2 of 12

CONFIDENTIAL
ATARI
PRELIMINARY

MONITOR

Back to Book

MONITOR

```
                        TITLE "MONITOR  *****  MONITP.SRC  ***** 3/9/79  *****  4:00:00 P.M."


                                CONSTANT EQUATES

0009            PUTTXT  =       $9              "PUT TEXT RECORD" CIO COMMAND CODE
0007            GETCAR  =       $7              "GET CHARACTER" CIO COMMAND CODE
0008            PUTCAR  =       $8              "PUT CHARACTER" CIO COMMAND CODE
0000            INIMLL  =       $00             INITIAL MEM LO  LOW BYTE
0007            INIMLH  =       $07             INITIAL MEM LO  HIGH BYTE
009B            CR      =       $9B             ASCII CARRIAGE RETURN
0001            GOOD    =       $1              GOOD STATUS CODE
0057            WRITE   =       $57             WRITE COMMAND
0052            READ    =       $52             READ COMMAND
0053            STATC   =       $53             STATUS COMMAND
0000            SEX     =       $0              SCREEN EDITOR IOCB INDEX
007D            CLS     =       $7D             CLEAR SCREEN CODE
0092            CTRLC   =       $92             KEYBOARD CODE FOR 'CONTROL C'
0088            EOF     =       $88             CASSETTE END OF FILE CODE
0000            LIRQ    =       $0              LONG IRQ TYPE CODE

0004            BUFFH   =       CASBUF+3/256
0000            BUFFL   =       -256*BUFFH+CASBUF+3     BUFFER POINTER


                THE FOLLOWING EQUATES ARE IN THE CARTRIDGE ADDRESS SPACE.

                        "B" CARTRIDGE ADDR'S ARE 8000-9FFF (36K CONFIG. ONLY)
                        "A" CART. ADDR'S ARE A000-BFFF (36K CONFIG. ONLY)

                        "A" CART. ADDR'S ARE 8000-BFFF (48K CONFIG. ONLY)

                        * = $BFFA
BFFA            CARTCS  *=*+2                   CARTRIDGE COLD START ADDRESS.
BFFC            CART    *=*+1                   CARTRIDGE AVAILABLE FLAG BYTE.
BFFD            CARTFG  *=*+1                   CARTRIDGE FLAG BYTE.  BIT 0=FLAG1,
BFFE            CARTAD  *=*+2                   2-BYTE CARTRIDGE START VECTOR

                        CARTRIDGE FLAG ACTION DEFINITIONS

                        BIT             ACTION IF SET

                        7               SPECIAL -- DON'T POWER-UP, JUST RUN CARTRIDGE
                        6-3             NONE
                        2               RUN CARTRIDGE
                        1               NONE
                        0               BOOT DOS

                        *****
                        NOTE
                        *****

                                1. IF BIT2 IS 0, GOTO BLACKBOARD MODE.
```

```
                        2. IF BITO SET, THE DISK WILL BE BOOTED BEFORE ANY
                           OTHER ACTION.



                        POWER-UP VECTOR
                        ***************************
                                * = $FFFC

                        PVECT   .WORD   PWRUP           POWER-UP VECTOR
                        ************************



                        ENTRY POINT VECTOR
                                *=BLKBDV

F471 4C 23 F2                   JMP     BIONON          BLACK BOARD VECTOR

                                *=WARMBV

F474 4C 06 90                   JMP     $9006           WARM START VECTOR

                                *=COLDBV

F477 4C 00 90                   JMP     $9000           COLD START VECTOR (9000 FOR RAM VECTOR WRITING)

                                *=$9000

9000 20 0C 90                   JSR     $900C
9003 4C 25 F1                   JMP     PWRUP           (TO HANDLE RAM VECTOR WRITING)
9006 20 0C 90                   JSR     $900C
9009 4C 18 F1                   JMP     REBET



                                * = MONORG



                        HANDLER TABLE ENTRIES

F0E3 50                 TBLENT  .BYTE   "P"
F0E4 30 E4                      .WORD   PRINTV
F0E6 43                         .BYTE   "C"
F0E7 40 E4                      .WORD   CASETV
F0E9 45                         .BYTE   "E"
F0EA 00 E4                      .WORD   EDITRV
F0EC 53                         .BYTE   "S"
F0ED 10 E4                      .WORD   SCRENV
```

```
1·0EF 4B                        .BYTE   "K"
1·0F0 20 E4                     .WORD   KEYBDV

                        ;
00DE                    TBLLEN  =       IDENT-TBLENT-1  HANDLER TABLE LENGTH
                        ;
                        ;               ***** PRINT MESSAGES *****
                        ;
1·0F2 7D 41 54          IDENT   .BYTE CLS,"ATARI COMPUTER - MEMO PAD",CR
1·0F5 41 52 49
1·0F8 20 43 4F
1·0FB 4D 50 55
1·0FE 54 45 52
1·101 20 2D 20
1·104 4D 45 4D
1·107 4F 20 50
1·10A 41 44 9B
                        ;
00F0                    IDENTH  =       IDENT/256
00F2                    IDENTL  =       -256*IDENTH+IDENT               SYSTEM I.D. MSG POINTER
                        ;
1·10D 42 4F 4F          DERR    .BYTE "BOOT ERROR",CR
F110 54 20 45
F113 52 52 4F
F116 52 9B
                        ;
00F1                    DERRH   =       DERR/256
00DD                    DERRL   =       -256*DERRH+DERR         DISK ERROR MSG POINTER
                        ;
                        ;               DEVICE/FILENAME SPECIFICATIONS
                        ;
F118 45 3A 9B           OPNEDT  .BYTE 'E:',CR           ;OPEN SCREEN EDITOR DEVICE SPEC
                        ;
00F1                    OPNH    =       OPNEDT/256
0018                    OPNL    =       -256*OPNH+OPNEDT                SCREEN EDITOR OPEN POINTER
                        ;
                        ;
                        ;***************************************************************
                        ;               RESET BUTTON ROUTINE STARTS HERE
                        ;***************************************************************
                        ;
F11B 78                 RESET   SEI                     DISABLE IRQ INTERRUPTS
F11C AD 44 02                   LDA     COLDST          WERE WE IN MIDDLE OF COLDSTART?
1·11F D0 04                     BNE     PWRUP           YES, GO TRY IT AGAIN
1·121 A9 FF                     LDA     #$FF
1·123 D0 03                     BNE     PWRUP1          SET WARM START FLAG
                        ;
                        ;***************************************************************
                        ;               POWER UP ROUTINES START HERE
                        ;***************************************************************
                        ;
```

```
F125 78          PWRUP   SEI             DISABLE IRQ INTERRUPTS
F126 A9 00               LDA     #0      CLEAR WARMSTART FLAG
F128 85 08       PWRUP1  STA     WARMST
F12A D8                  CLD             CLEAR DECIMAL FLAG.
F12B A2 FF               LDX     #$FF
F12D 9A                  TXS             SET STACK POINTER
F12E 20 3F F2            JSR     SPECL   CARTRIDGE SPECIAL CASE?
F131 20 81 F2            JSR     HARDI   DO HARDWARE INITIALIZATION
F134 A5 08               LDA     WARMST  IS IT WARMSTART?
F136 D0 28               BNE     ZOSRAM  YES, ONLY ZERO OS RAM

F138 A9 00       ZERORM  LDA     #0
F13A A0 08               LDY     #WARMST
F13C 85 04               STA     RAMLO
F13E 85 05               STA     RAMLO+1 INITIALIZE RAM POINTER
F140 91 04       CLRRAM  STA     (RAMLO),Y CLEAR MEMORY LOC.
F142 C8                  INY
F143 C0 00               CPY     #0      AT END OF PAGE?
F145 D0 F9               BNE     CLRRAM
F147 E6 05               INC     RAMLO+1 YES, INCR PAGE POINTER
F149 A6 05               LDX     RAMLO+1
F14B E4 06               CPX     TRAMSZ  AT END OF MEM?
F14D D0 F1               BNE     CLRRAM  NO.

                ; INITIALIZE DOSVEC TO POINT TO SIGNON (BLACKBOARD).
F14F AD 72 E4            LDA     BLKBDV+1
F152 85 0A               STA     DOSVEC  USE BLACKBOARD VECTOR
F154 AD 73 E4            LDA     BLKBDV+2 FOR DOSVEC
F157 85 0B               STA     DOSVEC+1
F159 A9 FF               LDA     #$FF
F15B 8D 44 02            STA     COLDST  ;SET TO SHOW IN MIDDLE OF COLDSTART
F15E D0 13               BNE     ESTBCM  ;GO AROUND ZOSRAM

                ; CLEAR OS RAM (FOR WARMSTART)
F160 A2 00       ZOSRAM  LDX     #0
F162 8A                  TXA
F163 9D 00 02    ZOSRM2  STA     $200,X  CLEAR PAGES 2 AND 3
F166 9D 00 03            STA     $300,X
F169 CA                  DEX
F16A D0 F7               BNE     ZOSRM2
F16C A2 10               LDX     #INTZBS
F16E 95 00       ZOSRM3  STA     0,X     CLEAR ZERO PAGE LOCATIONS INTZBS-7F
F170 E8                  INX
F171 10 FB               BPL     ZOSRM3

                ; ESTABLISH SCREEN MARGINS
F173 A9 02       ESTBCM  LDA     #LEDGE
F175 85 52               STA     LMARGN
F177 A9 27               LDA     #REDGE
F179 85 53               STA     RMARGN

                ; MOVE VECTOR TABLE FROM ROM TO RAM
F17B A2 25       DOSYS   LDX     #$25
F17D BD 80 E4    MOVVEC  LDA     VCTABL,X ROM TABLE
F180 9D 00 02            STA     INTABS,X TO RAM
F183 CA                  DEX
F184 10 F7               BPL     MOVVEC
F186 20 94 F2            JSR     OSRAM   DO O.S. RAM SETUP
F189 58                  CLI             ENABLE IRQ INTERRUPTS
```

```
                        ;
                        ;       LINK HANDLERS
                        ;
F18A A2 0E              LDX     #TBLLEN
F18C BD E3 F0   NXTENT  LDA     TBLENT,X        READ HANDLER TABLE ENTRY
F18F 9D 1A 03           STA     HATABS,X        PUT IN TABLE
F192 CA                 DEX
F193 10 F7              BPL     NXTENT          DONE WITH ALL ENTRIES?
                        ;
                        ;
                        ;       INTERROGATE CARTRIDGE ADDR. SPACE TO SEE WHICH CARTRIDGES THERE ARE
                        ;
F195 A2 00              LDX     #0
F197 86 07              STX     TSTDAT          CLEAR "B" CART. FLAG
F199 86 06              STX     TRAMSZ          CLEAR "A" CART. FLAG
F19B AE E4 02           LDX     RAMSIZ
F19E E0 90              CPX     #$90            RAM IN "B" CART. SLOT?
F1A0 B0 0A              BCS     ENDBCK
F1A2 AD FC 9F           LDA     CART-$2000      NO,
F1A5 D0 05              BNE     ENDBCK          CART. PLUGGED INTO "B" SLOT?
F1A7 E6 07              INC     TSTDAT          YES, SET "B" CART. FLAG
F1A9 20 3C F2           JSR     CBINI           INITIALIZE CARTRIDGE "B"
                        ;
F1AC AE E4 02   ENDBCK  LDX     RAMSIZ
F1AF E0 B0              CPX     #$B0            RAM IN "A" CART. SLOT?
F1B1 B0 0A              BCS     ENDACK
F1B3 AE FC BF           LDX     CART            NO,
F1B6 D0 05              BNE     ENDACK          CART. PLUGGED INTO "A" SLOT?
F1B8 E6 06              INC     TRAMSZ          YES, SET "A" CART. FLAG
F1BA 20 39 F2           JSR     CAINI           INITIALIZE CARTRIDGE "A"
                        ;
                        ;       OPEN SCREEN EDITOR
                        ;
F1BD A9 03      ENDACK  LDA     #3
F1BF A2 00              LDX     #$EX
F1C1 9D 42 03           STA     ICCOM,X         OPEN I/O COMMAND
F1C4 A9 18              LDA     #OPNL
F1C6 9D 44 03           STA     ICBAL,X
F1C9 A9 F1              LDA     #OPNH
F1CB 9D 45 03           STA     ICBAH,X         SET BUFFER POINTER TO OPEN SCREEN EDITOR
F1CE A9 0C              LDA     #$C
F1D0 9D 4A 03           STA     ICAX1,X         SET UP OPEN FOR INPUT/OUTPUT
F1D3 20 56 E4           JSR     CIOV            GO TO CIO
                        ;
F1D6 10 03              BPL     SCRNOK          BR IF NO ERROR
F1D8 4C 25 F1           JMP     PWRUP           RETRY PWRUP IF ERROR (SHOULD NEVER HAPPEN!)
F1DB E8         SCRNOK  INX                     SCREEN OK, GO WAIT FOR VBLANK TO
F1DC D0 FD              BNE     SCRNOK          BRING UP THE DISPLAY
F1DE C8                 INY
F1DF 10 FA              BPL     SCRNOK
                        ;
                        ; DO CASSETTE BOOT
F1E1 20 B2 F3           JSR     CSBOOT          CHECK, BOOT, AND INIT
```

```
                        ; CHECK TO SEE IF EITHER CARTRIDGE WANTS DISK BOOT
F1E4 A5 06              NOCART  LDA     TRAMSZ          CHECK BOTH CARTRIDGES
F1E6 05 07                      ORA     TSTDAT
F1E8 F0 12                      BEQ     NOCART          NEITHER CARTRIDGE LIVES
F1EA A5 06                      LDA     TRAMSZ          "A" CART?
F1EC F0 03                      BEQ     NOA1            NO
F1EE AD FD BF                   LDA     CARTFG          GET CARTRIDGE MODE FLAG
F1F1 A6 07              NOA1    LDX     TSTDAT          "B" CART?
F1F3 F0 03                      BEQ     NOB1            NO
F1F5 0D FD 9F                   ORA     CARTFG-$2000    ADD OTHER FLAG
F1F8 29 01              NOB1    AND     #1              DOES EITHER CART WANT BOOT?
F1FA F0 03                      BEQ     NOBOOT          NO
                        ;
                        ; DO DISK BOOT
F1FC 20 CF F2           NOCART  JSR     BOOT            CHECK, BOOT, AND INIT
                        ;
                        ; GO TO ONE OF THE CARTRIDGES IF THEY SO DESIRE
F1FF A9 00              NOBOOT  LDA     #0
F201 8D 44 02                   STA     COLDST          RESET TO SHOW DONE WITH COLDSTART
F204 A5 06                      LDA     TRAMSZ          "A" CART?
F206 F0 0A                      BEQ     NOA2            NO
F208 AD FD BF                   LDA     CARTFG          GET CARTRIDGE MODE FLAG
F20B 29 04                      AND     #4              DOES IT WANT TO RUN?
F20D F0 03                      BEQ     NOA2            NO
F20F 6C FA BF                   JMP     (CARTCS)        RUN "A" CARTRIDGE
F212 A5 07              NOA2    LDA     TSTDAT          "B" CART?
F214 F0 0A                      BEQ     NOCAR2          NO
F216 AD FD 9F                   LDA     CARTFG-$2000    GET "B" MODE FLAG
F219 29 04                      AND     #4              DOES IT WANT TO RUN?
F21B F0 DF                      BEQ     NOCART          NO
F21D 6C FA 9F                   JMP     (CARTCS-$2000)  RUN "B" CARTRIDGE
                        ;
                        ; NO CARTRIDGES, OR NEITHER WANTS TO RUN,
                        ; SO GO TO DOSVEC (DOS, CASSETTE, OR BLACKBOARD)
F220 6C 0A 00           NOCAR2  JMP     (DOSVEC)
                        ;
                        ; PRINT SIGN-ON MESSAGE
F223 A2 F2              SIGNON  LDX     #IDENTL
F225 A0 F0                      LDY     #IDENTH
F227 20 85 F3                   JSR     PUTLIN          GO PUT SIGN-ON MSG ON SCREEN
                        ;
                        ;
                        ; BLACKBOARD ROUTINE
F22A 20 30 F2           BLACKB  JSR     BLKB2           "JSR EGETCH"
F22D 4C 2A F2                   JMP     BLACKB          FOREVER
F230 AD 05 E4           BLKB2   LDA     EDITRV+5        HIGH BYTE
F233 48                         PHA
F234 AD 04 E4                   LDA     EDITRV+4        LOW BYTE
F237 48                         PHA
F238 60                         RTS                     SIMULATES "JMP (EDITRV)"
                        ;
                        ; CARTRIDGE INITIALIZATION INDIRECT JUMPS
F239 6C FE BF           CAINI   JMP     (CARTAD)
F23C 6C FE 9F           CBINI   JMP     (CARTAD-$2000)
```

S U B R O U T I N E S

```
                                  CHECK FOR HOW MUCH RAM & SPECIAL CARTRIDGE CASE.
                                        IF SPECIAL CARTRIDGE CASE, DON'T GO BACK -- GO TO CART.

F23F AD FC BF       SPECL   LDA     CART            CHECK FOR RAM OR CART
F242 D0 12                  BNE     ENSPEC          GO IF NOTHING OR MAYBE RAM
F244 EE FC BF               INC     CART            NOW DO RAM CHECK
F247 AD FC BF               LDA     CART            IS IT ROM?
F24A D0 0A                  BNE     ENSPEC          NO
F24C AD FD BF               LDA     CARTFG          YES,
F24F 29 80                  AND     #$80            MASK OFF SPECIAL BIT
F251 F0 03                  BEQ     ENSPEC          BIT SET?
F253 6C FE BF               JMP     (CARTAD)        YES, GO RUN CARTRIDGE

                            CHECK FOR AMOUNT OF RAM

F256 CE FC BF       ENSPEC  DEC     CART            RESTORE RAM IF NEEDED
F259 A9 00                  LDA     #0
F25B 85 05                  STA     RAMLO+1
F25D A9 10                  LDA     #$10
F25F 85 06                  STA     TRAMSZ          SET RAM POINTER TO 4K.
F261 A0 00                  LDY     #0
F263 B1 05          HOWMCH  LDA     (RAMLO+1),Y     READ RAM LOCATION
F265 85 07                  STA     TSTDAT          SAVE DATA
F267 49 FF                  EOR     #$FF            INVERT IT.
F269 85 04                  STA     RAMLO           SAVE INVERTED DATA
F26B 91 05                  STA     (RAMLO+1),Y     WRITE INVERTED DATA.
F26D B1 05                  LDA     (RAMLO+1),Y     READ RAM AGAIN
F26F C5 04                  CMP     RAMLO           IS IT THE INVERTED DATA?
F271 D0 00                  BNE     ENDRAM
F273 A5 07                  LDA     TSTDAT          YES.
F275 91 05                  STA     (RAMLO+1),Y     RESTORE ORIGINAL RAM DATA
F277 A5 06                  LDA     TRAMSZ
F279 18                     CLC
F27A 69 10                  ADC     #$10
```

```
1:27C 85 06                 STA     TRAMBZ          INCR. RAM POINTER BY 4K.
1:27E D0 E3                 BNE     HOWMCH          GO FIND HOW MUCH RAM.
1:280 60            ENDRAM  RTS


                            HARDWARE INITIALIZATION


1:281 A9 00         HARDI   LDA     #0
1:283 AA                    TAX
1:284 9D 00 D0      CLRCHP  STA     $D000,X
1:287 9D 00 D4              STA     $D400,X
1:28A 9D 00 D2              STA     $D200,X
1:28D 9D 00 D3              STA     $D300,X
1:290 E8                    INX
1:291 D0 F1                 BNE     CLRCHP
1:293 60                    RTS


                            O.S. RAM SETUP

1:294 C6 11         OSRAM   DEC     BRKKEY          TURN OFF BREAK KEY FLAG
1:296 A5 06                 LDA     TRAMBZ          READ RAM SIZE IN TEMP. REG.
1:298 8D E4 02              STA     RAMSIZ          SAVE IT IN RAM SIZE.
