 20 53 F9   540          JSR PCADJ ; ADJUST PC AFTER EACH INSTRUCTION.
FE6A: 85 3A      541          STA PCL
FE6C: 84 3B      542          STY PCH
FE6E: 68          543          PLA
FE6F: 38          544          SEC
FE70: E9 01      545          SBC #B01 ; NEXT OF 20 INSTRUCTIONS
FE72: D0 EF FE63 546          BNE LIST2
FE74: 60          547          RTS
FE75: 8A          548 A1PC     TXA ; IF USER SPECIFIED AN ADDRESS.
FE76: F0 07 FE7F 549          BEQ A1PCRT5 ; COPY IT FROM A1 TO PC.
FE7B: B5 3C      550 A1PCPLP LDA A1L,X ; YEP, SO COPY IT.
FE7A: 95 3A      551          STA PCL,X
FE7C: CA          552          DEX
FE7D: 10 F9 FE7B 553          BPL A1PCPLP
FE7F: 60          554 A1PCRT5 RTS
FE80: A0 3F      555 SETINV   LDY #B3F ; SET FOR INVERSE VID
FE82: D0 02 FE86 556          BNE SETIFLG ; VIA COUT1
FE84: A0 FF      557 SETNORM  LDY #BFF ; SET FOR NORMAL VID
FE86: 84 32      558 SETIFLG STY INVFLG
FE8B: 60          559          RTS
FE89: A9 00      560 SETKBD   LDA #B00 ; DO 'IN#0'
FE8B: 85 3E      561 INPRT   STA A2L ; DO 'IN#AREG'
FE8D: A2 38      562 INPRT   LDX #KSWL
FE8F: A0 1B      563          LDY #KEYIN
FE91: D0 0B FE9B 564          BNE IOPRT
FE93: A9 00      565 SETVID   LDA #B00 ; DO 'PR#0'
FE95: 85 3E      566 OUTPRT  STA A2L ; DO 'PR#AREG'
FE97: A2 36      567 OUTPRT  LDX #CSWL
FE99: A0 F0      568          LDY #COUT1
FE9B: A5 3E      569 IOPRT   LDA A2L ; SET INPUT/OUTPUT VECTORS
FE9D: 29 0F      570          AND #B0F
FE9F: F0 06 FEA7 571          BEQ IOPRT1
FEA1: 09 C0      572          ORA #C1OADR
FEA3: A0 00      573          LDY #B00
FEA5: F0 02 FEA9 574          BEQ IOPRT2
FEA7: A9 FD      575 IOPRT1  LDA #C0OUT1
FEA9: FE A9      576 IOPRT2  EQU *
FEA?: 94 00      577          STY LOC0,X
FEAB: 95 01      578          STA LOC1,X
FEAD: 60          579          RTS
FEAE: EA          580          NOP
FEAF: 00          581 CKSUMFIX DFB 0 ; /RRA09B1
FEB0:          582 * ;--CORRECT CKSUM AT CREATE TIME.
FEB0: 4C 00 E0   583 XBASIC  JMP BASIC ; TO BASIC, COLD START
FEB3: 4C 03 E0   584 BASCONT JMP BASIC2 ; TO BASIC, WARM START
FEB6: 20 75 FE   585 GO      JSR A1PC ; ADDR TO PC IF SPECIFIED
FEB9: 20 3F FF   586          JSR RESTORE ; RESTORE FAKE REGISTERS
FEBB: 6C 3A 00   587          JMP (PCL) ; AND GO!
FEBF: 4C D7 FA   588 REGZ    JMP REGDSP ; GO DISPLAY REGISTERS
FEC2: 60          589 TRACE   RTS ; TRACE IS GONE
FEC3: EA          590          NOP
FEC4: 60          591 STEPZ   RTS ; STEP IS GONE
FEC5: C2 F2 F9 E1 592          ASC 'Bryan'
FECA: 4C F8 03   593 USR     JMP USRADR ; JUMP TO CONTROL-Y VECTOR IN RAM
FECD: A9 40      594 WRITE   LDA #B40 ; TAPE WRITE ROUTINE
FECF: 20 C9 FC   595          JSR HEADR ; WRITE 10-SEC HEADER
FED2: A0 27      596          LDY #B27
FED4: A2 00      597 WR1     LDX #B00
FED6: 41 3C      598          EOR (A1L,X)
FEDB: 4B          599          PHA
FED9: A1 3C      600          LDA (A1L,X)
FEDB: 20 ED FE   601          JSR WRBYTE
FEDE: 20 BA FC   602          JSR NXTA1
FEE1: A0 1D      603          LDY #B1D
FEE3: 68          604          PLA

```

```

FEE4: 90 EE FED4 605 BCC WR1
FEE6: A0 22 606 LDY ##22
FEE8: 20 ED FE 607 JSR WRBYTE
FEED: F0 4D FF3A 608 BEG BELL
FEED: A2 10 609 WRBYTE LDX ##10
FEED: 0A 610 WRBYTE2 ASL A
FEF0: 20 D6 FC 611 JSR WRBIT
FEF3: D0 FA FEEF 612 BNE WRBYT2
FEF5: 60 613 RTS
FEF6: 20 00 FE 614 CRMON JSR BL1 ; HANDLE CR AS BLANK
FEF9: 68 615 PLA ; THEN POP STACK
FEFA: 68 616 PLA ; AND RETURN TO MON
FEFB: D0 6C FF69 617 BNE MONZ ; (ALWAYS)
FEFD: 20 FA FC 618 READ JSR RD2BIT ; TAPE READ - FIND TAPE IN EDGE
FF00: A9 16 619 LDA ##16 ; DELAY 3.5 SECONDS
FF02: 20 C9 FC 620 JSR HEADR
FF03: 85 2E 621 STA CHKSUM ; INITIAL CHECKSUM = $FF
FF07: 20 FA FC 622 JSR RD2BIT ; FIND AN EDGE
FF0A: A0 24 623 RD2 LDY ##24 ; LOOK FOR SYNC BIT
FF0C: 20 FD FC 624 JSR RDBIT ; (SHORT 0)
FF0F: B0 F9 FF0A 625 BCS RD2 ; LOOP 'TIL FOUND
FF11: 20 FD FC 626 JSR RDBIT ; SKIP 2ND HALF CYCLE
FF14: A0 3B 627 LDY ##3B ; INDEX FOR 0/1 TEST
FF16: 20 EC FC 628 RD3 JSR RDBYTE ; READ A BYTE
FF19: 81 3C 629 STA (A1,X) ; PUT IT AT (A1)
FF1B: 45 2E 630 EDR CHKSUM ; UPDATE RUNNING CHECKSUM
FF1D: 85 2E 631 STA CHKSUM
FF1F: 20 BA FC 632 JSR NXTA1 ; INCR A1, COMPARE TO A2
FF22: A0 35 633 LDY ##33 ; COMPENSATE 0/1 INDEX
FF24: 90 F0 FF16 634 BCC RD3 ; REPEAT 'TIL DONE
FF26: 20 EC FC 635 JSR RDBYTE ; READ CHECKSUM BYTE
FF29: C5 2E 636 CMP CHKSUM ; DOES THE RECORDED CHKSM MATCH OURS?
FF2B: F0 0D FF3A 637 BEQ BELL ; YEP, READ OK, BEEP AND RETURN.
FF2D: A9 C5 638 PRERR LDA ##C5 ; PRINT 'ERR', THEN FALL INTO
FF2F: 20 ED FD 639 JSR COUT ; FWEEPER.
FF32: A9 D2 640 LDA ##D2
FF34: 20 ED FD 641 JSR COUT
FF37: 20 ED FD 642 JSR COUT
FF3A: A9 B7 643 BELL LDA ##B7 ; MAKE A JOYFUL NOISE, THEN RETURN.
FF3C: 4C ED FD 644 JMP COUT
FF3F: A5 48 645 RESTORE LDA STATUS ; RESTORE 6502 REGISTER CONTENTS
FF41: 48 646 PHA ; USED BY DEBUG SOFTWARE
FF42: A5 45 647 LDA A5H
FF44: A6 46 648 RESTR1 LDX XREG
FF46: A4 47 649 LDY YREG
FF4B: 28 650 PLS
FF49: 60 651 RTS
FF4A: 85 45 652 SAVE STA A5H ; SAVE 6502 REGISTER CONTENTS
FF4C: B6 46 653 SAV1 STX XREG ; FOR DEBUG SOFTWARE
FF4E: B4 47 654 STY YREG
FF50: 08 655 PHP
FF51: 68 656 PLA
FF52: 85 48 657 STA STATUS
FF54: BA 658 TSX
FF55: B6 49 659 STX SPNT
FF57: D8 660 CLD
FF5B: 60 661 RTS
FF59: 20 84 FE 662 OLDRST JSR SETNORM ; SET SCREEN MODE
FF5C: 20 2F FB 663 JSR INIT ; AND INIT KBD/SCREEN
FF5F: 20 93 FE 664 JSR SETVID ; AS I/O DEVS.
FF62: 20 89 FE 665 JSR SETKBD
FF65: D8 666 MON CLD ; MUST SET HEX MODE!
FF66: 20 3A FF 667 JSR BELL ; FWEEPER.
FF69: A9 AA 668 MONZ LDA ##AA ; '*' PROMPT FOR MONITOR
FF6B: 85 33 669 STA PROMPT
FF6D: 20 67 FD 670 JSR GETLNZ ; READ A LINE OF INPUT
FF70: 20 C7 FF 671 JSR ZMODE ; CLEAR MONITOR MODE, SCAN IDX
FF73: 20 A7 FF 672 NXTITM JSR GETNUM ; GET ITEM, NON-HEX
FF76: B4 34 673 STY YSAV ; CHAR IN A-REG.
FF7B: A0 17 674 LDY ##17 ; X-REG=0 IF NO HEX INPUT
FF7A: B8 675 DEY
FF7B: 30 EB FF65 676 BNE CHRSRCH
FF7D: D9 CC FF 677 CMP CHRTBL,Y ; COMMAND NOT FOUND, BEEP & TRY AGAIN.
FF80: D0 FB FF7A 678 BNE CHRSRCH ; FIND COMMAND CHAR IN TABLE
FF82: 20 BE FF 679 JSR TOSUB ; NOT THIS TIME
FF83: A4 34 680 LDY YSAV ; GOT IT! CALL CORRESPONDING SUBROUTINE
FF87: 4C 73 FF 681 JMP NXTITM ; PROCESS NEXT ENTRY ON HIS LINE
FF8A: A2 03 682 DIG LDX ##03
FF8C: 0A 683 ASL A
FF8D: 0A 684 ASL A ; GOT HEX DIGIT,
FF8E: 0A 685 ASL A ; SHIFT INTO A2
FF8F: 0A 686 ASL A
FF90: 0A 687 NXTBIT ASL A
FF91: 26 3E 688 ROL A2L
FF93: 26 3F 689 ROL A2H
FF95: CA 690 DEX ; LEAVE X=$FF IF DIG
FF96: 10 FB FF90 691 BPL NXTBIT
FF9B: A5 31 692 LDA MODE
FF9A: D0 06 FFA2 693 BNE NXTBS2 ; IF MODE IS ZERO,
FF9C: B5 3F 694 LDA A2H,X ; THEN COPY A2 TO A1 AND A3
FF9E: 95 3D 695 STA A1H,X
FFA0: 95 41 696 STA A3H,X

```

```

FFA2: E8          697 NXTBS2      INX
FFA3: F0 F3 FF9B 698          BEQ NXTBAS
FFA5: D0 06 FFAD 699          BNE NXTCHR
FFA7: A2 00          700 GETNUM      LDX #00          ; CLEAR A2
FFA9: 86 3E          701          STX A2L
FFAB: 86 3F          702          STX A2H
FFAD: B9 00 02      703 NXTCHR      LDA IN, Y        ; GET CHAR
FFB0: C8          704          INY
FFB1: 49 B0          705          EOR ##B0
FFB3: C9 0A          706          CMP ##0A
FFB5: 90 D3 FFBA    707          BCC DIG        ; BR IF HEX DIGIT
FFB7: 69 BB          708          ADC ##BB
FFB9: C9 FA          709          CMP ##FA
FFBB: B0 CD FFBA    710          BCS DIG
FFBD: 60          711          RTS
FFBE: A9 FE          712 TOSUB        LDA #CGO        ; DISPATCH TO SUBROUTINE, BY
FFC0: 48          713          PHA            ; PUSHING THE HI-ORDER SUBR ADDR.
FFC1: B9 E3 FF      714          LDA SUBTBL, Y  ; THEN THE LO-ORDER SUBR ADDR
FFC4: 48          715          PHA            ; ONTO THE STACK.
FFC5: A5 31          716          LDA MDDE       ; (CLEARING THE MODE, SAVE THE OLD
FFC7: A0 00          717 ZMODE      LDY #00        ; MODE IN A-REG).
FFC9: 84 31          718          STY MDDE
FFCB: 40          719          RTS
FFCC: BC          720 CHR_TBL    DFB #BC        ; AND 'RTS' TO THE SUBROUTINE!
FFCD: B2          721          DFB #B2        ; (BASIC WARM START)
FFCE: BE          722          DFB #BE        ; (USER VECTOR)
FFCF: B2          723          DFB #BE        ; (OPEN AND DISPLAY REGISTERS)
FFD0: EF          724          DFB #EF        ; (ONCE WAS TRACE; NEVER USED.)
FFD1: C4          725          DFB #C4        ; (MEMORY VERIFY)
FFD2: B2          726          DFB #B2        ; (IN#SLOT)
FFD3: A9          727          DFB #A9        ; (ONCE WAS STEP; NOW NEVER USED.)
FFD4: BB          728          DFB #BB        ; (PR#SLOT)
FFD5: A6          729          DFB #A6        ; (BASIC COLD START)
FFD6: A4          730          DFB #A4        ; (SUBTRACTION)
FFD7: 06          731          DFB #06        ; (+) (ADDITION)
FFD8: 95          732          DFB #95        ; (M) (MEMORY MOVE)
FFD9: 07          733          DFB #07        ; (C) (DELIMITER FOR MOVE, VFY)
FFDA: 02          734          DFB #02        ; (N) (SET NORMAL VIDE)
FFDB: 05          735          DFB #05        ; (I) (SET INVERSE VIDE)
FFDC: F0          736          DFB #F0        ; (L) (DISASSEMBLE 20 INSTRS)
FFDD: 00          737          DFB #00        ; (W) (WRITE TO TAPE)
FFDE: EB          738          DFB #EB        ; (G) (EXECUTE PROGRAM)
FFDF: 93          739          DFB #93        ; (R) (READ FROM TAPE)
FFE0: A7          740          DFB #A7        ; (F) (MEMORY FILL)
FFE1: C6          741          DFB #C6        ; (A) (ADDRESS DELIMITER)
FFE2: 99          742          DFB #99        ; (CR) (END OF INPUT)
FFE3: B2          743 SUBTBL      DFB #B2        ; BLANK
FFE4: C9          744          DFB #C9        ; TABLE OF LO-ORDER MONITOR ROUTINE
FFE5: BE          745          DFB #BE        ; DISPATCH ADDRESSES
FFE6: C1          746          DFB #C1
FFE7: 35          747          DFB #35
FFE8: 8C          748          DFB #8C
FFE9: C4          749          DFB #C4
FFEA: 96          750          DFB #96
FFEB: AF          751          DFB #AF
FFEC: 17          752          DFB #17
FFED: 17          753          DFB #17
FFEE: 2B          754          DFB #2B
FFEF: 1F          755          DFB #1F
FFF0: 83          756          DFB #83
FFF1: 7F          757          DFB #7F
FFF2: 5D          758          DFB #5D
FFF3: CC          759          DFB #CC
FFF4: B5          760          DFB #B5
FFF5: FC          761          DFB #FC
FFF6: 17          762          DFB #17
FFF7: 17          763          DFB #17
FFF8: F5          764          DFB #F5
FFF9: 03          765          DFB #03
FFFA: FB 03          766          DW NMI         ; NON-MASKABLE INTERRUPT VECTOR
FFFC: 62 FA          767          DW RESET      ; RESET VECTOR
FFFE: 40 FA          768          DW IRG        ; INTERRUPT REQUEST VECTOR

```


Monitor Symbol Table, Sorted by Symbol

3D A1H	3C A1L	FE75 A1PC	FE78 A1PCLP
FE7F A1PCRTS	3F A2H	3E A2L	41 A3H
40 A3L	43 A4H	42 A4L	43 A5H
44 ASL	45 ACC	FDD1 ADD	FDB4 ADDINP
FBF4 ADVANCE	?03F5 AMPERV	01 APPLE2E	FB60 APPLE11
? 2B BAS2H	? 2A BAS2L	FB01 BASCALC	FB00 BASCLC2
?FE83 BASCONT	? 29 BASH	E000 BASIC	E003 BASIC2
?2B BASL	F071 BCKSPC	FBD9 BELL1	FB44 BELL2
FF3A BELL	FE00 BL1	?FE04 BLANK	FA4C BREAK
03F0 BRKV	FC10 BS	FD62 CANCEL	FD7E CAPTST
F9B4 CHAR1	F9BA CHAR2	?E CHKSUM	FF7A CHRSRCH
? 24 CH	FFCC CHRIBL	?FEAF CKSUMFIX	?C059 CLRANO
?C05B CLRAN1	?C05D CLRAN2	?C05F CLRAN3	FC9C CLRREDL
?FC9E CLRREDZ	?FC42 CLREDP	CFFF CLRRDM	FB38 CLRSC2
FB3C CLRSC3	?FB32 CLRSCR	FB36 CLRTOP	30 COLGR
FDED COUT	FDFO COUT1	FD66 COUTZ	FD8E CROUT
FC62 CR	?FEF6 CRMDN	? 37 CSMH	36 C5WL
? 25 CV	FB06 DATAOUT	FFBA DIO	FB02 DISK1D
FB45 ERR	FD2F ESC	FC2C ESC1	FB45 ESCNEW
FB9B ESCNOW	FB97 ESCOLD	FA9B FIXSEV	F962 FMT1
F9A6 FMT2	?E FORMAT	FB47 GBASCALC	27 GBASH
? 26 GBASL	FB56 GBCALC	FB49 GETFMT	?FD6A GETLN
FD67 GETLNZ	FFA7 GETNUM	FB84 GOTDCX	FB66 GO
? 2C H2	FC09 HEADR	?C057 HIRES	?C055 HISCR
?FB19 HLINE	FB1C HLINE1	FC58 HOME	FB9B IEVEN
?FA6F INITAN	FB2F INIT	?FE8B INPORT	?FE8D INPRT
FB82 INSDS1	?FB8C INSDS2	FB00 INSTDSP	32 INVFLG
0200 IN	C000 IQADR	FEA7 IOPRT1	FEA9 IOPRT2
FE9B IOPRT	03FE IRQLDC	FA40 IRQ	FC99 ISPAGE1
FC91 ISSLOTS	C000 KBD	CO10 KBDSTRB	FB88 KBDWAIT
FD1B KEYIN	? 39 KSWH	38 KSWL	? 2F LASTIN
? 2F LENGTH	FC66 LF	0400 LINE1	FE63 LIST2
?FE5E LIST	? 2C LMNEM	00 LOCO	01 LOC1
C056 LORES	C054 LOWSCR	?FE20 LT	FE22 LT2
? 2E MASK	?C052 MIXCLR	C053 MIXSET	F9C0 MNMML
FA00 MNEMR	FB8E MNNDX1	FB02 MNNDX2	FB09 MNNDX3
FDAD MDOBCHK	? 31 MDDE	FB65 MDN	FF69 MDNZ
FE2C MOVE	07FB MSLDT	?FA81 NEMMDN	C03B NM1
FAA3 NOFIX	FD5F NDCR1	FD3D NDCR	FB94 NOWAIT
FC8A NXTA1	FC84 NXTA4	FF98 NXTBAS	FF90 NXTBIT
FFA2 NXTBS2	FAC7 NXTBYT	FD75 NXTCHAR	FFAD NXTCHR
?FB5F NXTCOL	FF73 NXTILTM	FA59 OLDBRK	?FF59 OLDRST
FCE2 ONEDLY	?FE95 OUTPORT	?FE97 OUTPRT	C064 PADDL0
?F934 PCADJ2	F956 PCADJ3	F953 PCADJ	F95C PCADJ4
? 3B PCH	? 3A PCL	? 95 PICK	FB0E PLOT1
FB00 PLOT	FD92 PRA1	F910 PRADR1	F914 PRADR2
F926 PRADR3	F92A PRADR4	F930 PRADR5	F94A PRBL2
?F94C PRBL3	F948 PRBLNK	FD4A PRBYTE	FB23 PREAD2
?FB1E PREAD	?FE2D PRERR	FDE5 PRHEXZ	?FD03 PRHEX
FBF5 PRMN1	FBF9 PRMN2	?F941 PRNTAX	FBDB PRNTBL
FB44 PRNTOF	?F944 PRNTAX	F940 PRNTYX	33 PRDMP1
FD96 PRYX2	C070 PTRIG	FAFD PWRCON	03F4 PWREDUP
FAA6 PWRUP	FCFA RD2BIT	FF0A RD2	FF16 RD3
CO1B RDBOSTORE	FCFD RDBIT	FCEE RDBY2	FCEC RDBYTE
FD35 RDCHAR	CO15 RDCXRDM	FC84 RDCX	FD21 RDESC
FD0C RDKEY	CO1C RDPAGE2	FAE4 RDSP1	?FEFD READ
FAD7 REGDSP	?FEFB REGZ	?F93B RELADR	FA62 RESET
FF3F RESTORE	?FF44 RESTR1	FADA RGDSP1	2D RMNEM
? 4F RNDH	? 4E RNDL	FB7 RTBL	FB0C RTMASK
FB7F RTMSKZ	FB01 RTS1	F961 RTS2	FBEF RTS2B
FB2E RTS2D	FBFC RTS3	FC0B RTS4B	?FDC5 RTS4C
FC2B RTS4	FE17 RTS5	FF4C SAV1	?FF4A SAVE
?FB71 SCRN	FB79 SCRND	?FC70 SCROLL	C058 SETANO
C05A SETAN1	?C05C SETAN2	?C05E SETAN3	?FB64 SETCOL
?FB40 SETOR	FB86 SETIFLG	C007 SETINTCXROM	?FE80 SETINV
FB89 SETKBD	FE1D SETMDZ	?FE18 SETMODE	FB84 SETNORM
?FAA9 SETPG3	FAAB SETPLP	?FB6F SETPWRC	C006 SETSLOTXROM
?FB39 SETTXT	FE93 SETVID	FB48 SETWND	? 2F SIGN
FABA SLODP	03F2 SOFTEV	CO30 SPKR	49 SPNT
4B STATUS	?FECA STEPZ	FB5B STITLE	?FE8B STOR
FB99 STORADV	FEF3 SUBTBL	?FB5B TABV	CO40 TAPEIN
CO20 TAPEOUT	FB09 TITLE	FFBE TDSUB	?FECC TRACE
CO50 TXTCLR	CO51 TXTSET	FC1A UP	?FECA USR
03FB USRADR	? 2D V2	?FB83 VERSION	FE5B VFYOK
FE36 VFY	FBFD VIDOUT	FB7B VIDWAIT	FB2B VLINL
FB26 VLINLZ	FC22 VTAB	FC24 VTABZ	FCAB WAIT
FC9A WAIT2	FCAA WAIT3	23 WNDBTM	20 WNDLFT

```
22 WNDTOP          21 WNDWIDTH      FED4 WR1          FCD6 WRBIT
FEFF WRBYT2      FEED WRBYTE      ?FECB WRITE      FCE5 WRTAPE
FDB3 XAM          FDA3 XAMB        FDC6 XAMPM       ?FEB0 XBASIC
FC72 XGOTDCX     FB11 XLTBL       46 XREG          47 YREG
34 YSAV          35 YSAV1        FCDB ZERDLY      FFC7 ZMODE
** SUCCESSFUL ASSEMBLY :- NO ERRORS
** ASSEMBLER CREATED ON 05-JAN-82 000004
** TOTAL LINES ASSEMBLED 1435
** FREE SPACE PAGE COUNT 67
2 BJS.SRC2
```

Monitor Symbol Table, Sorted by Address

00 LOCO	01 APPLE2E	01 LOC1	20 WNDLFT
21 WNDWDTH	22 WNDTOP	23 WNDBTM	24 CH
25 CV	26 GBASL	27 GBASH	28 BASL
29 BASH	? 2A BAS2L	? 2B BAS2H	2C HZ
2C LHMEN	2D V2	2D RNMEN	2E MASK
2E CHKSUM	2E FORMAT	2F LASTIN	2F LENGTH
? 2F SIGN	30 COLOR	31 MODE	32 INVFLG
33 PROMPT	34 YSAV	35 YSAV1	36 CSWL
? 37 CSWH	38 KSWL	? 39 KSWH	3A PCL
3B PCH	3C A1L	3D A1H	3E A2L
3F A2H	40 A3L	41 A3H	42 A4L
43 A4H	44 A5L	45 A5H	45 ACC
46 XREG	47 YREG	48 STATUS	49 SPNT
? 4E RNDL	? 4F RNDH	? 55 PICK	0200 IN
03F0 BRKV	03F2 SOFTEV	03F4 PUREDUP	?03F5 AMPERV
03FB USRADR	03FB NM1	03FE IRQLDC	0400 LINE1
07FB MSL0T	C000 IOADR	C000 KBD	C006 SETSL0TCXROM
C007 SETINTCXROM	C010 KBDSTRB	C015 RDCXROM	C018 RDBOSTORE
C01C RDPAGE2	C020 TAPEOUT	C030 SPKR	C050 TXTCCLR
C051 TXTSET	?C052 MIXCLR	C053 MIXSET	C054 LOWSCR
?C055 HISC	C056 LORES	?C057 HIRES	C058 SETANO
?C059 CLRANO	?C05A SETAN1	?C05B CLRAN1	?C05C SETAN2
?C05D CLRAN2	?C05E SETAN3	?C05F CLRAN3	C060 TAPEIN
C064 PADDL0	C070 PTRIG	CFFF CLRROM	E000 BASIC
E003 BASIC2	FB00 PLOT	FB0C RTMASK	FB0E PLOT1
?FB19 HLINE	FB1C HLINE1	FB26 VLINEZ	FB28 VLINE
FB31 RTS1	?FB32 CLRSCR	FB36 CLRTOP	FB38 CLRSC2
FB3C CLRSC3	FB47 GBASCALC	FB56 GBALC	?FB5F NXTCDL
?FB64 SETCDL	?FB71 SCRNI	FB79 SCRNI	FB7F RTMSKZ
FB82 INSDS1	?FB8C INSDS2	FB9B IEVEN	FB8A5 ERR
FB8F GETFMT	FB8E MNNDX1	FBCC MNNDX2	FB8C9 MNNDX3
FBDD INSTDSP	FBDA PRNTP	FBDB PRNTBL	FBF5 PRMN1
FBF9 PRMN2	F910 PRADR1	F914 PRADR2	F926 PRADR3
F92A PRADR4	F930 PRADR5	?F938 RELADR	F940 PRNTYX
?F941 PRNTAX	?F944 PRNTX	F948 PRBLNK	F94A PRBL2
?F94C PRBL3	F953 PCADJ	?F954 PCADJ2	F956 PCADJ3
F95C PCADJ4	F961 RTS2	F962 FMT1	F966 FMT2
F9B4 CHAR1	F9BA CHAR2	F9C0 MNEM1	FA00 MNEMR
FA40 IRG	FA4C BREAK	FA59 DLDBRK	FA62 RESET
?FA6F INITAN	?FAB1 NEWMON	FA9B FIXSEV	FAA3 NDFIX
FAA6 PWRUP	?FAA9 SETPG3	FAAB SETPLP	FABA SLOOP
FAC7 NXTBYT	FAD7 RECDSP	FADA RGDSP1	FAE4 RDSP1
FADF PWRCON	FB02 DISKID	FB09 TITLE	FB11 XLTBL
FB19 RTBL	?FB1E PREAD	FB25 PREAD2	FB2E RTS2D
FB2F INIT	?FB39 SETTXT	?FB40 SETOR	FB48 SETWBD
?FB5B TABV	FB60 APPLEII	FB65 STITLE	?FB6F SETPWR
FB7B VIDWAIT	FB8B KBDWAIT	FB94 NOWAIT	FB97 ESCOLD
FB9B ESCNOW	FBAA ESCNEW	?FBB3 VERSION	FB84 GOTOCX
FB01 BASCALC	FBDD BASCLC2	FBDD BELL1	FBE4 BELL2
FB0F RTS2B	FBF0 STORADV	FBF4 ADVANCE	FBFC RTS3
FBFD VIDOUT	FC10 BS	FC1A UP	FC22 VTAB
FC24 VTABZ	FC2B RTS4	FC2C ESC1	?FC42 CLREOP
FC5B HOME	FC62 CR	FC66 LF	?FC70 SCROLL
FC72 XGBTDCX	FC84 RDCX	C91 ISSLOTS	FC79 ISPAGE1
FC9C CLREOL	?FC9E CLREOLZ	FCAB WAIT	FC49 WAIT2
FCAA WAIT3	FC84 NXTA4	FC8A NXTA1	FC0B RTS4B
FC9C HEADR	FCDA WRBIT	FCDB ZERDLY	FC2E DNEPLY
FC5E WRITAF	FC0C RDBYTE	FCE0 RDBYT2	FCFA RD2BIT
FCFD RDBIT	FD0C RDKEY	FD1B KEYIN	FD21 RDESC
FD2F ESC	FD35 RDCHAR	FD3D NOTCR	FD5F NOTCR1
FD62 CANCEL	FD67 GETLNZ	?FD6A GETLN	FD71 BCKSPC
FD75 NXTCHAR	FD7E CAPTST	FD84 ADDINP	FD8E CROUT
FD92 PRA1	FD96 PRYX2	FDAA XAMB	FDAD MODBCHK
FD83 XAM	FD86 DATAOUT	?FD0C RTS4C	FD06 XAMPM
FD81 ADD	FD8A PRBYTE	?FD0E PRHEX	FD05 PRHEXZ
FD8D COUT	FD90 COUT1	FD04 COUTZ	FD0B BL1
?FE04 BLANK	?FE0B STOR	FE17 RTS5	?FE1B SETHODE
FE1D SETMDZ	?FE20 LT	FE22 LT2	FE2C HODE
FE36 VFY	FE5B VFYOK	?FE5E LIST	FE63 LIST2
FE75 A1PC	FE7B A1PCPL	FE7F A1PCRTS	?FE80 SETINV
FE84 SETNORM	FE86 SETIFLG	FE89 SETKBD	?FE8B INPORT
?FE8D INPRT	FE93 SETVID	?FE95 OUTPORT	?FE97 OUTPRT
FE9B IOPRT	FEA7 IOPRT1	FEA9 IOPRT2	?FEAF CKSUMFIX
?FE80 XBASIC	?FE83 BASCON	FE86 GO	?FE8F REGZ
?FE82 TRACE	?FE84 STEPZ	?FE8A USR	?FE8D WRITE
FED4 WR1	FEED WRBYTE	FE0F WRBYT2	?FE06 CRMON

```
?FEFD READ          FFOA RD2          FF16 RD3          ?FF2D PRERR
FF3A BELL           FF3F RESTORE      ?FF44 RESTR1     ?FF4A SAVE
FF4C SAV1           ?FF59 OLDRST     FF65 MON         FF69 MONZ
FF73 NXTITM        FF7A CHRSRCH     FF8A DIG         FF90 NXTBIT
FF9B NXTBAS        FFA2 NXTBS2     FFA7 GETNUM     FFAD NXTCHR
FFBE TOSUB         FFC7 ZMODE       FFCC CHRTBL     FFE3 SUBTBL

** SUCCESSFUL ASSEMBLY := NO ERRORS
** ASSEMBLER CREATED ON 05-JAN-82 000004
** TOTAL LINES ASSEMBLED 1438
** FREE SPACE PAGE COUNT 67
  2 BJS.SRC2
```

80-Column Firmware Listing

```

0000:          2 *****
0000:          3 *
0000:          4 * Apple //e VIDEO FIRMWARE
0000:          5 *
0000:          6 * RICK AURICCHIO 08/81
0000:          7 *
0000:          8 * (C) 1981, APPLE COMPUTER INC.
0000:          9 * ALL RIGHTS RESERVED
0000:         10 *
0000:         11 *****
0000:         12 *
0000:    0006   13 GOODFS      EQU    6          ;FB ROM VERSION
0000:         14 *
0000:         15 * HARDWARE EQUATES:
0000:         16 *
0000:    0000   17 KBD        EQU    $C000      ;KEYBOARD PORT
0000:    0000   18 CLRBOCOL   EQU    $C000      ;DISABLE BOCOL STORE
0000:    0000   19 SETBOCOL   EQU    $C001      ;ENABLE BOCOL STORE
0000:    0000   20 RDMAINRAM  EQU    $C002      ;READ MAINBOARD RAM
0000:    0000   21 RDCARDRAM  EQU    $C003      ;READ CARD RAM
0000:    0000   22 WRMAINRAM  EQU    $C004      ;WRITE MAINBOARD RAM
0000:    0000   23 WRCARDRAM  EQU    $C005      ;WRITE CARD RAM
0000:    0000   24 SETINTCXROM EQU    $C007      ;SET INTERNAL CX00 ROM
0000:    0000   25 SETSTDZP   EQU    $C008      ;SET STD ZP/STK
0000:    0000   26 SETALTZP   EQU    $C009      ;SET ALT ZP/STK
0000:    0000   27 SETSLDTC3ROM EQU    $C00B      ;
0000:    0000   28 CLRBOVID   EQU    $C00C      ;DISABLE BOCOL VIDEO
0000:    0000   29 SETBOVID   EQU    $C00D      ;ENABLE BOCOL VIDEO
0000:    0000   30 CLRALTCHAR EQU    $C00E      ;NORM LC, FLASH UC
0000:    0000   31 SETALTCHAR EQU    $C00F      ;NORM/INV LC, NO FLASH
0000:    0000   32 KBDSTRB   EQU    $C010      ;CLEAR STROBE
0000:    0000   33 RDLCBNK2   EQU    $C011      ;READS LC BANK2
0000:    0000   34 RDLGRAM   EQU    $C012      ;READS LC RAM ENABLE
0000:    0000   35 RDRAMRD   EQU    $C013      ;READS RAMREAD STATE
0000:    0000   36 RDRAMWRT  EQU    $C014      ;READS BANKWRT STATE
0000:    0000   37 RDBOCOL   EQU    $C018      ;READS SETBOCOL
0000:    0000   38 RDBVLBAR  EQU    $C019      ;'VBL SIGNAL
0000:    0000   39 RDTEXT    EQU    $C01A      ;READS TXT MODE
0000:    0000   40 RDPAGE2   EQU    $C01C      ;PAGE1/2 STATUS
0000:    0000   41 RDBOVID   EQU    $C01F      ;READS SETBOVID
0000:    0000   42 SPKR      EQU    $C030      ;TOGGLE SPEAKER
0000:    0000   43 TXTPAGE1   EQU    $C054      ;PAGE1 TEXT
0000:    0000   44 TXTPAGE2   EQU    $C055      ;PAGE2 TEXT
0000:         45 *
0000:         46 * MONITOR EQUATES:
0000:         47 *
0000:    0000   48 FBVERSION  EQU    $FBB3      ;FB ROM ID
0000:    0000   49 RDKEY     EQU    $FD0C      ;GET A KEYSTROKE
0000:    0000   50 SETKBD    EQU    $FE89      ;IN#0
0000:    0000   51 SETVID    EQU    $FE93      ;PR#0
0000:    0000   52 IORTS     EQU    $FF58      ;KNOWN RTS
0000:         54 * ZERPAGE EQUATES:
0000:         55 *
0000:         56
0000:    001F:   57          DSECT
001F:    001F   58 YSAV1    DS    1          ORG    $1F
0020:    0001   59 WNDLFT    DS    1          ;SAFE PLACE IN ALL ENVIRONS
0021:    0001   60 WNDWTH   DS    1          ;SCROLLING WINDOW LEFT
0022:    0001   61 WNDTOP    DS    1          ;SCROLLING WINDOW WIDTH
0023:    0001   62 WNDBTM   DS    1          ;SCROLLING WINDOW TOP
0024:    0001   63 CH       DS    1          ;CURSOR HORIZONTAL
0025:    0001   64 CV       DS    1          ;CURSOR VERTICAL
0026:    0002   65          DS    2          ;GBASL
0028:    0002   66 BASL     DS    2          ;BASE ADDRESS
002A:    0029   67 BASH     EQU    BASL+1
002A:    0002   68 BAS2L    DS    2          ;BASE ADDR FOR SCROLL
002C:    0028   69 BAS2H    EQU    BAS2L+1
0032:    0032   70          ORG    $32
0032:    0001   71 INVFLG   DS    1          ;>127=NORMAL
0033:    0003   72          DS    3          ;N/A
0036:    0002   73 CSWL     DS    2          ;COUT HOOK
0038:    0037   74 CSWH     EQU    CSWL+1
0038:    0002   75 KSWL     DS    2          ;KEYIN HOOK
003A:    0039   76 KSWH     EQU    KSWL+1
003C:    003C   77          ORG    $3C
003C:    0002   78 A1L     DS    2          ;MONITOR TEMPS FOR MOVE

```

```

003E:      003D  79 A1H      EQU  A1L+1
003E:      0002  80 A2L      DS   2
0040:      003F  81 A2H      EQU  A2L+1
0040:      0002  82        DS   2           ; A3 NOT USED
0042:      0002  83 A4L      DS   2
0044:      0043  84 A4H      EQU  A4L+1
004E:      004E  85        ORG   $4E
004E:      0002  86 RNDL     DS   2           ; RANDOM NUMBER SEED
0050:      004F  87 RNDH     EQU  RNDL+1
0000:      88        DEND
0000:      90 * PERMANENT DATA IN SCREENHOLES
0000:      91 *
0000:      92 * NOTE: THESE RESIDE IN PAGE 1 OF
0000:      93 * THE 80-COLUMN SCREEN PAIR. ANY
0000:      94 * ROUTINE WHICH SETS PAGE2 *MUST*
0000:      95 * RESTORE BACK TO PAGE1 SO THAT
0000:      96 * WE CAN CORRECTLY ACCESS THESE
0000:      97 * PERMS. UNDER *NO* CIRCUMSTANCES
0000:      98 * IS ANY ROUTINE TO BE CALLED WHILE
0000:      99 * WE HAVE PAGE2 BANKED IN!
0000:     100 *
0000:     047B 101 TEMP1     EQU  $47B           ; A TEMP
0000:     047B 102 OLDCH   EQU  $47B+3       ; OLD CH SET FOR USER
0000:     04FB 103 MODE   EQU  $4FB+3       ; OPERATING MODE
0000:     104 * MODE BITS
0000:     105 * 0..... - ESC-R INACTIVE
0000:     106 * 1..... - ESC-R ACTIVE
0000:     107 * 0..... - BASIC PRINT
0000:     108 * 1..... - BASIC INPUT
0000:     109 * 0..... - LANGUAGE=BASIC
0000:     110 * 1..... - LANGUAGE=PASCAL
0000:     111 * 0..... - U/C RESTRICT MODE
0000:     112 * 1..... - LITERAL UC/LC MODE
0000:     113 * .....0 - GOTOXY N/A
0000:     114 * .....1 - GOTOXY IN PROGRESS
0000:     115 * .....0 - NORMAL VIDEO (PASCAL)
0000:     116 * .....1 - INVERSE VIDEO (PASCAL)
0000:     117 * .....0 - PASCAL 1.1 F/W ACTIVE
0000:     118 * .....1 - PASCAL 1.0 INTERFACE
0000:     119 * .....0 - CALLER SEI'D (BASIC)
0000:     120 * .....1 - CALLER CLI'D (BASIC)
0000:     121 * .....0 - NORMAL MODE (PASCAL)
0000:     122 * .....1 - TRANSPARENT MODE (PASCAL)
0000:     00B0 123 M.ESCR   EQU  $B0           ; ESC-R ACTIVE
0000:     0040 124 M.BINPUT EQU  $40           ; BASIC INPUTTING
0000:     0020 125 M.PASCAL EQU  $20           ; PASCAL RUNNING
0000:     0010 126 M.LIT   EQU  $10           ; LITERAL UC/LC INPUT
0000:     0008 127 M.GOXY   EQU  $08           ; GOTOXY IN PROGRESS
0000:     0004 128 M.VMODE   EQU  $04           ; PASCAL VIDEO MODE
0000:     0002 129 M.PAS1.0 EQU  $02           ; PASCAL 1.0 MODE
0000:     0001 130 M.IRG   EQU  $01           ; IRG ENABLED (BASIC ONLY)
0000:     0001 131 M.TRANS  EQU  $01           ; TRANSPARENT MODE IF F/W PROTOCOL
0000:     057B 132 DURCH   EQU  $57B+3       ; 80-COL CH
0000:     05FB 133 DURCV   EQU  $5FB+3       ; CURSOR VERTICAL
0000:     067B 134 CHAR    EQU  $67B+3       ; IN/OUT CHAR
0000:     06FB 135 XCOORD  EQU  $6FB+3       ; X-COORD (GOTOXY)
0000:     077B 136 OLDBASL EQU  $77B+3       ; PASCAL SAVED BASL
0000:     07FB 137 OLDBASH  EQU  $7FB+3       ; PASCAL SAVED BASH
0000:     138 *
0000:     139 * GENERAL SCREEN STUFF:
0000:     140 *
0000:     07FB 141 CBSLDT   EQU  $7FB           ; IRG CB PROTOCOL
0000:     142        CHR  '/'
0000:     4        INCLUDE BFUNC
----- NEXT OBJECT FILE NAME IS VIDEO.OBJ0
C100:     C100  2        ORG   $C100
C100:     C100  3 BFUNCPG  EQU  *
C100:     FD29  4 FUNCEXIT EQU  $FD29           ; RETURN ADDRESS
C100:     FBC1  5 F.BASCALC EQU  $FBC1
C100:     FC22  6 F.VTAB   EQU  $FC22
C100:     FC24  7 F.VTABZ   EQU  $FC24
C100:     8 -----
C100:     9 * BASIC FUNCTION HOOK.
C100:     10 -----
C100:     11 * THIS ROUTINE IS CALLED BY THE
C100:     12 * PATCHED FB ROM.
C100:     13 * THIS CODE WILL ALWAYS PERFORM THE
C100:     14 * FUNCTION HERE AND RETURN TO THE
C100:     15 * CALLER.
C100:     16 *
C100:     17 * NOTE: FB ROM DISABLES I/O TO GET US
C100:     18 * RUNNING HERE. WE RETURN TO FB SPACE.
C100:     19 -----
C100:     20 * INPUT: Y=FUNCTION AS FOLLOWS:
C100:     21 * 0=CLREOP
C100:     22 * 1=HOME
C100:     23 * 2=SCROLL
C100:     24 * 3=CLREOL
C100:     25 * 4=CLEOLZ
C100:     26 * 5=INIT & RESET
C100:     27 * 6=KEYIN
C100:     28 * 7=FIX ESCAPE CHAR

```

```

C100:          29 *          B=SETWND
C100:          30 *
C100:          31 *          STK HAS PHP FOR STATUS
C100:          32 *          OF BANK & IRQ BIT
C100:          33 * VOLATILE: AC,Y
C100:          34 -----
C100:          35 * NOTE: IF WE HAVE A CARD INSTALLED,
C100:          36 * THEN USE THE VIDEO ROUTINES, SINCE
C100:          37 * WE 'OWN' SLOT3 SCREENHOLES.
C100:          38 * IF NOT, DUPLICATE FROM ROUTINES
C100:          39 * AND AVOID SLOT3 INTERFERENCE.
C100:          40 -----
C100:          41 * VECTOR TO KEYIN/ESCFIX IMMEDIATELY
C100:          42 * TO AVOID AC DESTRUCTION.
C100:          43 *
C100:          44 B.FUNC   EGU *
C100:CO 06      45 CPY #6          ; IS IT KEYIN?
C102:DO 03     C107 46 BNE B.FUNCNK    ; NO
C104:4C 88 C2  47 JMP B.KEYIN
C107:         C107 48 B.FUNCNK   EGU *
C107:CO 07     49 CPY #7          ; IS IT ESCAPE-FIX?
C109:DO 03     C10E 50 BNE B.FUNCNE    ; NO
C10B:4C 6E C2  51 JMP B.ESCFIX    ; =>YES!
C10E:         C10E 52 B.FUNCNE   EGU *
C10E:98        53 TYA          ; SAVE Y
C10F:4B        54 PHA
C110:20 24 CB  55 JSR TESTCARD   ; DO WE HAVE A CARD?
C113:DO 0A     C11F 56 BNE B.OLDFUNC  ; =>NO
C115:         57 *
C115:         58 * NOTE: THIS TEST COULD TURN OUT
C115:         59 * WRONG ON POWER-UP, SINCE THE
C115:         60 * MODEBYTE IS UNDEFINED. HOWEVER,
C115:         61 * SINCE THE MONITOR IS DOING A
C115:         62 * SIMPLE 'SETWND' CALL, WE WON'T
C115:         63 * GET INTO TROUBLE EVEN IF WE
C115:         64 * MAKE THE WRONG DECISION...
C115:         65 *
C115:AD FB 04   66 LDA MODE        ; IS MODE VALID?
C118:29 2B     67 AND #M.PASCAL+M.GOXY ; FOR BASIC
C11A:DO 03     C11F 68 BNE B.OLDFUNC  ; =>DEFINITELY NOT!
C11C:4C A4 C1  69 JMP B.FUNCO    ; =>YES, GO NEW WAY
C11F:         70 *
C11F:         71 * NO CARD. DO THINGS THE OLD WAY.
C11F:         72 *
C11F:         C11F 73 B.OLDFUNC  EGU *
C11F:68        74 PLA
C120:AB        75 TAY          ; RESTORE Y
C121:A9 C1     76 LDA #CBFUNCPC  ; TRANSFER VIA
C123:4B        77 PHA          ; THE RTS-TRICK
C124:B7 EA CF  78 LDA F.TABLE,Y  ; GET LD ADDRESS
C127:4B        79 PHA
C128:60        80 RTS          ; TRANSFER TO ROUTINE
C129:         81 -----
C129:A4 24     82 F.CLEOP   LDY CH          ; ESC F IS CLR TO END OF PAGE
C12B:A5 25     83 LDA CV
C12D:4B        84 CLEOP1   PHA
C12E:20 24 FC  85 JSR F.VTABZ
C131:20 F4 C2  86 JSR X.CLEOLZ
C134:A0 00     87 LDY ##00
C136:6B        88 PLA
C137:67 00     89 ADC ##00
C139:C5 23     90 CMP WNDBTM
C13B:90 F0     C12D 91 BCC CLEOP1
C13D:20 22 FC  92 JSR F.VTAB
C140:4C EB C2  93 JMP F.RETURN   ; DONE
C143:         94 -----
C143:A5 22     95 F.HOME   LDA WNDTOP
C145:85 25     96 STA CV
C147:A0 00     97 LDY ##00
C149:84 24     98 STY CH
C14B:F0 E0     C12D 99 BEQ CLEOP1   ; (ALWAYS TAKEN)
C14D:         100 -----
C14D:A5 22     101 F.SCRDLL LDA WNDTOP
C14F:4B        102 PHA
C150:20 24 FC  103 JSR F.VTABZ
C153:A5 2B     104 SCRL1   LDA BASL
C155:85 2A     105 STA BAS2L
C157:A5 27     106 LDA BASH
C159:85 2B     107 STA BAS2H
C15B:A4 21     108 LDY WNDWDTH
C15D:8B        109 DEY
C15E:6B        110 PLA
C15F:67 01     111 ADC ##01
C161:C5 23     112 CMP WNDBTM
C163:B0 0D     C172 113 BCS SCRL3
C165:4B        114 PHA
C166:20 24 FC  115 JSR F.VTABZ
C169:B1 2B     116 SCRL2   LDA (BASL),Y
C16B:91 2A     117 STA (BAS2L),Y
C16D:8B        118 DEY
C16E:10 F9     C169 119 BPL SCRL2

```