1:29B 8D E6 02              STA     MEMTOP+1                INIT. MEMTOP ADDR HI BYTE
1:29E A9 00                 LDA     #0
1:2A0 8D E5 02              STA     MEMTOP          INIT. MEMTOP ADDR LO BYTE
1:2A3 A9 00                 LDA     #INIMLL
1:2A5 8D E7 02              STA     MEMLO
1:2A8 A9 07                 LDA     #INIMLH
1:2AA 8D E8 02              STA     MEMLO+1         INITIALIZE MEMLO ADDR VECTOR
1:2AD 20 0C E4              JSR     EDITRV+$C       EDITOR INIT.
1:2B0 20 1C E4              JSR     SCRENV+$C       SCREEN INIT.
1:2B3 20 2C E4              JSR     KEYBDV+$C       KEYBOARD INIT.
1:2B6 20 3C E4              JSR     PRINTV+$C       PRINTER HANDLER INIT.
1:2B9 20 4C E4              JSR     CASETV+$C       CASSETTE HANDLER INIT.
1:2BC 20 6E E4              JSR     CIOINV          CIO INIT.
1:2BF 20 65 E4              JSR     SIOINV          SIO INIT.
1:2C2 20 6B E4              JSR     INTINV          INTERRUPT HANDLER INIT.
1:2C5 AD 1F D0              LDA     CONSOL
1:2C8 29 01                 AND     #$1
1:2CA D0 02                 BNE     NOKEY           GAME START KEY DEPRESSED?
1:2CC E6 4A                 INC     CKEY            YES, SET KEY FLAG.
1:2CE 60            NOKEY   RTS


                            DO BOOT OF DISK

1:2CF A5 08         BOOT    LDA     WARMST
1:2D1 F0 0A                 BEQ     NOWARM          WARM START?
1:2D3 A5 09                 LDA     BOOT?           YES,
1:2D5 29 01                 AND     #1
1:2D7 F0 03                 BEQ     NOINIT          VALID BOOT?
1:2D9 20 7E F3              JSR     DINI            YES, RE-INIT. DOS SOFTWARE
1:2DC 60            NOINIT  RTS
1:2DD A9 01         NOWARM  LDA     #1
1:2DF 8D 01 03              STA     DUNIT           ASSIGN DISK DRIVE NO.
1:2E2 A9 53                 LDA     #STATC
```

```
F2E4 8D 02 03            STA   DCOMND        SET UP STATUS COMMAND
F2E7 20 53 E4            JSR   DSKINV        GO DO DISK STATUS
F2EA 10 01               BPL   DOBOOT        IS STATUS FROM SIO GOOD?
F2EC 60                  RTS                 NO, GO BACK WITH BAD BOOT STATUS

F2ED A9 00      DOBOOT   LDA   #0
F2EF 8D 0B 03            STA   DAUX2
F2F2 A9 01               LDA   #1
F2F4 8D 0A 03            STA   DAUX1         SET SECTOR # TO 1.
F2F7 A9 00               LDA   #BUFFL
F2F9 8D 04 03            STA   DBUFLO
F2FC A9 04               LDA   #BUFFH
F2FE 8D 05 03            STA   DBUFHI        SET UP BUFFER ADDR
F301 20 9D F3   SECT1    JSR   GETSEC        GET SECTOR
F304 10 08               BPL   ALLSEC        STATUS O.K. ?
F306 20 81 F3   BADDSK   JSR   DSKRDE        NO, GO PRINT DISK READ ERROR
F309 A5 48               LDA   CASSBT
F30B F0 E0               BEQ   DOBOOT        CASSETTE BOOT?
F30D 60                  RTS                 YES, QUIT
F30E A2 03      ALLSEC   LDX   #3
F310 BD 00 04   RDBYTE   LDA   CASBUF+3,X    READ A BUFFER BYTE
F313 9D 40 02            STA   DFLAGS,X      STORE IT
F316 CA                  DEX
F317 10 F7               BPL   RDBYTE        DONE WITH 4 BYTE TRANSFER ?
F319 AD 42 02            LDA   BOOTAD        YES.
F31C 85 04               STA   RAMLO
F31E AD 43 02            LDA   BOOTAD+1
F321 85 05               STA   RAMLO+1       PUT BOOT ADDR INTO Z. PAGE RAM
F323 AD 04 04            LDA   CASBUF+7
F326 85 00               STA   DOSINI        ESTABLISH DOS INIT ADDRESS
F328 AD 05 04            LDA   CASBUF+8
F32B 85 0D               STA   DOSINI+1
F32D A0 7F      MVBUFF   LDY   #$7F          YES, SET BYTE COUNT
F32F B9 00 04   MVNXB    LDA   CASBUF+3,Y
F332 91 04               STA   (RAMLO),Y     MOVE A BYTE FROM SECTOR BUFFER TO BOOT ADDR.
F334 88                  DEY
F335 10 F8               BPL   MVNXB         DONE ?
F337 18                  CLC                 YES,
F338 A5 04               LDA   RAMLO
F33A 69 80               ADC   #$80
F33C 85 04               STA   RAMLO
F33E A5 05               LDA   RAMLO+1
F340 69 00               ADC   #0
F342 85 05               STA   RAMLO+1       INCR BOOT LOADER BUFFER POINTER.
F344 CE 41 02            DEC   DBSECT        DECR # OF SECTORS.
F347 F0 11               BEQ   ENBOOT        MORE SECTORS ?
F349 EE 0A 03            INC   DAUX1         YES, INCR SECTOR #
F34C 20 9D F3   SECTX    JSR   GETSEC        GO GET SECTOR.
F34F 10 DC               BPL   MVBUFF        STATUS O.K. ?
F351 20 81 F3            JSR   DSKRDE        NO, GO PRINT DISK READ ERROR.
F354 A5 48               LDA   CASSBT
F356 D0 AE               BNE   BADDSK        IF CASSETTE, QUIT.
F358 F0 F2               BEQ   SECTX         IF DISK, TRY SECTOR AGAIN.
F35A A5 48      ENBOOT   LDA   CASSBT
F35C F0 03               BEQ   XBOOT         CASSETTE BOOT ?
F35E 20 9D F3            JSR   GETSEC        YES, GET EOF RECORD, BUT DON'T USE IT.
F361 20 6C F3   XBOOT    JSR   BLOAD        GO EXECUTE BOOT LOADER
F364 B0 A0               BCS   BADDSK        IF BAD BOOT, DO IT OVER AGAIN
F366 20 7E F3            JSR   DINI          GO INIT. SOFTWARE
F369 E6 09               INC   BOOT?         SHOW BOOT SUCCESS
```

```
F36B 60                 RTS
F36C 18         BLOAD   CLC
F36D AD 42 02           LDA     BOOTAD
F370 69 06              ADC     #6
F372 85 04              STA     RAMLO
F374 AD 43 02           LDA     BOOTAD+1
F377 69 00              ADC     #0
F379 85 05              STA     RAMLO+1         PUT START ADDR OF BOOTLOADER INTO RAM
F37B 6C 04 00           JMP     (RAMLO)
F37E 6C 0C 00   DINI    JMP     (DOSINI)

                        DISPLAY DISK READ ERROR MSG

F381 A2 0D      DSKRDE  LDX     #DERRL
F383 A0 F1              LDY     #DERRH

                        PUT LINE ON SCREEN AT PRESENT CURSOR POSITION

                                X-REG --> LO BYTE, BEGIN ADDR OF LINE
                                Y-REG --> HI BYTE, BEGIN ADDR OF LINE
F385 8A         PUTLIN  TXA
F386 A2 00              LDX     #$EX
F388 9D 44 03           STA     ICBAL,X
F38B 98                 TYA
F38C 9D 45 03           STA     ICBAH,X         SET UP ADDR OF BEGIN OF LINE
F38F A9 09              LDA     #PUTTXT
F391 9D 42 03           STA     ICCOM,X         "PUT TEXT RECORD" COMMAND
F394 A9 FF              LDA     #$FF
F396 9D 48 03           STA     ICBLL,X         SET BUFFER LENGTH
F399 20 56 E4           JSR     CIOV            PUT LINE ON SCREEN
F39C 60                 RTS


                        GET SECTOR FROM DISK 0

F39D A5 48      GETSEC  LDA     CASSBT
F39F F0 03              BEQ     DISKM           CASSETTE BOOT ?
F3A1 4C 7A E4           JMP     RBLOKV          YES, GO TO READ BLOCK ROUTINE
F3A4 A9 52      DISKM   LDA     #READ
F3A6 8D 02 03           STA     DCOMND          SET READ SECTOR COMMAND
F3A9 A9 01              LDA     #1
F3AB 8D 01 03           STA     DUNIT           SET DRIVE NO. TO DRIVE 0
F3AE 20 53 E4           JSR     DSKINV          GET SECTOR
F3B1 60                 RTS


                        DO CHECK FOR CASSETTE BOOT & IF SO, DO BOOT

F3B2 A5 08      CSBOOT  LDA     WARMST          WARMSTART?
F3B4 F0 0A              BEQ     CSBOT2          NO
F3B6 A5 09              LDA     BOOT?           GET BOOT FLAG
```

```
1-3B8 29 02                 AND     #2              WAS CASSETTE BOOT SUCCESFULL?
1-3BA F0 03                 BEQ     NOCSB2          NO
1-3BC 20 E1 F3              JSR     CINI            YES, INIT CASSETTE SOFTWARE
1-3BF 60            NOCSB2  RTS

1-3C0 A5 4A         CSBOT2  LDA     CKEY
1-3C2 F0 1C                 BEQ     NOCSBT          "C" KEY FLAG SET ?
1-3C4 A9 80                 LDA     #$80            YES,
1-3C6 85 3E                 STA     FTYPE           SET LONG IRQ TYPE
1-3C8 E6 4B                 INC     CASSBT          SET CASSETTE BOOT FLAG
1-3CA 20 7D E4              JSR     CSOPIV          OPEN CASSETTE FOR INPUT
1-3CD 20 01 F3              JSR     SECT1           DO BOOT & INIT
1-3D0 A9 00                 LDA     #0
1-3D2 85 4B                 STA     CASSBT          RESET CASSETTE BOOT FLAG
1-3D4 85 4A                 STA     CKEY            CLEAR KEY FLAG
1-3D6 06 09                 ASL     BOOT?           SHIFT BOOT FLAG (NOW=2 IF SUCCESS)
1-3D8 A5 0C                 LDA     DOSINI
1-3DA 85 02                 STA     CASINI          MOVE INIT ADDRESS FOR CASSETTE
1-3DC A5 0D                 LDA     DOSINI+1
1-3DE 85 03                 STA     CASINI+1
1-3E0 60            NOCSBT  RTS

1-3E1 6C 02 00      CINI    JMP     (CASINI)        INIT CASSETTE

                    ;***************************************************************

                    ;SPARE BYTE OR MODULE TOO LONG FLAG

1-3E4               CRNTPC  =       *

                            * = $14
0014 00             MONSPR  BYTE    KBDORG-CRNTPC   MONITP TOO LONG

                    .END
```

SYMBOL TABLE

ADDCHR	030E	ADRESS	0064	AFP	D800	ALLPOT	D208
ANTIC	F30E	ANTIC	D400	APPEND	0001	APPMHI	000E
ATACHR	02FB	ATAN	BE43	ATRACT	004D	AUDC1	D201
AUDC2	D203	AUDC3	D205	AUDC4	D207	AUDCTL	D208
AUDF1	D200	AUDF2	D202	AUDF3	D204	AUDF4	D206
BADIOK	F306	BADIOC	0086	BADMOD	0091	BFENHI	0035
BELNLO	0034	BITMSK	006E	BLACKB	E22A	BLIM	038A
BLKBDV	F230	BLKBDV	E471	BLOAD	F36C	BOOT	F2CF
BOOT?	0009	BOOTAD	0242	BOTSCR	02BF	BPTR	003D
BRKABT	0080	BRKKEY	0011	BUFADR	0015	BUFCNT	006B
BUFFHI	0004	BUFFLO	0000	BUFRFL	003B	BUFRHI	0033
BUFRLO	0032	BUFSTR	006C	CAINI	F239	CART	BFFC
CARTAD	BFFE	CARTCS	BFFA	CARTFG	BFFD	CASBUF	03FD
CASETV	E440	CASFLG	030F	CASINI	0002	CASORG	EF41
CASRDY	004B	CASSET	0043	CAUX1	023C	CAUX2	023D
CBAUDH	02EF	CBAUDL	02EE	CBINI	F23C	CCOMND	023B
CDEVIC	023A	CDTMA1	0226	CDTMA2	0228	CDTMF3	022A
CDTMF4	022C	CDTMF5	022E	CDTMV1	0218	CDTMV2	021A
CDTMV3	021C	CDTMV4	021E	CDTMV5	0220	CH	02FC
CHJ	02F2	CHACT	02F3	CHACTL	D401	CHAR	02FA
CHBAS	02F4	CHBASE	D409	CHKERR	008F	CHKSNT	003B
CHKSUM	0031	CHRORG	E000	CINI	F3E1	CIOCHR	002F
CIOINV	E46E	CIOORG	E4A6	CIOV	E456	CIX	00F2
CKEY	004A	CLOSE	000C	CLRCHF	F284	CLRRAM	F140
CLSVT	007D	COLAC	0072	COLBK	D01A	COLCRS	0055
COLDST	0244	COLDSV	E477	COLINC	007A	COLORG	02C4
COLOR1	02C5	COLOR2	02C6	COLOR3	02C7	COLOR4	02C8
COLPF0	D016	COLPF1	D017	COLPF2	D018	COLPF3	D019
COLPM0	D012	COLPM1	D013	COLPM2	D014	COLPM3	D015
COLRSH	004F	CONSOL	D01F	COS	BD73	COUNTR	007E
CR	009B	CRETRY	0036	CRITIC	0042	CRNTPG	F3E4
CRSINH	02F0	CRSROR	008D	CSBOOT	F3B2	CSBOT2	F3C0
CSOPIV	E47D	CSTAT	0288	CTIA	D000	CTRLC	0092
DAUX1	030A	DAUX2	0308	DBSECT	0241	DBUFHI	0305
DBUFLO	0304	DBYTHI	0309	DBYTLO	0308	DCB	0300
DCOMND	0302	DDEVIC	0300	DEGFLG	00FB	DEGON	0004
DELETE	0021	DELTAC	0077	DELTAR	0076	DERR	F10D
DERRH	00F1	DERRL	000D	DERROR	0090	DFLAGS	0240
DIGRT	00F1	DINDEX	0057	DINI	F37E	DIRECT	0002
DISK	0044	DISKIV	E450	DISKM	F3A4	DISPLY	0053
DLISTH	D403	DLISTL	D402	DMACTL	D400	DMASK	02A0
DNACK	008B	DOBOOT	F2ED	DOSINI	000C	DOSVEC	000A
DRAMN	0011	DRETRY	0037	DRKMSK	004E	DSKFMS	0010
DSKINV	E453	DSKORG	EDEA	DSKRDE	F381	DSKTIM	0246
DSKUTL	001A	DSPFLG	02FE	DSTAT	004C	DSTATS	0303
DUNIO	0306	DUNIT	0301	DUNUSE	0307	DVSTAT	02EA
EDITRV	E400	EEXP	00ED	ENBOOT	F35A	ENDACK	F18D
ENDIOK	F1AC	ENDPT	0074	ENDRAM	F280	ENSPEC	F256
EOF	0088	EOFERR	0088	ERRFLG	023F	ESCFLG	02A2
ESIGN	00EF	ESTSCM	F173	EXP	DDC0	EXP10	DDCC
FADD	DA66	FASC	D8E6	FCHRFL	00F0	FDIV	DB28
FILDAT	003F	FILDAT	02FD	FILFLG	02B7	FILLIN	0012
FLDOR	DD8D	FLDOR	DD89	FLDIP	DD9C	FLDIR	DD98
FLPTR	00FC	FMOVE	DD86	FMSZPG	0043	FMUL	DAD8
FNCNIT	0092	FORMAT	0022	FPI	D9D2	FPREG	0006
FPSCR	05E6	FPSCR1	05EC	FPTR2	00FE	FRO	00D4
FR1	00E0	FR2	00E6	FRE	00DA	FREQ	0040
FRMRR	00BC	FRX	00EC	FSCR	05E6	FSCR1	05EC
FSTOR	DDA7	FSTOR	DDA7	FSUB	DA60	FTYPE	003E

GETCAN	0007	GETCHR	0007	GETREC	0005	GETSEC	F39D
GLDABS	02E0	GOOD	0001	GPRIOR	026F	GRACTL	D01D
GRAFN	D011	GRAFPO	D00D	GRAFP1	D00E	GRAFP2	D00F
GRAFP3	D010	HARDI	F281	HATABS	031A	HITCLR	D01E
HOLD1	0051	HOLD2	029F	HOLD3	029D	HOLD4	02BC
HOLDS	02BD	HOLDCH	007C	HOWMCH	F263	HPOSMO	D004
HPOSM1	D005	HPOSM2	D006	HPOSM3	D007	HPOSPO	D000
HPOSP1	D001	HPOSP2	D002	HPOSP3	D003	HSCROL	D404
ICAX1	034A	ICAX1Z	002A	ICAX2	034B	ICAX2Z	002B
ICBAH	0345	ICBAHZ	0025	ICBAL	0344	ICBALZ	0024
ICBLH	0349	ICBLHZ	0029	ICBLL	0348	ICBLLZ	0028
ICCOM	0342	ICCOMT	0017	ICCOMZ	0022	ICDNO	0341
ICDNOZ	0021	ICHID	0340	ICHIDZ	0020	ICIDNO	002E
ICPTH	0347	ICPTHZ	0027	ICPTL	0346	ICPTLZ	0026
ICSPR	034C	ICSPRZ	002C	ICSTA	0343	ICSTAZ	0023
IDENT	F0F2	IDENTH	00F0	IDENTL	00F2	IFP	D9AA
INBUFF	00F3	INIMLH	0007	INIMLL	0000	INSCLR	0020
INTDAT	007D	INTABS	0200	INTEMP	022D	INTINV	E46B
INTVRQ	E6D5	INTZBS	0010	INVFLG	02B6	IOCB	0340
IOCHAR	0020	IOCBSZ	0010	IOCFRE	00FF	IRQEN	D20E
IRQST	D20E	KBCODE	D209	KBD	004B	KBDVRQ	F3E4
KEYINV	E420	KEYDEL	02F1	LBFEND	05FF	LBPR1	057E
LDRIVE	057F	LBUFF	0580	LEDGE	0002	LINBUF	0247
LINZBS	0000	LIRQ	0000	LMARGN	0052	LOCKFL	0023
LOG	DECD	LOG10	DED1	LOGCOL	0063	LOGMAP	02B2
LPENH	0234	LPENV	0235	MOPF	D000	MOPL	D008
M1PF	D001	M1PL	D009	M2PF	D002	M2PL	D00A
M3PF	D003	M3PL	D00B	MAXDEV	0021	MAXIOC	0080
MEMLO	02E7	MEMTOP	02E5	MLTTMP	0066	MODEM	004D
MONRQ	F0E3	MONSPR	0014	MOVVEC	F17D	MVBUFF	F32D
MVNXB	F32F	MXDMOD	0010	NEWCOL	0061	NEWROW	0060
NMIEN	D40E	NMIRES	D40F	NMIST	D40F	NOA1	F1F1
NOA2	F212	NOB1	F1F8	NOBOOT	F1FF	NOCAR2	F220
NOCART	F1FC	NOCKSM	003C	NOCSB2	F3BF	NOCSBT	F3E0
NODNT	F2DC	NOKEY	F2CE	NONDEV	0082	NOTE	0026
NOTOPN	0085	NOWARM	F2DD	NSIGN	00E6	NVALID	0084
NXTENT	F18C	OLDADR	005E	OLDCHR	005D	OLDCOL	005B
OLDROW	005A	OPEN	0003	OPNEDT	F118	OPNH	00F1
OPNIN	0004	OPNINO	000C	OPNL	0018	OPNOT	0008
OPNTMP	0066	OPSYS	F17B	OSRAM	F294	OVRRUN	008E
POKE	D004	POPL	D00C	P1PF	D005	P1PL	D00D
P2PF	D006	P2PL	D00E	P3PF	D007	P3PL	D00F
PACTL	D302	PADDL0	0270	PADDL1	0271	PADDL2	0272
PADDL3	0273	PADDL4	0274	PADDL5	0275	PADDL6	0276
PADDL7	0277	PBCTL	D303	PBPNT	001D	PBUFSZ	001E
PCOLR0	02C0	PCOLR1	02C1	PCOLR2	02C2	PCOLR3	02C3
PENH	D40C	PENV	D40D	PIA	D300	PLYARG	05E0
PLYEVL	DD40	PMBASE	D407	POINT	0025	POKEY	D200
POKMSK	0010	PORTA	D300	PORTB	D301	POT0	D200
POT1	D201	POT2	D202	POT3	D203	POT4	D204
POT5	D205	POT6	D206	POT7	D207	POTGO	D20B
PRINTR	0050	PRINTV	E430	PRIOR	D01B	PRNBUF	03C0
PRNORG	EE78	PRVOPN	0081	PTEMP	001F	PTIMOT	001C
PTRIG0	027C	PTRIG1	027D	PTRIG2	027E	PTRIG3	027F
PTRIG4	0280	PTRIG5	0281	PTRIG6	0282	PTRIG7	0283
PUTCAR	0008	PUTCHR	0008	PUTLIN	F385	PUTREC	0009
PUTTXT	0009	PWRUP	F125	PWRUP1	F128	RADFLG	00FB
RADON	0000	RAMLO	0004	RAMSIZ	02E4	RAMTOP	006A
RANDOM	D20A	RBLOKV	E47A	RDBYTE	F310	RDONLY	0087
READ	0052	RECVDN	0039	REDGE	0027	RENAME	0020

RESET	F11B	RMARGN	0053	ROWAC	0070	ROWCRS	0054
ROWING	0079	RTCLOK	0012	SAVADR	0068	SAVID	0316
SAVNEG	0058	SCREDT	0045	SCRENV	E410	SCRFLG	02BB
SCRNUM	0093	SCRNOK	F1DB	SDLSTH	0231	SDLSTL	0230
SDRCTL	022F	SECT1	F301	SECTX	F34C	SENDEV	E468
SERIN	D20D	SEROUT	D20D	SETVBV	E45C	SEX	0000
SHFANT	006F	SHFLOK	02BE	SIGNON	F223	SIN	BD81
SIOINV	E465	SIOORG	E944	SIOV	E459	SIZEM	D00C
SIZEP0	D008	SIZEP1	D009	SIZEP2	D00A	SIZEP3	D00B
SKCTL	D20F	SKRES	D20A	SKSTAT	D20F	SOUNDR	0041
SPECIL	000E	SPECL	F23F	SQR	BE81	SRTIMR	022B
SSFLAG	02FF	SSKCTL	0232	STACKP	0318	STATC	0053
STATUS	000D	STATUS	0030	STICK0	0278	STICK1	0279
STICK2	027A	STICK3	027B	STIMER	D209	STRIG0	0284
STRIG1	0285	STRIG2	0286	STRIG3	0287	SUBTMP	029E
SUCCES	0001	SWPFLG	007B	SYSVBV	E45F	TABMAP	02A3
TBLLNT	F0E3	TBLLEN	000E	TEMP	023E	TEMP1	0312
TEMP2	0314	TEMP3	0315	TIMER1	030C	TIMER2	0310
TINFLG	0317	TINOUT	008A	TINDEX	0293	TMPCHR	0050
TMPCOL	0289	TMPLBT	02A1	TMPROW	0288	TMPX1	029C
TRANSZ	0006	TRIG0	D010	TRIG1	D011	TRIG2	D012
TRIG3	D013	TRNRCD	0089	TSTAT	0319	TSTDAT	0007
TXTCOL	0291	TXTMSG	0294	TXTOLD	0296	TXTROW	0290
UNLOCK	0024	USAREA	0480	VBREAK	0206	VCOUNT	D40B
VCTABL	E480	VDELAY	DD1C	VDSLST	0200	VECTBL	E400
VIMIRQ	0216	VINTER	0204	VKEYBD	0208	VPRCED	0202
VSCRIN	D405	VSERIN	020A	VSEROC	020E	VSEROR	020C
VTIMR1	0210	VTIMR2	0212	VTIMR4	0214	VVBLKD	0224
VVBLKI	0222	WARMST	0008	WARMSV	E474	WMODE	0289
WRITE	0057	WRONLY	0083	WSYNC	D40A	XBOOT	F361
XITVBV	E462	XMTDON	003A	ZERORM	F138	ZIOCB	0020
ZORAM	F160	ZORAM2	F163	ZORAM3	F16E	ZTEMP1	00F5
ZTEMP3	00F9	ZTEMP4	00F7				