```

C170: 30 E1 C153 120 BMI SCRL1
C172: AD 00 121 LDY #800
C174: 20 F4 C2 122 JSR X.CLEDLZ
C177: 20 22 FC 123 JSR F.VTAB
C17A: 4C EB C2 124 JMP F.RETURN ;=>DONE
C17D: A4 24 125 F.CLREOL LDY CH
C17F: A9 A0 126 LDA #*AO
C181: 91 2B 127 CLEOL2 STA (BASL),Y
C183: CB 128 INY
C184: C4 21 129 CPY WNDWDTH
C186: 90 F9 C181 130 BCC CLEOL2
C188: B0 17 C1A1 131 BCS F.GORET ; DONE (ALWAYS TAKEN)
C18A: 132 -----
C18A: C18A 133 F.BETHND EGU *
C18A: A9 2B 134 LDA #40
C18C: B5 21 135 STA WNDWDTH
C18E: A9 1B 136 LDA #24
C190: B5 23 137 STA WNDBTH
C192: A9 17 138 LDA #23
C194: B5 25 139 STA CV
C196: 20 22 FC 140 JSR F.VTAB
C199: 4C EB C2 141 JMP F.RETURN
C19C: 142 -----
C19C: C19C 143 F.CLEOLZ EGU *
C19C: A4 1F 144 LDY YSAV1 ; RESTORE HORIZ POSITION
C19E: 20 F4 C2 145 JSR X.CLEOLZ ; DO IT
C1A1: 4C EB C2 146 F.GORET JMP F.RETURN ; DONE
C1A4: C1A4 148 B.FUNCO EGU *
C1A4: 6B 149 PLA ; RESTORE Y
C1A5: A8 150 TAY
C1A6: 151 *
C1A6: 152 * GET IRGMODE:
C1A6: 153 *
C1A6: AD FB 04 154 LDA MODE ; ASSUME IRG IS DISABLED
C1A9: 29 FE 155 AND #255-M.IRG
C1AB: BD FB 04 156 STA MDEE
C1AE: 6B 157 PLA ; PULL CXBANK STATUS
C1AF: BD 7B 04 158 STA TEMP1 ; OFF STACK
C1B2: 6B 159 PLA ; GET USER'S PSTATUS
C1B3: 4B 160 PHA ; (LEAVE ALONE ON STACK)
C1B4: 4A 161 LSR A ; MOVE '1' BIT TO
C1B5: 4A 162 LSR A ; THE CARRY
C1B6: 4A 163 LSR A
C1B7: AD 7B 04 164 LDA TEMP1 ; PUT CXBANK STATUS
C1BA: 4B 165 PHA ; BACK ON STACK
C1BB: B0 0B C1C5 166 BCS NOI ;=>HE'S INHIBITED
C1BD: AD FB 04 167 LDA MODE
C1C0: 09 01 168 DRA #M.IRG
C1C2: BD FB 04 169 STA MODE
C1C5: C1C5 170 NOI EGU *
C1C9: A5 25 171 LDA CV ; COPY USER CV
C1C7: BD FB 05 172 STA OURCV ; TO OURS
C1CA: 4C FF C1 173 JMP B.VECTOR ; CONTINUE
C1CD: 174 *
C1CD: 175 * NOTE: THIS KEEPS B.XXXX ROUTINES
C1CD: 176 * ALL IN THE C100 PAGE...
C1CD: 177 *
C1CD: 178 *
C1CD: C1CD 179 B.SCROLL EGU *
C1D0: 20 A4 CC 180 JSR SCROLLUP ; DO IT FOR CALLER
C1D0: 4C EB C2 181 JMP F.RETURN ; AND RETURN DIRECTLY
C1D3: 182 *
C1D3: C1D3 183 B.CLREOL EGU *
C1D3: 20 4B CD 184 JSR X.QS ; CLEAR TO EOL
C1D6: 4C EB C2 185 JMP F.RETURN ; RETURN DIRECTLY TO CALLER
C1D9: 186 -----
C1D9: C1D9 187 B.CLEOLZ EGU *
C1D9: A4 1F 188 LDY YSAV1 ; RESTORE HORIZ POSITION
C1DB: 20 4E CD 189 JSR X.QSEOLZ ; DO IT TO EOL
C1DE: 4C EB C2 190 JMP F.RETURN
C1E1: 191 -----
C1E1: C1E1 192 B.CLREOP EGU *
C1E1: 20 23 CD 193 JSR X.VT ; CLEAR TO EOB
C1E4: 4C EB C2 194 JMP F.RETURN ; RETURN DIRECTLY TO CALLER
C1E7: 195 -----
C1E7: 4C 19 C2 196 B.SETWND JMP B.SETWNDX
C1EA: 4C 34 C2 197 B.RESET JMP B.RESETX ; MUST BE IN BFUNC PAGE
C1ED: 198 -----
C1ED: C1ED 199 B.HOME EGU *
C1ED: 20 42 CD 200 JSR X.FF ; HOME & CLEAR
C1F0: AD 7B 05 201 LDA OURCH
C1F3: B5 24 202 STA CH ; COPY CH/CV FOR CALLER
C1F5: BD 7B 04 203 STA OLDCH ; REMEMBER WHAT WE SET
C1FB: AD FB 05 204 LDA OURCV
C1FB: B5 25 205 STA CV
C1FD: 10 2F C22E 206 BPL GOBACK ; (ALWAYS TAKEN)
C1FF: 207 *
C1FF: 208 * COPY USER'S CURSOR IF IT DIFFERS
C1FF: 209 * FROM OURS (AND WE'RE RUNNING
C1FF: 210 * IN BO-COLUMN MODE). IF WE ARE
C1FF: 211 * NOT IN BO-MODE, THEN ALWAYS USE

```



```

C1FF:          212 * THE USER'S CH VALUE SINCE OURS
C1FF:          213 * IS PROBABLY INVALID.
C1FF:          214 *
C1FF:          215 B. VECTOR      EQU      *
C1FF: 20 51 CB      216          JSR      BASCALC
C202: A5 24      217          LDA      CH          ; GET USER CH VALUE
C204: 2C 1F C0    218          BIT      RDBOVID     ; DISPLAYING 80-COLS?
C207: 10 05 C20E  219          BPL      B.GETCH     ; =>NO, USER CH IS IT
C209: CD 7B 04    220          CMP      OLDCH      ; IS IT DIFFERENT?
C20C: FO 03 C211  221          BEQ      B.FUNC1    ; =>NO, USE OURS
C20E:          C20E  222 B.GETCH   EQU      *
C20E: 8D 7B 05 C211  223          STA      OURCH     ; USE HIS CH
C211:          C211  224          EQU      *
C211: A9 C1      225          LDA      #CBFUNCPG  ; TRANSFER TO ROUTINE
C213: 48          226          PHA          ; VIA RTS-TRICK
C214: B9 F3 CF    227          LDA      B.TABLE.Y  ; GET LO ADDRESS
C217: 48          228          PHA
C218: 60          229          RTS
C219:          230 -----
C219:          C219  231 B.SETWDX  EQU      *
C219: A9 50      232          LDA      #80        ; ASSUME 80-COLS
C21B: 2C 1F C0    233          BIT      RDBOVID     ; WHICH MODE?
C21E: 30 01 C221  234          BMI      B.SETWDX2  ; =>IT'S 80
C220: 4A          235          LSR          ; MAKE IT 40
C221:          C221  236 B.SETWDX2 EQU      *
C221: B5 21      237          STA      WNDWIDTH
C223: A9 18      238          LDA      #24        ; SET BOTTOM
C225: B5 23      239          STA      WNDBTM
C227: A9 17      240          LDA      #23        ; VTAB TO BOTTOM
C229: 8D FB 05    241          STA      OURCV
C22C: B5 23      242          STA      CV
C22E: 20 51 CB    243          JSR      BASCALC
C231: 4C EB C2    244          JMP      F.RETURN
C234:          245 *
C234:          246 * HANDLE RESET FOR MONITOR:
C234:          247 *
C234:          C234  248 B.RESETX  EQU      *
C234: A9 FF      249          LDA      #0FF        ; DESTROY MODE BYTE
C236: 8D FB 04    250          STA      MODE
C239: AD 5D C0    251          LDA      #C05D      ; SETUP
C23C: AD 5F C0    252          LDA      #C05F      ; ANNUNCIATORS
C23F:          253 *
C23F:          254 * IF THE OPEN APPLE KEY
C23F:          255 * (ALIAS PADDLE BUTTONS 0) IS
C23F:          256 * DEPRESSED, COLDSTART THE SYSTEM
C23F:          257 * AFTER DESTROYING MEMORY:
C23F:          258 *
C23F: AD 62 C0    259          LDA      #C062      ; GET BUTTON 1 (SOLID)
C242: 30 1D C261  260          BMI      DIAGS     ; =>DOWN, DO DIAGS
C244: AD 61 C0    261          LDA      #C061      ; GET BUTTON 0 (OPEN)
C247: 10 1B C264  262          BPL      RESETRET   ; =>NOT JIVE OR DIAGS
C249:          263 *
C249:          264 * BLAST 2 BYTES OF EACH PAGE,
C249:          265 * INCLUDING THE RESET VECTOR:
C249:          266 *
C249: A0 80      267          LDY      #80        ; LET IT PRECESS DOWN
C24B: A9 00      268          LDA      #0
C24D: B5 3C      269          STA      A1L
C24F: A9 BF      270          LDA      #8BF
C251: 38          271          SEC          ; START FROM BFXX DOWN
C252:          C252  272 BLAST    EQU      *
C252: B5 3D      273          STA      A1H
C254: 91 3C      274          STA      (A1L).Y
C256: 88          275          DEY
C257: 91 3C      276          STA      (A1L).Y
C259: E9 01      277          SBC      #1        ; BACK DOWN TO NEXT PAGE
C25B: C9 01      278          CMP      #1        ; STAY AWAY FROM STACK!
C25D: D0 F3 C252  279          BNE      BLAST
C25F: FO 03 C264  280          BEQ      RESETRET   ; (ALWAYS)
C261:          281 *
C261: 4C 01 C4    282          JMP      #C401      ; RUN DIAGS
C264:          283 *
C264:          C264  284 RESETRET EQU      *
C264: 20 24 C3      285          JSR      TESTCARD   ; CARD PLUGGED IN?
C267: FO 14 C27D  286          BEQ      QORETN     ; =>YES
C269: 8D 0B C0    287          STA      SETSL0TC3ROM ; NO, DISABLE ROM
C26C: D0 0F C27D  288          BNE      QORETN     ; (ALWAYS TAKEN)
C26E:          289 -----
C26E:          C26E  291 B.ESCFIX EQU      *
C26E: 29 DF      292          AND      #0DF        ; FORCE TO UPPERCASE
C270: A0 03      293          LDY      #4-1       ; SCAN FOR A MATCH
C272:          C272  294 B.ESCFIX2 EQU      *
C272: D9 80 C2    295          CMP      ESCIN.Y    ; IS IT?
C275: D0 03 C27A  296          BNE      B.ESCFIX3  ; =>NAW
C277: B9 B4 C2    297          LDA      ESCOUT.Y   ; YES, TRANSLATE IT
C27A:          C27A  298 B.ESCFIX3 EQU      *
C27A: 8B          299          DEY
C27B: 10 F5 C272  300          BPL      B.ESCFIX2
C27D: 4C EB C2    301          JMP      F.RETURN   ; RETURN: CHAR IN AC
C280:          302 -----

```

```

C280: 88 95 BA 8B 303 ESCIN DFB *88, *95, *9A, *8B
C284: CA CB CD C9 304 ESCOUT ASC /JKMI ; THE ARROWS
C288: 305 -----
C288: C288 306 B.KEYIN EQU *
C288: 8D 78 04 307 STA TEMP1 ;SAVE ORIGINAL CHAR
C288: 68 308 PLA ;HOLD ONTO
C28C: AB 309 TAY ; CXBANK STATUS
C28D: 68 310 PLA ;GET USER'S
C28E: 48 311 PHA ; IRQ STATE
C28F: 6A 312 ROR A ; MOVE IRQ BIT TO
C290: 6A 313 ROR A ; THE
C291: 6A 314 ROR A ; CARRY
C292: 98 315 TYA ;PUT CXBANK STATUS
C293: 48 316 PHA ; BACK ON STACK
C294: BA 317 TXA ;SAVE
C295: 48 318 PHA ; XREG
C296: 319 *
C296: 88 320 CLV ;ASSUME NOT INTERRUPTIBLE
C297: B0 03 C29C 321 BCS B.KEYIN2 ;=>WE WERE RIGHT
C299: 2C 00 CF 322 BIT SEV ;SAY "INTERRUPTIBLE"
C29C: C29C 323 B.KEYIN2 EQU *
C29C: A9 FF 324 LDA *#FF ;CURSOR=NORMAL DELETE
C29E: A4 24 325 LDY CH
C2A0: 91 28 326 STA (BASL),Y
C2A2: 20 C6 C2 327 JSR KEYDLY ;WAIT FOR A KEY
C2A5: B0 0E C2B5 328 BCS GOTKEY ;=>GOT ONE
C2A7: AD 78 04 329 LDA TEMP1 ;REPLACE ORIG CHAR
C2AA: A4 24 330 LDY CH
C2AC: 71 28 331 STA (BASL),Y
C2AE: 20 C6 C2 332 JSR KEYDLY ;WAIT FOR A KEY
C2B1: B0 02 C2B5 333 BCS GOTKEY ;=>GOT ONE
C2B3: 90 E7 C29C 334 BCC B.KEYIN2 ;(ALWAYS TAKEN)
C2B5: 335 *
C2B5: C2B5 336 GOTKEY EQU *
C2B5: AD 78 04 337 LDA TEMP1 ;RESTORE ORIGINAL
C2B8: A4 24 338 LDY CH
C2BA: 91 28 339 STA (BASL),Y ; CHARACTER
C2BC: 68 340 PLA ; RESTORE
C2BD: AA 341 TAX ; XREG
C2BE: AD 00 C0 342 LDA KBD ;GET THE NEW KEYSTROKE
C2C1: BD 10 C0 343 STA KBDSTRB ;CANCEL THE STROBE
C2C4: 30 25 C2EB 344 BMI F.RETURN ;(ALWAYS TAKEN)
C2C6: 345 ***
C2C6: C2C6 346 *** INPUT: VFLAG SET IF INTERRUPTIBLE
C2C6: C2C6 347 KEYDLY EQU *
C2C6: A2 0C 348 LDX *#0C ;SHORT DELAY FOR IRQ
C2C8: 70 02 C2CC 349 BVS IK1 ;->INTERRUPTIBLE
C2CA: A2 31 350 LDX *#31 ;LONG DELAY FOR NO IRQ
C2CC: C2CC 351 IK1 EQU *
C2CC: A0 00 352 LDY #0
C2CE: C2CE 353 IK2 EQU *
C2CE: 50 05 C2D5 354 BVC IK2A ;=>NOT INTERRUPTIBLE
C2D0: 08 355 PHP ;SAVE OFLOW
C2D1: 20 75 FC 356 JSR SNIFFIRQ ;ALLOW IRQ
C2D4: 28 357 PLP ;RESTORE OFLOW
C2D5: C2D5 358 IK2A EQU *
C2D5: E6 4E 359 INC RNDL
C2D7: D0 02 C2DB 360 BNE IK3
C2D9: E6 4F 361 INC RNDH
C2DB: C2DB 362 IK3 EQU *
C2DB: AD 00 C0 363 LDA KBD ;KEYPRESS?
C2DE: 30 09 C2E9 364 BMI KDRETY ;=>YES
C2E0: 88 365 DEY
C2E1: D0 EB C2CE 366 BNE IK2
C2E3: CA 367 DEX
C2E4: D0 E6 C2CC 368 BNE IK1
C2E6: C2E6 369 KDRETN EQU *
C2E6: 18 370 CLC
C2E7: 90 01 C2EA 371 BCC KDRET
C2E9: C2E9 372 KDRETY EQU *
C2E9: 3B 373 SEC
C2EA: C2EA 374 KDRET EQU *
C2EA: 60 375 RTS
C2EB: 376 *
C2EB: 377 * EXIT. EITHER EXIT WITH OR WITHOUT
C2EB: 378 * ENABLING I/O SPACE.
C2EB: 379 *
C2EB: C2EB 380 F.RETURN EQU *
C2EB: 28 381 PLP ;GET PRIOR I/O DISABLE
C2EC: 30 03 C2F1 382 BMI F.RET1 ;=>LEAVE IT DISABLED
C2EE: 4C 29 FD 383 JMP FUNCEXIT ;=>EXIT & ENABLE I/O
C2F1: 4C 2C FD 384 F.RET1 JMP FUNCEXIT+3 ;EXIT DISABLED
C2F4: 385 -----
C2F4: C2F4 386 X.CLEDLZ EQU *
C2F4: A9 A0 387 LDA *#A0
C2F6: C2F6 388 X.CLEDL2 EQU *
C2F6: 91 28 389 STA (BASL),Y
C2F8: CB 390 INY
C2F9: C4 21 391 CPY WNDWIDTH
C2FB: 90 F9 C2F6 392 BCC X.CLEDL2
C2FD: 60 393 RTS ;ALWAYS RETURN DIRECTLY
C2FE: 0002 394 ZSPAREC2 EQU *C300-*

```

```

C2FE:          0002 395          DS  %C300-#.0
C300:          5          INCLUDE C38SPACE
C300:          6          -----
C300:          7          *
C300:          8          * THIS IS THE %C3XX ROM SPACE:
C300:          9          *
C300:          10         -----
C300:          C300 11 CNOO      EQU  *
C300:          C300 12 BASICINT EQU  *
C300: 2C 38 FF 13          BIT  IORTS  ;SET VFLAG (INIT)
C303: 70 12 C317 14          BVS  BASICENT ; (ALWAYS TAKEN)
C305:          C305 15 BASICIN  EQU  *
C305: 38          16          SEC
C306: 90          17          DFB  %90   ;BCC OPCCODE (NEVER TAKEN)
C307:          C307 18 BASICOUT EQU  *
C308: 88          19          CLC
C309: 50 0C C317 20          CLV  ;CLEAR VFLAG (NOT INIT)
C30B:          21          BVC  BASICENT ; (ALWAYS TAKEN)
C30B:          22          *
C30B:          23          * PASCAL 1.1 FIRMWARE PROTOCCOL TABLE:
C30B:          24          *
C30B: 01          25          DFB  %01   ;GENERIC SIGNATURE BYTE
C30C: 88          26          DFB  %88   ;DEVICE SIGNATURE BYTE
C30D:          27          *
C30D: 48          28          DFB  >JPINIT ;PASCAL INIT
C30E: 51          29          DFB  >JPREAD ;PASCAL READ
C30F: 57          30          DFB  >JPWRITE ;PASCAL WRITE
C310: 5D          31          DFB  >JPSTAT ;PASCAL STATUS
C311:          32          -----
C311:          33          *
C311:          34          * 128K SUPPORT ROUTINE ENTRIES:
C311:          35          *
C311: 4C 63 C3 36          JMP  MOVE   ;MEMORY MOVE ACROSS BANKS
C314: 4C 80 C3 37          JMP  XFER   ;TRANSFER ACROSS BANKS
C317:          38          -----
C317:          39          *
C317:          40          * BASIC I/O ENTRY POINT:
C317:          41          *
C317: 8D 7B 06 C317 42 BASICENT EQU  *
C31A: 48          43          STA  CHAR  ;SAVE CHARACTER
C31B: 98          44          PHA  ;SAVE AC
C31C: 48          45          TYA  ;AND Y
C31D: 8A          46          PHA  ;AND X
C31E: 48          47          TXA  ;AND X
C31F: 08          48          PHP  ;SAVE CARRY & VFLAG
C320:          49          *
C320:          50          * SET IRQMODE:
C320:          51          *
C320:  AD FB 04 52          LDA  MODE   ;ASSUME IRQ IS DISABLED
C323: 29 FE 53          AND  #255-M. IRQ
C325: 8D FB 04 54          STA  MODE
C328: 68 55          PLA  ;GET PSTATUS
C329: 48 56          PHA  ;AND LEAVE ON STACK
C32A: 29 04 57          AND  #04   ;IS 'I' BIT SET?
C32C: D0 08 C336 58          BNE  BASICENT2 ;=>YES, DISABLED
C32E: AD FB 04 59          LDA  MODE
C331: 09 01 60          ORA  #M. IRQ
C333: 8D FB 04 61          STA  MODE   ;SET IT ENABLED
C336:          C336 62 BASICENT2 EQU  *
C336:  AD FF CF 63          LDA  %CFFF ;KICK OUT ALL C8 ROMS
C339: A5 25 64          LDA  CV    ;GET USER CV AND
C33B: 8D FB 05 65          STA  DURCV  ;STUFF IT FOR US
C33E: 20 EB C3 66          JSR  SETCB   ;SETUP CB INDICATOR
C341: 28 67          PLS  ;GET VFLAG (INIT)
C342: 08 68          PHP
C343: 70 03 C348 69          BVS  JBASINIT ;=>DD THE INIT
C345: 4C 66 C8 70          JMP  CBBASIC ;GET OUT OF CN SPACE
C348: 4C 03 C8 71          JMP  JBASINIT ;=>GOTO C8 SPACE
C348:          72          *
C348:          73          *
C348: 20 EB C3 74          JSR  SETCB   ;SETUP CB INDICATOR
C34E: 4C 4F CA 75          JMP  PINIT   ;XFER TO PASCAL INIT
C351:          C351 76 JPREAD  EQU  *
C351: 20 EB C3 77          JSR  SETCB   ;SETUP CB INDICATOR
C354: 4C 74 CA 78          JMP  PREAD  ;XFER TO PASCAL READ
C357:          C357 79 JPWRITE EQU  *
C357: 20 EB C3 80          JSR  SETCB   ;SETUP CB INDICATOR
C35A: 4C 8E CA 81          JMP  PWRITE  ;XFER TO PASCAL WRITE
C35D:          C35D 82 JPSTAT  EQU  *
C35D: 20 EB C3 83          JSR  SETCB   ;SETUP CB INDICATOR
C360: 4C 94 C9 84          JMP  PSTATUS ;XFER TO PASCAL STATUS
C363:          85          -----
C363:          86          *
C363:          87          * NAME : MOVE
C363:          88          * FUNCTION: PERFORM CROSSBANK MEMORY MOVE
C363:          89          * INPUT : A1=SOURCE ADDRESS
C363:          90          * : A2=SOURCE END
C363:          91          * : A4=DESTINATION START
C363:          92          * : CARRY SET=MAIN-->CARD
C363:          93          * : CLR=CARD-->MAIN
C363:          94          * OUTPUT : NONE
C363:          95          * VOLATILE: NOTHING

```

```

C363:          92 * CALLS : NOTHING
C363:          93 -----
C363:          94 *
C363:          C363 95 MOVE      EQU *
C363: 4B          96 PHA                ; SAVE AC
C364: 9B          97 TYA                ; AND Y
C363: 4B          98 PHA
C366: AD 13 CO   99 LDA RDRAMRD ; SAVE STATE OF
C367: 4B         100 PHA                ; MEMORY FLAGS
C36A: AD 14 CO  101 LDA RDRAMWRT
C36D: 4B         102 PHA
C36E:          103 *
C36E:          104 * SET FLAGS FOR CROSSBANK MOVE:
C36E:          105 *
C36E: 90 0B C37B 106 BCC MOVEC2M ;=>CARD-->MAIN
C370: 8D 02 CO  107 STA RDMMAINRAM ; SET FOR MAIN
C373: 8D 05 CO  108 STA WRCARDRAM ; TO CARD
C376: 80 06 C37E 109 BCS MOVESTRT ;=>(ALWAYS TAKEN)
C37B:          110 *
C37B:          C37B 111 MOVEC2M EQU *
C37B: 8D 04 CO  112 STA WRMAINRAM ; SET FOR CARD
C37B: 8D 03 CO  113 STA RDCARDRAM ; TO MAIN
C37E:          114 *
C37E:          C37E 115 MOVESTRT EQU *
C37E: A0 00         116 LDY #0 ; DUMMY INDEX
C380:          117 *
C380:          C380 118 MOVELOOP EQU *
C380: 81 3C         119 LDA (A1L),Y ; GET A BYTE
C382: 91 42         120 STA (A4L),Y ; MOVE IT
C384: E6 42         121 INC A4L
C386: D0 02 C38A 122 BNE NXTA1
C38B: E6 43         123 INC A4H
C38A: A5 3C         124 LDA A1L
C38C: C5 3E         125 CMP A2L
C38E: A5 3D         126 LDA A1H
C390: E5 3F         127 SBC A2H
C392: E6 3C         128 INC A1L
C394: D0 02 C39B 129 BNE C01
C396: E6 3D         130 INC A1H
C39B: 90 E6 C380 131 C01 BCC MOVELOOP ;=>MORE TO MOVE
C39A:          132 *
C39A:          133 * RESTORE ORIGINAL FLAGS:
C39A:          134 *
C39A: 8D 04 CO  135 STA WRMAINRAM ; CLEAR FLAG2
C39D: 6B         136 PLA ; GET ORIGINAL STATE
C39E: 10 03 C3A3 137 BPL C03 ;=>IT WAS OFF
C3AD: 8D 05 CO  138 STA WRCARDRAM
C3A3:          C3A3 139 C03 EQU *
C3A3: 8D 02 CO  140 STA RDMMAINRAM ; CLEAR FLAG1
C3A6: 6B         141 PLA ; GET ORIGINAL STATE
C3A7: 10 03 C3AC 142 BPL MOVERET ;=>IT WAS OFF
C3A9: 8D 03 CO  143 STA RDCARDRAM
C3AC:          C3AC 144 MOVERET EQU *
C3AC: 6B         145 PLA ; RESTORE Y
C3AD: AB         146 TAY
C3AE: 6B         147 PLA ; AND AC
C3AF: 60         148 RTS
C3B0:          149 -----
C3B0:          150 * NAME : XFER
C3B0:          151 * FUNCTION: TRANSFER CONTROL CROSSBANK
C3B0:          152 * INPUT : #03ED=TRANSFER ADDR
C3B0:          153 * CARRY SET=XFER TO CARD
C3B0:          154 * CLR=XFER TO MAIN
C3B0:          155 * VFLAG CLR=USE STD ZP/STK
C3B0:          156 * SET=USE ALT ZP/STK
C3B0:          157 * OUTPUT : NONE
C3B0:          158 * VOLATILE: #03ED/03EE IN DEST BANK
C3B0:          159 * CALLS : NOTHING
C3B0:          160 * NOTE : ENTERED VIA JMP, NOT JSR
C3B0:          161 -----
C3B0:          162 *
C3B0:          C3B0 163 XFER EQU *
C3B0: 4B          164 PHA ; SAVE AC ON CURRENT STACK
C3B1:          165 *
C3B1:          166 * COPY DESTINATION ADDRESS TO THE
C3B1:          167 * OTHER BANK SO THAT WE HAVE IT
C3B1:          168 * IN CASE WE DO A SWAP:
C3B1:          169 *
C3B1: AD ED 03  170 LDA #03ED ; GET XFERADDR LO
C3B4: 4B          171 PHA ; SAVE ON CURRENT STACK
C3B5: AD EE 03  172 LDA #03FE ; GET XFERADDR HI
C3B8: 4B          173 PHA ; SAVE IT TOO
C3B9:          174 *
C3B9:          175 * SWITCH TO APPROPRIATE BANK:
C3B9:          176 *
C3B9: 90 0A C3C5 177 BCC XFERC2M ;=>CARD-->MAIN
C3BB: 8D 03 CO  178 STA RDCARDRAM ; SET FOR RUNNING
C3BE: 8D 05 CO  179 STA WRCARDRAM ; IN CARD RAM
C3C1: 90 19 C3DC 180 BVC XFERSZP ;=>USE STD ZP/STK
C3C3: 70 0B C3CD 181 BVS XFERAZP ;=>USE ALT ZP/STK
C3C5:          C3C5 182 XFERC2M EQU *

```