```
                        TITLE 'DISPLAY HANDLER -- 10-30-78 -- DISPLC'

                ; HANDLER DEPENDENT EQUATES
009B            CR      =       $9B             SYSTEM EOL
007D            CLRCOD  =       $7D             CLEAR SCREEN ATASCI CODE
009F            CNTL1   =       $9F             POKEY KEY CODE FOR ^1

0068            FRMADR  =       SAVADR
0066            TOADR   =       MLTTMP
```

```
                        #=EDITRV

                    ;SCREEN EDITOR HANDLER ENTRY POINT

E400 FB F3      EDITOR  .WORD   EOPEN-1
E402 33 F6              .WORD   RETURI-1        (CLOSE)
E404 3D F6              .WORD   EGETCH-1
E406 A3 F6              .WORD   EOUTCH-1
E408 33 F6              .WORD   RETURI-1        (STATUS)
E40A 3C F6              .WORD   NOFUNC-1        (SPECIAL)
E40C 4C E4 F3           JMP     PWRONA
E40F 00                 .BYTE   0               ROM FILLER BYTE

                        #=SCRENV

                    ;DISPLAY HANDLER ENTRY POINT

E410 F9 F3      DISPLA  .WORD   DOPEN-1
E412 33 F6              .WORD   RETURI-1        (CLOSE)
E414 92 F5              .WORD   GETCH-1
E416 B6 F5              .WORD   OUTCH-1
E418 33 F6              .WORD   RETURI-1        (STATUS)
E41A FB FC              .WORD   DRAW-1          (SPECIAL)
E41C 4C E4 F3           JMP     PWRONA
E41F 00                 .BYTE   0               ROM FILLER BYTE

                        #=KEYBDV

                    ;KEYBOARD HANDLER ENTRY POINT

E420 33 F6      KBDHND  .WORD   RETURI-1
E422 33 F6              .WORD   RETURI-1        (CLOSE)
E424 E1 F6              .WORD   KGETCH-1
E426 3C F6              .WORD   NOFUNC-1        (OUTCH)
E428 33 F6              .WORD   RETURI-1        (STATUS)
E42A 3C F6              .WORD   NOFUNC-1        (SPECIAL)
E42C 4C E4 F3           JMP     PWRONA
E42F 00                 .BYTE   0               ROM FILLER BYTE

                    ;INTERRUPT VECTOR TABLE ENTRY
                        #=VCTABL-INTABS+VKEYBD
E488 BE FF              .WORD   PIRQ            KEYBOARD IRQ INTERRUPT VECTOR
```

```
                        *=KBDORQ

1:3E4 A9 FF     PWRONA  LDA     #$FF
1:3E6 8D FC 02          STA     CH
1:3E9 AD E6 02          LDA     MEMTOP+1
1:3EC 29 F0             AND     #$F0    INSURE 4K PAGE BOUNDARY
1:3EE 85 6A             STA     RAMTOP
1:3F0 A9 40             LDA     #$40    DEFAULT TO UPPER CASE ALPHA AT PWRON
1:3F2 8D BE 02          STA     SHFLOK
1:3F5 60                RTS             POWER ON COMPLETED
```

```
                        ;
                        ; BEGIN DISPLAY HANDLER OPEN PROCESSING
                        ;
F3F6 A5 2B      DOPEN   LDA     ICAX2Z  GET AUX 2 BYTE
F3F8 29 0F              AND     #$F
F3FA D0 08              BNE     OPNCOM  IF MODE ZERO, CLEAR ICAX1Z
F3FC A5 2A      EOPEN   LDA     ICAX1Z  CLEAR "CLR INHIBIT" AND "MXD MODE" BITS
F3FE 29 0F              AND     #$F
F400 85 2A              STA     ICAX1Z
F402 A9 00              LDA     #0
F404 85 57      OPNCOM  STA     DINDEX
F406 A9 E0              LDA     #$E0    INITIALIZE GLOBAL VBLANK RAM
F408 8D F4 02           STA     CHBAS
F40B A9 02              LDA     #2
F40D 8D F3 02           STA     CHACT
F410 8D 2F 02           STA     SDMCTL  TURN OFF DMA NEXT VBLANK
F413 A9 01              LDA     #SUCCES
F415 85 4C              STA     DSTAT   CLEAR STATUS
F417 A9 C0              LDA     #$C0    DO IRGEN
F419 05 10              ORA     POKMSK
F41B 85 10              STA     POKMSK
F41D 8D 0E D2           STA     IRGEN
F420 A9 00              LDA     #0
F422 8D 93 02           STA     TINDEX  TEXT INDEX MUST ALWAYS BE 0 !
F425 85 64              STA     ADRESS
F427 85 7B              STA     SWPFLG
F429 8D F0 02           STA     CRSINH  TURN CURSOR ON AT OPEN
F42C A0 0E              LDY     #14     CLEAR TAB STOPS
F42E A9 01              LDA     #1      INIT TAB STOPS TO EVERY 8 CHARACTERS
F430 99 A3 02   CLRTBS  STA     TABMAP,Y
F433 88                 DEY
F434 10 FA              BPL     CLRTBS
F436 A2 04              LDX     #4      LOAD COLOR REGISTERS
F438 BD C1 FE   DOPENS  LDA     COLRTB,X
F43B 9D C4 02           STA     COLOR0,X
F43E CA                 DEX
F43F 10 F7              BPL     DOPENS
F441 A4 6A              LDY     RAMTOP  DO TXTMSC=$2C40 (IF MEMTOP=$3000)
F443 88                 DEY
F444 8C 95 02           STY     TXTMSC+1
F447 A9 40              LDA     #$40
F449 8D 94 02           STA     TXTMSC
F44C A6 57              LDX     DINDEX
F44E BD 69 FE           LDA     ANCONV,X CONVERT IT TO ANTIC CODE
F451 D0 04              BNE     DOPENA  IF ZERO, IT IS ILLEGAL
F453 A9 91      OPNERR  LDA     #BADMOD SET ERROR STATUS
F455 85 4C              STA     DSTAT
F457 85 51      DOPENA  STA     HOLD1
F459 A5 6A              LDA     RAMTOP  SET UP AN INDIRECT POINTER
F45B 85 65              STA     ADRESS+1
F45D BC 45 FE           LDY     ALOCAT,X ALLOCATE N BLOCKS OF 40 BYTES
F460 A9 28      DOPEN1  LDA     #40
F462 20 21 F9           JSR     DBSUB
F465 88                 DEY
F466 D0 F8              BNE     DOPEN1
F468 AD 6F 02           LDA     OPRIOR  CLEAR GTIA MODES
F46B 29 3F              AND     #$3F
F46D 85 67              STA     OPNTMP+1
F46F A8                 TAY
```

```
F470 E0 08              CPX   #8        TEST IF 320X1
F472 90 17              BCC   NOT8
F474 8A                 TXA             GET 2 LOW BITS
F475 6A                 ROR   A
F476 6A                 ROR   A
F477 6A                 ROR   A
F478 29 C0              AND   #$C0      NOW 2 TOP BITS
F47A 05 67              ORA   OPNTMP+1
F47C A8                 TAY
F47D A9 10              LDA   #16     . SUBTRACT 16 MORE FOR PAGE BOUNDARY
F47F 20 21 F9           JSR   DBSUB
F482 E0 0B              CPX   #11       TEST MODE 11
F484 D0 05              BNE   NOT8      IF MODE = 11
F486 A9 06              LDA   #6        PUT GTIA LUM VALUE INTO BACKGROUND REGISTER
F488 8D C8 02           STA   COLOR4
F48B 8C 6F 02   NOT8    STY   GPRIOR    STORE NEW PRIORITY
F48E A5 64              LDA   ADRESS    SAVE MEMORY SCAN COUNTER ADDRESS
F490 85 58              STA   SAVMSC
F492 A5 65              LDA   ADRESS+1
F494 85 59              STA   SAVMSC+1
F496 AD 0B D4   VBWAIT  LDA   VCOUNT    WAIT FOR NEXT VBLANK BEFORE MESSING
F499 C9 7A              CMP   #$7A      WITH THE DISPLAY LIST
F49B D0 F9              BNE   VBWAIT
F49D 20 1F F9           JSR   DBDEC     START PUTTING DISPLAY LIST RIGHT UNDER RAM
F4A0 BD 75 FE           LDA   PAGETB,X  TEST IF DISPLAY LIST WILL BE IN TROUBLE
F4A3 F0 06              BEQ   NOMOD     OF CROSSING A 256 BYTE PAGE BOUNDARY
F4A5 A9 FF              LDA   #$FF      IF SO, DROP DOWN A PAGE
F4A7 85 64              STA   ADRESS
F4A9 C6 65              DEC   ADRESS+1
F4AB A5 64      NOMOD   LDA   ADRESS    SAVE END OF DISPLAY LIST FOR LATER
F4AD 85 68              STA   SAVADR
F4AF A5 65              LDA   ADRESS+1
F4B1 85 69              STA   SAVADR+1
F4B3 20 13 F9           JSR   DBDDEC    (DOUBLE BYTE DOUBLE DECREMENT)
F4B6 A9 41              LDA   #$41      (ANTIC) WAIT FOR VBLANK AND JMP TO TOP
F4B8 20 17 F9           JSR   STORE
F4BB 86 66              STX   OPNTMP
F4BD A9 18              LDA   #24       INITIALIZE BOTSCR
F4BF 8D BF 02           STA   BOTSCR
F4C2 A5 57              LDA   DINDEX    DISALLOW MIXED MODE IF MODE.GE.9
F4C4 C9 09              CMP   #9
F4C6 B0 20              BCS   NOTMXD
F4C8 A5 2A              LDA   ICAX1Z    TEST MIXED MODE
F4CA 29 10              AND   #$10
F4CC F0 27              BEQ   NOTMXD
F4CE A9 04              LDA   #4
F4D0 8D BF 02           STA   BOTSCR
F4D3 A2 02              LDX   #2        ADD 4 LINES OF TEXT AT BOTTOM OF SCREEN
F4D5 A9 02      DOPEN2  LDA   #2
F4D7 20 17 F9           JSR   STORE
F4DA CA                 DEX
F4DB 10 F8              BPL   DOPEN2
F4DD A4 6A              LDY   RAMTOP    RELOAD MSC FOR TEXT
F4DF 88                 DEY
F4E0 98                 TYA
F4E1 20 17 F9           JSR   STORE
F4E4 A9 60              LDA   #$60
F4E6 20 17 F9           JSR   STORE
F4E9 A9 42              LDA   #$42
F4EB 20 17 F9           JSR   STORE
```

```
F4EE 18                    CLC
F4EF A9 0C                 LDA   #MXDMDE-NUMDLE  POINT X AT MIXED MODE TABLE
F4F1 65 66                 ADC   OPNTMP
F4F3 85 66                 STA   OPNTMP
F4F5 A4 66           NOTMXD LDY   OPNTMP
F4F7 BE 51 FE              LDX   NUMDLE,Y GET NUMBER OF DISPLAY LIST ENTRIES
F4FA A5 51           DOPEN3 LDA   HOLD1   STORE N DLE'S
F4FC 20 17 F9              JSR   STORE
F4FF CA                    DEX
F500 D0 F8                 BNE   DOPEN3
F502 A5 57                 LDA   DINDEX  DO THE MESSY 320X1 PROBLEM
F504 C9 08                 CMP   #8
F506 90 1C                 BCC   DOPEN5
F508 A2 9D                 LDX   #93     GET REMAINING NUMBER OF DLE'S
F50A A5 6A                 LDA   RAMTOP  RELOAD MEMORY SCAN COUNTER
F50C 38                    SEC
F50D E9 10                 SBC   #$10
F50F 20 17 F9              JSR   STORE
F512 A9 00                 LDA   #0
F514 20 17 F9              JSR   STORE
F517 A9 4F                 LDA   #$4F    (ANTIC) RELOAD MSC CODE
F519 20 17 F9              JSR   STORE
F51C A5 51           DOPEN4 LDA   HOLD1   DO REMAINING DLE'S
F51E 20 17 F9              JSR   STORE
F521 CA                    DEX
F522 D0 F8                 BNE   DOPEN4
F524 A5 59           DOPEN5 LDA   SAVMSC+1 POLISH OFF DISPLAY LIST
F526 20 17 F9              JSR   STORE
F529 A5 58                 LDA   SAVMSC
F52B 20 17 F9              JSR   STORE
F52E A5 51                 LDA   HOLD1
F530 09 40                 ORA   #$40
F532 20 17 F9              JSR   STORE
F535 A9 70                 LDA   #$70    24 BLANK LINES
F537 20 17 F9              JSR   STORE
F53A A9 70                 LDA   #$70
F53C 20 17 F9              JSR   STORE
F53F A5 64                 LDA   ADRESS  SAVE DISPLAY LIST ADDRESS
F541 8D 30 02              STA   SDLSTL
F544 A5 65                 LDA   ADRESS+1
F546 8D 31 02              STA   SDLSTL+1
F549 A9 70                 LDA   #$70    ADD LAST BLANK LINE ENTRY
F54B 20 17 F9              JSR   STORE   POSITION ADRESS=SDLSTL-1
F54E A5 64                 LDA   ADRESS  STORE NEW MEMTOP
F550 8D E5 02              STA   MEMTOP
F553 A5 65                 LDA   ADRESS+1
F555 8D E6 02              STA   MEMTOP+1
F558 A5 68                 LDA   SAVADR
F55A 85 64                 STA   ADRESS
F55C A5 69                 LDA   SAVADR+1
F55E 85 65                 STA   ADRESS+1
F560 AD 31 02              LDA   SDLSTL+1
F563 20 17 F9              JSR   STORE
F566 AD 30 02              LDA   SDLSTL
F569 20 17 F9              JSR   STORE
F56C A5 4C                 LDA   DSTAT   IF ERROR OCURRED ON ALLOCATION, OPEN THE EDITOR
F56E 10 07                 BPL   DOPEN9
F570 48                    PHA           SAVE STATUS
F571 20 FC F3              JSR   EOPEN   OPEN THE EDITOR
F574 68                    PLA           RESTORE STATUS
```

```
F575 A8                    TAY          AND RETURN IT TO CIO
F576 60                    RTS
F577 A5 2A          DOPEN9 LDA   ICAX1Z TEST CLEAR INHIBIT BIT
F579 29 20                 AND   #$20
F57B D0 08                 BNE   DOPEN7
F57D 20 B9 F7              JSR   CLRSCR CLEAR SCREEN
F580 8D 90 02              STA   TXTROW AND HOME TEXT CURSOR (AC IS ZERO)
F583 A5 52                 LDA   LMARGN
F585 8D 91 02              STA   TXTCOL
F588 A9 22          DOPEN7 LDA   #$22   EVERYTHING ELSE IS SET UP
F58A 0D 2F 02              ORA   SDMCTL  GO TURN ON DMACTL
F58D 8D 2F 02              STA   SDMCTL
F590 4C 21 F6              JMP   RETURN

F593 20 96 FA       GETCH  JSR   RANGE  GETCH DOES INCRSR, GETPLT DOESN'T
F596 20 A2 F5              JSR   GETPLT
F599 20 32 FB              JSR   INATAC CONVERT INTERNAL CODE TO ATASCII
F59C 20 D4 F9              JSR   INCRSR
F59F 4C 34 F6              JMP   RETUR1
F5A2 20 47 F9       GETPLT JSR   CONVRT CONVERT ROW/COLUMN TO ADRESS
F5A5 B1 64                 LDA   (ADRESS),Y
F5A7 2D A0 02              AND   DMASK
F5AA 46 6F          SHIFTD LSR   SHFAMT SHIFT DATA DOWN TO LOW BITS
F5AC B0 03                 BCS   SHIFT1
F5AE 4A                    LSR   A
F5AF 10 F9                 BPL   SHIFTD  (UNCONDITIONAL)
F5B1 8D FA 02       SHIFT1 STA   CHAR
F5B4 C9 00                 CMP   #0     RESTORE FLAGS ALSO
F5B6 60                    RTS

F5B7 8D FB 02       OUTCH  STA   ATACHR
F5BA 20 96 FA              JSR   RANGE
F5BD 20 FC 02              JSR   OFFCRS
F5C0 AD FB 02       OUTCHA LDA   ATACHR TEST FOR CLEAR SCREEN
F5C3 C9 7D                 CMP   #CLRCOD
F5C5 D0 06                 BNE   OUTCHE
F5C7 20 B9 F7              JSR   CLRSCR
F5CA 4C 21 F6              JMP   RETURN
F5CD AD FB 02       OUTCHE LDA   ATACHR TEST FOR CARRIAGE RETURN
F5D0 C9 9B                 CMP   #CR
F5D2 D0 06                 BNE   OUTCHB
F5D4 20 30 FA              JSR   DOCRWB DO CR
F5D7 4C 21 F6              JMP   RETURN
F5DA 20 E0 F5       OUTCHB JSR   OUTPLT
F5DD 20 D8 F9              JSR   INCRSR
F5E0 4C 21 F6              JMP   RETURN

F5E3 AD FF 02       OUTPLT LDA   SSFLAG *****LOOP HERE IF START/STOP FLAG IS NON-0
F5E6 D0 FB                 BNE   OUTPLT
F5E8 A2 02                 LDX   #2
F5EA B5 54          CRLOOP LDA   ROWCRS,X SAVE CURSOR LOCATION FOR DRAW LINE TO DRAW FROM
F5EC 95 5A                 STA   OLDROW,X
F5EE CA                    DEX
F5EF 10 F9                 BPL   CRLOOP
F5F1 AD FB 02              LDA   ATACHR  CONVERT ATASCII(ATACHR) TO INTERNAL(CHAR)
F5F4 A8                    TAY          SAVE ATACHR
```

```
F5F2 2A                     ROL     A
F5F3 2A                     ROL     A
F5F4 2A                     ROL     A
F5F5 2A                     ROL     A
F5F6 29 03                  AND     #3
F5F8 AA                     TAX             X HAS INDEX INTO ATAINT
F5F9 98                     TYA             RESTORE ATACHR
F5FA 29 9F                  AND     #$9F    STRIP OFF COLUMN ADDRESS
F5FC 1D F6 FE               ORA     ATAINT,X OR IN NEW COLUMN ADDRESS
F5FF 8D FA 02       OUTCH2  STA     CHAR
F602 20 47 F9               JSR     CONVRT
F605 AD FA 02               LDA     CHAR
F608 46 6F          SHIFTU  LSR     SHFAMT  SHIFT UP TO PROPER POSITION
F60A B0 04                  BCS     SHIFT2
F60C 0A                     ASL     A
F60D 4C 08 F6               JMP     SHIFTU
F610 2D A0 02       SHIFT2  AND     DMASK
F613 85 50                  STA     TMPCHR  SAVE SHIFTED DATA
F615 AD A0 02               LDA     DMASK   INVERT MASK
F618 49 FF                  EOR     #$FF
F61A 31 64                  AND     (ADRESS),Y MASK OFF OLD DATA
F61C 05 50                  ORA     TMPCHR  OR IN NEW DATA
F61E 91 64                  STA     (ADRESS),Y
F620 60                     RTS

F621 20 A2 F5       RETURN  JSR     GETPLT  DO CURSOR ON THE WAY OUT
F624 85 5D                  STA     OLDCHR
F626 A6 57                  LDX     DINDEX  GRAPHICS HAVE INVISIBLE CURSOR
F628 D0 0A                  BNE     RETUR1
F62A AE F0 02               LDX     CRSINH  TEST CURSOR INHIBIT
F62D D0 05                  BNE     RETUR1
F62F 49 80                  EOR     #$80    TOGGLE MSB
F631 20 FF F5               JSR     OUTCH2  DISPLAY IT
F634 A4 4C          RETUR1  LDY     DSTAT           RETURN TO CIO WITH STATUS IN Y
F636 A9 01                  LDA     #SUCCES
F638 85 4C                  STA     DSTAT   SET STATUS= SUCCESSFUL COMPLETION
F63A AD FB 02               LDA     ATACHR  PUT ATACHR IN AC FOR RETURN TO CIO
F63D 60             NOFUNC  RTS             (NON-EXISTENT FUNCTION RETURN POINT)

                    END OF DISPLAY HANDLER
```

```
F63E 20 B3 FC      EGETCH  JSR     SWAP
F641 20 88 FA              JSR     ERANGE
F644 A5 6B                 LDA     BUFCNT  ANYTHING IN THE BUFFER?
F646 D0 34                 BNE     EGETC3  YES
F648 A5 54                 LDA     ROWCRS  NO, SO SAVE BUFFER START ADDRESS
F64A 85 6C                 STA     BUFSTR
F64C A5 55                 LDA     COLCRS
F64E 85 6D                 STA     BUFSTR+1
F650 20 E2 F6      EGETC1  JSR     KGETCH  LET'S FILL OUR BUFFER
F653 84 4C                 STY     DSTAT   SAVE KEYBOARD STATUS
F655 AD FB 02              LDA     ATACHR  TEST FOR CR
F658 C9 9B                 CMP     #CR
F65A F0 12                 BEQ     EGETC2
F65C 20 AD F6              JSR     DOSS    NO, GO PRINT IT
F65F 20 B3 FC              JSR     SWAP    JSR DOSS DID SWAP SO SWAP BACK
F662 A5 63                 LDA     LDGCOL  BEEP IF NEARING LOGICAL COL 120
F664 C9 71                 CMP     #113
F666 D0 03                 BNE     EGETC6
F668 20 0A F9              JSR     BELL
F66B 4C 50 F6      EGETC6  JMP     EGETC1
F66E 20 E4 FA      EGETC2  JSR     OFFCRS  GET BUFFER COUNT
F671 20 00 FC              JSR     DOBUFC
F674 A5 6C                 LDA     BUFSTR  RETURN A CHARACTER
F676 85 54                 STA     ROWCRS
F678 A5 6D                 LDA     BUFSTR+1
F67A 85 55                 STA     COLCRS
F67C A5 6B         EGETC3  LDA     BUFCNT
F67E F0 11                 BEQ     EGETC5
F680 C6 6B         EGETC7  DEC     BUFCNT  AND RETURN TILL BUFCNT=0
F682 F0 0D                 BEQ     EGETC5
F684 A5 4C                 LDA     DSTAT   IF ERROR, LOOP ON EGETC7 UNTIL BUFFER IS EMPTIED
F686 30 F8                 BMI     EGETC7
F688 20 93 F8              JSR     GETCH
F68B 8D FB 02              STA     ATACHR
F68E 4C B3 FC              JMP     SWAP    AND RETURN WITHOUT TURNING CURSOR BACK ON
F691 20 30 FA      EGETC5  JSR     DOCRWS  DO REAL CARRIAGE RETURN
F694 A9 9B                 LDA     #CR     AND RETURN EOL
F696 8D FB 02              STA     ATACHR
F699 20 21 F6              JSR     RETURN  TURN ON CURSOR THEN SWAP
F69C 84 4C                 STY     DSTAT   SAVE KEYBOARD STATUS
F69E 4C B3 FC              JMP     SWAP    AND RETURN THROUGH RETURI

F6A1 6C 64 00      JSRIND  JMP     (ADRESS)        JSR TO THIS CAUSES JSR INDIRECT

F6A4 8D FB 02      EOUTCH  STA     ATACHR  SAVE ATASCII VALUE
F6A7 20 B3 FC              JSR     SWAP
F6AA 20 88 FA              JSR     ERANGE
F6AD 20 E4 FA      DOSS    JSR     OFFCRS  TURN OFF CURSOR
F6B0 20 8D FC              JSR     TSTCTL  TEST FOR CONTROL CHARACTERS (Z=1 IF CTL)
F6B3 F0 09                 BEQ     EOUTC5
F6B5 0E A2 02      EOUTC6  ASL     ESCFLG  ESCFLG ONLY WORKS ONCE
F6B8 20 CA F5              JSR     OUTCHE
F6BB 4C B3 FC      ERETN   JMP     SWAP    AND RETURN THROUGH RETURI
F6BE AD FE 02      EOUTC5  LDA     DSPFLG  DO DSPFLG AND ESCFLG
F6C1 0D A2 02              ORA     ESCFLG
F6C4 D0 EF                 BNE     EOUTC6  IF NON-0 DISPLAY RATHER THAN EXECUTE IT
```

```
F6C6 0E A2 02            ASL   ESCFLG
F6C9 E8                  INX                PROCESS CONTROL CHARACTERS
F6CA BD C6 FE            LDA   CNTRLS,X GET DISPLACEMENT INTO ROUTINE
F6CD 85 64               STA   ADRESS
F6CF BD C7 FE            LDA   CNTRLS+1,X   GET HIGH BYTE
F6D2 85 65               STA   ADRESS+1
F6D4 20 A1 F6            JSR   JSRIND  DO COMPUTED JSR
F6D7 20 21 F6            JSR   RETURN  DO CURSOR
F6DA 4C B3 FC            JMP   SWAP    ALL DONE SO RETURN THROUGH RETURI

                    END SCREEN EDITOR.

                    BEGIN KEYBOARD HANDLER

F6DD A9 FF       KGETC2  LDA   #$FF
F6DF 8D FC 02            STA   CH
F6E2 A5 2A       KGETCH  LDA   ICAX1Z  TEST LSB OF AUX1 FOR SPECIAL EDITOR READ MODE
F6E4 4A                  LSR   A
F6E5 B0 62               BCS   GETOUT
F6E7 A9 80               LDA   #BRKABT
F6E9 A6 11               LDX   BRKKEY  TEST BREAK
F6EB F0 58               BEQ   K7      IF BREAK, PUT BRKABT IN DSTAT AND CR IN ATACHR
F6ED AD FC 02            LDA   CH
F6F0 C9 FF               CMP   #$FF
F6F2 F0 EE               BEQ   KGETCH
F6F4 85 7C               STA   HOLDCH  SAVE CH FOR SHIFT LOCK PROC
F6F6 A2 FF               LDX   #$FF    "CLEAR" CH
F6F8 8E FC 02            STX   CH
F6FB 20 DB FC            JSR   CLICK   DO KEYBOARD AUDIO FEEDBACK (A IS OK)
F6FE AA          KGETC0  TAX           DO ASCCON
F6FF E0 C0               CPX   #$C0    TEST FOR CTL & SHIFT TOGETHER
F701 90 02               BCC   ASCC01
F703 A2 03               LDX   #3      BAD CODE
F705 BD FE FE    ASCC01  LDA   ATASCI,X
F708 8D FB 02            STA   ATACHR  DONE
F70B C9 80               CMP   #$80    DO NULLS
F70D F0 CE               BEQ   KGETC2
F70F C9 81               CMP   #$81    CHECK ATARI KEY
F711 D0 0B               BNE   KGETC1
F713 AD B6 02            LDA   INVFLG
F716 49 80               EOR   #$80
F718 8D B6 02            STA   INVFLG
F71B 4C DD F6            JMP   KGETC2  DONT RETURN A VALUE
F71E C9 82       KGETC1  CMP   #$82    CAPS/LOWER
F720 D0 07               BNE   K1
F722 A9 00               LDA   #0      CLEAR SHFLOK
F724 8D BE 02            STA   SHFLOK
F727 F0 B4               BEQ   KGETC2
F729 C9 83       K1      CMP   #$83    SHIFT CAPS/LOWER
F72B D0 07               BNE   K2
F72D A9 40               LDA   #$40
F72F 8D BE 02            STA   SHFLOK  SHIFT BIT
F732 D0 A9               BNE   KGETC2
```

```
1734 C9 84        K2      CMP     #$84    CNTL CAPS/LOWER
1736 D0 07                BNE     K3
1738 A9 80                LDA     #$80    CNTL BIT
173A 8D BE 02             STA     SHFLOK
173D D0 9E                BNE     KGETC2
173F C9 85        K3      CMP     #$85    DO EOF
1741 D0 0A                BNE     K6
1743 A9 88                LDA     #EOFERR
1745 85 4C        K7      STA     DSTAT
1747 85 11                STA     BRKKEY  RESTORE BREAK
1749 A9 9B        GETOUT  LDA     #CR     PUT CR IN ATACHR
174B D0 26                BNE     K8      (UNCONDITIONAL)
174D A5 7C        K6      LDA     HOLDCH  PROCESS SHIFT LOCKS
174F C9 40                CMP     #$40    REGULAR SHIFT AND CONTROL TAKE PRECEDENCE
1751 B0 15                BCS     K5      OVER LOCK
1753 AD FB 02             LDA     ATACHR  TEST FOR ALPHA
1756 C9 61                CMP     #$61    LOWER CASE A
1758 90 0E                BCC     K5      NOT ALPHA IF LT
175A C9 7B                CMP     #$7B    LOWER CASE Z+1
175C B0 0A                BCS     K5      NOT ALPHA IF GE
175E AD BE 02             LDA     SHFLOK  DO SHIFT/CONTROL LOCK
1761 F0 05                BEQ     K5      IF NO LOCK, DONT RE-DO IT
1763 05 7C                ORA     HOLDCH
1765 4C FE F6             JMP     KGETC3  DO RETRY
1768 20 8D FC     K5      JSR     TSTCTL  DONT INVERT MSB OF CONTROL CHARACTERS
176B F0 09                BEQ     K4
176D AD FB 02             LDA     ATACHR
1770 4D B6 02             EOR     INVFLG
1773 8D FB 02     K8      STA     ATACHR
1776 4C 34 F6     K4      JMP     RETURI  ALL DONE
```

```
                        ;
                        ; CONTROL CHARACTER PROCESSORS
                        ;
1779 A9 80      ESCAPE  LDA     #$80    SET ESCAPE FLAG
177B 8D A2 02           STA     ESCFLG
177E 60                 RTS
177F C6 54      CRSRUP  DEC     ROWCRS
1781 10 06              BPL     COMRET
1783 AE BF 02           LDX     BOTSCR  WRAPAROUND
1786 CA                 DEX
1787 86 54      UPDNCM  STX     ROWCRS
1789 4C 5C FC   COMRET  JMP     STRBEQ  COLVERT ROW AND COL TO LOGCOL AND RETURN
178C E6 54      CRSRDN  INC     ROWCRS
178E A5 54              LDA     ROWCRS
1790 CD BF 02           CMP     BOTSCR
1793 90 F4              BCC     COMRET
1795 A2 00              LDX     #0
1797 F0 EE              BEQ     UPDNCM  (UNCONDITIONAL)
1799 C6 55      CRSRLF  DEC     COLCRS
179B A5 55              LDA     COLCRS
179D 30 04              BMI     CRSRL1  (IF LMARGN=0, THIS ELIMINATES PROBLEM CASE)
179F C5 52              CMP     LMARGN
17A1 B0 04              BCS     COMRE1
17A3 A5 53      CRSRL1  LDA     RMARGN
17A5 85 55      LFRTCM  STA     COLCRS
17A7 4C DD FB   COMRE1  JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN
17AA E6 55      CRSRRT  INC     COLCRS
17AC A5 55              LDA     COLCRS
17AE C5 53              CMP     RMARGN
17B0 90 F5              BCC     COMRE1
17B2 F0 F3              BEQ     COMRE1  (CAUSE BLE)
17B4 A5 52              LDA     LMARGN
17B6 4C A5 F7           JMP     LFRTCM  UNCONDITIONAL TO COMMON STORE
17B9 20 F3 FC   CLRSCR  JSR     PUTMSC
17BC A0 00              LDY     #0
17BE 98                 TYA             PUT 0 IN THE AC
17BF 91 64      CLRSC2  STA     (ADRESS),Y (AC IS ZERO)
17C1 C8                 INY
17C2 D0 FB              BNE     CLRSC2
17C4 E6 65              INC     ADRESS+1
17C6 A6 65              LDX     ADRESS+1
17C8 E4 6A              CPX     RAMTOP
17CA 90 F3              BCC     CLRSC2
17CC A9 FF              LDA     #$FF    CLEAN UP LOGICAL LINE BIT MAP
17CE 99 B2 02   CLRSC3  STA     LOGMAP,Y (Y IS ZERO AFTER CLRSC2 LOOP)
17D1 C8                 INY
17D2 C0 04              CPY     #4
17D4 90 F8              BCC     CLRSC3
17D6 20 E4 FC   HOME    JSR     COLCR   PLACE COLCRS AT LEFT EDGE
17D9 85 63              STA     LOGCOL
17DB 85 6D              STA     BUFSTR+1
17DD A9 00              LDA     #0
17DF 85 54              STA     ROWCRS
17E1 85 56              STA     COLCRS+1
17E3 85 6C              STA     BUFSTR
17E5 60                 RTS
```

```
F7EA F0 21              BEQ     BS1
F7EC A5 55      BSA     LDA     COLCRS  LEFT EDGE?
F7EE C5 52              CMP     LMARGN
F7F0 D0 03              BNE     BS3     NO
F7F2 20 73 FC           JSR     DELTIM  YES, SEE IF LINE SHOULD BE DELETED
F7F5 20 99 F7   BS3     JSR     CRSRLF
F7F8 A5 55              LDA     COLCRS
F7FA C5 53              CMP     RMARGN
F7FC D0 07              BNE     BS2
F7FE A5 54              LDA     ROWCRS
F800 F0 03              BEQ     BS2
F802 20 7F F7           JSR     CRSRUP
F805 A9 20      BS2     LDA     #$20    MAKE BACKSPACE DESTRUCTIVE
F807 8D FB 02           STA     ATACHR
F80A 20 E0 F5           JSR     OUTPLT
F80D 4C DD FB   BS1     JMP     DOLCOL  AND RETURN
F810 20 AA F7   TAB     JSR     CRSRRT  BEGIN SEARCH
F813 A5 55              LDA     COLCRS  TEST FOR NEW LINE
F815 C5 52              CMP     LMARGN
F817 D0 0A              BNE     TAB1    NO
F819 20 34 FA           JSR     DOCR    DO CARRIAGE RETURN
F81C 20 20 FB           JSR     LOGGET  CHECK IF END OF LOGICAL LINE
F81F 90 02              BCC     TAB1    NO, CONTINUE
F821 B0 07              BCS     TAB2    (UNCONDITIONAL)
F823 A5 63      TAB1    LDA     LOGCOL  CHECK FOR TAB STOP
F825 20 25 FB           JSR     BITGET
F828 90 E6              BCC     TAB     NO, SO KEEP LOOKING
F82A 4C DD FB   TAB2    JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN
F82D A5 63      SETTAB  LDA     LOGCOL
F82F 4C 06 FB           JMP     BITSET  SET BIT IN MAR AND RETURN
F832 A5 63      CLRTAB  LDA     LOGCOL
F834 4C 12 FB           JMP     BITCLR  CLEAR " " " " "
F837 20 9D FC   INSCHR  JSR     PHACRS
F83A 20 A2 F5           JSR     GETPLT  GET CHARACTER UNDER CURSOR
F83D 85 7D              STA     INSDAT
F83F A9 00              LDA     #0
F841 8D BB 02           STA     SCRFLG
F844 20 FF F8   INSCH4  JSR     OUTCH2  STORE DATA
F847 A5 63              LDA     LOGCOL  SAVE LOGCOL: IF AFTER INCRSA LOGCOL IS
F849 48                 PHA             < THAN IT IS NOW, END LOOP
F84A 20 DC F9           JSR     INCRSA  SPECIAL INCRSR ENTRY POINT
F84D 68                 PLA
F84E C5 63              CMP     LOGCOL
F850 B0 0C              BCS     INSCH3  QUIT
F852 A5 7D      INSCH1  LDA     INSDAT  KEEP GOING
F854 48                 PHA
F855 20 A2 F5           JSR     GETPLT
F858 85 7D              STA     INSDAT
F85A 68                 PLA
F85B 4C 44 F8           JMP     INSCH4
F85E 20 A8 FC   INSCH3  JSR     PLACRS
F861 CE BB 02   INSCH6  DEC     SCRFLG
F864 30 04              BMI     INSCH5  IF SCROLL OCCURRED
F866 C6 54              DEC     ROWCRS  MOVE CURSOR UP
F868 D0 F7              BNE     INSCH6  (UNCOND) CONTINUE UNTIL SCRFLG IS MINUS
F86A 4C DD FB   INSCH5  JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN


F86D 20 9D FC   DELCHR  JSR     PHACRS
F870 20 47 F9   DELCH1  JSR     CONVRT  GET DATA TO THE RIGHT OF THE CURSOR
```

```
F873 A5 64              LDA     ADRESS
F875 85 68              STA     SAVADR  SAVE ADRESS TO KNOW WHERE TO PUT DATA
F877 A5 65              LDA     ADRESS+1
F879 85 69              STA     SAVADR+1
F87B A5 63              LDA     LOGCOL
F87D 48                 PHA
F87E 20 D4 F9           JSR     INCRSB  PUT CURSOR OVER NEXT CHARACTER
F881 68                 PLA
F882 C5 63              CMP     LOGCOL  TEST NEW LOGCOL AGAINST OLD LOGCOL
F884 B0 10              BCS     DELCH2  IF OLD.GE.NEW THEN QUIT
F886 A5 54              LDA     ROWCRS  IS ROW OFF SCREEN?