```

C3C9: 8D 02 C0      183      STA  RDMAINRAM ;SET FOR RUNNING
C3CB: 8D 04 C0      184      STA  WRMAINRAM ; IN MAIN RAM
C3CC: 50 0F C3DC    185      BVC  XFERSZP ;=>USE STD ZP/STK
C3CD:                186      *
C3CD:                C3CD    187      XFERAZP EQU  * ;SWITCH TO ALT ZP/STK
C3CD: 68           188      PLA  ;STUFF XFERADDR
C3CE: 8D EE 03     189      STA  %03EE ; HI AND
C3D1: 68           190      PLA
C3D2: 8D ED 03     191      STA  %03ED ; LO
C3D3: 68           192      PLA  ;RESTORE AC
C3D4: 8D 09 C0     193      STA  SETALTZP ;SWITCH TO ALT ZP/STK
C3D7: 6C ED 03     194      JMP  (%03ED) ;=>OFF WE GO!
C3DC:                195      *
C3DC:                C3DC    196      XFERSZP EQU  *
C3DC: 68           197      PLA  ;STUFF XFERADDR
C3DD: 8D EE 03     198      STA  %03EE ; HI AND
C3DE: 68           199      PLA
C3E1: 8D ED 03     200      STA  %03ED ; LO
C3E4: 68           201      PLA  ;RESTORE AC
C3E5: 8D 08 C0     202      STA  SETSTDZP ;=>SWITCH TO STD ZP/STK
C3E8: 6C ED 03     203      JMP  (%03ED) ;OFF WE GO!
C3EB:                204
-----
C3EB:                205      * NAME : SETCB
C3EB:                206      * FUNCTION: SETUP IRQ #CB00 PROTOCOL
C3EB:                207      * INPUT : NONE
C3EB:                208      * OUTPUT : NONE
C3EB:                209      * VOLATILE: NOTHING
C3EB:                210      * CALLS : NOTHING
C3EB:                211
-----
C3EB:                212      *
C3EB:                C3EB    213      SETCB EQU  *
C3EB: 4B           214      PHA  ;SAVE AC
C3EC: A9 C3        215      LDA  #CCNOO ;SLOT NUMBER
C3EE: 8D FB 07     216      STA  CBSLOT ;STUFF IT
C3F1: 68           217      PLA  ;RESTORE AC
C3F2: 60           218      RTS
C3F3:                6      INCLUDE CBSPACE
-----
C3F3:                2      * THIS IS THE CBXX SPACE:
-----
C3F3:                4
C3F3:                0000    5      DD  TEST
C3F3:                S      6      DRG #DB00
C3F3:                7      ELSE
-----
NEXT OBJECT FILE NAME IS VIDEO.DBJ1
CB00:                8      ORG  #CB00
CB00:                9      FIN
CB00: 4C 4A CA       10      JMP  PINIT1.0 ;PASCAL 1.0 INIT
CB03:                11      * BASIC INITIALIZATION:
CB03:                12
-----
CB03:                C803    13      BASICINIT EQU  *
CB03: A9 06          14      LDA  #00DFB ;CHECK FB ROM
CB05: CD B3 FB     15      CMP  FBVERSION ; IS IT OK?
CB08: F0 0C CB16   16      BEQ  BINIT1 ;=>YES
CB0A: 20 78 CF     17      JSR  COPYROM ;TRY COPYING TO RAMCARD
CB0D: CD B3 FB     18      CMP  FBVERSION
CB10: F0 04 CB16   19      BEQ  BINIT1 ;=>NOW IT'S GOOD
CB12: 78           20      SEI  ;CRASH THE SYSTEM!
CB13:                CB13    21      HANG EQU  *
CB13: 4C 13 C8       22      JMP  HANG ;HANG FOREVER
CB16:                23      *
CB16:                C816    24      BINIT1 EQU  *
CB16: A9 C3          25      LDA  #CCNOO ;SET HOOKS FOR
CB18: B5 37        26      STA  CSWH
CB1A: B5 39        27      STA  KSWH ; IN & OUT
CB1C: A9 05        28      LDA  #>BASICIN
CB1E: B5 38        29      STA  KSWL
CB20: A9 07        30      LDA  #>BASICOUT
CB22: B5 36        31      STA  CSWL
CB24: A9 00        32      LDA  #0 ;SET FULL 40-COL WINDOW
CB26: B5 20        33      STA  WNDLFT
CB28: A9 00        34      LDA  #0
CB2A: 2C 1A C0     35      BIT  RTEXT ; ASSUME TEXT MODE
CB2D: 30 02 CB31   36      BMI  BINIT1A ;=>YES
CB2F: A9 14        37      LDA  #20 ;IF CR, SET 4 LINES
CB31:                CB31    38      BINIT1A EQU  *
CB31: B5 22          39      STA  WNDTOP
CB33: A9 18         40      LDA  #24
CB35: B5 23         41      STA  WNDBTM
CB37: A9 28         42      LDA  #40
CB39: B5 21         43      STA  WNDWDTH
CB3B: A5 24         44      LDA  CH ; COPY USER CH
CB3D: 8D 7B 04     45      STA  QLDCH ; AS 'OLD' SETTING
CB40: A9 01         46      LDA  #M_IRQ ; GET READY TO CLEAR
CB42: 2D FB 04     47      AND  MODE ; PRESERVE IRQ STATUS
CB43: 8D FB 04     48      STA  MODE ; CLEAR MODES
CB48: 4C 50 C8     49      JMP  BINIT2 ;=>CONTINUE AFTER PASCAL 1.0 HOOK
CB4B:                50
-----
CB4B:                51      *
CB4B:                52      * PASCAL 1.0 INPUT HOOK:
CB4B:                53      *
CB4B:                54      BRK

```

```

CB4C: 00          55          BRK
CB4D:          0000      56          IFNE  *-CB4D      ;ERR IF WRONG ADDR
S          57          FAIL  2,'CB4D      ;HOOK ALIGNMENT'
CB4D:          58          FIN
CB4D: 4C 51 C3     59          JMP   JPREAD      ;=>GO TO STANDARD READ
CB50:          60
-----
CB50:          61 *
CB50:          62 * IS THERE A CARD?
CB50:          63 *
CB50:          C850     64 BINIT2 EQU *
CB50: 20 24 C8      65 JSR  TESTCARD    ;SEE IF CARD PLUGGED IN
CB53: D0 08 C85D   66 BNE  CLEARIT    ;=>IT'S 40
CB55: 06 21          67 ABL  WNDWDTH    ;SET 80-COL WINDOW
CB57: 8D 01 C0     68 STA  SETBOCOL   ;ENABLE 80 STORE
CB5A: 8D 0D C0     69 STA  SETBOVID   ; AND 80 VIDED
CB5D:          70 *
CB5D:          71 * HOME & CLEAR:
CB5D:          72 *
CB5D:          C85D     73 CLEARIT EQU *
CB5D: 8D 0F C0     74 STA  SETALTCHAR ;SET NORM/INV LCASE
CB60: 20 42 CD     75 JSR  X.FF       ;CLEAR IT
CB63: 28          76 PLS          ;CLC ASSURES THAT
CB64: 18          77 CLC           ; WE PRINT THIS
CB65: 08          78 PHP           ; INITIAL CHARACTER
CB66:          80 *
CB66:          81 * COMPENSATE FOR INTEGER BASIC'S
CB66:          82 * HITTING OF $COOD ON INITIAL ENTRY:
CB66:          83 *
CB66:          C866     84 CBBASIC EQU * ;BASIC IN/OUT
CB66: 2C 1F C0     85 BIT  RBOVID    ;WHICH MODE?
CB69: 10 09 C874   86 BPL  CBB2      ;=>40. LEAVE ALONE
CB6B: 8D 01 C0     87 STA  SETBOCOL  ;80. ENABLE STORE.
CB6E:          88 *
CB6E:          89 * MAKE SURE SCROLLING WINDOW IS
CB6E:          90 * AN EVEN NUMBER FOR 80-COLS:
CB6E:          91 *
CB6E: A5 21          92 LDA  WNDWDTH
CB70: 29 FE          93 AND  #$FE
CB72: 85 21          94 STA  WNDWDTH   ;ROUND IT TO LOWER EVEN
CB74:          95 *
CB74:          96 * COPY USER'S CH IF IT DIFFERS FROM
CB74:          97 * WHAT WE LAST PUT THERE:
CB74:          98 *
CB74:          C874     99 CBB2 EQU *
CB74: A5 24          100 LDA  CH        ;GET IT
CB76: CD 78 04     101 CMP  OLDCH     ;IS IT THE SAME?
CB79: F0 03 C87E   102 BEQ  CBB3      ;=>YES, USE OUR OWN
CB7B: 8D 78 05     103 STA  DURCH     ;=>NO, USE HIS
CB7E:          C87E   104 CBB3 EQU *
CB7E: A9 06          105 LDA  $GOODFB  ;CHECK FB ROM
CB80: CD B3 FB     106 CMP  FBVERSION ;IF DIFFERENT, USER
CB83: F0 08 C890   107 BEQ  CBB4      ; HAS RELOADED RAMCARD
CB85:          108 *
CB85:          109 * COPY FB ROM TO LANG CARD:
CB85:          110 *
CB85: 20 78 CF     111 JSR  COPYROM   ;COPY IT AGAIN
CB8B: CD B3 FB     112 CMP  FBVERSION ;IS IT NOW CORRECT?
CB8B: F0 03 C890   113 BEQ  CBB4      ;=>GREAT
CB8D: 4C 13 C8      114 JMP  HANG      ;=>WE HAVE WRONG ROM!
CB90:          115 *
CB90:          C890   116 CBB4 EQU *
CB90: 28          117 PLS          ;RECOVER CARRY (IN/OUT)
CB91: 90 03 C896   118 BCC  BOUT     ;=>PRINT A CHAR
CB93: 4C F6 C8      119 JMP  BINPUT   ;=>INPUT A CHAR
CB96:          C896   120 BOUT EQU *
CB96: AD FB 04     121 LDA  MODE     ;SAY THAT WE'RE
CB99: 29 BF     122 AND  #$25-M.BINPUT ; PRINTING
CB9B: 8D FB 04     123 STA  MODE
CB9E: 4C A1 C8     124 JMP  BPRINT   ;->OUTPUT A CHAR
CBA1:          7
CBA1:          2
-----
CBA1:          3 * BASIC OUTPUT:
CBA1:          4
CBA1:          CBA1   5 BPRINT EQU *
CBA1: AD 78 06     6 LDA  CHAR     ;GET CHARACTER
CBA4: C9 8D     7 CMP  #$8D     ;IS IT C/R?
CBA6: D0 18 C8C0   8 BNE  NDAWAIT  ;=>NOPE, NO VIDWAIT
CBA8: AC 00 C0     9 LDY  KBD      ;IS KEY PRESSED?
CBA8: 10 13 C8C0   10 BPL  NDAWAIT  ;NO
CBA8: C0 93     11 CPY  #$93     ;IS IT CTL-S?
CBAF: D0 0F C8C0   12 BNE  NDAWAIT  ;NO, IGNORE IT
CBB1: 2C 10 C0     13 BIT  KBDSTRB  ;CLEAR STROBE
CBB4: AC 00 C0     14 KBDWAIT LDY  KBD ;WAIT FOR NEXT KEYPRESS
CBB7: 10 FB C8B4   15 BPL  KBDWAIT
CBB9: C0 83     16 CPY  #$83     ;IF CTL-C, LEAVE IT
CBBB: F0 03 C8C0   17 BEQ  NDAWAIT  ; IN THE KBD BUFFER
CBBD: 2C 10 C0     18 BIT  KBDSTRB  ;CLEAR OTHER CHARACTER
C8C0:          19 NDAWAIT EQU *
C8C0: 29 7F     20 AND  #$7F     ;DROP POSSIBLE HI BIT
C8C2: C9 20     21 CMP  #$20     ;IS IT CONTROL CHAR?
C8C4: B0 06 C8CC   22 BCS  BPNTL    ;=>NOPE

```

```

C8C6: 20 99 CB      23          JSR  CTLCHAR      ; EXECUTE POSSIBLE CTL CHAR
C8C7: 4C E2 CB      24          JMP   BIOPRET      ; => EXECUTED OR IGNORED
C8CC:                25 *
C8CC:                26 * NOT A CTL CHAR. PRINT IT.
C8CC:                27 *
C8CC:                28 BPNCTL      EQU   *
C8CC: AC 7B 05      29          LDY   DURCH      ; GET CH
C8CF: AD 7B 06      30          LDA   CHAR      ; GET CHAR (ALL 8 BITS)
C8D2: 20 F2 CE      31          JSR  STORCHAR   ; STUFF ONTO SCREEN
C8D5:                32 *
C8D5:                33 * BUMP THE CURSOR HORIZONTAL:
C8D5:                34 *
C8D5: EE 7B 05      35          INC   DURCH      ; BUMP IT
C8D8: AD 7B 05      36          LDA   DURCH      ; ARE WE PAST THE
C8DB: C5 21          37          CMP   WNDWIDTH   ; END OF THE LINE?
C8DD: 90 03 CBE2    38          BCC   BIOPRET      ; => NO, NO PROBLEM
C8DF: 20 EC CB      39          JSR  X.CR        ; YES, DO C/R
C8E2:                40 *
C8E2:                41 BIOPRET     EQU   *
C8E2: AD 7B 05      42          LDA   DURCH      ; SET CH AND CV
C8E5: 20 AF CE      43          JSR  SETCH      ; FOR BASIC
C8E8: AD FB 05      44          LDA   DURCV
C8EB: 85 25          45          STA   CV
C8ED: 68            46          PLA
C8EE: AA            47          TAX
C8EF: 68            48          PLA
C8F0: A8            49          TAY
C8F1: 68            50          PLA
C8F2: AD 7B 06      51          LDA   CHAR      ; AND AC
C8F5: 60            52          RTS
C8F6:                8          INCLUDE BINPUT
C8F6:                2 * BASIC INPUT:
C8F6:                3 *
C8F6:                4 BINPUT      EQU   *
C8F6: AD FB 04      5          LDA   MODE
C8F9: 09 40          6          ORA   #B.BINPUT ; SAY THAT
C8FB: 8D FB 04      7          STA   MODE      ; WE'RE INPUTTING
C8FE: AD 7B 06      8          LDA   CHAR
C901: A4 24          9          LDY   CH
C903: 91 28         10          STA   (BASL),Y
C905:                11 B.INPUT      EQU   *
C905: 20 DD CE      12          JSR  INVERT      ; CREATE OUR OWN CURSOR IMAGE
C908: 20 15 CB      13          JSR  GETKEY
C908: 8D 7B 06      14          STA   CHAR
C90E: 20 DD CE      15          JSR  INVERT
C911: C9 98         16          CMP   ##9B
C913: F0 03 C918    17          BEQ   ESCAPING ; ESCAPE KEY?
C915: 4C 37 C9      18          JMP   NOESC      ; => YES IT IS
C918:                20 * START AN ESCAPE SEQUENCE:
C918:                21 * WE HANDLE THE FOLLOWING ONES:
C918:                22 * @ - HOME & CLEAR
C918:                23 * E - CLR TO EDL
C918:                24 * F - CLR TO EDS
C918:                25 * I - CURSOR UP
C918:                26 * J - CURSOR LEFT
C918:                27 * K - CURSOR RIGHT
C918:                28 * M - CURSOR DOWN
C918:                29 * R - RESTRICT TO UPPERCASE
C918:                30 * T - TURN OFF ESC-R
C918:                31 * 4 - GOTO 40 COLUMN MODE
C918:                32 * 8 - GOTO 80 COLUMN MODE
C918:                33 * CTL-G- GUIT (PR#0/IN#0)
C918:                34 * THE FOUR ARROW KEYS (AS IJKM)
C918:                35 *
C918:                36          MSB   OFF
C918:                37 ESCAPING  EQU   *
C918: 20 52 CF         38          JSR  ESCDN      ; ESCAPE CURSOR ON
C918: 20 15 CB         39          JSR  GETKEY
C91E: 20 45 CF         40          JSR  ESCOFF     ; GET ESCAPE FUNCTION
C921: 29 7F         41          AND   #57F      ; REPLACE ORIGINAL CHARACTER
C923: C9 60         42          CMP   #56        ; DROP HI BIT
C925: 90 02 C929     43          BCC   ESC1
C927: 29 DF         44          AND   #255-$20 ; IS IT LOWERCASE?
C929:                45 ESC1      EQU   *
C929: A0 11          46          LDY   #ESCNUM    ; UPSHIFT
C92B:                47 ESC2      EQU   *
C92B: D9 72 C9       48          CMP   ESCTAB,Y  ; COUNT/INDEX
C92E: F0 05 C935     49          BEQ   ESC3
C930: 88            50          DEY
C931: 10 FB C92B     51          BPL   ESC2
C933: 30 10 C945     52          BMI   ESCSPEC ; TRY 'EM ALL...
C935:                53 *
C935:                54 ESC3      EQU   *
C935: B9 83 C9         55          LDA   ESCCHAR,Y ; GET CHAR TO "PRINT"
C938: 29 7F         56          AND   #57F
C93A: 20 99 CB         57          JSR  CTLCHAR   ; DROP HI BIT (FLAG)
C93D: B9 83 C9       58          LDA   ESCCHAR,Y ; EXECUTE IT
C940: 30 D6 C918     59          BMI   ESCAPING ; GET FLAG
C942: 4C 05 C9        60          JMP   B.INPUT    ; => STAY IN ESCAPE MODE
C945:                61 *
C945:                62 ESCSPEC  EQU   *

```

```

C945: C9 11 63 CMP #511 ; IS IT ESC-CTLG?
C947: DO 0B C954 64 BNE ESCSPEC2 ;=>NO
C949: 20 AA CD 65 JSR QUIT ; DO THE QUITTING STUFF
C94C: A9 9B 66 LDA #98 ; RETURN CTL-X AS
C94E: 8D 7B 06 67 STA CHAR ; THE CHARACTER
C951: 4C E2 CB 68 JMP BIOPRET ;=>QUIT THE CARD FOREVER
C954: 69 *
C954: C954 70 ESCSPEC2 EQU *
C954: C9 52 C954 71 CMP #'R' ; IS IT ESC-R?
C956: DO 0B C963 72 BNE ESCSPEC3 ;=>NO
C958: AD FB 04 73 LDA MODE ; YES, SET IT
C95B: 09 80 74 ORA #M.ESCR
C95D: 8D FB 04 75 STA MODE
C960: 4C 05 C9 76 ESCNONE JMP B. INPUT ; QUIT ESCAPE MODE
C963: 77 *
C963: C963 78 ESCSPEC3 EQU *
C963: C9 54 79 CMP #'T' ; IS IT ESC-T?
C965: DO F9 C960 80 BNE ESCNONE ;=>NOTHING
C967: AD FB 04 81 LDA MODE
C96A: 29 7F 82 AND #255-M.ESCR
C96C: 8D FB 04 83 STA MODE
C96F: 4C 05 C9 84 JMP B. INPUT ; QUIT ESCAPE MODE
C972: C972 86 ESCTAB EQU *
C972: 40 87 ASC 'e'
C973: 41 88 ASC 'A' ; HANDLE OLD ESCAPES
C974: 42 89 ASC 'B'
C975: 43 90 ASC 'C'
C976: 44 91 ASC 'D'
C977: 45 92 ASC 'E'
C978: 46 93 ASC 'F'
C979: 49 94 ASC 'I'
C97A: 4A 95 ASC 'J'
C97B: 4B 96 ASC 'K'
C97C: 4D 97 ASC 'M'
C97D: 34 98 ASC '4'
C97E: 38 99 ASC '8'
C97F: 0B 100 DFB #0B ; LEFT ARROW
C980: 0A 101 DFB #0A ; DOWN ARROW
C981: 0B 102 DFB #0B ; UP ARROW
C982: 15 103 DFB #15 ; RITE ARROW
C983: 0011 104 ESCNUM EQU *-ESCTAB
C983: 105 MSB DN
C983: C983 106 ESCCHAR EQU *
C983: 0C 107 DFB #0C+#00 ; @: FORMFEED
C984: 1C 108 DFB #1C ; A: FS
C985: 0B 109 DFB #0B ; B: BS
C986: 0A 110 DFB #0A ; C: LF
C987: 1F 111 DFB #1F ; D: US
C988: 1D 112 DFB #1D+#00 ; E: GS
C989: 0B 113 DFB #0B+#00 ; F: VT
C98A: 9F 114 DFB #1F+#80 ; I: US (STAY ESC)
C98B: 8B 115 DFB #0B+#80 ; J: BS (STAY ESC)
C98C: 9C 116 DFB #1C+#80 ; K: FS (STAY ESC)
C98D: 8A 117 DFB #0A+#80 ; M: LF (STAY ESC)
C98E: 11 118 DFB #11+#00 ; 4 : DC1
C98F: 12 119 DFB #12+#00 ; 8 : DC2
C990: 8B 120 DFB #0B+#80 ; <-: BS (STAY ESC)
C991: 8A 121 DFB #0A+#80 ; DN: LF (STAY ESC)
C992: 9F 122 DFB #1F+#80 ; UP: US (STAY ESC)
C993: 9C 123 DFB #1C+#80 ; ->: FS (STAY ESC)
C994: 124 -----
C994: 126 * PASCAL STATUS:
C994: 127 -----
C994: 128 -----
C994: C994 129 PSTATUS EQU *
C994: AA 130 TAX ; SAVE REQUEST CODE
C995: 20 CB CF 131 JSR PSETUP ; SETUP ZP STUFF
C99B: 8A 132 TXA ; IS IT 'READY FOR OUTPUT?'
C997: DO 03 C99E 133 BNE PSTATUS2 ;=>NO
C99B: 3B 134 SEC ; YES: READY FOR OUTPUT
C99C: B0 16 C9B4 135 BCS PSTATUS4
C99E: 136 *
C99E: C99E 137 PSTATUS2 EQU *
C99E: C9 01 C990 138 CMP #1 ; IS IT 'ANY INPUT?'
C9A0: F0 0E C9B0 139 BEQ PSTATUS3 ;=>YES
C9A2: A2 03 140 LDX #3 ; IDRESULT='ILGL OPERATION'
C9A4: 1B 141 CLC
C9A5: 60 142 RTS
C9A6: 143 -----
C9A6: 144 * PASCAL I.0 OUTPUT HOOK:
C9A6: 145 -----
C9A6: 00 146 BRK ; PADDING
C9A7: 00 147 BRK
C9A8: 00 148 BRK
C9A9: 00 149 BRK
C9AA: 0000 150 IFNE *-C9AA
C9AA: S 151 FAIL Z,'C9AA HOOK ALIGNMENT'
C9AA: 152 FIN
C9AA: AD 7B 06 153 LDA CHAR ; GET OUTPUT CHARACTER
C9AD: 4C 57 C3 154 JMP JPWRITE ;=>USE STANDARD WRITE
C9B0: 155 *

```