F888 CD BF 02           CMP     BOTSCR
F88B B0 09              BCS     DELCH2  YES, SO QUIT
F88D 20 A2 F5           JSR     GETPLT  GET DATA UNDER CURSOR
F890 A0 00              LDY     #0
F892 91 68              STA     (SAVADR),Y      PUT IT IN PREVIOUS POSITION
F894 F0 DA              BEQ     DELCH1  AND LOOP (UNCONDITIONAL)
F896 A0 00      DELCH2  LDY     #0
F898 98                 TYA
F899 91 68              STA     (SAVADR),Y      CLEAR THE LAST POSITION
F89B 20 68 FC           JSR     DELTIA  TRY TO DELETE A LINE
F89E 20 A8 FC           JSR     PLACRS
F8A1 4C DD FB           JMP     DOLCOL  AND RETURN
F8A4 38         INSLIN  SEC             NORMAL INSLIN PUTS "1" INTO BIT MAP
F8A5 20 79 F9   INSLIA  JSR     EXTEND  ENTRY POINT FOR C=0
F8A8 A5 52              LDA     LMARGN  DO CARRIAGE RETURN (NO LF)
F8AA 85 55              STA     COLCRS
F8AC 20 47 F9           JSR     CONVRT  GET ADDRESS
F8AF A5 64              LDA     ADRESS  SET UP TO=40+FROM (FROM = CURSOR)
F8B1 85 68              STA     FRMADR
F8B3 18                 CLC
F8B4 69 28              ADC     #40
F8B6 85 66              STA     TOADR
F8B8 A5 65              LDA     ADRESS+1
F8BA 85 69              STA     FRMADR+1
F8BC 69 00              ADC     #0
F8BE 85 67              STA     TOADR+1
F8C0 A6 54              LDX     ROWCRS  SET UP LOOP COUNTER
F8C2 E0 17              CPX     #23
F8C4 F0 08              BEQ     INSLI2
F8C6 20 4E FB   INSLI1  JSR     MOVLIN
F8C9 E8                 INX
F8CA E0 17              CPX     #23
F8CC D0 F8              BNE     INSLI1
F8CE 20 98 FB   INSLI2  JSR     CLRLIN  CLEAR CURRENT LINE
F8D1 4C DD FB           JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN
F8D4 20 DD FB   DELLIN  JSR     DOLCOL  GET BEGINNING OF LOG LINE (HOLD1)
F8D7 A4 51      DELLIA  LDY     HOLD1   SQUEEZE BIT MAP
F8D9 84 64              STY     ROWCRS  PUT CURSOR THERE
F8DB A4 54      DELLIB  LDY     ROWCRS
F8DD 98         DELLI1  TYA
F8DE 38                 SEC
F8DF 20 23 FB           JSR     LOGGET  GET NEXT BIT
F8E2 08                 PHP
F8E3 98                 TYA
F8E4 18                 CLC
F8E5 69 78              ADC     #120
F8E7 28                 PLP
F8E8 20 04 FB           JSR     BITPUT  WRITE IT OVER PREBENT BIT
F8EB C8                 INY
```

```
1·8EC C0 18                     CPY     #24
1·8EE D0 ED                     BNE     DELLI1  LOOP
1·8F0 AD B4 02                  LDA     LOGMAP+2        SET LSB
1·8F3 09 01                     ORA     #1
1·8F5 8D B4 02                  STA     LOGMAP+2
1·8F8 A5 52             DELLI2  LDA     LMARGN  DELETE LINE OF DATA USING PART OF SCROLL
1·8FA 85 59                     STA     COLCRS  CR NO LF
1·8FC 20 47 F9                  JSR     CONVRT
1·8FF 20 B7 FB                  JSR     SCROL1
1·902 20 20 FB                  JSR     LOGGET  TEST NEXT LINE FOR CONTINUATION
                                                IS IT A NEW LOG LINE?
1·905 90 D4                     BCC     DELLI2  NO SO DELETE ANOTHER
1·907 4C DD FB                  JMP     DOLCOL  YES SO DOLCOL AND RETURN
1·90A A0 20             BELL    LDY     #$20
1·90C 20 DB FC          BELL1   JSR     CLICK
1·90F 88                        DEY
1·910 10 FA                     BPL     BELL1
1·912 60                        RTS
```

```
                        ;
                        ; ROUTINES
                        ;
                        ; DOUBLE BYTE DECREMENT OF INDIRECT POINTER
                        ;    INCLUDING DB SUBTRACT AND DB DOUBLE DECREMENT
F913 A9 02              DBDDEC  LDA     #2
F915 D0 0A                      BNE     DBSUB   (UNCONDITIONAL)
                        ;
                        ; STORE DATA INDIRECT AND DECREMENT POINTER
                        ; (PLACED HERE TO SAVE JMP DBDEC AFTER STORE)
F917 A4 4C              STORE   LDY     DSTAT   RETURN ON ERROR
F919 30 2B                      BMI     STROK
F91B A0 00                      LDY     #0
F91D 91 64              STORE1  STA     (ADRESS),Y
                                JMP     DBDEC   DECREMENT AND RETURN
                        ;
F91F A9 01              DBDEC   LDA     #1
F921 8D 9E 02           DBSUB   STA     SUBTMP
F924 A5 4C                      LDA     DSTAT   RETURN ON ERROR
F926 30 1E                      BMI     STROK
F928 A5 64                      LDA     ADRESS
F92A 38                         SEC
F92B ED 9E 02                   SBC     SUBTMP
F92E 85 64                      STA     ADRESS
F930 B0 02                      BCS     DBSUB1
F932 C6 65                      DEC     ADRESS+1
F934 A5 0F              DBSUB1  LDA     APPMHI+1 MAKE SURE NOTHING EVER OVERWRITES APPMHI
F936 C5 65                      CMP     ADRESS+1
F938 90 0C                      BCC     STROK   OK
F93A D0 06                      BNE     STRERR  ERROR
F93C A5 0E                      LDA     APPMHI
F93E C5 64                      CMP     ADRESS
F940 90 04                      BCC     STROK
F942 A9 93              STRERR  LDA     #SCRMEM SHOW MEM TOO SMALL FOR SCREEN ERROR
F944 85 4C                      STA     DSTAT
F946 60                 STROK   RTS
                        ;
                        ; CONVERT ROW/COLUMN CURSOR INTO REAL ADDRESS (FROM SAVMSC ON UP)
F947 A5 54              CONVRT  LDA     ROWCRS  SAVE CURSOR
F949 48                         PHA
F94A A5 55                      LDA     COLCRS
F94C 48                         PHA
F94D A5 56                      LDA     COLCRS+1
F94F 48                         PHA
F950 20 F3 FC                   JSR     PUTMSC
F953 A5 54                      LDA     ROWCRS  PUT 10*ROWCRS INTO MLTTMP
F955 85 66                      STA     MLTTMP
F957 A9 00                      LDA     #0
F959 85 67                      STA     MLTTMP+1
F95B A5 66                      LDA     MLTTMP  QUICK X8
F95D 0A                         ASL     A
F95E 26 67                      ROL     MLTTMP+1
F960 85 91                      STA     HOLD1   (SAVE 2X VALUE)
F962 A4 67                      LDY     MLTTMP+1    ""
```

```
F964 8C 9F 02              STY     HOLD2           ""
F967 0A                    ASL     A
F968 26 67                 ROL     MLTTMP+1
F96A 0A                    ASL     A
F96B 26 67                 ROL     MLTTMP+1
F96D 18                    CLC                     ADD IN 2X
F96E 65 51                 ADC     HOLD1
F970 85 66                 STA     MLTTMP
F972 A5 67                 LDA     MLTTMP+1
F974 6D 9F 02              ADC     HOLD2
F977 85 67                 STA     MLTTMP+1
F979 A6 57                 LDX     DINDEX  NOW SHIFT MLTTMP LEFT DHLINE TIMES TO FINISH
F97B BC B1 FE              LDY     DHLINE,X MULTIPLY
F97E 88              CONVR1 DEY             LOOP N TIMES
F97F 30 07                 BMI     CONVR2
F981 06 66                 ASL     MLTTMP
F983 26 67                 ROL     MLTTMP+1
F985 4C 7E F9              JMP     CONVR1
F988 BC A5 FE        CONVR2 LDY     DIV2TB,X NOW DIVIDE HCHAR TO ACCOUNT FOR PARTIAL BYTES
F98B A5 55                 LDA     COLCRS
F98D A2 07                 LDX     #7      * TRICKY *
F98F 88              CONVR3 DEY
F990 30 0A                 BMI     CONVR4
F992 CA                    DEX
F993 46 56                 LSR     COLCRS+1
F995 6A                    ROR     A
F996 6E A1 02              ROR     TMPLBT  SAVE LOW BITS FOR MASK
F999 4C 8F F9              JMP     CONVR3
F99C C8              CONVR4 INY             SO Y IS ZERO UPON RETURN FROM THIS ROUTINE
F99D 18                    CLC
F99E 65 66                 ADC     MLTTMP  ADD SHIFTED COLCRS TO MLTTMP
F9A0 85 66                 STA     MLTTMP
F9A2 90 02                 BCC     CONVR5
F9A4 E6 67                 INC     MLTTMP+1
F9A6 38              CONVR5 SEC             * TRICKY *
F9A7 6E A1 02        CONVR6 ROR     TMPLBT  SLIDE A "1" UP AGAINST LOW BITS (CONTINUE TILL X=-1)
F9AA 18                    CLC             AND FINISH SHIFT SO LOW BITS ARE
F9AB CA                    DEX             RIGHT JUSTIFIED.
F9AC 10 F9                 BPL     CONVR6
F9AE AE A1 02              LDX     TMPLBT  TMPLBT IS NOW THE INDEX INTO DMASKTB
F9B1 A5 66                 LDA     MLTTMP  PREPARE FOR RETURN
F9B3 18                    CLC
F9B4 65 64                 ADC     ADRESS
F9B6 85 64                 STA     ADRESS
F9B8 85 5E                 STA     OLDADR  REMEMBER THIS ADDRESS FOR CURSOR
F9BA A5 67                 LDA     MLTTMP+1
F9BC 65 65                 ADC     ADRESS+1
F9BE 85 65                 STA     ADRESS+1
F9C0 85 5F                 STA     OLDADR+1
F9C2 BD B1 FE              LDA     DMASKT,X
F9C5 8D A0 02              STA     DMASK
F9C8 85 6F                 STA     SHFAMT
F9CA 68                    PLA
F9CB 85 56                 STA     COLCRS+1
F9CD 68                    PLA
F9CE 85 55                 STA     COLCRS
F9D0 68                    PLA
F9D1 85 54                 STA     ROWCRS
F9D3 60                    RTS
```

```
                        ;
                        ; INCREMENT CURSOR AND DETECT BOTH END OF LINE AND END OF SCREEN
                        ;
1-9D4 A9 00         INCRSB  LDA     #0      NON-EXTEND ENTRY POINT
1-9D6 F0 02                 BEQ     INCRSC
1-9D8 A9 9B         INCRSR  LDA     #$9B    SPECIAL CASE ELIMINATOR
1-9DA 85 7D         INCRSC  STA     INSDAT
1-9DC E6 63         INCRSA  INC     LOGCOL  (INSCHR ENTRY POINT)
1-9DE E6 59                 INC     COLCRS
1-9E0 D0 02                 BNE     INCRS2  DO HIGH BYTE
1-9E2 E6 56                 INC     COLCRS+1
1-9E4 A5 55         INCRS2  LDA     COLCRS  TEST END OF LINE
1-9E6 A6 57                 LDX     DINDEX
1-9E8 DD BD FE              CMP     COLUMN,X        TEST TABLED VALUE FOR ALL SCREEN MODES
1-9EB F0 0B                 BEQ     INC2A   DO CR IF EQUAL
1-9ED E0 00                 CPX     #0      MODE 0?
1-9EF D0 06                 BNE     INCRS3  IF NOT, JUST RETURN
1-9F1 C5 53                 CMP     RMARGN  TEST AGAINST RMARGN
1-9F3 F0 02                 BEQ     INCRS3  EQUAL IS OK
1-9F5 B0 01                 BCS     INC2A   IF GREATER THAN, DO CR
1-9F7 60            INCRS3  RTS
1-9F8 E0 0B         INC2A   CPX     #8              CHECK MODE
1-9FA 90 04                 BCC     DOCR1           NOT 320X1 SO DO IT
1-9FC A5 56                 LDA     COLCRS+1        TEST MSD
1-9FE F0 F7                 BEQ     INCRS3          ONLY AT 64 SO DON'T DO IT
1-A00 A5 57         DOCR1   LDA     DINDEX  DONT MESS WITH LOGMAP IF NO MODE ZERO
1-A02 D0 30                 BNE     DOCR
1-A04 A5 63                 LDA     LOGCOL  TEST LINE OVERRUN
1-A06 C9 51                 CMP     #81
1-A08 90 0A                 BCC     DOCR1B  IF LESS THAN 81 IT IS DEFINITELY NOT LINE 3
1-A0A A5 7D                 LDA     INSDAT
1-A0C F0 26                 BEQ     DOCR    ONLY DO LOG LINE OVERFLOW IF INSDAT <>0
1-A0E 20 30 FA              JSR     DOCRWS  LOG LINE OVERFLOW IS SPECIAL CASE
1-A11 4C 77 FA              JMP     INCRS1  RETURN
1-A14 20 34 FA      DOCR1B  JSR     DOCR    GET IT OVER WITH
1-A17 A5 54                 LDA     ROWCRS
1-A19 18                    CLC             TEST LOGICAL LINE BIT MAP
1-A1A 69 78                 ADC     #120
1-A1C 20 25 FB              JSR     BITGET
1-A1F 90 0B                 BCC     DOCR1A  DONT EXTEND IF OVERRUN IS INTO MIDDLE OF LOG LINE
1-A21 A5 7D                 LDA     INSDAT  DONT EXTEND IF INSDAT IS ZERO
1-A23 F0 04                 BEQ     DOCR1A  (INSCHR SPECIAL CASE)
1-A25 18                    CLC             INSERT "0" INTO BIT MAP
1-A26 20 A5 FB              JSR     INSLIA
1-A29 4C DD FB      DOCR1A  JMP     DOLCOL  CONVERT ROW AND COL TO LOGCOL AND RETURN
1-A2C A9 00         NOSCRL  LDA     #0      DOCR WITHOUT SCROLL
1-A2E F0 02                 BEQ     NOSCR1  (UNCONDITIONAL)
1-A30 A9 9B         DOCRWS  LDA     #$9B    DOCR WITH SCROLLING (NORMAL MODE)
1-A32 85 7D         NOSCR1  STA     INSDAT
1-A34 20 E4 FC      DOCR    JSR     COLCR   PLACE COLCRS AT LEFT EDGE
1-A37 A9 00                 LDA     #0
1-A39 85 56                 STA     COLCRS+1
1-A3B E6 54                 INC     ROWCRS
1-A3D A6 57         DOCR2   LDX     DINDEX
1-A3F A0 18                 LDY     #24     SET UP SCROLL LOOP COUNTER
1-A41 24 7B                 BIT     SHFLG
1-A43 10 05                 BPL     DOCR2A  BRANCH IF NORMAL
1-A45 A0 04                 LDY     #4
1-A47 98                    TYA
1-A48 D0 03                 BNE     DOCR2B  (UNCONDITIONAL)
```

```
1A4A BD 99 FE     DOCR2A  LDA     NOROWS,X GET NO OF ROWS
1A4D C5 54        DOCR2B  CMP     ROWCRS
1A4F D0 26                BNE     INCRS1
1A51 8C 9D 02             STY     HOLD3
1A54 8A                   TXA             DONT SCROLL IF MODE <> 0
1A55 D0 20                BNE     INCRS1
1A57 A5 7D                LDA     INSDAT  OR IF INSDAT = 0
1A59 F0 1C                BEQ     INCRS1
                          LDA     INSDAT  IF INSDAT <> $9B THEN ROLL IN A 0
1A5B C9 9B                CMP     #$9B    TO EXTEND BOTTOM LOGICAL LINE
1A5D 38                   SEC
1A5E F0 01                BEQ     DOCR4B
1A60 18                   CLC
1A61 20 AC FB     DOCR4B  JSR     SCROLL  LOOP BACK TO HERE IF >1 SCROLLS
1A64 EE BB 02             INC     SCRFLG
1A67 C6 6C                DEC     BUFSTR  ROWS MOVE UP SO BUFSTR SHOULD TOO
1A69 CE 9D 02             DEC     HOLD3
1A6C AD B2 02             LDA     LOGMAP
1A6F 38                   SEC             FOR PARTIAL LINES, ROLL IN A "1"
1A70 10 EF                BPL     DOCR4B  AGAIN IF PARTIAL LOGICAL LINE
1A72 AD 9D 02             LDA     HOLD3   PLACE CURSOR AT NEW LINE NEAR THE BOTTOM
1A75 85 54                STA     ROWCRS
1A77 4C DD FB     INCRS1  JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN

                  ;SUBEND: SUBTRACT ENDPT FROM ROWAC OR COLAC, (X=0 OR 2)

1A7A 38           SUBEND  SEC
1A7B B5 70                LDA     ROWAC,X
1A7D E5 74                SBC     ENDPT
1A7F 95 70                STA     ROWAC,X
1A81 B5 71                LDA     ROWAC+1,X
1A83 E5 75                SBC     ENDPT+1
1A85 95 71                STA     ROWAC+1,X
1A87 60                   RTS

                  ;RANGE: DO CURSOR RANGE TEST, IF ERROR, POP STACK TWICE AND JMP RETURN
                  ;       (ERANGE IS EDITOR ENTRY POINT AND TEST IF EDITOR IS OPEN.