```

C9B0:          C9B0 156 PSTATUS3 EQU *
C9B0: AD 00 CO 157 LDA KBD ; IS THERE A KEYPRESS?
C9B3: 0A 158 ASL A ; STROBE-->CARRY
C9B4: A2 00 159 PSTATUS4 LDX #0 ; IDRESULT='GOOD'
C9B6: 60 160 RTS
C9B7: 162 -----
C9B7: 163 *---- BASIC INPUT, CONTINUED:
C9B7: 164 *---- NOT AN ESCAPE SEQUENCE----
C9B7: 165 -----
C9B7:          C9B7 166 NOESC EQU * ; NOT ESCAPE KEY
C9B7: 167 *
C9B7: C9 95 168 CMP #95 ; IS IT PICK?
C9B9: D0 0B C9C6 169 BNE B.NOPICK ; =>NOPE
C9BB: AC 7B 05 170 LDY DURCH ; YOU CAN PICK YER FRIENDS....
C9BE: 20 01 CF 171 JSR PICK ; YES, PICK THE CHAR
C9C1: 09 80 172 ORA #80 ; ALWAYS PICK AS NORMAL
C9C3: 8D 7B 06 173 STA CHAR ; SAVE AS KEYSTROKE
C9C6: 174 *
C9C6: 175 * TRACK QUOTATION MARKS FOR THE
C9C6: 176 * RESTRICT-UPPERCASE FEATURE:
C9C6: 177 *
C9C6:          C9C6 178 B.NOPICK EQU *
C9C6: AD FB 04 179 LDA MODE ; ARE WE DOING LITERAL INPUT?
C9C9: 29 10 180 AND #M.LIT
C9CB: D0 12 C9DF 181 BNE B.CHKCAN ; =>YES
C9CD: 182 *
C9CD: 183 * LITERAL INPUT'S INACTIVE. SEE IF
C9CD: 184 * WE CAN START LITERAL INPUT:
C9CD: 185 *
C9CD: AD 7B 06 186 LDA CHAR ; GET THE CHAR
C9D0: C9 A2 187 CMP #A2 ; IS IT A DOUBLE QUOTE?
C9D2: F0 23 C9F7 188 BEG B.FLIP ; =>YES, FLIP LITERAL MODE
C9D4: C9 8B 189 CMP #8B ; IS HE MOVING LEFT?
C9D6: D0 32 CA0A 190 BNE B.FIXCHR ; =>NOPE, JUST REG CHAR
C9D8: 20 27 CA 191 JSR GETPRIOR ; GRAB PRIOR CHAR
C9DB: D0 2D CA0A 192 BNE B.FIXCHR ; =>NOT DELETING A QUOTE
C9DD: F0 18 C9F7 193 BEG B.FLIP ; (ALWAYS) HIE'S DELETED THE QUOTE
C9DF: 194 *
C9DF: 195 * LITERAL INPUT'S ACTIVE. SEE IF
C9DF: 196 * IT SHOULD BE CANCELLED YET:
C9DF: 197 *
C9DF:          C9DF 198 B.CHKCAN EQU *
C9DF: AD 7B 06 199 LDA CHAR ; GET CURRENT CHAR
C9E2: C9 A2 200 CMP #A2 ; IS CURRENT CHAR THE CLOSING QUOTE?
C9E4: F0 1C CA02 201 BEG B.CANLIT ; =>YES
C9E6: C9 9B 202 CMP #9B ; CANCEL LITERAL INPUT
C9E8: F0 18 CA02 203 BEG B.CANLIT ; IF CTLX OR RETURN
C9EA: C9 8D 204 CMP #8D ; OR BACK OVER "
C9EC: F0 14 CA02 205 BEG B.CANLIT
C9EE: C9 8B 206 CMP #8B ; BACKSPACE?
C9F0: D0 18 CA0A 207 BNE B.FIXCHR ; =>NO, NOT DELETING QUOTE
C9F2: 20 27 CA 208 JSR GETPRIOR ; GET CHAR HE'S DELETING
C9F5: D0 13 CA0A 209 BNE B.FIXCHR ; =>NOT DELETING A QUOTE
C9F7: 210 *
C9F7:          C9F7 211 B.FLIP EQU *
C9F7: AD FB 04 212 LDA MODE ; FLIP THE MODE
C9FA: 49 10 213 EOR #M.LIT
C9FC: 8D FB 04 214 STA MODE
C9FF: 4C 0A CA 215 JMP B.FIXCHR
CA02:          CA02 216 B.CANLIT EQU *
CA02: AD FB 04 217 LDA MODE
CA05: 29 EF 218 AND #255-M.LIT ; CANCEL LITERAL INPUT
CA07: 8D FB 04 219 STA MODE
CA0A: 220 *
CA0A:          CA0A 221 B.FIXCHR EQU *
CA0A: AD FB 04 222 LDA MODE ; ESC-R FACILITY ACTIVE?
CA0D: 29 80 223 AND #M.ESCR
CA0F: F0 13 CA24 224 BEG B.INRET ; =>NOPE
CA11: AD FB 04 225 LDA MODE ; LITERAL INPUT ACTIVE?
CA14: 29 10 226 AND #M.LIT
CA16: D0 0C CA24 227 BNE B.INRET ; =>YES, NO UPSHIFT
CA18: AD 7B 06 228 LDA CHAR ; GET THE CHAR
CA1B: C9 E0 229 CMP #E0 ; IS CHAR LOWERCASE?
CA1D: 90 05 CA24 230 BCC B.INRET ; =>NO, NO NEED TO SHIFT IT
CA1F: 29 DF 231 AND #8DF ; RESTRICT TO U/C
CA21: 8D 7B 06 232 STA CHAR
CA24: 233 -----
CA24:          CA24 234 B.INRET EQU *
CA24: 4C E2 CB 235 JMP BIOPRET ; =>RETURN TO CALLER
CA27: 237 -----
CA27: 238 * NAME : GETPRIOR
CA27: 239 * FUNCTION: GET CHAR BEFORE CURSOR
CA27: 240 * INPUT : DURCH, DURCV
CA27: 241 * OUTPUT : 'BEG' IF CHAR=DBL QUOTE
CA27: 242 * : 'BNE' IF NOT
CA27: 243 * VOLATILE: AC, 'TEMP1'
CA27: 244 * CALLS : PICK, X.BS, X.FS
CA27: 245 -----
CA27: 246 *
CA27:          CA27 247 GETPRIOR EQU *
CA27: AD FB 05 248 LDA DURCV ; DON'T TRY TO LOOK

```

```

CA2A: 0D 7B 05      249      ORA   OURCH      ; BACK IF @ UPPER-LEFT
CA2D: FO 1A CA49    250      BEQ   GPX        ; CORNER OF WINDOW!!!
CA2F: 9B           251      TYA                   ; SAVE Y
CA30: 4B           252      PHA                   ;
CA31: 20 DB CB     253      JSR   X.BS         ; BACK UP 1 CHAR
CA34: AC 7B 05     254      LDY   OURCH      ; GET CH AND
CA37: 20 01 CF     255      JSR   PICK        ; PICK PRIOR CHAR
CA3A: 09 80        256      ORA   ##80        ; PICK AS NORMAL VIDEO
CA3C: 8D 7B 04     257      STA   TEMP1       ; HOLD CHAR
CA3F: 20 26 CC     258      JSR   X.FS        ;
CA42: 6B           259      PLA                   ; RESTORE
CA43: AB           260      TAY                   ; Y
CA44: AD 7B 04     261      LDA   TEMP1       ;
CA47: C9 A2        262      CMP   ##A2        ; IS IT DBL QUOTE?
CA49:           CA49  263 GPX     EQU   *            ;
CA49: 60           264      RTS                ; RETURN WITH BEG/BNE
CA4A:           265      INCLUDE PINIT
CA4A:           266      -----
CA4A:           267      * PASCAL INITIALIZATION:
CA4A:           268      -----
CA4A:           CA4A  269 PINIT1.0 EQU *
CA4A: A9 22        270 LDA   ##M.PASCAL+M.PAS1.0
CA4C: 4C 51 CA    271 JMP   PINIT2
CA4F:           CA4F  272 PINIT EQU *
CA4F: A9 20        273 LDA   ##M.PASCAL ; SAY WE'RE
CA51:           274 *
CA51:           CA51  275 PINIT2 EQU *
CA51: 8D FB 04     276 STA   MODE        ; RUNNING PASCAL
CA54: 20 9B CD     277 JSR   FULLBO     ; SET FULL 24X80 WINDOW
CA57: 20 CB CF     278 JSR   PSETUP     ; SETUP ZP STUFF
CA5A:           279 * BASE ADDR IS WRONG, BUT X.FF FIXES IT BELOW:
CA5A:           280 * JSR BASCALC ; FORCE A GOOD BASCALC
CA5A:           281 *
CA5A:           282 * SEE IF THE CARD'S PLUGGED IN:
CA5A:           283 *
CA5A: 20 24 CB     284 JSR   TESTCARD   ; IS IT THERE?
CA5D: FO 03 CA62   285 BEQ   PIGOOD     ; =>YES
CA5F: A2 09        286 LDX   #9         ; IORESULT='NO DEVICE'
CA61: 60           287 RTS
CA62:           288 *
CA62:           CA62  289 PIGOOD EQU *
CA62: 8D 01 C0     290 STA   SETBOCOL   ; ENABLE 80 STORE
CA65: 8D 0D C0     291 STA   SETBOVID   ; AND 80 VIDEO
CA68: 8D 0F C0     292 STA   SETALTCHAR ; NORM+INV LCASE
CA6B: 20 42 CD     293 JSR   X.FF       ; HOME & CLEAR IT
CA6E: 20 DD CE     294 JSR   INVERT     ; PUT CURSOR THERE
CA71: A2 00        295 LDX   #0         ; IORESULT='GOOD'
CA73: 60           296 RTS
CA74:           297 INCLUDE PREAD
CA74:           298 -----
CA74:           299 * PASCAL INPUT:
CA74:           300 -----
CA74:           CA74  301 PREAD EQU *
CA74: 20 CB CF     302 JSR   PSETUP     ; SETUP ZP STUFF
CA77:           303 *
CA77: 20 15 CB     304 JSR   GETKEY     ; GET A KEYSTROKE
CA7A: 29 7F        305 AND   ##7F       ; DROP HI BIT
CA7C: 8D 7B 06     306 STA   CHAR       ; SAVE THE CHAR
CA7F: A2 00        307 LDX   #0         ; IORESULT='GOOD'
CAB1: AD FB 04     308 LDA   MODE       ; ARE WE IN 1.0-MODE?
CAB4: 29 02        309 AND   ##M.PAS1.0
CAB6: FO 02 CABA   310 BEQ   PREADRET2 ; =>NOPE
CAB8: A2 C3        311 LDX   #<CNOO    ; YES, RETURN CN IN X
CABA:           312 *
CABA:           CABA  313 PREADRET2 EQU *
CABA: AD 7B 06     314 LDA   CHAR       ; RESTORE CHAR
CABD: 60           315 RTS
CABE:           316 INCLUDE PWRITE
CABE:           317 -----
CABE:           318 * PASCAL OUTPUT:
CABE:           319 -----
CABE:           CABE  320 PWRITE EQU *
CABE: 8D 7B 06     321 STA   CHAR       ; SAVE CHARACTER
CA91: 20 CB CF     322 JSR   PSETUP     ; SETUP ZP STUFF
CA94:           323 *
CA94: 20 DD CE     324 JSR   INVERT     ; TURN CURSOR OFF
CA97: AD FB 04     325 LDA   MODE       ; ARE WE DOING GOTOXY?
CA9A: 29 0B        326 AND   ##M.GDXY
CA9C: FO 2D CACB   327 BEQ   PWRITE3    ; =>NO, PRINT IT
CA9E:           328 *
CA9E:           329 * HANDLE GOTOXY STUFF:
CA9E:           330 *
CA9E:           CAAE  331 PWRITE2 EQU *
CA9E: AD FB 06     332 LDA   XCOORD    ; ARE WE WAITING FOR X?
CAA1: 10 0C CAAF   333 BPL   GETY       ; =>NO, THIS IS Y
CAA3: AD 7B 06     334 LDA   CHAR
CAA6: 3B           335 SEC
CAA7: E9 20        336 SBC   #32        ; MAKE BINARY
CAA9: 8D FB 06     337 STA   XCOORD
CAAC: 4C 0F CB     338 JMP   PWRITERET ; =>NOW WAIT FOR Y
CAAF:           339 *

```

```

CAAF:          25 * NOW DO THE GOTOXY:
CAAF:          26 *
CAAF:          27 GETY      EQU      *
CAAF:AD 7B 06  28      LDA      CHAR      ; CONVERT YCOORD
CAB2:3B       29      SEC
CAB3:E9 20    30      SBC      #32
CAB5:8D FB 05 31      STA      OURCV
CABB:20 51 CB 32      JSR      BASCALC    ; COMPUTE BASE ADDRESS
CABB:AD FB 06 33      LDA      XCOORD
CABE:BD 7B 05 34      STA      OURCH
CAC1:AD FB 04 35      LDA      MODE      ; TURN OFF GOTOXY
CAC4:29 F7    36      AND      #255-M.GOXY
CAC6:BD FB 04 37      STA      MODE
CAC9:DO 44 CBOF 38      BNE      PWRITERET ; =>DONE (ALWAYS TAKEN)
CACB:          39 *
CACB:          40 PWRITE3   EQU      *
CACB:AD 7B 06  41      LDA      CHAR      ; GET CHAR TO PRINT
CACE:C9 1E    42      CMP      #*1E      ; IS IT GOTOXY?
CAD0:F0 0A CADC 43      BEQ      STARTXY    ; =>YES
CAD2:C9 20    44      CMP      #*20      ; IS IT OTHER CTL?
CAD4:B0 15 CAEB 45      BCS      PWRITE4    ; =>NO, PRINT IT
CAD6:20 99 CB 46      JSR      CTLCHAR    ; EXECUTE IT IF POSSIBLE
CAD9:4C 0F CB 47      JMP      PWRITERET ; =>EXECUTED OR IGNORED
CADC:          48 *
CADC:          49 * START THE GOTOXY SEQUENCE:
CADC:          50 *
CADC:          51 STARTXY   EQU      *
CADC:AD FB 04  52      LDA      MODE      ; TURN ON FLAG
CADF:09 0B    53      ORA      #M.GOXY
CAE1:BD FB 04  54      STA      MODE
CAE4:A9 FF    55      LDA      #-1
CAE6:BD FB 06  56      STA      XCOORD    ; SHOW WE NEED IT
CAE9:30 24 CBOF 57      BMI      PWRITERET ; =>EXIT TILL COORDS COME BY (ALWAYS)
CAEB:          58 *
CAEB:          59 * JUST A PRINTABLE CHARACTER:
CAEB:          60 *
CAEB:          61 PWRITE4   EQU      *
CAEB:09 80    62      DRA      #*80      ; FORCE TO NORMAL
CAED:AC 7B 05  63      LDY      OURCH      ; GET CH
CAF0:20 F2 CE  64      JSR      STORCHAR    ; STUFF IT!
CAF3:          65 *
CAF3:          66 * BUMP CURSOR HORIZONTAL:
CAF3:          67 *
CAF3:EE 7B 05  68      INC      OURCH      ; BUMP IT
CAF6:AD 7B 05  69      LDA      OURCH      ; ARE WE PAST THE
CAF9:C5 21    70      CMP      WNDWDTH    ; END OF THE LINE?
CAFB:90 12 CBOF 71      BCC      PWRITERET ; =>NO, NO PROBLEM
CAFJ:          72 *
CAFJ:          73 * IF IN TRANSPARENT MODE, DON'T
CAFJ:          74 * WRAPAROUND THE RIGHT EDGE...
CAFJ:          75 *
CAFJ:AD FB 04  76      LDA      MODE      ; GET MODE
CB00:29 01    77      AND      #M.TRANS    ; WELL??
CB02:F0 05 CB09 78      BEQ      PWRAP      ; =>NOT TRANSPARENT
CB04:CE 7B 05  79      DEC      OURCH      ; PIN AT RIGHT EDGE
CB07:DO 06 CBOF 80      BNE      PWRITERET ; (ALWAYS TAKEN)
CB09:          81 *
CB09:          82 PWRAP     EQU      *
CB09:20 EC CB  83      JSR      X.CR      ; YES, DO C/R
CB0C:20 91 CC  84      JSR      X.LF      ; AND L/F
CB0F:          85 *
CB0F:          86 PWRITERET EQU      *
CB0F:20 DD CE  87      JSR      INVERT    ; TURN CURSOR ON
CB12:A2 00    88      LDX      #0      ; IORESULT='GOOD'
CB14:60       89      RTS
CB15:          12      INCLUDE SUBS1
CB15:          2 -----
CB15:          3 * NAME      : GETKEY
CB15:          4 * FUNCTION: GET A KEYSTROKE
CB15:          5 * INPUT   : NONE
CB15:          6 * OUTPUT  : AC=KEYCODE
CB15:          7 * VOLATILE: NONE
CB15:          8 -----
CB15:          9 *
CB15:          10 GETKEY   EQU      *
CB15:E6 4E    11      INC      RNDL      ; BUMP RANDOM SEED
CB17:DO 02 CB18 12      BNE      GETK2
CB19:E6 4F    13      INC      RNDH
CB1B:          14 GETK2   EQU      *
CB1B:AD 00 CO  15      LDA      KBD      ; KEYPRESS?
CB1E:10 F3 CB15 16      BPL      GETKEY    ; =>NOPE
CB20:BD 10 CO  17      STA      KBDSTRB   ; CLEAR STROBE
CB23:60       18      RTS
CB24:          19 -----
CB24:          20 * NAME      : TESTCARD
CB24:          21 * FUNCTION: SEE IF BOCOL CARD PLUGGED IN
CB24:          22 * INPUT   : NONE
CB24:          23 * OUTPUT  : 'BE' IF CARD AVAILABLE
CB24:          24 *          : 'BNE' IF NOT
CB24:          25 * VOLATILE: AC,Y
CB24:          26 -----
CB24:          27 *

```

```

CB24:      CB24  2B TESTCARD  EQU  *
CB24: AD 1C CO  29          LDA  RDPAGE2    ;REMEMBER CURRENT VIDEO DISPLAY
CB27: 0A      30          ASI  A                ; IN THE CARRY
CB2B: A9 8B    31          LDA  #5BB          ;USEFUL CHAR FOR TESTING
CB2A: 2C 1B CO  32          BIT  RDBOCOL     ;REMEMBER VIDEO MODE IN 'N'
CB2D: 8D 01 CO  33          STA  SETBOCOL   ;ENABLE 80COL STORE
CB30: 0B      34          PHP                    ;LOCK INTERRUPTS WHILE
CB31: 7B      35          SETI                   ; SCREENHOLES ARE WRONG
CB32: 0B      36          PHP                    ;SAVE 'N' AND 'C' FLAGS
CB33: 8D 55 CO  37          STA  TXTPAGE2   ;SET PAGE2
CB36: AC 00 04  38          LDY  #0400        ;GET FIRST CHAR
CB39: 8D 00 04  39          STA  #0400        ;SET TO A '/'
CB3C: AD 00 04  40          LDA  #0400        ;GET IT BACK FROM RAM
CB3F: 8C 00 04  41          STY  #0400        ;RESTORE ORIG CHAR
CB42: 2B      42          PLP                    ;RESTORE 'N' AND 'C' FLAGS
CB43: B0 03 CB48 43          BCS  STAY2     ;STAY IN PAGE2
CB45: 8D 54 CO  44          STA  TXTPAGE1   ;RESTORE PAGE1
CB4B:          CB4B  45 STAY2     EQU  *
CB4B: 30 03 CB4D 46          BMI  STAY80    ;=>STAY IN 80COL MODE
CB4A: 8D 00 CO  47          STA  CLRBOCOL  ;TURN OFF 80COL STORE
CB4D:          CB4D  48 STAY80    EQU  *
CB4D: 2B      49          PLP                    ;ALLOW IRQ AGAIN
CB4E:          CB4E  50 TESTFAIL  EQU  *
CB4E: C9 8B    51          CMP  #5BB      ;WAS CHAR VALID?
CB50: 60      52          RTS                    ;RETURN RESULT AS BEQ/BNE
CB51:          53 -----
CB51:          54 * NAME      : BASCALC, BASCALCZ
CB51:          55 * FUNCTION:  CALC BASE ADDR FOR SCREEN LINE
CB51:          56 * INPUT   : DURCV (BASCALC)
CB51:          57 * OUTPUT  : AC=CV (BASCALCZ)
CB51:          58 * OUTPUT  : BASL/BASH
CB51:          59 * VOLATILE: NOTHING
CB51:          60 * CALLS   : SNIFFIRQ
CB51:          61 -----
CB51:          62 *
CB51:          FC75 63 SNIFFIRQ  EQU  #FC75
CB51:          64 *
CB51:          CB51 65 BASCALC  EQU  *
CB51:          66          OLC                    ;RIPPED OFF FROM FB ROM
CB51: 1B      67          BCC  BSCLC1    ;SHOW ENTRY POINT
CB52: 90 01 CB54 68 BASCALCZ  EQU  *
CB54:          69          SEC                    ;SHOW ENTRY POINT
CB55:          CB55 70 BSCLC1  EQU  *
CB55: 4B      71          PHA                    ;SAVE AC
CB56: B0 03 CB5B 72          BCS  BSCLC1A  ;=>CV ALREADY IN AC
CB5B: AD FB 05 CB58 73          LDA  DURCV
CB5B:          CB5B 74 BSCLC1A EQU  *
CB5B: 4B      75          PHA                    ;SAVE AC
CB5C: 4A      76          LSR                    ;SAVE AC
CB5D: 29 03 CB5D 77          AND  #503
CB5F: 09 04 CB5E 78          ORA  #504
CB61: 85 29 CB5F 79          STA  BASH
CB63: 8D FB 07 CB63 80          STA  OLDBASH  ;SAVE FOR F/W PROTOCOL
CB66: 6B      81          PLA
CB67: 29 1B CB64 82          AND  #51B
CB69: 90 02 CB6D 83          BCC  BSCLC2
CB6B: 69 7F CB6E 84          ADC  #57F
CB6D: 85 2B CB6F 85 BSCLC2  STA  BASL
CB6F: 0A      86          ASL  A
CB70: 0A      87          ASL  A
CB71: 05 2B CB70 88          ORA  BASL
CB73: 85 2B CB71 89          STA  BASL
CB75:          90 *
CB75:          91 * HANDLE THE SCROLLING WINDOW:
CB75:          92 *
CB75: A5 20 CB75 93          LDA  WNDLFT
CB77: 0B      94          PHP                    ;PRESERVE CARRY
CB78: 2C 1F CO CB78 95          BIT  RDBOVID   ;WHICH MODE?
CB7B: 10 01 CB7E 96          BPL  BASCLC3   ;=>40, NO DIVIDE
CB7D: 4A      97          LSR                    ;DIVIDE BY 2 FOR 80COL WINDOW
CB7E:          CB7E 98 BASCLC3  EQU  *
CB7E: 2B      99          PLP                    ;RESTORE CARRY
CB7F: 65 2B CB7F 100         ADC  BASL      ;ADJUST BASE FOR WNDLFT
CB81: 85 2B CB80 101         STA  BASL
CB83: 8D 7B 07 CB81 102         STA  OLDBASL  ;SAVE FOR F/W PROTOCOL
CB86:          103 *
CB86:          104 * SNIFF FOR IRQ IF NECESSARY:
CB86:          105 *
CB86: AD FB 04 CB86 106         LDA  MODE
CB89: 29 01 CB87 107         AND  #M_IRQ
CB8B: F0 0A CB97 108         BEQ  BASCLCX   ;=>IRQ DISABLED, RETURN
CB8D: AD FB 04 CB88 109         LDA  MODE
CB90: 29 20 CB89 110         AND  #M_PASCAL
CB92: D0 03 CB97 111         BNE  BASCLCX   ;=>DON'T SNIFF UNDER PASCAL
CB94: 20 75 FC CB90 112         JSR  SNIFFIRQ  ;GO DO IT
CB97:          CB97 113 BASCLCX  EQU  *
CB97: 6B      114          PLA                    ;RESTORE AC
CB9B: 60      115          RTS
CB99:          116 -----
CB99:          117 * NAME      : CTLCHAR
CB99:          118 * FUNCTION:  EXECUTE CTL CHAR
CB99:          119 * INPUT   : AC=CHAR

```

```

CB99:          120 * OUTPUT   : 'BCS' IF NOT CTL
CB99:          121 *          : 'BCC' IF CTL EXECUTED
CB99:          122 * VOLATILE: NOTHING
CB99:          123 * CALLS   : MANY THINGS
CB99:          124
-----
CB99:          125 *
CB99:          CB99 126 CTLCHAR EQU *
CB99: 8D 7B 04 127 STA TEMP1 ; TEMP SAVE OF CHAR
CB9C: 48          128 PHA ; SAVE AC
CB9D: 98          129 TYA ; SAVE Y
CB9E: 48          130 PHA
CB9F:          131 *
CB9F: AC 7B 04 132 LDY TEMP1 ; GET CHAR IN QUESTION
CBA2: C0 07 133 CPY ##07 ; IS IT NUL...ACK?
CBA4: 90 05 CBAB 134 BCC CTLCHARX ;=>YES, NOT USED
CBA6: B9 71 CC CBAB 135 LDA CTLADH-7, Y ; IS IT CTL?
CBA9: D0 03 CBAE 136 BNE CTLGD ;=>YES
CBAB:          CBAB 137 CTLCHARX EQU *
CBAB: 3B          138 SEC ; SAY 'NOT CTL'
CBAC: B0 04 CBB2 139 BCS CTLRET ;=>DONE
CBAE:          140 *
CBAE:          CBAE 141 CTLGD EQU *
CBAE: 20 B6 CB 142 JSR CTLXFER ; EXECUTE SUBROUTINE
CBB1:          143 *
CBB1: 1B          144 CLC ; SAY 'CTL CHAR EXECUTED'
CBB2:          CBB2 145 CTLRET EQU *
CBB2: 68          146 PLA ; RESTORE
CBB3: AB          147 TAY ; Y
CBB4: 68          148 PLA ; AND AC
CBB5: 60          149 RTS
CBB6:          150 *
CBB6:          CBB6 151 CTLXFER EQU *
CBB6: 48          152 PHA ; PUSH ONTO STACK FOR
CBB7: B9 5B CC 153 LDA CTLADL-7, Y ; TRANSFER TRICK
CBB8: 48          154 PHA
CBBB: 60          155 RTS ; XFER TO ROUTINE
CBBC:          156 *
CBBC:          CBBC 157 * EXECUTE BELL:
CBBC:          158 *
CBBC:          CBBC 159 X.BELL EQU *
CBBC: A9 4C 160 LDA ##40 ; RIPPED OFF FROM MONITOR
CBBE: 20 CF CB 161 JSR WAIT
CBC1: A0 C0 162 LDY ##C0
CBC3: A9 0C 163 BELL2 LDA ##0C
CBC5: 20 CF CB 164 JSR WAIT
CBCB: AD 30 C0 165 LDA SPKR
CBCB: 8B          166 DEY
CBCC: D0 F5 CBC3 167 BNE BELL2
CBCE: 60          168 RTS
CBCF:          CBCF 169 *
CBCF: 3B          170 WAIT EQU * ; RIPPED OFF FROM MONITOR ROM
CBD0: 48          171 SEC
CBD1: E9 01 172 WAIT2 PHA
CBD3: D0 FC CBD1 173 WAIT3 SBC #1
CBD5: 68          174 PLA WAIT3
CBD6: E9 01 175 SBC #1
CBD8: D0 F6 CBD0 176 BNE WAIT2
CBDA: 60          177 RTS
CBD9:          178 *
CBD9:          CBD9 179 * EXECUTE BACKSPACE:
CBD9:          180 *
CBD9:          CBD9 181 *
CBD9: CE 7B 05 182 X.BS EQU *
CBD9: DE 10 0B CBEB 183 DEC DURCH ; BACK UP CH
CBE0: A5 21 184 BPL BSDONE ;=>DONE
CBE2:          CBE2 185 LDA WNDWIDTH ; BACK UP TO PRIOR LINE
CBE2: 8D 7B 05 186 B540 EQU *
CBE5: CE 7B 05 187 STA DURCH ; SET CH
CBE8: 20 34 CC 188 DEC DURCH
CBE8:          CBE8 189 JSR X.US ; NOW DO REV LINEFEED
CBE8: 60          190 BSDONE EQU *
CBE8:          191 RTS
CBE9:          192 *
CBE9:          CBE9 193 * EXECUTE CARRIAGE RETURN:
CBE9:          194 *
CBE9:          CBE9 195 X.CR EQU *
CBE9: AD FB 04 196 LDA MODE ; WHICH LANGUAGE?
CBEF: 29 20 197 AND ##F PASCAL
CBF1: D0 0A Cbfd 198 BNE X.CRPAS ;=>PASCAL, NO CLR EOL
CBF3: AD FB 04 199 LDA MODE ; INPUT OR OUTPUT?
CBF6: 29 40 200 AND ##B INPUT
CBF8: F0 03 Cbfd 201 BEQ X.CRPAS ;=>OUTPUT, NO CLEARING
CBFA: 20 48 CD 202 JSR X.GS ; CLEAR TO EOL
CBFD:          203 *
CBFD:          Cbfd 204 X.CRPAS EQU *
CBFD: A9 00 205 LDA #0 ; BACK UP CH TO
CBFF: 8D 7B 05 206 STA DURCH ; BEGINNING OF LINE
CC02: AD FB 04 207 LDA MODE ; ARE WE IN BASIC?
CC05: 29 20 208 AND ##F PASCAL
CC07: D0 03 CC0C 209 BNE X.CRRET ;=>PASCAL, AVOID AUTO L/F
CC09: 20 91 CC 210 JSR X.LF ; EXECUTE AUTO LF FOR BASIC
CC0C:          CC0C 211 X.CRRET EQU *

```

```

CC0C: 60          212          RTS
CC0D:          0000 213          DO 0          ; NO MORE ROM SPACE!
S              214 *
S              215 *          EXECUTE SYNC:
S              216 *
S              217 X.SYN      EGU *
S              218          LDA RDVBLBAR      ; WAIT FOR VBL
S              219          BPL X.SYN        ; =>WAIT FOR VIDED SCAN
S              220 X.SYN2    LDA RDVBLBAR      ; NOW WAIT FOR
S              221          BMI X.SYN2      ; BLANKING TO BEGIN
S              222          RTS
CC0D:          223          FIN
CC0D:          224 *
CC0D:          225 * EXECUTE HOME:
CC0D:          226 *
CC0D:          CC0D 227 X.EM      EGU *
CC0D: A5 22      228          LDA WNDTOP
CC0F: BD FB 05   229          STA OURCV      ; STUFF CV
CC12: A9 00      230          LDA #0
CC14: BD 7B 05   231          STA OURCH      ; STUFF CH
CC17: 4C 51 CB   232          JMP BASCALC     ; RETURN VIA BASCALC (UGH!)
CC1A:          233 *
CC1A:          234 * EXECUTE CLEAR LINE:
CC1A:          235 *
CC1A: A4 21      CC1A 236 X.SUB      EGU *
CC1C: BB          237          LDY WNDWDTH
CC1D:          CC1D 238          DEY
CC1D: A9 A0      CC1D 239 X.SUBBO     EGU *
CC1F:          CC1F 240          LDA #'          ; BLANKIE BLANK
CC1F: 20 F2 CE   CC1F 241 X.SUBLP     EGU *
CC22: BB          242          JSR STORCHAR    ; STUFF THE BLANK
CC23: 10 FA      CC1F 243          DEY
CC25: 60          244          BPL X.SUBLP     ; =>CLEAR THE LINE
CC26:          245          RTS
CC26:          246 *
CC26:          247 * EXECUTE FORWARD SPACE:
CC26:          248 *
CC26:          CC26 249 X.FS        EGU *
CC26: EE 7B 05   250          INC OURCH      ; BUMP CH
CC29: AD 7B 05   251          LDA OURCH      ; GET THE POSITION
CC2C: C5 21      252          CMP WNDWDTH    ; OFF THE RIGHT SIDE?
CC2E: 90 03      CC33 253          BCC X.FSRET    ; =>NO, GOOD
CC30: 20 EC CB   254          JSR X.CR       ; =>YES, WRAP AROUND
CC33:          255 *
CC33:          CC33 256 X.FSRET     EGU *
CC33: 60          257          RTS
CC34:          258 *
CC34:          259 * EXECUTE REVERSE LINEFEED:
CC34:          260 *
CC34:          CC34 261 X.US        EGU *
CC34: CE FB 05   262          DEC OURCV      ; BACK UP CV
CC37: 30 07      CC40 263          BMI X.US1      ; =>OFF TOP OF SCREEN
CC39: AD FB 05   264          LDA OURCV
CC3C: C5 22      265          CMP WNDTOP     ; OFF TOP OF WINDOW?
CC3E: B0 05      CC45 266          BCS X.US2      ; =>NO, STILL IN WINDOW
CC40:          267 *
CC40:          268 * PIN CV TO WINDOW TOP:
CC40:          269 *
CC40:          CC40 270 X.US1      EGU *
CC40: EE FB 05   271          INC OURCV      ; PUT BACK WHERE IT WAS
CC43: F0 03      CC48 272          BEG X.USRET    ; IT GOES TO 0 ALWAYS
CC45:          CC45 273 X.US2      EGU *
CC45: 20 51 CB   274          JSR BASCALC    ; RECOMPUTE BASE ADDR
CC48:          CC48 275 X.USRET     EGU *
CC48: 60          276          RTS
CC49:          277 *
CC49:          278 * EXECUTE "NORMAL VIDED"
CC49:          279 *
CC49:          CC49 280 X.S0        EGU *
CC49: AD FB 04   281          LDA MODE       ; SET MODE BIT
CC4C: 29 FB      282          AND #255-M.VMODE ; SET 'NORMAL'
CC4E: A0 FF      283          LDY #255
CC50: D0 07      CC59 284          BNE STUFFINV   ; (ALWAYS)
CC52:          285 *
CC52:          286 * EXECUTE "INVERSE VIDED"
CC52:          287 *
CC52:          CC52 288 X.SI        EGU *
CC52: AD FB 04   289          LDA MODE       ; SET MODE BIT
CC55: 09 04      290          ORA #M.VMODE   ; SET 'INVERSE'
CC57: A0 7F      291          LDY #127
CC59:          CC59 292 STUFFINV    EGU *
CC59: BD FB 04   293          STA MODE       ; SET MODE
CC5C: 84 32      294          STY INVFLG     ; STUFF FLAG TOO
CC5E: 60          295          RTS
CC5F:          CC5F 297 CTLADL     EGU *
CC5F: BB          298          DFB >X.BELL-1 ; BEL
CC60: DA          299          DFB >X.BS-1    ; BS
CC61: 00          300          DFB 0          ; HT
CC62: 90          301          DFB >X.LF-1    ; LF
CC63: 22          302          DFB >X.VT-1    ; VT
CC64: 41          303          DFB >X.FF-1    ; FF
CC65: EB          304          DFB >X.CR-1    ; CR

```

```

CC66: 4B      305      DFB >X. SO-1      ; SO
CC67: 51      306      DFB >X. SI-1      ; SI
CC68: 00      307      DFB 0              ; DLE
CC69: 9B      308      DFB >X. DC1-1     ; DC1
CC6A: 76      309      DFB >X. DC2-1     ; DC2
CC6B: 00      310      DFB 0              ; DC3
CC6C: 00      311      DFB 0              ; DC4
CC6D: 8F      312      DFB >X. NAK-1     ; NAK
CC6E: A9      313      DFB >SCROLLDN-1   ; SYN
CC6F: A3      314      DFB >SCROLLUP-1   ; ETB
CC70: 00      315      DFB 0              ; CAN
CC71: 0C      316      DFB >X. EM-1      ; EM
CC72: 19      317      DFB >X. SUB-1     ; SUB
CC73: 00      318      DFB 0              ; ESC
CC74: 25      319      DFB >X. FS-1      ; FS
CC75: 47      320      DFB >X. GS-1      ; GS
CC76: 00      321      DFB 0              ; RS
CC77: 33      322      DFB >X. US-1      ; US
CC78:         323      *
CC78:         CC78 324 CTLADH      EGU *
CC78: CB      325      DFB <X. BELL-1    ; BEL
CC79: CB      326      DFB <X. BS-1      ; BS
CC7A: 00      327      DFB 0              ; HT
CC7B: CC      328      DFB <X. LF-1      ; LF
CC7C: CD      329      DFB <X. VT-1      ; VT
CC7D: CD      330      DFB <X. FF-1      ; FF
CC7E: CB      331      DFB <X. CR-1      ; CR
CC7F: CC      332      DFB <X. SO-1      ; SO
CC80: CC      333      DFB <X. SI-1      ; SI
CC81: 00      334      DFB 0              ; DLE
CC82: CD      335      DFB <X. DC1-1    ; DC1
CC83: CD      336      DFB <X. DC2-1    ; DC2
CC84: 00      337      DFB 0              ; DC3
CC85: 00      338      DFB 0              ; DC4
CC86: CD      339      DFB <X. NAK-1    ; NAK
CC87: CC      340      DFB <SCROLLDN-1 ; SYN
CC88: CC      341      DFB <SCROLLUP-1 ; ETB
CC89: 00      342      DFB 0              ; CAN
CC8A: CC      343      DFB <X. EM-1      ; EM
CC8B: CC      344      DFB <X. SUB-1     ; SUB
CC8C: 00      345      DFB 0              ; ESC
CC8D: CC      346      DFB <X. FS-1      ; FS
CC8E: CD      347      DFB <X. GS-1      ; GS
CC8F: 00      348      DFB 0              ; RS
CC90: CC      349      DFB <X. US-1      ; US
CC91:         13      INCLUDE SUBS2
CC91:         2      *
CC91:         3      * EXECUTE LINEFEED:
CC91:         4      *
CC91:         CC91 5 X. LF      EGU *
CC91: EE FB 05 6          INC   DURCV      ; BUMP CV
CC91: AD FB 05 7          LDA   DURCV      ; SEE IF OFF BOTTOM
CC91: C5 23 8          CMP   WNDBTM     ; OFF THE END?
CC91: B0 03 CC9E 9        BCS   X.LF2      ;=>YES
CC91: 4C 20 CD 10       JMP   X.LFRET    ;=>NO, DONE
CC91:         CC9E 11 X.LF2     EGU *
CC91: A4 23 12       LDY   WNDBTM     ; SET TO
CAA0: 88 13       DEY
CAA1: 8C FB 05 14      STY   DURCV      ; THE BOTTOM
CAA4:         15      *
CAA4:         16      * SCROLL THE SCREEN:
CAA4:         17      *
CAA4:         CAA4 18 SCROLLUP   EGU *
CAA4: BA 19       TXA
CAA4: 48 20       PHA
CAA4: A2 01 21       LDX   #1
CAA4: D0 04 CAAE 22     BNE   SCROLL1   ; DIRECTION=UP
CAA4: BA CAA 23 SCROLLDN EGU *
CAA4: 8A 24       TXA
CAA4: 48 25       PHA
CAA4: A2 00 26       LDX   #0
CAA4:         27      *
CAA4:         CAAE 28 SCROLL1   EGU *
CAA4: 2C 1F C0 29       BIT   RDBOVID   ; WHICH MODE?
CCB1: 10 05 CCB8 30     BPL   SCROLL2   ;=>40 DO WITH EXISTING WIDTH
CCB3: A5 21 31       LDA   WNDWDTH   ; TEMPORARILY SAVE
CCB5: 48 32       PHA
CCB6: 46 21 33       LSR   WNDWDTH   ; THE WIDTH AND
CCB8:         34      *
CCB8:         CCB8 35 SCROLL2   EGU *
CCB8: 20 D1 CC 36       JSR   SCRLSUB   ; SCROLL 40 COLS
CCB8: 2C 1F C0 37       BIT   RDBOVID   ; ARE WE IN BO-MODE?
CCB8: 10 51 CD11 38     BPL   X.SCRLRET ;=>NO, DONE
CCCO:         39      *
CCCO:         40      * FOR B0, DO THE OTHER PAGE...
CCCO:         41      *
CCCO: 08 42       PHP
CCCO: 1: 78 43       SEI
CCCO: 2: AD 55 C0 44     LDA   TXTPAGE2  ; WHILE TXTPAGE2 MAPPED IN
CCCO: 5: 20 D1 CC 45     JSR   SCRLSUB   ; SET PAGE2
CCCO: 8: AD 54 C0 46     LDA   TXTPAGE1  ; SCROLL PAGE 2
CCCO: B: 28 47     PLP
CCCO:         ; RESTORE PAGE1
CCCO:         ; RESTORE IRQ STATE NOW

```

```

CCCC: 6B          48          PLA
CCCD: 85 21      49          STA WNDWDTH
CCCF: DO 40     CD11 50          BNE X.SCRLRET ;=>DONE SCROLLBO (ALWAYS TAKEN)
CD1:             51 *
CD1:             52 * 40-COLUMN WINDOWED SCROLL:
CD1:             53 *
CD1:             54 SCRLSUB EQU *
CD1: BC F9 CF    55          LDY WNDTAB, X ;GET WINDOW TOP/BOT
CCD4: B9 00 00   56          LDA O, Y
CD7: E0 01       57          CPX #1 ;SCROLLING UP?
CCD9: B0 02     CCDD 58          BCS MSCRLO ;=>YES, NO PROBLEM
CCDB: E9 00      59          SBC #0 ;-1 IF DOWN (SRC=BTM-1)
CCDD:             60 MSCRLO EQU *
CCDD: 48         61          PHA
CCDE: 20 54 CB   62          JSR BASCALCZ
CCE1: A5 2B      63 MSCRL1 LDA BASL
CCE3: 85 2A      64          STA BAS2L
CCE5: A5 29      65          LDA BASH
CCE7: B5 2B      66          STA BAS2H
CCE9: A4 21      67          LDY WNDWDTH
CCEB: 88         68          DEY
CCEC: 68         69          PLA
CCED: 1B         70          CLC
CCEE: 7D F0 CF   71          ADC PLUSMINUS1, X ;UP/DOWN
CCF1: D5 22      72          CMP WNDTOP, X ;AT THE END?
CCF3: F0 0D     CD02 73          BEQ MSCRLRET
CCF5: 4B         74          PHA
CCF6: 20 54 CB   75          JSR BASCALCZ
CCF9: B1 2B      76 MSCRL2 LDA (BASL), Y
CCFB: 91 2A      77          STA (BAS2L), Y
CCFD: 8B         78          DEY
CCFE: 10 F9     CCF9 79          BPL MSCRL2
CD00: 30 DF     CCE1 80          BMI MSCRL1
CD02:             81 *
CD02:             82 MSCRLRET EQU *
CD02: E0 00      83          CPX #0 ;SCROLLING DOWN?
CD04: D0 0A     CD10 84          BNE MSCRLRTS ;=>NO
CD06: 20 54 CB   85          JSR BASCALCZ
CD09:             86 DNEMORE EQU *
CD09: B1 2B      87          LDA (BASL), Y
CD0B: 91 2A      88          STA (BAS2L), Y
CD0D: 8B         89          DEY
CD0E: 10 F9     CD09 90          BPL DNEMORE
CD10:             91 MSCRLRTS EQU *
CD10: 60         92          RTS
CD11:             93 *
CD11:             94 * DONE WITH THE SCROLLING JAZZ:
CD11:             95 *
CD11:             96 X.SCRLRET EQU *
CD11: B4 22     CD11 97          LDY WNDTOP, X ;CLEAR TOP OR BOTTOM LINE
CD13: 8A         98          TXA ;IF GETTING TOP,
CD14: F0 01     CD17 99          BEQ X.SCRLRET2 ;DON'T DECREMENT!
CD16: 8B         100         DEY
CD17:             CD17 101 X.SCRLRET2 EQU *
CD17: 9B         102         TYA ;TEMP CV SETUP
CD18: 20 54 CB   103         JSR BASCALCZ ;COMPUTE BASE OF LINE TO CLEAR
CD18: 6B         104         PLA ;RESTORE
CD1C: AA         105         TAX ;X
CD1D: 20 1A CC   106         JSR X.SUB ;CLEAR BOTTOM LINE
CD20:             107 *
CD20:             108 X.LFRET EQU *
CD20: 4C 51 CB   109         JMP BASCALC ;RETURN VIA BASCALC (UGH!)
CD23:             110 *
CD23:             111 * EXECUTE CLR TO EOS:
CD23:             112 *
CD23:             113 X.VT EQU *
CD23: 20 48 CD   114         JSR X.OS ;CLEAR TO EOL
CD26: AD FB 05   115         LDA DURCV ;SAVE CV
CD29: 4B         116         PHA
CD2A: 10 06     CD32 117         BPL EQU *
CD2C:             CD2C 118 X.VTLOOP EQU *
CD2C: 20 51 CB   119         JSR BASCALC ;BASCALC IT
CD2F: 20 1A CC   120         JSR X.SUB ;CLEAR LINE
CD32:             CD32 121 X.VTNEXT EQU *
CD32: EE FB 05   122         INC DURCV ;BUMP CV
CD35: AD FB 05   123         LDA DURCV
CD38: C5 23     124         CMP WNDBTM ;OFF SCREEN?
CD3A: 90 F0     CD2C 125         BCC X.VTLOOP ;=>NO, KEEP GOING
CD3C: 6B         126         PLA ;RESTORE
CD3D: BD FB 05   127         STA DURCV ;CV
CD40: 10 DE     CD20 128         BPL X.LFRET ;RETURN VIA SIMILAR CODE
CD42:             129 *
CD42:             130 * EXECUTE CLEAR:
CD42:             131 *
CD42:             132 X.FF EQU *
CD42: 20 0D CC   133         JSR X.EM ;HOME THE CURSOR
CD45: 4C 23 CD   134         JMP X.VT ;RETURN VIA CLREDS (UGH!)
CD4B:             135 *
CD4B:             136 * EXECUTE CLEAR TO EOL:
CD4B:             137 *
CD4B:             138 X.OS EQU *

```



```

CD4B: AC 7B 05 139 LDY DURCH ;GET CH
CD4B: 4C 34 CD 140 JMP X.GS2 ;CHECK FOR END FIRST!
CD4E CD4E 141 X.GSEOLZ EQU * ;FER U HACKERS
CD4E A9 A0 142 LDA #'
CD50 20 F2 CE 143 JSR STORCHAR ;STUFF IT
CD53 CB 144 INY *
CD54 C4 21 CD54 145 X.GS2 EQU *
CD56 90 F6 CD4E 146 WNDWDTH ;STOP SOMETIME
CD5B 60 147 BCC X.GSEOLZ ;YASL DO MORE
CD59: 148 RTS
CD59: 149 *
CD59: 150 * EXECUTE '40COL MODE':
CD59: 151 *
CD59 CD59 152 X.DC1 EQU *
CD59 A9 00 153 LDA #0 ;ASSUME TEXTMODE
CD5B 85 20 154 STA WNDLFT
CD5D 2C 1A CO 155 BIT RDTEXT ;ARE WE IN TEXT MODE?
CD60 30 02 CD64 156 BMI X.DC1B ;=>YES
CD62 A9 14 157 LDA #20 ;IF GR, SET SPLITSCREEN
CD64: CD64 158 X.DC1B EQU *
CD64: 85 22 159 STA WNDTOP
CD66 A9 18 160 LDA #24
CD68 85 23 161 STA WNDBTM
CD6A A9 28 162 LDA #40
CD6C 85 21 163 STA WNDWDTH
CD6E 2C 1F CO 164 BIT RDBOVID ;WERE WE IN 80-MODE?
CD71 10 03 CD76 165 BPL X.DC1RTS ;=>NO, NO CVT NEEDED
CD73 20 DB CD 166 JSR SCRNB4 ;CVT 80-->240
CD76: CD76 167 X.DC1RTS EQU *
CD76: 60 168 RTS
CD77: 169 *
CD77: 170 * EXECUTE '80COL MODE':
CD77: 171 *
CD77: CD77 172 X.DC2 EQU *
CD77: 20 24 CB 173 JSR TESTCARD ;IS CARD THERE?
CD7A: D0 1E CD9A 174 BNE X.DC2RET ;=>NOPE, FORGET IT
CD7C 20 9B CD 175 JSR FULL80 ;SET FULL WINDOW
CD7F 2C 1A CO 176 BIT RDTEXT ;ARE WE IN TEXT MODE?
CD82 30 04 CD8B 177 BMI X.DC2B ;=>YES
CD84 A9 14 178 LDA #20 ;IF GR, SET SPLITSCREEN
CD86 85 22 179 STA WNDTOP
CD8B: CD8B 180 X.DC2B EQU *
CD8B: 2C 18 CO 181 BIT RDBOCOL ;REMEMBER PRIOR MODE
CD8B: 30 0D CD9A 182 BMI X.DC2RET ;=>NO CVT NEEDED IF WAS 80
CD8D 4C 32 CE 183 JMP SCRNB4 ;RET VIA CONVERT 40-->80
CD90: 184 *
CD90: 185 * EXECUTE 'QUIT':
CD90: 186 *
CD90: CD90 187 X.NAK EQU *
CD90: AD FB 04 188 LDA MODE ;ONLY VALID IN BASIC
CD93 29 20 189 AND #M.PASCAL
CD95 D0 03 CD9A 190 BNE X.NAKRET ;IGNORE IF PASCAL
CD97 20 AA CD 191 JSR QUIT ;GET SETUP TO QUIT
CD9A: CD9A 192 X.NAKRET EQU *
CD9A: 60 CD9A 193 X.DC2RET EQU *
CD9A: 194 RTS ;DONE! CALLER WON'T RETURN
CD9B: 195 -----
CD9B: 196 * NAME : FULL80
CD9B: 197 * FUNCTION: SET FULL 80COL WINDOW
CD9B: 198 * INPUT : NONE
CD9B: 199 * OUTPUT : WINDOW PARAMETERS
CD9B: 200 * VOLATILE: AC
CD9B: 201 -----
CD9B: 202 *
CD9B: CD9B 203 FULL80 EQU *
CD9B A9 00 204 LDA #0
CD9D 85 22 205 STA WNDTOP
CD9F 85 20 206 STA WNDLFT
CDA1 A9 50 207 LDA #80
CDA3 85 21 208 STA WNDWDTH
CDA5 A9 18 209 LDA #24
CDA7 85 23 210 STA WNDBTM
CDA9 60 211 RTS
CDA: 212 -----
CDA: 213 * NAME : QUIT
CDA: 214 * FUNCTION: SETUP TO QUIT THE CARD
CDA: 215 * INPUT : NOTHING
CDA: 216 * OUTPUT : NOTHING
CDA: 217 * VOLATILE: ALL REGS
CDA: 218 * CALLS : X.FF, FULL80, BASCALC
CDA: 219 * : SETKBD, SETVID
CDA: 220 -----
CDA: 221 *
CDA: CDAA 222 QUIT EQU *
CDA: A9 00 223 LDA #0 ;SET FULL 40-COL WINDOW
CDAC 85 22 224 STA WNDTOP
CDAE 85 20 225 STA WNDLFT
CDB0 A9 18 226 LDA #24
CDB2 85 23 227 STA WNDBTM
CDB4 A9 28 228 LDA #40
CDB6 85 21 229 STA WNDWDTH

```