                  ;        IF IT ISNT IT OPENS THE EDITOR AND CONTINUES)

1A88 AD BF 02     ERANGE  LDA     BOTSCR  IN BOTSCR=4
1A8B C9 04                CMP     #4
1A8D F0 07                BEQ     RANGE   THEN IT IS IN MIXED MODE AND OK
1A8F A5 57                LDA     DINDEX  IF MODE = 0
1A91 F0 03                BEQ     RANGE   THEN IT IS IN EDITOR MODE AND OK
1A93 20 FC F3             JSR     EOPEN   IF NOT, OPEN EDITOR
1A96 A9 27        RANGE   LDA     #39     ***** RANGE CHECK RMARGN ***** SET UP AC
1A98 C5 53                CMP     RMARGN  ***** RANGE CHECK RMARGN ***** COMPARE
1A9A B0 02                BCS     RANGE3  ***** RANGE CHECK RMARGN ***** BRANCH OK
1A9C 85 53                STA     RMARGN  ***** RANGE CHECK RMARGN ***** BAD SO STORE 39
1A9E A6 57        RANGE3  LDX     DINDEX
1AA0 BD 99 FE             LDA     NOROWS,X        CHECK ROWS
1AA3 C5 54                CMP     ROWCRS
1AA5 90 2A                BCC     RNGERR          (ERROR IF TABLE,OE,ROWCRS)
1AA7 F0 28                BEQ     RNGERR
1AA9 E0 08                CPX     #8      CHECK FOR 320X1
1AAB D0 0A                BNE     RANGE1  SPECIAL CASE IT
1AAD A5 56                LDA     COLCRS+1
1AAF F0 13                BEQ     RNGOK   IF HIGH BYTE IS 0, COL IS OK
```

```
1AB1 C9 01                      CMP     #1
1AB3 D0 1C                      BNE     RNGERR  IF >1, BAD
1AB5 F0 04                      BEQ     RNGER2  IF 1, GO CHECK LOW BYTE
1AB7 A5 56              RANGE1  LDA     COLCRS  FOR OTHERS, NON-ZERO HIGH BYTE IS BAD
1AB9 D0 16                      BNE     RNGERR
1ABB BD BD FE           RANGE2  LDA     COLUMN,X CHECK LOW BYTE
1ABE C5 55                      CMP     COLCRS
1AC0 90 0F                      BCC     RNGERR
1AC2 F0 0D                      BEQ     RNGERR
1AC4 A9 01              RNGOK   LDA     #SUCCES SET STATUS OK
1AC6 85 4C                      STA     DSTAT
1AC8 A9 80                      LDA     #BRKABT PREPARE BREAK ABORT STATUS
1ACA A6 11                      LDX     BRKKEY  CHECK BREAK KEY FLAG
1ACC 85 11                      STA     BRKKEY  'CLEAR' BREAK
1ACE F0 06                      BEQ     RNGER2  IF BREAK, QUIT IMMEDIATELY AND RETURN TO CIO
1AD0 60                         RTS
1AD1 20 D6 E7           RNGERR  JSR     HOME    ON RANGE ERROR, BRING CURSOR BACK
1AD4 A9 8D                      LDA     #CRSROR SHOW CURSOR OVERRANGE ERROR
1AD6 85 4C              RNGER2  STA     DSTAT
1AD8 68                 RNGER1  PLA             RESTORE STACK (THIS ROUTINE IS ALWAYS 1 LEVEL
1AD9 68                         PLA             AWAY FROM RETURN TO CIO)
1ADA A5 7B                      LDA     SWPFLG  IF SWAPPED, SWAP BACK
1ADC 10 03                      BPL     RETUR3
1ADE 20 B9 FC                   JSR     SWAPA   AND DONT DO RETUR1
1AE1 4C 34 E6           RETUR3  JMP     RETUR1  RETURN TO CIO

                        ;
                        ; OFFCRS: RESTORE OLD DATA UNDER CURSOR SO IT CAN BE MOVED
                        ;
1AE4 A0 00              OFFCRS  LDY     #0
1AE6 A5 5D                      LDA     OLDCHR
1AE8 91 5E                      STA     (OLDADR),Y
1AEA 60                         RTS

                        ;
                        ; BITMAP ROUTINES:
                        ;
                        ; BITCON: PUT MASK IN BITMSK AND INDEX IN X
                        ; BITPUT: PUT CARRY INTO BITMAP
                        ; BITROL: ROL CARRY INTO BOTTOM OF BITMAP (SCROLL)
                        ; BITSET: SET PROPER BIT
                        ; BITCLR: CLEAR PROPER BIT
                        ; BITGET: RETURN CARRY SET IF BIT IS THERE
                        ; LOGGET: DO BITGET FOR LOGMAP INSTEAD OF TABMAP
                        ;
1AEB 48                 BITCON  PHA
1AEC 29 07                      AND     #7
1AEE AA                         TAX             GET MASK
1AEF BD B9 FE                   LDA     MASKTB,X
1AF2 85 6E                      STA     BITMSK
1AF4 68                         PLA             PROCESS INDEX
1AF5 4A                         LSR     A
1AF6 4A                         LSR     A
1AF7 4A                         LSR     A
1AF8 AA                         TAX
1AF9 60                         RTS
```

```
FAFA 2E B4 02     BITROL  ROL     LOGMAP+2
FAFD 2E B3 02             ROL     LOGMAP+1
FB00 2E B2 02             ROL     LOGMAP
FB03 60                   RTS

FB04 90 0C        BITPUT  BCC     BITCLR  AND RETURN
                                          OTHERWISE FALL THROUGH TO BITSET AND RETURN
FB06 20 EB FA     BITSET  JSR     BITCON
FB09 BD A3 02             LDA     TABMAP,X
FB0C 05 6E                ORA     BITMSK
FB0E 9D A3 02             STA     TABMAP,X
FB11 60                   RTS

FB12 20 EB FA     BITCLR  JSR     BITCON
FB15 A5 6E                LDA     BITMSK
FB17 49 FF                EOR     #$FF
FB19 3D A3 02             AND     TABMAP,X
FB1C 9D A3 02             STA     TABMAP,X
FB1F 60                   RTS

FB20 A5 54        LOGGET  LDA     ROWCRS
FB22 18           LO1GET  CLC
FB23 69 78        LO2GET  ADC     #120
FB25 20 EB FA     BITGET  JSR     BITCON
FB28 18                   CLC
FB29 BD A3 02             LDA     TABMAP,X
FB2C 25 6E                AND     BITMSK
FB2E F0 01                BEQ     BITGE1
FB30 38                   SEC
FB31 60           BITGE1  RTS


                  ; INATAC: INTERNAL(CHAR) TO ATASCII(ATACHR) CONVERSION
FB32 AD FA 02     INATAC  LDA     CHAR
FB35 A4 57                LDY     DINDEX  IF GRAPHICS MODES
FB37 C0 03                CPY     #3
FB39 B0 0F                BCS     INATA1  THEN DONT CHANGE CHAR
FB3B 2A                   ROL     A
FB3C 2A                   ROL     A
FB3D 2A                   ROL     A
FB3E 2A                   ROL     A
FB3F 29 03                AND     #3
FB41 AA                   TAX
FB42 AD FA 02             LDA     CHAR
FB45 29 9F                AND     #$9F
FB47 1D FA FE             ORA     INTATA,X
FB4A 8D FB 02     INATA1  STA     ATACHR
FB4D 60                   RTS


                  ; MOVLIN: MOVE 40 BYTES AT FRMADR TO TOADR SAVING OLD TOADR
                          DATA IN THE LINBUF. THEN MAKE NEXT FRMADR
                          BE AT LINBUF FOR NEXT TRANSFER & TOADR=TOADR+40

FB4E A9 02        MOVLIN  LDA     #LINBUF/256     SET UP ADRESS=LINBUF=$247
```

```
F850 85 65              STA     ADRESS+1
F852 A9 47              LDA     #LINBUF
F854 85 64              STA     ADRESS
F856 A0 27              LDY     #39
F858 B1 66      MOVLI1  LDA     (TOADR),Y       SAVE TO DATA
F85A 85 50              STA     TMPCHR
F85C B1 68              LDA     (FRMADR),Y      STORE DATA
F85E 91 66              STA     (TOADR),Y
F860 A5 50              LDA     TMPCHR
F862 91 64              STA     (ADRESS),Y
F864 88                 DEY
F865 10 F1              BPL     MOVLI1
F867 A5 65              LDA     ADRESS+1        SET UP FRMADR=LAST LINE
F869 85 69              STA     FRMADR+1
F86B A5 64              LDA     ADRESS
F86D 85 68              STA     FRMADR
F86F 18                 CLC                     ADDD 40 TO TOADR
F870 A5 66              LDA     TOADR
F872 69 28              ADC     #40
F874 85 66              STA     TOADR
F876 90 02              BCC     MOVLI2
F878 E6 67              INC     TOADR+1
F87A 60         MOVLI2  RTS

                ; EXTEND: EXTEND BIT MAP FROM ROWCRS (EXTEND LOGICAL LINE
                ;
F87B 08         EXTEND  PHP             SAVE CARRY
F87C A0 17              LDY     #23
F87E 98         EXTEN1  TYA
F87F 20 22 FB           JSR     LOIGET
F882 08                 PHP
F883 98                 TYA
F884 18                 CLC
F885 69 79              ADC     #121
F887 28                 PLP
F888 20 04 FB           JSR     BITPUT
F88B 88         EXTEN3  DEY
F88C 30 04              BMI     EXTEN4
F88E C4 54              CPY     ROWCRS
F890 B0 EC              BCS     EXTEN1
F892 A5 54      EXTEN4  LDA     ROWCRS
F894 18                 CLC
F895 69 78              ADC     #120
F897 28                 PLP
F898 4C 04 FB           JMP     BITPUT  STORE NEW LINE'S BIT AND RETURN

                ; CLRLIN: CLEAR LINE CURSOR IS ON
                ;
F89B A5 52      CLRLIN  LDA     LMARGN
F89D 85 55              STA     COLCRS
F89F 20 47 F9           JSR     CONVRT
F8A2 A0 27              LDY     #39
F8A4 A9 00              LDA     #0
F8A6 91 64      CLRLI1  STA     (ADRESS),Y
F8A8 88                 DEY
F8A9 10 FB              BPL     CLRLI1
```

```
FBAB 60                 RTS

                        ;SCROLL: SCROLL SCREEN

FBAC 20 FA FA   SCROLL  JSR     BITROL  ROLL IN CARRY
FBAF A5 58              LDA     SAVMSC  SET UP WORKING REGISTERS
FBB1 85 64              STA     ADRESS
FBB3 A5 59              LDA     SAVMSC+1
FBB5 85 65              STA     ADRESS+1
FBB7 A0 28      SCROL1  LDY     #40     LOOP
FBB9 B1 64              LDA     (ADRESS),Y
FBBB A6 6A              LDX     RAMTOP  TEST FOR LAST LINE
FBBD CA                 DEX
FBBE E4 65              CPX     ADRESS+1
FBC0 D0 08              BNE     SCROL2
FBC2 A2 D7              LDX     #$D7
FBC4 E4 64              CPX     ADRESS
FBC6 B0 02              BCS     SCROL2
FBC8 A9 00              LDA     #0      YES SO STORE ZERO DATA FOR THIS ENTIRE LINE
FBCA A0 00      SCROL2  LDY     #0
FBCC 91 64              STA     (ADRESS),Y
FBCE E6 64              INC     ADRESS
FBD0 D0 E5              BNE     SCROL1
FBD2 E6 65              INC     ADRESS+1
FBD4 A5 65              LDA     ADRESS+1
FBD6 C5 6A              CMP     RAMTOP
FBD8 D0 DD              BNE     SCROL1
FBDA 4C DD FB           JMP     DOLCOL  AND RETURN

                        ;DOLCOL: DO LOGICAL COLUMN FROM BITMAP AND COLCRS

FBDD A9 00      DOLCOL  LDA     #0      START WITH ZERO
FBDF 85 63              STA     LOGCOL
FBE1 A5 54              LDA     ROWCRS
FBE3 85 51              STA     HOLD1
FBE5 A5 51      DOLCO1  LDA     HOLD1   ADD IN ROW COMPONENT
FBE7 20 22 FB           JSR     LOIGET
FBEA B0 0C              BCS     DOLCO2  FOUND BEGINNING OF LINE
FBEC A5 63              LDA     LOGCOL  ADD 40 AND LOOK BACK ONE
FBEE 18                 CLC
FBEF 69 28              ADC     #40
FBF1 85 63              STA     LOGCOL
FBF3 C6 51              DEC     HOLD1   UP ONE LINE
FBF5 4C E5 FB           JMP     DOLCO1
FBF8 18         DOLCO2  CLC             ADD IN COLCRS
FBF9 A5 63              LDA     LOGCOL
FBFB 65 55              ADC     COLCRS
FBFD 85 63              STA     LOGCOL
FBFF 60                 RTS

                        ;DOBUFC: COMPUTE BUFFER COUNT AS THE NUMBER OF BYTES FROM
                        ;       BUFSTR TO END OF LOGICAL LINE WITH TRAILING SPACES REMOVED

FC00 20 9D FC   DOBUFC  JSR     PHACRS
```

```
FC03 A5 63              LDA     LOGCOL
FC05 48                 PHA
FC06 A5 6C              LDA     BUFSTR  START
FC08 85 54              STA     ROWCRS
FC0A A5 6D              LDA     BUFSTR+1
FC0C 85 55              STA     COLCRS
FC0E A9 01              LDA     #1
FC10 85 6B              STA     BUFCNT
FC12 A2 17      DOBUF1  LDX     #23     NORMAL
FC14 A5 7B              LDA     SWPFLG  IF SWAPPED, ROW 3 IS THE LAST LINE ON SCREEN
FC16 10 02              BPL     DOB1
FC18 A2 03              LDX     #3
FC1A E4 54      DOB1    CPX     ROWCRS  TEST IF CRSR IS AT LAST SCREEN POSITION
FC1C D0 0B              BNE     DOBUF4
FC1E A5 55              LDA     COLCRS
FC20 C5 53              CMP     RMARGN
FC22 D0 05              BNE     DOBUF4
FC24 E6 6B              INC     BUFCNT  YES SO FAKE INCRSR TO AVOID SCROLLING
FC26 4C 39 FC           JMP     DOBUF2
FC29 20 D4 F9   DOBUF4  JSR     INCRSR
FC2C E6 6B              INC     BUFCNT
FC2E A5 63              LDA     LOGCOL
FC30 C5 52              CMP     LMARGN
FC32 D0 DE              BNE     DOBUF1  NOT YET EOL
FC34 C6 54              DEC     ROWCRS  BACK UP ONE INCRSR
FC36 20 99 F7           JSR     CRSRLF
FC39 20 A2 F5   DOBUF2  JSR     GETPLT  TEST CURRENT COLUMN FOR NON-ZERO DATA
FC3C D0 17              BNE     DOBUF4  QUIT IF NON-ZERO
FC3E C6 6B              DEC     BUFCNT  DECREMENT COUNTER
FC40 A5 63              LDA     LOGCOL  BEGINNING OF LOGICAL LINE YET?
FC42 C5 52              CMP     LMARGN
FC44 F0 0F              BEQ     DOBUF4  YES, SO QUIT
FC46 20 99 F7           JSR     CRSRLF  BACK UP CURSOR
FC49 A5 55              LDA     COLCRS  IF LOGCOL=RMARGN, GO UP 1 ROW
FC4B C5 53              CMP     RMARGN
FC4D D0 02              BNE     DOBUF3
FC4F C6 54              DEC     ROWCRS
FC51 A5 6B      DOBUF3  LDA     BUFCNT
FC53 D0 E4              BNE     DOBUF2  LOOP UNLESS BUFCNT JUST WENT TO ZERO
FC55 68         DOBUF4  PLA
FC56 85 63              STA     LOGCOL
FC58 20 AB FC           JSR     PLACRS
FC5B 60                 RTS

;
;       STRBEG: MOVE BUFSTR TO BEGINNING OF LOGICAL LINE
;
FC5C 20 DD FB   STRBEG  JSR     DOLCOL  USE DOLCOL TO POINT HOLD1 AT BOL
FC5F A5 51              LDA     HOLD1
FC61 85 6C              STA     BUFSTR
FC63 A5 52              LDA     LMARGN
FC65 85 6D              STA     BUFSTR+1
FC67 60                 RTS
;
;
;
```

```
                        ; DELTIM: TIME TO DELETE A LINE IF IT IS EMPTY AND AN EXTENSION
FC68 A5 63      DELTIA  LDA     LOGCOL  IF LOGCOL<>LMARGN
FC6A C5 52              CMP     LMARGN  THEN DONT MOVE UP ONE
FC6C D0 02              BNE     DELTIB  LINE BEFORE TESTING DELTIM
FC6E C6 54              DEC     ROWCRS
FC70 20 DD F8   DELTIB  JSR     DOLCOL
FC73 A5 63      DELTIM  LDA     LOGCOL  TEST FOR EXTENSION
FC75 C5 52              CMP     LMARGN
FC77 F0 13              BEQ     DELTI3  NO
FC79 20 47 F9           JSR     CONVRT
FC7C A5 53              LDA     RMARGN  SET UP COUNT
FC7E 38                 SEC
FC7F E5 52              SBC     LMARGN
FC81 A8                 TAY
FC82 B1 64      DELTI1  LDA     (ADRESS),Y
FC84 D0 06              BNE     DELTI3  FOUND A NON-0 SO QUIT AND RETURN
FC86 88                 DEY
FC87 10 F9              BPL     DELTI1
FC89 4C DB F8   DELTI2  JMP     DELLIB  DELETE A LINE AND RETURN
FC8C 60         DELTI3  RTS

                        ; TSTCTL: SEARCH CNTRLS TABLE TO SEE IF ATACHR IS A CNTL CHAR
FC8D A2 2D      TSTCTL  LDX     #45     PREPARE TO SEARCH TABLE
FC8F BD C6 FE   TSTCT1  LDA     CNTRLS,X
FC92 CD FB 02           CMP     ATACHR
FC95 F0 05              BEQ     TSTCT2
FC97 CA                 DEX
FC98 CA                 DEX
FC99 CA                 DEX
FC9A 10 F3              BPL     TSTCT1
FC9C 60         TSTCT2  RTS

                        ; PUSH ROWCRS,COLCRS AND COLCRS+1
FC9D A2 02      PHACRS  LDX     #2
FC9F B5 54      PHACR1  LDA     ROWCRS,X
FCA1 9D BB 02           STA     TMPROW,X
FCA4 CA                 DEX
FCA5 10 F8              BPL     PHACR1
FCA7 60                 RTS

                        ; PULL COLCRS+1,COLCRS,AND ROWCRS
FCA8 A2 02      PLACRS  LDX     #2
FCAA BD BB 02   PLACR1  LDA     TMPROW,X
FCAD 95 54              STA     ROWCRS,X
FCAF CA                 DEX
FCB0 10 F8              BPL     PLACR1
FCB2 60                 RTS

                        ; SWAP: IF MIXED MODE, SWAP TEXT CURSORS WITH REGULAR CURSORS
```

```
                            ;
FCB3 20 B9 FC      SWAP     JSR    SWAPA    THIS ENTRY POINT DOES RETUR1
FCB6 4C 34 F6               JMP    RETUR1
FCB9 AD BF 02      SWAPA    LDA    BOTSCR
FCBC C9 18                  CMP    #24
FCBE F0 17                  BEQ    SWAP3
FCC0 A2 0B                  LDX    #11
FCC2 B5 54         SWAP1    LDA    ROWCRS,X
FCC4 48                     PHA
FCC5 BD 90 02               LDA    TXTROW,X
FCC8 95 54                  STA    ROWCRS,X
FCCA 68                     PLA
FCCB 9D 90 02               STA    TXTROW,X
FCCE CA                     DEX
FCCF 10 F1                  BPL    SWAP1
FCD1 A5 7B                  LDA    SWPFLG
FCD3 49 FF                  EOR    #$FF
FCD5 85 7B                  STA    SWPFLG
FCD7 60            SWAP3    RTS
                            ;
                            ; CLICK: MAKE CLICK THROUGH KEYBOARD SPEAKER
                            ;
FCD8 A2 7F         CLICK    LDX    #$7F
FCDA 8E 1F D0      CLICK1   STX    CONSOL
FCDD 8E 0A D4               STX    WSYNC
FCE0 CA                     DEX
FCE1 10 F7                  BPL    CLICK1
FCE3 60                     RTS
                            ;
                            ; COLCR: PUTS EITHER 0 OR LMARGN INTO COLCRS BASED ON MODE AND SWPFLG
                            ;
FCE4 A9 00         COLCR    LDA    #0
FCE6 A6 7B                  LDX    SWPFLG
FCE8 D0 04                  BNE    COLCR1
FCEA A6 57                  LDX    DINDEX
FCEC D0 02                  BNE    COLCR2
FCEE A5 52         COLCR1   LDA    LMARGN
FCF0 85 55         COLCR2   STA    COLCRS
FCF2 60                     RTS
                            ;
                            ; PUTMSC: PUT SAVMSC INTO ADRESS
                            ;
FCF3 A5 58         PUTMSC   LDA    SAVMSC   SET UP ADDRESS
FCF5 85 64                  STA    ADRESS
FCF7 A5 59                  LDA    SAVMSC+1
FCF9 85 65                  STA    ADRESS+1
FCFB 60                     RTS
                            ;
```

```
                        ;
                        ; DRAW -- DRAW A LINE FROM OLDROW,OLDCOL TO NEWROW,NEWCOL
                        ; (THE AL MILLER METHOD FROM BASKETBALL)
 FCFC A2 00             DRAW    LDX     #0
 FCFE A5 22                     LDA     ICCOMZ  TEST COMMAND: $11=DRAW  $12=FILL
 FD00 C9 11                     CMP     #$11
 FD02 F0 08                     BEQ     DRAWA
 FD04 C9 12                     CMP     #$12    TEST FILL
 FD06 F0 03                     BEQ     DRAWB   YES
 FD08 A0 84                     LDY     #INVALID NO, SO RETURN INVALID COMMAND
 FD0A 60                        RTS
 FD0B E8                DRAWB   INX
 FD0C 8E B7 02          DRAWA   STX     FILFLG
 FD0F A5 54                     LDA     ROWCRS  PUT CURSOR INTO NEWROW,NEWCOL
 FD11 85 60                     STA     NEWROW
 FD13 A5 55                     LDA     COLCRS
 FD15 85 61                     STA     NEWCOL
 FD17 A5 56                     LDA     COLCRS+1
 FD19 85 62                     STA     NEWCOL+1
 FD1B A9 01                     LDA     #1
 FD1D 85 79                     STA     ROWINC  SET UP INITIAL DIRECTIONS
 FD1F 85 7A                     STA     COLINC
 FD21 38                        SEC
 FD22 A5 60                     LDA     NEWROW          DETERMINE DELTA ROW
 FD24 E5 5A                     SBC     OLDROW
 FD26 85 76                     STA     DELTAR
 FD28 B0 00                     BCS     DRAW1           DO DIRECTION AND ABSOLUTE VALUE
 FD2A A9 FF                     LDA     #$FF            BORROW WAS ATTEMPTED
 FD2C 85 79                     STA     ROWINC          SET DIRECTION=DOWN
 FD2E A5 76                     LDA     DELTAR
 FD30 49 FF                     EOR     #$FF            DELTAR = |DELTAR|
 FD32 18                        CLC
 FD33 69 01                     ADC     #1
 FD35 85 76                     STA     DELTAR
 FD37 38                DRAW1   SEC
 FD38 A5 61                     LDA     NEWCOL          NOW DELTA COLUMN
 FD3A E5 5B                     SBC     OLDCOL
 FD3C 85 77                     STA     DELTAC
 FD3E A5 62                     LDA     NEWCOL+1        TWO-BYTE QUANTITY
 FD40 E5 5C                     SBC     OLDCOL+1
 FD42 85 78                     STA     DELTAC+1
 FD44 B0 14                     BCS     DRAW2           DIRECTION AND ABSOLUTE VALUE
 FD46 A9 FF                     LDA     #$FF            BORROW WAS ATTEMPTED
 FD48 85 7A                     STA     COLINC          SET DIRECTION = LEFT
 FD4A A5 77                     LDA     DELTAC
 FD4C 49 FF                     EOR     #$FF            DELTAC = |DELTAC|
 FD4E 85 77                     STA     DELTAC
 FD50 A5 78                     LDA     DELTAC+1
 FD52 49 FF                     EOR     #$FF
 FD54 85 78                     STA     DELTAC+1
 FD56 E6 77                     INC     DELTAC          ADD ONE FOR TWOS COMPLEMENT
 FD58 D0 02                     BNE     DRAW2
 FD5A E6 78                     INC     DELTAC+1
 FD5C A2 02             DRAW2   LDX     #2      ZERO RAM FOR DRAW LOOP
 FD5E A0 00                     LDY     #0
 FD60 84 73                     STY     COLAC+1
 FD62 98                DRAW3A  TYA
 FD63 95 70                     STA     ROWAC,X
 FD65 B5 5A                     LDA     OLDROW,X
```

```
                        STA     ROWCRS,X
                        DEX
                        BPL     DRAW3A
                        LDA     DELTAC          ;FIND LARGER ONE (ROW OR COL)
                        STA     COUNTR          ;(ALSO SAVE COUNTR AND ENDPT)
                        STA     ENDPT
                        INX                     ;MAKE X 0
                        TAY
                        LDA     DELTAR
                        STA     COUNTR+1
                        STA     ENDPT+1
                        BNE     DRAW3           ;AUTOMATICALLY LARGER IF MSD>0
                        LDA     DELTAC
                        CMP     DELTAR          ;LOW COL >LOW ROW?