```

CDBB: 2C 1F C0      230      BIT   RDBOVID      ;WHAT WIDTH?
CDBB: 10 03 CDC0    231      BPL   QUIT2      ;=>NO CVT NEEDED IF 40
CDBB: 20 DB CD      232      JSR   SCRNB4     ; CONVERT 40-->80
CDBB: CDC0:                233      *
CDBB: CDC0:                234      * QUIT2      EQU   *
CDBB: CDC0: A9 17    235      LDA   #23        ;VTAB TO THE
CDBB: CDC2: 8D FB 05 236      STA   OURCV      ; BOTTOM LINE
CDBB: CDC9: 20 51 CB 237      JSR   BASCALC
CDBB: CDC8: A9 00    238      LDA   #0         ; AND PLACE CURSOR
CDBB: CDCA: 8D 7B 05 239      STA   OURCH      ; AT LEFT SIDE
CDBB: CDCD: 8D 0E C0 240      STA   CLRALTCHAR ;LCASE CHARS OFF
CDBB: CDD0: A9 FF    241      LDA   #FF        ; DESTROY THE
CDBB: CDD2: 8D FB 04 242      STA   MODE       ; MODE BYTE
CDBB: CDD5: 20 93 FE 243      JSR   SETVID     ; PR#0
CDBB: CDD8: 4C 89 FE 244      JMP   SETKBD     ; RETURN VIA IN#0 (UGH!)
CDBB: CDD8:                245      -----
CDBB: CDD8:                246      * NAME      : SCRNB4
CDBB: CDD8:                247      * FUNCTION:  : CONVERT BOVID-->40VID
CDBB: CDD8:                248      * INPUT      : NONE
CDBB: CDD8:                249      * OUTPUT     : NONE
CDBB: CDD8:                250      * VOLATILE:  : ALL REGISTERS
CDBB: CDD8:                251      * NOTE      : USES 'BAS2H/L' AS TEMPS
CDBB: CDD8:                252      -----
CDBB: CDD8:                253      *
CDBB: CDD8:                254      * SCRNB4     EQU   *
CDBB: CDD8: AD FB 05 255      LDA   OURCV      ; SAVE CURRENT
CDBB: CDE: 48        256      PHA
CDBB: CDF: AD 7B 05 257      LDA   OURCH      ; SETTINGS
CDBB: CDE2: 48      258      PHA
CDBB: CDE3:                259      *
CDBB: CDE3: A9 17    260      LDA   #23
CDBB: CDE5: B5 2A    261      STA   BAS2L      ; USE AS A TEMP
CDBB: CDE7: 8D 01 C0 262      STA   SETBOCOL
CDBB: CDEA: A5 2A    263      LDA   BAS2L
CDBB: CDEC: 20 54 CB 264      JSR   BASCALCZ   ; BEGIN AT BOTTOM AND WORK UP
CDBB: CDEF: 20 0A CE 265      JSR   ATEFOR     ; DO THIS LINE
CDBB: CDF2: C6 2A    266      DEC   BAS2L
CDBB: CDF4: 30 0B CE01 267      BMI   SCR40RET   ;=>DONE (HIT TOP)
CDBB: CDF6: 2C 1A C0 268      BIT   RDTEXT     ; ARE WE IN MIDEXMODE?
CDBB: CDF9: 30 EF CDEA 269      BMI   SCR40     ;=>NO, DO ENTIRE SCREEN
CDBB: CDFB: A5 2A    270      LDA   BAS2L     ; IF SO, ONLY DO BOTTOM
CDBB: CDFD: C9 14    271      CMP   #20       ; FOUR (4) LINES OF WINDOW
CDBB: CDFE: B0 E9 CDEA 272      BCS   SCR40
CDBB: CE01:                273      * SCR40RET
CDBB: CE01: BD 00 C0 274      STA   CLRBOCOL
CDBB: CE04: BD 0C C0 275      STA   CLRBOVID
CDBB: CE07: 4C 58 CE 276      JMP   SCRNRRET   ; RETURN VIA SIMILAR CODE
CDBB: CE0A:                277      *
CDBB: CE0A: CE0A:    278      * ATEFOR     EQU   *
CDBB: CE0A: 08        279      PHP
CDBB: CE0B: 78        280      SETI
CDBB: CE0C: A0 28    281      LDY   #40
CDBB: CE0E: B4 28    282      STY   BAS2H
CDBB: CE10: 2C 54 C0 283      BIT   TXTPAGE1
CDBB: CE13: 20 22 CE 284      JSR   GETB4
CDBB: CE16: 2C 55 C0 285      BIT   TXTPAGE2
CDBB: CE19: 20 22 CE 286      JSR   GETB4
CDBB: CE1C: A4 2B    287      LDY   BAS2H     ; DONE?
CDBB: CE1E: D0 F3 CE13 288      BNE   ATEFOR1   ;=>NO, DO WHOLE LINE
CDBB: CE20: 28        289      PLP
CDBB: CE21: 60        290      RTS
CDBB: CE22:                291      *
CDBB: CE22: C6 2B    292      * GETB4     DEC   BAS2H
CDBB: CE24: A5 2B    293      LDA   BAS2H
CDBB: CE26: 4A        294      LSR   A
CDBB: CE27: A8        295      TAY
CDBB: CE28: B1 28    296      LDA   (BASL),Y
CDBB: CE2A: A4 28    297      LDY   BAS2H
CDBB: CE2C: 2C 54 C0 298      BIT   TXTPAGE1
CDBB: CE2F: 91 28    299      STA   (BASL),Y
CDBB: CE31: 60        300      RTS
CDBB: CE32:                301      -----
CDBB: CE32:                302      * NAME      : SCRNB4
CDBB: CE32:                303      * FUNCTION:  : CONVERT 40VID-->BOVID
CDBB: CE32:                304      * INPUT      : NONE
CDBB: CE32:                305      * OUTPUT     : NONE
CDBB: CE32:                306      * VOLATILE:  : ALL REGISTERS
CDBB: CE32:                307      * NOTE      : USES 'BAS2H/L' AS TEMPS
CDBB: CE32:                308      -----
CDBB: CE32:                309      *
CDBB: CE32:                310      * SCRNB4     EQU   *
CDBB: CE32: AD FB 05 311      LDA   OURCV      ; SAVE CV
CDBB: CE35: 48        312      PHA
CDBB: CE36: AD 7B 05 313      LDA   OURCH      ; AND CH
CDBB: CE39: 48        314      PHA
CDBB: CE3A:                315      *
CDBB: CE3A: A9 17    316      LDA   #23
CDBB: CE3C: B5 2A    317      STA   BAS2L      ; USE AS A TEMP
CDBB: CE3E: A5 2A    318      LDA   BAS2L
CDBB: CE40: 20 54 CB 319      JSR   BASCALCZ   ; BEGIN AT BOTTOM AND WORK UP
CDBB: CE43: 20 43 CE 320      JSR   FORATE     ; DO THIS LINE
CDBB: CE46: C6 2A    321      DEC   BAS2L

```

```

CE48: 30 0B CE55 322 BMI SCRBORET ;=>DONE (HIT TOP)
CE4A: 2C 1A CO 323 BIT RDTEXT ;ARE WE IN MIXEDMODE?
CE4D: 30 EF CE3E 324 BMI SCRBO ;NO, DO FULL SCREEN
CE4F: A5 2A 325 LDA BAS2L ;IF 80, ONLY DO BOTTOM
CE51: C9 14 326 CMP #20 ; FOUR (4) LINES OF WINDOW
CE53: B0 E9 CE3E 327 BCS SCRBO
CE55: CE55 329 *
CE55: CE55 329 SCRBORET EQU *
CE55: BD OD CO 330 STA SETBOVID ;DISPLAY IN 80-MODE
CE58: CE58 331 SCRNRRET EQU * ;USED BY SCRNB4
CE58: 68 332 PLA ;RESTORE
CE59: 8D 7B 05 333 STA DURCH ; CH AND
CE5C: 68 334 PLA ; CV
CE5D: BD FB 05 335 STA DURCV
CE60: 4C 51 CB 336 JMP BASCALC ;RETURN VIA BASCALC (UGH!)
CE63: CE63 337 *
CE63: CE63 338 *
CE63: CE63 339 FORATE EQU *
CE63: 08 340 PHF ;DON'T ALLOW IRQ WHILE
CE64: 78 341 SEI ; SCREENHOLES ARE WRONG
CE65: A0 00 342 LDY #0
CE67: 84 2B 343 STY BAS2H
CE69: BC 01 CO 344 STY SETBOCOL
CE6C: 2C 54 CO 345 BIT TXTPAGE1
CE6F: B1 2B 346 LDA (BASL),Y FORATE1
CE71: 2C 55 CO 347 BIT TXTPAGE2
CE74: 20 A3 CE 348 JSR D048
CE77: 2C 54 CO 349 BIT TXTPAGE1
CE7A: B1 2B 350 LDA (BASL),Y
CE7C: 20 A3 CE 351 JSR D048
CE7F: C0 2B 352 CPY #40
CEB1: 90 EC CE6F 353 BCC FORATE1
CEB3: CEB3 354 *
CEB3: 20 91 CE 355 JSR CLRHALF ;CLEAR RIGHT HALF
CEB4: 2C 55 CO 356 BIT TXTPAGE2 ; OF BOTH PAGES
CEB9: 20 91 CE 357 JSR CLRHALF
CEBC: 2C 54 CO 358 BIT TXTPAGE1
CEBF: 28 359 PLP ;OK TO ALLOW IRQ NOW
CE90: 60 360 RTS
CE91: CE91 361 *
CE91: A0 14 CE91 362 CLRHALF EQU *
CE93: A9 A0 363 LDY #20
CE95: 24 32 CE95 365 BIT INVFLG ;
CE97: 30 02 CE9B 366 BMI CLRHALF2 ;WHICH MODE?
CE99: 29 7F CE9B 367 AND #57F ;=NORMAL
CE98: CE98 368 CLRHALF2 EQU * ;INVERSE
CE98: 91 2B CE98 369 STA (BASL),Y ;STUFF THE BLANK
CE9D: C8 370 INY
CE9E: C0 2B CE98 371 CPY #40
CEA0: D0 F9 CE9B 372 BNE CLRHALF2
CEA2: 60 373 RTS
CEA3: CEA3 374 *
CEA3: 48 375 DO48 PHA
CEA4: 98 376 TYA
CEA5: 4A 377 LSR A
CEA6: A8 378 TAY
CEA7: 68 379 PLA
CEA8: 91 2B 380 STA (BASL),Y
CEAA: E6 2B 381 INC BAS2H
CEAC: A4 2B 382 LDY BAS2H
CEAE: 60 383 RTS
CEAF: CEAF 14 INCLUDE SUBS3
CEAF: CEAF 2 -----
CEAF: CEAF 3 * NAME : SETCH
CEAF: CEAF 4 * FUNCTION: SET DURCH AND CH
CEAF: CEAF 5 * INPUT : AC=CH VALUE
CEAF: CEAF 6 * OUTPUT : DURCH, CH MOD 40
CEAF: CEAF 7 * VOLATILE: NOTHING
CEAF: CEAF 8 * CALLS : NOTHING
CEAF: CEAF 9 -----
CEAF: CEAF 10 *
CEAF: CEAF 11 SETCH EQU *
CEAF: BD 7B 05 CEAF 12 STA DURCH ;STUFF DURCH
CEB2: 85 24 13 STA CH ;STUFF IN CASE WE'RE 40 MODE
CEB4: BD 7B 04 14 STA OLDCH
CEB7: 2C 1F CO 15 BIT RDBOVID ;IN 80-MODE?
CEBA: 10 1D CED9 16 BPL SETCHRTS ;=>NO, DONE
CEBC: CEBC 17 *
CEBC: CEBC 18 * IF WE'RE NEAR THE END OF OUR
CEBC: CEBC 19 * 80COL LINE, MOVE CH UP, IF NOT,
CEBC: CEBC 20 * LEAVE CH PINNED AT ZERO...
CEBC: CEBC 21 *
CEBC: A9 00 22 LDA #0 ;PIN CH AT ZERO
CEBE: 85 24 23 STA CH
CECO: BD 7B 04 24 STA OLDCH ;REMEMBER THE SETTING
CEC3: A5 21 25 LDA WNDWIDTH ;CHECK IF NEAR THE END
CEC5: 38 26 SEC
CEC6: ED 7B 05 27 SBC DURCH ;GET ABS CH
CEC9: C9 0B 28 CMP #8 ;NEAR THE END?
CECB: B0 0C CED9 29 BCS SETCHRTS ;=NOPE
CECD: 83 24 30 STA CH ;YES: MOVE CH UP NEAR RIGHT

```

```

CECF: A9 2B      31      LDA #40
CEB1: 3B        32      SEC
CED2: E5 24     33      SBC CH
CED4: 85 24     34      STA CH ; BASIC WILL SEE THAT NOW
CED6: BD 7B 04  35      STA QLDCH ; REMEMBER THE SETTING
CED9:           36 *
CED9:           CED9 37  SETCHRIS EQU *
CED9: AD 7B 05  38      LDA OURCH ; RESTORE AC
CEDC: 60        39      RTS
CEDD:           40 -----
CEDD:           41 * NAME : INVERT
CEDD:           42 * FUNCTION: INVERT CHAR AT CH/CV
CEDD:           43 * INPUT : NOTHING
CEDD:           44 * OUTPUT : CHAR AT CH/CV INVERTED
CEDD:           45 * VOLATILE: NOTHING
CEDD:           46 * CALLS : PICK, STORCHAR
CEDD:           47 -----
CEDD:           48 *
CEDD:           CEDD 49  INVERT EQU *
CEDD: 4B        50      PHA ; SAVE AC
CEDE: 9B        51      TYA ; AND Y
CEF0: 4B        52      PHA
CEE0: AC 7B 05  53      LDY OURCH ; GET CH
CEE3: 20 01 CF  54      JSR PICK ; GET CHARACTER
CEE6: 49 80     55      EOR ##80 ; FLIP INVERSE/NORMAL
CEE8: 2C 00 CF  56      BIT SEV ; PUT DIRECTLY BACK
CEEB: 20 06 CF  57      JSR SCREENIT ; ONTO SCREEN
CEEE: 68        58      PLA ; RESTORE Y
CEEF: AB        59      TAY ; AND AC
CEFO: 68        60      PLA
CEF1: 60        61      RTS
CEF2:           62 -----
CEF2:           63 * NAME : STORCHAR
CEF2:           64 * FUNCTION: STORE A CHAR ON SCREEN
CEF2:           65 * INPUT : AC=CHAR
CEF2:           66 * : Y=CH POSITION
CEF2:           67 * OUTPUT : CHAR ON SCREEN
CEF2:           68 * VOLATILE: NOTHING
CEF2:           69 * CALLS : SCREENIT
CEF2:           70 -----
CEF2:           71 *
CEF2:           CF2 72  STORCHAR EQU *
CEF2: 4B        73      PHA ; SAVE AC
CEF3: 24 32     74      BIT INVFLG ; NORMAL OR INVERSE?
CEF5: 30 02     CF9 75      BMI STOR2 ; =>NORMAL
CEF7: 49 80     76      EOR ##80 ; INVERSE
CEF9:           CF9 77  STOR2 EQU *
CEF9: 2C 00 CF  78      BIT SEV ; V SET FOR STORE
CEFC: 20 06 CF  79      JSR SCREENIT ; =>DO IT!
CEFF: 68        80      PLA ; RESTORE AC
CF00: 60        81      SEV
CF01:           82 -----
CF01:           83 * NAME : PICK
CF01:           84 * FUNCTION: GET A CHAR FROM SCREEN
CF01:           85 * INPUT : Y=CH POSITION
CF01:           86 * OUTPUT : AC=CHARACTER
CF01:           87 * VOLATILE: NOTHING
CF01:           88 * CALLS : SCREENIT
CF01:           89 -----
CF01:           90 *
CF01:           CF01 91  PICK EQU *
CF01: B8        92      CLV ; V CLEAR FOR PICK
CF02: 20 06 CF  93      JSR SCREENIT ; DO IT!
CF03: 60        94      RTS
CF06:           95 -----
CF06:           96 * NAME : SCREENIT
CF06:           97 * FUNCTION: STORE OR PICK CHAR
CF06:           98 * INPUT : V CLR FOR PICK
CF06:           99 * : V SET FOR STORE
CF06:           100 * : AC=CHAR FOR STORE
CF06:           101 * : Y=CH POSITION
CF06:           102 * OUTPUT : AC=CHAR (PICK)
CF06:           103 * VOLATILE: NOTHING
CF06:           104 * CALLS : NOTHING
CF06:           105 -----
CF06:           106 *
CF06:           CF06 107 SCREENIT EQU *
CF06: B4 1F     108      STY YSAV1 ; SAVE Y
CF08: 4B        109      PHA ; SAVE CHARACTER IF STORING
CF09:           110 * AVOID CHANGING VFLAG VIA BIT!
CF09: AD 1F C0  111      LDA RDBOVID ; WHAT DISPLAY MODE?
CF0C: 10 32     CF40 112      BPL SCR40 ; =>40-COL MODE
CF0E:           113 *
CF0E:           114 * 80-COLUMN MODE:
CF0E:           115 *
CF0E: A5 1F     116      LDA YSAV1 ; GET CURSOR HORIZ
CF10: 4A        117      LSR A ; DIVIDE BY TWO FOR PAGE
CF11: AB        118      TAY ; CH TO YREG
CF12: 70 16     CF2A 119      BVS STORBO ; =>GONNA STORE THE CHAR
CF14:           120 *
CF14:           121 * 80-COL PICK:

```

```

CF14:          122 *
CF14: 0B      123      PHP          ; LOCK INTERRUPTS WHILE
CF15: 7B      124      SET          ; SCREENHOLES ARE WRONG
CF16: AD 55 CO 125      LDA          TXTPAGE2 ; ASSUME PAGE 2 (EVENS)
CF19: 90 03   CF1E   126      BCC      SCR2N  ; =>IT IS
CF1B: AD 54 CO 127      LDA          TXTPAGE1 ; ODDS GO TO PAGE1
CF1E:          128      EQU          *
CF1E: B1 2B   CF1E   129      LDA          (BASL),Y ; PICK THE CHARACTER
CF20: AB      130      TAY          ; HOLD CHAR TEMPORARILY
CF21: AD 54 CO 131      LDA          TXTPAGE1 ; RESTORE PAGE1
CF24: 2B      132      PLP          ; AND ALLOW IRQ AGAIN
CF25: 6B      133      PLA          ; TRASH SAVED AC
CF26: 9B      134      TYA          ;
CF27: 4B      135      PHA          ; MAKE CHAR GET RESTORED TO AC
CF2B: 50 24   CF4E   136      BVC      STPKEXIT ; =>DONE (ALWAYS TAKEN)
CF2A:          137 *
CF2A:          138      STORB0 EQU      *
CF2A: 6B      139      PLA          ; RESTORE CHARACTER
CF2B: 4B      140      PHA          ; (LEAVE ON STACK)
CF2C: 0B      141      PHP          ; LOCK INTERRUPTS WHILE
CF2D: 7B      142      SET          ; THE SCREENHOLES ARE WRONG
CF2E: 4B      143      PHA          ; HOLD THE CHAR TEMPORARILY
CF2F: AD 55 CO 144      LDA          TXTPAGE2 ; ASSUME PAGE2 (EVENS)
CF32: 90 03   CF37   145      BCC      SCR3N  ; =>IT IS
CF34: AD 54 CO 146      LDA          TXTPAGE1 ; ODDS GO TO PAGE1
CF37:          147      EQU          *
CF37: 6B      148      PLA          ; GET CHAR TO BE STORED
CF3B: 91 2B   CF3A   149      STA          (BASL),Y ; STUFF ONTO SCREEN
CF3A: AD 54 CO 150      LDA          TXTPAGE1 ; RESTORE PAGE1
CF3D: 2B      151      PLP          ; AND ALLOW IRQ AGAIN
CF3E: 70 0E   CF4E   152      BVS      STPKEXIT ; =>DONE (ALWAYS TAKEN)
CF40:          153 *
CF40:          154 * 40-COLUMN MODE:
CF40:          155 *
CF40:          156      SCR40 EQU      *
CF40: A4 1F   CF40   157      LDY      YSAV1  ; GET CURSOR HORIZ
CF42: 70 06   CF4A   158      BVS      STOR40 ; =>STORE IT
CF44: 6B      159      PLA          ; TRASH SAVED CHAR
CF45: B1 2B   CF4E   160      LDA          (BASL),Y ; PICK THE CHARACTER
CF47: 4B      161      PHA          ; SAVE CHAR FOR RESTORE
CF4B: 50 04   CF4E   162      BVC      STPKEXIT ; DONE (ALWAYS TAKEN)
CF4A:          163 *
CF4A:          164      STOR40 EQU      *
CF4A: 6B      165      PLA          ; GET THE CHARACTER
CF4B: 4B      166      PHA          ; (LEAVE ON STACK)
CF4C: 91 2B   CF4E   167      STA          (BASL),Y ; STUFF ONTO SCREEN
CF4E:          168 *
CF4E:          169      STPKEXIT EQU      *
CF4E: 6B      170      PLA          ; RESTORE AC
CF4F: A4 1F   CF4E   171      LDY      YSAV1  ; RESTORE Y
CF51: 60      172      RTS
CF52:          173 -----
CF52:          174 * NAME      : ESCDN
CF52:          175 * FUNCTION:  TURN ON 'ESCAPE' CURSOR
CF52:          176 * INPUT      : NONE
CF52:          177 * OUTPUT     : 'CHAR'='ORIGINAL CHAR
CF52:          178 * VOLATELE:  NOTHING
CF52:          179 * CALLS      : PICK, STORCHAR
CF52:          180 -----
CF52:          181 *
CF52:          182      ESCDN EQU      *
CF52: 4B      183      PHA          ; SAVE AC
CF53: 9B      184      TYA          ; AND Y
CF54: 4B      185      PHA          ;
CF55: AC 7B 05 186      LDY      DURCH  ; GET CH
CF5B: 20 01 CF 187      JSR      PICK   ; GET ORIGINAL CHARACTER
CF5B: 8D 7B 06 188      STA      CHAR   ; AND REMEMBER FOR ESCOFF
CF5E: 29 80   CF5E   189      AND      ##B0    ; SAVE NORMAL/INVERSE BIT
CF60: 49 AB   CF5E   190      EOR      ##AB    ; MAKE IT AN INVERSE '+'
CF62: 4C 6E CF 191      JMP      ECRET   ; RETURN VIA SIMILAR CODE
CF65:          192 -----
CF65:          193 * NAME      : ESCOFF
CF65:          194 * FUNCTION:  TURN OFF 'ESCAPE' CURSOR
CF65:          195 * INPUT      : 'CHAR'='ORIGINAL CHAR
CF65:          196 * OUTPUT     : NONE
CF65:          197 * VOLATILE:  NOTHING
CF65:          198 * CALLS      : STORCHAR
CF65:          199 -----
CF65:          200 *
CF65:          201      ESCOFF EQU      *
CF65: 4B      202      PHA          ; SAVE AC
CF66: 9B      203      TYA          ; AND Y
CF67: 4B      204      PHA          ;
CF6B: AC 7B 05 205      LDY      DURCH  ; GET CH
CF6B: AD 7B 06 206      LDA      CHAR   ; GET ORIGINAL CHARACTER
CF6E:          207      EQU          *
CF6E: 2C 00 CF 208      BIT          ; USED BY ESCON
CF71: 20 06 CF 209      JSR      SEV     ; AND PUT IT BACK
CF74: 6B      210      PLA          ; EXACTLY AS IT WAS
CF75: AB      211      TYA          ; RESTORE Y
CF76: 6B      212      PLA          ; AND AC

```