                        BCS     DRAW3           ;YES
                        LDA     DELTAR
                        LDX     #2
                        TAY
DRAW3                   TYA                     ;PUT IN INITIAL CONDITIONS
                        STA     COUNTR
                        STA     ENDPT
                        PHA                     ;SAVE AC
                        LDA     ENDPT+1         ;PUT LSB OF HIGH BYTE
                        LSR     A               ;INTO CARRY
                        PLA                     ;RESTORE AC
                        ROR     A               ;ROR THE 9 BIT ACUMULATOR
                        STA     ROWAC,X
DRAW4A                  LDA     COUNTR          ;TEST ZERO
                        ORA     COUNTR+1
                        BNE     DRAW11          ;IF COUNTER IS ZERO, LEAVE DRAW
                        JMP     DRAW10
DRAW11                  CLC                     ;ADD ROW TO ROWAC (PLOT LOOP)
                        LDA     ROWAC
                        ADC     DELTAR
                        STA     ROWAC
                        BCC     DRAW5
                        INC     ROWAC+1
DRAW5                   LDA     ROWAC+1         ;COMPARE ROW TO ENDPOINT
                        CMP     ENDPT+1         ;IF HIGH BYTE OF ROW IS .LT. HIGH
                        BCC     DRAW6           ;BYTE OF ENDPT, BLT TO COLUMN
                        BNE     DRAW5A
                        LDA     ROWAC           ;HIGH BYTE
                        CMP     ENDPT           ;ALSO BLT
                        BCC     DRAW6           ;OR SO MOVE POINT
DRAW5A                  CLC
                        LDA     ROWCRS
                        ADC     ROWINC
                        STA     ROWCRS          ;AND SUBTRACT ENDPT FROM ROWAC
                        LDX     #0
                        JSR     SUBEND
DRAW6                   CLC                     ;DO SAME FOR COLUMN (DOUBLE BYTE ADD)
                        LDA     COLAC           ;ADD
                        ADC     DELTAC
                        STA     COLAC
                        LDA     COLAC+1
                        ADC     DELTAC+1
                        STA     COLAC+1
                        CMP     ENDPT+1         ;COMPARE HIGH BYTE
                        BCC     DRAW8
                        BNE     DRAW6A
```

```
FDD1 A5 72               LDA   COLAC           COMPARE LOW BYTE
FDD3 C5 74               CMP   ENDPT
FDD5 90 1F               BCC   DRAW8
FDD7 24 7A      DRAW6A   BIT   COLINC  + OR - ?
FDD9 10 10               BPL   DRAW6B
FDDB C6 55               DEC   COLCRS          DO DOUBLE BYTE DECREMENT
FDDD A5 55               LDA   COLCRS
FDDF C9 FF               CMP   #$FF
FDE1 D0 0E               BNE   DRAW7
FDE3 A5 56               LDA   COLCRS+1
FDE5 F0 0A               BEQ   DRAW7           DONT DEC IF ZERO
FDE7 C6 56               DEC   COLCRS+1
FDE9 10 06               BPL   DRAW7   (UNCONDITIONAL)
FDEB E6 55      DRAW6B   INC   COLCRS          DO DOUBLE BYTE INCREMENT
FDED D0 02               BNE   DRAW7
FDEF E6 56               INC   COLCRS+1
FDF1 A2 02      DRAW7    LDX   #2              AND SUBTRACT ENDPT FROM COLAC
FDF3 20 7A FA            JSR   SUBEND
FDF6 20 96 FA   DRAW8    JSR   RANGE
FDF9 20 E0 F5            JSR   OUTPLT          PLOT POINT
FDFC AD B7 02            LDA   FILFLG  TEST RIGHT FILL
FDFF F0 2F               BEQ   DRAW9
FE01 20 9D FC            JSR   PHACRS
FE04 AD FB 02            LDA   ATACHR
FE07 8D BC 02            STA   HOLD4
FE0A A5 54      DRAW8A   LDA   ROWCRS  SAVE ROW IN CASE OF CR
FE0C 48                  PHA
FE0D 20 DC F9            JSR   INCRSA  POSITION CURSOR ONE PAST DOT
FE10 68                  PLA           RESTORE ROWCRS
FE11 85 54               STA   ROWCRS
FE13 20 96 FA   DRAW8C   JSR   RANGE
FE16 20 A2 F5            JSR   GETPLT  GET DATA
FE19 D0 0C               BNE   DRAW8B  STOP IF NON-ZERO DATA IS ENCOUNTERED
FE1B AD FD 02            LDA   FILDAT  FILL DATA
FE1E 8D FB 02            STA   ATACHR
FE21 20 E0 F5            JSR   OUTPLT  DRAW IT
FE24 4C 0A FE            JMP   DRAW8A  LOOP
FE27 AD BC 02   DRAW8B   LDA   HOLD4
FE2A 8D FB 02            STA   ATACHR
FE2D 20 A8 FC            JSR   PLACRS
FE30 38         DRAW9    SEC                   DO DOUBLE BYTE SUBTRACT
FE31 A5 7E               LDA   COUNTR
FE33 E9 01               SBC   #1
FE35 85 7E               STA   COUNTR
FE37 A5 7F               LDA   COUNTR+1
FE39 E9 00               SBC   #0
FE3B 85 7F               STA   COUNTR+1
FE3D 30 03               BMI   DRAW10
FE3F 4C 90 FD            JMP   DRAW4A
FE42 4C 34 F6   DRAW10   JMP   RETUR1
```

```
                        ;
                        ; TABLES
                        ;
                        ; MEMORY ALLOCATION
E45 18 10 0A    ALOCAT  .BYTE   24,16,10,10,16,28,52,100,196,196,196,196
E48 0A 10 1C
E4B 34 64 C4
E4E C4 C4 C4
                        ;
                        ; NUMBER OF DISPLAY LIST ENTRIES
                        ;
E51 17 17 0B    NUMDLE  .BYTE   23,23,11,23,47,47,95,95,97,97,97,97
E54 17 2F 2F
E57 5F 5F 61
E5A 61 61 61
E5D 13 13 09    MXDMDE  .BYTE   19,19,9,19,39,39,79,79,65,65,65,65     (EXTENSION OF NUMDLE)
E60 13 27 27
E63 4F 4F 41
E66 41 41 41
                        ;
                        ; ANTIC CODE FROM INTERNAL MODE CONVERSION TABLE
                        ;
                        ;       INTERNAL        ANTIC CODE      DESCRIPTION
                        ;           0               2           40X2X8  CHARACTERS
                        ;           1               6           20X5X8      ""
                        ;           2               7           20X5X16     ""
                        ;           3               8           40X4X8  GRAPHICS
                        ;           4               9           80X2X4      ""
                        ;           5               A           80X4X4      ""
                        ;           6               B           160X2X2     ""
                        ;           7               D           160X4X2     ""
                        ;           8               F           320X2X1     ""
                        ;           9           SAME AS 8 BUT GTIA 'LUM' MODE
                        ;           10          SAME AS 8 BUT GTIA 'COL/LUM REGISTER' MODE
                        ;           11          SAME AS 8 BUT GTIA 'COLOR' MODE
E69 02 04 07    ANCONV  .BYTE   2,4,7,8,9,$A,$B,$D,$F,0,0,0  ;ZEROS FOR RANGE TEST IN PAGETB
E6C 08 09 0A
E6F 0B 0D 0F
E72 0F 0F 0F
                        ;
                        ; PAGE TABLE TELLS WHICH DISPLAY LISTS ARE IN DANGER OF
                        ; CROSSING A 256 BYTE PAGE BOUNDARY
                        ;
E75 00 00 00    PAGETB  .BYTE   0,0,0,0,0,0,0,1,1,1,1,1
E78 00 00 00
E7B 00 01 01
E7E 01 01 01
                        ;
                        ; THIS IS THE NUMBER OF LEFT SHIFTS NEEDED TO MULTIPLY
                        ; COLCRS BY 10, 20, OR 40.  (ROWCRS*10)/(2**DHLINE)
                        ;
E81 02 01 01    DHLINE  .BYTE   2,1,1,0,0,1,1,2,2,2,2,2
```

```
1E84 00 00 01
1E87 01 02 02
1E8A 02 02 02

                        ; COLUMN: NUMBER OF COLUMNS

1E8D 28 14 14    COLUMN  .BYTE   40,20,20,40,80,80,160,160,64,80,80,80   MODE 8 IS SPECIAL CASE
1E90 28 50 50
1E93 A0 A0 40
1E96 50 50 50

                        ; NOROWS: NUMBER OF ROWS

1E99 18 18 0C    NOROWS  .BYTE   24,24,12,24,48,48,96,96,192,192,192,192
1E9C 18 30 30
1E9F 60 60 C0
1EA2 C0 C0 C0

                        ; DIV2TB: HOW MANY RIGHT SHIFTS FOR HCHAR FOR PARTIAL BYTE MODES

1EA5 00 00 00    DIV2TB  .BYTE   0,0,0,2,3,2,3,2,3,1,1,1
1EA8 02 03 02
1EAB 03 02 03
1EAE 01 01 01

                        ; DMASKT: DISPLAY MASK TABLE

1EB1 00 FF F0    DMASKT  .BYTE   $00,$FF,$F0,$0F
1EB4 0F
1EB5 C0 30 0C           .BYTE   $C0,$30,$0C,$03
1EB8 03

                        ; MASKTB: BIT MASK.   (ALSO PART OF DMASKTB! DO NOT SEPARATE)

1EB9 80 40 20    MASKTB  .BYTE   $80,$40,$20,$10,$08,$04,$02,$01
1EBC 10 08 04
1EBF 02 01

1EC1 28 CA 94    COLRTB  .BYTE   $28,$CA,$94,$46,$00
1EC4 46 00

                        ; CNTRLS: CONTROL CODES AND THEIR DISPLACEMENTS INTO THE
                        ;         CONTROL CHARACTER PROCESSORS

1EC6 1B          CNTRLS  .BYTE   $1B
1EC7 79 F7               .WORD   ESCAPE
1EC9 1C                  .BYTE   $1C
```

```
1ECA 7F F7                  .WORD   CRSRUP
1ECC 1D                     .BYTE   $1D
1ECD BC F7                  .WORD   CRSRDN
1ECF 1E                     .BYTE   $1E
1ED0 99 F7                  .WORD   CRSRLF
1ED2 1F                     .BYTE   $1F
1ED3 AA F7                  .WORD   CRSRRT
1ED5 7D                     .BYTE   $7D
1ED6 B9 F7                  .WORD   CLRSCR
1ED8 7E                     .BYTE   $7E
1ED9 E6 F7                  .WORD   BS
1EDB 7F                     .BYTE   $7F
1EDC 10 F8                  .WORD   TAB
1EDE 9B                     .BYTE   $9B
1EDF 30 FA                  .WORD   DOCRWB
1EE1 9C                     .BYTE   $9C
1EE2 D4 F8                  .WORD   DELLIN
1EE4 9D                     .BYTE   $9D
1EE5 A4 F8                  .WORD   INSLIN
1EE7 9E                     .BYTE   $9E
1EE8 32 F8                  .WORD   CLRTAB
1EEA 9F                     .BYTE   $9F
1EEB 2D F8                  .WORD   SETTAB
1EED FD                     .BYTE   $FD
1EEE 0A F9                  .WORD   BELL
1EF0 FE                     .BYTE   $FE
1EF1 6D F8                  .WORD   DELCHR
1EF3 FF                     .BYTE   $FF
1EF4 37 F8                  .WORD   INSCHR

                    ;ATAINT: ATASCI TO INTERNAL TABLE
1EF6 40 00 20       ATAINT  .BYTE   $40,$00,$20,$60
1EF9 60

                    ;INTATA: INTERNAL TO ATASCI TABLE
1EFA 20 40 00       INTATA  .BYTE   $20,$40,$00,$60
1EFD 60

                    ;ATASCI: ATASCII CONVERSION TABLE
1EFE 6C 6A 38       ATASCI  .BYTE   $6C,$6A,$38,$80,$80,$68,$28,$2A LOWER CASE
1F01 80 80 68
1F04 28 2A
1F06 6F 80 70               .BYTE   $6F,$80,$70,$75,$98,$69,$2D,$3D
1F09 75 98 69
1F0C 2D 3D

1F0E 76 80 63               .BYTE   $76,$80,$63,$80,$80,$62,$78,$7A
1F11 80 80 62
1F14 78 7A
1F16 34 80 33               .BYTE   $34,$80,$33,$36,$1B,$35,$32,$31
1F19 36 1B 35
```

```
1-F1C 32 31

1-F1E 2C 20 2E          .BYTE   $2C,$20,$2E,$6E,$80,$6D,$2F,$81
1-F21 6E 80 6D
1-F24 2F 81
1-F26 72 80 65          .BYTE   $72,$80,$65,$79,$7F,$74,$77,$71
1-F29 79 7F 74
1-F2C 77 71

1-F2E 39 80 30          .BYTE   $39,$80,$30,$37,$7E,$38,$3C,$3E
1-F31 37 7E 38
1-F34 3C 3E
1-F36 66 68 64          .BYTE   $66,$68,$64,$80,$82,$67,$73,$61
1-F39 80 82 67
1-F3C 73 61


1-F3E 4C 4A 3A          .BYTE   $4C,$4A,$3A,$80,$80,$4B,$5C,$5E UPPER CASE
1-F41 80 80 4B
1-F44 5C 5E
1-F46 4F 80 50          .BYTE   $4F,$80,$50,$55,$9B,$49,$5F,$7C
1-F49 55 9B 49
1-F4C 5F 7C

1-F4E 56 80 43          .BYTE   $56,$80,$43,$80,$80,$42,$58,$5A
1-F51 80 80 42
1-F54 58 5A
1-F56 24 80 23          .BYTE   $24,$80,$23,$26,$1B,$25,$22,$21
1-F59 26 1B 25
1-F5C 22 21

1-F5E 5B 20 5D          .BYTE   $5B,$20,$5D,$4E,$80,$4D,$3F,$81
1-F61 4E 80 4D
1-F64 3F 81
1-F66 52 80 45          .BYTE   $52,$80,$45,$59,$9F,$54,$57,$51
1-F69 59 9F 54
1-F6C 57 51

1-F6E 28 80 29          .BYTE   $28,$80,$29,$27,$9C,$40,$7D,$9D
1-F71 27 9C 40
1-F74 7D 9D
1-F76 46 4B 44          .BYTE   $46,$4B,$44,$80,$83,$47,$85,$41
1-F79 80 83 47
1-F7C 85 41

1-F7E 0C 0A 7B          .BYTE   $0C,$0A,$7B,$80,$80,$0B,$1E,$1F CONTROL
1-F81 80 80 0B
1-F84 1E 1F
1-F86 0F 80 10          .BYTE   $0F,$80,$10,$15,$9B,$09,$1C,$1D
1-F89 15 9B 09
1-F8C 1C 1D

1-F8E 16 80 03          .BYTE   $16,$80,$03,$80,$80,$02,$18,$1A
1-F91 80 80 02
1-F94 18 1A
1-F96 80 80 85          .BYTE   $80,$80,$85,$80,$1B,$80,$FD,$80
1-F99 80 1B 80
1-F9C FD 80
```

```
1F9E 00 20 60                .BYTE   $00,$20,$60,$0E,$80,$0D,$80,$81
1FA1 0E 80 0D
1FA4 80 81
1FA6 12 80 05                .BYTE   $12,$80,$05,$19,$9E,$14,$17,$11
1FA9 19 9E 14
1FAC 17 11

1FAE 80 80 80                .BYTE   $80,$80,$80,$80,$FE,$80,$7D,$FF
1FB1 80 FE 80
1FB4 7D FF
1FB6 06 08 04                .BYTE   $06,$08,$04,$80,$84,$07,$13,$01
1FB9 80 84 07
1FBC 13 01


1FBE AD 09 D2        PIRQ    LDA     KBCODE
1FC1 CD F2 02                CMP     CH1     TEST AGAINST LAST KEY PRESSED
1FC4 D0 05                   BNE     PIRQ3   IF NOT, GO PROCESS KEY
1FC6 AD F1 02                LDA     KEYDEL  IF KEY DELAY BYTE > 0
1FC9 D0 20                   BNE     PIRQ4   IGNORE KEY AS BOUNCE
1FCB AD 09 D2        PIRQ3   LDA     KBCODE  RESTORE AC
1FCE C9 9F                   CMP     #CNTL1  TEST CONTROL 1 (SSFLAG)
1FD0 D0 0A                   BNE     PIRQ1
1FD2 AD FF 02                LDA     SSFLAG
1FD5 49 FF                   EOR     #$FF
1FD7 8D FF 02                STA     SSFLAG
1FDA B0 0F                   BCS     PIRQ4   (UNCONDITIONAL) MAKE ^1 INVISIBLE
1FDC 8D FC 02        PIRQ1   STA     CH
1FDF 8D F2 02                STA     CH1
1FE2 A9 03                   LDA     #3
1FE4 8D F1 02                STA     KEYDEL  INITIALIZE KEY DELAY FOR DEBOUNCE
1FE7 A9 00                   LDA     #0      CLEAR COLOR SHIFT BYTE
1FE9 85 4D                   STA     ATRACT
1FEB A9 30           PIRQ4   LDA     #$30
1FED 8D 2B 02                STA     BRTIMR
1FF0 68              PIRQ2   PLA
1FF1 40                      RTI


1FF2 FF FF FF                .BYTE   $FF,$FF,$FF,$FF,$FF,$FF
1FF5 FF FF FF

1FF8                 CRNTPC  =*
                             *=$14
0014 00              KBDSPR  .BYTE   $FFF8-CRNTPC    DISPLC IS TOO LONG
                             .END
```

SYMBOL TABLE

ADDCOR	030E	ADRESS	0064	AFP	D800	ALLPOT	D208
ALDCAT	FE45	ANCONV	FE69	ANTIC	D400	APPEND	0001
APPMHI	000E	ASCCO1	F705	ATACHR	02FB	ATAINT	FEF6
ATAN	BE43	ATASCI	FEFE	ATRACT	004D	AUDC1	D201
AUDC2	D203	AUDC3	D205	AUDC4	D207	AUDCTL	D208
AUDF1	D200	AUDF2	D202	AUDF3	D204	AUDF4	D206
BADDLG	0086	BADMOD	0091	BELL	F90A	BELL1	F90C
BFENHI	0035	BFENLO	0034	BITCLR	FB12	BITCON	FAEB
BITGET	FB31	BITGET	FB25	BITMSK	006E	BITPUT	FB04
BITROL	FAFA	BITSET	FB06	BLIM	028A	BLKBDV	E471
BOOT?	0009	BOOTAD	0242	BOTSCR	02BF	BPTR	003D
BRKABT	0080	BRKKEY	0011	BS	F7E6	BS1	F80D
BSP	F805	BS3	F7F5	BSA	F7EC	BUFADR	0015
BUFCNT	006B	BUFRFL	0038	BUFRHI	0033	BUFRLO	0032
BUFSTR	006C	CASBUF	03FD	CASETV	E440	CASFLG	030F
CASINI	0002	CASORG	EF41	CASSBT	004B	CASSET	0043
CAUX1	023C	CAUX2	023D	CBAUDH	02EF	CBAUDL	02EE
CCOMND	0238	CDEVIC	023A	CDTMA1	0226	CDTMA2	0228
CDTMF3	022A	CDTMF4	022C	CDTMF5	022E	CDTMV1	0218
CDTMV2	021A	CDTMV3	021C	CDTMV4	021E	CDTMV5	0220
CH	02FC	CH1	02F2	CHACT	02F3	CHACTL	D401
CHAR	02FA	CHBAS	02F4	CHBASE	D409	CHKERR	008F
CHKCNT	0038	CHKSUM	0031	CHRORG	E000	CIOCHR	002F
CIOINV	E46E	CIOORG	E4A6	CIOV	E456	CIX	00F2
CKEY	004A	CLICK	FCDB	CLICK1	FCDA	CLOSE	000C
CLRCHN	007D	CLRLI1	FBA6	CLRLIN	FB9B	CLRSC2	F7BF
CLRSCR	F7CE	CLRSCR	F7B9	CLRTAB	FB32	CLRTBB	F430
CNTL	009F	CNTRLB	FEC6	COLAC	0072	COLBK	D01A
COLCR	FCE4	COLCR1	FCE6	COLCR2	FCF0	COLCR8	0055
COLDST	0244	COLDBV	E477	COLINC	007A	COLOR0	02C4
COLOR1	02C5	COLOR2	02C6	COLOR3	02C7	COLOR4	02C8
COLPF0	D016	COLPF1	D017	COLPF2	D018	COLPF3	D019
COLPM0	D012	COLPM1	D013	COLPM2	D014	COLPM3	D015
COLRSH	004F	COLRTB	FEC1	COLUMN	FE8D	COMRE1	F7A7
CONRET	F789	CONSOL	D01F	CONVR1	F97E	CONVR2	F988
CONVR3	F98F	CONVR4	F99C	CONVR5	F9A6	CONVR6	F9A7
CONVRT	F947	COS	BD73	COUNTR	007E	CR	009B
CRETRY	0036	CRITIC	0042	CRLOOP	F5E7	CRNTPC	FFF8
CRSINH	02F0	CRSRDN	F78C	CRSRL1	F7A3	CRSRLF	F799
CRSROR	008D	CRSRRT	F7AA	CRSRUP	F77F	CSOPIV	E47D
CSTAT	0288	CTIA	D000	DAUX1	030A	DAUX2	030B
DBDDEC	F913	DBDEC	F91F	DBSECT	0241	DBSUB	F921
DBSUB1	F934	DBUFHI	0305	DBUFLO	0304	DBYTHI	0309
DBYTLO	0308	DCB	0300	DCOMND	0302	DDEVIC	0300
DEGFLG	00FB	DEGON	0006	DELCH1	F870	DELCH2	F896
DELCUR	F86D	DELETE	0021	DELLI1	F8DD	DELLI2	F8F8
DELLIA	F8D7	DELLIB	F8DB	DELLIN	F8D4	DELTAC	0077
DELTAR	0076	DELTI1	FC82	DELTI2	FC89	DELTI3	FC8C
DELTIA	FC68	DELTIB	FC70	DELTIM	FC73	DERROR	0090
DFLAGS	0240	DHLINE	FE81	DIGRT	00F1	DINDEX	0057
DIRECT	0002	DISK	0044	DISKIV	E450	DISPLA	E410
DISPLY	0053	DIV2TB	FEA5	DLISTH	D403	DLISTL	D402
DMACTL	D400	DMASK	02A0	DMASKT	FEB1	DNACK	008B
DOBUI	FC1A	DOBUI1A	FC29	DOBUF1	FC12	DOBUF2	FC39
DOBUF3	FC51	DOBUF4	FC55	DOBUFC	FC00	DOCR	FA34
DOCR1	FA00	DOCR1A	FA29	DOCR1B	FA14	DOCR2	FA3D
DOCR2A	FA4A	DOCR2B	FA4D	DOCR4B	FA61	DOCRWB	FA30
DOLCO1	FBE5	DOLCO2	FBF8	DOLCOL	FBDD	DOPEN	F3F6
DOPEN1	F460	DOPEN2	F4D5	DOPEN3	F4FA	DOPEN4	F51C

DOPEN5	F524	DOPEN7	F588	DOPEN8	F438	DOPEN9	F577
DOPENA	E457	DOSINI	000C	DOS8	F6A0	DOSVEC	000A
DRAW	FCFC	DRAW1	FD37	DRAW10	FE42	DRAW11	FD99
DRAW2	FD5C	DRAW3	FD83	DRAW3A	FD62	DRAW4A	FD90
DRAW5	FDA4	DRAW5A	FDB2	DRAW6	FDBE	DRAW6A	FDD7
DRAW6B	FDEB	DRAW7	FDF1	DRAW8	FDF6	DRAW8A	FE0A
DRAW8B	FE27	DRAW8C	FE13	DRAW9	FE30	DRAWA	FD0C
DRAWB	FD08	DRAWLN	0011	DRETRY	0037	DRKMSK	004E
DSKINT	0018	DSKINV	E453	DSKORG	EDEA	DSKTIM	0246
DSKUTL	001A	DSPFLG	02FE	DSTAT	004C	DSTATS	0303
DUINIO	0306	DUNIT	0301	DUNUSE	0307	DVSTAT	02EA
EDITOR	E400	EDITRV	E400	EEXP	00ED	EGETC1	F650
EGETC2	F66E	EGETC3	F67C	EGETC5	F691	EGETC6	F66B
EGETC7	F680	EGETCH	F63E	ENDPT	0074	EOFERR	008B
EINPEN	F3FC	EOUTC5	F6BE	EOUTC6	F6B5	EOUTCH	F6A4
ERANGE	FA88	ERETN	F6BB	ERRFLG	023F	ESCAPE	F779
ESCFLG	02A2	ESIGN	00EF	EXP	DDC0	EXP10	DDCC
EXTEN1	FB7E	EXTEN3	FB8B	EXTEN4	FB92	EXTEND	FB7B
FADD	DA66	FASC	DBE6	FCHRFL	00F0	FDIV	DB2B
FILDAT	003F	FILDAT	02FD	FILFLG	02B7	FILLIN	0012
FLDOR	DD8D	FLDOR	DD89	FLD1P	DD9C	FLD1R	DD9B
FLPTR	00FC	FMOVE	DDB6	FMSZPO	0043	FMUL	DADB
FNCNUT	0092	FORMAT	0022	FPI	D9D2	FPREC	0006
FPSCR	05E6	FPSCR1	05EC	FPTR2	00FE	FRO	00D4
FR1	00E0	FR2	00E6	FRE	00DA	FREQ	0040
FRNADR	0068	FRMERR	008C	FRX	00EC	FSCR	05E6
FSCR1	05EC	FSTOP	DDAB	FSTOR	DDA7	FSUB	DA60
FTYPE	003E	GETCH	F593	GETCHR	0007	GETOUT	F749
GETPLT	F5A2	GETREC	0005	GLBABS	02E0	GPRIOR	026F
GRACTL	D01D	GRAFM	D011	GRAFPO	D00D	GRAFP1	D00E
GRAFP2	D00F	GRAFP3	D010	HATABS	031A	HITCLR	D01E
HOLD1	0051	HOLD2	029F	HOLD3	029D	HOLD4	02BC
HOLDS	02BD	HOLDCH	007C	HOME	F7D6	HPOSMO	D004
HPOSM1	D005	HPOSM2	D006	HPOSM3	D007	HPOSPO	D000
HPOSP1	D001	HPOSP2	D002	HPOSP3	D003	HSCROL	D404
ICAX1	034A	ICAX1Z	002A	ICAX2	0348	ICAX2Z	002B
ICBAH	0345	ICBAHZ	0025	ICBAL	0344	ICBALZ	0024
ICBLH	0349	ICBLHZ	0029	ICBLL	0348	ICBLLZ	0028
ICCOM	0342	ICCOMT	0017	ICCOMZ	0022	ICDNO	0341
ICDNOZ	0021	ICHID	0340	ICHIDZ	0020	ICIDNO	002E
ICPTH	0347	ICPTHZ	0027	ICPTL	0346	ICPTLZ	0026
ICSPR	034C	ICSPRZ	002C	ICSTA	0343	ICSTAZ	0023
IIP	D9AA	INATAH	FB4A	INATAC	FB32	INBUFF	00F3
INCRA	F9FB	INCRB1	FA77	INCRB2	F9E4	INCRB3	F9F7
INCRBA	F9DC	INCRSB	F9D4	INCRSC	F9DA	INCRSR	F9DB
INSCH1	F852	INSCH3	F85E	INSCH4	F844	INSCH5	F86A
INSCH6	F861	INSCHR	F837	INSCLR	0020	INSDAT	007D
INSLI1	F8C6	INSLI2	F8CE	INSLIA	F8A5	INSLIN	F8A4
INTABP	0200	INTATA	FEFA	INTEMP	022D	INTINV	E46B
INTURQ	E6D5	INTZBB	0010	INVFLG	02B6	IOCB	0340
IOCBA3	0020	IOCBBZ	0010	IOCFRE	00FF	IRQEN	D20E
IROST	D20E	JSRIND	F6A1	K1	F729	K2	F734
K3	F73F	K4	F776	K5	F768	K6	F74D
K7	F745	K8	F773	KBCODE	D209	KBD	004B
KBDIND	E420	KBDORQ	F3E4	KBDSPR	0014	KEYBDV	E420
KEYDEL	02F1	KGETC1	F71E	KGETC2	F6DD	KGETC3	F6FE
KGETCH	F6E2	LBFEND	05FF	LBPR1	057E	LBPR2	057F
LBUFF	0580	LEDGE	0002	LFRTCM	F7A5	LINBUF	0247
LINZBB	0000	LMARGN	0052	LO1GET	FB22	LO2GET	FB23
LOCREL	0023	LOG	DECD	LOG10	DED1	LOGCOL	0063

LOGCT	FB20	LOGMAP	02B2	LPENH	0234	LPENV	0235
M0PF	D000	M0PL	D008	M1PF	D001	M1PL	D009
M2PF	D002	M2PL	D00A	M3PF	D003	M3PL	D00B
MAXCTB	FEB9	MAXDEV	0021	MAXIOC	0080	MEMLO	02E7
MLTIMP	02E5	MLTTMP	0066	MODEM	004D	MONORO	F0E3
MOVLI	FB5B	MOVLI2	FB7A	MOVLIN	FB4E	MXDMDE	FE5D
MXDMOD	0010	NEWCOL	0061	NEWROW	0060	NMIEN	D40E
NMITMSK	D40F	NMIST	D40F	NOCKSM	003C	NOFUNC	F63D
NONDEV	F4AB	NONDEV	0082	NOROWS	FE99	NOSCR1	FA32
NOSCR0	FA2C	NOTB	F48B	NOTE	0026	NOTMXD	F4F5
NOTUPN	0085	NSIGN	00EE	NUMDLE	FE51	NVALID	0084
OFFCRT	FAE4	OLDADR	005E	OLDCHR	005D	OLDCOL	005B
OLDROW	005A	OPEN	0003	OPNCOM	F404	OPNERR	F453
OPNIN	0004	OPNINO	000C	OPNOT	0008	OPNTMP	0066
OUTCH	F5B7	OUTCH2	F5FF	OUTCHA	F5BD	OUTCHB	F5D7
OUTCHR	F5CA	OUTPLT	F5E0	OVRRUN	008E	POPF	D004
POPL	D00C	P1PF	D005	P1PL	D00D	P2PF	D006
P2PL	D00E	P3PF	D007	P3PL	D00F	PACTL	D302
PADDL0	0270	PADDL1	0271	PADDL2	0272	PADDL3	0273
PADDL4	0274	PADDL5	0275	PADDL6	0276	PADDL7	0277
PACTB	FE75	PBCTL	D303	PBPNT	001D	PBUFSZ	001E
PCOLR0	02C0	PCOLR1	02C1	PCOLR2	02C2	PCOLR3	02C3
PENH	D40C	PENV	D40D	PHACR1	FC9F	PHACR8	FC9D
PIA	D300	PIRQ	FFBE	PIRQ1	FFDC	PIRQ2	FFFO
PIRQ3	FFCB	PIRQ4	FFEB	PLACR1	FCAA	PLACR8	FCA8
PLYARG	05E0	PLYEVL	DD40	PMBASE	D407	POINT	0025
POKEY	D200	POKMSK	0010	PORTA	D300	PORTB	D301
POT0	D200	POT1	D201	POT2	D202	POT3	D203
POT4	D204	POT5	D205	POT6	D206	POT7	D207
POTGO	D20B	PRINTR	0050	PRINTV	E430	PRIOR	D01B
PRNBUF	03C0	PRNORO	EE7B	PRVOPN	0081	PTEMP	001F
PTINOT	001C	PTRIG0	027C	PTRIG1	027D	PTRIG2	027E
PTRIG3	027F	PTRIG4	0280	PTRIG5	0281	PTRIG6	0282
PTRIG7	0283	PUTCHR	000B	PUTMSG	FCF3	PUTREC	0009
PWRONA	F3E4	RADFLG	00FB	RADON	0000	RAMLO	0004
RANSIZ	02E4	RAMTOP	006A	RANDOM	D20A	RANGE	FA96
RANGE1	FAB7	RANGE2	FABB	RANGE3	FA9E	RBLOKV	E47A
RDONLY	0087	RECVDN	0039	REDGE	0027	RENAME	0020
RETUR1	F634	RETUR3	FAE1	RETURN	F621	RMARGN	0053
RNGER1	FAD8	RNGER2	FAD6	RNGERR	FAD1	RNGOK	FAC4
ROWAC	0070	ROWCRS	0054	ROWINC	0079	RTCLOK	0012
SAVADR	0068	SAVIO	0316	SAVMSC	0058	SCREDT	0045
SCRENV	E410	SCRFLG	02BB	SCRMEM	0093	SCROL1	FBB7
SCROL2	FBCA	SCROLL	FBAC	SDLSTH	0231	SDLSTL	0230
SDMCTL	022F	SENDEV	E468	SERIN	D20D	SEROUT	D20D
SETVAR	F82D	SETVBV	E45C	SHFAMT	006F	SHFLOK	02BE
SHIFT1	F5B1	SHIFT2	F610	SHIFTD	F5AA	SHIFTU	F608
SIN	BD81	SIOINV	E465	SIOORG	E944	SIOV	E459
SIZEM	D00C	SIZEP0	D008	SIZEP1	D009	SIZEP2	D00A
SIZEP3	D00B	SKCTL	D20F	SKRES	D20A	SKSTAT	D20F
SOUNDR	0041	SPECIL	000E	SQR	BEB1	SRTIMR	022B
SSFLAG	02FF	SSKCTL	0232	STACKP	0318	STATIB	000D
STATUS	0030	STICK0	0278	STICK1	0279	STICK2	027A
STICK3	027B	STIMER	D209	STORE	F917	STORE1	F91D
STRDEV	FC5C	STRERR	F942	STRIG0	0284	STRIG1	0285
STRIG2	0286	STRIG3	0287	STROK	F946	SUBEND	FA7A
SUBTMP	029E	SUCCES	0001	SWAP	FCB3	SWAP1	FCC2
SWAP3	FCD7	SWAPA	FCB9	SWPFLG	007B	SYSVBV	E45F
TAB	F810	TAB1	F823	TAB2	F82A	TABMAP	02A3
TEMP	023E	TEMP1	0312	TEMP2	0314	TEMP3	0315

TIMER1	030C	TIMER2	0310	TIMFLG	0317	TIMOUT	008A
TINDEX	0293	TMPCHR	0050	TMPCOL	0289	TMPLBT	02A1
TMPROW	02B8	TMPX1	029C	TOADR	0066	TRAMSZ	0006
TRIG0	D010	TRIG1	D011	TRIG2	D012	TRIG3	D013
TRNRCD	0089	TSTAT	0319	TSTCT1	FC8F	TSTCT2	FC9C
TSTCTL	FC8D	TSTDAT	0007	TXTCOL	0291	TXTMSC	0294
TXTROW	0296	TXTROW.	0290	UNLOCK	0024	UPDNCM	F787
VBAREA	0480	VBREAK	0206	VBWAIT	E496	VCOUNT	D40B
VCTABL	E480	VDELAY	D01C	VDSLST	0200	VECTBL	E400
VTHORG	0216	VINTER	0204	VKEYBD	0208	VPRCED	0202
VSCROL	D405	VSERIN	020A	VSEROC	020E	VSEROR	020C
VTIMR1	0210	VTIMR2	0212	VTIMR4	0214	VVBLKD	0224
VVBLKI	0222	WARMST	0008	WARMSV	E474	WMODE	0289
WSYNC	D40A	XITVBV	E462	XMTDON	003A		
ZIOCB	0020	ZTEMP1	00F5	ZTEMP3	00F9	ZTEMP4	00F7