```

CF77: 60          213          RTS
CF78:            214          -----
CF78:            215 * NAME      : COPYROM
CF78:            216 * FUNCTION: COPY FB ROM TO LCARD
CF78:            217 * INPUT    : NOTHING
CF78:            218 * VOLATILE: X, Y
CF78:            219 * CALLS    : NOTHING
CF78:            220          -----
CF78:            221 *
CF78:            CF78 222 COPYROM EQU *
CF78: 48          223 PHA          ; SAVE AC
CF79: 08          224 PHP          ; ENSURE IRQ INHIBITED
CF7A: 78          225 SEI          ; WHILE COPYING ROM
CF7B:            226 *
CF7B: AD 11 CO    227 LDA RDLCBNK2 ; GET BANK2
CF7E: 48          228 PHA
CF7F:            229 *
CF7F: AE 12 CO    230 LDX RDLGRAM ; AND RAM FLAGS
CF82: AD 81 CO    231 LDA %COB1  ; SET READ-ROM
CF85: AD 81 CO    232 LDA %COB1  ; WRITE-RAM MODE
CF8B:            233 *
CF8B: A0 00       234 LDY #0
CF8A: A9 FB       235 LDA ##FB
CF8C: B5 37       236 STA CSWH   ; USE HOOK FOR MOVE
CF8E: A5 36       237 LDA CSWL   ; PRESERVE LO BYTE
CF90: 48          238 PHA
CF91: A9 00       239 LDA #0
CF93: B5 36       240 STA CSWL
CF95:            CF95 241 COPYROM2 EQU *
CF95: B1 36       242 LDA (CSWL),Y ; COPY ONLY PATCHED PAGES
CF97: 91 36       243 STA (CSWL),Y ; MOVE THE ROM
CF99: C8          244 INY
CF9A: D0 F9       CF95 245 BNE COPYROM2
CF9C: E6 37       246 INC CSWH
CF9E: D0 F5       CF95 247 BNE COPYROM2
CFA0:            248 *
CFA0: 68          249 PLA          ; RESTORE THE
CFA1: B5 36       250 STA CSWL   ; HOOK
CFA3: A9 C3       251 LDA %CN00
CFA5: B5 37       252 STA CSWH
CFA7:            253 *
CFA7: 68          254 PLA          ; WHICH LC BANK?
CFA8: 10 0F       CF89 255 BPL LCB1   ; =>BANK1
CFAA: BA          256 TXA          ; RAM OR ROM READ?
CFAB: 10 06       CF83 257 BPL LCB2ROM ; =>ROM
CFAD: AD 80 CO    258 LDA %COB0  ; BANK2, RAM
CFB0: 4C C5 CF    259 JMP COPYRET ; COPYRET
CFB3: AD 81 CO    260 LDA %COB1  ; BANK2, ROM
CFB6: 4C C5 CF    261 JMP COPYRET ; COPYRET
CFB9: BA          262 LCB1 TXA    ; RAM OR ROM READ?
CFBA: 10 06       CFC2 263 BPL LCB1ROM ; =>ROM
CFBC: AD 88 CO    264 LDA %COB8  ; BANK1, RAM
CFBF: 4C C5 CF    265 JMP COPYRET ; COPYRET
CFC2: AD 89 CO    266 LCB1ROM LDA %COB9 ; BANK1, ROM
CFC5:            267 *
CFC5:            CFC5 268 COPYRET EQU *
CFC5: 28          269 PLP          ; RESTORE IRQ STATE NOW
CFC6: 68          270 PLA          ; AND AC
CFC7: 60          271 RTS
CFC8:            272          -----
CFC8:            273 * NAME      : PSETUP
CFC8:            274 * FUNCTION: SETUP ZP FOR PASCAL
CFC8:            275 * INPUT    : NONE
CFC8:            276 * OUTPUT   : NONE
CFC8:            277 * VOLATILE: AC
CFC8:            278 * CALLS    : NOTHING
CFC8:            279          -----
CFC8:            280 *
CFC8:            CFC8 281 PSETUP EQU *
CFC8: AD FB 04    282 LDA MODE   ; TRANSPARENT MODE?
CFCB: 29 01       283 AND %M_TRANS
CFCD: D0 03       CFD2 284 BNE PSETUP2 ; =>YES, TRUST WINDOW
CFCF: 20 78 CD    285 JSR FULLB0 ; SET FULL BOOCL WINDOW
CFD2:            286 *
CFD2:            CFD2 287 PSETUP2 EQU *
CFD2: A9 FF       288 LDA #255
CFD4: B5 32       289 STA INVFLG ; ASSUME NORMAL MODE
CFD6:            290 *
CFD6: AD FB 04    291 LDA MODE
CFD9: 29 04       292 AND %M_VMODE
CFDB: F0 02       CFDF 293 BEQ PSETUPRET ; =>IT'S NORMAL
CFDD: 46 32       294 LSR INVFLG ; MAKE IT INVERSE
CFDF:            295 *
CFDF:            CFDF 296 PSETUPRET EQU *
CFDF: AD 78 07    297 LDA QLDASL ; SET UP BASE ADDRESS
CFE2: B5 28       298 STA BASL
CFE4: AD FB 07    299 LDA QLDASL
CFE7: B5 29       300 STA BASH
CFE9: 60          301 RTS
CFEA:            302          -----
CFEA:            303 * NOTE: ENTRIES 6-7 OF THESE TABLES

```

```

CFEA:          304 * ARE NOT USED. THUS THERE ARE
CFEA:          305 * SOME OTHER VALUES STUFFED IN.
CFEA:          306 *
CFEA: 28       307 F. TABLE      DFB  >F. CLREOP-1
CFEB: 42       308             DFB  >F. HOME-1
CFEC: 4C       309             DFB  >F. SCROLL-1
CFED: 7C       310             DFB  >F. CLREOL-1
CFEE: 98       311             DFB  >F. CLEQLZ-1
CFEF: E9       312             DFB  >B. RESET-1 ; USE SAME RESET
CFF0: FF 01    313 PLUSMINUS1 DFB  -1, 1 ; SCROLL USES THIS
CFF2: B9       314             DFB  >F. SETWND-1
CFF3:          315 *
CFF3: E0       316 B. TABLE      DFB  >B. CLREOP-1
CFF4: EC       317             DFB  >B. HOME-1
CFF5: CC       318             DFB  >B. SCROLL-1
CFF6: D2       319             DFB  >B. CLREOL-1
CFF7: D8       320             DFB  >B. CLEQLZ-1
CFF8: E9       321             DFB  >B. RESET-1 ; USE SAME RESET
CFF9: 23 22    322 WNDTAB      DFB  WNDBTM, WNDTOP ; SCROLL USES THIS
CFFB: E6       323             DFB  >B. SETWND-1
CFFC: 00       324             DFB  0 ; AVOID CFFF PIPELINING
CFFD:          CFFD  15 ZZEND      EQU  *

```


80-Column Symbol Table, Sorted by Symbol

3B AIH	3C A1L	3F A2H	3E A2L
43 A4H	42 A4L	CE13 ATEFOR1	CE04 ATEFOR
CA02 B. CANLIT	C9DF B. CHKCAN	C1D9 B. CLEOLZ	C1D3 B. CLREOL
C1E1 B. CLREOP	C24E B. ESCFIX	C272 B. ESCFIX2	C27A B. ESCFIX3
CA0A B. FIXCHR	C9F7 B. FLIP	C1A4 B. FUNC0	C10E B. FUNCNE
?C100 B. FUNC	C211 B. FUNC1	C107 B. FUNCNK	C20E B. GETCH
C1ED B. HDME	C905 B. INPUT	CA24 B. INRET	C28B B. KEYIN
C29C B. KEYIN2	C9C6 B. NOPICK	C11F B. OLDFUNC	C1EA B. RESET
C234 B. RESETX	C1CD B. SCROLL	C221 B. SETWIND2	C219 B. SETWINDX
C1E7 B. SETWIND	CF73 B. TABLE	C1FF B. VECTOR	2B BAS2H
2A BAS2L	CB54 BASCALCZ	CB51 BASCALC	CB7E BASCLC3
CB97 BASCLCX	29 BASH	C317 BASICENT	C336 BASICENT2
CB03 BASICINIT	C305 BASICIN	?C300 BASICINT	C307 BASICOUT
2B BASL	CB03 BELL2	C100 BFUNCP6	CB31 BINITIA
CB16 BINIT1	CB50 BINIT2	CBF6 BINPUT	CBE2 BIORET
C232 BLAST	CB96 BOUT	CBCC BPNCTL	CB81 BPRINT
?CBE2 BS40	CB5B BSCLC1A	CB55 BSCLC1	CB6D BSCLC2
CBEB BSDONE	C398 C01	C3A3 C03	CB74 CBB2
CB7E CBB3	CB90 CBB4	CB66 CBASIC	Q7FB CBSLOT
24 CH	Q67B CHAR	CB5D CLEARIT	C181 CLEOL2
C12D CLEOP1	C000 CLRBOCOL	CB00 CLRBOVID	CO0C CLRALTCCHAR
CE9B CLRHALF2	CE91 CLRHALF	C300 CNO0	CF03 CDPYRET
CF95 CDPYROM2	CF7B CDPYROM	37 CSMH	36 CSHL
CC7B CTLADH	CC5F CTLADL	CB99 CTLCHAR	CBAB CTLCHARX
CB8E CTLQD	CB82 CTLRET	CB86 CTLXFER	23 CV
C241 DIAGS	CEA3 DD4B	C929 ESC1	C92B ESC2
C935 ESC3	C91B ESCAPING	C983 ESCCHAR	C280 ESCIN
C960 ESCNDONE	0011 ESCNUM	CF65 ESCOFF	CF32 ESCON
C2B4 ESCOUT	CF6E ESCRRET	C945 ESCSPEC	C934 ESCSPEC2
C963 ESCSPEC3	C972 ESCTAB	?FB01 F. BASCALC	C19C F. CLEOLZ
C17D F. CLREOL	C129 F. CLREOP	C1A1 GORET	C143 F. HOME
C2F1 F. RET1	C2E8 F. RETURN	C14D F. SCROLL	C18A F. SETWIND
CFAA F. TABLE	FC22 F. VTAB	FC24 F. VTABZ	FBB3 FVERSION
CE6F FORATE1	CE63 FORATE	CD9B FULL80	FD29 FUNCEXIT
CE22 GETB4	CB1B GETK2	CB15 GETKEY	CA27 GETPRIOR
CAAF GETY	C22E GOBACK	04 GOODFB	C27D GORETN
C2B5 GDTKEY	CA49 GPX	CB13 HANG	C2CC IK1
C2D5 IAK2A	C2CE IK2	C2DB IK3	CEDD INVERT
32 INWFLQ	FF9B IDRTS	C34B JBASINIT	C34B JPINIT
C351 JPREAD	C35D JPSTAT	C337 JPARITE	C010 KBDSTRB
C000 KBD	CB8A KBDWAIT	C2EA KDRET	?C2EA KDRETN
C2E9 KDRETY	C2C6 KEYDLY	39 KSMH	3B KSHL
CFB9 LCB1	CF02 LCB1ROM	CFB3 LC22ROM	40 M. BINPUT
80 M. ESCR	08 M. GDXY	01 M. IRG	10 M. LIT
02 M. PAS1.0	20 M. PASCAL	01 M. TRANS	04 M. VMODE
04FB MDDE	C37B MOVEC2M	C3B0 MOVELOOP	C3AC MOVERET
C37E MOVEBSTR	C363 MOVE	CCDD MSCRL0	CC01 MSCRL1
CCF9 MSCRL2	CD02 MSCRLRET	CD10 MSCRLRIS	C9B7 NDESC
C1C5 NDI	CB00 NDMWAIT	C3BA NKTA1	Q7FB QLD2ASH
Q77B QLD2ASL	Q47B QLDCH	CD09 ONEMORE	Q57B QURCH
Q5FB QURCV	CF01 PICK	CA62 PIGOOD	CA4A PINIT1.0
CA51 PINIT2	CA4F PINIT	CF00 PLUSMINUS1	CA74 PREAD
CABA PREADRET2	CFD2 PSETUP2	CF08 PSETUP	CFDF PSETUPRET
C99E PSTATUS2	C994 PSTATUS	C980 PSTATUS3	C984 PSTATUS4
?CA9E PWRITE2	CACB PWRITE3	CABE PWRITE	CABE PWRITE4
CB0F PWRITERET	CB09 PWRAP	CD00 QUIT2	CDAA QUIT
CO1B RDBOCOL	CO1F RDBOVID	CO03 RDCARDRAM	?FD0C RDKEY
CO11 RDLCBNK2	CO12 RDLCRAM	CO02 RDMAINRAM	CO1C RDPAGE2
CO13 RDRAMRD	CO14 RDRAMWRT	CO1A RTEXT	?CO19 RDBLBAR
C244 RESETRET	4F RNDH	4E RNDL	CE01 SCR4ORET
CDEA SCR40	CE3E SCR80	CE35 SCRBORET	CF06 SCREENIT
C153 SCRL1	C169 SCRL2	C172 SCRL3	CCD1 SCRLSUB
CF1E SCRN2	CF37 SCRN3	CF40 SCRN40	CE32 SCRN48
CDBB SCRN84	CE5B SCRNRET	CCAE SCROLL1	CCB8 SCROLL2
CCAA SCROLLDN	CCA4 SCROLLUP	CO01 SETBOVID	CO0D SETBOVID
CO0F SETALTCCHAR	CO09 SETALTZP	CB3E SETCB	CEAF SETCH
CEDP SETCHRIS	?CO07 SETINTCXROM	FE89 SETKBD	CO0B SETSLOTC3ROM
CO08 SETBYDZP	FE93 SETVID	CF00 SEV	FC73 SNIFFIRG
CO30 SPKX	CADC STARTXY	CB48 STAY2	CB4D STAY80
CE99 STOR2	CF4A STOR40	CF2A STOR80	CE22 STORCHAR
CF4E STPKEXIT	CC59 STUFFINV	Q47B TEMP1	00 TEST
CB24 TESTCARD	?CB4E TESTFAIL	CO54 TXTPAGE1	CO35 TXTPAGE2
CBFC WAIT	CBDD WAIT2	CBD1 WAIT3	23 WND8TH
20 WNDLFT	CF79 WNDTAB	22 WNDTOP	21 WNDWDTH
CO05 WRCARDRAM	CO04 WRMAINRAM	CB8C X. BELL	CBDB X. BS
C2F6 X. CLEOL2	C2F4 X. CLEOLZ	CC0C X. CRRET	CBEC X. CR

```

CBFD X. CRPAS          CD64 X. DC1B          CD76 X. DC1RTS        CD59 X. DC1
CDBB X. DC2B          CD9A X. DC2RET        CD77 X. DC2          CC0D X. EM
CD42 X. FF            CC33 X. FSRET        CC26 X. FS           CD48 X. QS
CD54 X. QS2          CD4E X. GSEDLZ       CC91 X. LF           CC9E X. LF2
CD20 X. LFRET        CD9D X. NAK          CD9A X. NAKRET       CD11 X. SCLRET
CD17 X. SCLRET2      CC52 X. SI           CC49 X. SQ           ?CC1D X. SUBB0
CC1F X. SUBLP        CC1A X. SUB          CC34 X. US           CC40 X. US1
CC45 X. US2          CC4B X. USRET        CD23 X. VT           CD2C X. VTLOOP
CD32 X. VTNEXT       O6FB XCOORD         C3B0 XFER           C3CD XFERAZP
C3C5 XFERC2M        C3DC XFERSZP        1F YSAV1           ? 02 ZSPARECZ
?CFDD ZZEND
** SUCCESSFUL ASSEMBLY : = NO ERRORS
** ASSEMBLER CREATED ON 05-JAN-82 000004
** TOTAL LINES ASSEMBLED 2419
** FREE SPACE PAGE COUNT 49
 2 EQUATES
 3 BFUNC
 4 C3SPACE
 5 CBSPACE
 6 BPRINT
 7 BINPUT
 8 PINIT
 9 PREAD
10 PWRITE
11 SUBS1
12 SUBS2
13 SUBS3

```

80-Column Symbol Table, Sorted by Address

00 TEST	01 M. TRANS	01 M. IRQ	02 M. PAS1.0
02 ZSPAREC2	04 M. VMODE	06 GOODFB	08 M. GQXY
10 M. LIT	0011 ESCNUM	1F YSAV1	20 M. PASCAL
20 WNDLFT	21 WNDWIDTH	22 WNDTOP	23 WNDBTM
24 CH	25 CV	28 BASL	29 BASH
2A BAS2L	2B BAS2H	32 INVFLG	36 CSWL
37 CSWH	3B KSWL	39 KSWH	3C AIL
3D A1H	3E A2L	3F A2H	40 M. BINPUT
42 A4L	43 A4H	4E RNDL	4F RNDH
80 M. ESCR	047B TEMP1	047B OLDCH	04FB MODE
057B DURCH	05FB OURCV	067B CHAR	06FB XCDORD
077B OLDBASL	07FB CBSLOT	07FB OLDBASH	0000 CLRBOCDL
0000 KBD	0001 SETBOCDL	0002 RDMAINRAM	0003 RDCARDAM
0004 WRMAINRAM	0005 WRCARDAM	?0007 SETINTCXDRM	0008 SETSTOZP
0009 SETALT7ZP	000B SETSL0TC3ROM	000C CLRBOVID	000D SETBOVID
000E CLRALTCHAR	000F SETALTCHAR	0010 KBDSTRB	0011 RDLCBNK2
0012 RDLGRAM	0013 RDRAMRD	0014 RDRAMRRT	0018 RDSOCDL
?0019 RDVBLBAR	001A RDTEXT	001C RDPAGE2	001F RDSOVID
0030 SPKR	0054 TXTPAGE1	0055 TXTPAGE2	?0100 B. FUNC
0100 BFUNCPG	0107 B. FUNCNK	010E B. FUNCNE	0110 B. OLDFUNC
0129 F. CLREDP	012D CLEDP1	0143 F. HOME	0140 F. SCROLL
0153 SCRL1	0169 SCRL2	0172 SCRL3	0170 F. CLREOL
0181 CLEOL2	018A F. SETWND	019C F. CLEOLZ	01A1 F. GORET
01A4 B. FUNC0	01C5 NDI	01CD B. SCROLL	01D3 B. CLREOL
01D9 B. CLEOLZ	01E1 B. CLREDP	01E7 B. SETWND	01EA B. RESET
01ED B. HOME	01FF B. VECTDR	020E B. GETCH	0211 B. FUNC1
0219 B. SETWINDX	0221 B. SETWND2	022E QDBACK	0234 B. RESETX
0252 BLAST	0241 DIAQS	0244 RESETRET	024E B. ESCFIX
0272 B. ESCFIX2	027A B. ESCFIX3	027D QDRETN	0280 ESCIN
0284 ESCOUT	028B B. KEYIN	029C B. KEYIN2	02B5 GDTKEY
0266 KEYDLY	02CC IK1	02CE IK2	02D5 IK2A
02DB IK3	?02E6 KDRETN	02E9 KDRETY	02EA KDRET
02EB F. RETURN	02F1 F. RET1	02F4 X. CLEOLZ	02F6 X. CLEOL2
0300 CNOO	?0300 BASICINT	0305 BASICIN	0307 BASICOUT
0317 BASICENT	0306 BASICENT2	034B JBASINIT	034B JPINIT
0351 JPREAD	0357 JWRITE	035D JBSTAT	0363 MOVE
0378 MOVEC2M	037E MOVESTRT	0380 MOVELOOP	038A NYTA1
0398 C01	03A3 C03	03AC MOVERET	0380 XFER
03C5 XFERC2M	03CD XFERAZP	03DC XFER5ZP	03EB SETCB
0B03 BASICINIT	0B13 HANG	0B16 BINIT1	0B31 BINITIA
0B50 BINIT2	0B5D CLEARIT	0B66 CBBASIC	0B74 CBB2
0B7E CBB3	0B90 CBB4	0B96 BOUT	0BA1 BPRINT
0BB4 KBDWAIT	0BCC NOWAIT	0BCC BPNCPL	0BE2 BIDRET
0BF6 BINPUT	0903 B. INPUT	0918 ESCAPING	0929 ESC1
092B ESC2	0933 ESC3	0945 ESCSPEC	0954 ESCSPEC2
0960 ESCNDNE	0963 ESCSPEC3	0972 ESCTAB	0983 ESCCHAR
0994 PSTATUS	099E PSTATUS3	09B0 PSTATUS3	09B4 PSTATUS4
09B7 NOESC	09C6 B. NOPICK	09DF B. CHKCAN	09F7 B. FLIP
CA02 B. CANLIT	CA0A B. FIXCHR	CA24 B. INRET	CA27 GETPRIOR
CA49 GPX	CA4A PINIT1.0	CA4F PINIT	CA51 PINIT2
CA62 PIGOOD	CA74 PREAD	CA8A PREADRET2	CA8E PWRITE
?CA9E PWRITE2	CAAF GETY	CACB PWRITE3	CADC STARTXY
CAEB PWRITE4	CB09 PWRWRAP	CB0F PWRIETER	CB15 GETKEY
CB1B GETKE	CB24 TESTCARD	CB48 STAY2	CB4D STAY80
?CB4E TESTFAIL	CB51 BASCALC	CB54 BASCALCZ	CB59 BSCLC1
CB5B BSCLC1A	CB5D BSCLC2	CB7E BASCLG3	CB97 BASCLCX
CB99 CTLCHAR	CBAB CTLCHARX	CBAE CTLG0	CB82 CTLRET
CB86 CTLXFER	CB8C X. BELL	CB83 BELL2	CBFC WAIT
CBDD WAIT2	CBDD X. BS	CBDB X. BS	?CBE2 BS40
CBEB BSDONE	CBEC X. CR	CBFD X. CRPAS	CC0C X. CRRET
CC0D X. EM	CC1A X. SUB	?CC1D X. SUBB0	CC1F X. SUBLP
CC26 X. FS	CC33 X. FSRET	CC34 X. US	CC40 X. US1
CC45 X. US2	CC4B X. USRET	CC49 X. S0	CC52 X. S1
CC59 STUFFINV	CC5F CTLADL	CC7B CTLADH	CC91 X. LF
CC9E X. LFR2	CCA4 SCROLLUP	CCA4 SCROLLDN	CCAE SCROLL1
CCBB SCROLL2	CCD1 SCRLSUB	CCD MSCLRO	CCF1 MSCLR1
CCF9 MSCRL2	CCD2 MSCRLRET	CD09 ONEMORE	CD10 MSCRLR1S
CD11 X. SCRLRET	CD17 X. SCRLRET2	CD20 X. LFRET	CD23 X. VT
CD2C X. VLOOP	CD32 X. VTNEXT	CD42 X. FF	CD4B X. GS
CD4E X. GSEDLZ	CD54 X. GS2	CD59 X. DC1	CD64 X. DC1B
CD76 X. DC1RTS	CD77 X. DC2	CD88 X. DC2B	CD90 X. NAK
CD9A X. NAKRET	CD9A X. DC2RET	CD9B FULLB0	CDAA QUIT
CDCC QUIT2	CDDB SCRN84	CDEA SCR40	CE01 SCR40RET
CE0A ATEFOR	CE13 ATEFOR1	CE22 GETB4	CE32 SCRN4B
CE3E SCR80	CE55 SCRBORET	CE5B SCRNRET	CE63 FORATE
CE6F FORATE1	CE91 CLRHALF	CE9B CLRHALF2	CEA3 DD4B

CEAF SETCH	CED9 SETCHRTS	CEDD INVERT	CEF2 STORCHAR
CEF9 STOR2	CF00 SEV	CF01 PICK	CF06 SCREENIT
CF1E SCRNR2	CF2A STOR80	CF37 SCRNR3	CF40 SCRNR40
CF4A STOR40	CF4E STPKEXIT	CF52 ESCDN	CF65 ESCOFF
CF6E ESCRET	CF78 COPYROM	CF95 COPYROM2	CFB3 LCB2ROM
CFB9 LCB1	CFC2 LCB1ROM	CFC5 COPYRET	CFC8 PSETUP
CFD2 PSETUP2	CDFD PSETUPRET	CFEA F. TABLE	CFFO PLUSMINUS1
FFF3 B. TABLE	FFF9 WNDTAB	?CFFD ZZEND	FBB3 FBVERSION
?FBC1 F. BASCALC	FC22 F. VTAB	FC24 F. VTABZ	FC75 SNIFFIRG
?FD0C RDKEY	FD29 FUNCEXIT	FE89 SETKBD	FE93 SETVID
FF9B IORTS			

** SUCCESSFUL ASSEMBLY :- NO ERRORS
** ASSEMBLER CREATED ON 05-JAN-82 000004
** TOTAL LINES ASSEMBLED 2421
** FREE SPACE PAGE COUNT 49

2	EQUATES
3	BFUNC
4	C3SPACE
5	C8SPACE
6	BPRINT
7	BINPUT
8	PINIT
9	FREAD
10	PWRITE
11	SUBS1
12	SUBS2
13	SUBS3



Lined writing area with horizontal red lines and light beige highlighted sections.



20525 Mariani Avenue
Cupertino, CA 95014
(408) 996-1010
TLX 171576

031-0357-A